Imperial College London

Department of Electrical and Electronic Engineering

Wireless Coded Caching and Computing

Mohammad Mohammadi Amiri

September 2019

Supervised by Dr. Deniz Gündüz

A thesis submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical and Electronic Engineering of Imperial College London
and the Diploma of Imperial College London

Abstract

The ever-increasing demands for both content and computation over wireless networks require moving some of the core processing capabilities close to the network edge. This dissertation considers coded caching and delivery which makes content delivery more efficient by moving content to the edge, as well as distributed learning at the network edge that can bring network intelligence close to edge devices and speed up large-scale data collection and learning problems.

First proactive content caching is studied, where a server with a library of files transmit contents to the users simultaneously. Each user requests a single file from the library and stores content in its cache with limited size proactively, before revealing the demands. The performance is first analysed in terms of the minimum number of bits transmitted by the server to satisfy the users' demands over an error-free shared link. Then, by considering various models for the shared link, physical layer aspect of fulfilling users' demands is studied. The highest achievable rate of each file in the library is characterized, upper and lower bounds on the transmit power are derived, and finally a caching system with delivering files to the users at different rates is investigated, and the rate tuples at which the requested contents can be delivered to the users is characterized.

Next machine learning (ML) at the wireless edge is studied. First, by considering scheduling of computation tasks across multiple computational nodes to compute an arbitrary function, upper and lower bounds on the minimum average completion time are developed. Then collaborative ML at the wireless edge is studied, where power and bandwidth-limited wireless devices with local datasets carry out a learning task with the help of a remote parameter server (PS). Digital and analog approaches are introduced for transmission from the users to the PS over a shared wireless medium.

Declaration of Originality

I hereby certify that this thesis is the result of my own work under the guidance of my Ph.D. advisor, Dr. Deniz Gündüz. Any ideas or quotations from the work of other people are appropriately referenced.

Imperial College London
London, United Kingdom
16 September 2019

Mohammad Mohammadi Amiri

Copyright Declaration

Acknowledgements

First and foremost I would like to express my sincere gratitude to my advisor Dr. Deniz Gündüz for his dedicated help, advice, immense knowledge, inspiration, motivation, encouragement and continuous support, throughout my Ph.D. His enthusiasm, integral view on research and his mission for providing high-quality work, has made a deep impression on me. I would like to thank him for encouraging my research and for providing me with the best environment to grow as a research scientist. His guidance helped me in all the time of research and writing of this thesis. Without his precious support it would not be possible to conduct this research. I could not have imagined having a better advisor and mentor for my Ph.D. study. I am thankful for the excellent example he has provided me as a successful advisor.

I would also like to thank my examiners: Prof. Kai-Kit Wong and Dr. Bruno Clerckx, for reading my thesis carefully, and their insightful comments and useful discussion during the viva examination, which incented me to widen my research from various perspectives.

I am very thankful to the Intelligent Systems and Networks group's administrators, Patrick and Joan, who were always ready to give their timely help whenever required. I would also like to thank all the members of staff at Imperial College London, in particular Electrical and Electronic Engineering department, who helped me during my Ph.D. study.

I thank my fellow lab mates for their support, and for all the fun we have had in the last four years. I am thankful to them for making my experience in the Information Processing and Communications Lab exciting and fun, and for sharing a great relationship as compassionate friends. I will always cherish the warmth shown by them.

Last but not least, I would like to thank my family for their unconditional and continuous support, warm love, constant inspiration and encouragement, without which none of this would have been possible. Words can not express how grateful I am to my family. I owe them everything I have reached so far in my life, and this thesis is dedicated to them.

London, United Kingdom
16 September 2019

Mohammad

To my family,

my greatest source of inspiration.

Contents

List of Figures

List of Tables

List of Acronyms

A-DSGD	Analog DSGD
AMP	Approximate message passing
AWGN	Additive white Gaussian noise
BC	Broadcast channel
CDF	Cumulative distribution function
CDPC	Joint cache and dirty paper coding
CS	Cyclic scheduling
CSC	Joint cache and superposition coding
CTDC	Joint cache and time-division coding
D-DSGD	Digital DSGD
DGD	Distributed gradient descent
DSGD	Distributed stochastic gradient descent
EPA	Equal power allocation
GBC	Group-based centralized
GBD	Group-based decentralized
GD	Gradient descent
IoT	Internet of things
i.i.d.	Independent and identically distributed
MAC	Multiple access channel
MDS	Maximum-distance separable
ML	Machine learning
PC	Polynomially coded
PCC	Pair-wise coded caching
PCMM	Polynomially coded multi-message
PDF	Probability density function
PS	Parameter serve
RA	Random assignment
SCC	Successive cache-channel coding

SGD	Stochastic gradient descent
SS	Staircase scheduling
TO	Task ordering
TS	Time slot
UPA	Unequal power allocation
XOR	Exclusive or
XR	Extended reality

Notation

Throughout this dissertation we will use the following notation. We denote sets of real and integer values by $\mathbb{R}$ and $\mathbb{Z}$, respectively. Notations $\lfloor x \rfloor$ and $\lceil x \rceil$ represent the floor and ceiling functions, respectively. For $i, j \in \mathbb{Z}$, $[i:j]$ denotes the set $\{i, i+1, ..., j\}$, and, for $x \in \mathbb{R}$, we let $[x] \triangleq \{1, 2, ..., \lceil x \rceil\}$, and $(x)^+ \triangleq \max\{x, 0\}$. We denote the sequence $X_i, X_{i+1}, \dots, X_{j-1}, X_j$ shortly by $X_{[i:j]}$. $\binom{j}{i}$ represents the binomial coefficient, and $\mathbf{A}(i,j)$ returns (i,j)-th entry of matrix $\mathbf{A}$. For two sets $\mathcal{Q}$ and $\mathcal{P}$, $\mathcal{Q} \backslash \mathcal{P}$ is the set of elements in $\mathcal{Q}$ that do not belong to $\mathcal{P}$, and for $q \notin \mathcal{Q}$, we define $\{\mathcal{Q}, q\} \triangleq \mathcal{Q} \bigcup \{q\}$. We let $\mathcal{N}\left(0, \sigma^2\right)$ represent a zero-mean normal distribution with variance σ^2, $\mathcal{U}\left(a, b\right)$ denote a uniform distributed over $[a, b]$. Notation $|\cdot|$ returns cardinality of a set, or the length of a file. Also, $\oplus$ refers to bitwise XOR operation, while $\bar{\oplus}$ represents bitwise XOR operation where the arguments are first zero-padded to have the same length as the longest argument. Finally, $\|\boldsymbol{x}\|_2$ returns l_2 norm of vector $\boldsymbol{x}$.

Chapter 1

Introduction

Emerging technologies along with ever-increasing wireless devices with their demands have brought new challenges on the network core. The backhaul network experiences a significantly high network load at the times of heavy demand. Also, efficiently processing the massive amount of data, growing explosively, imposes a huge burden on the network core. To alleviate the load on the network, it is vital to move some of the core processing capabilities to the network edge in order to deal with the growing demand for content, as well as computation over wireless networks.

In this dissertation, we focus on developing tools to exploit "edge processing" capabilities in wireless networks. We study distributed frameworks and develop techniques to facilitate communication and computation, which are the two main core components of many emerging technologies. In particular, we develop tools to convert distributed cache memories into valuable bandwidth through coded delivery. Similarly, we utilize computational capabilities distributed across multiple devices in a network to speed up computation tasks involving massive datasets, particularly for machine learning applications. In both parts of the thesis, we exploit ideas from communication and coding theory to utilize distributed network resources in the most efficient manner.

In the following sections we will briefly overview the problems considered in these two components of the thesis, provide a brief summary of the most relevant literature, and the objectives dealt with.

1.1 Wireless Coded Caching

The ever-increasing mobile data traffic is imposing a great challenge on the current network architectures. The growing demand has been typically addressed by increasing the achievable data rates; however, there has been a recent revival of interest in *content caching*, particularly focusing on wireless networks. This interest stems from a very practical problem: exponential growth in mobile traffic cannot be matched by the increase in the spectral efficiency of wireless networks. This, in turn, leads to congestion in the radio access as well as the backhaul links, and increased delay and outages for users, particularly during peak traffic periods, whereas the resources are often underutilized during off-peak periods. *Proactively caching* popular contents at the network edge during off-peak hours has been recently proposed as a potential remedy for this problem (see [1–5], and references therein). Proactive caching shifts traffic from peak to off-peak hours, reduces latency for users, and potentially provides energy savings.

Caching in this model consists of two distinct phases: In the first phase, which takes place during off-peak periods, i.e., when the network is not congested, caches at user terminals are filled by the server. This first phase is called the *placement phase.* The only constraint on the data transmitted and stored in a user cache in this phase is the cache size. However, due to the “proactive” nature of the cache placement, it is carried out without the knowledge of the particular user requests. A shared communication channel is considered to be available from the server to all the users during the peak traffic period. Once the user demands are revealed in this period, the server broadcasts additional information over the common error-free channel in order to satisfy all the user requests simultaneously. This constitutes the *delivery phase.* Since the delivery phase takes place during peak traffic period, the goal is to minimize the rate of transmission over the shared link, called the *delivery rate*, by exploiting the contents that are available at the caches.

1.1.1 Literature Review

Here we present the most fundamental and relevant papers studying content caching and delivery, which is by no means an exhaustive literature survey.

Over the past decade, research on caching has mainly focused on the placement phase; the goal has been to decide which contents to cache, typically at a server that serves many users, by anticipating future demands based on the history (see [6–9], and references therein). This is an uncoded caching approach, where parts of popular contents are stored in the local cache memories in an uncoded manner, and once the user requests are revealed, remaining parts are delivered by the server over the shared link. The corresponding gain relative to not having a cache is called *local caching gain*, and depends on the local cache size. On the other hand, it has been shown in [3] that *coded caching* provides a *global caching gain*, where users benefit not only from their own local cache, but also from the available cache memory across the network. Coded caching provides a novel method to mitigate network congestion during peak traffic hours by creating and exploiting coded multicasting opportunities across users.

In a *centralized coded caching* scheme, it is assumed that during the placement phase the central server knows both the number and the identity of users participating in the delivery phase, and carefully places contents in the user caches during off-peak hours. A novel centralized coded caching scheme for a network of K users requesting N popular files of the same size is proposed in [3]. For a caching factor $r \in [0 : K]$, with the scheme in [3], each file is split into $\binom{K}{r}$ equal-length subfiles, and each set of r users store a distinct subfile of each file, resulting in a normalized cache size $M = N\binom{K-1}{r-1}/\binom{K}{r} = rN/K$ at each user. After the user demands are revealed, for any demand combination, a coded packet is delivered to each set of $r+1$ users, from which each user in that set can obtain its missing subfile thanks to its cached content. In total, for any demand combination, $\binom{K}{r+1}$ coded packets are delivered, each of normalized size $1/\binom{K}{r}$. Thus, this scheme delivers the same number of coded packets in the delivery phase regardless of any specific demand combination. However, in practice files have different popularities, and a fraction of the files might be highly popular and requested by more than a user. The scheme proposed in [3], which has been shown to be optimal for the worst case user demands when $K \leq N$ and the cache placement phase is uncoded [10], has been improved by taking into account the repeated demands across the users [11–20]. Authors in [11] consider an alternative coded caching scheme, which was originally proposed in [3] for two users, and show that it is optimal when the number of users, K, is not less than the number of files in the library, i.e., $N \leq K$, and the normalized cache size M at each user satisfies $M \leq 1/K$, i.e., a relatively small cache size. For $N \leq K$, the delivery rate is further improved for various special settings including $N = 2$ files [12, 13], cache capacities $M = (N-1)/K$ [14], $M = N/K$ [15] and $1/K \leq M \leq N/K$ [16], and other special cases investigated in [17, 18]. Also, [19] proposes a coded caching scheme over a finite field of order 2^2 approaching the performance of that of introduced in [17]. An achievable scheme along with a lower bound proving its optimality for uncoded cache placement for whole range of N, K and M is proposed in [20].

Theoretical lower bounds on the delivery rate have been derived to characterize the optimal performance of a caching system [3, 21–25]. In general, the minimum delivery rate for coded caching remains an open problem even in the symmetric setting considered in the aforementioned previous works.

The scheme of [3] has been extended to a *decentralized caching* scenario [26], in which neither identity nor the number of the users requesting files during the delivery phase are not known in advance to perform the placement in coordination across caches. The decentralized coded caching scheme proposed in [26] introduces a random cache placement phase, where a randomly selected portion of each file is cached by each user, followed by a coded delivery sending common packets to any subset of users. Similarly to the centralized caching scheme introduced in [3], the decentralized caching scheme in [26] has been improved for various different cases in [12, 15, 27].

The information-theoretic coded caching schemes introduced in [3, 26] have received significant attention, and have been extended to a variety of scenarios considering a one-to-many communications, where placement and delivery phases are jointly designed to improve the performance. A multi-layer caching scheme is proposed in [28–30], providing coded multicasting opportunities within each layer, as well as across multiple layers. Coded caching has also been employed when files have distinct sizes [31]. The authors in [32–34] utilize multicasting opportunities in caching systems with nonidentical cache capacities across users. Coded caching gain has also been exploited for scenarios when the files in the library have different popularities across users [5, 35–37], and when the files in the library are correlated, and users require different levels of reconstruction distortion [38–42]. An online caching approach is introduced in [43, 44], where the set of popular files are updated. As opposed to the caching model introduced in [3,26], authors in [45–49] have designed coded caching techniques when the files are with finite size. Also, correlation-aware coded caching schemes have been introduced in [50–54] when the files in the library are correlated.

Coded caching technique has also been deployed in various applications. Device-to-device caching [4,55] studies the case when users communicate with other to collectively satisfy the demands along with the cached contents. Also, femtocaching [56,57] considers caching popular files in intermediate nodes between the server and the users, referred to as *helpers*. The requested files not available at the helpers are transmitted by the server at a higher cost, while the helpers serve the users with their cached contents. Coded caching gain has further been exploited in *combination networks*, where the server communicates with users, each equipped with a separate cache, through intermediate relay nodes [58–63]. In addition, coded caching technique has been investigated in cellular networks with multiple transmitters [64–70]. Recently, coded caching has been generalized to the scenario when the users are grouped, and each group of users share the same cache [71].

In contrast to the setting introduced in [3], the channel from the server to the users in the delivery phase is modelled as a noisy broadcast channel in [72–95]. The works in [72–85] study multi-antenna caching designing beamforming to maximize the throughput, interplay between caching gain and the amount of feedback on channel state, and coded caching over noisy broadcast channels (BCs) in high power regime. Also, coded caching techniques have been designed for various interference channels, considering the availability of caches at the transmitters, receivers, or both sides [86–89]. In [91], a centralized caching is considered while the delivery phase takes place over a packet-erasure BC. The capacity-memory trade-off is investigated in this setting assuming that only the weak users have caches, which requires knowledge about the channel qualities in the delivery phase in advance. A degraded BC is considered in [92], where the placement phase is designed in a centralized manner with the full knowledge of the channel during the delivery phase in order to maximize the common rate of each file in the library.

1.1.2 Objectives

In this dissertation, we first aim to improve the delivery rate of the conventional centralized caching setting studied in [3] as well as the decentralized one considered in [26] for some special settings. We then relax the assumption of identical cache capacities across users, since in practice users access content through diverse devices, typically with very different storage capacities. We study decentralized caching for users with distinct cache capacities, and develop upper and lower bounds on the delivery rate. We then aim to investigate the benefits of caching by considering the physical layer aspect of fulfilling users' demands. To this end, we study various noisy models for the shared link from the server to the users. We consider a memoryless packet erasure BC to model the channel from the server to the users in the delivery phase. This models a packetized communication system, where each packet is separately channel coded against errors at the physical layer, so that a packet either arrives at the receiver correctly, or is lost. Communication over the Internet is usually modeled as a packet erasure channel. Assuming equal-rate files in the library, our goal is to maximize the achievable rate of the files as a function of the cache size. Next, we study a Gaussian BC from the server to the users in the delivery phase. We study the benefits of proactive caching in reducing the transmit power, assuming that the noiseless cache placement phase is carried out without the knowledge of channel conditions during the delivery phase. We also consider a Gaussian BC model for the transmission in the delivery phase, and allow each user to request the files at a different quality, and equivalently, at a different rate. Our goal is to characterize the rate tuples at which the requested contents can be delivered to the users.

1.2 Distributed Machine Learning

Many emerging technologies involve massive amounts of data collection, and collaborative intelligence that can process and make sense of this data. Internet of things (IoT), autonomous driving, or extended reality (XR) technologies are prime examples, where data from sensors must be continuously collected, communicated, and processed to make inferences about the state of a system, or predictions about its future states. Many specialized machine learning (ML) algorithms are being developed tailored for various types of sensor data; however, the current trend focuses on training a powerful learning algorithm, often a neural network, on a massive dataset. It is essential to distribute the task of training across multiple devices, as exploiting a single device for large-scale ML problems is prohibitively slow. Distributed ML techniques typically assume the availability of the data at a server, which distributes the data across different devices, and each device processes its local data and returns the result to the server. While this inherently assumes the availability of data at a central processor, in the case of wireless edge devices, transmitting the collected data to a central processor in a reliable manner may be too costly in terms of energy and bandwidth, and undesirable due to privacy concerns. Communication is typically more costly compared to processing; therefore, a much more desirable and practically viable alternative is to develop distributed ML techniques that can exploit the local processing capabilities of edge devices, requiring limited communications (see [96] for a survey of applications of edge intelligence and existing approaches to enable it). Here we consider a distributed ML network, where distributed processors with local data samples and connected to a central parameter server (PS), jointly train a learning model.

1.2.1 Literature Review

Extensive efforts have been made in recent years to speed up large-scale distributed learning. Research in this direction can be categorized into two: those reducing the *computation time* at each worker, and those reducing the *communication load* between the workers and the PS.

The techniques aiming to reduce the computation time can be further categorized as coded and uncoded computation techniques, which are designed to mitigate the overall performance degradation caused by slow workers, referred to as *stragglers*. For this purpose, coded computation techniques, inspired by erasure codes against packet losses, have been proposed recently [97–102]. With coded computation, computations from only a subset of non-straggling workers are sufficient to complete the computation task, thanks to redundant computations performed by the faster workers. In [97] the authors employ a maximum-distance separable (MDS) code-inspired distributed computation scheme in a distributed matrix-vector multiplication problem. A more general distributed gradient descent (DGD) problem is considered in [98], where labeled dataset is distributed across workers, each evaluating the gradient on its own partition. Various coding schemes have been introduced in [98–102], that assign redundant computations to workers to attain tolerance against stragglers. Coded distributed computation has also been studied for matrix-matrix multiplication, where the labeled data is coded before being delivered to workers [103–105], and for distributed computing of a polynomial function [106]. Also, for a linear regression problem, a polynomially coded approach is proposed in [107], where the data is encoded and distributed across the workers to compute the gradient of the loss function.

Most existing coded computation techniques are designed to tolerate persistent stragglers, and discard computations performed by stragglers. However, in practice we often encounter *non-persistent stragglers*, which, despite being slower, complete a significant portion of the assigned tasks by the time faster workers complete all their tasks [108]. Recently, there have been efforts to exploit the computations carried out by non-persistent stragglers at the expense of increasing the communication load from the workers to the PS [108–112]. Techniques studied in [108–111] are based on coding with associated encoding and decoding complexities, which require the availability and processing of all the data points at the PS. In [111] a linear regression problem is studied, and the scheme in [107] is extended by allowing each worker to communicate multiple computations sequentially, where the computations are carried out using coded data. The authors in [108] propose to split the computation tasks into multiple levels, and code each level using MDS coding. However, the coding scheme depends on the statistical behavior of the stragglers, which may not be possible to predict accurately in practice. Distributed matrix-vector multiplication is studied in [109]. It is shown that, by performing random coding across the dataset, the results can be obtained from a subset of all the tasks assigned to the workers with high probability, where each completes the assigned tasks sequentially. To execute the tasks which are linear functions of their arguments, e.g., matrix-vector multiplication, rateless codes are used in [110], requiring a large number of data points assigned to each worker to guarantee decodability of the target function at the PS.

While significant research efforts have been invested in designing coded computation [98–107] techniques, uncoded computing and communication are also shown to be effective in tackling stragglers and reducing the average computation time [112, 113]. Unlike coded computation, uncoded computing approach does not introduce any encoding and decoding delays and complexities; hence, can be particularly efficient for edge learning where the data is inherently distributed [114]. It also allows partial decoding, which can be exploited to reduce the communication load for distributed learning [114–116]. An uncoded computation approach is considered in [112], where the dataset is split into a limited number of mini-batches, and each worker is randomly assigned a mini-batch of data. This approach requires a large number of workers compared to the number of mini-batches to ensure that the master can recover all the data from the workers with high probability. The authors in [113] study dynamic computation allocation across the workers with feedback providing information about the workers' speeds. The proposed uncoded computation approach in this paper does not impose any constraint on the number of workers, and is designed without any prior knowledge or feedback on the computation and communication delays at the workers.

In many practical implementations, however, bandwidth of the communication channel from the workers to the PS turns out to be the main bottleneck for speeding up distributed learning [117, 118]. Therefore, reducing the communication requirements of distributed stochastic gradient descent (DSGD) is as important as reducing the average computation time. To reduce the communication load, three main approaches, namely *quantization*, *sparsification*, and *local updates*, and their various combinations have been considered in the literature. Quantization methods implement lossy compression of the gradient vectors by quantizing each of their entries to a finite-bit low precision value [115, 117, 119–125]. Sparsification reduces the communication time by transmitting only some values of the gradient vectors [118,126–132]. Sparsification can be considered as another way of lossy compression, but it is assumed that the chosen entries of the gradient vectors are transmitted reliably, e.g., at a very high resolution.

Another approach is to reduce the frequency of communication from the workers by allowing local parameter updates [114,118,133–136]. The above works ignore the physical aspects of the underlying communication channel, and focus on reducing the amount of data sent by each worker to the PS. On the other hand, the authors in [137–139] consider *ML at the wireless network edge* by assuming a wireless shared medium from the workers to the PS. Distributed learning in this scenario is attractive also due to privacy and personalization [140]. The authors in [137] study a distributed federated learning problem over a fading multiple access channel (MAC), where each entry of a gradient vector is scheduled for transmission depending on its corresponding channel condition. A wireless MAC with beamforming at the PS, which is equipped with multiple antennas, is considered in [138] for distributed federated learning, where the goal is to maximize the number of workers scheduled for transmission with an acceptable quality for the retrieved signal.

1.2.2 Objectives

While significant research efforts have been invested in designing coded computation [98–107] techniques, we consider computation of an arbitrary function over a dataset, and introduce a centralized scheduling strategy for uncoded distributed computation, where the tasks are assigned to the workers by the master. Each worker can compute a limited number of tasks, referred to as the *computation load.* Computations are carried out sequentially, and the result of each computation is sent to the master right after it is completed. Communication delay from the workers to the master is also taken into account. We assume that both the computation and communication delays are independent across the workers since they have different computation capabilities with various dynamic behaviours of processing speed, and they communicate with the PS over different links experiencing distinct mediums, but the delays be correlated for different tasks carried out at the same worker. We highlight here that the reason behind assuming the independence of the computation delay across the worker The computation is assumed to be completed when the master receives sufficient number of distinct computations, referred to as the *computation target.* Assuming that the computation and communication delays are random variables, our goal is to characterize the minimum *average completion time* as a function of the computation load and computation target.

In the scenario above while we take into account the amount of communication between the PS and the workers, the communication channel in between is assumed error-free[1]. This is the standard assumption in the distributed computing literature, which mainly focuses on the reduction of the communication load, and ignores the communication channel. In this dissertation, we consider DSGD over-the-air; that is, we consider a wireless shared medium from the workers to the PS, and treat each iteration of the DSGD algorithm as a distributed over-the-air computation problem. This can model *machine learning at the wireless network edge*, where the workers correspond to IoT devices or sensor nodes that collect their local data samples. Distributed learning in this scenario is attractive also due to privacy and personalization [140]. We will provide two distinct approaches for this wireless DSGD problem, based on *digital* and *analog* computation approaches, respectively. We will show that analog "over-the-air" computation can significantly speed up wireless DSGD, particularly in bandwidth-limited and low-power settings, typically experienced by wireless edge devices.

1.3 Outline and Contributions

The first four technical chapters of this dissertation, Chapters 2-5, present results on coded caching and delivery for various scenarios with different communication channel models to address the physical layer aspects of the transmission over the delivery phase, while Chapters 6 and 7 focus on distributed computing and machine learning, respectively. In the following, we outline the content and results of each chapter, as well as the corresponding publications.

[1]Noisy communication link in this framework can be considered by introducing additional delay leading to a reliable communication.

Chapter 2

In Chapter 2 we consider a caching system, in which the users are served in the delivery phase through an error-free shared link. We highlight the fundamental limits of a caching system for a simple error-free shared link from the server to the users in this chapter, and consider more sophisticated channel models in the following chapters. Assuming the same cache size at each user, we present a centralized coded caching technique providing an improved delivery rate for cache size $M = (N-1)/K$. We then extend the improvement of the proposed scheme to the cache size $M = N/K$, and employ the proposed caching scheme in the decentralized scenario, and after deriving the delivery rate, we show that it reduces the state-of-the-art decentralized delivery rate. We finally study a decentralized caching for users with distinct cache capacities. We provide upper and lower bounds on the optimum delivery rate. The results in this chapter have been published in:

- M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Coded caching for a large number of users," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Cambridge, UK, pp. 171-175, Sep. 2016,
- M. Mohammadi Amiri and D. Gündüz, "Improved delivery rate-cache size trade-off for centralized coded caching," in *Proc. IEEE Int'l Symp. on Inform. Theory and Its Applications (ISITA)*, Monterey, CA, USA, pp. 26-30, Oct.-Nov. 2016.
- M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Decentralized coded caching with distinct cache capacities," in *Proc. Asilomar Conf. on Signals, Systems and Computers*, Pacific Grove, CA, pp. 734-738, Nov. 2016.
- M. Mohammadi Amiri and D. Gündüz, "Fundamental limits of coded caching: Improved delivery rate-cache size trade-off," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 806-815, Feb. 2017.
- M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Decentralized caching and coded delivery with distinct cache capacities," *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 4657-4669, Nov. 2017.

Chapter 3

In Chapter 3 we study coded caching, where the delivery phase takes place over a memoryless packet erasure BC rather than an error-free shared link. The receivers in the system are grouped into two disjoint sets of weak and strong receivers. All the weak receivers are assumed to have statistically worse channels than the strong receivers, while the users in each set can have arbitrary erasure probabilities. To compensate for their worse channel quality, each weak receiver is equipped with a cache memory of equal size. Assuming equal-rate files in the library, we derive a trade-off between the size of the caches provided to the weak receivers and the rate of the files, for which any demand combination can be reliably satisfied over the erasure BC. We show that, when specified to the homogeneous scenario considered in [91], in which all the receivers in the same set (i.e., weak and strong receivers) have the same erasure probability, the proposed scheme outperforms the one in [91]. The results in this chapter have been published in:

- M. Mohammadi Amiri and D. Gündüz, "Cache-aided data delivery over erasure broadcast channels," in *Proc. IEEE Int'l Conf. on Commun. (ICC)*, Paris, France, pp. 1-6, May 2017.
- M. Mohammadi Amiri and D. Gündüz, "Cache-aided content delivery over erasure broadcast channels," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 370-381, Jan. 2018.

Chapter 4

With the aim of considering a more realistic channel model, in Chapter 4 we study caching with delivery phase over a Gaussian BC, which is typically adopted as the channel model for wireless communication. Assuming uniform popularity across the files, we first study the minimum required *average power* to serve all the users, averaged over all the user demand combinations. We allow the transmitter to change its power depending on the demand combination in order to minimize the average power consumption. We then consider the transmit power required to satisfy the worst-case demand combination, called the *peak power.* We provide upper bounds on the minimum average and peak power values as a function of the rate of the files in the library and the capacity of the user caches, for centralized cache placement. We then extend the proposed scheme by considering *decentralized* cache placement. The proposed delivery strategy employs superposition coding and power allocation, and the achievable transmit power for any demand combination is derived thanks to the degradedness of a Gaussian BC. We further derive lower bounds on the performance assuming uncoded cache placement. The results in this chapter have been partially published in:

- M. Mohammadi Amiri and D. Gündüz, "Decentralized caching and coded delivery over Gaussian broadcast channels," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Aachen, Germany, pp. 2785-2789, Jun. 2017.
- M. Mohammadi Amiri and D. Gündüz, "Caching and coded delivery over Gaussian broadcast channels for energy efficiency," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 8, pp. 1706-1720, Aug. 2018.

Chapter 5

In Chapter 5 we study a caching system by considering a Gaussian BC in the delivery phase. We allow the users to request the files at different rates. Each file in the library is coded into K layers, K being the number of users, ordered in increasing channel qualities, where user k receives layers 1 to k of its request, $k = 1, \dots, K$. We consider a centralized placement phase, and assume that the channel qualities of the users in the delivery phase are known in advance. By allowing users to have different cache capacities, we consider a total cache size in the network as a constraint, and optimize cache allocation across the users and different layers of the files. We aim at characterizing the rate tuples at which the requested contents can be delivered to the users as a function of the total cache size. The results in this chapter have been published/submitted for publication in:

- M. Mohammadi Amiri and D. Gündüz, "On the capacity region of a cache-aided Gaussian broadcast channel with multi-layer messages," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Vail, CO, USA, pp. 1909-1913, Jun. 2018.

- M. Mohammadi Amiri and D. Gündüz, "On the capacity region of a cache-aided Gaussian broadcast channel with multi-layer messages," *arXiv:1806.09894 [cs.IT]*, Jun. 2018.

Chapter 6

In Chapter 6 we study a distributed computation problem, where an arbitrary function is to be computed over a dataset through K users. We introduce a centralized scheduling strategy for uncoded distributed computation, where the tasks are assigned to the users by the PS. The computation of the function over the dataset is announced to be completed, if the PS receives a sufficient number of distinct computations from the users. The goal is to characterize the minimum value of the average completion time as a function of the size of local dataset at each user, as well as the required number of distinct computations at the PS for completion, where the randomness is due to the random computation and communication delays at the users. We develop upper and lower bounds on the optimum average completion time. The results in this chapter have been published/submitted for publication in:

- M. Mohammadi Amiri and D. Gündüz, "Computation scheduling for distributed machine learning with straggling workers," in *Proc. IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Brighton, UK, pp. 8177-8181, May 2019.
- M. Mohammadi Amiri and D. Gündüz, "Computation scheduling for distributed machine learning with straggling workers," *arXiv:1810.09992 [cs.DC]*, Oct. 2018.

Chapter 7

In Chapter 7 we study distributed ML at the wireless edge, where power and bandwidth-limited wireless devices with limited local datasets carry out DSGD with the help of a remote PS. We model the communication channel from the edge users to the PS by a wireless MAC with limited number of channel uses. We will provide two different approaches for this wireless DSGD problem, based on *digital* and *analog* computation approaches, respectively. We will show that analog "over-the-air" computation can significantly speed up wireless DSGD, particularly in bandwidth-limited and low-power settings, typically experienced by wireless edge devices. The results in this chapter have been published/submitted for publication in:

- M. Mohammadi Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Paris, France, Jul. 2019.
- M. Mohammadi Amiri and D. Gündüz, "Over-the-air machine learning at the wireless edge," in *Proc. IEEE Int'l Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Cannes, France, Jul. 2019.
- M. Mohammadi Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *arXiv:1901.00844 [cs.DC]*, Jan. 2019.

Chapter 8

Finally, in Chapter 8 we provide the conclusions of the research presented in this dissertation, and discuss potential research directions that can be considered in the future, as well as open questions and challenges that need to be addressed.

Other Publications

The following papers have also been published/submitted for publication as results of the works carried out during the Ph.D. studies, but their contents have not been included in this dissertation:

- M. Mohammadi Amiri and D. Gündüz, "Federated learning over wireless fading channels," *arXiv:1907.09769 [cs.IT]*, Jul. 2019.
- Q. Yang, M. Mohammadi Amiri, and D. Gündüz, "Audience-retention-rate-aware caching and coded video delivery with asynchronous demands," *IEEE Transactions on Communications*, to appear.
- M. Mohammadi Amiri, T. M. Duman, and D. Gündüz, "Collaborative machine learning at the wireless edge with blind transmitters," in *Proc. IEEE Global*

Conference on Signal and Information Processing (GlobalSIP), Ottawa, Canada, Nov. 2019.

- J. Zhao, M. Mohammadi Amiri, and D. Gündüz, "A low-complexity cache-aided multi-antenna content delivery scheme," in *Proc. IEEE Int'l Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Cannes, France, Jul. 2019.
- Q. Yang, M. Mohammadi Amiri, and D. Gündüz, "Audience retention rate aware coded video caching," in *Proc. IEEE Int'l Conf. on Commun. (ICC)*, Paris, France, pp. 1189-1194, May 2017.

Chapter 2

Fundamental Limits of Coded Caching

2.1 Overview

In this chapter, we first consider a *centralized coded caching* system, where a server, hosting popular contents, fulfills users' demands, each equipped with a cache memory, through an error-free shared link. The goal is to minimize the number of bits delivered by the server over the shared link, known as the *delivery rate*, over all user demand combinations. We first consider a homogeneous scenario, where each user is equipped with a cache of the same size. A novel coded caching scheme for relatively small cache sizes is proposed. It is shown that the proposed scheme achieves a smaller delivery rate than the state-of-the-art schemes when the number of users in the system is not less than the number of files, which may arise after releasing new episodes of popular TV series, breaking news videos, or broadcasting different software updates to clients. Furthermore, we prove that the delivery rate of the proposed scheme is within a constant multiplicative factor of 2 of the optimal delivery rate for relatively small cache sizes. We then extend the scheme to a *decentralized coded caching* system, as well as the case when each user has access to a cache of distinct size and show that the proposed scheme outperforms the state-of-the-art results.

2.2 Introduction

In this chapter, we focus on the coded caching model proposed in [3], which considers a single server with a database of N popular contents of equal size (F bits), serving K

users, where user i has local storage space, sufficient to store M_i files, that can be used to proactively cache content during off-peak hours, for $i \in [K]$. In this proactive caching model, the *placement phase* takes place during off-peak traffic hours when the resources are abundant, without the knowledge of particular user demands. When the user demands are revealed, the *delivery phase* is performed, in which a multicasting common message is transmitted from the server to all the users over the shared communication channel. Each user decodes its requested file by combining the bits received in the delivery phase with the contents stored in its local cache.

We first consider $M_i = M$, $\forall i \in [K]$, and propose a novel centralized coded caching scheme, when the normalized cache size of the users is given by $M = (N-1)/K$. This new caching scheme utilizes coded content placement, in which contents are partitioned into smaller chunks, and pairwise XOR-ed contents are placed in the user caches. The delivery phase utilizes both coded and uncoded transmission. We show that the proposed caching scheme requires a smaller delivery rate (evaluated for the worst-case user demands) compared to the state-of-the-art scheme for the same cache size, when $N < K$. We highlight here that this scenario is valid for contents that become highly popular over the Internet, and are demanded by a huge number of users, each equipped with a cache memory of comparatively small size, in a relatively short time interval, for example, viral videos distributed over social networks, new episodes of popular TV series, breaking news videos, or for broadcasting different software updates to millions of clients. We also show that the delivery rate achieved by the proposed caching scheme is within a constant multiplicative factor of 2 of the optimal delivery rate for cache capacities satisfying $1/K \leq M \leq (N-1)/K$, when $K > N \geq 3$. We then extend the scheme and propose a novel *group-based centralized* (GBC) coded caching scheme for a cache size of $M = N/K$ at the users. It is shown that the GBC scheme achieves a lower delivery rate compared to the state-of-the-art results when $K > N \geq 3$. We further employ the idea behind the GBC scheme in the decentralized caching setting, and introduce the *group-based decentralized* (GBD) caching scheme, which is shown to achieve a smaller delivery rate compared to the state-of-the-art schemes. We finally extend the GBD scheme to the decentralized caching scenario where users have caches of distinct sizes.

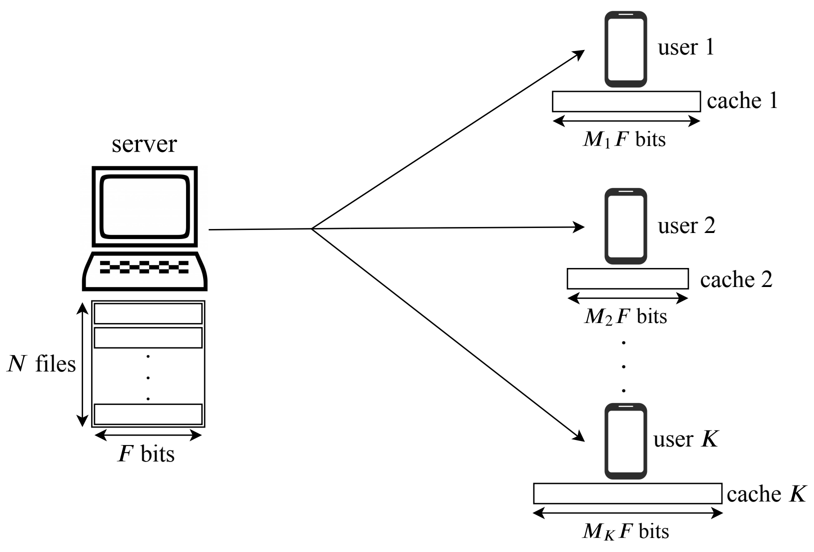


FIGURE 2.1: Illustration of a caching system consisting of a server with a database of N files, each with size F bits, serving K users. User i has access to a cache of capacity M_iF bits, and requests a single file from the database, $i \in [K]$. These requests are served simultaneously through an error-free shared link.

The remainder of this chapter is organized as follows. In Section 2.3 we present the system model. We describe and analyze the centralized coded caching and decentralized coded caching schemes in Section 2.4 and Section 2.5, respectively. We present the numerical results in Section 2.6. Conclusions are drawn in Section 2.7. The detailed proofs are provided in Appendix A.

2.3 System Model

A server with a content library of N independent files $\mathbf{W} \triangleq (W_1, ..., W_N)$ is considered. All the files in the library are assumed to be of length F bits, and each of them is chosen uniformly randomly over the set $\left[2^F\right]$. As depicted in Fig. 2.1, there are K users, where user i is equipped with a cache memory of size M_iF bits, with the normalized cache size $M_i < N$, $\forall i \in [K]$, and each user requests a single file from the library. The files have uniform popularity across the users, and users request their desired files almost simultaneously. Data delivery is divided into two phases. User caches are filled during the *placement phase*. Let B_i denote the contents of user i's cache at the end of the placement phase, which is a function of the database $\mathbf{W}$ given by $B_i = \phi_i^{(F)}(\mathbf{W})$, for a caching function

$$\phi_i^{(F)} : \left[2^F\right]^N \to \left[2^{\lfloor FM_i \rfloor}\right], \quad \text{for } i \in [K]. \tag{2.1}$$

For centralized caching systems, the placement phase is designed by knowing the identities of the users that will participate in the delivery phase. On the other hand, for decentralized caching, cache contents of each user are independent of the number and identities of other users in the system. User requests are revealed after the placement phase, where $d_i \in [N]$ denotes the demand of user i, for $i \in [K]$. These requests are served simultaneously through an error-free shared link in the *delivery phase*.[1] The DF-bit message sent over the shared link by the server in response to the demand vector $\mathbf{d} \triangleq (d_1, ..., d_K)$ is denoted by $X^{(F)}$, where $X^{(F)} \in [2^{DF}]$, and it is generated by the encoding function

$$\psi^{(F)} : \left[2^F\right]^N \times [N]^K \to \left[2^{\lfloor FD \rfloor}\right], \tag{2.2}$$

i.e., $X^{(F)} = \psi^{(F)}(\mathbf{W}, \mathbf{d})$. User i reconstructs its requested file W_{d_i} after receiving the common message $X^{(F)}$ in the delivery phase along with its cache contents B_i. The reconstruction at user i for the demand combination $\mathbf{d}$ is given by $\hat{W}_i = \mu_i^{(F)}(B_i, X, \mathbf{d})$, $\forall i \in [K]$, where

$$\mu_i^{(F)} : \left[2^{\lfloor FM_i \rfloor}\right] \times \left[2^{\lfloor FD \rfloor}\right] \times [N]^K \to \left[2^F\right] \tag{2.3}$$

is the decoding function at user i. For a given content delivery network, the tuple $\left(\phi_{[K]}^{(F)}, \psi^{(F)}, \mu_{[K]}^{(F)}\right)$ constitute a caching and delivery code with delivery rate[2] D. We are interested in the *worst-case* delivery rate, that is the delivery rate that is sufficient to satisfy all demand combinations. Accordingly, the error probability is defined over all demand combinations as follows.

[1]We first consider an error-free communication channel to study the fundamental limits of caching in a simple setting. More complicated channel models for the delivery phase will be considered in the following chapters.

[2]Delivery rate is a unitless metric specifying the normalized number of bits transmitted by the server during the delivery phase, normalized by F.

Definition: The error probability of a $\left(\phi_{[K]}^{(F)}, \psi^{(F)}, \mu_{[K]}^{(F)}\right)$ caching and delivery code described above is given by

$$P_e \triangleq \max_{\mathbf{d}\in[N]^K} \Pr\left\{\bigcup_{i=1}^{K} \left\{\hat{W}_i \neq W_{d_i}\right\}\right\}. \tag{2.4}$$

where the family of demand combinations maximizing P_e are referred to as the worst-case user demands.

Definition: For a content delivery network with N files and K users, we say that a cache size-delivery rate tuple $\left(M_{[K]}, D\right)$ is achievable if, for every $\varepsilon > 0$, there exists a caching and delivery code $\left(\phi_{[K]}^{(F)}, \psi^{(F)}, \mu_{[K]}^{(F)}\right)$ with error probability $P_e < \varepsilon$, for F large enough.

We highlight that any specific achievable scheme has a distinct worst-case user demands maximizing the error probability. Please refer to Remarks 2.4.1 and 2.4.2 for the worst-case user demands analysis for a specific achievable scheme studied in Section 2.4.1.

There is a trade-off between the achievable delivery rate D and the cache capacities $M_1, \ldots, M_K$, defined as

$$D^*\left(M_{[K]}\right) \triangleq \min\left\{D : \left(M_{[K]}, D\right) \text{ is achievable}\right\}. \tag{2.5}$$

The main goal of this chapter is to characterize $D^*\left(M_{[K]}\right)$.

We would like to remark that in the system model considered above the library of the files is assumed to remain the same from the placement phase to the delivery phase, and the delivery phase takes place at once with a single-shot delivery. On the other hand, online coded caching introduces techniques to benefit from the caches across the network when the files are updated [43, 44].

2.4 Centralized Coded Caching

We consider $M_i = M$, $\forall i \in [K]$, and denote the optimum delivery rate-cache size trade-off defined in (2.5) by $D^*(M)$. We first propose an achievable scheme, referred to as the *pair-wise coded caching* (PCC) scheme for a normalized cache size $M = (N-1)/K$. We then propose the *group-based centralized coded caching* (GBC) scheme for $M = N/K$.

2.4.1 Pair-wise Coded Caching (PCC) Scheme

Here we introduce the placement and delivery phases of the PCC scheme for a normalized cache size of $M = (N-1)/K$. We first present the PCC scheme through a simple example highlighting its main ingredients.

Example 1. Consider a caching system with a database of $N = 3$ files, W_1, W_2 and W_3. There are $K = 5$ users in the system, each of which is equipped with a cache of capacity $M = (N-1)/K = 2/5$. To perform the placement phase, each file W_i, $\forall i \in [3]$, is first divided into $K = 5$ non-overlapping subfiles $W_{i,j}$, each of the same length $F/5$ bits, for $j \in [5]$. The following contents are then cached by user i, $i \in [5]$, in the placement phase:

$$B_i = \left(W_{1,i} \oplus W_{2,i}, W_{2,i} \oplus W_{3,i}\right). \tag{2.6}$$

Since each subfile $W_{i,j}$ has a length of $F/5$ bits, the cache placement phase satisfies the memory constraint. With the coded cache placement given above, three subfiles are placed in the cache of each user in the form of two coded packets, coded in the XOR-ed from. We note that by receiving any of the subfiles $W_{j,i}$, user i can recover the other two cached subfiles.

We argue that for the proposed caching scheme with $N < K$, the worst-case user demands happens when each file is requested by at least one user. This fact will later be clarified in Remark 2.4.1. By re-labeling the files and re-ordering the users, without loss of generality, the user demand combination is assumed to be $\mathbf{d} = (1, 1, 1, 2, 3)$.

In the delivery phase, each subfile $W_{i,j}$, $\forall i, j$, is further divided into $N-1=2$ distinct pieces $W_{i,j}^{(l)}$, for $l=1,2$, each of size $F/10$ bits, i.e., $W_{i,j}=\left(W_{i,j}^{(1)}, W_{i,j}^{(2)}\right)$, $\forall i, j$. Accordingly, cache contents (2.6) can be rewritten as

$$B_i=\bigcup\nolimits_{l=1}^{2}\left(W_{1,i}^{(l)}\oplus W_{2,i}^{(l)}, W_{2,i}^{(l)}\oplus W_{3,i}^{(l)}\right). \tag{2.7}$$

The contents are then delivered by the server in three different parts. The following contents are sent in each part of the delivery phase:

- Part 1: $W_{2,1}^{(1)}$, $W_{3,1}^{(2)}$, $W_{2,2}^{(1)}$, $W_{3,2}^{(2)}$, $W_{2,3}^{(1)}$, $W_{3,3}^{(2)}$, $W_{1,4}^{(1)}$, $W_{3,4}^{(2)}$, $W_{1,5}^{(1)}$, $W_{2,5}^{(2)}$,
- Part 2: $W_{1,1}\oplus W_{1,2}$, $W_{1,2}\oplus W_{1,3}$,
- Part 3: $W_{2,1}^{(2)}\oplus W_{2,2}^{(2)}$, $W_{2,2}^{(2)}\oplus W_{2,3}^{(2)}$, $W_{1,4}^{(2)}\oplus W_{2,3}^{(2)}$, $W_{3,1}^{(1)}\oplus W_{3,2}^{(1)}$, $W_{3,2}^{(1)}\oplus W_{3,3}^{(1)}$, $W_{1,5}^{(2)}\oplus W_{3,3}^{(1)}$, $W_{2,5}^{(1)}\oplus W_{3,4}^{(1)}$.

Having received the contents delivered in part 1, each user can retrieve all the subfiles placed in its own cache in XOR-ed form. For example, user 1 can decode all the subfiles $W_{i,1}$, for $i\in[3]$, after receiving the pair $\left(W_{2,1}^{(1)}, W_{3,1}^{(2)}\right)$.

With the second part, each user can obtain the subfiles of its desired file that have been cached by another user with the same demand. For example, the contents $W_{1,1}\oplus W_{1,2}$ and $W_{1,2}\oplus W_{1,3}$ help users 1, 2 and 3 to obtain the subfiles of their request, W_1, which have been cached by each other.

Finally, the last part of the delivery phase enables each user to decode the missing pieces of its requested file having been cached by another user with a different demand. For example, the delivered contents $W_{2,1}^{(2)}\oplus W_{2,2}^{(2)}$, $W_{2,2}^{(2)}\oplus W_{2,3}^{(2)}$, and $W_{1,4}^{(2)}\oplus W_{2,3}^{(2)}$ help users 1, 2, and 3 to obtain the piece $W_{1,4}^{(2)}$, and user 4 can also decode the pieces $W_{2,1}^{(2)}$, $W_{2,2}^{(2)}$, and $W_{2,3}^{(2)}$. It can be verified that having received all the bits sent in three parts in the delivery phase, each user can obtain its desired file with a total delivery rate of 2.1. On the other hand, the state-of-the-art achievable scheme, described in details in [141, Section II], achieves a delivery rate of 2.12 for the setting under consideration. We highlight that the gain of the proposed scheme is due to taking into account the users with the same demand, and designing the delivery phase to reduce the delivery rate when there are repeated demands. Also, caching contents in a coded manner and exploiting the cache memories efficiently is beneficial. □

In the sequel, we present the cache placement and delivery phases of the proposed PCC scheme in the general setting, and analyze its delivery rate.

Placement phase: We first generate K non-overlapping subfiles, each of size F/K bits, for each file W_i, $\forall i \in [N]$, denoted by $W_{i,1}, \dots, W_{i,K}$. Similar to [11] we use coded placement; that is, contents are cached in XOR-ed form in the placement phase. However, unlike in [11], instead of XORing subfiles of all the files in the database, we XOR subfiles in pairs. In particular, the following contents are cached by user j, for $j \in [K]$, in the placement phase:

$$B_j = \bigcup\nolimits_{i=1}^{N-1} \left(W_{i,j} \oplus W_{i+1,j}\right). \tag{2.8}$$

Since each subfile has a size of F/K bits, the limited memory of each cache is filled completely by the proposed placement scheme. In this way, each subfile of all the files is cached by exactly one user in the XOR-ed form. Hence, the whole of each file can be found in the caches of the users across the network (in coded form). All the cached subfiles at each user can be recovered by receiving any one of them.

Delivery phase: Note that, in the proposed caching scheme all the database is stored across the caches of the users. Therefore, in the delivery phase, the server first transmits the appropriate subfiles so that each user can recover all the subfiles stored in its cache in XOR-ed form. Then, the server transmits XOR of contents that are available at two different users, where each content is requested by the other user. This, equivalently, enables the two users to exchange their contents. By appropriately pairing subfiles, the server guarantees that each user receives the subfiles of its requested file that have been cached by every other user in the system.

Without loss of generality, by re-ordering the users, it is assumed that the first K_1 users, referred to as the group $\mathcal{G}_1$, request file W_1, the next K_2 users, which form the group $\mathcal{G}_2$, demand W_2, and so on so forth. For notational convenience, we define

$$S_i \triangleq \sum\nolimits_{l=1}^{i} K_l, \tag{2.9}$$

where we set $S_0 \triangleq 0$. Thus, the general-case user demands can be expressed as follows:

$$d_k = i, \quad S_{i-1} + 1 \leq k \leq S_i, \text{ for } i = 1, ..., N. \tag{2.10}$$

It is illustrated in [141, Appendix A] that the delivery rate of the proposed coded caching scheme does not depend on K_i, $\forall i$, i.e., the proposed scheme is not affected by the popularity of the files, as long as $K_i > 0$. Therefore, when $N < K$, the worst-case user demands for the proposed scheme happens when each file is requested by at least one user, i.e., $K_i \geq 1$, for $i \in [K]$.

The proposed delivery phase is divided into three distinct parts, and the contents delivered in part i is denoted by $X_i^{(F)}$, for $i = 1, 2, 3$. Hence, $X^{(F)} = \left(X_1^{(F)}, X_2^{(F)}, X_3^{(F)}\right)$ is transmitted over the shared link in the delivery phase. The delivery phase algorithm is presented for the worst-case user demands when $N < K$, i.e., when there is at least one user requesting each file. The proposed delivery phase algorithm is then extended to all values of N and K for a generic user demand combination assumption by introducing a new variable $N_{\mathbf{d}}$ as the number of distinct files requested by the users for a demand vector $\mathbf{d}$.

To symmetrize the contents transmitted in the delivery phase, this phase is performed by further partitioning each subfile; that is, each subfile $W_{i,j}$, $\forall i, j$, is divided into $(N-1)$ distinct pieces $W_{i,j}^{(1)}, \ldots, W_{i,j}^{(N-1)}$, each of length $F/(K(N-1))$ bits. Considering these smaller pieces, the content placed in the cache of user j, for $j \in [K]$, can be re-written as follows:

$$B_j = \bigcup\nolimits_{l=1}^{N-1} \bigcup\nolimits_{i=1}^{N-1} \left(W_{i,j}^{(l)} \oplus W_{i+1,j}^{(l)}\right). \tag{2.11}$$

The first part of the delivery phase is stated in Algorithm 1. The main purpose of this part is to enable each user j, $\forall j \in [K]$, to retrieve all the subfiles $W_{i,j}$, $\forall i \in [N]$, that have been cached in the cache of user j in XOR-ed form. We remark that according to

Algorithm 1 Part 1 of the delivery phase of PCC

```
1: procedure Per-user coding
2:     for $i = 1, \dots, N$ do
3:         for $l = S_{i-1} + 1, \dots, S_i$ do
4:             for $j = 1, \dots, N$ and $j \neq i$ do
5:                 $m_{j,l} = \begin{cases} j, & j < i \\ j-1, & j > i \end{cases}$
6:                 $X_1^{(F)} \leftarrow \left( X_1^{(F)}, W_{j,l}^{(m_{j,l})} \right)$
7:             end for
8:         end for
9:     end for
10: end procedure
```

the cache placement in (2.11), for each $l \in [N-1]$, by delivering only one of the pieces $W_{1,j}^{(l)}, ..., W_{N,j}^{(l)}$, user j can recover all the pieces $W_{1,j}^{(l)}, ..., W_{N,j}^{(l)}$, $\forall j$. Hence, each user requires a total of $(N-1)$ distinct pieces to recover all the subfiles placed in its cache in XOR-ed form. To perform an efficient and symmetric delivery phase, $(N-1)$ distinct pieces, which are in the cache of user j in XOR-ed form, corresponding to $(N-1)$ different subfiles of the files that are not requested by user j are delivered to that user. For example, for user 1 requesting file W_1, the pieces $\left(W_{2,1}^{(1)}, W_{3,1}^{(2)}, ..., W_{N,1}^{(N-1)} \right)$ are delivered by Algorithm 1. Accordingly, for user j that has requested file W_i, the pieces $\left(W_{1,j}^{(1)}, ..., W_{i-1,j}^{(i-1)}, W_{i+1,j}^{(i)}, ..., W_{N,j}^{(N-1)} \right)$ are delivered over the shared link. Thus, each user j can recover all subfiles $W_{i,j}$, $\forall i \in [N]$, stored in its cache in XOR-ed form. Note also that, Algorithm 1 delivers at most one piece of each subfile over the shared link. In Algorithm 1, we denote the index of the piece of subfile $W_{i,j}$, that is delivered in part 1 of the delivery phase by $m_{i,j}$. We will later refer to these indexes in explaining the other parts of the delivery phase. Note that, the pieces $\left(W_{1,j}^{(1)}, ..., W_{i-1,j}^{(i-1)}, W_{i+1,j}^{(i)}, ..., W_{N,j}^{(N-1)} \right)$ are targeted for user j in group G_i demanding file W_i, for $i \in [N]$, and $j = S_{i-1} + 1, ..., S_i$. Accordingly, for $j \in [N] \setminus \{i\}$, we have

$$m_{j,l} = \begin{cases} j, & j < i, \\ j-1, & j > i, \end{cases} \tag{2.12}$$

which results in $m_{j,l} \leq N - 1$.

Algorithm 2 Part 2 of the delivery phase of PCC

```
1: procedure INTER-GROUP CODING
2:     for $i = 1, \dots, N$ do
3:         $X_2^{(F)} \leftarrow \left(X_2^{(F)}, \bigcup_{j=S_{i-1}+1}^{S_i-1} (W_{i,j} \oplus W_{i,j+1})\right)$
4:     end for
5: end procedure
```

For example, in Example 1 given above, we have $(m_{1,4}, m_{1,5}) = (1,1)$, $(m_{2,1}, m_{2,2}, m_{2,3}, m_{2,5}) = (1,1,1,2)$, and $(m_{3,1}, m_{3,2}, m_{3,3}, m_{3,4}) = (2,2,2,2)$.

Algorithm 2 presents the second part of the proposed delivery phase, which allows each user to obtain its missing subfiles that have been cached by the other users in the same group. Note that, having received part 1 of the delivery phase, user j in group $\mathcal{G}_i$, for $i \in [N]$, and $j \in [S_{i-1}+1 : S_i]$, can recover subfile $W_{i,j}$. Algorithm 2 delivers $\bigcup_{j=S_{i-1}+1}^{S_i-1} (W_{i,j} \oplus W_{i,j+1})$, with which user j can recover all the subfiles $W_{i,S_{i-1}+1}, \dots, W_{i,S_i}$, i.e., the subfiles of file W_i placed in the caches of users in group $\mathcal{G}_i$.

The last part of the proposed delivery phase is presented in Algorithm 3, with which each user can receive the missing pieces of its desired file that have been placed in the cache of users in other groups. We deliver these pieces by exchanging data between the users in different groups. Observe that, for each user in group $\mathcal{G}_i$, for $i \in [N]$, one piece of the subfile of its requested file W_i which is available to the users in $\mathcal{G}_j$, for $j \in [N]$, $j \neq i$, was delivered in the first part of the delivery phase. Therefore, there are $(N-2)$ missing pieces of a file requested by a user, which have been placed in the cache of a user in a different group. For example, by delivering the pieces $\left(W_{2,1}^{(1)}, W_{3,1}^{(2)}, ..., W_{N,1}^{(N-1)}\right)$ to user 1 demanding file W_1 in part 1 of the delivery phase, each user in group $\mathcal{G}_j$ with demand W_j can obtain the piece $W_{j,1}^{(j-1)}$, for $j \in [2 : N]$. Therefore, there are $(N-2)$ missing pieces of the files requested by the users in groups $\mathcal{G}_2, ..., \mathcal{G}_N$, that are available in the cache of user 1. Consider exchanging data between each user p in group $\mathcal{G}_i$ (demanding file W_i) and each user q in group $\mathcal{G}_j$ (demanding file W_j), for $i \in [N-1]$ and $j \in [i+1 : N]$, where $p \in [S_{i-1}+1 : S_i]$ and $q \in [S_{j-1}+1 : S_j]$. The subfile cached by user p (q) requested by user q (p) is $W_{j,p}$ ($W_{i,q}$). According to (2.12), the index of the piece of subfile $W_{j,p}$ ($W_{i,q}$) delivered in the first part of the

Algorithm 3 Part 3 of the delivery phase of PCC

```
1: procedure INTRA-GROUP CODING
2:     for $i = 1, \dots, N-1$ do
3:         for $j = i+1, \dots, N$ do
4:             for $l = 1, \dots, N-2$ do
5:                 $m_1 = \pi_1^{i,j}(l)$
6:                 $m_2 = \pi_2^{i,j}(l)$
7:                 $X_3^{(F)} \leftarrow \left( X_3^{(F)}, \bigcup_{n=S_{j-1}+1}^{S_j-1} \left( W_{i,n}^{(m_1)} \oplus W_{i,n+1}^{(m_1)} \right), \bigcup_{n=S_{i-1}+1}^{S_i-1} \left( W_{j,n}^{(m_2)} \oplus W_{j,n+1}^{(m_2)} \right), W_{i,S_j}^{(m_1)} \oplus W_{j,S_i}^{(m_2)} \right)$
8:             end for
9:         end for
10:    end for
11: end procedure
```

delivery phase is equal to m_{j,S_i} (m_{i,S_j}), $\forall p \in [S_{i-1}+1 : S_i]$ and $\forall q \in [S_{j-1}+1 : S_j]$. Hence, the indexes of the missing pieces of each user in group $\mathcal{G}_i$ $(\mathcal{G}_j)$ available in the cache of a user in group $\mathcal{G}_j$ $(\mathcal{G}_i)$ are $[N-1] \setminus \{m_{i,S_j}\}$ $([N-1] \setminus \{m_{j,S_i}\})$. Let $\pi_1^{i,j}(\cdot)$ and $\pi_2^{i,j}(\cdot)$ be arbitrary permutations on sets $[N-1] \setminus \{m_{i,S_j}\}$ and $[N-1] \setminus \{m_{j,S_i}\}$, respectively, for $i \in [N-1]$ and $j \in [i+1 : N]$. For $m_1 = \pi_1^{i,j}(l)$ and $m_2 = \pi_2^{i,j}(l)$, $\forall l \in [N-2]$, after receiving the corresponding contents delivered by Algorithm 3, all the users in $\mathcal{G}_i$ can recover the pieces $W_{i,S_{j-1}+1}^{(m_1)}, \dots, W_{i,S_j}^{(m_1)}$, and all the users in $\mathcal{G}_j$ can recover the pieces $W_{j,S_{i-1}+1}^{(m_2)}, \dots, W_{j,S_i}^{(m_2)}$.

Having received all three parts of the delivery phase, each user j, $\forall j \in [K]$, can recover all the pieces of its desired file W_{d_j} that have been placed in any of the caches in the system. Together with the proposed placement phase, which guarantees that all the subfiles of each file is available in one of the caches across the network, we can conclude that the demand of each user is satisfied by the proposed caching algorithm. It is to be noted that when $N = 2$, the proposed scheme is equivalent to the one proposed in [11].

We highlight that the delivery phase of the PCC scheme takes into account the repeated demands and reduces the number of delivered packets when the demands are not distinct.

Delivery rate analysis: The delivery rate of the proposed coded caching scheme is provided in the following theorem, whose detailed proof can be found in Appendix A.1.

Theorem 2.1. *In centralized caching with N files, each of length F bits, and K users, each equipped with a cache of capacity MF bits, if $N < K$ and $M = (N-1)/K$, the following worst-case delivery rate is achievable:*

$$D_{\text{PCC}}\left(\frac{N-1}{K}\right) = N\left(1 - \frac{N}{2K}\right). \tag{2.13}$$

It is proved in [141, Appendix B] that for $K > N \geq 3$ the PCC scheme achieves a smaller delivery rate than the state-of-the-art scheme for $M = (N-1)/K$.

Remark 2.4.1. To perform the proposed delivery phase for the general case, without loss of generality, the user demand combination is assumed as in (2.10), such that $K_i \geq 1$, for $i \leq \mathbf{d}$; and $K_i = 0$, otherwise, for some $N_{\mathbf{d}} \leq N$, that is a total of $N_{\mathbf{d}}$ files are requested by the users in the system. In this case, each subfile is divided into $(N_{\mathbf{d}} - 1)$ distinct equal-length pieces, and in the delivery phase algorithm, the value N is substituted by $N_{\mathbf{d}}$. Hence, according to the delivery rate analysis provided in Appendix A.1, all the users' demands can be satisfied by delivering a total number of $D_c = N_{\mathbf{d}}\left(1 - N_{\mathbf{d}}/\left(2K\right)\right)$ file(s). Since D_c is an increasing function of $N_{\mathbf{d}}$, we can conclude that, for $N < K$, the worst-case user demands happens when all the N files in the database are requested by at least one user, i.e., $K_i \geq 1$, for $i \in [N]$.

Remark 2.4.2. Based on Remark 2.4.1, when $N \geq K$, the worst-case user demands corresponds to the case when $N_{\mathbf{d}} = K$, i.e., all the users request distinct files in the database. Hence, the proposed scheme achieves a delivery rate of $K/2$ when $M = (N-1)/K$, which is equal to the delivery rate of the state-of-the-art for the same cache size when $N = K$ and $N = K+1$. However, the scheme proposed in [3] achieves a delivery rate smaller than $K/2$ for $M = (N-1)/K$, when $N \geq K+2$.

In the following, we present an achievable delivery rate-cache size trade-off through memory sharing between the scheme in [11] for $M = 1/K$ and the PCC scheme for $M = (N-1)/K$, when $K \geq N$.

Corollary 2.1. *The delivery rate-cache size trade-off*

$$D_{\text{PCC}}(M) = -N\left(\frac{M}{2} + \frac{1}{2K} - 1\right), \quad \frac{1}{K} \leq M \leq \frac{N-1}{K}, \tag{2.14}$$

is achievable in a centralized caching system with a database of N files, and $K \geq N$.

Theorem 2.2. *For a caching system with N files, and K users, satisfying $K > N \geq 3$, and a normalized cache size of $M \in [1/K, (N-1)/K]$, we have*

$$\frac{D_{\text{PCC}}(M)}{D^*(M)} \leq 2. \tag{2.15}$$

Proof. The proof can be found in [141, Appendix C]. □

2.4.2 Group-Based Centralized Coded Caching (GBC) Scheme

Here we introduce the placement and delivery phases of the GBC scheme along with its delivery rate analysis. We first illustrate the GBC scheme on an example.

Example 2. In this example, we consider $K = 10$ users, $N = 3$ files, and a normalized cache size of $M = 3/10$. Each file W_i is first divided into 10 non-overlapping subfiles $W_{i,j}$, for $j \in [10]$, and $i \in [3]$, each of length $F/10$ bits. Then one subfile from each file is placed into each user's cache; that is, we have $B_j = (W_{1,j}, W_{2,j}, W_{3,j})$, for $j \in [10]$.

The worst-case of user demands is when the requested files are as distinct as possible. Without loss of generality, by re-ordering the users, we consider the following user demands:

$$d_j = \begin{cases} 1, & 1 \leq j \leq 4, \\ 2, & 5 \leq j \leq 7, \\ 3, & 8 \leq j \leq 10. \end{cases} \tag{2.16}$$

In the delivery phase, the users are grouped according to their demands. Users that request file W_i from the server constitute group $\mathcal{G}_i$, for $i \in [3]$. The delivery phase is divided into two distinct parts that are designed based on the group structure.

- Part 1: The first part of the delivery phase is designed to enable each user to retrieve all the subfiles of its demand that have been placed in the caches of users in the same group. As an example, all users 1, ..., 4, i.e., members of group $\mathcal{G}_1$, should be able to decode all the subfiles $W_{1,1}$, $W_{1,2}$, $W_{1,3}$, $W_{1,4}$, i.e., the subfiles stored in the cache of users in $\mathcal{G}_1$, after receiving the message transmitted in part 1. Accordingly, in our example, in part 1 of the delivery phase the server sends the following coded subfiles over the shared link. $W_{1,1} \oplus W_{1,2}$, $W_{1,2} \oplus W_{1,3}$, $W_{1,3} \oplus W_{1,4}$, $W_{2,5} \oplus W_{2,6}$, $W_{2,6} \oplus W_{2,7}$, $W_{3,8} \oplus W_{3,9}$, $W_{3,9} \oplus W_{3,10}$.

- Part 2: The purpose of part 2 is to make sure that each user can retrieve all the subfiles of its desired file, which have been placed in the cache of users in other groups. Hence, the server transmits the following coded subfiles in the second part of the delivery phase. $W_{1,5} \oplus W_{1,6}$, $W_{1,6} \oplus W_{1,7}$, $W_{2,1} \oplus W_{2,2}$, $W_{2,2} \oplus W_{2,3}$, $W_{2,3} \oplus W_{2,4}$, $W_{1,7} \oplus W_{2,4}$, $W_{1,8} \oplus W_{1,9}$, $W_{1,9} \oplus W_{1,10}$, $W_{3,1} \oplus W_{3,2}$, $W_{3,2} \oplus W_{3,3}$, $W_{3,3} \oplus W_{3,4}$, $W_{1,10} \oplus W_{3,4}$, $W_{2,8} \oplus W_{2,9}$, $W_{2,9} \oplus W_{2,10}$, $W_{3,5} \oplus W_{3,6}$, $W_{3,6} \oplus W_{3,7}$, $W_{3,7} \oplus W_{2,10}$.

It can be easily verified that, together with the contents placed locally, user j can decode its requested file W_{d_j}, $\forall j \in [10]$, from the message transmitted in the delivery phase. As a result, by delivering a total of $12F/5$ bits, which corresponds to a delivery rate of 2.4, all user demands are satisfied. The delivery rate of the state-of-the-art achievable scheme for normalized cache size of $M = 3/10$, presented in [15], is given by 2.43. We highlight that, similarly to the PCC scheme, the benefit of the GBC scheme is that it delivers less number of bits when there is repetition in the demand combination. $\square$

Here we introduce the GBC scheme for any N and K values. We consider a normalized cache size of $M = N/K$, i.e., the aggregate size of the cache memories distributed across the network is equivalent to the total size of the database.

Placement phase: We employ the same placement phase proposed in [3] for $M = N/K$, in which each file W_i is divided into K non-overlapping subfiles $W_{i,j}$, for $i \in [N]$ and $j \in [K]$, each with the same size F/K bits, and the cache contents of user j is given by $B_j = (W_{1,j}, W_{2,j}, \dots, W_{N,j})$, $\forall j \in [K]$. It is easy to see that the cache size constraint is satisfied.

Delivery phase: After demand combination $(d_1, \dots, d_K)$ is revealed, the delivery phase is performed. Without loss of generality, by re-ordering the users, it is assumed that the first K_1 users, referred to as group $\mathcal{G}_1$, have the same request W_1, the next K_2

Algorithm 4 Delivery Phase of the GBC Scheme

```
1: Part 1: Exchanging contents between users in the same group
2: for $i = 1, \dots, N$ do
3:     $X_1^{(F)} = \left( \bigcup_{j=S_{i-1}+1}^{S_i - 1} (W_{i,j} \oplus W_{i,j+1}) \right)$
4: end for

5: Part 2: Exchanging contents between users in different groups
6: for $i = 1, \dots, N-1$ do
7:     for $j = i+1, \dots, N$ do
8:         $X_2^{(F)} = \left( \bigcup_{l=S_{j-1}+1}^{S_j - 1} (W_{i,l} \oplus W_{i,l+1}), \bigcup_{l=S_{i-1}+1}^{S_i - 1} (W_{j,l} \oplus W_{j,l+1}), W_{i,S_j} \oplus W_{j,S_i} \right)$
9:     end for
10: end for
```

users, i.e., group $\mathcal{G}_2$, request the same file W_2, and so on so forth. The coded delivery phase of the GBC scheme is carried out in two parts, and the content delivered in part i is denoted by $X_i^{(F)}$, $i \in [2]$. Algorithm 4 presents the delivery phase of the GBC scheme.

Having received the contents delivered in the first part of Algorithm 4, each user can obtain the missing subfiles of its requested file, which are in the cache of users in the same group; that is, each user j in group $\mathcal{G}_i$ requesting file W_i has access to subfile $W_{i,j}$, and can decode all subfiles $W_{i,l}$, $\forall l \in [S_{i-1}+1 : S_i]$, after receiving the contents delivered in line 3 of Algorithm 4, for $i \in [N]$ and $j \in [S_{i-1}+1 : S_i]$. With the contents delivered in the second part, each user can decode the subfiles of its requested file which are in the cache of users in other groups. Note that, all users in group $\mathcal{G}_i$ demanding file W_i have decoded subfile W_{j,S_i}, and they can obtain all subfiles $W_{i,l}$, $\forall l \in [S_{j-1}+1 : S_j]$, i.e., subfiles of file W_i having been cached by users in group $\mathcal{G}_j$, after receiving $\bigcup_{l=S_{j-1}+1}^{S_j-1} (W_{i,l} \oplus W_{i,l+1})$ and $W_{i,S_j} \oplus W_{j,S_i}$, for $i \in [N-1]$ and $j \in [i+1 : N]$. Similarly, all users in group $\mathcal{G}_j$ can decode all subfiles of their requested file W_j, which are in the cache of users in group $\mathcal{G}_i$ by receiving $\bigcup_{l=S_{i-1}+1}^{S_i-1} (W_{j,l} \oplus W_{j,l+1})$ and $W_{i,S_j} \oplus W_{j,S_i}$, for $i \in [N-1]$ and $j \in [i+1 : N]$. In this way, at the end of the proposed delivery phase, the users can recover all the bits of their requested files.

Delivery rate analysis: We argue that when $K \geq N$, the worst-case user demands for the GBC scheme is when there is at least one user requesting each file, i.e., $K_i > 0$, $\forall i \in [N]$. On the other hand, when $K < N$, without loss of generality, by re-ordering the users, the worst-case user demands is assumed to happen when $K_i = 1$, if $i \in [K]$; and $K_i = 0$, otherwise.

When $K < N$, considering the worst-case user demands, the server transmits the contents $\left(\bigcup_{i=1}^{K-1} \bigcup_{j=i+1}^{K} \left(W_{i,S_j} \oplus W_{j,S_i}\right)\right)$ using Algorithm 4, which are similar to the contents delivered using the delivery phase proposed in [3, Algorithm 1] for a normalized cache size of $M = N/K$. Thus, when $N > K$, the GBC scheme achieves a delivery rate $D_{\text{GBC}}(N/K) = (K-1)/2$ for $M = N/K$.

For $K \geq N$, the delivery rate of the GBC scheme is stated in the next theorem, whose proof can be found in Appendix A.2.

Theorem 2.3. *In a centralized coded caching system with N files, each of size F bits, K users, each equipped with a cache of capacity MF bits, where $M = N/K$, the following delivery rate is achievable by the proposed GBC scheme, if $K \geq N$:*

$$D_{\text{GBC}}\left(\frac{N}{K}\right) = N - \frac{N\left(N+1\right)}{2K}. \tag{2.17}$$

The superiority of the GBC scheme over the state-of-the-art is proved in [142, Appendix B] for $K > N \geq 3$ and $M = N/K$. We also highlight here that the GBC scheme is optimal for $0 \leq M \leq N/K$ constrained to uncoded cache placement [20].

2.5 Decentralized Coded Caching

In practice, the number or identities of active users that will participate in the delivery phase might not be known in advance during the placement phase to design a centralized cache placement. In such a scenario, called *decentralized caching*, coordination across users is not possible during the placement phase. In this section, we first consider $M_i = M$, $\forall i \in [K]$, and we propose *group-based decentralized coded caching* (GBD) scheme by extending the GBC scheme to the decentralized caching scenario. We then study the scenario of non-equal cache sizes, and propose an achievable scheme and develop a lower bound on the delivery rate.

2.5.1 Group-Based Decentralized Coded Caching (GBD) Scheme

Here we consider a decentralized caching system, in which neither the number nor the identities of the users that participate in the delivery phase are known during the placement phase. We apply the techniques with the GBC scheme we have developed for the centralized scenario to decentralized caching.

To simplify the notation, for $i \in [K]$, we define $W_{d_i,\mathcal{S}}$ as the bits of file W_{d_i}, the file requested by user i, which have been placed exclusively in the caches of the users in set $\mathcal{S}$, where $\mathcal{S} \subset [K]$.

Placement phase: In the placement phase, each user caches a random subset of MF/N bits of each file independently. Since there are N files, each of length F bits, this placement phase satisfies the memory constraint.

Delivery phase: Similarly to the delivery phase of GBC, we re-order the users and re-label the files such that the first K_1 users, referred to as group $\mathcal{G}_1$, have the same request W_1, the next K_2 users, group $\mathcal{G}_2$, request W_2, and so on so forth.

There are two different procedures for the delivery phase, called DELIVERY-CODED and DELIVERY-RANDOM, presented in Algorithm 5. The server follows either of the two, whichever achieves a smaller delivery rate.

Let us start with the DELIVERY-CODED procedure of Algorithm 5, in which the contents are delivered in three distinct parts, where $X_i^{(F)}$ denotes the contents delivered with part i, $i \in [3]$, and $X_2^{(F)} = (X_{2,1}^{(F)}, X_{2,2}^{(F)})$. The main idea behind the coded delivery phase is to deliver each user the missing bits of its requested file, that have been cached by i user(s), $\forall i \in [0 : K-1]$.

In the first part, the bits of each requested file that are not in the cache of any user are directly delivered by the server. Each transmitted content is destined for all the users in a separate group, which have the same request.

In the second part, the bits of each requested file that have been cached by only one user are served to the users requesting the file by utilizing the GBC scheme developed for the centralized scenario.

Algorithm 5 Coded Delivery Phase of the GBD Scheme

```
1: procedure DELIVERY-CODED
2:     Part 1: Delivering bits that are not in the cache of any user
3:     for $i = 1, 2, \dots, N$ do
4:         $X_1^{(F)} = W_{d_{S_{i-1}+1}, \{\emptyset\}}$
5:     end for
6:     Part 2: Delivering bits that are in the cache of only one user
7:     $X_{2,1}^{(F)} = \bigcup_{i=1}^{N} \bigcup_{n=S_{i-1}+1}^{S_i - 1} \left(W_{i,\{n\}} \oplus W_{i,\{n+1\}}\right)$
8:     $X_{2,2}^{(F)} = \bigcup_{i=1}^{N-1} \bigcup_{j=i+1}^{N} \Big(\bigcup_{n=S_{j-1}+1}^{S_j - 1} \left(W_{i,\{n\}} \oplus W_{i,\{n+1\}}\right),$
           $\bigcup_{n=S_{i-1}+1}^{S_i - 1} \left(W_{j,\{n\}} \oplus W_{j,\{n+1\}}\right), W_{i,\{S_j\}} \oplus W_{j,\{S_i\}}\Big)$
9:     Part 3: Delivering bits that are in the cache of more than one user
10:    for $i = 3, 4, \dots, K$ do
11:        for $\mathcal{S} \subset [1:K], |\mathcal{S}| = i$ do
12:            $X_3^{(F)} = \bigoplus_{s \in \mathcal{S}} W_{d_s, \mathcal{S} \setminus \{s\}}$
13:        end for
14:    end for
15: end procedure
16: procedure DELIVERY-RANDOM
17:    for $i = 1, 2, \dots, N$ do
18:        server delivers enough random linear combination of the bits of the file $W_i$ to the users requesting it in order to decode it
19:    end for
20: end procedure
```

Each user j in group $\mathcal{G}_i$ requests W_i and has already cached $W_{i,\{j\}}$ for $i \in [N]$ and $j \in [S_{i-1}+1 : S_i]$. Having received the bits delivered in line 7 of Algorithm 5, user j can decode all bits $W_{i,\{l\}}$, $\forall l \in [S_{i-1}+1 : S_i]$. The users also receive the missing bits of their requested files having been cached by a user in a different group; that is, by receiving $\bigcup_{n=S_{i-1}+1}^{S_i-1} \left(W_{j,\{n\}} \oplus W_{j,\{n+1\}}\right)$, $\bigcup_{n=S_{j-1}+1}^{S_j-1} \left(W_{i,\{n\}} \oplus W_{i,\{n+1\}}\right)$, and $W_{i,\{S_j\}} \oplus W_{j,\{S_i\}}$, each user in groups $\mathcal{G}_i$ and $\mathcal{G}_j$ can decode the bits of its request which have been placed in the cache of users in the other group, for $i \in [N-1]$ and $j \in [i+1 : N]$.

In the last part, the same procedure as the one proposed in [26] is performed for the missing bits of each file that have been cached by more than one user. Hence, following the DELIVERY-CODED procedure presented in Algorithm 5, each user recovers all the bits of its desired file.

The novelty of the DELIVERY-CODED procedure in the GBD scheme is about delivering the contents that are not cached by any user or cached by only one user, where the delivery phase is performed by taking into account the user demands, and it improves upon the scheme proposed in [26] when the demands are not distinct.

The second delivery procedure, DELIVERY-RANDOM, is as presented in In the last part, the same procedure as the one proposed in [26], and the server delivers enough random linear combinations of the bits of each requested file targeted for the users in the same group requesting that file to decode it.

Delivery rate analysis: In the following, we derive an expression for the delivery rate-cache size trade-off of the proposed GBD scheme, denoted by $D_{\text{GBD}}(M)$. All discussions in this section are stated assuming that $M \leq N$, and F is large enough. For each randomly chosen bit of each file, the probability of having been cached by each user is M/N. Since the contents are cached independently by each user in the placement phase, a random bit of each file is cached exclusively by the users in set $\mathcal{S} \subset [K]$ (and no user outside this set) with probability $(M/N)^{|\mathcal{S}|}(1 - M/N)^{K-|\mathcal{S}|}$.

Similar to the arguments presented for GBC, when $N \leq K$, the worst-case user demands correspond to the scenario in which each file is requested by at least one user, i.e., $K_i > 0$, $\forall i \in [N]$. On the other hand, when $N > K$, without loss of generality, the worst-case user demands can be assumed as $d_i = i$, $\forall i \in [K]$.

When $N \geq K$, for the worst-case user demands described above, similar conditions as the GBC scheme hold, and the GBD scheme achieves the same delivery rate as the decentralized caching scheme proposed in [26], called the decentralized MAN scheme.

Next we consider the more interesting $N < K$ case. We start with the first procedure of Algorithm 5. In part 1, the server delivers N groups of bits, each group corresponding to a different file, which have not been cached by any user. The delivery rate-cache size trade-off for this part of Algorithm 5, $D_{\text{GBD}_1}(M)$, can be evaluated as

$$D_{\text{GBD}_1}(M) = N\left(1 - \frac{M}{N}\right)^K. \tag{2.18}$$

For the second part, we first need to find the total number of XOR-ed contents delivered by the server, each of which has length $(M/N)\left(1-M/N\right)^{K-1}F$ bits (the length of each XOR-ed content is equivalent to the number of bits of a file that have been cached by only one user). Since the delivery phase of the GBC scheme is applied for this part, based on (2.17), it can be easily evaluated that $(NK - N(N+1)/2)$ XOR-ed contents are served[3]. Thus, the delivery rate-cache size trade-off corresponding to the second part of the first procedure of Algorithm 5 is given by

$$D_{\mathrm{GBD}_2}(M) = \left(NK - \frac{N(N+1)}{2}\right)\left(\frac{M}{N}\right)\left(1-\frac{M}{N}\right)^{K-1}. \tag{2.19}$$

The last part of the proposed delivery phase is equivalent to the first delivery phase procedure proposed in [26, Algorithm 1], with which each user can decode the bits of its requested file, which have been cached by more than one user. Following the same technique as [26], the delivery rate corresponding to this part is derived as follows:

$$\begin{aligned} D_{\mathrm{GBD}_3}(M) =& \sum\nolimits_{i=1}^{K-2}\sum\nolimits_{j=2}^{K-i}\binom{K-i}{j}\left(\frac{M}{N}\right)^{j}\left(1-\frac{M}{N}\right)^{K-j} \\ =& -(K-2)\left(1-\frac{M}{N}\right)^{K} - \frac{1}{2}(K-2)(K+1)\left(\frac{M}{N}\right)\left(1-\frac{M}{N}\right)^{K-1} \\ &+\frac{N}{M}\left(1-\left(1-\frac{M}{N}\right)^{K-1}\right)-1. \end{aligned} \tag{2.20}$$

The overall delivery rate-cache size trade-off for the first procedure of Algorithm 5, $D^1_{\mathrm{GBD}}(M)$, is the sum of the delivery rates of all three parts in (2.18), (2.19), and (2.20), and is evaluated as

$$\begin{aligned} &D^1_{\mathrm{GBD}}(M) = D_{\mathrm{GBD}_1}(M) + D_{\mathrm{GBD}_2}(M) + D_{\mathrm{GBD}_3}(M) \\ &= \frac{N}{M} - 1 - \left[(K-N-2)\left(1+\frac{1}{2}(K-N-1)\frac{M}{N}\right)+\frac{N}{M}\right]\left(1-\frac{M}{N}\right)^{K-1}. \end{aligned} \tag{2.21}$$

For the worst-case user demands, it is shown in [26, Appendix A] that the second delivery procedure achieves the same delivery rate-cache size trade-off as the uncoded

[3]Note that, in the delivery phase of the GBC scheme for $M = N/K$, a total of $(NK - N(N+1)/2)$ XOR-ed contents, each of size F/K bits, are delivered, which results in $D_{\mathrm{GBC}}(N/K) = N - N(N+1)/2K$.

scheme, given by

$$D^2_{\text{GBD}}\left(M\right) = K\left(1-\frac{M}{N}\right)\min\left\{1,\frac{N}{K}\right\}. \tag{2.22}$$

The delivery rate of the proposed GBD scheme is evidently the minimum value of $D^1_{\text{GBD}}(M)$ and $D^2_{\text{GBD}}(M)$, which is presented in the following theorem.

Theorem 2.4. *In a decentralized caching system with K users requesting contents from a server with N files in its database, when $N < K$, the following delivery rate-cache size trade-off is achievable by the GBD scheme:*

$$\begin{aligned} &D_{\text{GBD}}\left(M\right) = \left(1-\frac{M}{N}\right)\\ &\times\min\left\{\frac{N}{M}-\left[\left(K-N-2\right)\left(1+\frac{1}{2}\left(K-N-1\right)\frac{M}{N}\right)+\frac{N}{M}\right]\left(1-\frac{M}{N}\right)^{K-2},N\right\}. \end{aligned} \tag{2.23}$$

The analytical proof of the superiority of the GBD scheme over the state-of-the-art decentralized caching scheme is provided in [142, Section C].

Remark 2.5.1. We note that the complexity of the decentralized caching scheme GBD is higher than that of its corresponding centralized caching scheme GBC. The GBD scheme requires a higher number of subpacketization and more overhead for processing the subfiles cached by different users. This is the penalty to be paid due to the lack of knowledge about the identities of the users participating in the delivery phase in advance.

2.5.2 Distinct Cache Capacities

Here we extend the proposed GBD scheme to the scenario with distinct cache capacities. A new lower bound on the delivery rate $D^*(M_{[K]})$ is also provided.

Placement phase: In the placement phase, user i caches a random subset of M_iF/N bits of each file independently, for $i \in [K]$. Since there are N files in the database, a total of M_iF bits are cached by user i satisfying the cache size constraint with equality.

Since each user fills its cache independently, a bit of each file is cached exclusively by the users in set $\mathcal{S} \subset [K]$ with probability $\prod_{i\in\mathcal{S}} (M_i/N) \prod_{j\in[K]\setminus\mathcal{S}} (1 - M_j/N)$.

Delivery phase: We apply the same re-labeling of users into groups based on their requests as with the GBD scheme. We remind that the user demands are as follows:

$$d_j = i, \quad \text{for } i \in [N],\ j \in [S_{i-1} + 1 : S_i]. \tag{2.24}$$

We further order the users within a group according to their cache capacities, and assume, without loss of generality, that $M_{S_{i-1}+1} \leq M_{S_{i-1}+2} \leq \cdots \leq M_{S_i}$, for $i \in [N]$.

The delivery phase of the proposed GBD scheme for distinct cache capacities is presented in Algorithm 6. Similar to GBD, it has two distinct delivery procedures, CODED DELIVERY and RANDOM DELIVERY; and the server chooses the one with the smaller delivery rate.

The CODED DELIVERY procedure in Algorithm 6 follows the similar steps as the CODED DELIVERY procedure in Algorithm 5, except that $\oplus$ is replaced with $\bar{\oplus}$, due to the asymmetry across the users' cache capacities, and consequently, the size of the cached subfiles by different users. We remark that the correctness of the CODED DELIVERY in Algorithm 6 follows similarly to the correctness of the CODED DELIVERY procedure in Algorithm 5.

Delivery rate analysis: Consider first the case $N \geq K$. It can be argued in this case that the worst-case user demands happens if each file is requested by at most one user. Hence, by re-ordering the users, for the worst-case user demands, we have $K_i = 1$, for $1 \leq i \leq K$, and $K_i = 0$, otherwise. In this case, it can be shown that the CODED DELIVERY procedure requires a lower delivery rate than the RANDOM DELIVERY procedure; hence, the server uses the former. In this case, it is possible to simplify the CODED DELIVERY procedure such that, only coded message $X_2^{(F)} = \bigcup_{i=1}^{N-1} \bigcup_{j=i+1}^{N} W_{i,\{S_{j-1}+1\}} \bar{\oplus} W_{j,\{S_{i-1}+1\}}$ is transmitted in Part 2. The corresponding common message $X^{(F)} = \left(X_1^{(F)}, X_2^{(F)}, X_3^{(F)}\right)$ transmitted over the CODED DELIVERY procedure reduces to the delivery phase of [32, Algorithm 2]. Thus, the GBD scheme achieves the same delivery rate as [32, Algorithm 2] when $N \geq K$.

Algorithm 6 Coded Delivery Phase for Distinct Cache Capacities Scenario

```
1: procedure Coded Delivery
2:     Part 1: Delivering bits that are not in the cache of any user
3:     for $i = 1, 2, \ldots, N$ do
4:         $X_1^{(F)} = W_{d_{S_{i-1}+1},\emptyset}$
5:     end for

6:     Part 2: Delivering bits that are in the cache of only one user
7:     $X_{2,1}^{(F)} = \bigcup_{i=1}^{N} \bigcup_{n=S_{i-1}+1}^{S_i-1} \left(W_{i,\{n\}} \bar{\oplus} W_{i,\{n+1\}}\right)$
8:     $X_{2,2}^{(F)} = \bigcup_{i=1}^{N-1} \bigcup_{j=i+1}^{N} \left(\bigcup_{n=S_{j-1}+1}^{S_j-1} \left(W_{i,\{n\}} \bar{\oplus} W_{i,\{n+1\}}\right),\right.$
           $\left.\bigcup_{n=S_{i-1}+1}^{S_i-1} \left(W_{j,\{n\}} \bar{\oplus} W_{j,\{n+1\}}\right), W_{i,\{S_{j-1}+1\}} \bar{\oplus} W_{j,\{S_{i-1}+1\}}\right)$

9:     Part 3: Delivering bits that are in the cache of more than one user
10:    for $i = 3, 4, \ldots, K$ do
11:        for $\mathcal{S} \subset [1:K], |\mathcal{S}| = i$ do
12:            $X_3^{(F)} = \overline{\bigoplus}_{s \in \mathcal{S}} W_{d_s, \mathcal{S} \setminus \{s\}}$
13:        end for
14:    end for
15: end procedure

16: procedure Random Delivery
17:     for $i = 1, 2, \ldots, N$ do
18:         send enough random linear combinations of the bits of $W_i$ to enable the
        users demanding it to decode it
19:     end for
20: end procedure
```

Next, consider the case $N < K$. It is illustrated in Appendix A.3 that the worst-case user demands happens when N users with the smallest cache capacities all request different files, i.e., they end up in different groups. The corresponding delivery rate is presented in the following theorem, the proof of which can also be found in Appendix A.3.

Theorem 2.5. *In a decentralized content delivery network with N files in the database, each of size F bits, and K users with cache capacities $M_1, \ldots, M_K$ satisfying $M_1 \leq M_2 \leq \cdots \leq M_K$, the following delivery rate-cache size trade-off is achievable when $N < K$:*

$$D_{\mathrm{GBD}}\left(M_{[K]}\right)$$

$$= \min \left\{ \sum_{i=1}^{K} \prod_{j=1}^{i} \left(1 - \frac{M_j}{N}\right) - \Delta D_1\left(M_{[K]}\right) - \Delta D_2\left(M_{[K]}\right), \sum_{i=1}^{N} \left(1 - \frac{M_i}{N}\right) \right\}, \quad (2.25)$$

where

$$\Delta D_1\left(M_{[K]}\right) \triangleq (K-N) \prod_{i=1}^{K} \left(1 - \frac{M_i}{N}\right), \quad (2.26a)$$

$$\Delta D_2\left(M_{[K]}\right) \triangleq \left[\sum_{i=1}^{K-N} (i-1)\frac{M_{i+N}}{N - M_{i+N}}\right] \prod_{j=1}^{K} \left(1 - \frac{M_j}{N}\right). \quad (2.26b)$$

It is proved in [143] that the proposed coded delivery scheme outperforms the one introduced in [32] for $N < K$.

<u>Lower bound:</u> In the next theorem, we generalize the information theoretic lower bound proposed in [21] to the content delivery network with distinct cache capacities.

Theorem 2.6. *In a content delivery network with N files in the database, serving K users with distinct cache capacities, $M_1, \dots, M_K$ assorted in an ascending order, the optimal delivery rate satisfies*

$$D^*\left(M_{[K]}\right) \geq D_{\mathrm{LB}_1}\left(M_{[K]}\right)$$
$$= \max_{\substack{m \in [K], \\ l \in [\lceil N/m \rceil]}} \frac{1}{l} \left\{ N - \frac{m}{m+\gamma} \sum\nolimits_{i=1}^{m+\gamma} M_i - \frac{\gamma(N-ls)^+}{m+\gamma} - (N-Kl)^+ \right\}, \quad (2.27)$$

where $\gamma \triangleq \min\left\{(\lfloor N/l \rfloor - m)^+, K - m\right\}$*,* $\forall m, l$*.*

Proof. The proof of the theorem can be found in [143, Appendix C]. □

Also, assuming that $M_1 \leq M_2 \leq \cdots \leq M_K$, the cut-set based lower bound derived in [32] is given by

$$D_{\mathrm{LB}_2}\left(M_{[K]}\right) = \max_{l \in [1:\min\{N,K\}]} \left\{ l - \frac{\sum_{i=1}^{l} M_i}{\lfloor N/l \rfloor} \right\}. \quad (2.28)$$

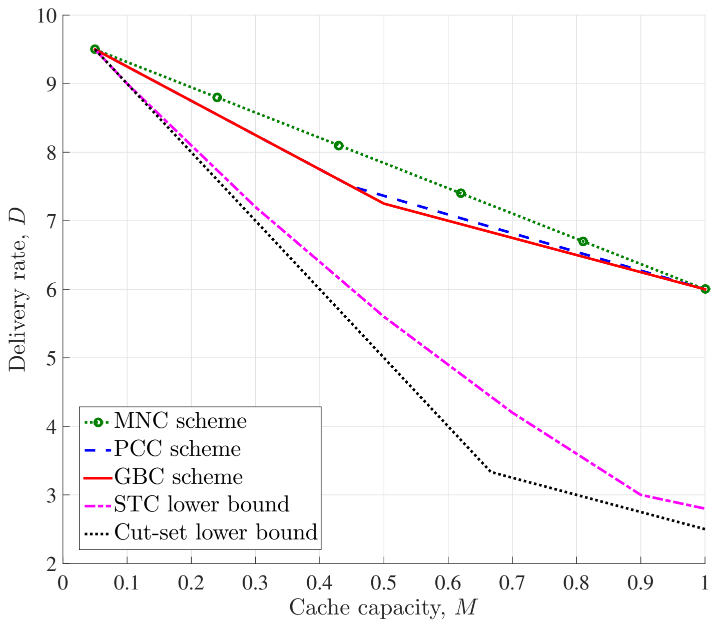


FIGURE 2.2: Delivery rate-cache size trade-off for centralized caching with $N = 10$ files and $K = 20$ users when $1/K \leq M \leq 2N/K$.

2.6 Numerical Results

Here the delivery rates of the proposed schemes for both centralized and decentralized caching approaches are compared numerically with the state-of-the-art results.

2.6.1 Centralized Caching

In Fig. 2.2, we compare the delivery rate-cache size trade-off achieved by the proposed schemes PCC and GBC with the trade-off achieved by memory-sharing between the schemes proposed in [26] and [11], referred to as the MNC scheme here. We consider $N = 10$ files, $K = 20$ users in the system, and a normalized cache size $1/K \leq M \leq 2N/K$ at each user. Observe that the PCC and GBC schemes achieve smaller delivery rate compared to the MNC scheme for all cache size values satisfying $1/K < M < 2N/K$. GBC slightly outperforms PCC, which is due to the improvement for $M = N/K$ extended to a range of cache capacities through memory-sharing. We also include in the figure the two lower bounds on the delivery rate derived in [21, Theorem 1] and [3, Theorem 2], referred to as the STC and cut-set lower bounds, respectively. We observe that, despite the improvement, there is still a gap between the

GBC scheme and the lower bounds, which is mainly due to the looseness of the lower bounds. It has been shown in [20] that, considering only uncoded cache placement phase, the performance of the GBC scheme is optimal for $0 \leq M \leq N/K$.

2.6.2 Decentralized Caching

In Fig. 2.3, we compare the achievable delivery rate of the proposed GBD scheme with the decentralized caching schemes proposed in [26] and [12], referred to as MAN and WTP, respectively, when $M_1 = \cdots = M_K = M$. We consider $N = 30$ files, $K = 50$ users, and relatively small cache capacities $M \in [0, 4]$. We observe that the GBD scheme outperforms the existing schemes in the literature. In Fig. 2.3, we also include the state-of-the-art centralized caching scheme in this setting. Note that, this is not a lower bound on the optimal decentralized delivery rate in general since it is not the optimal centralized delivery rate. However, the difference between the decentralized curves and the centralized curve indicates the loss due to decentralization for these specific coded caching schemes under consideration. The improvement of the GBD scheme over the state-of-the-art is more pronounced for the relatively smaller cache capacities, for which the performance of GBD approaches the best achievable centralized caching performance in a decentralized manner.

Here we evaluate the performance of the proposed GBD scheme for distinct cache capacities numerically. To highlight the gains from the proposed scheme, we also evaluate the performance of uncoded caching, in which user i, $i \in [K]$, caches the first M_i/N bits of each file during the placement phase; and in the delivery phase the remaining $1 - M_i/N$ bits of file W_{d_i} is delivered to user i. By a simple analysis, it can be verified that the worst-case delivery rate is given by

$$D_{\mathrm{uc}}\left(M_{[K]}\right) = \sum\nolimits_{i=1}^{\min\{N,K\}} \left(1 - \frac{M_i}{N}\right), \tag{2.29}$$

which is equal to the delivery rate of the RANDOM DELIVERY procedure in Algorithm 6.

For the numerical results, we consider an exponential cache size distribution among

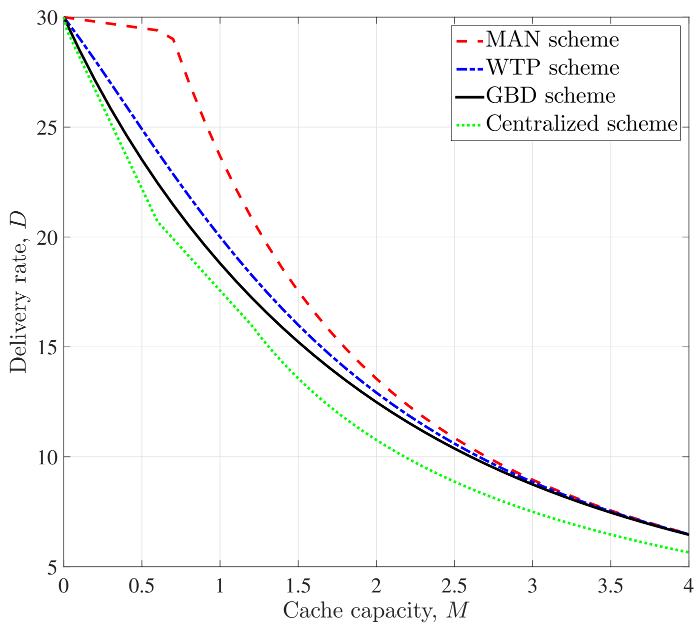


FIGURE 2.3: Delivery rate-cache size trade-off for decentralized caching with $N = 30$ files, $K = 50$ users, and $M \in [0, 4]$.

users, such that the cache size of user i is given by $M_i = \upsilon^{K-i} M_{\max}$, where $0 \le \upsilon \le 1$, for $i \in [K]$, and $M_{\max}$ denotes the maximum cache size in the system. Thus, we have $M_1 \le M_2 \le \cdots \le M_K$, and the total cache size across the network is given by

$$\sum\nolimits_{i=1}^{K} M_i = M_{\max} \upsilon^i \frac{1-\upsilon}{1-\upsilon^{K+1}}. \tag{2.30}$$

In Fig. 2.4, the delivery rate of the GBD scheme for distinct cache capacities scenario is compared with that of studied in [32], referred to as the WLTL scheme, and the uncoded scheme, when $N = 50$, $K = 70$, and $\upsilon = 0.97$. We clearly observe that the proposed GBD scheme outperforms both achievable schemes at all values of $M_{\max}$. The improvement is particularly significant for lower values of $M_{\max}$, and it diminishes as $M_{\max}$ increases. The lower bound given in Theorem 2.6 and the cut-set lower bound are also included in the figure. Although the delivery rate of the proposed scheme meets the lower bounds when $M_{\max} = 0$, the gap in between quickly expands with $M_{\max}$.

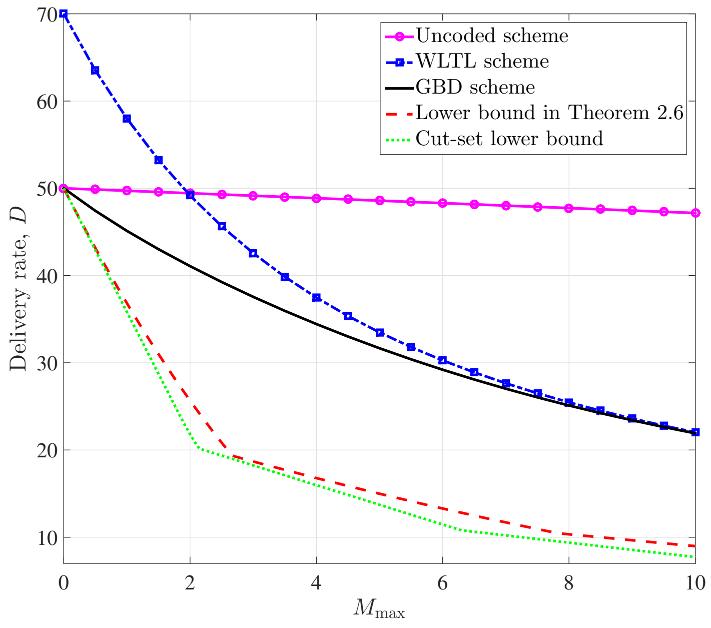


FIGURE 2.4: Delivery rate versus $M_{\max}$, where the cache size of user i is $M_i = \delta^{K-i} M_{\max}$, $i \in [K]$, with $\delta = 0.97$, $N = 50$, and $K = 70$.

2.7 Conclusions

We have considered a caching system with K users and N files of the same size, where user i is equipped with a cache sufficient to store M_i, for $i \in [K]$. The system considered here models wireless networks, in which the caches are filled over off-peak periods without any cost constraint or rate limitation (apart from the limited cache capacities), but without knowing the user demands; and all the user demands arrive (almost) simultaneously, and they are served simultaneously through an error-free shared link. We first considered a homogeneous case, in which $M_i = M$, $\forall i \in [K]$. We have proposed a novel centralized coded caching scheme for a cache size $M = (N-1)/K$, called the PCC scheme, with a delivery rate lower than the state-of-the-art scheme for the same cache size. The PCC scheme performs a coded placement phase, and achieves an order-optimal delivery rate, which is shown to be within a constant multiplicative factor of 2 of the theoretically optimal delivery rate for cache capacities satisfying $1/K \leq M \leq (N-1)/K$, when $K > N \geq 3$. We have then extended the idea behind the delivery phase of the PCC scheme, and proposed GBC scheme for a cache size of $M = N/K$. We have shown that the GBC scheme outperforms the centralized caching schemes in the literature in terms of the delivery rate. We have next employed the GBC scheme in the decentralized setting, where the users cache bits of contents randomly, and referred to the scheme as GBD. We have shown that the GBD scheme also achieves a smaller delivery rate than other decentralized caching schemes in the literature. The improvement is achieved by creating more multicasting opportunities for the delivery of bits that have not been cached by any of the users, or cached by only a single user. We have then applied the GBD scheme in the asymmetric scenario of distinct cache capacities at the users, and shown that the proposed scheme improves upon the state-of-the-art scheme.

Chapter 3

Caching over Erasure Broadcast Channels

3.1 Overview

In this chapter we study a cache-aided broadcast network, in which a server delivers contents to a group of receivers over a packet erasure BC. The receivers are divided into two sets with regards to their channel qualities: the *weak* and *strong* receivers, where all the weak receivers have statistically worse channel qualities than all the strong receivers. The weak receivers, in order to compensate for the high erasure probability they encounter over the channel, are equipped with cache memories of equal size, while the receivers in the strong set have no caches. Data can be pre-delivered reliably to weak receivers' caches over the off-peak traffic period before the receivers reveal their demands. We propose a joint caching and channel coding scheme, which divides each file into several subfiles, and applies a different caching and delivery scheme for each subfile. It is shown that all the receivers, even those without any cache memories, benefit from the presence of caches across the network. An information theoretic trade-off between the cache size and the achievable rate is formulated. It is shown that the proposed scheme improves upon the state-of-the-art in terms of the achievable trade-off.

3.2 Introduction

In contrast to the setting introduced in [3], we consider a noisy channel for the delivery phase [76, 77, 85, 86, 89, 91]. Here, we follow the model considered in [91], and assume that the delivery phase takes place over a memoryless packet erasure BC, which models a packetized communication system, where each packet is separately channel coded against errors at the physical layer, so that a packet either arrives at the receiver correctly, or is lost. Communication over the Internet is usually modeled as a packet erasure channel. The receivers in the system are grouped into two disjoint sets of weak and strong receivers. All the weak receivers are assumed to have statistically worse channels than the strong receivers, while the users in each set can have arbitrary erasure probabilities. To compensate for their worse channel quality, each weak receiver is equipped with a cache memory of equal size.

Assuming equal-rate files in the library, we derive a trade-off between the size of the caches provided to the weak receivers and the rate of the files, for which any demand combination can be reliably satisfied over the erasure BC. The proposed scheme exploits file subpacketization, and performs a different caching and delivery scheme for different subpackets. Moreover, the delivery of the contents to the weak and strong receivers are coupled through the use of a joint encoding scheme to maximally benefit from the available cache memories. We show that, when specified to the homogeneous scenario considered in [91], in which all the receivers in the same set (i.e., weak and strong receivers) have the same erasure probability, the proposed scheme outperforms the one in [91].

The remainder of this chapter is organized as follows. We introduce the system model in Section 3.3. The proposed scheme is elaborated and analyzed in Section 3.4. We present the numerical results in Section 3.5. We conclude this chapter in Section 3.6. The proofs are provided in Appendix B.

3.3 System Model

We consider a server with a library of N files $\mathbf{W}$. Each file is distributed uniformly over the set $\left[\lceil 2^{nR} \rceil\right]$, where R denotes the rate of a file, and n is the number of channel uses during the delivery phase. Receiver i's demand is represented by d_i, where $d_i \in [N]$, $\forall i \in [K]$. All the receivers are served simultaneously over a BC.

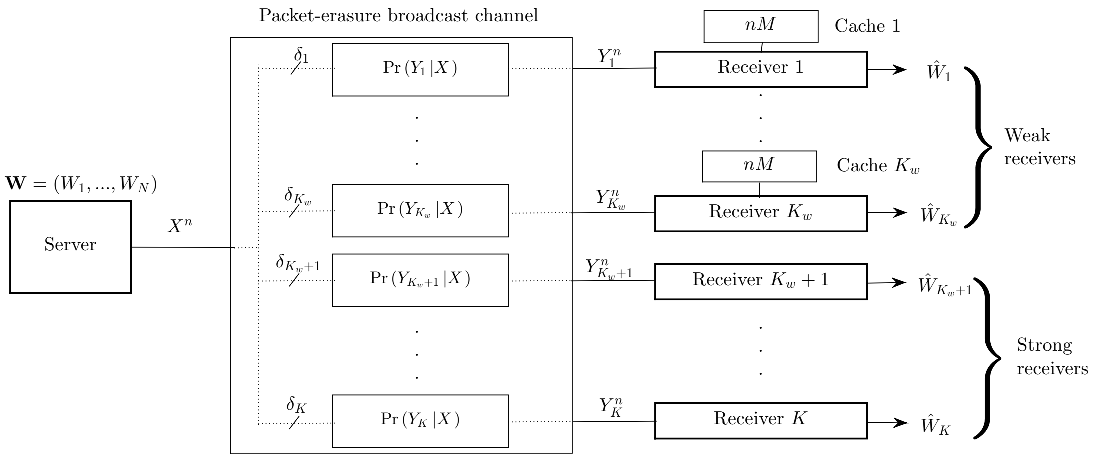


FIGURE 3.1: Cache-aided packet erasure BC. The first K_w receivers have statically worse channels than the last K_s receivers, but each of them is equipped with a cache of normalized size M.

Following [91], the channel between the server and the receivers is modeled as a memoryless packet erasure BC. At each channel use, the server transmits an E-bit codeword from the alphabet $\mathcal{X} \triangleq \{0,1\}^E$, and the output alphabet at each receiver is $\mathcal{Y} \triangleq \mathcal{X} \cup \{\Delta\}$, where the erasure symbol Δ corresponds to a packet that is not received at the receiver. Receiver i, $i \in [K]$, receives the transmitted codeword correctly with probability $1 - \delta_i$, and the erasure symbol Δ with probability δ_i. Thus, given the transmitted codeword $x \in \mathcal{X}$, receiver $i \in [K]$ observes the output $y_i \in \mathcal{Y}$ with the conditional probability

$$\Pr(Y_i = y_i | X = x) = \begin{cases} 1 - \delta_i, & \text{if } y_i = x, \\ \delta_i, & \text{if } y_i = \Delta. \end{cases} \tag{3.1}$$

Two disjoint sets of receivers, *weak* and *strong* receivers, are considered, grouped according to the erasure probabilities of their channels. These groups may model users located in areas with relatively bad and good network coverage, respectively. We assume that the channel condition of each strong receiver is statistically better than that of each weak receiver; that is, the erasure probability of a strong receiver is lower than any weak receiver. Without loss of generality, we enumerate the receivers in the order of improving channel quality, that is, we have $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_K$. We denote the set of erasure probabilities by $\boldsymbol{\delta} \triangleq \{\delta_1, \delta_2, ..., \delta_K\}$. We denote the first K_w receivers as the *weak receivers*, and the next $K_s = K - K_w$ receivers as the *strong receivers*, and we call this case, in which each receiver in each set of weak or strong receivers can have a different erasure probability, as the *heterogeneous scenario*. To compensate for their worse channel quality, each weak receiver is equipped with a cache memory of size nM bits, as depicted in Fig. 3.1. The special case in which all the receivers in the same set have the same erasure probability; that is, all the weak receivers have erasure probability δ_w, and all the strong receivers have erasure probability δ_s, with $\delta_s \leq \delta_w$, is called the *homogeneous scenario*. The set of erasure probabilities for the homogeneous scenario is represented by $\boldsymbol{\delta}_{ws}$.

In the placement phase, the caches of the weak receivers are filled without the knowledge of their future demands, and the contents of the cache of receiver i, for $i \in [K_w]$, at the end of this phase is denoted by B_i. The caching function for receiver $i \in [K_w]$ is given by

$$\phi_{\boldsymbol{\delta},i}^{(nR)} : \left[\lceil 2^{nR} \rceil\right]^N \to \left[\lfloor 2^{nM} \rfloor\right], \tag{3.2}$$

which maps the entire library to the cache content B_i, i.e., $B_i = \phi_{\boldsymbol{\delta},i}^{(nR)}(\mathbf{W})$. Since the placement phase is performed over a low-congestion period, it is assumed that no erasure occurs during this phase[1]. Also, due to the abundance of the resources during the placement phase, we do not take into consideration the transmission cost for this phase.

The delivery phase follows once the demands of the receivers are revealed to the server, which transmits a length-n codeword X^n over the erasure BC. For a demand vector $\mathbf{d}$, a coded delivery function

$$\psi_{\boldsymbol{\delta}}^{(nR)} : \left[\lceil 2^{nR} \rceil\right]^N \times [N]^K \to \mathcal{X}^n \tag{3.3}$$

generates a common channel input X^n as a function of the entire library and the receiver demands, i.e., $X^n = \psi_{\boldsymbol{\delta}}^{(nR)}(\mathbf{W}, \mathbf{d})$. Each receiver $i \in [K]$ observes Y_i^n according to (3.1). Weak receiver $i \in [K_w]$ tries to decode W_{d_i} from its channel output Y_i^n along with the cache content available locally and the demand vector $\mathbf{d}$, with the decoding

[1]To guarantee a reliable communication during the placement phase, we can assume that an automatic repeat request protocol is utilized for transmission.

function

$$\mu_{\boldsymbol{\delta},i}^{(nR)} : \mathcal{Y}^n \times \left[\lfloor 2^{nM} \rfloor\right] \times [N]^K \to \left[\lceil 2^{nR} \rceil\right], \tag{3.4}$$

i.e., the reconstructed file by each weak receiver $i \in [K_w]$ is

$$\hat{W}_i = \mu_{\boldsymbol{\delta},i}^{(nR)} \left(Y_i^n, B_i, \mathbf{d}\right). \tag{3.5}$$

On the other hand, each strong receiver $i \in [K_w + 1 : K]$ reconstructs its demand W_{d_i} solely from its channel output Y_i^n through the decoding function

$$\mu_{\boldsymbol{\delta},i}^{(nR)} : \mathcal{Y}^n \times [N]^K \to \left[\lceil 2^{nR} \rceil\right], \tag{3.6}$$

which generates the reconstructed file

$$\hat{W}_i = \mu_{\boldsymbol{\delta},i}^{(nR)} \left(Y_i^n, \mathbf{d}\right). \tag{3.7}$$

An error occurs if $\hat{W}_i \neq W_{d_i}$ for any $i \in [K]$, and the probability of error P_e is defined as in (2.4).

Definition: A memory-rate pair (M, R) is said to be achievable, if for every $\varepsilon > 0$, there exists a large enough n, and corresponding caching function (3.2), coded delivery function (3.3), and decoding functions (3.4) and (3.6) at weak and strong receivers, respectively, such that $P_e < \varepsilon$.

Definition: For a given cache size M at the weak receivers, the capacity of the network is defined as

$$R^* (M) \triangleq \sup \left\{R : (M, R) \text{ is achievable}\right\}. \tag{3.8}$$

We note that the capacity of the above caching network remains an open problem even when the delivery channel is an error-free shared bit pipe except for uncoded cache placement phase [20]. Here, our goal is to identify achievable memory-rate pairs that improve upon the state-of-the-art.

Remark 3.3.1. It is reasonable to assume that cache memories are placed at receivers with relatively weaker coverage. Indeed, it is shown in [91] that placing cache memories at the strong receivers, which already have good coverage, results in a lower capacity. This is mainly due to the definition of the capacity in this framework. Note that, the capacity here characterizes the highest rate of equal-size messages delivered to all the receivers in the network for any demand combination. Since any receiver can request any of the files, and the files in the library all have the same rate, the system performance is determined by bad receivers. Therefore, in order to increase the capacity the goal of the placement phase should be to improve the performance of the weak receivers. We remark, however, that, equipping weak receivers with cache memories and exploiting the coding scheme proposed in this paper also benefits the strong receivers.

The following results will be instrumental in deriving our results later in the paper.

Proposition 3.3.1. [144] The capacity region of a packet erasure BC with K receivers, where file W_i with rate R_i is targeted for receiver i with erasure probability δ_i, for $i \in [K]$, is the closure of the set of non-negative rate tuples $(R_1, ..., R_K)$ that satisfy

$$\sum\nolimits_{i=1}^{K} \frac{R_i}{(1-\delta_i)\,F} \le 1, \tag{3.9}$$

where the size of the binary channel input per channel use is F bits.

Next, we consider the packet erasure BC with side information, and provide an achievable rate pair based on the joint encoding scheme of [145]. Here we briefly overview the coding scheme and the proof of achievability, and refer the reader to [145] for details. Consider two receivers with erasure probabilities $\delta_1 \ge \delta_2$. Let W_1 and W_2, distributed uniformly over[2] $\left[2^{nR_1}\right]$ and $\left[2^{nR_2}\right]$, denote the messages targeted for receivers 1 and 2, respectively. We assume that message W_2 is available as side information at receiver 1, the weak receiver. We present a coding scheme and the corresponding achievable rate region based on the joint encoding scheme of [145]. For a fixed distribution $\Pr(X)$, we generate $2^{n(R_1+R_2)}$ codewords of length n, $x^n(w_1, w_2)$, $w_1 \in \left[2^{nR_1}\right]$, $w_2 \in \left[2^{nR_2}\right]$, where each entry of each codeword is generated independently according to $\Pr(X)$. The codebook is revealed to the transmitter and the receivers. To transmit particular messages $W_1 = w_1$ and $W_2 = w_2$, the codeword $x^n(w_1, w_2)$ is transmitted over the BC. In the proposed coding scheme, the good receiver, i.e., receiver 2, decodes

[2]We assume that, for any real number $R > 0$, 2^{nR} is an integer for large enough n.

both messages; and therefore, it tries to find a unique pair $(\hat{w}_1, \hat{w}_2) \in \left[2^{nR_1}\right] \times \left[2^{nR_2}\right]$, such that $\left(X^n\left(\hat{w}_1, \hat{w}_2\right), y_2^n\right)$ belongs to the jointly typical set defined in [146]. The probability of decoding error tends to 0 as n goes to infinity, if

$$R_1 + R_2 \leq I\left(X; Y_2\right), \tag{3.10}$$

where $I(\cdot;\cdot)$ represents the mutual information of the arguments. The first receiver already knows W_2 as side information; therefore, it only needs to decode W_1; thus, it looks for a unique index $\hat{w}_1 \in \left[2^{nR_1}\right]$ such that $\left(X^n\left(\hat{w}_1, W_2\right), y_1^n\right)$ belongs to the typical set [146]. The probability of error tends to 0 as n goes to infinity, if

$$R_1 \leq I\left(X; Y_1\right). \tag{3.11}$$

For the packet erasure BC, both mutual information terms are maximized with a uniform input, and the following conditions are obtained:

$$R_1 \leq \left(1 - \delta_1\right) F, \tag{3.12}$$

$$R_1 + R_2 \leq \left(1 - \delta_2\right) F, \tag{3.13}$$

We can easily generalize this coding scheme to multiple receivers and obtain the achievable rate region stated in the following proposition (also provided in [91]).

Proposition 3.3.2. Consider a packet erasure BC with two disjoint sets of receivers $\mathcal{S}_1$ and $\mathcal{S}_2$, where the channels of the receivers in set $\mathcal{S}_i$ have erasure probability δ_i, for $i = 1, 2$. A common message W_i at rate R_i is to be transmitted to the receivers in set $\mathcal{S}_i$, for $i = 1, 2$, while message W_2 is known to the receivers in set $\mathcal{S}_1$ as side information. With the joint encoding scheme outlined above, rate pairs (R_1, R_2) satisfying the following conditions can be achieved

$$R_1 \leq \left(1 - \delta_1\right) F, \tag{3.14}$$

$$R_1 + R_2 \leq \left(1 - \delta_2\right) F, \tag{3.15}$$

which is equivalent to

$$\max\left\{\frac{R_1}{(1-\delta_1)\,F},\frac{R_1+R_2}{(1-\delta_2)\,F}\right\}\leq 1. \tag{3.16}$$

For notational convenience, in the rest of the paper we use

$$\mathrm{JE}\left((W_1)_{\mathcal{S}_1},(W_2)_{\mathcal{S}_2}\right) \tag{3.17}$$

to represent the transmission of message W_1 to the receivers in set $\mathcal{S}_1$, and message W_2 to the receivers in set $\mathcal{S}_2$ using the outlined joint encoding scheme, where $\mathcal{S}_1\cap\mathcal{S}_2=\emptyset$, and W_2 is available at all the receivers in $\mathcal{S}_1$ as side information.

3.4 Successive Cache-Channel Coding (SCC) Scheme

A coding scheme as well as an information theoretic upper bound on the capacity of the above caching and delivery network are proposed in [91] for the homogeneous scenario. Here, we present a new coding scheme, called the *successive cache-channel coding* (SCC) scheme, for delivery over any packet erasure BC, and show that it improves upon [91] in the homogeneous scenario.

Before presenting the SCC scheme for the general heterogeneous scenario, in which we allow the weak and strong receivers to have distinct erasure probabilities, the main ideas behind this scheme are illustrated on an example in the simplified homogeneous scenario.

For notational convenience, the i-element subsets of set $[K_w]$ are enumerated by $\mathcal{S}^i_{[K_w],1},\mathcal{S}^i_{[K_w],2},...,\mathcal{S}^i_{[K_w],\binom{K_w}{i}}$, i.e.,

$$\mathcal{S}^i_{[K_w],j}\subset[K_w] \text{ and } \left|\mathcal{S}^i_{[K_w],j}\right|=i,\quad \text{for } i\in[0:K_w], \text{ and } j\in\left[\binom{K_w}{i}\right]. \tag{3.18}$$

3.4.1 Example

Consider the cache-aided packet erasure homogeneous BC with $K_w = 3$ weak and $K_s = 2$ strong receivers. Here, we investigate the achievable memory-rate pair (M, R) given as follows:

$$R = \frac{F \sum_{i=0}^{2} \gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)}{\frac{1}{1-\delta_w} \sum_{i=0}^{2} \left(\frac{3-i}{i+1} \gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)\right) + \frac{2}{1-\delta_s}}, \tag{3.19a}$$

$$M = \frac{N \sum_{i=0}^{2} i \gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)}{3 \sum_{i=0}^{2} \gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)} R, \tag{3.19b}$$

where $\gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)$, for $p \in [0 : K_w]$ and $q \in [p : K_w]$, is defined as follows:

$$\gamma\left(p, \boldsymbol{\delta}_{ws}, i\right) = \frac{\binom{K_w}{i}}{\binom{K_w}{p} {K_s}^{i-p}} \left(\frac{1-\delta_s}{1-\delta_w} - 1\right)^{i-p}, \quad \text{for } i \in [p : q]. \tag{3.19c}$$

Each file W_l, $l \in [N]$, is divided into three subfiles $W_l^{(0)}$, $W_l^{(1)}$ and $W_l^{(2)}$, where subfile $W_l^{(i)}$ has a rate of

$$R^{(i)} \triangleq \frac{\gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)}{\sum_{j=0}^{2} \gamma\left(0, \boldsymbol{\delta}_{ws}, j\right)} R, \quad \text{for } i \in [0 : 2], \tag{3.20}$$

where $\gamma\left(0, \boldsymbol{\delta}_{ws}, i\right)$ is defined in (3.19c). We have $\sum_{i=0}^{2} R^{(i)} = R$.

Placement phase: In the placement phase, subfiles $W_1^{(i)}, ..., W_N^{(i)}$ are placed in the caches of $K_w = 3$ weak receivers using the procedure in [3, Algorithm 1], specified for a cache size of iN/K_w, for $i \in [0 : 2]$. In this cache placement procedure, each subfile $W_l^{(i)}$ is first divided into $\binom{3}{i}$ non-overlapping pieces, each at a rate of $R^{(i)}/\binom{3}{i}$.

$$W_l^{(i)} = \left(W^{(i)}_{l,\mathcal{S}^i_{[3],1}}, W^{(i)}_{l,\mathcal{S}^i_{[3],2}}, ..., W^{(i)}_{l,\mathcal{S}^i_{[3],\binom{3}{i}}}\right), \quad \forall l \in [N], \forall i \in [0 : 2], \tag{3.21}$$

For the example under consideration, we have, $\forall l \in [N]$,

$$W_l^{(0)} = \left(W_{l,\emptyset}^{(0)}\right), \tag{3.22a}$$

$$W_l^{(1)} = \left(W_{l,\{1\}}^{(1)}, W_{l,\{2\}}^{(1)}, W_{l,\{3\}}^{(1)}\right), \tag{3.22b}$$

TABLE 3.1: Contents sent with messages 1 to 4 in the delivery phase.

Message 1	$W^{(2)}_{d_1,\{2,3\}} \oplus W^{(2)}_{d_2,\{1,3\}} \oplus W^{(2)}_{d_3,\{1,2\}}$ to receivers $1,2,3$	
Message 2	Sub-message 1	$\mathrm{JE}\left(\left(W^{(1)}_{d_1,\{2\}} \oplus W^{(1)}_{d_2,\{1\}}\right)_{\{1,2\}}, \left(W^{(2)}_{d_4,\{1,2\}}, W^{(2)}_{d_5,\{1,2\}}\right)_{\{4,5\}}\right)$
	Sub-message 2	$\mathrm{JE}\left(\left(W^{(1)}_{d_1,\{3\}} \oplus W^{(1)}_{d_3,\{1\}}\right)_{\{1,3\}}, \left(W^{(2)}_{d_4,\{1,3\}}, W^{(2)}_{d_5,\{1,3\}}\right)_{\{4,5\}}\right)$
	Sub-message 3	$\mathrm{JE}\left(\left(W^{(1)}_{d_2,\{3\}} \oplus W^{(1)}_{d_3,\{2\}}\right)_{\{2,3\}}, \left(W^{(2)}_{d_4,\{2,3\}}, W^{(2)}_{d_5,\{2,3\}}\right)_{\{4,5\}}\right)$
Message 3	Sub-message 1	$\mathrm{JE}\left(\left(W^{(0)}_{d_1,\emptyset}\right)_{\{1\}}, \left(W^{(1)}_{d_4,\{1\}}, W^{(1)}_{d_5,\{1\}}\right)_{\{4,5\}}\right)$
	Sub-message 2	$\mathrm{JE}\left(\left(W^{(0)}_{d_2,\emptyset}\right)_{\{2\}}, \left(W^{(1)}_{d_4,\{2\}}, W^{(1)}_{d_5,\{2\}}\right)_{\{4,5\}}\right)$
	Sub-message 3	$\mathrm{JE}\left(\left(W^{(0)}_{d_3,\emptyset}\right)_{\{3\}}, \left(W^{(1)}_{d_4,\{3\}}, W^{(1)}_{d_5,\{3\}}\right)_{\{4,5\}}\right)$
Message 4	$W^{(0)}_{d_4,\emptyset}$ to receiver 4, and $W^{(0)}_{d_5,\emptyset}$ to receiver 5	

$$W_l^{(2)} = \left(W^{(2)}_{l,\{1,2\}}, W^{(2)}_{l,\{1,3\}}, W^{(2)}_{l,\{2,3\}}\right). \tag{3.22c}$$

The piece $W^{(i)}_{l,\mathcal{S}^i_{[3],j}}$ is placed in the cache of each receiver $k \in \mathcal{S}^i_{[3],j}$, for $j \in \left[\binom{3}{i}\right]$. Therefore, the cache contents of the weak receivers after the placement phase are as follows:

$$B_1 = \bigcup\nolimits_{l\in[N]} \left(W^{(1)}_{l,\{1\}}, W^{(2)}_{l,\{1,2\}}, W^{(2)}_{l,\{1,3\}}\right), \tag{3.23a}$$

$$B_2 = \bigcup\nolimits_{l\in[N]} \left(W^{(1)}_{l,\{2\}}, W^{(2)}_{l,\{1,2\}}, W^{(2)}_{l,\{2,3\}}\right), \tag{3.23b}$$

$$B_3 = \bigcup\nolimits_{l\in[N]} \left(W^{(1)}_{l,\{3\}}, W^{(2)}_{l,\{1,3\}}, W^{(2)}_{l,\{2,3\}}\right), \tag{3.23c}$$

where the required cache size for each weak receiver is:

$$M = \left(\frac{R^{(1)}}{3} + \frac{2R^{(2)}}{3}\right)N = \frac{\gamma\left(0, \boldsymbol{\delta}_{ws}, 1\right) + 2\gamma\left(0, \boldsymbol{\delta}_{ws}, 2\right)}{3\sum_{j=0}^{2}\gamma\left(0, \boldsymbol{\delta}_{ws}, j\right)} NR. \tag{3.24}$$

Delivery phase: The server tries to satisfy all the demands in the delivery phase by sending four distinct messages in an orthogonal fashion, i.e., by time division multiplexing, where the codewords corresponding to the i-th message, $i = 1, ..., 4$, are of length $\beta_i n$ channel uses, such that $\sum_{i=1}^{4} \beta_i = 1$. The contents delivered with each message are shown in Table 3.1.

The first message is targeted only for the weak receivers, and its goal is to deliver the missing subfiles of file $W_{d_i}^{(2)}$ to receiver i, $i \in [3]$, that is, having received this message, each weak receiver should be able to decode the third subfile of its desired file. Exploiting the delivery phase of [3, Algorithm 1] for cache size $2N/K_w$, the coded content with message 1 in Table 3.1 is delivered to the weak receivers $\{1, 2, 3\}$. Having received message 1 given in Table 3.1, receiver i can recover its missing piece $W_{d_i,[3]\backslash\{i\}}^{(2)}$ of subfile $W_{d_i}^{(2)}$ using its cache contents B_i. Thus, together with its cache content, receiver i can recover subfile $W_{d_i}^{(2)}$, for $i \in [3]$.

Through the second message of the delivery phase, the server simultaneously delivers subfile $W_{d_i}^{(2)}$ to strong receiver i, $i = 4, 5$, and the missing bits of subfile $W_{d_j}^{(1)}$ to weak receiver j, $j = 1, 2, 3$. The content targeted to the weak receivers is delivered by using the delivery phase of [3, Algorithm 1] for the cache size of N/K_w; that is, the contents

$$\left\{W_{d_1,\{2\}}^{(1)} \oplus W_{d_2,\{1\}}^{(1)}, W_{d_1,\{3\}}^{(1)} \oplus W_{d_3,\{1\}}^{(1)}, W_{d_2,\{3\}}^{(1)} \oplus W_{d_3,\{2\}}^{(1)}\right\} \tag{3.25}$$

are transmitted to the weak receivers. Therefore, the goal is to deliver $W_{d_i}^{(2)}$ to strong receiver i, $i = 4, 5$, while delivering the contents in (3.25) to the weak receivers in parallel. The transmission is performed by sending three sub-messages, transmitted over orthogonal time periods. With the first sub-message of message 2 given in Table 3.1, receivers 1 and 2 receive $W_{d_1,\{2\}}^{(1)} \oplus W_{d_2,\{1\}}^{(1)}$ since they both have $W_{d_4,\{1,2\}}^{(2)}$ and $W_{d_5,\{1,2\}}^{(2)}$ in their caches as side information. Accordingly, receiver 1 and receiver 2 can recover $W_{d_1,\{2\}}^{(1)}$ and $W_{d_2,\{1\}}^{(1)}$, respectively. On the other hand, with the joint encoding scheme, $W_{d_4,\{1,2\}}^{(2)}$ and $W_{d_5,\{1,2\}}^{(2)}$ are directly delivered to receiver 4 and receiver 5, respectively. With the second sub-message of message 2 in Table 3.1, $W_{d_4,\{1,3\}}^{(2)}$ and $W_{d_5,\{1,3\}}^{(2)}$, which are available in the caches of receivers 1 and 3 as side information, are delivered to receivers 4 and 5, while $W_{d_1,\{3\}}^{(1)} \oplus W_{d_3,\{1\}}^{(1)}$ is delivered to receivers 1 and 3. By receiving sub-message 2, receiver 1 and receiver 3 can obtain $W_{d_1,\{3\}}^{(1)}$ and $W_{d_3,\{1\}}^{(1)}$, respectively. Finally, sub-message 3 of message 2 aims to deliver $W_{d_4,\{2,3\}}^{(2)}$ and $W_{d_5,\{2,3\}}^{(2)}$, which are in the cache of receivers 2 and 3, to receivers 4 and 5, respectively, and $W_{d_2,\{3\}}^{(1)} \oplus W_{d_3,\{2\}}^{(1)}$ to receivers 2 and 3 by the joint encoding scheme. Having received coded content $W_{d_2,\{3\}}^{(1)} \oplus W_{d_3,\{2\}}^{(1)}$, receiver 2 and receiver 3 can recover $W_{d_2,\{3\}}^{(1)}$ and $W_{d_3,\{2\}}^{(1)}$, respectively. Thus, having received message 2, each weak receiver i,

$i = 1, 2, 3$, can recover all the missing bits of subfile $W_{d_i}^{(1)}$ of its request, while each strong receiver j, $j = 4, 5$, can obtain subfile $W_{d_j}^{(2)}$ of its request.

The third message of the delivery phase is designed to deliver $W_{d_j}^{(1)}$ to strong receiver j, $j = 4, 5$, and $W_{d_i}^{(0)}$ is delivered to weak receiver i, $i = 1, 2, 3$. Third message is also divided into three sub-messages, transmitted over orthogonal time periods. With sub-message i, given in Table 3.1, $W_{d_4,\{i\}}^{(1)}$ and $W_{d_5,\{i\}}^{(1)}$, both of which are available locally at receiver i as side information, $i = 1, 2, 3$, are delivered to receivers 4 and 5, respectively, while $W_{d_i,\emptyset}^{(0)}$ is delivered to receiver i. Therefore, with the third message in Table 3.1, each weak receiver i, $i = 1, 2, 3$, can obtain $W_{d_i,\emptyset}^{(0)}$, while each strong receiver j, $j = 4, 5$, can recover $W_{d_j}^{(1)}$. Thus, after receiving message 3 in Table 3.1, the demands of the weak receivers are fully satisfied.

The last and fourth message of the delivery phase is generated only for the strong receivers with the goal of delivering them the missing bits of their demands, in particular, subfile $W_{d_j,\emptyset}^{(0)}$ is delivered to each strong receiver j, $j = 4, 5$.

Observe that message 1 in Table 3.1 has a rate of $R^{(2)}/3$. The capacity region of the standard packet erasure BC presented in Proposition 3.3.1 suggests that all the weak receivers can decode message 1, for n large enough, if

$$\frac{R^{(2)}}{3\,(1-\delta_w)\,F} \leq \beta_1. \tag{3.26}$$

From Table 3.1, with each sub-message of the second message, messages of rate $2R^{(2)}/3$, available at the weak receivers as side information, are delivered to the strong receivers; while, simultaneously, a common message at rate $R^{(1)}/3$ is transmitted to the weak receivers. Overall, $\left(W_{d_4}^{(2)}, W_{d_5}^{(2)}\right)$ and the contents in (3.25) with a total rate of $2R^{(2)}$ and $R^{(1)}$ are delivered to the strong and weak receivers, respectively, through three different sub-messages by using the joint encoding scheme of [145] that exploits the side information at the weak receivers. Using the achievable rate region of the joint encoding scheme for the packet erasure channels stated in Proposition 3.3.2, $\left(W_{d_4}^{(2)}, W_{d_5}^{(2)}\right)$ and the contents in (3.25) can be simultaneously decoded by the strong and weak receivers, respectively, for n large enough, if

$$\max\left\{\frac{R^{(1)}}{(1-\delta_w)\,F}, \frac{R^{(1)}+2R^{(2)}}{(1-\delta_s)\,F}\right\} \leq \beta_2. \tag{3.27}$$

From the expressions for $R^{(1)}$ and $R^{(2)}$ in (3.20), it can be verified that the two terms in the maximization in (3.27) are equal for the setting under consideration. Thus, the condition in (3.27) can be simplified as

$$\frac{R^{(1)}}{(1-\delta_w)\,F} \leq \beta_2. \tag{3.28}$$

According to Table 3.1, with each sub-message of message 3, a message at rate $R^{(0)}$ is targeted for the weak receivers, while message at rate $2R^{(1)}/3$, available locally at the weak receivers, is aimed for the strong receivers. Therefore, through joint encoding scheme over three periods, messages with a total rate of $3R^{(0)}$ are delivered to the weak receivers, while the strong receivers receive a total rate of $2R^{(1)}$. According to Proposition 3.3.2, all the weak and strong receivers can decode their messages, for n large enough, if

$$\max\left\{\frac{3R^{(0)}}{(1-\delta_w)\,F}, \frac{3R^{(0)}+2R^{(1)}}{(1-\delta_s)\,F}\right\} \leq \beta_3. \tag{3.29}$$

Again, from the expressions of $R^{(0)}$ and $R^{(1)}$ in (3.20), it can be verified that, when $K_w = 3$ and $K_s = 2$, (3.29) can be simplified as

$$\frac{3R^{(0)}}{(1-\delta_w)\,F} \leq \beta_3. \tag{3.30}$$

From the capacity region of the standard erasure BC in Proposition 3.3.1, each receiver j, $j = 4, 5$, can decode $W^{(0)}_{d_j,\emptyset}$, delivered with message 4, successfully for n sufficiently large, if

$$\frac{2R^{(0)}}{(1-\delta_s)\,F} \leq \beta_4. \tag{3.31}$$

Combining (3.26), (3.28), (3.30), (3.31), and the fact that $\sum_{i=1}^{4}\beta_i = 1$, we have the condition

$$\frac{R^{(2)}}{3\,(1-\delta_w)\,F} + \frac{R^{(1)}}{(1-\delta_w)\,F} + \frac{3R^{(0)}}{(1-\delta_w)\,F} + \frac{2R^{(0)}}{(1-\delta_s)\,F} \leq 1. \tag{3.32}$$

By replacing $R^{(i)}$ with the expressions from (3.20), for $i = 0, 1, 2$, and using the fact that $\gamma\left(0, \boldsymbol{\delta}_{ws}, 0\right) = 1$, (3.32) is reduced to

$$R \leq \frac{\sum_{j=0}^{2} \gamma\left(0, \boldsymbol{\delta}_{ws}, j\right) F}{\frac{1}{1-\delta_w}\left(3 + \gamma\left(0, \boldsymbol{\delta}_{ws}, 1\right) + \frac{1}{3}\gamma\left(0, \boldsymbol{\delta}_{ws}, 2\right)\right) + \frac{2}{1-\delta_s}}. \tag{3.33}$$

Observe that, the term on the right hand side of inequality (3.33) is the same as rate R given in (3.19a). The cache size of each weak receiver exploited by our coding scheme is given by (3.24), which is the same as (3.19b). Thus, the memory-rate pair (M, R) given in (3.19) is achievable for the setting under consideration.

The subpacketization with the proposed SCC scheme provides flexibility to perform a different caching and delivery scheme for different subpackets. This is beneficial in delivering the contents, each in a separate time slot, and jointly useful for both groups of weak and strong receivers. The SCC scheme provides more multicasting opportunities, and enables all the users to exploit the cache memories of the weak users.

In the sequel, we present the placement and delivery phases of the SCC scheme for a general heterogeneous scenario.

3.4.2 Placement Phase

For a given (p, q) pair, where $p \in [0 : K_w]$ and $q \in [p : K_w]$, each file W_l, $l \in [N]$, is divided into $(q - p + 1)$ non-overlapping subfiles, represented by

$$W_l = \left(W_l^{(p)}, ..., W_l^{(q)}\right), \tag{3.34}$$

where subfile $W_l^{(i)}$, for $i \in [p : q]$, has a rate of

$$R^{(i)} \triangleq \frac{\gamma\left(p, \boldsymbol{\delta}, i\right)}{\sum_{j=p}^{q} \gamma\left(p, \boldsymbol{\delta}, j\right)} R, \tag{3.35}$$

where $\gamma(p, \boldsymbol{\delta}, i)$ is defined as follows:

$$\gamma(p, \boldsymbol{\delta}, i) \triangleq \frac{\binom{K_w}{i}}{\binom{K_w}{p} {K_s}^{i-p}} \prod_{j=p}^{i-1} \left(\frac{K_s}{(1-\delta_{K_w-j}) \sum_{l=K_w+1}^{K} \frac{1}{1-\delta_l}} - 1 \right), \text{ for } i \in [p:q]. \quad (3.36)$$

We note that $\sum_{i=p}^{q} R^{(i)} = R$. In the placement phase, for each set of subfiles $W_1^{(i)}, ..., W_N^{(i)}$ a cache placement procedure, corresponding to the one proposed in [3, Algorithm 1] for a cache size of iN/K_w, is performed, $\forall i \in [p:q]$; that is, each subfile $W_l^{(i)}$ is partitioned into $\binom{K_w}{i}$ independent equal-rate pieces,

$$W_l^{(i)} = \left(W^{(i)}_{l,\mathcal{S}^i_{[K_w],1}}, W^{(i)}_{l,\mathcal{S}^i_{[K_w],2}}, ..., W^{(i)}_{l,\mathcal{S}^i_{[K_w],\binom{K_w}{i}}} \right), \quad \forall l \in [N], \forall i \in [p:q]. \quad (3.37)$$

The piece $W^{(i)}_{l,\mathcal{S}^i_{[K_w],l}}$ of rate $R^{(i)}/\binom{K_w}{i}$ is cached by receivers in set $\mathcal{S}^i_{[K_w],j}$, for $j \in \left[\binom{K_w}{i}\right]$. Thus, the content placed in the cache of each weak receiver $k \in [K_w]$ is given by

$$B_k = \bigcup_{l \in [N]} \bigcup_{i \in [p:q]} \bigcup_{j \in \left[\binom{K_w}{i}\right] : k \in \mathcal{S}^i_{[K_w],j}} W^{(i)}_{l,\mathcal{S}^i_{[K_w],j}}. \quad (3.38)$$

Accordingly, $\binom{K_w-1}{i-1}$ pieces, each of rate $R^{(i)}/\binom{K_w}{i}$, corresponding to each subfile $W_d^{(i)}$ are cached by each weak receiver, which requires a total cache size of

$$M = N \sum_{i=p}^{q} \binom{K_w - 1}{i-1} \frac{R^{(i)}}{\binom{K_w}{i}} = \frac{N}{K_w} \sum_{i=p}^{q} i R^{(i)} = \frac{N \sum_{i=p}^{q} i \gamma(p, \boldsymbol{\delta}, i)}{K_w \sum_{i=p}^{q} \gamma(p, \boldsymbol{\delta}, i)} R. \quad (3.39)$$

3.4.3 Delivery Phase

In the delivery phase, the goal is to satisfy all the demands for an arbitrary demand combination $(d_1, ..., d_K)$. The delivery phase consists of $(q-p+2)$ different messages, transmitted over orthogonal time periods, where the codewords of the i-th message are of length $\beta_i n$ channel uses, for $i \in [q-p+2]$, such that $\sum_{i=1}^{q-p+2} \beta_i = 1$.

The first message of the delivery phase is only targeted for the weak receivers, and the goal is to deliver the missing bits of subfile $W_{d_i}^{(q)}$ to receiver i, $\forall i \in [K_w]$. It is to be noted that, for $q = K_w$, based on the cache contents in (3.38), all the weak receivers

TABLE 3.2: Contents sent with messages 1 to $q-p+2$ in the delivery phase of the SCC scheme for the heterogeneous scenario.

Message index	Sub-message index	Delivered content
1	1	$W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],1}}$ to receivers in $\mathcal{S}^{q+1}_{[K_w],1}$
	$\vdots$	$\vdots$
	$\binom{K_w}{q+1}$	$W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],\binom{K_w}{q+1}}}$ to receivers in $\mathcal{S}^{q+1}_{[K_w],\binom{K_w}{q+1}}$
$t=2,...,$ $q-p+1$	1	$\mathrm{JE}\left(\left(W^{(q-t+1)}_{\mathcal{S}^{q-t+2}_{[K_w],1},m}\right)_{\mathcal{S}^{q-t+2}_{[K_w],1}}, \left(W^{(q-t+2)}_{d_{K_w+m},\mathcal{S}^{q-t+2}_{[K_w],1}}\right)_{\{K_w+m\}}\right)$ in m-th time period, for $m=1,...,K_s$
	$\vdots$	$\vdots$
	$\binom{K_w}{q-t+2}$	$\mathrm{JE}\left(\left(W^{(q-t+1)}_{\mathcal{S}^{q-t+2}_{[K_w],\binom{K_w}{q-t+2}},m}\right)_{\mathcal{S}^{q-t+2}_{[K_w],\binom{K_w}{q-t+2}}}, \left(W^{(q-t+2)}_{d_{K_w+m},\mathcal{S}^{q-t+2}_{[K_w],\binom{K_w}{q-t+2}}}\right)_{\{K_w+m\}}\right)$ in m-th time period, for $m=1,...,K_s$
$q-p+2$	$W^{(p)}_{d_i}$ to receiver $i\in[K_w+1:K]$	

have all the subfiles $W^{(q)}_l$, $\forall l\in[N]$; therefore, no message needs to be delivered. In the sequel, we consider $q<K_w$. The first message of the delivery phase is transmitted over $\binom{K_w}{q+1}$ orthogonal time slots, where in each slot, a sub-message is delivered to a group of $q+1$ weak receivers. Sub-message j is a codeword of length $\beta_{1,j}n$ channel uses, and is targeted to the receivers in set $\mathcal{S}^{q+1}_{[K_w],j}$, for $j\in\left[\binom{K_w}{q+1}\right]$, such that $\sum_{j=1}^{\binom{K_w}{q+1}}\beta_{1,j}=\beta_1$. Following the procedure in [3, Algorithm 1], the content delivered by sub-message j is given by

$$W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],j}} \triangleq \bigoplus\nolimits_{i\in\mathcal{S}^{q+1}_{[K_w],j}} W^{(q)}_{d_i,\mathcal{S}^{q+1}_{[K_w],j}\setminus\{i\}}, \quad \text{for } j\in\left[\binom{K_w}{q+1}\right]. \tag{3.40}$$

After receiving $W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],j}}$, each receiver $i\in\mathcal{S}^{q+1}_{[K_w],j}$ can obtain $W^{(q)}_{d_i,\mathcal{S}^{q+1}_{[K_w],j}\setminus\{i\}}$, for $j\in\left[\binom{K_w}{q+1}\right]$, i.e., the missing bits of subfile $W^{(q)}_{d_i}$ of its desired file, having access to Z_i given in (3.38). Thus, together with its cache content, receiver $i\in[K_w]$ can recover $W^{(q)}_{d_i}$.

The delivery technique performed to transmit messages $2,3,...,q-p+1$ follows the same procedure. With the message $q-i+1$ of length $\beta_{q-i+1}n$ channel uses, the server delivers the missing bits of subfile $W_{d_k}^{(i)}$ to each weak receiver k, $k \in [K_w]$, and $W_{d_l}^{(i+1)}$ to each strong receiver l, $l \in [K_w+1:K]$, for $i = q-1, q-2, ..., p$.[3] Message $(q-i+1)$ is delivered through $\binom{K_w}{i+1}$ sub-messages, transmitted over orthogonal time periods, where sub-message j is of length $\beta_{q-i+1,j}n$ channel uses, such that $\sum_{j=1}^{\binom{K_w}{i+1}} \beta_{q-i+1,j} = \beta_{q-i+1}$. With the j-th sub-message, using the coded delivery procedure in [3, Algorithm 1], the coded content

$$W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j}} = \bigoplus\nolimits_{k \in \mathcal{S}^{i+1}_{[K_w],j}} W^{(i)}_{d_k, \mathcal{S}^{i+1}_{[K_w],j} \backslash \{k\}}, \tag{3.41}$$

is delivered to the weak receivers in set $\mathcal{S}^{i+1}_{[K_w],j}$, while

$$\left\{ W^{(i+1)}_{d_{K_w+1}, \mathcal{S}^{i+1}_{[K_w],j}}, ..., W^{(i+1)}_{d_K, \mathcal{S}^{i+1}_{[K_w],j}} \right\}, \tag{3.42}$$

is delivered to the strong receivers, for $i = q-1, ..., p$, and $j = 1, ..., \binom{K_w}{i+1}$. Observe that, after receiving sub-message $W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j}}$, each receiver $k \in \mathcal{S}^{i+1}_{[K_w],j}$ can obtain $W^{(i)}_{d_k, \mathcal{S}^{i+1}_{[K_w],j} \backslash \{k\}}$, for $j = 1, ..., \binom{K_w}{i+1}$, i.e., the missing bits of subfile $W_{d_k}^{(i)}$ of its desired file, for $i = q-1, ..., p$. Note that, the content in (3.42), which is targeted to the strong receivers, is known by each weak receiver in set $\mathcal{S}^{i+1}_{[K_w],j}$. Therefore, the j-th sub-message of message $q-i+1$ can be transmitted using joint encoding, for $i = q-1, ..., p$, $j \in \left[\binom{K_w}{i+1}\right]$:

$$\text{JE}\left(\left(W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j}} \right)_{\mathcal{S}^{i+1}_{[K_w],j}}, \left(W^{(i+1)}_{d_{K_w+1}, \mathcal{S}^{i+1}_{[K_w],j}}, ..., W^{(i+1)}_{d_K, \mathcal{S}^{i+1}_{[K_w],j}} \right)_{[K_w+1:K]} \right). \tag{3.43}$$

However, to increase the efficiency of the delivery phase, the j-th sub-message is delivered via K_s orthogonal time periods, such that in the m-th period a codeword of length $\beta_{q-i+1,j,m}n$ channel uses is transmitted, where $\sum_{m=1}^{K_s} \beta_{q-i+1,j,m} = \beta_{q-i+1,j}$. Coded content $W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j}}$, targeted for receivers in set $\mathcal{S}^{i+1}_{[K_w],j}$, is divided into K_s non-overlapping equal-rate pieces

$$W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j}} = \left(W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j},1}, ..., W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j},K_s} \right), \tag{3.44}$$

[3]For example, with the second message, subfile $W_{d_l}^{(q)}$ is delivered to each strong receiver $l \in [K_w+1 : K]$, and subfile $W_{d_k}^{(q-1)}$ to each weak receiver $k \in [K_w]$. With the third message, subfile $W_{d_l}^{(q-1)}$ is delivered to each strong receiver $l \in [K_w+1:K]$, and subfile $W_{d_k}^{(q-2)}$ to each weak receiver $k \in [K_w]$, and so on so forth.

and the delivery over the m-th time period is performed by joint encoding:

$$\text{JE}\left(\left(W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j},m}\right)_{\mathcal{S}^{i+1}_{[K_w],j}}, \left(W^{(i+1)}_{d_{K_w+m},\mathcal{S}^{i+1}_{[K_w],j}}\right)_{\{K_w+m\}}\right), \text{ for } m=1,...,K_s. \tag{3.45}$$

We note that, after receiving messages 2 to $q-p+1$, each weak receiver $k \in [K_w]$ can obtain subfiles $\left(W^{(q-1)}_{d_k}, W^{(q-2)}_{d_k}, ..., W^{(p)}_{d_k}\right)$, while each strong receiver $l \in [K_w+1:K]$ can decode subfiles $\left(W^{(q)}_{d_l}, W^{(q-1)}_{d_l}, ..., W^{(p+1)}_{d_l}\right)$; therefore, together with message 1, the demand of weak receivers are fully satisfied. However, strong receiver $l \in [K_w+1:K]$ only requires to receive subfile $W^{(p)}_{d_l}$.

The last message delivers subfile $W^{(p)}_{d_l}$ to the strong receiver $l \in [K_w+1:K]$ using the channel coding scheme for standard packet erasure BCs.

The contents delivered with each message in the delivery phase for the heterogeneous case are summarized in Table 3.2.

3.4.4 Achievable Memory-Rate Pair Analysis

In the following theorem, whose proof can be found in Appendix B.1, we provide the memory-rate pair achieved by the SCC scheme.

Theorem 3.1. *Consider cache-aided delivery of N files over a packet erasure BC with K_w weak and K_s strong receivers, where each weak receiver is equipped with a cache of normalized capacity M. Memory-rate pairs $\left(M_{(p,q)}, R_{(p,q)}\right)$ are achievable for any $p \in [0:K_w]$ and $q \in [p:K_w]$, where*

$$R_{(p,q)} \triangleq \frac{F\sum_{i=p}^{q}\gamma\left(p,\boldsymbol{\delta},i\right)}{\sum_{i=p}^{q}\left(\frac{\gamma(p,\boldsymbol{\delta},i)}{\binom{K_w}{i}}\sum_{j=1}^{K_w-i}\frac{\binom{K_w-j}{i}}{1-\delta_j}\right)+\sum_{j=K_w+1}^{K}\frac{1}{1-\delta_j}}, \tag{3.46a}$$

$$M_{(p,q)} \triangleq \frac{N\sum_{i=p}^{q} i\gamma\left(p,\boldsymbol{\delta},i\right)}{K_w\sum_{i=p}^{q}\gamma\left(p,\boldsymbol{\delta},i\right)}R_{(p,q)}, \tag{3.46b}$$

with $\gamma(p,\boldsymbol{\delta},i)$ defined as in (3.36). The upper convex hull of these $(K_w+1)(K_w+2)/2$ memory-rate pairs can also be achieved through memory-sharing.

Corollary 3.1. *For the homogeneous scenario, the achievable memory-rate pairs* $\left(M_{(p,q)}, R_{(p,q)}\right)$*, for any* $p \in [0 : K_w]$ *and* $q \in [p : K_w]$*, are simplified as follows:*

$$R_{(p,q)} = \frac{F \sum_{i=p}^{q} \gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)}{\frac{1}{1-\delta_w} \sum_{i=p}^{q} \left(\frac{K_w - i}{i+1} \gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)\right) + \frac{K_s}{1-\delta_s}}, \tag{3.47a}$$

$$M_{(p,q)} = \frac{N \sum_{i=p}^{q} i \gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)}{K_w \sum_{i=p}^{q} \gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)} R_{(p,q)}, \tag{3.47b}$$

where $\gamma\left(p, \boldsymbol{\delta}_{ws}, i\right)$ *is given in* (3.19c)*.*

3.5 Numerical Results

Here we compare the achievable rate of the SCC scheme for the homogeneous scenario with the scheme of [91], which we will refer to as the STW scheme. In Fig. 3.2, we plot the achievable rates for both schemes in the homogeneous scenario with $K_w = 7$, $K_s = 10$, $N = 50$, $F = 20$, $\delta_s = 0.2$, and $\delta_w = 0.9$. Observe that, for relatively small cache sizes, where the best memory-rate trade-off is achieved by time-sharing between $\left(M_{(0,0)}, R_{(0,0)}\right)$ and $\left(M_{(0,1)}, R_{(0,1)}\right)$, and for relatively large cache sizes, where the best memory-rate trade-off is achieved by time-sharing between $\left(M_{(6,7)}, R_{(6,7)}\right)$ and $\left(M_{(7,7)}, R_{(7,7)}\right)$, both schemes achieve the same rate; however, the proposed SCC scheme achieves a higher rate than STW for all other intermediate cache sizes, and reduces the gap to the upper bound on the capacity derived in [91, Theorem 7]. For a cache size of $M = 30$, SCC provides approximately 15% increase in the achievable rate compared to STW.

In Fig. 3.3, the achievable rates of the SCC and STW schemes in the homogeneous scenario are compared for different values of δ_w for $K_w = 20$, $K_s = 10$, $N = 100$, $F = 50$, $\delta_s = 0.2$, and $\delta_w \in \{0.7, 0.8, 0.9\}$. Observe that, unlike STW, the performance of SCC does not deteriorate notably for intermediate and relatively high cache capacities when δ_w increases, i.e., having worse channel qualities for the weak receivers. This is because SCC successfully exploits the available cache capacities, and there is little to lose from increasing δ_w when M is sufficiantly large. Moreover, the superiority of SCC

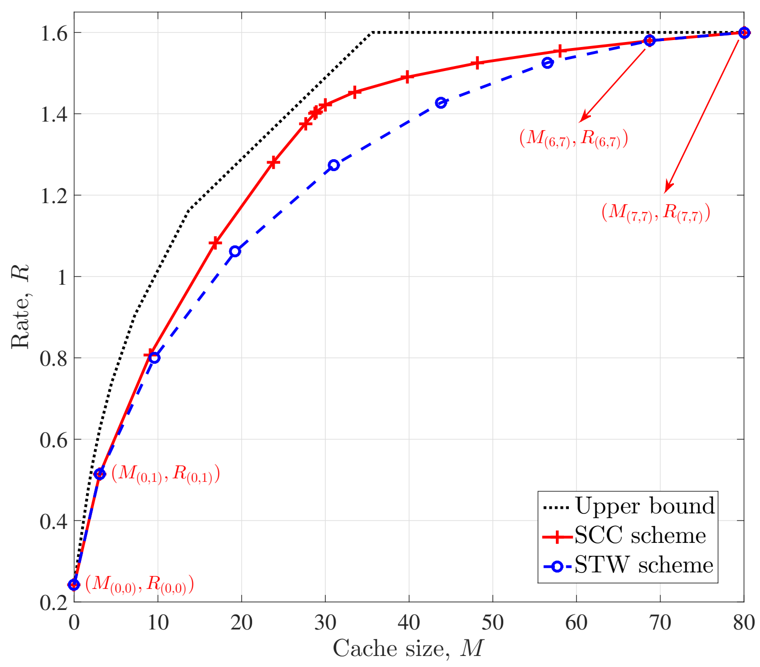


FIGURE 3.2: Lower and upper bounds on the capacity for the homogeneous scenario with $K_w = 7$, $K_s = 10$, $N = 50$, $F = 20$, $\delta_s = 0.2$, and $\delta_w = 0.9$.

over STW is more pronounced for higher values of δ_w, in which case, exploiting the caches of the weak receivers more effectively by SCC becomes more important.

For the heterogeneous scenario, the capacity of the network under consideration is upper bounded by [91]

$$\min_{\mathcal{S}\subset[K]} \left\{ F\left(\sum\nolimits_{i\in\mathcal{S}} \frac{1}{1-\delta_i}\right)^{-1} + \frac{M}{N}\left|\mathcal{S}\cap[K_w]\right| \right\}. \tag{3.48}$$

In Fig. 3.4, the effect of K_w is considered for the heterogeneous scenario for $K = 15$, $N = 100$, $F = 10$, $\delta_i = 0.9 - 0.01i$, for $i \in [5]$, and $\delta_j = 0.2 - 0.01j$, for $j \in [6 : 15]$. Achievable rates are plotted with respect to the total cache size of $K_w M$ for four different values for the number of weak receivers in the system, $K_w \in \{4, 5, 10, 15\}$. Note that the erasure probabilities are set such that the first 5 receivers have significantly worse channels than the remaining 10 receivers. Note also that the parameter K_w determines which receivers are provided with cache memories. As it can be seen, the setting with $K_w = 5$ achieves significantly higher rates over a wide range of total cache capacities compared to the other settings under consideration. If receiver 5, which has a relatively bad channel quality, is not provided with any cache memory, and only

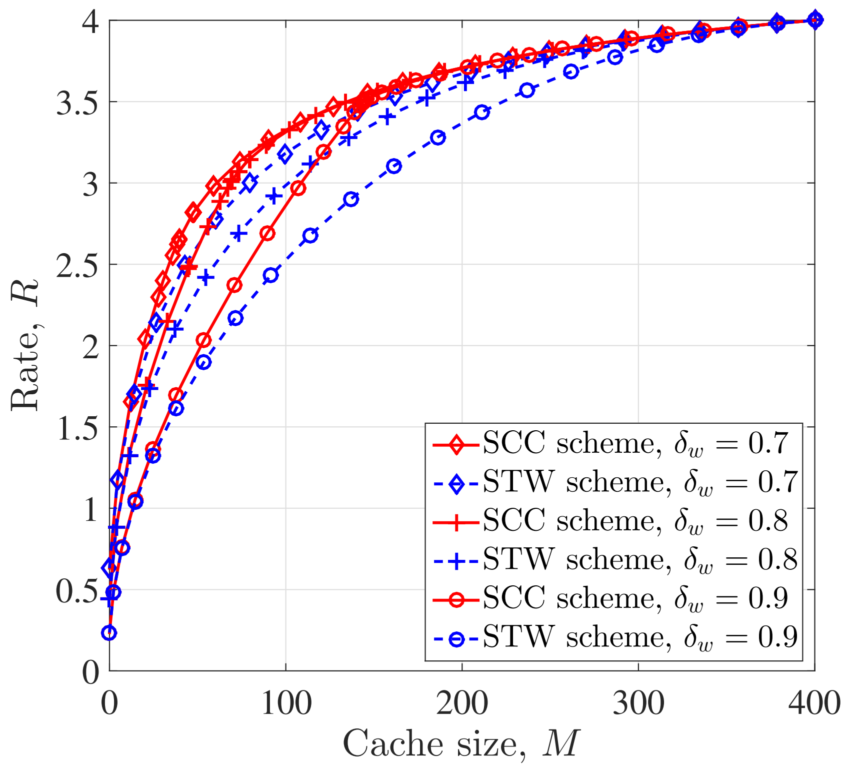


FIGURE 3.3: Lower bounds on the capacity for the homogeneous scenario with $K_w = 20$, $K_s = 10$, $N = 100$, $F = 50$, $\delta_s = 0.2$, and different values for δ_w given by $\delta_w \in \{0.7, 0.8, 0.9\}$.

the first 4 receivers are equipped with cache memories, i.e., $K_w = 4$, the performance degrades significantly except for very small values of total cache size. This is because the first five receivers have much worse channel qualities, and the performance depends critically on the caches provided to all these five weak receivers. On the other hand, equipping receivers with relatively good channel qualities with cache memories deteriorates the performance of the system in terms of the achievable rate. Note that this is because the total available cache size is allocated across a larger number of receivers. This result confirms that it is more beneficial to allocate cache memories to the receivers with relatively worse channel qualities. The upper bound on the achievable rate for the setting with $K_w = 5$ and $K_s = 10$ is also included in this figure. We observe that the gap between the upper bound and the achievable rate for the same setting is relatively small for a wide range of cache sizes.

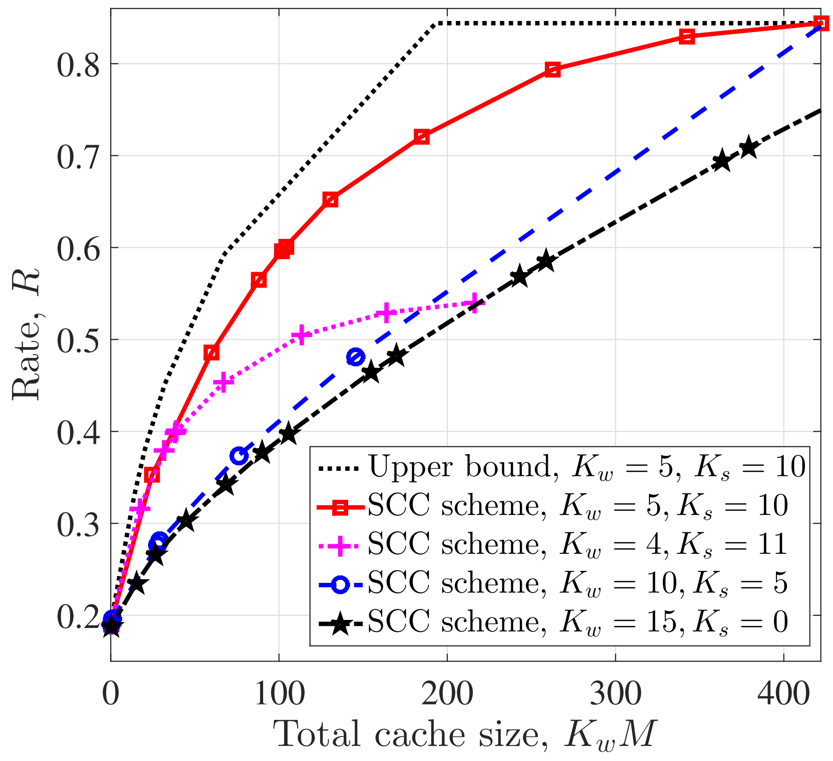


FIGURE 3.4: Lower and upper bounds on the capacity for the heterogeneous scenario with $K = 15$, $N = 100$, $F = 10$, $\delta_i = 0.9 - 0.0ik$, for $i \in [5]$, and $\delta_j = 0.2 - 0.01j$, for $j \in [6:15]$ with variable $K_w \in \{4, 5, 10, 15\}$ and $K_s \in \{0, 5, 10, 11\}$.

3.6 Conclusions

In this chapter We have studied cache-enabled content delivery over a packet erasure BC with arbitrary erasure probabilities. The capacity of this network is defined as the maximum common rate of files in the library, which allows reliable delivery to all the receivers, independent of their demands. We have derived a lower bound on the capacity by proposing a novel caching and delivery scheme, which enables each receiver, even the strong receivers without a cache memory, to benefit from the cache memories available at the weak receivers. The proposed scheme utilizes a finer subpacketization of the files in the library, and provides a better exploitation of the available cache memories with a higher achievable rate than the state-of-the-art. This model and the presented results illustrate that even limited storage can be converted into spectral efficiency in noisy communication networks, benefiting the whole network, if it is placed strategically across the network, and exploited intelligently.

Chapter 4

Caching over Gaussian Broadcast Channels

4.1 Overview

In this chapter we consider a cache-aided K-user Gaussian BC, where the transmitter has a library of N equal-rate files, from which each user demands one. The impact of the equal-capacity receiver cache memories on the minimum required transmit power to satisfy all user demands is studied. Considering uniformly random demands across the library, both the minimum average power (averaged over all demand combinations) and the minimum peak power (minimum power required to satisfy all demand combinations) are studied. Upper bounds are presented on the minimum required average and peak transmit power as a function of the cache size considering both *centralized* and *decentralized* caching. The lower bounds on the minimum required average and peak power values are also derived assuming uncoded cache placement. The bounds for both the peak and average power values are shown to be tight in the centralized scenario through numerical simulations. The results show that proactive caching and coded delivery can provide significant energy savings in wireless networks, even when the caches have a relatively small size.

4.2 Introduction

In this chapter we study the benefits of proactive caching in reducing the transmit power, assuming that the noiseless cache placement phase is carried out without the knowledge of channel conditions during the delivery phase. Assuming uniform popularity across the files, we first study the minimum required *average power* to serve all the users, averaged over all the user demand combinations. Note that we allow the transmitter to change its power depending on the demand combination in order to minimize the average power consumption. We then consider the transmit power required to satisfy the worst-case demand combination, called the *peak power.* We first provide upper bounds on the minimum average and peak power values as a function of the rate of the files in the library and the capacity of the user caches, for centralized cache placement. We then extend the proposed scheme by considering *decentralized* cache placement. The proposed delivery strategy employs superposition coding and power allocation, and the achievable transmit power for any demand combination is derived thanks to the degradedness of a Gaussian BC. We further derive lower bounds on the performance assuming uncoded cache placement.

The main novelty of the proposed proactive caching and coded delivery scheme is the way the coded packets designed for each user are generated for any demand combination, particularly when a file may be requested by more than one user, and the way these coded packets are delivered over a Gaussian BC in order to minimize the transmit power. We show that the proposed achievable scheme reduces the transmit power significantly, even with the availability of only a small cache size at each receiver, in both the centralized and decentralized scenarios. It is also shown that the power loss between the centralized and the more practical decentralized scenario is quite small. Furthermore, numerical results show that the gaps between the peak and average transmit powers of the proposed achievable scheme for the centralized scenario and the corresponding lower bounds are negligible. In particular, we have observed numerically that the multiplicative gap between the two bounds for the centralized caching is below 2 for the examples considered. Numerical results also illustrate that adjusting the transmit power based on the demand vector can significantly reduce the average power consumption compared to the worst-case demand combination.

The remainder of this chapter is organized as follows. In Section 4.3 system model and preliminaries are introduced. The proposed caching and coded delivery scheme is presented for the centralized and decentralized settings in Section 4.4 and Section

4.5, respectively. We derive a lower bound on the performance metric in Section 4.6. Numerical results are presented in Section 4.7. Conclusions are drawn in Section 4.8.

4.3 System Model

We study cache-aided content delivery over a K-user Gaussian BC. The server has a library of N files, $\mathbf{W}$, with uniform popularity across the users, and each distributed uniformly over the set $\left[\lceil 2^{nR} \rceil\right]$. Each user has a cache of size nMR bits. We define the *normalized global cache size* as $r \triangleq MK/N$.

Data delivery from the server to the users takes place in two phases. Caches of the users are filled during the initial *placement phase*, which takes place over a period of low traffic and high energy efficiency; and therefore, data delivery in the placement phase is assumed to be error-free and at a negligible energy cost[1]; however, without either the knowledge of the user demands, or the users' future channel gains when they place their requests. The caching function for user $i \in [K]$ is

$$\phi_i^{(nR)} : \left[\lceil 2^{nR} \rceil\right]^N \to \left[\lfloor 2^{nMR} \rfloor\right], \tag{4.1}$$

which maps the library to the cache contents B_i of user i, i.e., $B_i = \phi_i^{(nR)}(\mathbf{W})$.

We assume that the user demands are independent and uniformly distributed over the file library; that is $\Pr\{d_k = i\} = 1/N$, $\forall i \in [N], \forall k \in [K]$. The requests must be satisfied simultaneously during the *delivery phase*. As opposed to the placement phase, the delivery phase takes place over a noisy BC, characterized by

$$Y_{k,i}(\mathbf{W}, \mathbf{d}) = h_k X_i(\mathbf{W}, \mathbf{d}) + Z_{k,i}, \text{ for } i \in [n], k \in [K], \tag{4.2}$$

where $X_i(\mathbf{W}, \mathbf{d})$ denotes the transmitted signal from the server at time i, h_k is a real channel gain between the server and user k, $Z_{k,i}$ is the zero-mean unit-variance real Gaussian noise at user k at time i, i.e., $Z_{k,i} \sim \mathcal{N}(0,1)$, and $Y_{k,i}(\mathbf{W}, \mathbf{d})$ is the signal

[1]We assume that the placement phase takes place over a significantly longer period of time and over orthogonal high-quality links; which, in theory, allows the server to achieve the minimum energy per bit required to send the cache contents to the users.

received at user k.[2] The noise components are assumed independent and identically distributed (i.i.d.) across time and users. Without loss of generality, we assume that $h_1^2 \leq h_2^2 \leq \cdots \leq h_K^2$. With the knowledge of the channel vector $\mathbf{h} \triangleq (h_1, ..., h_K)$ at the server, the channel input vector $X^n(\mathbf{W}, \mathbf{d})$ is generated by

$$\psi^{(nR)} : \left[\lceil 2^{nR} \rceil\right]^N \times \mathbb{R}^K \times [N]^K \to \mathbb{R}^n, \tag{4.3}$$

and its average power is given by $P(\mathbf{W}, \mathbf{d}) \triangleq \frac{1}{n} \sum_{i=1}^n X_i^2(\mathbf{W}, \mathbf{d})$. We define the average power of this encoding function for demand vector $\mathbf{d}$ as

$$P(\mathbf{d}) \triangleq \max_{W_1, \ldots, W_N} P(\mathbf{W}, \mathbf{d}), \tag{4.4}$$

where the maximization is over all possible realizations of the file library.

User $k \in [K]$ reconstructs $\hat{W}_k$ using its channel output $Y_k^n(\mathbf{W}, \mathbf{d})$, local cache contents B_k, channel vector $\mathbf{h}$, and demand vector $\mathbf{d}$ through the function

$$\mu_k^{(nR)} : \mathbb{R}^n \times \left[\lfloor 2^{nMR} \rfloor\right] \times \mathbb{R}^K \times [N]^K \to \left[\lceil 2^{nR} \rceil\right]. \tag{4.5}$$

An (n, R, M) *code* consists of K caching functions $\phi_1^{(nR)}, \ldots, \phi_K^{(nR)}$, channel encoding function $\psi^{(nR)}$, and K decoding functions $\mu_1^{(nR)}, \ldots, \mu_K^{(nR)}$. We say that an $\left(R, M, \bar{P}, \hat{P}\right)$ tuple is *achievable* if for every $\varepsilon > 0$, there exists an (n, R, M) code with sufficiently large n, which satisfies $P_e < \varepsilon$, $\mathrm{E}_{\mathbf{d}}[P(\mathbf{d})] \leq \bar{P}$, and $P(\mathbf{d}) \leq \hat{P}$, $\forall \mathbf{d}$ with P_e defined in (2.4). For given rate R and normalized cache size M, the average and peak power-memory trade-offs are defined, respectively, as

$$\bar{P}^*(R, M) \triangleq \inf \left\{\bar{P} : \left(R, M, \bar{P}, \infty\right) \text{ is achievable}\right\}, \tag{4.6a}$$

$$\hat{P}^*(R, M) \triangleq \inf \left\{\hat{P} : \left(R, M, \hat{P}, \hat{P}\right) \text{ is achievable}\right\}. \tag{4.6b}$$

Remark 4.3.1. In the above definition, $\bar{P}^*$ is evaluated by allowing a different transmission power for each demand combination, and minimizing the average power across demands; while $\hat{P}^*$ characterizes the worst-case transmit power, which can also be

[2]For simplicity, we consider a real Gaussian channel here, and the extension of the results to the complex channel case is straightforward.

considered as the minimum transmit power required to satisfy all possible demands if the transmitter is not allowed to adapt its transmit power according to user demands.

We will consider both centralized and decentralized caching, and assume, in both scenarios that the placement phase is performed without any information about the channel gains during the delivery phase.

Proposition 4.3.1. *[147]* Consider a K-user Gaussian BC presented above with $M = 0$, where a distinct message of rate R_k is targeted for user k, $k \in [K]$. The minimum total power P that is required to deliver all K messages reliably can be achieved by superposition coding with Gaussian codewords of power $\alpha_k P$ allocated for user k, $\forall k \in [K]$, where[3]

$$\alpha_k P = \left(\frac{2^{2R_k}-1}{h_k^2}\right)\left(1 + h_k^2 \sum\nolimits_{i=k+1}^{K} \left(\frac{2^{2R_i}-1}{h_i^2}\right) \prod\nolimits_{j=k+1}^{i-1} 2^{2R_j}\right), \tag{4.7}$$

and the total transmitted power is

$$P = \sum\nolimits_{k=1}^{K} \left(\frac{2^{2R_k}-1}{h_k^2}\right) \prod\nolimits_{i=1}^{k-1} 2^{2R_i}. \tag{4.8}$$

For a demand vector $\mathbf{d}$ in the delivery phase, we denote the number of distinct demands by $N_{\mathbf{d}}$, where $N_{\mathbf{d}} \leq \min\{N, K\}$. Let $\mathcal{U}_{\mathbf{d}}$ denote the set of users with distinct requests, which have the worst channel qualities; that is, $\mathcal{U}_{\mathbf{d}}$ consists of $N_{\mathbf{d}}$ indices corresponding to users with distinct requests, where a user is included in set $\mathcal{U}_{\mathbf{d}}$ if and onlt if it has the worst channel quality among all the users with the same demand, i.e., if $k \in \mathcal{U}_{\mathbf{d}}$ and $d_k = d_m$ for some $m \in [K]$, then $h_k^2 \leq h_m^2$, or equivalently, $k \leq m$. Note that, for any demand vector $\mathbf{d}$, $1 \in \mathcal{U}_{\mathbf{d}}$. For given $\mathbf{d}$ and user k, $k \in [K]$, let $\mathcal{U}_{\mathbf{d},k}$ denote the set of users in $\mathcal{U}_{\mathbf{d}}$ which have better channels than user k:

$$\mathcal{U}_{\mathbf{d},k} \triangleq \{i \in \mathcal{U}_{\mathbf{d}} : i > k\}, \quad k \in [K]. \tag{4.9}$$

We denote the cardinality of $\mathcal{U}_{\mathbf{d},k}$ by $N_{\mathbf{d},k}$, i.e., $N_{\mathbf{d},k} \triangleq |\mathcal{U}_{\mathbf{d},k}|$.

[3]For two integers i and j, if $i > j$, we assume that $\sum_{n=i}^{j} a_n = 0$, and $\prod_{n=i}^{j} a_n = 1$, where a_n is an arbitrary sequence.

4.4 Centralized Caching and Delivery

Here we present our centralized caching and coded delivery scheme. We follow the placement phase in [20] since the users' channel gains are not known in advance. In the delivery phase, our goal is to identify the coded packets targeted to each user in order to minimize the transmit power. We will use superposition coding in sending multiple coded packets, and benefit heavily from the degradedness of the underlying Gaussian BC.

Theorem 4.1. *In centralized caching followed by delivery over a Gaussian BC, we have*

$$\bar{P}^*(R,M) \leq \bar{P}^{\mathrm{C}}_{\mathrm{UB}}(R,M) \triangleq \frac{1}{N^K} \sum_{\mathbf{d}\in[N]^K} \left[\sum_{i=1}^{K} \left(\frac{2^{2R^{\mathrm{C}}_{\mathbf{d},i}}-1}{h_i^2} \right) \prod_{j=1}^{i-1} 2^{2R^{\mathrm{C}}_{\mathbf{d},j}} \right], \tag{4.10a}$$

where

$$R^{\mathrm{C}}_{\mathbf{d},k} \triangleq \begin{cases} \frac{\binom{K-k}{\lfloor r \rfloor}}{\binom{K}{\lfloor r \rfloor}} \left(\lfloor r \rfloor + 1 - r \right) R + \frac{\binom{K-k}{\lfloor r \rfloor+1}}{\binom{K}{\lfloor r \rfloor+1}} \left(r - \lfloor r \rfloor \right) R, & \textit{if } k \in \mathcal{U}_{\mathbf{d}}, \\ \frac{\binom{K-k}{\lfloor r \rfloor} - \binom{K-k-N_{\mathbf{d},k}}{\lfloor r \rfloor}}{\binom{K}{\lfloor r \rfloor}} \left(\lfloor r \rfloor + 1 - r \right) R + \frac{\binom{K-k}{\lfloor r \rfloor+1} - \binom{K-k-N_{\mathbf{d},k}}{\lfloor r \rfloor+1}}{\binom{K}{\lfloor r \rfloor+1}} \left(r - \lfloor r \rfloor \right) R, & \textit{otherwise}, \end{cases} \tag{4.10b}$$

and

$$\hat{P}^*(R,M) \leq \hat{P}^{\mathrm{C}}_{\mathrm{UB}}(R,M) \triangleq \sum_{i=1}^{K} \left(\frac{2^{2\hat{R}^{\mathrm{C}}_{\mathbf{d},i}}-1}{h_i^2} \right) \prod_{j=1}^{i-1} 2^{2\hat{R}^{\mathrm{C}}_{\mathbf{d},j}}, \tag{4.11a}$$

where

$$\hat{R}^{\mathrm{C}}_{\mathbf{d},k} \triangleq \begin{cases} \frac{\binom{K-k}{\lfloor r \rfloor}}{\binom{K}{\lfloor r \rfloor}} \left(\lfloor r \rfloor + 1 - r \right) R + \frac{\binom{K-k}{\lfloor r \rfloor+1}}{\binom{K}{\lfloor r \rfloor+1}} \left(r - \lfloor r \rfloor \right) R, & \textit{if } k \in [\min\{N,K\}], \\ 0, & \textit{otherwise.} \end{cases} \tag{4.11b}$$

Proof. For simplicity, we assume that both nR and nMR are integers. The proposed scheme is first presented for integer normalized global cache capacities, that is $r \in [0 : K]$. The scheme is then extended to any $r \in [0, K]$.

4.4.1 Integer r Values

Here, we assume that $r \in [0:K]$. We remind that we denote the r-element subsets of K users by $\mathcal{S}^r_{[K],1}, \mathcal{S}^r_{[K],2}, \dots, \mathcal{S}^r_{[K],\binom{K}{r}}$.

Placement phase: A centralized cache placement phase is performed without the knowledge of the future user demands or the channel gains in the delivery phase. Each file W_i, $i \in [N]$, is split into $\binom{K}{r}$ equal-length subfiles $W_{i,\mathcal{S}^r_{[K],1}}, W_{i,\mathcal{S}^r_{[K],2}}, \dots, W_{i,\mathcal{S}^r_{[K],\binom{K}{r}}}$, each of rate $R/\binom{K}{r}$. User k, $k \in [K]$, caches subfile $W_{i,\mathcal{S}^r_{[K],l}}$, if $k \in \mathcal{S}^r_{[K],l}$, for $i \in [N]$ and $l \in \left[\binom{K}{r}\right]$. Hence, the cache contents of user k is given by

$$B_k = \bigcup\nolimits_{i \in [N]} \bigcup\nolimits_{l \in \left[\binom{K}{r}\right]: k \in \mathcal{S}^r_{[K],l}} W_{i,\mathcal{S}^r_{[K],l}}, \quad \text{for } k \in [K], \tag{4.12}$$

where the cache size constraint is satisfied with equality.

Delivery phase: For an arbitrary $\mathbf{d}$, we will deliver the following coded message to the users in $\mathcal{S}^{r+1}_{[K],l}$, $\forall l \in \left[\binom{K}{r+1}\right]$:

$$W_{\mathcal{S}^{r+1}_{[K],l}} \triangleq \bigoplus\nolimits_{k \in \mathcal{S}^{r+1}_{[K],l}} W_{d_k, \mathcal{S}^{r+1}_{[K],l} \backslash \{k\}}, \tag{4.13}$$

Then, each user $i \in \mathcal{S}^{r+1}_{[K],l}$ can recover subfile $W_{d_i,\mathcal{S}^{r+1}_{[K],l}\backslash\{i\}}$, since it has cached all the other subfiles $W_{d_j,\mathcal{S}^{r+1}_{[K],l}\backslash\{j\}}$, $\forall j \in \mathcal{S}^{r+1}_{[K],l}\backslash\{i\}$. Note that each coded message $W_{\mathcal{S}^{r+1}_{[K],l}}$, $l \in \left[\binom{K}{r+1}\right]$, is of rate $R/\binom{K}{r}$. Note also that, for $k \in [K]$, sending $\bigcup_{l:k\in\mathcal{S}^{r+1}_{[K],l}} W_{\mathcal{S}^{r+1}_{[K],l}}$ to user k enables that user to obtain all the subfiles $W_{d_k,\mathcal{S}^r_{[K],l}}$, $\forall l \in \left[\binom{K}{r}\right]$ and $k \notin \mathcal{S}^r_{[K],l}$. Thus, together with its cache contents, the demand of user k, $k \in [K]$ would be fully satisfied after receiving $\bigcup_{l:k\in\mathcal{S}^{K,r+1}_l} W_{\mathcal{S}^{r+1}_{[K],l}}$. As observed in [20], for a demand vector $\mathbf{d}$ with $N_{\mathbf{d}}$ distinct requests, if $K - N_{\mathbf{d}} \geq r+1$, not all the coded messages $W_{\mathcal{S}^{r+1}_{[K],l}}$, $\forall l \in \left[\binom{K}{r}\right]$, need to be delivered.

Following [20, Lemma 1], for a demand vector $\mathbf{d}$ with $N_{\mathbf{d}}$ distinct requests, let $\mathcal{U}_{\mathbf{d}} \subset \mathcal{V} \subset [K]$. We define $\mathcal{G}_{\mathcal{V}}$ as the set consisting of all the subsets of $\mathcal{V}$ with cardinality $N_{\mathbf{d}}$, such that all $N_{\mathbf{d}}$ users in each subset request distinct files. For any $\mathcal{V}$, we have $\bigoplus_{\mathcal{G}\in\mathcal{G}_{\mathcal{V}}} W_{\mathcal{V}\backslash\mathcal{G}} = \mathbf{0}$, where $\mathbf{0}$ denotes the zero vector.

Remark 4.4.1. Given a demand vector $\boldsymbol{d}$ with $N_{\boldsymbol{d}} < K$, and any set $\mathcal{S} \subset [K] \backslash \mathcal{U}_{\boldsymbol{d}}$ of users, by setting $\mathcal{V} = \mathcal{S} \cup \mathcal{U}_{\boldsymbol{d}}$, we have

$$\bigoplus\nolimits_{\mathcal{G} \in \mathcal{G}_{\mathcal{V}}} W_{\mathcal{V} \backslash \mathcal{G}} = \left(\bigoplus\nolimits_{\mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_d} W_{\mathcal{V} \backslash \mathcal{G}} \right) \oplus W_{\mathcal{V} \backslash \mathcal{U}_d} = \left(\bigoplus\nolimits_{\mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_d} W_{\mathcal{V} \backslash \mathcal{G}} \right) \oplus W_{\mathcal{S}} = \boldsymbol{0}, \tag{4.14}$$

which leads to

$$W_{\mathcal{S}} = \bigoplus\nolimits_{\mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_d} W_{\mathcal{V} \backslash \mathcal{G}}. \tag{4.15}$$

Thus, having received all the coded messages $W_{\mathcal{V} \backslash \mathcal{G}}$, $\forall \mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\boldsymbol{d}}$, $W_{\mathcal{S}}$ can be recovered through (4.15). Note that, for any $\mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\boldsymbol{d}}$, we have

$$\left| \mathcal{V} \backslash \mathcal{G} \right| = \left| \mathcal{S} \right|, \tag{4.16a}$$

$$(\mathcal{V} \backslash \mathcal{G}) \cap \mathcal{U}_{\boldsymbol{d}} \neq \emptyset, \tag{4.16b}$$

that is, each coded message on the right hand side (RHS) of (4.15) is targeted for a set of $|\mathcal{S}|$ users, at least one of which is in set $\mathcal{U}_{\boldsymbol{d}}$. Furthermore, for each $k \in \mathcal{S}$, there is a user $k' \in \mathcal{U}_{\boldsymbol{d}}$ with $h_{k'}^2 \leq h_k^2$, such that $d_{k'} = d_k$. Note that, since no two users with the same demand are in any of the sets $\mathcal{G} \in \mathcal{G}_{\mathcal{V}}$, for any set $\mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\boldsymbol{d}}$, we have either $k \in \mathcal{V} \backslash \mathcal{G}$ or $k' \in \mathcal{V} \backslash \mathcal{G}$.

Given a demand vector $\mathbf{d}$, the delivery phase is designed such that only the coded messages $W_{\mathcal{S}_{[K],l}^{r+1}}$, $\forall l \in \left[\binom{K}{r+1} \right]$ such that $\mathcal{S}_{[K],l}^{r+1} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$, are delivered, i.e., the coded messages that are targeted for at least one user in $\mathcal{U}_{\mathbf{d}}$ are delivered, and the remaining coded messages can be recovered through (4.15). To achieve this, for any such set $\mathcal{S}_{[K],l}^{r+1}$ with $\mathcal{S}_{[K],l}^{r+1} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$, the transmission power is adjusted such that the worst user in $\mathcal{S}_{[K],l}^{r+1}$ can decode $W_{\mathcal{S}_{[K],l}^{r+1}}$; and so can all the other users in $\mathcal{S}_{[K],l}^{r+1}$ due to the degradedness of the Gaussian BC. As a result, the demand of every user in $\mathcal{U}_{\mathbf{d}}$ will be satisfied.

We aim to find the coded packets targeted for each user that will minimize the transmitted power, while guaranteeing that all the user demands are satisfied. In delivering the coded messages, we start from the worst user, i.e., user 1, and first transmit all the coded messages targeted for user 1. We then target the second worst user, and transmit the coded messages targeted for it that have not been already delivered, keeping in mind that only the coded messages $W_{\mathcal{S}_{[K],l}^{r+1}}$ for which $\mathcal{S}_{[K],l}^{r+1} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$ are delivered. We continue similarly until we deliver the messages targeted for the best user in $\mathcal{U}_{\mathbf{d}}$.

We denote the contents targeted for user k by $\tilde{W}_k$, and their total rate by $R_{\mathbf{d},r,k}$, $k \in [K]$. For a demand vector $\mathbf{d}$, contents

$$\tilde{W}_1 = \bigcup\nolimits_{l: 1 \in \mathcal{S}_{[K],l}^{r+1}} W_{\mathcal{S}_{[K],l}^{r+1}} \tag{4.17}$$

are targeted for user 1. Note that there are $\binom{K-1}{r}$ different $(r+1)$-element subsets $\mathcal{S}_{[K],l}^{r+1}$, in which user 1 is included. Thus, the total rate of the messages targeted for user 1 is

$$R_{\mathbf{d},r,1} = \frac{\binom{K-1}{r}}{\binom{K}{r}} R = \frac{K-r}{K} R. \tag{4.18}$$

For $k \in [2:K]$, the coded contents targeted for user k that have not been sent through the transmissions to the previous $k-1$ users and are targeted for at least one user in $\mathcal{U}_{\mathbf{d}}$ are delivered. Thus, for $k \in [2:K]$, we deliver

$$\tilde{W}_k = \bigcup\nolimits_{l: \mathcal{S}_{[K],l}^{r+1} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, \mathcal{S}_{[K],l}^{r+1} \cap [k-1] = \emptyset, k \in \mathcal{S}_{[K],l}^{r+1}} W_{\mathcal{S}_{[K],l}^{r+1}}, \tag{4.19}$$

which is equivalent to

$$\begin{aligned}\tilde{W}_k = &\left(\bigcup\nolimits_{l: \mathcal{S}_{[K],l}^{r+1} \cap [k-1] = \emptyset, k \in \mathcal{S}_{[K],l}^{r+1}} W_{\mathcal{S}_{[K],l}^{r+1}} \right) - \\ &\left(\bigcup\nolimits_{l: \mathcal{S}_{[K],l}^{r+1} \cap \mathcal{U}_{\mathbf{d}} = \emptyset, \mathcal{S}_{[K],l}^{r+1} \cap [k-1] = \emptyset, k \in \mathcal{S}_{[K],l}^{r+1}} V_{\mathcal{S}_{[K],l}^{r+1}} \right).\end{aligned} \tag{4.20}$$

For each user $k \in [2:K]$, there are $\binom{K-k}{r}$ different $(r+1)$-element subsets $\mathcal{S}_{[K],l}^{r+1}$, such that $\mathcal{S}_{[K],l}^{r+1} \cap [k-1] = \emptyset$ and $k \in \mathcal{S}_{[K],l}^{r+1}$, for $l = 1, \dots, \binom{K}{r+1}$. On the other hand, for each user $k \in [2:K] \backslash \mathcal{U}_{\mathbf{d}}$, since there are $N_{\mathbf{d},k}$ users among the set of users $[k:K]$ that belong to set $\mathcal{U}_{\mathbf{d}}$, there are $\binom{K-k-N_{\mathbf{d},k}}{r}$ different $(r+1)$-element subsets $\mathcal{S}_{[K],l}^{r+1}$, such that $\mathcal{S}_{[K],l}^{r+1} \cap (\mathcal{U}_{\mathbf{d}} \cup [k-1]) = \emptyset$ and $k \in \mathcal{S}_{[K],l}^{r+1}$, for $l = 1, \dots, \binom{K}{r+1}$. Note that, if $k \in \mathcal{U}_{\mathbf{d}}$, the second term on the RHS of (4.20) includes no content. Thus, if $k \in [2:K] \cap \mathcal{U}_{\mathbf{d}}$,

Algorithm 7 Centralized Caching and Coded Packet Generation

```
1: procedure PLACEMENT PHASE
2:     $W_i = \bigcup_{l=1}^{\binom{K}{r}} W_{i,\mathcal{S}^r_{[K],l}}$,    for $i = 1, ..., N$
3:     $U_k = \bigcup_{i \in [N]} \bigcup_{l \in [\binom{K}{r}]: k \in \mathcal{S}^r_{[K],l}} W_{i,\mathcal{S}^r_{[K],l}}$,    for $k = 1, \dots, K$
4: end procedure
5: procedure CODED PACKET GENERATION
6:     $W_{\mathcal{S}^{r+1}_{[K],l}} = \bigoplus_{k \in \mathcal{S}^{r+1}_{[K],l}} W_{d_k, \mathcal{S}^{r+1}_{[K],l} \backslash \{k\}}$,    for $l = 1, ..., \binom{K}{r+1}$
7:     $\tilde{W}_k = \bigcup_{l: \mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, \mathcal{S}^{r+1}_{[K],l} \cap [k-1] = \emptyset, k \in \mathcal{S}^{r+1}_{[K],l}} W_{\mathcal{S}^{r+1}_{[K],l}}$,    for $k = 1, \dots, K$
8: end procedure
```

total rate targeted for user k is

$$R_{\mathbf{d},r,k} = \frac{\binom{K-k}{r}}{\binom{K}{r}} R = \left(\prod_{i=0}^{k-1} \frac{K-r-i}{K-i} \right) R, \tag{4.21}$$

while the total rate targeted for user k, $k \in [2:K] \backslash \mathcal{U}_{\mathbf{d}}$, is

$$R_{\mathbf{d},r,k} = \frac{\binom{K-k}{r} - \binom{K-k-N_{\mathbf{d},k}}{r+1}}{\binom{K}{r}} R. \tag{4.22}$$

In summary, the proposed achievable scheme intends to deliver contents of total rate $R_{\mathbf{d},r,k}$ to user k, for $k \in [K]$, where

$$R_{\mathbf{d},r,k} \triangleq \begin{cases} \frac{\binom{K-k}{r}}{\binom{K}{r}} R, & \text{if } k \in \mathcal{U}_{\mathbf{d}}, \\ \frac{\binom{K-k}{r} - \binom{K-k-N_{\mathbf{d},k}}{r}}{\binom{K}{r}} R, & \text{otherwise.} \end{cases} \tag{4.23}$$

The centralized caching and coded packet generation explained above is summarized in Algorithm 7.

Once the coded packets targeted to each user are determined, the next step is to design the physical layer coding scheme to deliver these packets over the Gaussian BC. Given $\mathbf{d}$, we generate K codebooks, where the codebook $k \in [K]$ is designed to deliver the contents $\tilde{W}_k$ to user k. The k-th codebook consists of $2^{nR_{\mathbf{d},r,k}}$ i.i.d. Gaussian codewords $x_k^n(\mathbf{W}, \mathbf{d})$, generated according to the normal distribution $\mathcal{N}\left(0, \alpha_k P^{\mathrm{C}}_{\mathrm{UB}}(R, M, \mathbf{d})\right)$, where $\alpha_k \geq 0$ and $\sum_{i=1}^{K} \alpha_i = 1$. The transmission is performed through superposition coding; that is, the channel input is given by $\sum_{k=1}^{K} x_k^n \left(\mathbf{W}, \mathbf{d}, \tilde{q}_k\right)$, where $\tilde{q}_k \in \left[2^{nR^{\mathrm{C}}_{\mathbf{d},r,k}}\right]$. User k decodes codewords $x_1^n \left(\mathbf{W}, \mathbf{d}\right), ..., x_k^n \left(\mathbf{W}, \mathbf{d}\right)$ by successive decoding, considering all the codewords in higher levels as noise. If all the k codewords are decoded successfully, user $k \in [K]$ can recover contents $\tilde{W}_1, ..., \tilde{W}_k$. Accordingly, any user $k \in [K]$ will receive all the coded contents targeted for it except those that are not intended for at least one user in $\mathcal{U}_{\mathbf{d}}$ (which have not been delivered); that is, user k receives all the coded contents

$$\bigcup\nolimits_{l: \mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, k \in \mathcal{S}^{r+1}_{[K],l}} W_{\mathcal{S}^{r+1}_{[K],l}}. \tag{4.24}$$

Thanks to the degradedness of the underlying Gaussian BC, it can also obtain all the coded contents targeted for users in $[k-1]$, which are also intended for at least one user in $\mathcal{U}_{\mathbf{d}}$, i.e., all the coded contents in

$$\bigcup\nolimits_{l: \mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, [k-1] \cap \mathcal{S}^{r+1}_{[K],l} \neq \emptyset} W_{\mathcal{S}^{r+1}_{[K],l}}. \tag{4.25}$$

Note that, if $k \in \mathcal{U}_{\mathbf{d}}$, (4.24) reduces to $\bigcup_{l: k \in \mathcal{S}^{r+1}_{[K],l}} W_{\mathcal{S}^{r+1}_{[K],l}}$, which shows that the demand of user $k \in \mathcal{U}_{\mathbf{d}}$ is satisfied. Next, we illustrate that the users in $[K] \backslash \mathcal{U}_{\mathbf{d}}$ can obtain their requests without being delivered any extra messages. Given any set of users $\mathcal{S}^{r+1}_{[K],l}$ such that $\mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} = \emptyset$, we need to show that every user in $\mathcal{S}^{r+1}_{[K],l}$ can decode all the coded messages $W_{\mathcal{V} \backslash \mathcal{G}}$, $\forall \mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\mathbf{d}}$, where $\mathcal{V} = \mathcal{S}^{r+1}_{[K],l} \cup \mathcal{U}_{\mathbf{d}}$. In this case, they can also decode $W_{\mathcal{S}^{r+1}_{[K],l}}$ through (4.15).

Assume that there exists a set of users $\mathcal{S}^{r+1}_{[K],l}$ such that $\mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} = \emptyset$; set $\mathcal{V} = \mathcal{S}^{r+1}_{[K],l} \cup \mathcal{U}_{\mathbf{d}}$. According to (4.16b), there is at least one user in $\mathcal{U}_{\mathbf{d}}$ in any set of users $\mathcal{V} \backslash \mathcal{G}$, $\forall \mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\mathbf{d}}$. Thus, all the coded messages $W_{\mathcal{V} \backslash \mathcal{G}}$ have been delivered by the proposed delivery scheme. Remember that, for each user $k \in \mathcal{S}^{r+1}_{[K],l}$ and $\forall \mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\mathbf{d}}$, either $k \in \mathcal{V} \backslash \mathcal{G}$ or $\exists k' \in \mathcal{V} \backslash \mathcal{G}$, where $d_{k'} = d_k$ and $k' \in \mathcal{U}_{\mathbf{d}}$, i.e., $h^2_{k'} \leq h^2_k$. If $k \in \mathcal{V} \backslash \mathcal{G}$, since $\mathcal{V} \backslash \mathcal{G} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$, according to (4.24), user k can obtain $W_{\mathcal{V} \backslash \mathcal{G}}$. If $\exists k' \in \mathcal{V} \backslash \mathcal{G}$ with $d_{k'} = d_k$ and $k' \in \mathcal{U}_{\mathbf{d}}$, then user k' can decode $W_{\mathcal{V} \backslash \mathcal{G}}$, and since $h^2_{k'} \leq h^2_k$, user k can also decode $W_{\mathcal{V} \backslash \mathcal{G}}$. Thus, each user $k \in \mathcal{S}^{r+1}_{[K],l}$ can decode $W_{\mathcal{S}^{r+1}_{[K],l}}$ successfully, for any set of users $\mathcal{S}^{r+1}_{[K],l}$ that satisfies $\mathcal{S}^{r+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} = \emptyset$. This fact confirms that the demands of all the users in $[K] \backslash \mathcal{U}_{\mathbf{d}}$ are satisfied.

We remark here that, with the proposed coded delivery scheme, the coded packets targeted to user k are delivered only to users in $[k]$, and not to the any of the users in $[k+1:K]$, for $k \in [K]$. Next, we provide an example to illustrate the proposed joint cache and channel coding scheme.

Example: Consider $K = 5$ users, each equipped with a cache of normalized size $M = N/5$, i.e., $r = 1$. File W_i is partitioned into 5 equal-length subfiles $W_{i,\{1\}}$, ..., $W_{i,\{5\}}$, each of which is of rate $R/5$, for $i \in [N]$. Cache contents of user j after the placement phase is given by

$$B_j = \bigcup\nolimits_{i \in [N]} W_{i,\{j\}}, \quad j \in [5]. \tag{4.26}$$

Assuming that $N \geq 3$, let the user demands be $\mathbf{d} = (1,2,1,1,3)$, where we have $N_{\mathbf{d}} = 3$, and $\mathcal{U}_{\mathbf{d}} = \{1,2,5\}$. We generate the following coded packets:

$$W_{\{1,2\}} = W_{1,\{2\}} \oplus W_{2,\{1\}}, \tag{4.27a}$$
$$W_{\{1,3\}} = W_{1,\{3\}} \oplus W_{1,\{1\}}, \tag{4.27b}$$
$$W_{\{1,4\}} = W_{1,\{4\}} \oplus W_{1,\{1\}}, \tag{4.27c}$$
$$W_{\{1,5\}} = W_{1,\{5\}} \oplus W_{3,\{1\}}, \tag{4.27d}$$
$$W_{\{2,3\}} = W_{2,\{3\}} \oplus W_{1,\{2\}}, \tag{4.27e}$$
$$W_{\{2,4\}} = W_{2,\{4\}} \oplus W_{1,\{2\}}, \tag{4.27f}$$
$$W_{\{2,5\}} = W_{2,\{5\}} \oplus W_{3,\{2\}}, \tag{4.27g}$$
$$W_{\{3,5\}} = W_{1,\{5\}} \oplus W_{3,\{3\}}, \tag{4.27h}$$
$$W_{\{4,5\}} = W_{1,\{5\}} \oplus W_{3,\{4\}}. \tag{4.27i}$$

The coded contents $\tilde{W}_k$ targeted to user k, $k \in [5]$, are:

$$\tilde{W}_1 = W_{\{1,2\}}, W_{\{1,3\}}, W_{\{1,4\}}, W_{\{1,5\}}, \tag{4.28a}$$
$$\tilde{W}_2 = W_{\{2,3\}}, W_{\{2,4\}}, W_{\{2,5\}}, \tag{4.28b}$$
$$\tilde{W}_3 = W_{\{3,5\}}, \tag{4.28c}$$
$$\tilde{W}_4 = W_{\{4,5\}}, \tag{4.28d}$$
$$\tilde{W}_5 = \mathbf{0}. \tag{4.28e}$$

We perform superposition coding to deliver these contents, and send $\sum_{k=1}^{4} x_k^n \left(\mathbf{W}, \mathbf{d}, \tilde{q}_k\right)$, where $\tilde{q}_1 \in \left[2^{4nR/5}\right]$, $\tilde{q}_2 \in \left[2^{3nR/5}\right]$, $\tilde{q}_3 \in \left[2^{nR/5}\right]$, and $\tilde{q}_4 \in \left[2^{nR/5}\right]$. User k, $k \in [5]$, decodes the codewords $x_1^n \left(\mathbf{W}, \mathbf{d}\right), ..., x_k^n \left(\mathbf{W}, \mathbf{d}\right)$ by successive decoding, considering the higher level codewords as noise. If successful, user k can recover $\tilde{W}_1, \dots, \tilde{W}_k$, $k \in [5]$. Now, note that user 1 can recover W_1 from $\tilde{W}_1$ together with its cache contents; user 2 can decode W_2 having received the coded packets $W_{\{1,2\}}$, $W_{\{2,3\}}$, $W_{\{2,4\}}$, and $W_{\{2,5\}}$ along with its cache contents; coded packets $W_{\{1,5\}}$, $W_{\{2,5\}}$, $W_{\{3,5\}}$, and $W_{\{4,5\}}$, and its cache contents B_5 enable user 5 to decode W_3; user 3 can directly obtain $W_{\{1,3\}}$, $W_{\{2,3\}}$, and $W_{\{3,5\}}$, and it can generate $W_{\{3,4\}}$ as follows:

$$W_{\{3,4\}} = W_{\{1,3\}} \oplus W_{\{1,4\}}, \tag{4.29}$$

so it can decode W_1 together with its cache contents. Note that both $W_{\{1,3\}}$ and $W_{\{1,4\}}$ are delivered within $\tilde{W}_1$, which is decoded by user 3. Similarly, user 4 will be delivered $W_{\{1,4\}}$, $W_{\{2,4\}}$, and $W_{\{4,5\}}$, and it can also recover $W_{\{3,4\}}$ through (4.29), and together with its cache contents it can recover W_1. □

4.4.2 Non-integer r Values

Here we extend the proposed scheme to non-integer r values. We divide the whole database as well as the cache memory of the users into two, such that the corresponding r parameters for both parts are integer. This way we can employ the placement and delivery schemes introduced in Section 4.4.1 for each part separately.

Placement phase: Each file W_i, for $i = 1, ..., N$, is divided into two non-overlapping subfiles, W_i^1 of rate R_1 and W_i^2 of rate R_2. We set

$$R_1 = \left(\lfloor r \rfloor + 1 - r\right) R, \tag{4.30a}$$

$$R_2 = \left(r - \lfloor r \rfloor\right) R, \tag{4.30b}$$

such that $R_1 + R_2 = R$. For subfiles $\left\{W_1^1, \dots, W_N^1\right\}$, we perform the placement phase proposed in Section 4.4.1 corresponding to the normalized global cache size $\lfloor r \rfloor$, which requires a cache size of $n \left(\lfloor r \rfloor N/K\right) R_1$ bits. While, for subfiles $\left\{W_1^2, \dots, W_N^2\right\}$, we perform the placement phase proposed in Section 4.4.1 corresponding to the normalized global cache size $\lfloor r \rfloor + 1$, which requires a cache size of $n\left(\left(\lfloor r \rfloor + 1\right) N/K\right) R_2$ bits. By summing up, the total cache size is found to be nMR bits, which shows that the cache size constraint is satisfied.

Delivery phase: For any demand vector $\mathbf{d}$, we perform the proposed coded delivery phase in Section 4.4.1 corresponding to the normalized global cache size $\lfloor r \rfloor$ to deliver the missing bits of subfiles $\left\{W_1^1, \dots, W_N^1\right\}$ to the intended users. Moreover, the missing bits of subfiles $\left\{W_1^2, \dots, W_N^2\right\}$ are delivered to the intended users by performing the coded delivery scheme proposed in Section 4.4.1 corresponding to the normalized global cache size $\lfloor r \rfloor + 1$.

For an arbitrary demand vector $\mathbf{d} = (d_1, \dots, d_K)$, we define

$$W^i_{\mathcal{S}^{r_i+1}_{[K],l}} \triangleq \bigoplus\nolimits_{k \in \mathcal{S}^{r_i+1}_{[K],l}} W^i_{d_k, \mathcal{S}^{r_i+1}_{[K],l} \backslash \{k\}}, \quad \text{for } i = 1, 2, \tag{4.31}$$

where $r_1 \triangleq \lfloor r \rfloor$ and $r_2 \triangleq \lfloor r \rfloor + 1$. According to Algorithm 7, by performing the centralized caching and coded delivery scheme proposed in Section 4.4.1 for the normalized global cache size $\lfloor r \rfloor$ to serve the subfiles $\left\{W_1^1, \dots, W_N^1\right\}$, each of rate R_1, user $k \in [K]$ should receive

$$\tilde{W}_k^1 = \bigcup\nolimits_{l: \mathcal{S}^{\lfloor r \rfloor+1}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, \mathcal{S}^{\lfloor r \rfloor+1}_{[K],l} \cap [k-1] = \emptyset, k \in \mathcal{S}^{\lfloor r \rfloor+1}_{[K],l}} W^1_{\mathcal{S}^{\lfloor r \rfloor+1}_{[K],l}}, \tag{4.32}$$

of the following total rate obtained according to (4.23)

$$R_{\mathbf{d},\lfloor r \rfloor,k} = \begin{cases} \frac{\binom{K-k}{\lfloor r \rfloor}}{\binom{K}{\lfloor r \rfloor}} R_1, & \text{if } k \in \mathcal{U}_{\mathbf{d}}, \\ \frac{\binom{K-k}{\lfloor r \rfloor} - \binom{K-k-N_{\mathbf{d},k}}{\lfloor r \rfloor}}{\binom{K}{\lfloor r \rfloor}} R_1, & \text{otherwise.} \end{cases} \tag{4.33}$$

Similarly, to serve the subfiles $\left\{W_1^2, \dots, W_N^2\right\}$, each of rate R_2, user $k \in [K]$ should receive

$$\tilde{W}_k^2 = \bigcup\nolimits_{l: \mathcal{S}^{\lfloor r \rfloor+2}_{[K],l} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, \mathcal{S}^{\lfloor r \rfloor+2}_{[K],l} \cap [k-1] = \emptyset, k \in \mathcal{S}^{\lfloor r \rfloor+2}_{[K],l}} W^2_{\mathcal{S}^{\lfloor r \rfloor+2}_{[K],l}}, \tag{4.34}$$

of total rate

$$R_{\mathbf{d},\lfloor r\rfloor+1,k}=\begin{cases}\dfrac{\binom{K-k}{\lfloor r\rfloor+1}}{\binom{K}{\lfloor r\rfloor+1}}R_2, & \text{if } k\in\mathcal{U}_{\mathbf{d}},\\ \dfrac{\binom{K-k}{\lfloor r\rfloor+1}-\binom{K-k-N_{\mathbf{d},k}}{\lfloor r\rfloor+1}}{\binom{K}{\lfloor r\rfloor+1}}R_2, & \text{otherwise.}\end{cases} \tag{4.35}$$

Thus, $\tilde{W}_k\triangleq\left(\tilde{W}_k^1,\tilde{W}_k^2\right)$, at a total rate of $R^{\mathrm{S}}_{\mathbf{d},k}\triangleq R_{\mathbf{d},\lfloor r\rfloor,k}+R_{\mathbf{d},\lfloor r\rfloor+1,k}$ given by

$$R^{\mathrm{S}}_{\mathbf{d},k}=\begin{cases}\dfrac{\binom{K-k}{\lfloor r\rfloor}}{\binom{K}{\lfloor r\rfloor}}\left(\lfloor r\rfloor+1-r\right)R+\dfrac{\binom{K-k}{\lfloor r\rfloor+1}}{\binom{K}{\lfloor r\rfloor+1}}\left(r-\lfloor r\rfloor\right)R, & \text{if } k\in\mathcal{U}_{\mathbf{d}},\\ \dfrac{\binom{K-k}{\lfloor r\rfloor}-\binom{K-k-N_{\mathbf{d},k}}{\lfloor r\rfloor}}{\binom{K}{\lfloor r\rfloor}}\left(\lfloor r\rfloor+1-r\right)R+\dfrac{\binom{K-k}{\lfloor r\rfloor+1}-\binom{K-k-N_{\mathbf{d},k}}{\lfloor r\rfloor+1}}{\binom{K}{\lfloor r\rfloor+1}}\left(r-\lfloor r\rfloor\right)R, & \text{otherwise}\end{cases} \tag{4.36}$$

is delivered to user k, for $k=1,...,K$.

4.4.3 Transmit Power Analysis

In the proposed delivery scheme, user $k\in[K]$ decodes codewords $x_1^n\left(\mathbf{W},\mathbf{d}\right),\dots,$ $x_k^n\left(\mathbf{W},\mathbf{d}\right)$ successively considering all the other codewords in higher levels as noise. User k can decode its intended message successfully if, for $k\in[K]$,

$$R^{\mathrm{C}}_{\mathbf{d},k}\leq\frac{1}{2}\log_2\left(1+\frac{\alpha_k h_k^2 P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)}{h_k^2\sum_{i=k+1}^{K}\alpha_i P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)+1}\right). \tag{4.37}$$

From Proposition 4.3.1, the corresponding minimum required power is given by

$$P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)\triangleq\sum\nolimits_{i=1}^{K}\left(\frac{2^{2R^{\mathrm{C}}_{\mathbf{d},i}}-1}{h_i^2}\right)\prod\nolimits_{j=1}^{i-1}2^{2R^{\mathrm{C}}_{\mathbf{d},j}}. \tag{4.38}$$

Thus, the average and peak power-memory trade-offs of the proposed achievable scheme are given by $\mathrm{E}_{\mathbf{d}}\left[P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)\right]=\bar{P}^{\mathrm{C}}_{\mathrm{UB}}(R,M)$ and $\hat{P}^{\mathrm{C}}_{\mathrm{UB}}\left(R,M\right)=\max_{\mathbf{d}}\left\{P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)\right\}$ as stated in Theorem 4.1, respectively, where the demands are distributed uniformly.

Observe that for demand vectors with the same $\mathcal{U}_{\mathbf{d}}$ set the required power $P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)$ is the same. Let $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ denote the set of all demand vectors with the same $\mathcal{U}_{\mathbf{d}}$ set. We define $P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}\right)$ as the required power $P^{\mathrm{C}}_{\mathrm{UB}}\left(R,M,\mathbf{d}\right)$ for any demand vector

$\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$. Thus, we have

$$\hat{P}_{\text{UB}}^{\text{C}}(R, M) = \max_{\mathcal{U}_{\mathbf{d}}} \left\{ P_{\text{UB}}^{\text{C}}(R, M, \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}) \right\}. \tag{4.39}$$

It is shown in [94, Appendix B] that the worst-case demand combination for the proposed centralized caching scheme happens when the first $\min\{N, K\}$ users; that is, the users with the worst channel gains, request distinct files, i.e., when $\mathcal{U}_{\mathbf{d}} = [\min\{N, K\}]$, and $\hat{P}_{\text{UB}}^{\text{C}}(R, M)$ is given by (4.11). $\square$

4.5 Decentralized Caching and Delivery

Here we extend our centralized caching scheme to the decentralized caching. The corresponding average and peak power-memory trade-offs are given in the following theorem.

Theorem 4.2. *For decentralized caching followed by delivery over a Gaussian BC, we have*

$$\bar{P}^*(R, M) \le \bar{P}_{\text{UB}}^{\text{D}}(R, M) \triangleq \frac{1}{N^K} \sum_{\mathbf{d} \in [N]^K} \left[\sum_{i=1}^{K} \left(\frac{2^{2R_{\mathbf{d},i}^{\text{D}}} - 1}{h_i^2} \right) \prod_{j=1}^{i-1} 2^{2R_{\mathbf{d},j}^{\text{D}}} \right], \tag{4.40a}$$

where, for $k = 1, ..., K$,

$$R_{\mathbf{d},k}^{\text{D}} \triangleq \begin{cases} \left(1 - \frac{M}{N}\right)^k R, & \text{if } k \in \mathcal{U}_{\mathbf{d}}, \\ \left(1 - \frac{M}{N}\right)^k \left(1 - \left(1 - \frac{M}{N}\right)^{N_{\mathbf{d},k}}\right) R, & \text{otherwise}, \end{cases} \tag{4.40b}$$

and

$$\hat{P}^*(R, M) \le \hat{P}_{\text{UB}}^{\text{D}}(R, M) \triangleq \sum_{i=1}^{\min\{N,K\}} \left(\frac{2^{2R\left(1 - \frac{M}{N}\right)^i} - 1}{h_i^2} \right) 2^{2R\left(\frac{N}{M} - 1\right)\left(1 - \left(1 - \frac{M}{N}\right)^{i-1}\right)}. \tag{4.41}$$

Proof. The decentralized caching scheme to achieve the average and peak power-memory trade-offs outlined in Theorem 4.2 is described in the following.

4.5.1 Placement Phase

We perform decentralized uncoded cache placement [26], where each user caches nMR/N random bits of each file of length nR bits independently. Since there are a total of N files, the cache size constraint is satisfied. The part of file i cached exclusively by the users in set $\mathcal{S} \subset [K]$ is denoted by $W_{i,\mathcal{S}}$, for $i \in [N]$. For n large enough, the rate of $W_{i,\mathcal{S}}$ can be approximated by $\left(\frac{M}{N}\right)^{|\mathcal{S}|}\left(1-\frac{M}{N}\right)^{K-|\mathcal{S}|}R$. The cache contents at user j is given by

$$B_j = \bigcup\nolimits_{i\in[N]} \bigcup\nolimits_{\mathcal{S}\subset[K]:j\in\mathcal{S}} W_{i,\mathcal{S}}. \tag{4.42}$$

4.5.2 Delivery phase

Consider any non-empty set of users $\mathcal{S} \subset [K]$. For a demand vector $\mathbf{d}$, by delivering the coded message $W_{\mathcal{S}} = \bigoplus_{k\in\mathcal{S}} W_{d_k,\mathcal{S}\backslash\{k\}}$ of rate $(M/N)^{|\mathcal{S}|-1}(1-M/N)^{K-|\mathcal{S}|+1}R$ to users in $\mathcal{S}$, each user $i \in \mathcal{S}$ can recover subfile $W_{d_i,\mathcal{S}\backslash\{i\}}$ since it has cached all the subfiles $W_{d_j,\mathcal{S}\backslash\{j\}}$, $\forall j \in \mathcal{S}\backslash\{i\}$. For each $k \in [K]$, delivering $\bigcup_{\mathcal{S}\subset[K]:k\in\mathcal{S}} W_{\mathcal{S}}$ enables user k to recover all the subfiles $W_{d_k,\mathcal{S}\backslash\{k\}}$, $\forall\mathcal{S} \subset [K]$ and $k \in \mathcal{S}$. The demand of user k, $k \in [K]$, is fully satisfied after receiving $\bigcup_{\mathcal{S}\subset[K]:k\in\mathcal{S}} W_{\mathcal{S}}$ along with its cache contents.

Similar to the proposed scheme for the centralized caching scenario, given a demand vector $\mathbf{d}$, the delivery phase is designed such that only the coded messages $W_{\mathcal{S}}$, $\forall\mathcal{S} \subset [K]$ that satisfy $\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$, are delivered, and the remaining coded messages can be recovered through (4.15). To achieve this, for any such set $\mathcal{S}$ with $\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset$, the transmission power is adjusted such that the worst user in $\mathcal{S}$ can decode it; and so can all the other users in $\mathcal{S}$ due to the degradedness of the Gaussian BC. Therefore, the demand of every user in $\mathcal{U}_{\mathbf{d}}$ is satisfied.

Note that in the centralized scenario described in Section 4.4.1, each coded packet $W_{\mathcal{S}_l^{K,r+1}}$ is targeted for a $(r+1)$-element subset of users, where $r \in [0:K]$, for $l \in \left[\binom{K}{r+1}\right]$. While, in the decentralized scenario, coded packets are targeted for any subset of users, i.e., for $(r+1)$-element subset of users, $\forall r \in [0:K]$. By applying a similar technique as the delivery phase outlined in Algorithm 7, given a demand vector

$\mathbf{d}$, the following contents are targeted for user k, $k \in [K]$:

$$\tilde{W}_k = \bigcup\nolimits_{\mathcal{S} \subset [k:K]: \mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, k \in \mathcal{S}} W_{\mathcal{S}}, \tag{4.43}$$

which are equivalent to

$$\tilde{W}_k = \bigcup\nolimits_{\mathcal{S} \subset [k:K]: k \in \mathcal{S}} W_{\mathcal{S}} - \bigcup\nolimits_{\mathcal{S} \subset [k:K] \backslash \mathcal{U}_{\mathbf{d},k}, k \in \mathcal{S}} W_{\mathcal{S}}. \tag{4.44}$$

For each user k, $k \in [K]$, there are $\binom{K-k}{i}$ different $(i+1)$-element subsets of $[k:K]$, which include k, for $i \in [0:K-k]$. Thus, if $k \in \mathcal{U}_{\mathbf{d}}$, from (4.44), the total rate targeted for user k is

$$R^{\mathrm{D}}_{\mathbf{d},k} = \sum_{i=0}^{K-k} \binom{K-k}{i} \left(\frac{M}{N}\right)^i \left(1 - \frac{M}{N}\right)^{K-i} R = \left(1 - \frac{M}{N}\right)^k R. \tag{4.45}$$

On the other hand, for each user k, $k \in [K]$, there are $\binom{K-k-N_{\mathbf{d},k}}{i}$ different $(i+1)$-element subsets of $[k:K] \backslash \mathcal{U}_{\mathbf{d},k}$, which include k, for $i \in [0:K-k]$. Thus, if $k \notin \mathcal{U}_{\mathbf{d}}$, from (4.44), the total rate targeted for user $k \in [K]$ is given by

$$\begin{aligned} R^{\mathrm{D}}_{\mathbf{d},k} =& \sum\nolimits_{i=0}^{K-k} \binom{K-k}{i} \left(\frac{M}{N}\right)^i \left(1 - \frac{M}{N}\right)^{K-i} R \\ & - \sum\nolimits_{i=0}^{K-k-N_{\mathbf{d},k}} \binom{K-k-N_{\mathbf{d},k}}{i} \left(\frac{M}{N}\right)^i \left(1 - \frac{M}{N}\right)^{K-i} R \\ =& \left(1 - \frac{M}{N}\right)^k \left(1 - \left(1 - \frac{M}{N}\right)^{N_{\mathbf{d},k}}\right) R. \end{aligned} \tag{4.46}$$

In total, the rate of contents targeted for user k, $k \in [K]$, is given by

$$R^{\mathrm{D}}_{\mathbf{d},k} = \begin{cases} \left(1 - \frac{M}{N}\right)^k R, & \text{if } k \in \mathcal{U}_{\mathbf{d}}, \\ \left(1 - \frac{M}{N}\right)^k \left(1 - \left(1 - \frac{M}{N}\right)^{N_{\mathbf{d},k}}\right) R, & \text{otherwise.} \end{cases} \tag{4.47}$$

Given a demand vector $\mathbf{d}$, the transmitted codeword $x^n(\mathbf{W}, \mathbf{d})$ is generated as the linear superposition of K codewords $x_1^n(\mathbf{W}, \mathbf{d}), ..., x_K^n(\mathbf{W}, \mathbf{d})$, each chosen from an independent codebook. Codebook k consists of $2^{nR^{\mathrm{D}}_{\mathbf{d},k}}$ i.i.d. codewords $x_k^n(\mathbf{W}, \mathbf{d})$ generated according to the normal distribution $\mathcal{N}\left(0, \alpha_k P^{\mathrm{D}}_{\mathrm{UB}}(R, M, \mathbf{d})\right)$, where $\alpha_k \geq 0$ and $\sum_{i=1}^{K} \alpha_i = 1$, which satisfy the power constraint, for $k \in [K]$.

User k, $k \in [K]$, decodes codewords $x_1^n(\mathbf{W}, \mathbf{d}), ..., x_k^n(\mathbf{W}, \mathbf{d})$ through successive decoding, while considering all the codewords $x_{k+1}^n(\mathbf{W}, \mathbf{d}), ..., x_K^n(\mathbf{W}, \mathbf{d})$ as noise. Thus, if user $k \in [K]$ can successfully decode all the k channel codewords intended for it, it can then recover the contents $\tilde{W}_1, ..., \tilde{W}_k$, from which it can obtain all the coded contents $\bigcup_{\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, [k] \cap \mathcal{S} \neq \emptyset} W_{\mathcal{S}}$. Accordingly, each user $k \in [K]$ can obtain all the coded contents targeted for it except those that are not intended for at least one user in $\mathcal{U}_{\mathbf{d}}$ (which have not been delivered), i.e., all the coded contents

$$\bigcup\nolimits_{\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, k \in \mathcal{S}} W_{\mathcal{S}}. \tag{4.48}$$

It can further obtain all the coded contents targeted for users $[k-1]$, which are also intended for at least one user in $\mathcal{U}_{\mathbf{d}}$, i.e., all the coded contents $\bigcup_{\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} \neq \emptyset, [k-1] \cap \mathcal{S} \neq \emptyset} W_{\mathcal{S}}$. Note that, if $k \in \mathcal{U}_{\mathbf{d}}$, (4.48) reduces to $\bigcup_{k \in \mathcal{S}} W_{\mathcal{S}}$, which shows that the demand of each user $k \in \mathcal{U}_{\mathbf{d}}$ is satisfied through the proposed delivery scheme. Next, we illustrate that the users in $[K] \backslash \mathcal{U}_{\mathbf{d}}$ can obtain their requested files without being delivered any extra messages. Similarly to the centralized scenario, given any set of users $\mathcal{S}$ such that $\mathcal{S} \cap \mathcal{U}_{\mathbf{d}} = \emptyset$, by setting $\mathcal{V} = \mathcal{S} \cup \mathcal{U}_{\mathbf{d}}$, from the fact that, for each user $k \in \mathcal{S}$, $k \in \mathcal{V} \backslash \mathcal{G}$ or $k' \in \mathcal{V} \backslash \mathcal{G}$, where $k' < k$, it can be illustrated that every user in $\mathcal{S}$ can decode all coded contents $W_{\mathcal{V} \backslash \mathcal{G}}$, $\forall \mathcal{G} \in \mathcal{G}_{\mathcal{V}} \backslash \mathcal{U}_{\mathbf{d}}$. In this case, they all can also decode $W_{\mathcal{S}}$ through (4.15).

4.5.3 Transmit Power Analysis

For a demand vector $\mathbf{d}$, user k can decode the channel codewords up to level k successfully, considering all the other codewords in higher levels as noise, if

$$R_{\mathbf{d},k}^{\mathrm{D}} \leq \frac{1}{2} \log_2 \left(1 + \frac{\alpha_k h_k^2 P_{\mathrm{UB}}^{\mathrm{D}}(R, M, \mathbf{d})}{h_k^2 \sum_{i=k+1}^{K} \alpha_i P_{\mathrm{UB}}^{\mathrm{D}}(R, M, \mathbf{d}) + 1} \right), \quad \text{for } k \in [K]. \tag{4.49}$$

From Proposition 4.3.1, the minimum required power is given by

$$P_{\mathrm{UB}}^{\mathrm{D}}(R, M, \mathbf{d}) \triangleq \sum\nolimits_{i=1}^{K} \left(\frac{2^{2R_{\mathbf{d},i}^{\mathrm{D}}} - 1}{h_i^2} \right) \prod\nolimits_{j=1}^{i-1} 2^{2R_{\mathbf{d},j}^{\mathrm{D}}}. \tag{4.50}$$

Thus, the average power-memory trade-off for the proposed decentralized caching scheme is given by $\mathrm{E}_{\mathbf{d}} \left[P_{\mathrm{UB}}^{\mathrm{D}}(R, M, \mathbf{d}) \right] = \bar{P}_{\mathrm{UB}}^{\mathrm{D}}(R, M)$ stated in Theorem 4.2.

With the proposed decentralized caching scheme, the peak power $\hat{P}^{\mathrm{D}}(R,M) = \max_{\mathbf{d}} \left\{ P_{\mathrm{UB}}^{\mathrm{D}}(R,M,\mathbf{d}) \right\}$ can be achieved. Observe that, for demand vectors with the same set of users $\mathcal{U}_{\mathbf{d}}$, the required power $P_{\mathrm{UB}}^{\mathrm{D}}(R,M,\mathbf{d})$ is the same. We define $P_{\mathrm{UB}}^{\mathrm{D}}(R,M,\mathcal{D}_{\mathcal{U}_{\mathbf{d}}})$ as the required power $P_{\mathrm{UB}}^{\mathrm{D}}(R,M,\mathbf{d})$ for any demand vector $\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$. Thus, we have

$$\hat{P}^{\mathrm{D}}(R,M) = \max_{\mathcal{U}_{\mathbf{d}}} \left\{ P_{\mathrm{UB}}^{\mathrm{D}}(R,M,\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}) \right\}. \tag{4.51}$$

It is shown in [94, Appendix C] that the worst-case demand combination happens when $\mathcal{U}_{\mathbf{d}} = [\min\{N,K\}]$, and $\hat{P}_{\mathrm{UB}}^{\mathrm{D}}(R,M)$ is found as in (4.41). $\square$

4.6 Lower Bound

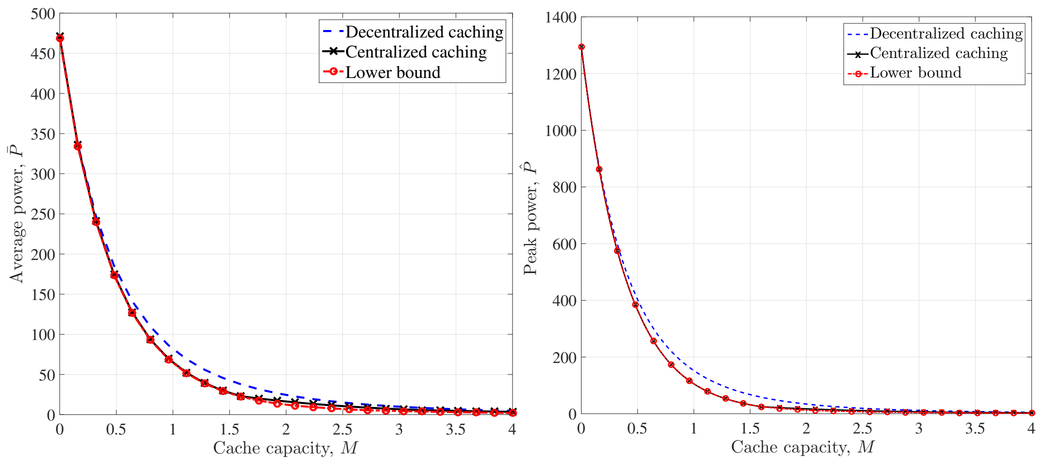
In the following theorem we lower bound $\bar{P}^*(R,M)$ and $\hat{P}^*(R,M)$ by constraining the placement phase to uncoded caching. The main challenge in deriving a lower bound for the cache-aided BC studied here is the lack of degradedness due to the presence of the caches. To derive a lower bound, we assume that the files requested by users in $[k-1]$ and their cache contents are provided to the other users. We then exploit the degradedness of the resultant system to lower bound the performance of the original model.

Theorem 4.3. *In cache-aided content delivery over a Gaussian BC with uncoded cache placement, the minimum average power is lower bounded by $\bar{P}_{\mathrm{LB}}(R,M)$ defined as*

$$\bar{P}_{\mathrm{LB}}(R,M) \triangleq \mathrm{E}_{\mathcal{U}_{\mathbf{d}}} \left[\sum\nolimits_{i=1}^{N_{\mathbf{d}}} \left(\frac{2^{2R(1-\min\{iM/N,1\})} - 1}{h_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^2} \right) \prod\nolimits_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})} \right], \tag{4.52}$$

where $\mathrm{E}_{\mathcal{U}_{\mathbf{d}}}[\cdot]$ takes the expectation over all possible sets $\mathcal{U}_{\mathbf{d}}$, and $\pi_{\mathcal{S}}$ is a permutation over any subset of users $\mathcal{S} \subset [K]$, such that $h_{\pi_{\mathcal{S}}(1)}^2 \leq h_{\pi_{\mathcal{S}}(2)}^2 \leq \cdots \leq h_{\pi_{\mathcal{S}}(|\mathcal{S}|)}^2$. The minimal required peak transmit power for the same system is lower bounded

$$\hat{P}_{\mathrm{LB}}(R,M) \triangleq \max_{\mathcal{S} \subset [\min\{N,K\}]} \left\{ \sum_{i=1}^{|\mathcal{S}|} \left(\frac{2^{2R(1-\min\{iM/N,1\})} - 1}{h_{\pi_{\mathcal{S}}(i)}^2} \right) \prod\nolimits_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})} \right\}. \tag{4.53}$$



(A) Average power-memory trade-off (B) Peak power-memory trade-off

Figure 4.1: Power-memory trade-off for a Gaussian BC with $K = 5$ users, and $N = 8$ files.

Proof. The detailed proof is provided in [94, Section V]. □

Remark 4.6.1. Considering all possible demand vectors, we have a total of 2^{K-1} different $\mathcal{U}_{\boldsymbol{d}}$ sets. This follows from the fact that $N_{\boldsymbol{d}} \le K$ and $1 \in \mathcal{U}_{\boldsymbol{d}}$, $\forall \boldsymbol{d}$. For a given demand vector $\boldsymbol{d}$, let $\mathcal{U}_{\boldsymbol{d}} = \{u_1, u_2, ..., u_{N_{\boldsymbol{d}}}\}$, where $1 = u_1 \le u_2 \le \cdots \le u_{N_{\boldsymbol{d}}}$. The number of demand vectors with the same $\mathcal{U}_{\boldsymbol{d}}$ is given by

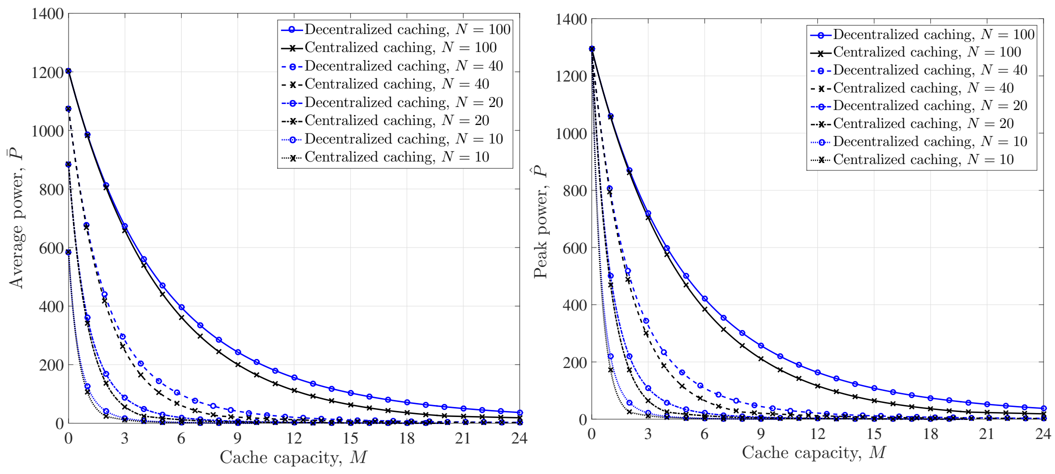
$$N_{\mathcal{U}_d} \triangleq \binom{N}{N_{\boldsymbol{d}}} N_{\boldsymbol{d}}! \left(\prod\nolimits_{j=2}^{N_d} j^{u_{j+1}-u_j-1}\right), \tag{4.54}$$

where we define $u_{N_{\boldsymbol{d}}+1} \triangleq K+1$. Thus, the lower bound in (4.52) reduces to

$$\bar{P}_{\mathrm{LB}}(R,M) = \frac{1}{N^K} \sum_{\mathcal{U}_d \subset [K], 1 \in \mathcal{U}_d} N_{\mathcal{U}_d} \sum_{i=1}^{N_d} \left(\frac{2^{2R(1-\min\{iM/N,1\})} - 1}{h^2_{\pi_{\mathcal{U}_d}(i)}} \right) \prod_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})}. \tag{4.55}$$

4.7 Numerical Results

For the numerical results, we assume that the rate of the files in the library is fixed to $R = 1$, and the channel gains are $1/h_i^2 = 2 - 0.2(i-1)$, $i \in [K]$.



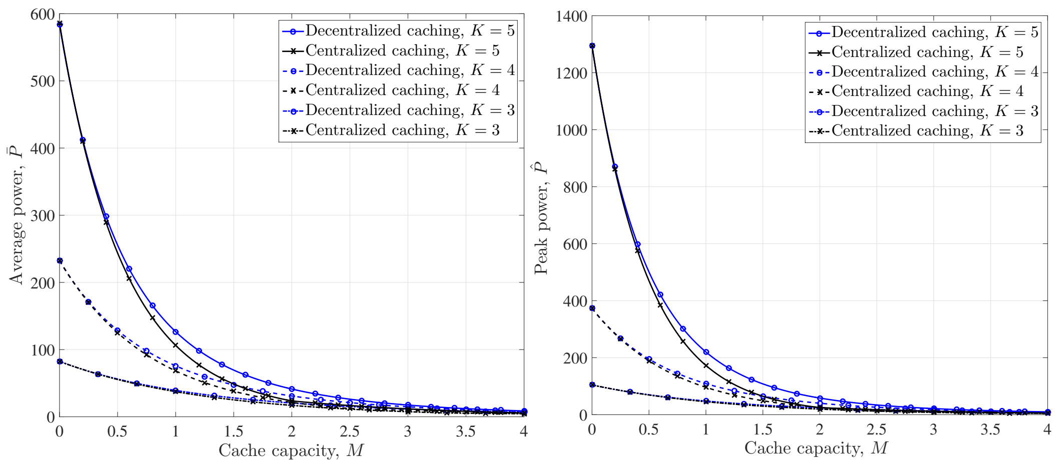
(A) Average power-memory trade-off (B) Peak power-memory trade-off

FIGURE 4.2: Power-memory trade-off for a Gaussian BC with $K = 5$ users, and various number of files $N \in \{10, 20, 40, 100\}$ in the library.

The bounds on the average power-memory trade-off $\bar{P}^*(R, M)$ and peak power-memory trade-off $\hat{P}^*(R, M)$ are shown in Fig. 4.1a and Fig. 4.1b, respectively, for $K = 5$ users, and $N = 8$ files in the library. The gap between the proposed centralized and decentralized caching schemes, which measures the power required to compensate for the decentralization of the cache placement phase, is relatively small, particularly for small and large values of the cache size. This shows that the proposed achievable scheme is robust against the decentralization. We observe that the minimum average and peak powers drop very quickly even with a small cache size available at the users. The lower bound is generally tight for both average and peak power values with respect to the upper bound for the centralized scenario for the whole range of cache capacities[4]. The peak power values exhibit similar behaviour to the average power, with significantly higher values. This shows that adapting the transmit power to the demand combination can reduce the average energy consumption significantly, while the transmitter spends much higher energy for some demand combinations.

Fig. 4.2 illustrates the effect of the number of files N on the upper bound. In Fig. 4.2a and Fig. 4.2b the average power values and the peak power values are considered, respectively, for $K = 5$ users, and different number of files $N \in \{10, 20, 40, 100\}$. The gap between the centralized and decentralized caching scenarios increases with N.

[4]Note that the figure does not include the cache size range $4 \leq M \leq 8$, in which case the three curves coincide in both figures.



(A) Average power-memory trade-off (B) Peak power-memory trade-off

FIGURE 4.3: Power-memory trade-off for a Gaussian BC with various number of users $K \in \{3, 4, 5\}$, and $N = 10$ in the library.

Another observation to be noted is the increase in the average power with N, even though the number of files requested by the users remains the same. This stems from two reasons: first, the effect of the local caches diminishes as the library size increases; and second, the users are less likely to request the same files from a larger library (and hence the increase in the average power values for low M values). We also observe that the peak power increases with N, but only for non-zero M values. This is because the increase in the peak power is only due to the diminished utility of the cache size, whereas the peak power depends only on the worst-case demand combination; and thus, does not depend on the likelihood of common requests. We note that, for a fixed number of users K, the gap between the average power and the peak power reduces by increasing N, and for $N \gg K$, the gap diminishes, since, with high probability, the users demand distinct files.

The effect of the number of users K on the average and peak power values for both the centralized and decentralized scenarios is shown in Fig. 4.3. Fig. 4.3a and Fig. 4.3b demonstrate the average and peak power values as a function of M, respectively, for $N = 10$ and $K \in \{3, 4, 5\}$ users. The gap between centralized and decentralized caching increases with K in both figures. The average and peak power values also increase with K, as expected. For a fixed N value, the increase in the peak power is higher than the one in the average power. This is due to the fact that the likelihood of common demands increases with K, and so does the gap between the average and peak power values.

4.8 Conclusions

In this chapter we have considered cache-aided content delivery over a Gaussian BC. Considering same rate contents in the library, we have studied both the *minimum peak transmission power*, which is the minimum transmit power that can satisfy all user demand combinations, and the *minimum average transmit power*, averaged across all demand combinations, assuming uniform demand distributions. We have proposed a centralized caching and coded delivery scheme assuming that the channel conditions in the delivery phase are not known beforehand. Coded contents are transmitted in the delivery phase to their intended receivers using superposition coding and power allocation. We have then extended the achievable scheme to the decentralized caching scenario. We have also provided a lower bound on the required peak and average transmission power values assuming uncoded cache placement. Our results indicate that even a small cache size at the receivers can provide a significant reduction in the required transmission power level highlighting the benefits of caching in improving the energy efficiency of wireless networks.

Chapter 5

Caching of Multi-Layer Messages

5.1 Overview

In this chapter we study a cache-aided K-user Gaussian BC. The users are equipped with caches of different sizes, which are filled without the knowledge of the user requests in a *centralized* manner. It is assumed that each file can be delivered to different users at different rates, which may correspond to different quality representations of the underlying content, e.g., scalable coded video segments. Accordingly, instead of a single achievable rate, the system performance is characterized by a rate tuple, which corresponds to the vector of rates users' requests can be delivered at. The goal is to characterize the set of all achievable rate tuples for a given total cache size by designing joint cache and channel coding schemes together with cache allocation across users. Assuming that the users are ordered in increasing channel quality, each file is coded into K layers, and only the first i layers of the requested file are delivered to user i, $i \in [K]$. Three different coding schemes are proposed, which differ in the way they deliver the coded contents over the BC; in particular, *time-division*, *superposition*, and *dirty paper coding* schemes are studied. Corresponding achievable rate regions are characterized, and compared with a novel outer bound.

5.2 Introduction

In most of the existing literature on coded caching, the key assumption is that the files in the library are coded at a single common rate, and each user requests one file from the library in its entirety. In this chapter, similar to [40], we allow the users to request the files at different rates; however, different from [40], considering a

Gaussian BC in the delivery phase, we aim at characterizing the rate tuples at which the requested contents can be delivered to the users [95].

We argue that this formulation allows us to better exploit the asymmetric resources available to users for content delivery over a noisy BC. To see the difference between the scalar capacity definition used in [92] and the capacity region formulation proposed here, consider a Gaussian BC without any caches, i.e., $M = 0$. In this case, the capacity as defined in [92] is limited by the rate that can be delivered to the worst user, whereas with our formulation any rate tuple within the capacity region of the underlying BC is achievable, providing a much richer characterization of the performance for cache-aided delivery over a noisy BC.

The motivation here is to deliver the contents at higher rates to users with better channels, rather than being limited by the weak users. As proposed in [40], the multiple rates of the same file may correspond to the video files in the library encoded into multiple quality layers using scalable coding, so the user with a higher delivery rate receives a better quality description of the same file. Accordingly, each file in the library is coded into K layers, K being the number of users, ordered in increasing channel qualities, where user i receives layers 1 to i of its request, for $i \in [K]$. We consider a centralized placement phase, and assume that the channel qualities of the users in the delivery phase are known in advance. By allowing users to have different cache capacities (similarly to [143] considering an error-free shared link during the delivery phase), we consider a total cache size in the network as a constraint, and optimize cache allocation across the users and different layers of the files. Contents cached during the placement phase provide multicasting opportunities to the server to deliver the missing parts in the same layer of the files to different users. When delivering these coded contents to users over the underlying BC, we consider three different techniques. Corresponding coding schemes are called joint cache and time-division coding (CTDC), joint cache and superposition coding (CSC), and joint cache and dirty paper coding (CDPC). We also present an outer bound on the rate region when the placement phase is constrained to uncoded caching, and compare it with the achievable rate tuples obtained though the proposed coding schemes.

The remainder of this chapter is organized as follows. In Section 5.3 we present the system model. We present different inner bounds on the capacity region in Section 5.4, while in Sections 5.5 and 5.6 we elaborate the achievable schemes providing the inner bounds and analyze their achievable rate regions. We also establish an outer bound on the capacity region in Section 5.7. Numerical results are presented in Section 5.8, and conclusions are drawn in Section 5.9.

5.3 System Model

We consider cache-aided content delivery over a K-user Gaussian BC. The transmitter has a library of N files, $\mathbf{W}$. File W_i is coded into K layers $W_i^{(1)}, \dots, W_i^{(K)}$, such that layer $W_i^{(l)}$ is distributed uniformly over the set $\left[\left\lceil 2^{nR^{(l)}} \right\rceil\right]$, where $R^{(l)}$ represents the rate of the l-th layer and n denotes the blocklength, for $i \in [N]$, $l \in [K]$. We denote the l-th layers of all the files by $\mathbf{W}^{(l)} \triangleq \left(W_1^{(l)}, ..., W_N^{(l)}\right)$, for $l \in [K]$.

Assume that user k, $k \in [K]$, has a cache of capacity nM_k bits. For a demand vector $\mathbf{d}$, the users are served by a common message $X^n(\mathbf{W}) \triangleq (X_1(\mathbf{W}), \dots, X_n(\mathbf{W}))$ satisfying the average power constraint. User k, receives $Y_k^n(\mathbf{W}) \triangleq (Y_{k,1}(\mathbf{W}), \dots, Y_{k,n}(\mathbf{W}))$ through a Gaussian channel[1]

$$Y_k^n(\mathbf{W}) = X^n(\mathbf{W}) + Z_{\sigma_k}^n, \quad \text{for } k \in [K], \tag{5.1}$$

where $Z_{\sigma_k}^n \triangleq (Z_{\sigma_k,1}, \dots, Z_{\sigma_k,n})$, and $Z_{\sigma_k,i}$ is an independent noise at user k at the i-th channel use distributed according to $\mathcal{N}\left(0, \sigma_k^2\right)$. Without loss of generality we order the users in increasing channel quality, i.e., we assume that $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_K^2$. We define $\boldsymbol{\sigma} \triangleq (\sigma_1, \dots, \sigma_K)$.

Placement phase is performed in a centralized manner assuming $\boldsymbol{\sigma}$ is known by the server. An $\left(n, R^{(1)}, \dots, R^{(K)}, M_1, \dots, M_K\right)$ code consists of the following:

- K caching functions $\phi_k^{(K)}$, $\forall k \in [K]$, where

$$\phi_k^{(K)} : \left\{\left[\left\lceil 2^{nR^{(1)}} \right\rceil\right] \times \cdots \times \left[\left\lceil 2^{nR^{(K)}} \right\rceil\right]\right\}^N \times \mathbb{R}^{+K} \to \left[\left\lfloor 2^{nM_k} \right\rfloor\right] \tag{5.2}$$

[1]For ease of presentation, we consider a simple channel model, and we do not consider fading in this chapter; however, the results here can be easily extended to the Gaussian fading channel model.

maps $\mathbf{W}$ and $\boldsymbol{\sigma}$ to the cache content B_k of user k, i.e., $B_k = \phi_k^{(K)}(\mathbf{W}, \boldsymbol{\sigma})$.

- An encoding function with the knowledge of the noise variances $\boldsymbol{\sigma}$

$$\psi^{(K)} : \left\{\left[\left\lceil 2^{nR^{(1)}}\right\rceil\right] \times \cdots \times \left[\left\lceil 2^{nR^{(K)}}\right\rceil\right]\right\}^N \times \mathbb{R}^{+K} \times [N]^K \to \mathbb{R}^n, \tag{5.3}$$

which generates the channel input as $X^n(\mathbf{W}) = \psi^{(K)}(\mathbf{W}, \boldsymbol{\sigma}, \mathbf{d})$, for demand vector $\mathbf{d}$, satisfying the average power constraint $\frac{1}{n}\sum_{i=1}^n X_i^2(\mathbf{W}) \le P$.

- K decoding functions $\mu_k^{(K)}$, $\forall k \in [K]$, where, for a demand vector $\mathbf{d}$,

$$\mu_k^{(K)} : \mathbb{R}^n \times \left[\left\lfloor 2^{nM_k}\right\rfloor\right] \times \mathbb{R}^{+K} \times [N]^K \to \left[\left\lceil 2^{nR^{(1)}}\right\rceil\right] \times \cdots \times \left[\left\lceil 2^{nR^{(k)}}\right\rceil\right] \tag{5.4}$$

reconstructs the layers $\hat{W}_k^{(1)}, \dots, \hat{W}_k^{(k)}$ from the channel output $Y_k^n(\mathbf{W})$, cache content B_k, and noise variances $\boldsymbol{\sigma}$.

The probability of error is defined as $P_e^{(K)} \triangleq \Pr\left\{\bigcup_{\mathbf{d}\in[N]^K} \bigcup_{k=1}^K \bigcup_{l=1}^k \left\{\hat{W}_k^{(l)} \neq W_{d_k}^{(l)}\right\}\right\}$.

Note that the generated code implicitly assumes that user k is interested only in the first k layers of its demand, i.e., $W_{d_k}^{(1)}, \dots, W_{d_k}^{(k)}$, for $k \in [K]$. In a more general formulation, we could instead consider an arbitrary ordering of the rates among the users, but here the goal is to deliver a higher rate to a user with a better channel.

For a given total cache size M_{tot}, we say that the rate tuple $(R_1, \dots, R_K)$ is achievable if for every $\varepsilon > 0$, there exists an $\left(n, R^{(1)}, \dots, R^{(K)}, M_1, \dots, M_K\right)$ code, which satisfies $P_e^{(K)} < \varepsilon$, $R_k \le \sum_{l=1}^k R^{(l)}$, and $\sum_{k=1}^K M_k \le M_{\text{tot}}$. For average power constraint P and a total cache size M_{tot}, the capacity region $\mathcal{C}(P, M_{\text{tot}})$ of the caching system described above is defined as the closure of the all achievable rate tuples. Our goal is to find inner and outer bounds on $\mathcal{C}(P, M_{\text{tot}})$.

Next, we present some definitions that will simplify our ensuing presentation. For a fixed value of r, $r \in [K-1]$, we define $g_l \triangleq \sum_{j=1}^l \binom{K-j}{r}$, $\forall l \in [K-r]$, and let $g_0 = 0$. We note that $g_{K-r} = \binom{K}{r+1}$. We denote the set of users $[l : K]$ by $\mathcal{K}_l$, for $l \in [K]$. We label $(r+1)$-element subsets of users in $\mathcal{K}_1$, so that the subsets with the smallest element l are labelled as

$$\mathcal{S}^{r+1}_{\mathcal{K}_1,1+g_{l-1}},\ldots,\mathcal{S}^{r+1}_{\mathcal{K}_1,g_l},\quad \text{for } l=1,...,K-r. \tag{5.5}$$

Thus, we have, for $l \in [K-r]$,

$$\left\{\mathcal{S}^{r+1}_{\mathcal{K}_1,1+g_{l-1}}\backslash\{l\},\ldots,\mathcal{S}^{r+1}_{\mathcal{K}_1,g_l}\backslash\{l\}\right\}=\left\{\mathcal{S}^{r}_{\mathcal{K}_{l+1},1},\ldots,\mathcal{S}^{r}_{\mathcal{K}_{l+1},\binom{K-l}{r}}\right\}, \tag{5.6}$$

i.e., the family of all $(r+1)$-element subsets of $\mathcal{K}_1$ excluding l, which is their smallest element, is the same as the family of all r-element subsets of $\mathcal{K}_{l+1}$. We note that the number of subsets of users in both sets in (5.6) is $\binom{K-l}{r}$, $l \in [K-r]$. Without loss of generality, we label the subsets of users so that, for $l \in [K-r]$,

$$\mathcal{S}^{r+1}_{\mathcal{K}_1,i+g_{l-1}}\backslash\{l\}=\mathcal{S}^{r}_{\mathcal{K}_{l+1},i},\quad \text{for } i \in \left[\binom{K-l}{r}\right]. \tag{5.7}$$

5.4 Achievability Results

Here we present the results of three achievable schemes, namely *joint cache and time-division coding* (CTDC), *joint cache and superposition coding* (CSC), and *joint cache and dirty paper coding* (CDPC), providing inner bounds on $\mathcal{C}(P, M)$.

5.4.1 CTDC Scheme

In the following, we present an achievable rate region achieved by the CTDC scheme. With CTDC, the missing bits corresponding to the layers in $\mathbf{W}^{(l)}$ are delivered in a coded manner exploiting the cached contents as in the standard coded caching framework. The coded contents are transmitted over the BC using time-division among layers. We elaborate the placement and delivery phases of the CTDC scheme in Section 5.5.

Proposition 5.4.1. For the system described in Section 5.3 with average power P and total cache size M_{tot}, the rate tuple $(R_1, ..., R_K)$ is achievable by the CTDC scheme, if there exist $r_1, \ldots, r_K$, where $r_l \in [0 : K-l]$, $\forall l \in [K]$, non-negative $R^{(1)}, \ldots, R^{(K)}$, and non-negative $\lambda^{(1)}, \dots, \lambda^{(K)}$, such that $R_k = \sum_{l=1}^{k} R^{(l)}$, $\sum_{l=1}^{K} \lambda^{(l)} = 1$, $\forall k \in [K]$, and

$$R^{(l)} \le \lambda^{(l)} \frac{\sum_{i=1}^{\binom{K-l+1}{t_l}} \prod_{k \in \mathcal{K}_l \backslash \mathcal{S}^{t_l}_{\mathcal{K}_l,i}} C^{P}_{\sigma_k^2}}{\sum_{i=1}^{\binom{K-l+1}{t_l+1}} \prod_{k \in \mathcal{K}_l \backslash \mathcal{S}^{t_l+1}_{\mathcal{K}_l,i}} C^{P}_{\sigma_k^2}}, \quad \text{for } l \in [K], \tag{5.8a}$$

$$M_{\text{tot}} = N \sum_{l=1}^{K} r_l R^{(l)}, \tag{5.8b}$$

where, for $a, b \in \mathbb{R}^+$, we define $C^a_b \triangleq \frac{1}{2}\log_2(1 + a/b)$.

Corollary 5.1. *The following rate region for a total cache size M_{tot} and average power P can be achieved by the CTDC scheme:*

$$\mathcal{C}_b(P, M_{\text{tot}}) = \bigcup_{\lambda^{(1)},\dots,\lambda^{(K)}: \sum_{l=1}^{K} \lambda^{(l)} = 1} \left(\{R_1, \dots, R_K\} : (R_1, \dots, R_K) \textit{ and } M_{\text{tot}} \textit{ satisfy } (5.8) \right). \tag{5.9}$$

Remark 5.4.1. Let $(\hat{R}_1, \dots, \hat{R}_K) \in \mathcal{C}_b(P, M_{\text{tot}})$ and $(\tilde{R}_1, \dots, \tilde{R}_K) \in \mathcal{C}_b(P, M_{\text{tot}})$. Then, for any $\lambda \in [0, 1]$, $(\lambda\hat{R}_1 + \bar{\lambda}\tilde{R}_1, \dots, \lambda\hat{R}_K + \bar{\lambda}\tilde{R}_K) \in \mathcal{C}_b(P, M_{\text{tot}})$, where $\bar{\lambda} \triangleq 1 - \lambda$. This can be shown by joint time and memory-sharing. The whole library is divided into two parts according to λ, and the delivery of the two parts are carried out over two orthogonal time intervals of length λn and $\bar{\lambda} n$ using the codes for the two achievable tuples. Thus, for a fixed total cache size M_{tot}, the rate pairs in the convex-hull of $\mathcal{C}_b(P, M_{\text{tot}})$ are achievable.

From convexity of rate region $\mathcal{C}_b(P, M_{\text{tot}})$, a rate vector $\mathbf{R}^* \triangleq (R^*_1, \dots, R^*_K)$ is on the boundary surface of $\mathcal{C}_b(P, M_{\text{tot}})$, if there exist non-negative coefficients $\omega_1, \dots, \omega_K$, $\sum_{i=1}^{K} \omega_i = 1$, for which $\mathbf{R}^*$ is a solution to the following optimization problem:

$$\max_{\lambda^{(1)},\dots,\lambda^{(K)}, R_1,\dots,R_K} \sum_{i=1}^{K} \omega_i R_i, \quad \text{subject to } \{R_1, ..., R_K\} \in \mathcal{C}_b(P, M_{\text{tot}}). \tag{5.10}$$

In the other words, for given weights $\omega_1, \ldots, \omega_K$, and total cache size M_{tot}, $\mathbf{R}^*$ solves the problem in (5.10), if $R^{(1)}, \ldots, R^{(K)}$ is a solution of the following problem:

$$\begin{aligned} &\max_{\lambda^{(1)},\ldots,\lambda^{(K)},R^{(1)},\ldots,R^{(K)}} \sum\nolimits_{i=1}^{K} \omega_i \sum\nolimits_{l=1}^{i} R^{(l)}, \\ &\text{subject to (5.8a) and (5.8b)}, \\ &\sum\nolimits_{l=1}^{K} \lambda^{(l)} = 1, \end{aligned} \tag{5.11}$$

and

$$R_k^* = \sum\nolimits_{l=1}^{k} R^{(l)}, \quad \text{for } k = 1, \ldots, K. \tag{5.12}$$

Remark 5.4.2. For given weights $\omega_1, \ldots, \omega_K$, it is easy to verify that the problem in (5.11) is a linear optimization problem; thus it is a convex optimization problem.

5.4.2 CSC and CDPC Schemes

Here we present the achievable rate regions for the CSC and CDPC schemes. We introduce s_1 and s_2 to distinguish between the two, where we set $s_1 = 0$ and $s_2 = 1$ for CSC, while $s_1 = 1$ and $s_2 = 0$ for CDPC. We briefly highlight here that with the CSC scheme, the coded packets of different layers are delivered over the Gaussian BC through superposition coding, while the CDPC scheme uses dirty paper coding to deliver the coded packets of different layers. The CSC scheme along with an example highlighting the main techniques and the CDPC scheme are elaborated in Section 5.6.

Theorem 5.1. *For the system described in Section 5.3 with average power P and total cache size M_{tot}, the rate tuple $(R_1, ..., R_K)$ is achievable, if there exist $r \in [K-1]$, and non-negative $R^{(1)}, \ldots, R^{(K)}$, such that $R_k = \sum_{l=1}^{k} R^{(l)}$, for $k \in [K]$, and*

$$R^{(l)} = \begin{cases} \sum_{i=1}^{\binom{K}{r}} R^{(1)}_{\mathcal{S}^r_{\mathcal{K}_1,i}}, & \textit{if } l = 1, \\ \sum_{i=1}^{\binom{K-l+1}{r-1}} R^{(l)}_{\mathcal{S}^{r-1}_{\mathcal{K}_l,i}}, & \textit{if } l = 2, ..., K-r+1, \\ 0, & \textit{otherwise}, \end{cases} \tag{5.13a}$$

and, for $i \in [1+g_{l-1} : g_l]$ *and* $l \in [K-r]$,

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} \leq \lambda_i C^{\rho_i P}_{\bar{\rho}_i P s_2+\sigma^2_{k_1}}, \ \forall k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}, \tag{5.13b}$$

$$R^{(l+1)}_{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k_2\}} \leq \lambda_i C^{\bar{\rho}_i P}_{\rho_i P s_1+\sigma^2_{k_2}}, \ \forall k_2 \in \mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}, \tag{5.13c}$$

where $\bar{\rho}_i \triangleq 1-\rho_i$, *and*

$$M_{\text{tot}} = N\left(rR^{(1)} + (r-1)\sum\nolimits_{l=2}^{K-r+1} R^{(l)}\right), \tag{5.13d}$$

for some

$$0 \leq \rho_i \leq 1, \quad \textit{for } i = 1,...,\tbinom{K}{r+1}, \tag{5.13e}$$

$$0 \leq \lambda_i \leq 1, \quad \textit{for } i = 1,...,\tbinom{K}{r+1}, \tag{5.13f}$$

$$\sum\nolimits_{i=1}^{\binom{K}{r+1}} \lambda_i = 1. \tag{5.13g}$$

Corollary 5.2. *The following rate region for a total cache size* M_{tot} *and average power constraint* P *can be achieved:*

$$\mathcal{C}_c(P, M_{\text{tot}}) = \bigcup_{\boldsymbol{\rho},\boldsymbol{\lambda}:\sum_{i=1}^{\binom{K}{t+1}}\lambda_i=1} \left(\{R_1,\dots,R_K\} : (R_1,\dots,R_K) \textit{ and } M_{\text{tot}} \textit{ satisfy } (5.13)\right), \tag{5.14}$$

where $\boldsymbol{\rho} \triangleq \left\{\rho_1,\dots,\rho_{\binom{K}{t+1}}\right\}$, *and* $\boldsymbol{\lambda} \triangleq \left\{\lambda_1,\dots,\lambda_{\binom{K}{t+1}}\right\}$.

For a fixed total cache size M_{tot}, the convexity of region $\mathcal{C}_c(P, M_{\text{tot}})$ is followed through the same argument as Remark 5.4.1, for both the CSC and CDPC schemes. As a result, for a given total cache size M_{tot}, and for given non-negative coefficients $\omega_1,\dots,\omega_K$, such that $\sum_{i=1}^K \omega_i = 1$, a rate vector $\mathbf{R}^*$ is on the boundary surface of the achievable rate region $\mathcal{C}_c(P, M_{\text{tot}})$, if $R^{(1)},\dots,R^{(K)}$ is a solution of the following problem:

$$\max_{\boldsymbol{\rho},\boldsymbol{\lambda},\mathbf{R}^{(1)},\dots,\mathbf{R}^{(K-t+1)}} \sum\nolimits_{i=1}^K \omega_i \sum\nolimits_{l=1}^i R^{(l)},$$

$$\text{subject to } R^{(1)},\dots,R^{(K-t+1)} \text{ satisfy (5.13a)},$$

$$\begin{aligned}&\mathbf{R}^{(1)} \text{ satisfy (5.13b)},\\ &\mathbf{R}^{(2)},\dots,\mathbf{R}^{(K-t+1)} \text{ satisfy (5.13c)},\\ &M_{\text{tot}} \text{ satisfies (5.13d)},\\ &\boldsymbol{\rho} \text{ and } \boldsymbol{\lambda} \text{ satisfy (5.13e)-(5.13g)},\end{aligned} \tag{5.15a}$$

where

$$\mathbf{R}^{(1)} \triangleq R^{(1)}_{\mathcal{S}^t_{\mathcal{K}_1,1}},\dots,R^{(1)}_{\mathcal{S}^t_{\mathcal{K}_1,\binom{K}{t}}}, \tag{5.15b}$$

$$\mathbf{R}^{(l)} \triangleq R^{(l)}_{\mathcal{S}^t_{\mathcal{K}_l,1}},\dots,R^{(l)}_{\mathcal{S}^t_{\mathcal{K}_l,\binom{K-l+1}{t-1}}}, \text{ for } l \in [2:K-t+1], \tag{5.15c}$$

and

$$R^*_k = \sum\nolimits_{l=1}^{k} R^{(l)}, \quad \text{for } k = 1,\dots,K. \tag{5.16}$$

Remark 5.4.3. Let $\tilde{\mathbf{R}} \triangleq (\tilde{R}_1,\dots,\tilde{R}_K)$ and $\hat{\mathbf{R}} \triangleq (\hat{R}_1,\dots,\hat{R}_K)$ be two achievable rate tuples for total cache capacities $\tilde{M}_{\text{tot}}$ and $\hat{M}_{\text{tot}}$, respectively. Then, $\lambda\tilde{\mathbf{R}} + \bar{\lambda}\hat{\mathbf{R}}$ can be achieved through joint time and memory-sharing for a total cache size $\lambda\tilde{M}_{\text{tot}} + \bar{\lambda}\hat{M}_{\text{tot}}$, for some $\lambda \in [0,1]$. For $M_{\text{tot}} = 0$, the system under consideration is equivalent to the Gaussian BC without user caches, where user k requests a file of rate $\sum_{l=1}^{k} R^{(l)}$, $k \in [K]$, and rate tuple $\mathbf{R}_z \triangleq (R_{z_1},...,R_{z_K})$ is achievable by superposition coding, where

$$R_{z_k} = C_{\sum_{i=k+1}^{K}\gamma_i P+\sigma_k^2}^{\gamma_k P}, \quad \text{for } k \in [K], \tag{5.17}$$

for some non-negative coefficients $\gamma_1,\dots,\gamma_K$, such that $\sum_{i=1}^{K}\gamma_i = 1$. Hence, rate tuples $\lambda\mathbf{R}_z + \bar{\lambda}\tilde{\mathbf{R}}$ and $\lambda\mathbf{R}_z + \bar{\lambda}\hat{\mathbf{R}}$ are also achievable for total cache capacities $\bar{\lambda}\tilde{M}_{\text{tot}}$ and $\bar{\lambda}\hat{M}_{\text{tot}}$, respectively, through time sharing.

5.5 CTDC Scheme (Proof of Proposition 5.4.1)

With the DTM scheme, the layers with $\mathbf{W}^{(l)}$, for $l \in [K]$, are cached and delivered via a distinct time slot (TS). We elaborate the placement and delivery phases of the CTDC scheme in the following.

Placement phase: The layers with $\mathbf{W}^{(l)}$ are cached partially by the users in $\mathcal{K}_l$ constrained by their cache capacities, for $l \in [K]$. For caching factors $r_1, \dots, r_K$, where $r_l \in [0 : K-l]$, layer $W_j^{(l)}$ is divided into $\binom{K-l+1}{r_l}$ disjoint subfiles $W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,1}}$, $\dots$, $W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,\binom{K-l+1}{r_l}}}$, where subfile $W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}$ is of rate $R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}$, for $i \in \left[\binom{K-l+1}{r_l}\right]$, $l \in [K]$, $j \in [N]$.[2] We note that

$$R^{(l)} = \sum\nolimits_{i=1}^{\binom{K-l+1}{r_l}} R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}, \quad \text{for } l \in [K]. \tag{5.18}$$

User k's cache content, for $k \in [K]$, is given by

$$B_k = \bigcup\nolimits_{j=1}^{N} \bigcup\nolimits_{l=1}^{k} \bigcup\nolimits_{i \in \left[\binom{K-l+1}{r_l}\right] : k \in \mathcal{S}^{r_l}_{\mathcal{K}_l,i}} W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}, \tag{5.19}$$

which leads to a total cache size of

$$M_{\text{tot}} = \sum\nolimits_{k=1}^{K} M_k = N \sum\nolimits_{l=1}^{K} r_l R^{(l)}. \tag{5.20}$$

Delivery phase: For a demand vector $\mathbf{d}$, the server aims to deliver the coded packet

$$W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} = \overline{\bigoplus}_{k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} W^{(l)}_{d_k,\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i} \setminus \{k\}} \tag{5.21}$$

of rate

$$R^{(l)}_{\text{XOR},\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} \triangleq \max_{k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} \left\{ R^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i} \setminus \{k\}} \right\} \tag{5.22}$$

to the users in $\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$, for $i \in \left[\binom{K-l+1}{r_l+1}\right]$ and $l \in [K]$. Each user $k \in \mathcal{K}_l$ can obtain all missing bits of its request $W^{(l)}_{d_k}$ after receiving

$$\bigcup\nolimits_{i \in \left[\binom{K-l+1}{r_l+1}\right] : k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} \tag{5.23}$$

along with its cache content, for $l \in [K]$. We allocate a distinct $\lambda^{(l)} n$ channel uses to deliver the coded packets $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,1}}, \dots, W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,\binom{K-l+1}{r_l+1}}}$ to the intended users in $\mathcal{K}_l$, for some $\lambda^{(l)} \in [0,1]$, $l \in [K]$, where each coded packet among them is delivered via a

[2]We assume throughout the paper that, for any real number $a \geq 0$, 2^{na} is an integer for n large enough.

different TS, and we have $\sum_{l=1}^{K}\lambda^{(l)} = 1$. The coded packet $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$ of the files in the l-th layer is delivered to the users $\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$ through a distinct time interval of length $\lambda^{(l)}_i n$, for $i \in \left[\binom{K-l+1}{r_l+1}\right]$ and $l \in [K]$, where $\sum_{i=1}^{\binom{K-l+1}{r_l+1}} \lambda^{(l)}_i = \lambda^{(l)}$. In order to recover the coded packet $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$, user $k_1 \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$ first generates

$$\overline{\bigoplus}_{k\in\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}\setminus\{k_1\}} W^{(l)}_{d_k,\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}\setminus\{k\}} \tag{5.24}$$

from its cache; it then only needs to decode $W^{(l)}_{d_{k_1},\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}\setminus\{k_1\}}$ of rate $R^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}\setminus\{k_1\}}$, which the decoding is successful for n large enough, if

$$R^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}\setminus\{k_1\}} \le \lambda^{(l)}_i C^P_{\sigma^2_{k_1}}, \quad \text{for } i \in \left[\binom{K-l+1}{r_l+1}\right] \text{ and } l \in [K]. \tag{5.25}$$

By choosing

$$\lambda^{(l)}_i = \frac{\prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}{\sum_{i=1}^{\binom{K-l+1}{r_l+1}} \prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}\lambda^{(l)}, \quad \text{for } i \in \left[\binom{K-l+1}{r_l+1}\right] \text{ and } l \in [K], \tag{5.26}$$

which satisfies $\sum_{i=1}^{\binom{K-l+1}{r_l+1}} \lambda^{(l)}_i = \lambda^{(l)}$ and leads to

$$R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}} \le \lambda^{(l)} \frac{\prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}{\sum_{i=1}^{\binom{K-l+1}{r_l+1}} \prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}, \tag{5.27}$$

it can be checked that all the conditions in (5.25) are satisfied. Therefore, the coded packets $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,1}},\ldots,W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,\binom{K-l+1}{r_l+1}}}$, each delivered with an average power P via a distinct TS, can be decoded by their intended users successfully, if, for n large enough,

$$R^{(l)} \le \lambda^{(l)} \frac{\sum_{i=1}^{\binom{K-l+1}{r_l}} \prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}{\sum_{i=1}^{\binom{K-l+1}{r_l+1}} \prod_{k\in\mathcal{K}_l\setminus\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} C^P_{\sigma^2_k}}, \quad \text{for } l \in [K], \tag{5.28}$$

which together with the total cache size given in (5.20) complete the proof of Proposition 5.4.1.

We remark here that the CTDC scheme applies the scheme in [92], proposed when the messages are of the same rate, to the scenario of multiple-layer messages through joint time and memory-sharing.

5.6 CSC and CDPC Schemes (Proof of Theorem 5.1)

Here we present the CSC and CDPC schemes, which achieve the rate tuple presented in Theorem 5.1 for $s_1 = 0, s_2 = 1$, and $s_1 = 1, s_2 = 0$, respectively, and $r \in [K-1]$.

Placement phase: As described in Section 5.3, user k receives layers 1 to k of its request, i.e., $W_{d_k}^{(1)}, \dots, W_{d_k}^{(k)}$, for $k \in [K]$. Thus, the l-th layer of the files, i.e., $\mathbf{W}^{(l)}$, are cached partially by the users in $\mathcal{K}_l$ constrained by their cache capacities, for $l \in [K]$. For $r \in [K-1]$, we set

$$r_l = \begin{cases} r, & \text{if } l = 1, \\ r-1, & \text{if } 2 \le l \le K-r+1, \\ 0 & \text{otherwise}, \end{cases} \tag{5.29}$$

and

$$R^{(l)} = 0, \quad \text{for } l \in [K-r+2 : K]. \tag{5.30}$$

The l-th layer of each file, i.e., each layer with $\mathbf{W}^{(l)}$, which are targeted for users in $\mathcal{K}_l$, is split into $\binom{K-l+1}{r_l}$ disjoint subfiles, for $l \in [K-r+1]$, represented by

$$W_j^{(l)} = \bigcup_{i=1}^{\binom{K-l+1}{r_l}} W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}, \quad \text{for } j \in [N], \tag{5.31}$$

where subfile $W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}$ is of rate $R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}$, $i \in \left[\binom{K-l+1}{r_l}\right]$. We note that $\sum_{i=1}^{\binom{K-l+1}{r_l}} R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}} = R^{(l)}$, $l \in [K-r+1]$. User k's cache content, $k \in [K]$, is given by

$$B_k = \bigcup_{j=1}^{N} \bigcup_{l=1}^{k} \bigcup_{i \in \left[\binom{K-l+1}{r_l}\right] : k \in \mathcal{S}^{r_l}_{\mathcal{K}_l,i}} W^{(l)}_{j,\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}, \tag{5.32}$$

leading to the cache size

$$M_k = N \sum\nolimits_{l=1}^{k} \sum\nolimits_{i \in \left[\binom{K-l+1}{r_l}\right] : k \in \mathcal{S}^{r_l}_{\mathcal{K}_l,i}} R^{(l)}_{\mathcal{S}^{r_l}_{\mathcal{K}_l,i}}. \tag{5.33}$$

We can obtain the total cache size in the system as

$$M_{\text{tot}} = \sum_{k=1}^{K} M_k = N \sum_{l=1}^{K-r+1} r_l R^{(l)} = N \left(r R^{(1)} + (r-1) \sum\nolimits_{l=2}^{K-r+1} R^{(l)} \right), \tag{5.34}$$

which is equal to the one in (5.13d). We note that the rate of the l-th layer of each file, i.e., $R^{(l)}$, for $l \in [K]$, corresponds to (5.13a).

Delivery phase: For a demand vector $\mathbf{d}$, the server delivers the coded packet

$$W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} = \overline{\bigoplus}_{k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} W^{(l)}_{d_k, \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i} \backslash \{k\}}, \text{ for } i \in \left[\binom{K-l+1}{r_l+1}\right], \tag{5.35}$$

of rate

$$R^{(l)}_{\text{XOR}, \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} = \max_{k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}} \left\{ R^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i} \backslash \{k\}} \right\} \tag{5.36}$$

to the users in $\mathcal{S}^{r_l}_{\mathcal{K}_l,i}$, for $l \in [K-r+1]$. Thus, after receiving $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$, each user $k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$ can recover the missing subfile $W^{(l)}_{d_k, \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i} \backslash \{k\}}$ of the l-th layer of its request, for $i \in \left[\binom{K-l+1}{r_l+1}\right]$ and $l \in [K-r+1]$. We note that, user k, for $k \in [K]$, only exists in the sets $\mathcal{K}_1, \dots, \mathcal{K}_k$, and also the rate of each layer with $\mathbf{W}^{(K-r+2)}, \dots, \mathbf{W}^{(K)}$ is set to zero. Thus, user k, for $k \in [K-r+1]$, can recover all missing subfiles of layers $W^{(1)}_{d_k}, \dots, W^{(k)}_{d_k}$ after receiving all the coded packets $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$, $\forall i \in \left[\binom{K-l+1}{r_l+1}\right]$ and $\forall l \in [k]$, such that $k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$. On the other hand, user k, for $k \in [K-r+2 : K]$, can recover the missing bits of all the layers $W^{(1)}_{d_k}, \dots, W^{(K-r+1)}_{d_k}$ after receiving $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$, $\forall i \in \left[\binom{K-l+1}{r_l+1}\right]$ and $\forall l \in [K-r+1]$, such that $k \in \mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}$. We remind here that r_l is given in (5.29).

The main technique to deliver the coded packets is to send the packet targeted to the users in $\mathcal{S}^{r}_{\mathcal{K}_{l+1},i}$ along with the packet targeted to the users in $\mathcal{S}^{r+1}_{\mathcal{K}_1,i+g_{l-1}}$ through different channel coding techniques, where, from (5.7), $\mathcal{S}^{r}_{\mathcal{K}_{l+1},i} = \mathcal{S}^{r+1}_{\mathcal{K}_1,i+g_{l-1}} \backslash \{l\}$, for $i \in \left[\binom{K-l}{r}\right]$ and $l \in [K-r]$. For this purpose, the transmission is performed via $\binom{K}{r+1}$

orthogonal TSs, where the i-th TS is of length $\lambda_i n$ channel uses, for $i \in \left[\binom{K}{r+1}\right]$, so that $\sum_{i=1}^{\binom{K}{r+1}} \lambda_i = 1$.

5.6.1 CSC Scheme

With TS i, for $i \in [1 + g_{l-1} : g_l]$ and $l \in [K - r]$, we generate two subcodebooks

$$\mathcal{A}_1 \triangleq \left\{ x_1^{\lambda_i n}(w_1) : w_1 \in \left[2^{nR^{(1)}_{\mathrm{XOR},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}}} \right] \right\}, \tag{5.37a}$$

$$\mathcal{A}_2 \triangleq \left\{ x_2^{\lambda_i n}(w_2) : w_2 \in \left[2^{nR^{(l+1)}_{\mathrm{XOR},\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}}} \right] \right\}, \tag{5.37b}$$

where all the entries in $\mathcal{A}_1$ and $\mathcal{A}_2$ are drawn i.i.d. according to $\mathcal{N}(0, \rho_i P)$ and $\mathcal{N}(0, \bar{\rho}_i P)$, respectively, for some $\rho_i \in [0, 1]$. The server then transmits the codeword

$$x_1^{\lambda_i n}\left(W^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}} \right) + x_2^{\lambda_i n}\left(W^{(l+1)}_{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}} \right), \tag{5.38}$$

sent through linear superposition of the codewords from subcodebooks $\mathcal{A}_1$ and $\mathcal{A}_2$, over the Gaussian BC with TS i, for $i \in [1 + g_{l-1} : g_l]$, $l \in [K - r]$. We note that $g_{K-r} = \sum_{j=1}^{K-r} \binom{K-j}{r} = \binom{K}{r+1}$. We also note that, if all the coded packets $W^{(l+1)}_{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}}$ are received by their targeted users successfully via all TSs i, $\forall i \in [1 + g_{l-1} : g_l]$, then the users in $\mathcal{K}_{l+1}$ can obtain the missing subfiles of the $(l+1)$-th layer of their demands, for $l \in [K - r]$. On the other hand, the users in $\mathcal{K}_1$ need to receive all coded packets $W^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}}$ targeted for them via all $\binom{K}{r+1}$ TSs to obtain the first layer of their requests. The users in $\mathcal{S}^{r+1}_{\mathcal{K}_1,i}$ first decode the message with $x_1^{\lambda_i n}$, while considering $x_2^{\lambda_i n}$ as noise. To decode the message with $x_1^{\lambda_i n}$, each user $k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$ first recovers

$$\overline{\bigoplus}_{k \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i} \setminus \{k_1\}} W^{(1)}_{d_k, \mathcal{S}^{r+1}_{\mathcal{K}_1,i} \setminus \{k\}} \tag{5.39}$$

from its cache, and it only needs to decode $W^{(1)}_{d_{k_1}, \mathcal{S}^{r+1}_{\mathcal{K}_1,i} \setminus \{k_1\}}$ of rate $R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i} \setminus \{k_1\}}$, which, using an optimal decoding, the decoding is successful for n large enough, if

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i} \setminus \{k_1\}} \le \lambda_i C^{\rho_i P}_{\bar{\rho}_i P + \sigma^2_{k_1}}, \quad \text{for } i \in [1 + g_{l-1} : g_l] \text{ and } l \in [K - r]. \tag{5.40}$$

TABLE 5.1: Codewords sent via 4 TSs in the delivery phase.

TS number	Transmitted codeword
1	$x_1^{\lambda_1 n}\left(W^{(1)}_{\{1,2,3\}}\right) + x_2^{\lambda_1 n}\left(W^{(2)}_{\{2,3\}}\right)$
2	$x_1^{\lambda_2 n}\left(W^{(1)}_{\{1,2,4\}}\right) + x_2^{\lambda_2 n}\left(W^{(2)}_{\{2,4\}}\right)$
3	$x_1^{\lambda_3 n}\left(W^{(1)}_{\{1,3,4\}}\right) + x_2^{\lambda_3 n}\left(W^{(2)}_{\{3,4\}}\right)$
4	$x_1^{\lambda_4 n}\left(W^{(1)}_{\{2,3,4\}}\right) + x_2^{\lambda_4 n}\left(W^{(3)}_{\{3,4\}}\right)$

We note, from (5.7), that

$$\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}} = \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}, \quad \text{for } i \in [1+g_{l-1} : g_l]; \tag{5.41}$$

thus, each user in $\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}$, for which the message with codeword $x_2^{\lambda_i n}$ is targeted to, can decode $x_1^{\lambda_i n}$ having (5.40) satisfied, for $l \in [K-r]$. Similarly, to decode the message with $x_2^{\lambda_i n}$, each user $k_2 \in \mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}$ first recovers

$$\overline{\bigoplus}_{k\in\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k_2\}} W^{(l+1)}_{d_k,\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k\}} \tag{5.42}$$

from its cache, and it only needs to decode $W^{(l+1)}_{d_{k_2},\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k_2\}}$ of rate $R^{(l+1)}_{\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k_2\}}$, which, using an optimal decoding, the decoding is successful for n large enough, if

$$R^{(l+1)}_{\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}\backslash\{k_2\}} \leq \lambda_i C^{\bar{\rho}_i P}_{\sigma^2_{k_2}}, \quad \text{for } i \in [1+g_{l-1} : g_l] \text{ and } l \in [K-r]. \tag{5.43}$$

Observe that the conditions in (5.40) and (5.43) prove the achievability of the rate tuple outlined in Theorem 5.1 for the total cache size M_{tot} given in (5.34), which is the same as the one in (5.13d), when $s_1 = 0$ and $s_2 = 1$.

We highlight that, at each TS of the CSC scheme two coded packets from different layers of messages are superposed for transmission. The first coded packet is always intended for the first layer of the messages and the second one is for a higher layer. The second coded packet is targeted for a subset of the users receiving the first coded packet.

In the following, we present an example of the CSC scheme for more clarification.

TABLE 5.2: Decoding the message with $x_1^{\lambda_i n}$ at TS i, for $i = 1, ..., 4$

TS number	Sufficient conditions
1	$R_{\{2,3\}}^{(1)} \leq \lambda_1 C_{\bar{\rho}_1 P+\sigma_1^2}^{\rho_1 P}$ $R_{\{1,3\}}^{(1)} \leq \lambda_1 C_{\bar{\rho}_1 P+\sigma_2^2}^{\rho_1 P}$ $R_{\{1,2\}}^{(1)} \leq \lambda_1 C_{\bar{\rho}_1 P+\sigma_3^2}^{\rho_1 P}$
2	$R_{\{2,4\}}^{(1)} \leq \lambda_2 C_{\bar{\rho}_2 P+\sigma_1^2}^{\rho_2 P}$ $R_{\{1,4\}}^{(1)} \leq \lambda_2 C_{\bar{\rho}_2 P+\sigma_2^2}^{\rho_2 P}$ $R_{\{1,2\}}^{(1)} \leq \lambda_2 C_{\bar{\rho}_2 P+\sigma_4^2}^{\rho_2 P}$
3	$R_{\{3,4\}}^{(1)} \leq \lambda_3 C_{\bar{\rho}_3 P+\sigma_1^2}^{\rho_3 P}$ $R_{\{1,4\}}^{(1)} \leq \lambda_3 C_{\bar{\rho}_3 P+\sigma_3^2}^{\rho_3 P}$ $R_{\{1,3\}}^{(1)} \leq \lambda_3 C_{\bar{\rho}_3 P+\sigma_4^2}^{\rho_3 P}$
4	$R_{\{3,4\}}^{(1)} \leq \lambda_4 C_{\bar{\rho}_4 P+\sigma_2^2}^{\rho_4 P}$ $R_{\{2,4\}}^{(1)} \leq \lambda_4 C_{\bar{\rho}_4 P+\sigma_3^2}^{\rho_4 P}$ $R_{\{2,3\}}^{(1)} \leq \lambda_4 C_{\bar{\rho}_4 P+\sigma_4^2}^{\rho_4 P}$

5.6.2 Example

Consider a cache-aided network as described in Section 5.3 with $K = 4$ users in the system. Here we exemplify the achievability of the rate region stated in Theorem 5.1 for the CSC scheme for $r = 2$. We set $R^{(4)} = 0$, and split the messages in the l-th layer, for $l \in [3]$, as follows:

$$W_j^{(1)} = \left(W_{j,\{1,2\}}^{(1)}, W_{j,\{1,3\}}^{(1)}, W_{j,\{1,4\}}^{(1)}, W_{j,\{2,3\}}^{(1)}, W_{j,\{2,4\}}^{(1)}, W_{j,\{3,4\}}^{(1)}\right), \tag{5.44a}$$

$$W_j^{(2)} = \left(W_{j,\{2\}}^{(2)}, W_{j,\{3\}}^{(2)}, W_{j,\{4\}}^{(2)}\right), \tag{5.44b}$$

$$W_j^{(3)} = \left(W_{j,\{3\}}^{(3)}, W_{j,\{4\}}^{(3)}\right), \tag{5.44c}$$

where subfile $W_{j,\mathcal{S}_{\mathcal{K}_l,i}^{r_l}}^{(l)}$ is of rate $R_{\mathcal{S}_{\mathcal{K}_l,i}^{r_l}}^{(l)}$, for $i \in \left[\binom{5-l}{r_l}\right]$, $\forall j \in [N]$, and $r_1 = 2, r_2 = r_3 = 1$, and $r_4 = 0$.

The cache content of each user is given by

$$B_1 = \bigcup\nolimits_{j\in[N]} \left(W_{j,\{1,2\}}^{(1)}, W_{j,\{1,3\}}^{(1)}, W_{j,\{1,4\}}^{(1)}\right), \tag{5.45a}$$

$$B_2 = \bigcup\nolimits_{j\in[N]} \left(W_{j,\{1,2\}}^{(1)}, W_{j,\{2,3\}}^{(1)}, W_{j,\{2,4\}}^{(1)}, W_{j,\{2\}}^{(2)}\right), \tag{5.45b}$$

Table 5.3: Decoding the message with $x_2^{\lambda_i n}$ at TS i, for $i = 1, ..., 4$

TS number	Sufficient conditions
1	$R^{(2)}_{\{3\}} \le \lambda_1 C^{\bar{\rho}_1 P}_{\sigma_2^2}$ $R^{(2)}_{\{2\}} \le \lambda_1 C^{\bar{\rho}_1 P}_{\sigma_3^2}$
2	$R^{(2)}_{\{4\}} \le \lambda_2 C^{\bar{\rho}_2 P}_{\sigma_2^2}$ $R^{(2)}_{\{2\}} \le \lambda_2 C^{\bar{\rho}_2 P}_{\sigma_4^2}$
3	$R^{(2)}_{\{4\}} \le \lambda_3 C^{\bar{\rho}_3 P}_{\sigma_3^2}$ $R^{(2)}_{\{3\}} \le \lambda_3 C^{\bar{\rho}_3 P}_{\sigma_4^2}$
4	$R^{(3)}_{\{4\}} \le \lambda_4 C^{\bar{\rho}_4 P}_{\sigma_3^2}$ $R^{(3)}_{\{3\}} \le \lambda_4 C^{\bar{\rho}_4 P}_{\sigma_4^2}$

$$B_3 = \bigcup\nolimits_{j \in [N]} \left(W^{(1)}_{j,\{1,3\}}, W^{(1)}_{j,\{2,3\}}, W^{(1)}_{j,\{3,4\}}, W^{(2)}_{j,\{3\}}, W^{(3)}_{j,\{3\}} \right), \tag{5.45c}$$

$$B_4 = \bigcup\nolimits_{j \in [N]} \left(W^{(1)}_{j,\{1,4\}}, W^{(1)}_{j,\{2,4\}}, W^{(1)}_{j,\{3,4\}}, W^{(2)}_{j,\{4\}}, W^{(3)}_{j,\{4\}} \right), \tag{5.45d}$$

where the total cache size in the system is

$$M_{\text{tot}} = N \left(2R^{(1)} + R^{(2)} + R^{(3)} \right). \tag{5.46}$$

For a demand vector $\mathbf{d}$ in the delivery phase, we generate coded packet $W^{(l)}_{\mathcal{S}^{r_l+1}_{\mathcal{K}_l,i}}$, for $i \in \left[\binom{5-l}{r_l+1}\right]$ and $l \in [3]$, as given in (5.35). The transmission is performed via 4 orthogonal TSs, where the i-th TS is of length $\lambda_i n$ channel uses, for $i \in [4]$, such that $\sum_{i=1}^{4} \lambda_i = 1$. After generating codebooks $\mathcal{A}_1$ and $\mathcal{A}_2$ in the i-th TS as defined in (5.37), for $i \in [4]$, the codeword outlined in Table 5.1 is sent over the channel via each TS. In TS i, the users, which the message with $x_1^{\lambda_i n}$ is targeted to, decode the message with $x_1^{\lambda_i n}$ while considering $x_2^{\lambda_i n}$ as noise, for $i \in [4]$. The sufficient conditions, for which the message with $x_1^{\lambda_i n}$ can be decoded successfully by each targeted user, for n large enough, at TS i are summarized in Table 5.2, for $i \in [4]$, thanks to the side information available at users' caches. We note that each user, for which the message with codeword $x_2^{\lambda_i n}$ is targeted to, can decode the message with $x_1^{\lambda_i n}$ having the conditions in Table 5.2 satisfied, for $i \in [4]$. Bearing this in mind, the sufficient conditions such that the message with $x_2^{\lambda_i n}$ is decoded successfully by the intended users are outlined in Table 5.3, for $i \in [4]$. We note that the conditions in Table 5.2

and Table 5.3 guarantee the achievability of the rate tuple presented in Theorem 5.1 for the corresponding total cache size M_{tot} given in (5.46), which is equivalent to the one in (5.13d) for the CSC scheme with $r = 2$, for some $\boldsymbol{\rho}$ and $\boldsymbol{\lambda}$ satisfying (5.13e)-(5.13g). We highlight that, as it can be seen in Table 5.1, in the CSC scheme coded packets are transmitted in different TSs, while at each TS two coded packets are superposed for the transmission, one of which is targeted for a subset of the users receiving the other coded packet.

5.6.3 CDPC Scheme

In the following, we investigate the delivery via TS i, for $i \in [1 + g_{l-1} : g_l]$ and $l \in [K-t]$. The codebook of the transmission with the CDPC scheme is also generated from the linear superposition of two subcodebooks. The subcodebook $\mathcal{A}_2$, given in (5.37b), is generated from i.i.d. codewords $x_2^{\lambda_i n}$, each according to distribution $X_2 \sim \mathcal{N}(0, \bar{\rho}_i P)$, used to send the coded packet $W^{(l+1)}_{\mathcal{S}^r_{\mathcal{K}_{l+1}, i-g_{l-1}}}$ of rate $R^{(l+1)}_{\text{XOR}, \mathcal{S}^r_{\mathcal{K}_{l+1}, i-g_{l-1}}}$. By treating $x_2^{\lambda_i n}$ as interference for user l, knowing $X_2^{\lambda_i}$ non-causally at the server, subcodebook $\mathcal{A}_1$ is generated using dirty paper coding [148]. The auxiliary random variable with the dirty paper coding is set as $Q = X_1 + \tau X_2$, where $\tau = \rho_i P / \left(\rho_i P + \sigma_l^2\right)$, and $X_1 \sim \mathcal{N}(0, \rho_i P)$ is independent of X_2. We extend the codebook generation, encoding, and decoding techniques of the Gelfand-Pinkser scheme for point-to-point transmission presented in the proof of [146, Theorem 7.3] to the transmission in the i-th TS of the setting under consideration, for $i \in [1 + g_{l-1} : g_l]$ and $l \in [K - r]$. We define a message tuple $\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i}} \triangleq \left(w^{(1)}_{d_{k_1}, \mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}, \text{for } k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1, i} \right)$, which concatenates $r + 1$ messages $w^{(1)}_{d_{k_1}, \mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}$, $\forall k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1, i}$, where $w^{(1)}_{d_{k_1}, \mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}$ is uniformly distributed over $\left[2^{nR^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}} \right]$ and represents the message used to send subfile $W^{(1)}_{d_{k_1}, \mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}$, for $k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1, i}$. For each realization of $\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i}}$, we generate a subcodebook $\mathcal{A}_1 \left(\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i}} \right)$ of $2^{\lambda_i n \tilde{R} - n \sum_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1, i}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}}$ sequences $q^{\lambda_i n}(m)$, for $m \in \left[2^{\lambda_i n \tilde{R} - n \sum_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1, i}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i} \backslash \{k_1\}}} \right]$.

Given $x_2^{\lambda_i n}$, in order to send $r + 1$ messages with message tuple $\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i}}$ jointly, we find a sequence $q^{\lambda_i n}(m) \in \mathcal{A}_1 \left(\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1, i}} \right)$ that is jointly typical with $x_2^{\lambda_i n}$ and represent the corresponding codeword, which is to be sent over the channel, by

$$x_1^{\lambda_i n}\left(\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}}, x_2^{\lambda_i n}\right). \tag{5.47}$$

The server then sends the following linearly superposed codeword over the Gaussian BC:

$$x_1^{\lambda_i n}\left(\left(W^{(1)}_{d_{k_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{k_1\}}, \text{for } k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\right), x_2^{\lambda_i n}\right) + x_2^{\lambda_i n}\left(W^{(l+1)}_{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}}\right). \tag{5.48}$$

User $k_2 \in \mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}$ first decodes the message with codeword $x_2^{\lambda_i n}$, while considering $x_1^{\lambda_i n}$ as noise, which by the same analysis as the CSC scheme, one can obtain the necessary condition for a successful decoding as follows:

$$R^{(l+1)}_{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}\setminus\{k_2\}} \leq \lambda_i C^{\bar{\rho}_i P}_{\rho_i P+\sigma^2_{k_2}}, \quad \text{for } i \in [1+g_{l-1} : g_l] \text{ and } l \in [K-r]. \tag{5.49}$$

On the other hand, to decode $r+1$ messages with $x_1^{\lambda_i n}$, upon receiving $y_{k_1}^{\lambda_i n}$, user k_1, for $k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$, declares that $r+1$ messages $\hat{w}^{(1)}_{d_{\tilde{k}_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{\tilde{k}_1\}} \in \left[2^{nR^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{\tilde{k}_1\}}}\right]$, $\forall \tilde{k}_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$, are sent if $\hat{\mathbf{w}}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}} \triangleq \left(\hat{w}^{(1)}_{d_{\tilde{k}_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{\tilde{k}_1\}}, \text{for } \tilde{k}_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\right)$ is the unique message tuple such that $q^{\lambda_i n}(m)$ and $y_{k_1}^{\lambda_i n}$ are jointly typical, for some $m \in \mathcal{A}_1\left(\hat{\mathbf{w}}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}}\right)$, where message $w^{(1)}_{d_{\tilde{k}_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{\tilde{k}_1\}}$ is decoded as $\hat{w}^{(1)}_{d_{\tilde{k}_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\setminus\{\tilde{k}_1\}}$, for $\tilde{k}_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$.[3] Here we note again that, for $l \in [K-r]$,

$$\mathcal{S}^{r+1}_{\mathcal{K}_1,i} = \left\{\mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}, l\right\}, \quad \text{for } i \in [1+g_{l-1} : g_l]. \tag{5.50}$$

We assume without loss of generality that the message tuple $\mathbf{w}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}} = (1,\ldots,1)$ is sent with $x_1^{\lambda_i n}$. The decoder at user $l \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$ makes an error, if one or both of the following events occur:

$$\mathcal{E}^1_l = \left\{Q^{\lambda_i n}(m) \text{ and } X_2^{\lambda_i n} \text{ are not jointly typical}, \forall Q^{\lambda_i n}(m) \in \mathcal{A}_1(1,\ldots,1)\right\}, \tag{5.51a}$$

$$\mathcal{E}^2_l = \left\{Q^{\lambda_i n}(m) \text{ and } Y_l^{\lambda_i n} \text{ are jointly typical, for some } Q^{\lambda_i n}(m) \notin \mathcal{A}_1(1,\ldots,1)\right\}. \tag{5.51b}$$

[3]For ease of notation, we drop the dependency of channel outputs $Y_1^n(\mathbf{W}),\ldots,Y_K^n(\mathbf{W})$ on $\mathbf{W}$.

According to [146, Lemma 3.3], $\Pr\left\{\mathcal{E}_l^1\right\}$ tends to zero, if, for n large enough,

$$\lambda_i \tilde{R} - \sum\nolimits_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} \geq \lambda_i I\left(Q; X_2\right). \tag{5.52}$$

Furthermore, since user $l \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$ has access to all r messages $W^{(1)}_{d_k,\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}}$, $\forall k \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}$, in its cache, it knows that $w^{(1)}_{d_k,\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}} = 1$, $\forall k \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}$, and $\Pr\left\{\mathcal{E}_l^2\right\}$ tends to zero, if, for n large enough,

$$\lambda_i \tilde{R} - \sum\nolimits_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}} \leq \lambda_i I\left(Q; Y_l\right), \tag{5.53}$$

where $Y_k = X_1 + X_2 + Z_k$, where $Z_k \sim \mathcal{N}\left(0, \sigma_k^2\right)$, for $k \in [K]$. Combining (5.52) and (5.53), we obtain that user $l \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$ decodes the message with $x_1^{\lambda_i n}$ successfully, if

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}} \leq \lambda_i \left(I\left(Q; Y_l\right) - I\left(Q; X_2\right)\right), \tag{5.54}$$

which is equivalent to

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\}} \leq \lambda_i C^{\rho_i P}_{\sigma_l^2}. \tag{5.55}$$

Now we investigate the sufficient conditions for which users in $\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{l\} = \mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}$ can decode the message with $x_1^{\lambda_i n}$. We note that having the conditions in (5.49) satisfied, each user in $\mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}$ can decode the message with $x_2^{\lambda_i n}$. The decoder at user k, for $k \in \mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}$, makes an error, if one or both of the following events occur:

$$\mathcal{E}_k^1 = \left\{Q^{\lambda_i n}(m) \text{ and } X_2^{\lambda_i n} \text{ are not jointly typical, } \forall Q^{\lambda_i n}(m) \in \mathcal{A}_1(1,\ldots,1)\right\}, \tag{5.56a}$$

$$\mathcal{E}_k^2 = \left\{Q^{\lambda_i n}(m) \text{ and } \left(Y_k^{\lambda_i n}, X_2^{\lambda_i n}\right) \text{ are jointly typical, for } Q^{\lambda_i n}(m) \notin \mathcal{A}_1(1,\ldots,1)\right\}. \tag{5.56b}$$

$\Pr\left\{\mathcal{E}_k^1\right\}$ tends to zero, if, for n large enough,

$$\lambda_i \tilde{R} - \sum\nolimits_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} \geq \lambda_i I\left(Q; X_2\right), \quad \text{for } k \in \mathcal{S}^r_{\mathcal{K}_{l+1},i-g_{l-1}}. \tag{5.57}$$

Furthermore, since user k, for $k \in \mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}$ has access to all r messages $W^{(1)}_{d_{k_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}}$, $\forall k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}$, in its cache, it knows that $w^{(1)}_{d_{k_1},\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} = 1$, $\forall k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}$, and $\Pr\left\{\mathcal{E}^{2}_{k}\right\}$ tends to zero, if, for n large enough,

$$\lambda_i \tilde{R} - \sum\nolimits_{k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}} R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} \leq \lambda_i I\left(Q; Y_l, X_2\right). \tag{5.58}$$

Combining (5.57) and (5.58), we obtain that user k, for $k \in \mathcal{S}^{r}_{\mathcal{K}_{l+1},i-g_{l-1}}$ decodes the message with $x_1^{\lambda_i n}$ successfully, if

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}} \leq \lambda_i \left(I\left(Q; Y_k, X_2\right) - I\left(Q; X_2\right)\right) = \lambda_i I\left(Q; Y_k \left| X_2\right.\right), \tag{5.59}$$

which leads to

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k\}} \leq \lambda_i C^{\rho_i P}_{\sigma_k^2}. \tag{5.60}$$

By combining the conditions in (5.55) and (5.60), we conclude that, at TS i, user k_1, for $k_1 \in \mathcal{S}^{r+1}_{\mathcal{K}_1,i}$, can decode the message with $x_1^{\lambda_i n}$ successfully, if

$$R^{(1)}_{\mathcal{S}^{r+1}_{\mathcal{K}_1,i}\backslash\{k_1\}} \leq \lambda_i C^{\rho_i P}_{\sigma_{k_1}^2}, \quad \text{for } i \in [1 + g_{l-1} : g_l] \text{ and } l \in [K - r]. \tag{5.61}$$

Having the conditions in (5.49) and (5.61) satisfied, the achievability of the rate tuple for the corresponding total cache size M presented in Theorem 5.1 for the CDPC scheme is proved.

5.7 Outer Bound

In the following, we develop an outer bound on the capacity region $\mathcal{C}(P, M_{\text{tot}})$ constrained to uncoded caching in the placement phase.

Theorem 5.2. *Consider the system described in Section 5.3 with average power P, where user k has a cache size of M_k, $k \in [K]$. If the placement phase is constrained to*

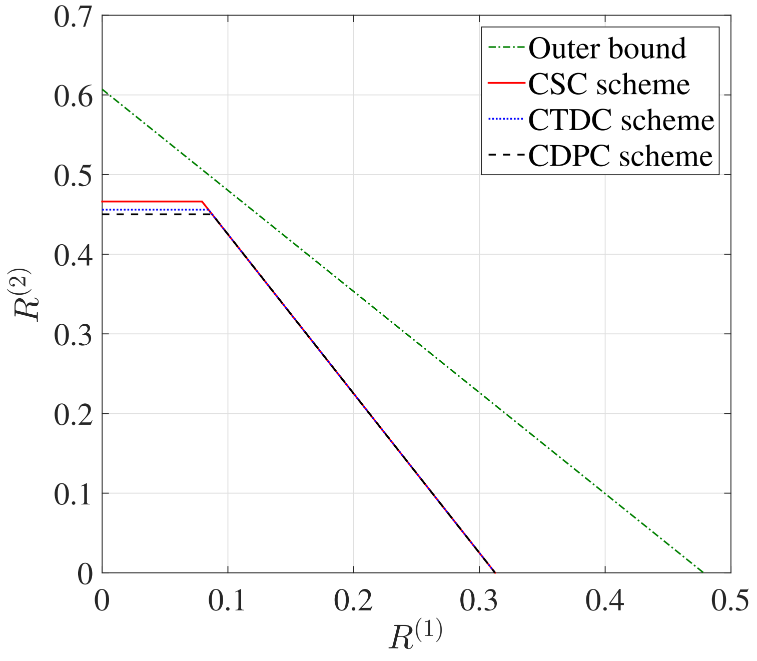


FIGURE 5.1: Achievable rate pair $\left(R^{(1)}, R^{(2)}\right)$ for a caching system with $K = N = 4$, and $M_{\text{tot}} = 2.5$, where $R^{(3)} = 0$, $r = 2$ and $r_1 = 2$, $r_2 = r_3 = 1$, and $r_4 = 0$. The noise variance at user i is $\sigma_i^2 = 5 - i$, for $k \in [4]$, and we set $P = 2$.

uncoded caching, for any non-empty subset $\mathcal{G} \subset [K]$, we have, for $k = 1, \dots, |\mathcal{G}|$,

$$R_{\pi_{\mathcal{G}}(k)} \leq C^{\eta_{\pi_{\mathcal{G}}(k)}P}_{\sum_{i=k+1}^{|\mathcal{G}|}\eta_{\pi_{\mathcal{G}}(i)}P+\sigma^2_{\pi_{\mathcal{G}}(k)}} + \frac{1}{N}\sum_{i=1}^{k} M_{\pi_{\mathcal{G}}(i)}, \tag{5.62}$$

for some non-negative coefficients $\eta_{\pi_{\mathcal{G}}(1)}, \dots, \eta_{\pi_{\mathcal{G}}(|\mathcal{G}|)}$, such that $\sum_{i=1}^{|\mathcal{G}|} \eta_{\pi_{\mathcal{G}}(i)} = 1$, where $\pi_{\mathcal{G}}$ is a permutation of the elements of $\mathcal{G}$, such that $\sigma^2_{\pi_{\mathcal{G}}(1)} \geq \sigma^2_{\pi_{\mathcal{G}}(2)} \geq \cdots \geq \sigma^2_{\pi_{\mathcal{G}}(|\mathcal{G}|)}$.

Proof. The proof is presented in Appendix D.1. □

5.8 Numerical Results

In this section, we compare the achievable rate regions of the CTDC, CSC, and CDPC schemes for a caching system with $K = N = 4$. We set the average power constraint to $P = 2$, and the noise variance at user i is assumed to be $\sigma_i^2 = 5 - k$, for $i \in [4]$. We assume a total cache size of $M_{\text{tot}} = 2.5$.

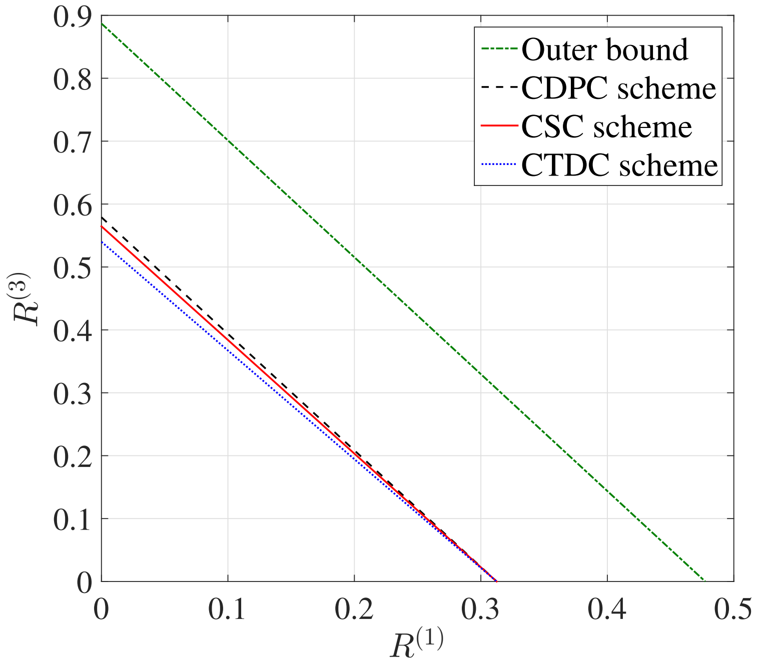


FIGURE 5.2: Achievable rate pair $\left(R^{(1)}, R^{(3)}\right)$ for a caching system with $K = N = 4$, and $M_{\text{tot}} = 2.5$, where $R^{(2)} = 0$, and $r = 2$ and $r_1 = 2$, $r_2 = r_3 = 1$, and $r_4 = 0$. The noise variance at user k is $\sigma_i^2 = 5 - i$, for $i \in [4]$, and we set $P = 2$.

We evaluate the performance in terms of the rate of different layers of the files, i.e., $R^{(1)}, \ldots, R^{(K)}$, where $R_i = \sum_{l=1}^{i} R^{(l)}$, for $i \in [K]$. We examine the performance of the CSC and CDPC schemes for $r = 2$. Thus, the achievable rate tuple (R_1, R_2, R_3, R_4) presented in Theorem 5.1 can be achieved by the CSC and CDPC schemes, for $s_1 = 0$, $s_2 = 1$, and $s_1 = 1$, $s_2 = 0$, respectively, where $R_4 = R_3$ since $R^{(4)} = 0$. The boundary surface of the rate region achieved by the CSC and CDPC schemes are computed through the optimization problem given in (5.15). For fairness of comparison, we consider caching factors $r_1 = 2$, $r_2 = r_3 = 1$, and $r_4 = 0$. The boundary of the rate region achievable by CTDC can be calculated by the optimization problem in (5.11), where, in order to have a fair comparison, we set $\lambda^{(4)} = 0$ leading to $R^{(4)} = 0$ and $R_4 = R_3$.

We investigate the convex hull of the achievable rate tuples calculated by the optimization problem corresponding to each of the CTDC, CSC, and CDPC schemes. Since the presentation of the three-dimensional rate region together with the outer bound does not provide a clear picture, here we fix one of the rates $R^{(1)}$, $R^{(2)}$ and $R^{(3)}$ and present the rate region on the two-dimensional planes corresponding to the

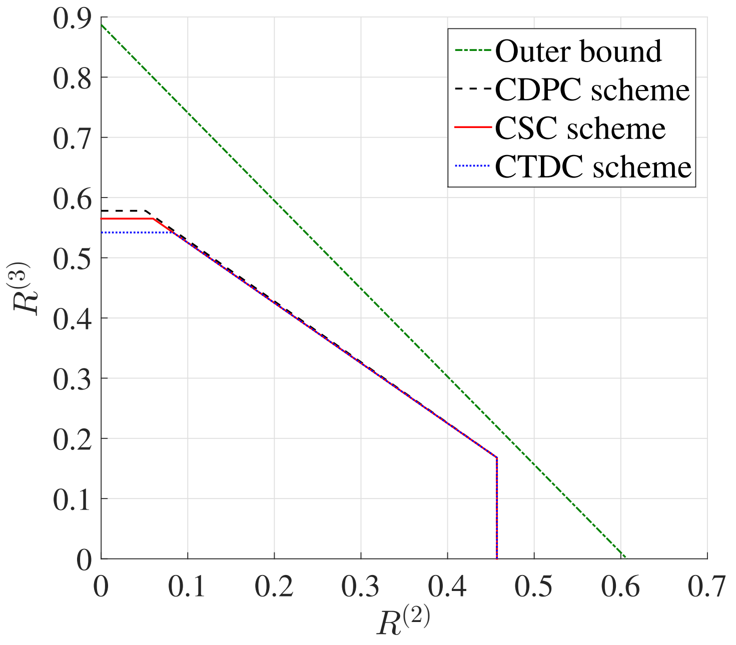


FIGURE 5.3: Achievable rate pair $\left(R^{(2)}, R^{(3)}\right)$ for a caching system with $K = N = 4$, and $M_{\text{tot}} = 2.5$, where $R^{(1)} = 0$, and $r = 2$ and $r_1 = 2$, $r_2 = r_3 = 1$, and $r_4 = 0$. The noise variance at user i is $\sigma_i^2 = 5 - i$, for $i \in [4]$, and we set $P = 2$.

other two rates. Two-dimensional plane of $\left(R^{(1)}, R^{(2)}\right)$, $\left(R^{(1)}, R^{(3)}\right)$ and $\left(R^{(2)}, R^{(3)}\right)$ for $R^{(3)} = 0$, $R^{(2)} = 0$ and $R^{(1)} = 0$ are illustrated in Figures 5.1, 5.2 and 5.3, respectively, together with the outer bound presented in Theorem 5.2. As it can be seen from the figures, for relatively small values of $R^{(1)}$, the CSC and CTDC schemes achieve higher values of $R^{(2)}$, while the CSC scheme outperforms the latter. For higher values of $R^{(1)}$, the improvement of the CSC scheme over CTDC and CDPC is negligible. For a fixed $R^{(1)}$ value, CDPC achieves higher values of $R^{(3)}$ compared to the other two achievable schemes, and CSC outperforms CTDC. On the other hand, given a relatively small value of $R^{(2)}$, CDPC improves upon the CSC and CTDC in terms of the achievable rate $R^{(3)}$, and CSC achieves higher values of $R^{(3)}$ than CTDC. Observe that the outer bound is not tight in general; however, for any achievable rate tuple $(R_1, \dots, R_4)$, which is achieved with a specific cache allocation $M_1, \dots, M_4$, the outer bound specialized to this cache allocation would be tighter.

We would like to point out that the reason of the relatively small gap between the CTDC scheme with time-division transmission, and CSC and CDPC introducing superposition and dirty paper coding, respectively, is due to the time-division nature of the CSC and CDPC schemes. With CSC (CDPC) scheme, coded packets are delivered in different TSs, where within each TS, two coded packets are superposed (dirty paper coded).

5.9 Conclusions

In this chapter we have studied cache-aided content delivery over a Gaussian BC, where each user is allowed to demand a file at a distinct rate. To model this asymmetry, we have assumed that the files are encoded into K layers corresponding to K users in the system, such that the k-th worst user is delivered only the k layers of its demand, $k \in [K]$. We have considered a centralized placement phase, where the server knows the channel qualities of the links in the delivery phase in addition to the identity of the users. By allowing the users to have different cache capacities, we have defined the capacity region for a total cache size. We designed a placement phase through cache allocation across the users and the files' layers to maximize the rates allocated to different layers. We have proposed three achievable schemes, which deliver coded multicast packets, generated thanks to the contents carefully cached during the placement phase, through different channel coding techniques over the Gaussian BC. Although the coded multicast packets are intended for a set of users with distinct link capacities, channel coding techniques can be employed to deliver requested files such that the users with better channels achieve higher rates. We have also developed an outer bound on the capacity region assuming uncoded caching. We are currently working to reduce the gap between the inner and outer bounds.

Chapter 6

Distributed Computing

6.1 Overview

In this chapter we study collaborative ML where K computing servers, called *workers*, carry out a learning problem distributively with the help of a remote PS. We study computation of an arbitrary function over a dataset through K workers, where the tasks are assigned to the workers by the PS. We investigate scheduling of computation tasks across these K workers considering sequential computation of tasks assigned to a worker, where the result of each computation is sent to the master right after its completion. Each computation round, which can model an iteration of the stochastic gradient descent (SGD) algorithm, is *completed* once the master receives k distinct computations, referred to as the *computation target.* Our goal is to characterize the *average completion time* as a function of the *computation load*, which denotes the portion of the dataset available at each worker, and the computation target. We propose two computation scheduling schemes that specify the tasks assigned to each worker, as well as their computation schedule, i.e., the order of execution. Assuming a general statistical model for computation and communication delays, we derive the average completion time of the proposed schemes. We also establish a lower bound on the minimum average completion time by assuming prior knowledge of the random delays. Experimental results carried out on Amazon EC2 cluster show a significant reduction in the average completion time over existing coded and uncoded computing schemes.

6.2 Introduction

In this chapter we study distributed computation of an arbitrary function, and introduce a centralized scheduling strategy, where computation tasks are assigned to the workers by the PS. While significant research efforts have been invested in designing coded computation [98–107] techniques, we argue in this chapter that uncoded computing and communication can be even more effective in tackling stragglers and reducing the average computation time. Each worker can compute a limited number of tasks, referred to as the *computation load*. Computations are carried out sequentially, and the result of each computation is sent to the master right after it is completed. Communication delay from the workers to the master is also taken into account. We assume that both the computation and communication delays are independent across the workers since they have different computation capabilities with various dynamic behaviours of processing speed, and they communicate with the PS over different links experiencing distinct mediums, but may be correlated for different tasks carried out at the same worker. This sequential computation and communication framework allows the master to exploit partial computations by slow workers. The main computation is assumed to be completed when the master receives sufficient number of distinct computations from the workers, referred to as the *computation target*. In other words, computation target specifies the number of distinct computations the master is required to receive from the workers to complete the main computation task. Unlike coded computation, uncoded computing approach does not introduce any encoding and decoding delays and complexities. It also allows partial decoding, which can be exploited to reduce the communication load for distributed learning [114–116]. Assuming that the computation and communication delays are random variables, our goal is to characterize the minimum *average completion time* as a function of the computation load and computation target. We first provide a generic expression for the average completion time as a function of the *computation schedule*, which specifies both the tasks assigned to each worker and their computation order. We propose two different computation scheduling schemes, and obtain closed-form expressions for their average completion times for a general statistical model of the random delays, which upper bound the minimum average completion time. We also establish a lower bound on the minimum average completion time. The experiments on Amazon EC2 cluster illustrate a substantial reduction in the average completion time with the proposed uncoded computing schemes with task scheduling compared to coded computation schemes and uncoded computation without scheduling of the tasks at the workers [112].

The remainder of this chapter is organized as follows. In Section 6.3 we present the system model. In Section 6.4 we provide a generic characterization of the minimum average completion time. We then provide upper and lower bounds on the minimum average completion time In Sections 6.5 and 6.6, respectively. In Section 6.7. We finally conclude this chapter in Section 6.8.

6.3 System Model

We consider distributed computation of a function h over a dataset $\{\Lambda_1, ..., \Lambda_K\}$ across K workers. Function $h : \mathbb{V} \to \mathbb{U}$ is an arbitrary function, where $\mathbb{V}$ and $\mathbb{U}$ are two vector spaces over the same field $\mathbb{F}$, and sub-data Λ_i is an element of $\mathbb{V}$, $i \in [K]$. The dataset $\{\Lambda_1, ..., \Lambda_K\}$ is distributed across the workers by the master, and a maximum number of $L \leq K$ sub-data are assigned to each worker, referred to as the *computation load.* We denote by $\mathcal{E}_i$ the indices of the sub-data assigned to worker i, $i \in [K]$, where $\mathcal{E}_i \subset [K]$, $|\mathcal{E}_i| \leq L$.

The computations of the tasks assigned to each worker are carried out sequentially. We define the task ordering (TO) matrix $\mathbf{C}$ as an $K \times L$ matrix of integers, $C \in [K]^{K \times L}$, specifying the assignment of the tasks to the workers $\boldsymbol{\mathcal{E}} \triangleq \{\mathcal{E}_i\}_{i=1}^K$, as well as the order these tasks are carried out by each worker $\boldsymbol{\mathcal{O}} \triangleq \{\mathcal{O}_i\}_{i=1}^K$, where $\mathcal{O}_i$ denotes the computing order of the tasks assigned to worker i, $i \in [K]$. Each row of matrix $\mathbf{C}$ corresponds to a different worker, and its elements from left to right represent the order of computations. That is, the (i, j)-th entry of $\mathbf{C}$, $\mathbf{C}(i, j) \in \mathcal{E}_i$, denotes the index of the element of the dataset that is computed by worker i as its j-th computation, i.e., worker i first computes $h(\Lambda_{\mathbf{C}(i,1)})$, then computes $h(\Lambda_{\mathbf{C}(i,2)})$, and so on so forth until either it computes $h(\Lambda_{\mathbf{C}(i,r)})$, or it receives the acknowledgement message from the master, and stops computations, $i \in [K]$, $j \in [L]$. Note that the task assignment $\boldsymbol{\mathcal{E}}$ and the order of computations $\boldsymbol{\mathcal{O}}$ are specified by a unique TO matrix $\mathbf{C}$. While any

$\mathbf{C}$ matrix is a valid TO matrix, it is easy to see that the optimal TO matrix will have L distinct entries in each of its rows.

The computations start at time $t = 0$ at all the workers, and each worker sends the result of each assigned task to the master right after its computation. We denote the time worker i spends to compute $h(\Lambda_j)$ by $T_{i,j}^{(1)}$, and the communication delay for sending $h(\Lambda_j)$ to the master by $T_{i,j}^{(2)}$, $j \in \mathcal{E}_i$, $i \in [K]$. Thus, the total delay of receiving $h(\Lambda_j)$ from worker i is $T_{i,j}^{(1)} + T_{i,j}^{(2)}$, $j \in \mathcal{E}_i$, $i \in [K]$. If $j \notin \mathcal{E}_i$, we set $T_{i,j}^{(l)} = \infty$, $\forall l \in [2]$, $i \in [K]$. We assume that the computation and communication delays, $T_{i,j}^{(1)}$ and $T_{i,j}^{(2)}$, $\forall i, j \in [K]$, are independent. We further assume that computation (communication) delays at different workers are independent. On the other hand, the computation (communication) delays associated with the tasks at the same worker can be dependent, and we denote the joint cumulative distribution function (CDF) of $T_{i,1}^{(l)}, \dots, T_{i,K}^{(l)}$ by $F_{i,[K]}^{(l)}$, and the joint probability density function (PDF) by $f_{i,[K]}^{(l)}$, $i \in [K]$, $l \in [2]$. We note that the statistical model of the computation (communication) delays at each worker do not depend on any specific order of computing (communicating) tasks, since we assume that the size and complexity of computing (communicating) each sub-data (computation) is the same.

Let $t_{i,j}$ denote the time the master receives $h(\Lambda_j)$ from worker i, for $i, j \in [K]$, where we set $t_{i,m} = \infty$ if $m \notin \mathcal{E}_i$. Then, the total computation delay of computing $h(\Lambda_{\mathbf{C}(i,1)})$, $h(\Lambda_{\mathbf{C}(i,2)}), \dots, h(\Lambda_{\mathbf{C}(i,j)})$ sequentially plus the communication delay for receiving $h(\Lambda_{\mathbf{C}(i,j)})$ is

$$t_{i,\mathbf{C}(i,j)} = \sum\nolimits_{m=1}^{j} T_{i,\mathbf{C}(i,m)}^{(1)} + T_{i,\mathbf{C}(i,j)}^{(2)}, \quad i, j \in [K], \tag{6.1}$$

As a result, the master receives computation $h(\Lambda_j)$ at time

$$t_j \triangleq \min\nolimits_{i \in [K]} \{t_{i,j}\}, \quad j \in [K] \tag{6.2}$$

where the minimization is over the workers.

For any TO matrix, the computation is considered completed once the master recovers $\tilde{K}$ distinct tasks, referred to as the *computation target.* We allow partial computations, i.e., $\tilde{K}$ can be smaller than K. Once the computation target is met, the master sends an acknowledgement message to all the workers to stop computations. Given the TO matrix $\mathbf{C}$, we denote the *completion time*; that is, the time it takes the master to receive $\tilde{K}$ distinct computations, by $t_{\mathbf{C}}(L, \tilde{K})$, which is a random variable. We define the average completion time as

$$\bar{t}_{\mathbf{C}}(L, \tilde{K}) \triangleq \mathbb{E}\left[t_{\mathbf{C}}(L, \tilde{K})\right], \tag{6.3}$$

where the randomness is due to the delays. We define the minimum average completion time

$$\bar{t}^*(L, \tilde{K}) \triangleq \min_{\mathbf{C}} \left\{\bar{t}_{\mathbf{C}}(L, \tilde{K})\right\}, \tag{6.4}$$

where the minimization is taken over all possible TO matrices $\mathbf{C}$. The goal is to characterize $\bar{t}^*(L, \tilde{K})$.

Remark 6.3.1. We have defined each $\Lambda_i \in \mathbb{V}$ as a single sub-data, and assumed that the result of $h(\Lambda_i)$ at a worker is transmitted immediately to the master. It is possible to generalize this model by considering N sub-data instead, with $N \gg K$, and grouping them into K mini-batches, such that each Λ_i in our model corresponds to a mini-batch of $\lceil N/K \rceil$ sub-data. A worker sends the average of the gradients for all the sub-data in a mini-batch after computing all of them. For a mini-batch size of c sub-data, this corresponds to communicating once every c computations.

Remark 6.3.2. Most coded computation schemes in the literature, mainly targeting DGD, require the master to recover the gradients (or, their average) for the whole dataset at each iteration. However, convergence of SGD is guaranteed even if the gradient is computed for a random portion of the dataset at each iteration [114,116–118, 127,129,149,150]. This is indeed the case for the random straggling model considered here with $k < n$, where the straggling workers; hence, the uncomputed gradients, vary at each iteration.

Remark 6.3.3. The problem under consideration is similar to the well-known job scheduling problem [151], in which a set of tasks are to be executed by multiple workers given a partial ordering of task execution and the delay associated with each task. The goal is to find a schedule minimizing the total delay, which is shown to be NP-complete [152]. This problem has been studied under different constraints for different applications, such as cloud computing [153–155], edge computing [156, 157], and dispersed computing [158, 159]. Our problem differs from the job scheduling one, since no ordering of task execution is imposed, and each task can be executed by an arbitrary number of workers. Also, in our model, the scheduling is designed without having any prior knowledge about the computation and communication delays of the tasks.

6.4 Average Completion Time Analysis

Here we analyze the average completion time $\bar{t}_{\mathbf{C}}(L, \tilde{K})$ for a given TO matrix $\mathbf{C}$.

Theorem 6.1. *For a given TO matrix* $\mathbf{C}$*, we have*

$$\begin{aligned}
&\Pr\left\{t_{\mathbf{C}}(L,\tilde{K}) > t\right\} = 1 - F_{t_{\mathbf{C}}}(t) \\
&= \sum\nolimits_{i=K-\tilde{K}+1}^{K} (-1)^{K-\tilde{K}+i+1} \binom{i-1}{K-\tilde{K}} \sum\nolimits_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} \Pr\left\{t_j > t, \forall j \in \mathcal{S}\right\}, \qquad (6.5)
\end{aligned}$$

which yields

$$\bar{t}_{\mathbf{C}}(L,\tilde{K}) = \sum_{i=K-\tilde{K}+1}^{K} (-1)^{K-\tilde{K}+i+1} \binom{i-1}{K-\tilde{K}} \sum_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} \int_0^{\infty} \Pr\left\{t_j > t, \forall j \in \mathcal{S}\right\} dt. \qquad (6.6)$$

Note that the dependence of the completion time on the TO matrix in (6.5) and (6.6) is through the statistics of t_j.

Proof. The event $\left\{t_{\mathbf{C}}(L,\tilde{K}) > t\right\}$ is equivalent to the union of the events, for which the time to complete any arbitrary set of at least $K - \tilde{K} + 1$ distinct computations is greater than t, i.e.,

$$\Pr\Big\{t_{\mathbf{C}}(L,\tilde{K}) > t\Big\} = \Pr\bigg\{\bigcup\nolimits_{\mathcal{G}\subset[K]:K-\tilde{K}+1\le|\mathcal{G}|\le K}\Big\{t_j > t, t_{j'} \le t, \forall j \in \mathcal{G}, \forall j' \in \mathcal{G}'\Big\}\bigg\}, \tag{6.7}$$

where we define $\mathcal{G}' \triangleq [K]\backslash\mathcal{G}$. Since the events $\big\{t_j > t, t_{j'} \le t, \forall j \in \mathcal{G}, \forall j' \in \mathcal{G}'\big\}$, for all distinct sets $\mathcal{G} \subset [K]$, are mutually exclusive (pairwise disjoint), we have

$$\begin{aligned}\Pr\Big\{t_{\mathbf{C}}(L,\tilde{K}) > t\Big\} &= \sum\nolimits_{i=K-\tilde{K}+1}^{K}\sum\nolimits_{\mathcal{G}\subset[K]:|\mathcal{G}|=i}\Pr\Big\{t_j > t, t_{j'} \le t, \forall j \in \mathcal{G}, \forall j' \in \mathcal{G}'\Big\}\\ &= \sum\nolimits_{i=K-\tilde{K}+1}^{K}\sum\nolimits_{\mathcal{G}\subset[K]:|\mathcal{G}|=i} H_{\mathcal{G},\mathcal{G}'},\end{aligned} \tag{6.8}$$

where, for $\mathcal{S}_1 \subset [K]$ and $\mathcal{S}_2 \subset [K]$, we define

$$H_{\mathcal{S}_1,\mathcal{S}_2} \triangleq \Pr\left\{t_{j_1} > t, t_{j_2} \le t, \forall j_1 \in \mathcal{S}_1, \forall j_2 \in \mathcal{S}_2\right\}. \tag{6.9}$$

Lemma 6.1. *Given a particular set $\mathcal{G} \subset [K]$, $|\mathcal{G}| = i$, for $i \in [K-\tilde{K}+1 : K]$, we have*

$$\begin{aligned}H_{\mathcal{G},\mathcal{G}'} &= \sum\nolimits_{m=i}^{K}(-1)^{i+m}\sum\nolimits_{\hat{\mathcal{G}}\subset\mathcal{G}':|\hat{\mathcal{G}}|=m-i} H_{\mathcal{G}\cup\hat{\mathcal{G}},\emptyset}\\ &= \sum\nolimits_{m=i}^{K}(-1)^{i+m}\sum\nolimits_{\hat{\mathcal{G}}\subset\mathcal{G}':|\hat{\mathcal{G}}|=m-i}\Pr\Big\{t_j > t, \forall j \in \mathcal{G}\cup\hat{\mathcal{G}}\Big\}.\end{aligned} \tag{6.10}$$

Proof. The proof of Lemma 6.1 can be found in [160, Appendix A], where we use the fact that, for any $g \in \mathcal{G}'$, we have

$$H_{\mathcal{G},\mathcal{G}'} = H_{\mathcal{G},\mathcal{G}'\backslash\{g\}} - H_{\mathcal{G}\cup\{g\},\mathcal{G}'\backslash\{g\}}. \tag{6.11}$$

□

According to Lemma 6.1, for $i \in [K-\tilde{K}+1 : K]$, we have

$$\begin{aligned}&\sum\nolimits_{\mathcal{G}\subset[K]:|\mathcal{G}|=i} H_{\mathcal{G},\mathcal{G}'} = \sum\nolimits_{\mathcal{G}\subset[K]:|\mathcal{G}|=i}\sum\nolimits_{m=i}^{K}(-1)^{i+m}\sum\nolimits_{\hat{\mathcal{G}}\subset\mathcal{G}':|\hat{\mathcal{G}}|=m-i} H_{\mathcal{G}\cup\hat{\mathcal{G}},\emptyset}\\ &= \sum\nolimits_{m=i}^{K}(-1)^{i+m}\sum\nolimits_{\mathcal{G}\subset[K]:|\mathcal{G}|=i}\sum\nolimits_{\hat{\mathcal{G}}\subset\mathcal{G}':|\hat{\mathcal{G}}|=m-i} H_{\mathcal{G}\cup\hat{\mathcal{G}},\emptyset}\\ &\overset{(a)}{=} \sum\nolimits_{m=i}^{K}(-1)^{i+m}\binom{m}{i}\sum\nolimits_{\mathcal{S}\subset[K]:|\mathcal{S}|=m} H_{\mathcal{S},\emptyset},\end{aligned} \tag{6.12}$$

where (a) follows since, for each set $\mathcal{S} = \mathcal{G} \cup \hat{\mathcal{G}}$ with $|\mathcal{S}| = m$, there are $\binom{m}{i}$ sets $\mathcal{G} \cup \hat{\mathcal{G}}$. Plugging (6.12) into (6.8) yields

$$\Pr\left\{t_{\mathbf{C}}(L,\tilde{K}) > t\right\} \sum_{i=K-\tilde{K}+1}^{K} \sum_{m=i}^{K} (-1)^{i+m} \binom{m}{i} \sum_{\mathcal{S}\subset[K]:|\mathcal{S}|=m} H_{\mathcal{S},\emptyset}. \tag{6.13}$$

For a particular set $\mathcal{S} \subset [K]$ with $|\mathcal{S}| = S$, for some $S \in [K-\tilde{K}+1 : K]$, the coefficient of $H_{\mathcal{S},\emptyset}$ in (6.13) is given by

$$\begin{aligned}
&\sum\nolimits_{i=K-\tilde{K}+1}^{S} (-1)^{i+S} \binom{S}{i} = \sum\nolimits_{i=0}^{S} (-1)^{i+S} \binom{S}{i} - \sum\nolimits_{i=0}^{K-\tilde{K}} (-1)^{i+S} \binom{S}{i} \\
&= 0 - (-1)^{K-\tilde{K}+S} \binom{S-1}{K-\tilde{K}} = (-1)^{K-\tilde{K}+S+1} \binom{S-1}{K-\tilde{K}},
\end{aligned} \tag{6.14}$$

which results in

$$\Pr\left\{t_{\mathbf{C}}(L,\tilde{K}) > t\right\} = \sum_{i=K-\tilde{K}+1}^{K} (-1)^{K-\tilde{K}+i+1} \binom{i-1}{K-\tilde{K}} \sum_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} H_{\mathcal{S},\emptyset}. \tag{6.15}$$

According to the definition of $H_{\mathcal{S},\emptyset}$, (6.15) concludes the proof of (6.5). Furthermore, since $t_{\mathbf{C}}(L,\tilde{K}) \geq 0$, we have

$$\bar{t}_{\mathbf{C}}(L,\tilde{K}) = \int_0^\infty (1 - F_{t_{\mathbf{C}}}(t))\, dt, \tag{6.16}$$

which yields the expression in (6.6). □

Remark 6.4.1. For $\tilde{K} = K$, we have

$$\Pr\{t_{\mathbf{C}}(L,K) > t\} = \sum\nolimits_{i=1}^{K} (-1)^{i+1} \sum\nolimits_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} \Pr\{t_j > t, \forall j \in \mathcal{S}\}, \tag{6.17}$$

and

$$\bar{t}_{\mathbf{C}}(L,K) = \sum\nolimits_{i=1}^{K} (-1)^{i+1} \sum\nolimits_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} \int_0^\infty \Pr\{t_j > t, \forall j \in \mathcal{S}\}\, dt. \tag{6.18}$$

The minimum average completion time $\bar{t}^*(L,\tilde{K})$ can be obtained as a solution of the optimization problem $\bar{t}^*(L,\tilde{K}) = \min_{\mathbf{C}} \bar{t}_{\mathbf{C}}(L,\tilde{K})$. Providing a general characterization for $\bar{t}^*(L, \tilde{K})$ is elusive. Next, we will propose two specific computation task assignment and scheduling schemes, and evaluate their average completion times.

6.5 Upper Bounds on the Minimum Average Completion Time

Here we introduce two computation task assignment and scheduling schemes, namely *cyclic scheduling* (CS) and *staircase scheduling* (SS). The average completion time for these schemes will provide upper bounds on $\bar{t}^*(L, \tilde{K})$.

6.5.1 CS Scheme

The CS scheme is motivated by the symmetry across the workers when we have no prior information on their computation speeds. CS makes sure that each computation task has the same order at different workers. This is achieved by a cyclic shift operator. The TO matrix is given by

$$\mathbf{C}_{\mathrm{CS}}(i,j) = g(i+j-1), \quad \text{for } i \in [K] \text{ and } j \in [L], \tag{6.19}$$

where function $g : \mathbb{Z} \to \mathbb{Z}$ is defined as follows:

$$g(m) \triangleq \begin{cases} m, & \text{if } 1 \leq m \leq K, \\ m-K, & \text{if } m \geq K+1, \\ m+K, & \text{if } m \leq 0. \end{cases} \tag{6.20}$$

Thus, we have

$$\mathbf{C}_{\mathrm{CS}} = \begin{bmatrix} g(1) & g(2) & \dots & g(L) \\ g(2) & g(3) & \dots & g(L+1) \\ \vdots & \vdots & \ddots & \vdots \\ g(K) & g(K+1) & \dots & g(K+L-1) \end{bmatrix}, \tag{6.21}$$

which, for $i \in [K]$ and $j \in [L]$, results in

$$t_{i,g(i+j-1)} = \sum\nolimits_{m=1}^{j} T^{(1)}_{i,g(i+m-1)} + T^{(2)}_{i,g(i+j-1)}. \tag{6.22}$$

For $i \in [K]$, we can re-write (6.22) as follows:

$$t_{g(i-j+1),i} = \begin{cases} \sum_{m=1}^{j} T^{(1)}_{g(i-j+1),g(i-j+m)} + T^{(2)}_{g(i-j+1),i}, & \text{if } j \in [L], \\ \infty, & \text{if } j \notin [L], \end{cases} \tag{6.23}$$

which results in

$$t_i = \min_{j\in[L]} \left\{ \sum\nolimits_{m=1}^{j} T^{(1)}_{g(i-j+1),g(i-j+m)} + T^{(2)}_{g(i-j+1),i} \right\}. \tag{6.24}$$

6.5.2 SS Scheme

We can observe that CS imposes the same step size and direction in computations across all the workers. Alternatively, here we propose the SS scheme, which introduces inverse computation orders at the workers. The entries of the TO matrix $\mathbf{C}_{\mathrm{SS}}$ for the SS scheme are given by, for $i \in [K]$, $j \in [L]$,

$$\mathbf{C}_{\mathrm{SS}}(i,j) = g(i + (-1)^{i-1}(j-1)). \tag{6.25}$$

It follows that

$$\mathbf{C}_{\mathrm{SS}} = \begin{bmatrix} g(1) & g(2) & \dots & g(L) \\ g(2) & g(1) & \dots & g(3-L) \\ \vdots & \vdots & \ddots & \vdots \\ g(K) & g(K+(-1)^{K-1}) & \dots & g(K+(-1)^{K-1}(L-1)) \end{bmatrix}, \tag{6.26}$$

which, for $i \in [K]$ and $j \in [L]$, results in

$$t_{i,g(i+(-1)^{i-1}(j-1))} = \sum\nolimits_{m=1}^{j} T^{(1)}_{i,g(i+(-1)^{i-1}(m-1))} + T^{(2)}_{i,g(i+(-1)^{i-1}(j-1))}. \tag{6.27}$$

For $i \in [K]$, we can re-write (6.27) as follows:

$$t_{g(i+(-1)^{i+j-1}(j-1)),i} = \begin{cases} \sum_{m=1}^{j} T^{(1)}_{g(i+(-1)^{i+j-1}(j-1)),g(i+(-1)^{i+j-1}(j-m))} + T^{(2)}_{g(i+(-1)^{i+j-1}(j-1)),i}, & \text{if } j \in [L], \\ \infty, & \text{if } j \notin [L], \end{cases} \tag{6.28}$$

which results in

$$t_i = \min_{j\in[L]} \left\{ \sum_{m=1}^{j} T^{(1)}_{g(i+(-1)^{i+j-1}(j-1)),g(i+(-1)^{i+j-1}(j-m))} + T^{(2)}_{g(i+(-1)^{i+j-1}(j-1)),i} \right\}. \tag{6.29}$$

We highlight that the CS and SS schemes may not be the optimal schedules for certain realizations of the straggling behaviour, but our interest is in the average performance. We will see in Section 6.7 that both perform reasonably well, and neither scheme outperforms the other at all settings.

6.5.3 Average Completion Time Analysis

Here we analyze the performance of CS and SS providing upper bounds on $\bar{t}^*(L, \tilde{K})$. We represent the average completion time of CS and SS by $\bar{t}_{\text{CS}}(L, \tilde{K})$ and $\bar{t}_{\text{SS}}(L, \tilde{K})$, respectively. In order to characterize these average values through (6.6), we need to obtain $H_{\mathcal{S},\emptyset} = \Pr\{t_i > t, \forall i \in \mathcal{S}\}$, for any set $\mathcal{S} \subset [K]$, $K - \tilde{K} + 1 \le |\mathcal{S}| \le K$, where $t_1, \dots, t_K$ are given in (6.24) and (6.29), for CS and SS, respectively. For ease of presentation, we denote $H_{\mathcal{S},\emptyset}$ for CS and SS by $H^{\text{CS}}_{\mathcal{S},\emptyset}$ and $H^{\text{SS}}_{\mathcal{S},\emptyset}$, respectively. For $\mathcal{S} \subset [K]$ with $K - \tilde{K} + 1 \le |\mathcal{S}| \le K$, we have

$$H^{\text{CS}}_{\mathcal{S},\emptyset} = \Pr\left\{ \sum_{m=1}^{j} T^{(1)}_{g(i-j+1),g(i-j+m)} + T^{(2)}_{g(i-j+1),i} > t, \forall j \in [L], \forall i \in \mathcal{S} \right\} = \Pr\left\{ \mathcal{T}^{\text{CS}}_{\mathcal{S}}(t) \right\}, \tag{6.30}$$

where we define

$$\mathcal{T}^{\text{CS}}_{\mathcal{S}}(t) \triangleq \left\{ \left(T^{(1)}_{1,1}, \dots, T^{(1)}_{K,K}, T^{(2)}_{1,1}, \dots, T^{(2)}_{K,K} \right) : \right.$$

$$\sum\nolimits_{m=1}^{j} T^{(1)}_{g(i-j+1),g(i-j+m)} + T^{(2)}_{g(i-j+1),i} > t, \forall j \in [L], \forall i \in \mathcal{S} \Big\}. \quad (6.31)$$

Similarly, for any set $\mathcal{S} \subset [K]$, $K-k+1 \le |\mathcal{S}| \le K$, we have

$$H^{\rm SS}_{\mathcal{S},\emptyset} = \Pr\Big\{ \sum\nolimits_{m=1}^{j} T^{(1)}_{g(i+(-1)^{i+j-1}(j-1)),g(i+(-1)^{i+j-1}(j-m))} + T^{(2)}_{g(i+(-1)^{i+j-1}(j-1)),i} > t, \forall j \in [L], \forall i \in \mathcal{S} \Big\} = \Pr\left\{\mathcal{T}^{\rm SS}_{\mathcal{S}}(t)\right\} \quad (6.32)$$

where we define

$$\mathcal{T}^{\rm SS}_{\mathcal{S}}(t) \triangleq \Big\{ \left(T^{(1)}_{1,1}, \ldots, T^{(1)}_{K,K}, T^{(2)}_{1,1}, \ldots, T^{(2)}_{K,K}\right) : \sum_{m=1}^{j} T^{(1)}_{g(i+(-1)^{i+j-1}(j-1)),g(i+(-1)^{i+j-1}(j-m))} + T^{(2)}_{g(i+(-1)^{i+j-1}(j-1)),i} > t, \forall j \in [L], \forall i \in \mathcal{S} \Big\}. \quad (6.33)$$

It follows that, for $\mathrm{X} \in \{\mathrm{CS}, \mathrm{SS}\}$,

$$H^{\rm X}_{\mathcal{S},\emptyset} = \underset{\mathcal{T}^{\rm X}_{\mathcal{S}}(t)}{\int \cdots \int} f^{(1)}_{1,[L]}\left(\tau^{(1)}_1\right) \cdots f^{(1)}_{K,[L]}\left(\tau^{(1)}_K\right) f^{(2)}_{1,[L]}\left(\tau^{(2)}_1\right) \cdots f^{(2)}_{K,[L]}\left(\tau^{(2)}_K\right) d\tau^{(1)}_1 \cdots d\tau^{(1)}_K d\tau^{(2)}_1 \cdots d\tau^{(2)}_K. \quad (6.34)$$

By plugging (6.34) into (6.6), we can obtain, for $\mathrm{X} \in \{\mathrm{CS}, \mathrm{SS}\}$,

$$\bar{t}_{\rm X}(L, \tilde{K}) = \sum\nolimits_{i=K-\tilde{K}+1}^{K} (-1)^{K-\tilde{K}+i+1} \binom{i-1}{K-\tilde{K}} \sum\nolimits_{\mathcal{S}\subset[K]:|\mathcal{S}|=i} \int_0^\infty H^{\rm X}_{\mathcal{S},\emptyset} dt. \quad (6.35)$$

Note that we have obtained a general characterization of the average completion time of CS and SS in terms of the CDFs of the delays associated with different tasks at different workers. The numerical evaluation of the performances of CS and SS and the lower bound will be presented in Section 6.7.

6.6 Lower Bound

Here we present a lower bound on $\bar{t}^*(L, \tilde{K})$ by considering an adaptive model. Note that the TO matrix, in general, may depend on the statistics of the computation and communication delays, i.e., $F^{(l)}_{i,[K]}$, $\forall l \in [2]$, but not on the realization of $T^{(l)}_{i,\mathbf{C}(i,j)}$, $i \in [K]$, $j \in [L]$. Let $\hat{T}^{(1)}_{i,j}$ and $\hat{T}^{(2)}_{i,j}$, respectively, represent the computation and communication delays associated with the task worker i executes with its j-th computation, $i \in [K]$, $j \in [L]$. We note that $\hat{T}^{(l)}_{i,j}$ is a random variable independent of the TO matrix, $i \in [K]$, $j \in [L]$, $l \in [2]$. We define

$$\mathbf{T} \triangleq \left(\hat{T}^{(1)}_{1,1}, \hat{T}^{(2)}_{1,1}, \dots, \hat{T}^{(1)}_{1,L}, \hat{T}^{(2)}_{1,L}, \dots, \hat{T}^{(1)}_{K,1}, \hat{T}^{(2)}_{K,1}, \dots, \hat{T}^{(1)}_{K,L}, \hat{T}^{(2)}_{K,L}\right). \tag{6.36}$$

For each realization of $\mathbf{T}$, we allow the master to employ a distinct TO matrix $\mathbf{C}_{\mathbf{T}}$, and denote the completion time by $t_{\mathbf{C}_{\mathbf{T}}}(\mathbf{T}, L, \tilde{K})$, which is a random variable due to the randomness of $\mathbf{T}$. We define

$$t_{\mathrm{LB}}(\mathbf{T}, L, \tilde{K}) \triangleq \min_{\mathbf{C}_{\mathbf{T}}} \left\{ t_{\mathbf{C}_{\mathbf{T}}}(\mathbf{T}, L, \tilde{K}) \right\}, \tag{6.37}$$

and

$$\bar{t}_{\mathrm{LB}}(L, \tilde{K}) \triangleq \mathbb{E}\left[t_{\mathrm{LB}}(\mathbf{T}, L, \tilde{K}) \right], \tag{6.38}$$

where the expectation is taken over $\mathbf{T}$. It is easy to verify that

$$\bar{t}^*(L, \tilde{K}) = \min_{\mathbf{C}} \left\{ \mathbb{E}\left[t_{\mathbf{C}}(L, \tilde{K}) \right] \right\} \geq \mathbb{E}\left[\min_{\mathbf{C}_{\mathbf{T}}} \left\{ t_{\mathbf{C}_{\mathbf{T}}}(\mathbf{T}, L, \tilde{K}) \right\} \right] = \bar{t}_{\mathrm{LB}}(L, \tilde{K}). \tag{6.39}$$

We denote by $\hat{t}_{i,j}$ the time at which the master receives the task computed by worker i with its j-th computation, $i \in [K]$, $j \in [L]$. It follows that

$$\hat{t}_{i,j} = \sum\nolimits_{l=1}^{j} \hat{T}^{(1)}_{i,l} + \hat{T}^{(2)}_{i,j}. \tag{6.40}$$

For a realization of $\mathbf{T}$, $t_{\mathrm{LB}}(\mathbf{T}, L, \tilde{K})$ is the $\tilde{K}$-th order statistics of $\{\hat{t}_{1,1}, \dots, \hat{t}_{1,L}, \dots, \hat{t}_{K,1}, \dots, \hat{t}_{K,L}\}$, i.e., the $\tilde{K}$-th smallest value among $\{\hat{t}_{1,1}, \dots, \hat{t}_{1,L}, \dots, \hat{t}_{K,1}, \dots, \hat{t}_{n,L}\}$, denoted by $\hat{t}_{\mathbf{T},(\tilde{K})}$. To prove that $t_{\mathrm{LB}}(\mathbf{T}, L, \tilde{K}) = \hat{t}_{\mathbf{T},(\tilde{K})}$, we note that $t_{\mathrm{LB}}(\mathbf{T}, L, \tilde{K})$ cannot be smaller than $\hat{t}_{\mathbf{T},(\tilde{K})}$, since, according to the definition, for any time before $\hat{t}_{\mathbf{T},(\tilde{K})}$ master has not received $\tilde{K}$ computations. Also, since master receives the $\tilde{K}$-th computation exactly at time $\hat{t}_{\mathbf{T},(\tilde{K})}$, knowing the realization of $\mathbf{T}$, one can design the TO matrix $\mathbf{C_T}$ such that the first $\tilde{K}$ computations received by the master are all distinct. Since finding the statistics of $\hat{t}_{\mathbf{T},(\tilde{K})}$ is analytically elusive, we obtain the lower bound on $\bar{t}^*(L, \tilde{K})$ through Monte Carlo simulations.

6.7 Performance Comparisons

In this section, we evaluate the average completion time of the proposed CS and SS schemes, and compare them with different results in the literature. We will focus on distributed linear regression as the reference scenario.

We would like to compare the performance of the proposed uncoded computation schemes with coded computation techniques that have received significant interest in recent years. We will consider, in particular, the *polynomially coded* (PC) scheme [107] and the *polynomially coded multi-message* (PCMM) scheme [111]. We also consider an uncoded computing scheme, reffered to as the *random assignment* (RA) scheme [112], in which $L = K$, i.e., the whole dataset is available at each worker, and each worker executes the computations sequentially through a random scheduling. PC and PCMM focus exclusively on linear computation tasks; and hence, we also consider a linear regression problem, in which the goal is to minimize

$$F(\boldsymbol{\theta}) = \frac{1}{N} \left\| \boldsymbol{\Lambda}\boldsymbol{\theta} - \boldsymbol{v} \right\|_2^2, \tag{6.41}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the model parameter vector, $\boldsymbol{\Lambda} \in \mathbb{R}^{N \times d}$ is the data matrix, and $\boldsymbol{v} \in \mathbb{R}^N$ is the vector of labels. We split $\boldsymbol{\Lambda}$ into K disjoint sub-matrices $\boldsymbol{\Lambda} = [\Lambda_1 \cdots \Lambda_K]^T$, where $\Lambda_i \in \mathbb{R}^{d \times N/K}$, and $\boldsymbol{v} = \left[v_1^T \cdots v_K^T\right]^T$, where $v_i \in \mathbb{R}^{N/K}$, $i \in [K]$. The gradient of loss function $F(\boldsymbol{\theta})$ is given by

$$\nabla F(\boldsymbol{\theta}) = \frac{2}{N} \boldsymbol{\Lambda}^T (\boldsymbol{\Lambda}\boldsymbol{\theta} - \boldsymbol{v}) = \frac{2}{N} \sum\nolimits_{i=1}^{K} \left(\Lambda_i \Lambda_i^T \boldsymbol{\theta} - \Lambda_i v_i\right). \tag{6.42}$$

We perform gradient descent to minimize (6.41), in which the model parameters at the l-th iteration, $\boldsymbol{\theta}_l$, are updated as

$$\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - \eta_l \cdot \nabla F(\boldsymbol{\theta}_l) = \boldsymbol{\theta}_l - \eta_l \cdot \frac{2}{N} \sum\nolimits_{i=1}^{n} \left(\Lambda_i \Lambda_i^T \boldsymbol{\theta}_l - \Lambda_i v_i\right), \tag{6.43}$$

where η_l is the learning rate at iteration l. We consider a DGD algorithm, in which the computation of $\nabla F(\boldsymbol{\theta})$ is distributed across K workers, and the master updates the parameter vector according to (6.43) after receiving enough computations from the workers, and sends the updated parameter vector to the workers. For the l-th iteration of DGD, we set

$$h(\Lambda_i) = \Lambda_i \Lambda_i^T \boldsymbol{\theta}_l, \quad \text{for } i \in [K], \tag{6.44}$$

A detailed description of the computation tasks carried out by the workers and the master along with computational complexities for different schemes is provided in [160, Section VI-B].

For the numerical experiments we generate each entry of data matrix $\boldsymbol{\Lambda}$ independently according to distribution $\mathcal{N}(0,1)$. We also generate the labels as $v_i = (\Lambda_i + \Upsilon)^T \Omega$, where $\Upsilon \in \mathbb{R}^{d \times N/K}$, with each entry distributed independently according to $\mathcal{N}(0, 0.01)$, and $\Omega \in \mathbb{R}^d$ with each entry distributed independently according to $\mathcal{U}(0,1)$. For fairness we use the same dataset for all the schemes.

We train a linear regression model using the DGD algorithm described above with a constant learning rate $\eta_l = 0.01$. We run experiments on an Amazon EC2 cluster over t2.micro instance with $K+1$ servers, where one of the servers is designated as the master and the rest serve as workers. We implement different schemes in Python and employ MPI4py library for message passing between different nodes.

At each iteration of the DGD algorithm, we measure the computation and communication delays of each task at each worker. We can then obtain the completion time of each scheme according to its completion criteria. We obtain the average completion time over 500 iterations.

We compare the average completion time of different schemes with respect to the computation load L, $L \geq 2$, in Fig. 6.1, where $K = 15$, $d = 400$, and $N = 900$. As it can be seen, CS and SS outperform PC and PCMM significantly; while PCMM improves upon PC. This result shows that standard coded computation framework cannot fully exploit the computing capabilities in the network, and splitting the computational tasks assigned to each worker and receiving partial computations performed by each worker

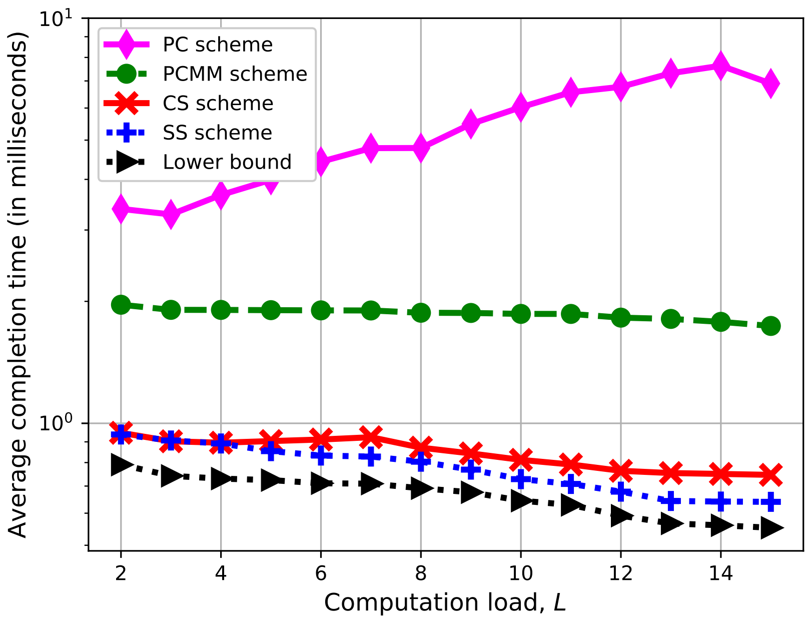


FIGURE 6.1: Average completion time of different schemes with respect to computation load, $L \geq 2$, with $\tilde{K} = K$.

can reduce the average completion time significantly. We also observe that the average completion time of PC increases with L. This is because the delays at different workers are not significantly different; and thus, increasing the computation load to reduce the number of received computations from different workers can increase the total delay. This is another limitation of the coded computation framework, as it requires careful tuning of the parameters based on the statistics of the delays in the system. We observe that the gap between the average completion time of SS and the lower bound is relatively small for the entire range, and reduces with L, and SS outperforms CS with the improvement slightly increasing with L. The average completion time of RA, which requires $L = K$, is 0.895 millisecond, while SS achieves 0.64 millisecond, i.e., around %28.5 reduction. Thus, designing the TO matrix, rather than random computations, can provide significant improvement in computation speed.

In Fig. 6.2, we compare the performances of different schemes with respect to the number of workers, K. We consider $d = 500$, $N = 1000$, and $L = K$. When N/K is not an integer, we zero-pad the dataset. We observe that, except PCMM, the average completion time of different schemes reduce slightly with K when N is fixed. For PC, when $L = K$, the computation received from the fastest worker determines the completion time, and, with all other parameters fixed, the computation delay at each

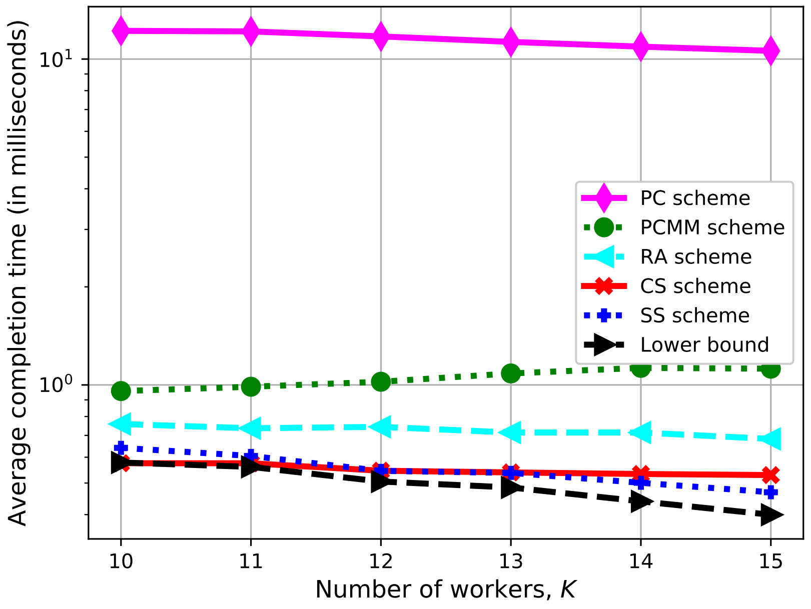


FIGURE 6.2: Average completion time of different schemes with respect to the number of workers, $10 \leq K \leq 15$, with $L = \tilde{K} = K$.

worker depends mostly on N. Thus, by introducing new workers when N is fixed, the average completion time is expected to decrease. Whereas, with PCMM, although the computation time of each task is expected to decrease with K, the average completion time increases, which is due to the increase in the number of communications required by a factor of two. For uncoded computing schemes, RA, CS and SS, the average completion time decreases with K, as they allow a better utilization of the computing resources. As before, we observe that CS and SS improve the average completion time significantly compared to PC and PCMM. Also, based on the superiority of the CS and SS over RA, we conclude that the TO matrix design is essential in reducing the average delay of uncoded computing schemes. CS outperforms SS for small K values, but SS takes over as K increases. The relatively small gap between the average completion times of CS and SS and the lower bound illustrates their efficiency in scheduling the tasks despite the lack of any information on the speeds of the workers.

6.8 Conclusions

In this chapter we have studied distributed computation across inhomogeneous workers with the goal of minimizing the average delay of computing an arbitrary function.

The computation may correspond to each iteration of a DGD algorithm applied on a large dataset. We assume that each worker has access to a limited portion of the dataset, defined as the computation load. In contrast to the growing literature on coded computation to mitigate straggling servers, we have studied uncoded computations and sequential communication to the master in order to benefit from all the computations carried out by the workers. Since the instantaneous computation speeds of the workers are not known in advance, allocation of the tasks to the workers and their scheduling become crucial in minimizing the average completion time. In particular, we have considered the assignment of sub-data to the workers with a predesigned computation order. Workers send the result of each computation to the master as soon as it is executed, and move on to compute the next task assigned to them. Assuming a general statistics for the computation and communication delays of different workers, we have obtained closed-form expressions for the average completion time of two particular computation allocation schemes, called CS and SS. We have compared the performance of these proposed schemes with the existing ones in the literature, particularly the coded PC [107], PCMM [111], and uncoded RA [112] schemes. The results of the experiments carried out on Amazon EC2 cluster show that the CS and SS schemes provide significant reduction in the average completion time over these schemes.

Chapter 7

Machine Learning Over-the-Air

7.1 Overview

In this chapter we study collaborative ML at the wireless edge, where power and bandwidth-limited wireless devices with local datasets carry out DSGD with the help of a remote PS. Standard approaches assume separate computation and communication, where local gradient estimates are compressed and communicated to the PS over orthogonal links. Following this *digital* approach, we introduce D-DSGD, in which the wireless terminals (*workers*) employ gradient quantization and error accumulation, and transmit their gradient estimates to the PS over the underlying wireless MAC. We then introduce an *analog* scheme, called A-DSGD, which exploits the additive nature of the wireless MAC for over-the-air gradient computation. In A-DSGD, the workers first sparsify their gradient estimates, and then project them to a lower dimensional space imposed by the available channel bandwidth. These projections are transmitted directly over the MAC without employing any digital code. Numerical results show that A-DSGD converges much faster than D-DSGD thanks to its more efficient use of the limited bandwidth and the natural alignment of the gradient estimates over the channel. The improvement is particularly compelling at low power and low bandwidth regimes. We also observe that the performance of A-DSGD improves with the number of workers (keeping the total size of the dataset constant), while D-DSGD deteriorates, limiting the ability of the latter in harnessing the computation power of edge devices. The lack of quantization and channel encoding/decoding in A-DSGD further speeds up communication, making it very attractive for low-latency ML applications at the wireless network edge.

7.2 Introduction

ML problems often require the minimization of the empirical loss function

$$F\left(\boldsymbol{\theta}\right) = \frac{1}{N}\sum\nolimits_{n=1}^{N} f\left(\boldsymbol{\theta}, \boldsymbol{u}_n\right), \tag{7.1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ denotes the model parameters to be optimized, $\boldsymbol{u}_n$ is the n-th training data sample, for $n \in [N]$, and $f(\cdot)$ is the loss function defined by the learning model. The minimization of (7.1) is typically carried out through iterative gradient descent (GD), in which the model parameters at the t-th iteration, $\boldsymbol{\theta}_t$, are updated according to

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla F\left(\boldsymbol{\theta}_t\right) = \boldsymbol{\theta}_t - \eta_t \frac{1}{N}\sum\nolimits_{n=1}^{N} \nabla f\left(\boldsymbol{\theta}_t, \boldsymbol{u}_n\right), \tag{7.2}$$

where η_t is the learning rate at iteration t. However, in the case of massive datasets each iteration of GD becomes prohibitively demanding. Instead, in SGD the parameter vector is updated with a stochastic gradient

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \cdot \boldsymbol{g}\left(\boldsymbol{\theta}_t\right), \tag{7.3}$$

which satisfies $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{\theta}_t\right)\right] = \nabla F\left(\boldsymbol{\theta}_t\right)$. SGD also allows parallelization when the dataset is distributed across tens or even hundreds of workers. Due to the growth in the dimensions of the datasets and complexity of the neural networks, exploiting a single machine to carry out a learning task is prohibitively slow, and it is essential to develop distributed ML algorithms. In distributed SGD (DSGD), workers process data samples in parallel while maintaining a globally consistent parameter vector $\boldsymbol{\theta}_t$. In each iteration, worker i computes a gradient vector based on the global parameter vector with respect to its local dataset, denoted by $\mathcal{B}_i$, and sends the result to the PS, which stores and updates the global parameter vector, $i \in [K]$. Once the PS receives the computed gradients from all the workers, it updates the global parameter vector according to

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \frac{1}{K}\sum\nolimits_{i=1}^{K} \boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right), \tag{7.4}$$

where $\boldsymbol{g}_i(\boldsymbol{\theta}_t) \triangleq \frac{1}{|\mathcal{B}_i|}\sum_{\boldsymbol{u}_n \in \mathcal{B}_i} \nabla f(\boldsymbol{\theta}_t, \boldsymbol{u}_n)$ is the stochastic gradient of the current model computed at worker i, $i \in [K]$, using the locally available portion of the dataset, $\mathcal{B}_i$. Ideally, the data parallelism with DSGD should speed up the process K times providing linear scalability. However, in practice, it suffers from extensive communications from the workers to the PS, which is to maintain a consistent global model, and communication is reported as the major bottleneck of DSGD [115,119,120,126,160]. There is no doubt that the communication will be an even bigger hurdle in wireless edge learning due to stringent bandwidth and energy constraints on the workers.

Numerous studies have been dedicated to the reduction of the communication load of DSGD [114,115,117,118,118–136]; however, these works ignore the communication channel, and simply focus on reducing the amount of data that needs to be transmitted from each worker to the PS. Here we consider DSGD over-the-air; that is, we consider a wireless shared medium from the workers to the PS, and treat each iteration of the DSGD algorithm as a distributed over-the-air computation problem. This can model *machine learning at the wireless network edge*, where the workers correspond to IoT devices or sensor nodes that collect their local data samples. Distributed learning in this scenario is attractive also due to privacy and personalization. We will provide two distinct approaches for this wireless DSGD problem, based on *digital* and *analog* computation approaches, respectively. We will show that analog "over-the-air" computation can significantly speed up wireless DSGD, particularly in bandwidth-limited and low-power settings, typically experienced by wireless edge devices.

The remainder of this chapter is organized as follows. In Section 7.3 we introduce the system model of ML over-the-air. We elaborate our DGSD algorithms with digital and analog transmission approaches in Sections 7.4 and 7.5, respectively. In Section 7.6 we provide experimental results. Conclusions are drawn in Section 7.7.

7.3 System Model

We consider distributed ML at the wireless network edge, where K wireless edge nodes, called the workers, employ SGD with the help of a remote PS, to which they are connected through a noisy wireless MAC (see Fig. 7.1). Let $\mathcal{B}_i$ denote the set of

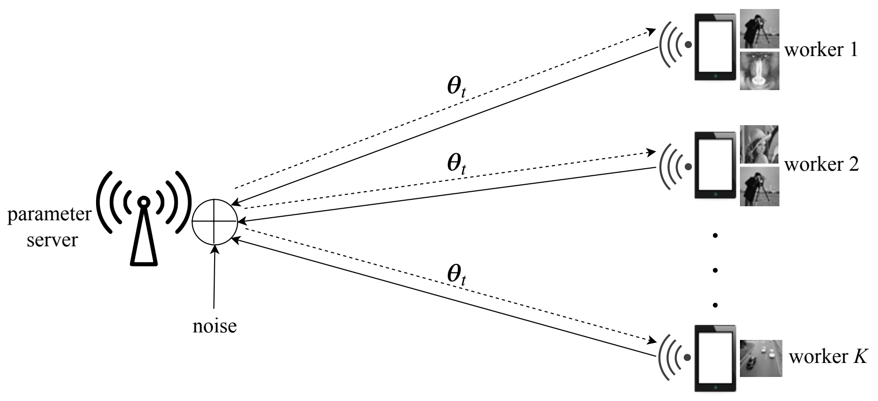


FIGURE 7.1: Illustration of the studied distributed ML framework at the wireless edge. Workers with local datasets collaborate through a PS to carry out DSGD at the network edge, where the local gradient estimates of the workers are transmitted to the PS over a shared wireless MAC.

data samples available at worker i, $i \in [K]$, and $\boldsymbol{g}_i(\boldsymbol{\theta}_t) \in \mathbb{R}^d$ be the stochastic gradient computed by worker i using local data samples, where we remind that $\boldsymbol{\theta}_t \in \mathbb{R}^d$ is the model parameters at the t-th iteration. At each iteration of the DSGD algorithm in (7.4), the local gradient estimates of the workers are sent to the PS over s uses of a Gaussian MAC, characterized by[1]:

$$\boldsymbol{y}_t = \sum\nolimits_{i=1}^{K} \boldsymbol{x}_i + \boldsymbol{z}_t, \tag{7.5}$$

where $\boldsymbol{x}_i \in \mathbb{R}^s$ is the length-s channel input vector transmitted by worker i at iteration t, $\boldsymbol{y}_t \in \mathbb{R}^s$ is the channel output received by the PS, and $\boldsymbol{z}_t \in \mathbb{R}^s$ is the independent additive white Gaussian noise (AWGN) vector with each entry independent and identically distributed (i.i.d.) according to $\mathcal{N}\left(0, \sigma^2\right)$. Since we focus on DSGD, the channel input vector of worker i at iteration t is a function of the current parameter vector $\boldsymbol{\theta}_t$ and the local dataset $\mathcal{B}_i$, and more specifically the current gradient estimate at worker i, $\boldsymbol{g}_i(\boldsymbol{\theta}_t)$, $i \in [K]$. A total average transmit power constraint is imposed:

$$\frac{1}{KT} \sum\nolimits_{t=1}^{T} \sum\nolimits_{i=1}^{K} ||\boldsymbol{x}_{i,t}||_2^2 \leq \bar{P}, \tag{7.6}$$

averaged over iterations of the DSGD algorithm and the workers. The goal is to recover the average of the locally computed gradients $\frac{1}{K}\sum_{i=1}^{K} \boldsymbol{g}_i(\boldsymbol{\theta}_t)$ at the PS, and update

[1]Here we study a Gaussian MAC to present the main ideas behind the proposed ML approach in a simple setting. This approach has been extended to a wireless fading MAC in [161].

the model parameter as in (7.4). However, due to the pre-processing performed at each worker and the noise added by the wireless channel, it is not possible to recover the average gradient perfectly at the PS, and instead, it uses a noisy estimate to update the model parameter vector; i.e., we have $\boldsymbol{\theta}_{t+1} = \phi(\boldsymbol{\theta}_t, \boldsymbol{y}_t)$ for some update function $\phi : \mathbb{R}^d \times \mathbb{R}^s \to \mathbb{R}^d$. The updated model parameter is then multicast to the workers by the PS through an error-free shared link. We assume that the PS has enough resources to deliver a consistent global parameter vector to the workers reliably for their computations in the next iteration.

The transmission of the local gradient computations, $\boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right)$, $i \in [K]$, to the PS with the goal of PS reconstructing their average can be considered as a distributed function computation problem over a MAC [162]. We will consider both a digital approach treating computation and communication separately, and an analog approach that does not use any coding, and instead applies gradient sparsification followed by a linear transformation to compress the gradients, which are then transmitted simultaneously over the channel in an uncoded fashion.

7.4 Digital DSGD (D-DSGD)

In this section, we present DSGD at the wireless network edge utilizing digital compression and transmission over the wireless MAC, referred to as the digital DSGD (D-DSGD) algorithm. Since we do not know the variances of the gradient estimates at different workers, we allocate the power equally among the workers, so that worker i sends $\boldsymbol{x}_{i,t}$ with power P_t, i.e., $||\boldsymbol{x}_{i,t}||_2^2 = P_t$, where P_t values are chosen to satisfy the average transmit power constraint over T iterations

$$\frac{1}{T} \sum\nolimits_{t=1}^{T} P_t \le \bar{P}. \tag{7.7}$$

Due to the intrinsic symmetry of the model, we assume that the workers transmit at the same rate at each iteration (while the rate may change across iterations depending on the allocated power, P_t). Accordingly, the total number of bits that can be transmitted from each of the workers over s uses of the Gaussian MAC, described in (7.5), is upper bounded by

$$R_{\mathrm{D},t} = \frac{s}{2K} \log_2 \left(1 + \frac{KP_t}{s\sigma^2} \right), \tag{7.8}$$

where KP_t/s is the sum-power per channel use. Note that this is an upper bound since it is the Shannon capacity of the underlying Gaussian MAC, and we further assumed that the capacity can be achieved over a finite blocklength of s.

Remark 7.4.1. We note that having a distinct sum power KP_t at each iteration t enables each user to transmit different numbers of bits at different iterations. This corresponds to a novel gradient compression scheme for DSGD, in which the workers can adjust over time the amount of information they send to the PS about their gradient estimates. They can send more information bits at the beginning of the DSGD algorithm when the gradient estimates have higher variances, and reduce the number of transmitted bits over time as the variance decreases. We observed empirically that this improves the performance compared to the standard approach in the literature, where the same compression scheme is applied at each iteration [129].

We will adopt the scheme proposed in [129] for gradient compression at each iteration of the DSGD scheme, as it provides the state-of-the-art in convergence speed with the minimum number of bits transmitted by each worker at each iteration. However, we modify this scheme by allowing different numbers of bits to be transmitted by the workers at each iteration.

At each iteration the workers sparsify their gradient estimates as described below. In order to retain the accuracy of their local gradient estimates, workers employ *error accumulation* [115, 116], where the accumulated error vector at worker i until iteration t is denoted by $\boldsymbol{\Delta}_{i,t-1} \in \mathbb{R}^d$, where we set $\boldsymbol{\Delta}_{i,0} = \mathbf{0}$, $\forall i \in [K]$. Hence, after the computation of the local gradient estimate for parameter vector $\boldsymbol{\theta}_t$, i.e., $\boldsymbol{g}_i(\boldsymbol{\theta}_t)$, worker i updates its estimate with the accumulated error as $\boldsymbol{g}_i(\boldsymbol{\theta}_t) + \boldsymbol{\Delta}_{i,t-1}$, $i \in [K]$. At iteration t, worker i, $i \in [K]$, sets all but the highest κ_t and the smallest κ_t of the entries of its gradient estimate vector $\boldsymbol{g}_i(\boldsymbol{\theta}_t) + \boldsymbol{\Delta}_{i,t-1}$, of dimension d, to zero, where $\kappa_t \leq d/2$ (to have a communication-efficient scheme, in practice, the goal is to have $\kappa_t \ll d$, $\forall t$). Then, it computes the mean values of all the remaining positive entries and all the remaining negative entries, denoted by $\mu_{i,t}^{+}$ and $\mu_{i,t}^{-}$, respectively. If $\mu_{i,t}^{+} > \left|\mu_{i,t}^{-}\right|$, then it sets all the entries with negative values to zero and all the entries with positive values to $\mu_{i,t}^{+}$, and vice versa. We denote the resulting sparse vector at worker i by $\hat{\boldsymbol{g}}_i\left(\boldsymbol{\theta}_t\right)$, and worker i updates the local accumulated error vector as $\boldsymbol{\Delta}_{i,t} = \boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right) + \boldsymbol{\Delta}_{i,t-1} - \hat{\boldsymbol{g}}_i\left(\boldsymbol{\theta}_t\right)$, $i \in [K]$. It then aims to send $\hat{\boldsymbol{g}}_i\left(\boldsymbol{\theta}_t\right)$ over the channel by transmitting its mean value and the positions of its non-zero entries. For this purpose, we use a 32-bit representation of the absolute value of the mean (either $\mu_{i,t}^{+}$ or $\left|\mu_{i,t}^{-}\right|$) along with 1 bit indicating its sign. To send the positions of the non-zero entries, it is assumed in [129] that the distribution of the distances between the non-zero entries is geometrical with success probability q_t, which allows them to use Golomb encoding to send these distances with a total number of bits

$$b^* + \frac{1}{1-(1-\kappa_t)^{2^{b^*}}}, \tag{7.9}$$

where $b^* = 1 + \left\lfloor \log_2\left(\frac{\log\left((\sqrt{5}-1)/2\right)}{\log(1-\kappa_t)}\right)\right\rfloor$. However, we argue that, sending $\log_2\binom{d}{\kappa_t}$ bits to transmit the positions of the non-zero entries is sufficient regardless of the distribution of the positions. This can be achieved by simply enumerating all possible sparsity patterns. Thus, with the D-DSGD scheme, the total number of bits sent by each worker at iteration t is given by

$$r_{\mathrm{D},t} = \log_2\binom{d}{\kappa_t} + 33, \tag{7.10}$$

where κ_t is chosen as the highest integer satisfying $r_{\mathrm{D},t} \leq R_{\mathrm{D},t}$.

7.5 Analog DSGD (A-DSGD)

Next, we propose an analog DSGD algorithm, called A-DSGD, which does not employ any digital coding scheme, either for compression or channel coding, and instead all the workers transmit their gradient estimates simultaneously in an uncoded manner. This is motivated by the fact that, the PS is not interested in the individual gradient vectors, but only in their average. The underlying wireless MAC naturally provides the sum of the gradients, which is the only information required at the PS to update the parameter vector. See Algorithm 8 for a description of the A-DSGD scheme.

Similarly to the D-DSGD scheme, workers employ local error accumulation. Hence, after the computation of the local gradient estimate for parameter vector $\boldsymbol{\theta}_t$, each worker updates its estimate with the accumulated error as $\boldsymbol{g}_i^{ec}\left(\boldsymbol{\theta}_t\right) \triangleq \boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right) + \boldsymbol{\Delta}_{i,t-1}$, $i \in [K]$.

The challenge in the analog transmission approach is to compress the gradient vectors to the available channel bandwidth. In many modern ML applications, such as deep neural networks, the parameter vector, and hence the gradient vectors, have extremely large dimensions, whereas the channel bandwidth, measured by parameter s, is small due to the bandwidth limitations, and to limit the latency of each DSGD iteration. Thus, transmitting all the model parameters one-by-one in an uncoded/analog fashion is not possible as we typically have $d \gg s$.

Lossy compression at any required level is at least theoretically possible in the digital domain. For the analog scheme, in order to reduce the dimension of the gradient vector to that of the channel, the workers apply gradient sparsification. In particular, worker i sets all but the k elements of the error-compensated resulting vector $\boldsymbol{g}_i^{ec}\left(\boldsymbol{\theta}_t\right)$ with the highest magnitudes to zero. We denote the sparse vector at worker i by $\boldsymbol{g}_i^{sp}\left(\boldsymbol{\theta}_t\right)$, $i \in [K]$. This k-level sparsification is represented by function sparse_k in Algorithm 8, i.e., $\boldsymbol{g}_i^{sp}\left(\boldsymbol{\theta}_t\right) = \text{sparse}_k\left(\boldsymbol{g}_i^{ec}\left(\boldsymbol{\theta}_t\right)\right)$. The accumulated error at worker i, $i \in [K]$, is then updated according to

$$\boldsymbol{\Delta}_{i,t} = \boldsymbol{g}_i^{ec}\left(\boldsymbol{\theta}_t\right) - \boldsymbol{g}_i^{sp}\left(\boldsymbol{\theta}_t\right) = \boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right) + \boldsymbol{\Delta}_{i,t-1} - \text{sparse}_k\left(\boldsymbol{g}_i\left(\boldsymbol{\theta}_t\right) + \boldsymbol{\Delta}_{i,t-1}\right). \tag{7.11}$$

We would like to transmit only the non-zero entries of these sparse vectors. However, simply ignoring the zero elements would require transmitting their indeces to the PS separately. To avoid this additional data transmission, we will employ a random projection matrix, similar to compressive sensing. A similar idea is recently used in [163] for analog image transmission over a bandwidth-limited channel.

Algorithm 8 A-DSGD

```
1: Initialize $\boldsymbol{\theta}_1 = 0$ and $\boldsymbol{\Delta}_{1,0} = \cdots = \boldsymbol{\Delta}_{K,0} = 0$
2: for $t = 1, \dots, T$ do
     • Workers do:
3:     for $i = 1, \dots, K$ in parallel do
4:         Compute $\boldsymbol{g}_i(\boldsymbol{\theta}_t)$ with respect to $\boldsymbol{u}_i \in \mathcal{B}_i$
5:         $\boldsymbol{g}_i^{ec}(\boldsymbol{\theta}_t) = \boldsymbol{g}_i(\boldsymbol{\theta}_t) + \boldsymbol{\Delta}_{i,t-1}$
6:         $\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) = \text{sparse}_k(\boldsymbol{g}_i^{ec}(\boldsymbol{\theta}_t))$
7:         $\boldsymbol{\Delta}_{i,t} = \boldsymbol{g}_i^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$
8:         EPA:
9:             $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) = \mathbf{A}_s \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$
10:            $\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) = \sqrt{\alpha_t}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)$
11:        UPA:
12:            $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) = \mathbf{A}_{s-1} \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$
13:            $\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) = \left[\sqrt{\alpha_{i,t}}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)^T \quad \sqrt{\alpha_{i,t}}\right]^T$
14:    end for
     • PS does:
15:    EPA:
16:        $\hat{\boldsymbol{g}}_{\text{EPA}}(\boldsymbol{\theta}_t) = \text{AMP}_{\mathbf{A}_s}\left(\frac{1}{K\sqrt{\alpha_t}}\boldsymbol{y}(\boldsymbol{\theta}_t)\right)$
17:        $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \cdot \hat{\boldsymbol{g}}_{\text{EPA}}(\boldsymbol{\theta}_t)$
18:    UPA:
19:        $\hat{\boldsymbol{g}}_{\text{UPA}}(\boldsymbol{\theta}_t) = \text{AMP}_{\mathbf{A}_{s-1}}\left(\frac{1}{y_s(\boldsymbol{\theta}_t)}\boldsymbol{y}^{s-1}(\boldsymbol{\theta}_t)\right)$
20:        $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \cdot \hat{\boldsymbol{g}}_{\text{UPA}}(\boldsymbol{\theta}_t)$
21: end for
```

Assuming identically distributed datasets across the workers, the local gradient estimates will also follow identical distributions, and hence, will have similar sparsity patterns. A pseudo-random matrix $\mathbf{A}_{\tilde{s}} \in \mathbb{R}^{\tilde{s}\times d}$, for some $\tilde{s} \leq s$, with each entry i.i.d. according to $\mathcal{N}(0, 1/\tilde{s})$, is generated and shared between the PS and the workers before starting the computations. At each iteration t, worker i computes $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) \triangleq \mathbf{A}_{\tilde{s}} \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) \in \mathbb{R}^{\tilde{s}}$, and transmits $\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) \triangleq \left[\sqrt{\alpha_{i,t}}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)^T \quad \boldsymbol{a}_{i,t}{}^T\right]^T$, where $\boldsymbol{a}_{i,t} \in \mathbb{R}^{s-\tilde{s}}$, over the MAC, $i \in [K]$, while satisfying the average power constraint (7.6). The PS receives

$$\boldsymbol{y}(\boldsymbol{\theta}_t) = \sum\nolimits_{i=1}^{K} \boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) + \boldsymbol{z}_t = \begin{bmatrix} \mathbf{A}_{\tilde{s}} \sum_{i=1}^{K} \sqrt{\alpha_{i,t}} \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) \\ \sum_{i=1}^{K} \boldsymbol{a}_{i,t} \end{bmatrix} + \boldsymbol{z}_t. \tag{7.12}$$

In the following, we propose two schemes for this analog transmission approach employing different scaling coefficients, or equivalently, different power allocation schemes.

7.5.1 Equal power allocation (EPA)

In the EPA scheme, we set $\tilde{s} = s$, and at iteration t, worker i computes $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) = \mathbf{A}_s \boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$, $i \in [K]$, and scales its computed low-dimensional gradient vector $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)$ with the same factor $\sqrt{\alpha_t}$, which is known by the workers and the PS, and sends $\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) = \sqrt{\alpha_t}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)$, i.e., $\boldsymbol{a}_{i,t} = \emptyset$. The scaling factor $\sqrt{\alpha_t}$ is chosen to satisfy the following average power constraint over T iterations of A-DSGD algorithm

$$\frac{1}{KT}\sum\nolimits_{t=1}^{T}\sum\nolimits_{i=1}^{K}\alpha_t \left\|\tilde{\boldsymbol{g}}_i(\theta_t)\right\|_2^2 \leq \bar{P}. \tag{7.13}$$

Thus, the received vector at the PS is given by

$$\boldsymbol{y}(\boldsymbol{\theta}_t) = \sqrt{\alpha_t}\sum\nolimits_{i=1}^{K}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) + \boldsymbol{z}_t. \tag{7.14}$$

Since α_t is known also at the PS, it can normalize the received vector to obtain:

$$\frac{1}{K\sqrt{\alpha_t}}\boldsymbol{y}(\boldsymbol{\theta}_t) = \frac{1}{K}\sum_{i=1}^{K}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) + \frac{1}{K\sqrt{\alpha_t}}\boldsymbol{z}_t = \mathbf{A}_s\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) + \frac{1}{K\sqrt{\alpha_t}}\boldsymbol{z}_t. \tag{7.15}$$

The goal of the PS is to recover $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$ from its noisy observation above. For this, we employ the approximate message passing (AMP) algorithm [164]. The AMP algorithm is represented by the $\text{AMP}_{\mathbf{A}_s}$ function in Algorithm 8. The estimate $\hat{\boldsymbol{g}}_{\text{EPA}}(\boldsymbol{\theta}_t)$ is then used to update the model parameters as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \cdot \hat{\boldsymbol{g}}_{\text{EPA}}(\boldsymbol{\theta}_t). \tag{7.16}$$

7.5.2 Unequal power allocation (UPA)

With the UPA scheme, we set $\tilde{s} = s - 1$, which requires $s \geq 2$. At iteration t, we set $\boldsymbol{a}_{i,t} = \sqrt{\alpha_{i,t}}$, and worker i computes $\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t) = \mathbf{A}_{s-1}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$, and sends vector $\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t) = \left[\sqrt{\alpha_{i,t}}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)^T \quad \sqrt{\alpha_{i,t}}\right]^T$ with the same power $P_t = ||\boldsymbol{x}_{i,t}(\boldsymbol{\theta}_t)||_2^2$ satisfying the average power constraint $\frac{1}{T}\sum_{t=1}^{T}P_t \leq \bar{P}$, for $i \in [K]$. Accordingly, scaling factor

$\sqrt{\alpha_{i,t}}$ is determined to satisfy

$$P_t = \alpha_{i,t} \left(\|\tilde{\boldsymbol{g}}_i (\theta_t)\|_2^2 + 1 \right), \tag{7.17}$$

which yields

$$\alpha_{i,t} = \frac{P_t}{\|\tilde{\boldsymbol{g}}_i (\theta_t)\|_2^2 + 1}, \quad \text{for } i \in [K]. \tag{7.18}$$

Since $\|\tilde{\boldsymbol{g}}_i (\theta_t)\|_2^2$ may vary across workers, so can the scaling factor $\sqrt{\alpha_{i,t}}$. That is why, at each iteration t, worker i allocates one channel use to provide the value of $\sqrt{\alpha_{i,t}}$ to the PS along with its scaled low-dimensional gradient vector $\tilde{\boldsymbol{g}}_i (\boldsymbol{\theta}_t)$, $i \in [K]$. Accordingly, the received vector at the PS is given by

$$\boldsymbol{y} (\boldsymbol{\theta}_t) = \begin{bmatrix} \mathbf{A}_{s-1} \sum_{i=1}^{K} \sqrt{\alpha_{i,t}} \boldsymbol{g}_i^{sp} (\boldsymbol{\theta}_t) \\ \sum_{i=1}^{K} \alpha_{i,t} \end{bmatrix} + \boldsymbol{z}_t, \tag{7.19}$$

where $\alpha_{i,t}$, $i \in [K]$, is replaced by (7.18). For $j \in [s]$, we define

$$\boldsymbol{y}^j (\boldsymbol{\theta}_t) \triangleq \begin{bmatrix} y_1 (\boldsymbol{\theta}_t) & y_2 (\boldsymbol{\theta}_t) & \cdots & y_j (\boldsymbol{\theta}_t) \end{bmatrix}^T \tag{7.20}$$

$$\boldsymbol{z}_t^j \triangleq \begin{bmatrix} z_{t,1} & z_{t,2} & \cdots & z_{t,j} \end{bmatrix}^T, \tag{7.21}$$

where $y_j (\boldsymbol{\theta}_t)$ and $z_{t,j}$ denote the j-th element of $\boldsymbol{y} (\boldsymbol{\theta}_t)$ and $\boldsymbol{z}_t$, respectively. Thus, we have

$$\boldsymbol{y}^{s-1} (\boldsymbol{\theta}_t) = \mathbf{A}_{s-1} \sum_{i=1}^{K} \sqrt{\alpha_{i,t}} \boldsymbol{g}_i^{sp} (\boldsymbol{\theta}_t) + \boldsymbol{z}_t^{s-1}, \tag{7.22a}$$

$$y_s (\boldsymbol{\theta}_t) = \sum_{i=1}^{K} \alpha_{i,t} + z_{t,s}. \tag{7.22b}$$

Note that the goal is to recover $\frac{1}{K} \sum_{i=1}^{K} \boldsymbol{g}_i^{sp} (\boldsymbol{\theta}_t)$ at the PS, while, from $\boldsymbol{y}^{s-1} (\boldsymbol{\theta}_t)$ given in (7.22a), the PS observes a noisy version of the weighted sum $\sum_{i=1}^{K} \sqrt{\alpha_{i,t}} \boldsymbol{g}_i^{sp} (\boldsymbol{\theta}_t)$ projected to a low-dimensional vector through $\mathbf{A}_{s-1}$. According to (7.18), each value of $\|\tilde{\boldsymbol{g}}_i (\boldsymbol{\theta}_t)\|_2^2$ results in a distinct scaling factor $\alpha_{i,t}$. However, due to the independence of data samples, for large enough d and $|\mathcal{B}_i|$, the values of $\|\tilde{\boldsymbol{g}}_i (\boldsymbol{\theta}_t)\|_2^2, \forall i \in [K]$, are not going to be too different across workers. As a result, scaling factors $\sqrt{\alpha_{i,t}}, \forall i \in [K]$, are not going to be very different either. Accordingly, to diminish the effect of scaled gradient vectors, we choose to scale down the received vector $\boldsymbol{y}^{s-1}(\boldsymbol{\theta}_t)$ at the PS, given in (7.22a), with the sum of the scaling factors, i.e., $\sum_{i=1}^{K} \sqrt{\alpha_{i,t}}$, whose noisy version is received by the PS as $y_s(\boldsymbol{\theta}_t)$ given in (7.22b). The resulting scaled vector at the PS is given by

$$\begin{aligned}\frac{1}{y_s(\boldsymbol{\theta}_t)}\boldsymbol{y}^{s-1}(\boldsymbol{\theta}_t) &= \frac{1}{y_s(\boldsymbol{\theta}_t)}\left(\mathbf{A}_{s-1}\sum\nolimits_{i=1}^{K}\sqrt{\alpha_{i,t}}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) + \boldsymbol{z}_t^{s-1}\right)\\ &= \mathbf{A}_{s-1}\sum_{i=1}^{K}\frac{\sqrt{\alpha_{i,t}}}{\sum_{i=1}^{K}\sqrt{\alpha_{i,t}} + z_{t,s}}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t) + \frac{1}{\sum_{i=1}^{K}\sqrt{\alpha_{i,t}} + z_{t,s}}\boldsymbol{z}_t^{s-1},\end{aligned} \tag{7.23}$$

where $\alpha_{i,t}$ is given in (7.18). By our choice, the PS tries to recover $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$ from $\boldsymbol{y}^{s-1}(\boldsymbol{\theta}_t)/y_s(\boldsymbol{\theta}_t)$ knowing the measurement matrix $\mathbf{A}_{s-1}$. The PS estimates $\hat{\boldsymbol{g}}_{\text{UPA}}(\boldsymbol{\theta}_t)$ using the AMP algorithm. The estimate $\hat{\boldsymbol{g}}_{\text{UPA}}(\boldsymbol{\theta}_t)$ is then used to update the model parameter as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \cdot \hat{\boldsymbol{g}}_{\text{UPA}}(\boldsymbol{\theta}_t). \tag{7.24}$$

Remark 7.5.1. We remark here that, with SGD the empirical variance of the stochastic gradient vectors reduce over time approaching zero asymptotically. The power should be allocated over iterations taking into account this decaying behaviour of gradient variance, while making sure that the noise term would not become dominant over time. To reduce the variation in the scaling factors $\alpha_{i,t}, \forall i \in [K]$, which is particularly efficient for the UPA power allocation scheme, variance reduction techniques can be used [165]. We also note that setting $P_t = \bar{P}$, $\forall t$, results in a special case of the UPA scheme, where the power is allocated uniformly over time to be resistant against the noise term.

Remark 7.5.2. We remark that in the considered model the main limitation is the channel bandwidth, s. In the proposed A-DSGD algorithm, first a k-level sparsification is applied at worker i, resulting in vector $\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$, $i \in [K]$. Thus, k can take different values satisfying $k < s$ leading to a tradeoff. For a relatively small value of k, $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$ can be more reliably recovered from $\frac{1}{K}\sum_{i=1}^{K}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)$; however it may not provide an accurate estimate of the actual average gradient $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i(\boldsymbol{\theta}_t)$. Whereas,

Table 7.1: Final test accuracy for various DSGD schemes considered in Fig. 7.2

D-DSGD $P_t = \bar{P}$ $\bar{P} = \bar{P}_1$	D-DSGD distinct P_t $\bar{P} = \bar{P}_1$	D-DSGD $P_t = \bar{P}$ $\bar{P} = \bar{P}_2$	D-DSGD distinct P_t $\bar{P} = \bar{P}_2$	A-DSGD UPA $\bar{P} = \bar{P}_1$	A-DSGD EPA $\bar{P} = \bar{P}_1$
0.459	0.501	0.698	0.705	0.811	0.812

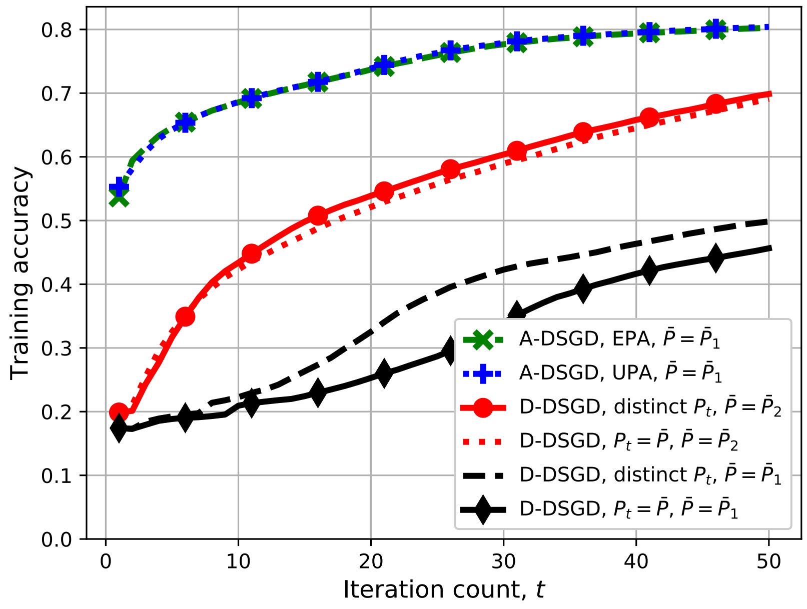


Figure 7.2: Performance of the A-DSGD and D-DSGD algorithms for different $\bar{P}$ values.

with a higher k value, $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$ provides a better estimate of $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i(\boldsymbol{\theta}_t)$, but reliable recovery of $\frac{1}{K}\sum_{i=1}^{K}\boldsymbol{g}_i^{sp}(\boldsymbol{\theta}_t)$ from the vector $\frac{1}{K}\sum_{i=1}^{K}\tilde{\boldsymbol{g}}_i(\boldsymbol{\theta}_t)$ is less likely.

Remark 7.5.3. The proposed A-DSGD algorithm mainly focuses on the analog transmission from the workers, where the dimension of the gradient vectors is reduced utilizing the compressive sensing technique, leading to a more efficient communication scheme with smaller bandwidth requirement. While our focus in this paper has been on the transmission of the gradients, we can apply the existing schemes in the literature that trade-off an increase in the computation load at each worker with a reduction in the communication load. Such schemes include introducing communication delay, where each worker performs SGD algorithm updating the model parameter locally multiple times, and communicates only after multiple local iterations [133]. Moreover, applying momentum correction [118] improves the convergence speed of the DSGD algorithms with communication delay.

7.6 Numerical Experiments

Here we evaluate the performances of the proposed A-DSGD and D-DSGD algorithms for the task of image classification. We run experiments on MNIST dataset [166] with $N = 60000$ training and 10000 test data samples, and train a single layer neural network with $d = 7850$ parameters utilizing ADAM optimizer [167]. The training dataset is split into K disjoint batches with equal size, and each batch is randomly assigned to a distinct worker. We set the channel noise variance to $\sigma^2 = 1$. The performance is measured as the accuracy with respect to the training dataset versus iteration count t, and the final accuracy with respect to the test samples, i.e., test accuracy, after 50 training iterations.

In Fig. 7.2, we compare the performance of the A-DSGD algorithm with both EPA and UPA with D-DSGD algorithm for different values of the available average transmit power $\bar{P}_1 = 127$ and $\bar{P}_2 = 422$. Since we need $r_{\mathrm{D},t} \leq R_{\mathrm{D},t}$ for the digital approach, we set number of channel uses s and $\bar{P}$ to relatively high values, and number of workers K to a relatively small value to make sure that $\kappa_t \geq 1$, $\forall t$, i.e., each worker can transmit at least one information bit at each iteration. We consider $K = 25$ workers, and $s = 0.5d$ channel uses. We set a fixed ratio $k = \lfloor s/2 \rfloor$ for sparsification. The final test accuracy of different DSGD algorithms based on the parameter vector obtained after 50 training iterations is given in Table 7.1. We observe that the analog approach significantly outperforms the standard digital approach of separating computation from communication. We did not include the performance of the A-DSGD algorithm for $\bar{P} = \bar{P}_2$ since it is very close to the one with $\bar{P} = \bar{P}_1$ for both power allocation schemes. Bearing this in mind, we observe that, unlike the A-DSGD scheme, the performance of D-DSGD significantly deteriorates by reducing $\bar{P}$ for both power allocation schemes under consideration. Therefore, analog computation approach is particularly attractive for learning across low-power devices as it allows them to align their limited transmission powers to dominate the noise term. For the UPA, we set $P_t = \bar{P}$, $\forall t$, which satisfies the average power constraint, and for the EPA, we set $\alpha_t = 100 + 10t/3$ and $\alpha_t = 300 + 10t$ resulting in $\bar{P} = \bar{P}_1$ and $\bar{P} = \bar{P}_2$, respectively. For each average power constraint $\bar{P}$, we consider two different power allocation schemes

TABLE 7.2: Final test accuracy for various DSGD schemes considered in Fig. 7.3

D-DSGD $M = 40$ $s = 0.3d$	D-DSGD $M = 20$ $s = 0.3d$	D-DSGD $M = 20$ $s = 0.5d$	A-DSGD $M = 20$ $s = 0.3d$	A-DSGD $M = 40$ $s = 0.3d$	A-DSGD $M = 20$ $s = 0.5d$
0.704	0.729	0.76	0.811	0.816	0.828

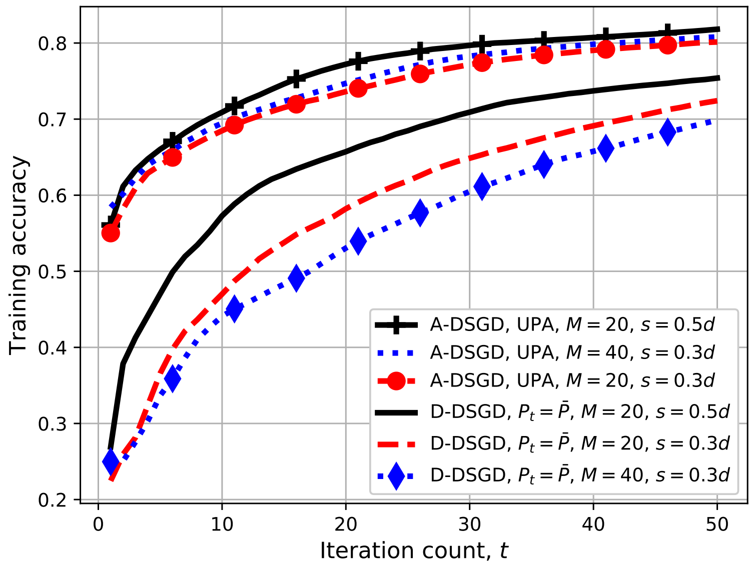


FIGURE 7.3: Performance of the A-DSGD and D-DSGD algorithms for different (M, s) pairs.

for transmission with the D-DSGD algorithm: in the first scheme, we set $P_t = \bar{P}$, $\forall t$, and in the second, we let P_t to be the same as the sum-power consumed by the workers at iteration t of the A-DSGD algorithm with EPA leading to a distinct P_t value at each iteration t. Observe that, for the D-DSGD algorithm, letting P_t vary over time improves the performance, particularly for the smaller $\bar{P}$ value; however, for the A-DSGD, UPA and EPA have a close performance and the improvement of EPA over UPA is negligible for the considered setting parameters.

In Fig. 7.3, we compare the performance of the A-DSGD algorithm with UPA and the D-DSGD algorithm, where, for both analog and digital communications, we set $P_t = \bar{P} = 1100$, $\forall t$, for different K and s values. We consider two different wireless networks $K \in \{20, 40\}$, and for each, we consider two different values of number of channel uses $s \in \{0.3d, 0.5d\}$, and a fixed ratio $k = \lfloor s/2 \rfloor$. We present the final test accuracy of different DSGD algorithms based on the parameter vector obtained after 50 training iterations in Table 7.2. As it can be seen, for $s = 0.3d$, increasing K by a factor of 2 deteriorates the performance of D-DSGD. Accordingly, the performance of D-DSGD algorithm is vulnerable to a relatively small increase in K, as well as a decrease in the average transmit power $\bar{P}$, whose effect was observed in Fig. 7.2. We can conclude that the digital scheme prefers to have a smaller number of workers, which are then allocated more channel resources to be able to transmit their gradient estimates to the PS more accurately. However, this means that D-DSGD cannot harvest the computation power of many edge devices; and its performance compared to A-DSGD will become even poorer when the computation time and energy is also taken into account. On the other hand, we observe that the performance of A-DSGD improves slightly by increasing K from $K = 20$ to $K = 40$ when $s = 0.3d$, and is significantly superior compared to D-DSGD, and the improvement increases remarkably with K. We further observe that reducing the available channel uses s from $s = 0.5d$ to $s = 0.3d$ degrades the performance of the D-DSGD algorithm considerably, whereas the sensitivity of A-DSGD to channel bandwidth is much weaker.

We highlight that in [161] we have extended the analog and digital approaches proposed here to a wireless fading MAC setting, and we have shown the advantage of the analog approach over the digital one for wireless fading MAC.

7.7 Conclusions

In this chapter we have studied distributed machine learning at the wireless edge, where K workers aim to minimize a loss function by performing DSGD with the help of a remote PS. Workers communicate with the PS over a wireless MAC. We have considered both a digital approach (D-DSGD) that separates computation and communication, and an analog approach (A-DSGD) that exploits the superposition property of the wireless channel to have the average gradient at the PS computed over-the-air. In the D-DSGD scheme, the amount of information bits sent by each worker at each iteration can be adaptively adjusted with respect to the average transmit power constraint $\bar{P}$. In the A-DSGD scheme, we have proposed gradient sparsification followed by compressive sensing employing the same measurement matrix at all the workers in order to reduce the typically very large parameter vector dimension to the limited channel bandwidth. This analog approach allows a much more efficient use of the limited channel bandwidth, and benefits from the beamforming effect thanks to the identical distributions of the gradients across the workers. Numerical results have shown significant improvement in performance with the analog approach, particularly in the low-power and low-bandwidth regimes. We have also observed that, unlike D-DSGD, the performance of A-DSGD improves with the number of workers.

Chapter 8

Conclusions

Growing demand not only for content but also for computation over wireless networks requires moving some of the core processing capabilities close to the network edge. This dissertation can be divided into two parts exploiting edge processing capabilities to make content delivery more efficient, as well as to bring network intelligence close to edge devices. In Chapters 2, 3, 4, and 5 we have studied coded caching techniques for various settings and developed information-theoretic tools to characterize the fundamental limits. The considered caching model and the presented results illustrate that even a limited storage can be converted into spectral efficiency in communication networks, benefiting the whole network, if it is exploited intelligently. Moreover, in Chapters 6 and 7 we have studied fundamental limits of exploiting the computational capabilities of edge devices, which are prevalent with their local datasets, to carry out a learning algorithm collaboratively. We have developed tools to analyze the performance of distributed ML at the edge of wireless networks.

In Chapter 2, we have studied proactive content caching at user terminals, each equipped with a cache of limited size. The system considered here models wireless networks, in which the caches are filled over off-peak periods without any cost constraint or rate limitation (apart from the limited cache capacities), but without knowing the user demands; and all the user demands arrive (almost) simultaneously, and they are served simultaneously through an error-free shared link by the server hosting the whole library. We have first considered the same cache size across the users, and proposed a novel centralized coded caching scheme that places coded contents in the users' caches, referred to as the PCC scheme, and provides improvement for relatively small cache sizes. The delivery phase of PCC exploits both coded and uncoded transmission of various pieces of contents, carefully created to retain the symmetry across users and files. We have then extended the PCC scheme to higher cache sizes, and proposed GBC and GBD schemes for centralized and decentralized caching scenarios, respectively. We have finally considered distinct cache capacities at different users, and proposed a novel coded caching scheme in a decentralized scenario that improves upon the state-of-the-art delivery rate. The improvement is achieved by creating more multicasting opportunities for the delivery of bits that have not been cached by any of the users, or cached by only a single user. In particular, the proposed scheme exploits the idea behind the GBC scheme introduced for centralized caching in a system with symmetric cache capacities.

In contrast to the setting considered in Chapter 2, we have studied a noisy channel for the transmission of contents in the delivery phase in Chapters 3, 4, and 5 in order to model the physical layer communication.

In Chapter 3 we have studied cache-aided content delivery over a packet erasure BC with arbitrary erasure probabilities. The capacity of this network is defined as the maximum common rate of files in the library, which allows reliable delivery to all the receivers, independent of their demands. We have derived a lower bound on the capacity by proposing a novel caching and delivery scheme, which enables all the receivers to benefit from the cache memories available at the network. The proposed scheme utilizes a finer subpacketization of the files in the library, and provides a better exploitation of the available cache memories with a higher achievable rate than the state-of-the-art.

In Chapter 4 we have considered cache-aided content delivery over a Gaussian BC. Considering same rate contents in the library, we have studied both the minimum peak transmission power, which is the minimum transmit power that can satisfy all user demand combinations, and the minimum average transmit power, averaged across all demand combinations, assuming uniform demand distributions. We have proposed a centralized caching and coded delivery scheme assuming that the channel conditions in the delivery phase are not known beforehand. Coded contents are transmitted in the delivery phase to their intended receivers using superposition coding and power allocation. We have then extended the achievable scheme to the decentralized caching scenario. We have also provided a lower bound on the required peak and average transmission power values assuming uncoded cache placement. Our results indicate that even a small cache capacity at the receivers can provide a significant reduction in the required transmission power level highlighting the benefits of caching in improving the energy efficiency of wireless networks.

In Chapter 5 we have studied cache-aided content delivery over a Gaussian BC, however, unlike the models studied in Sections 3 and 4, each user is allowed to demand a file at a distinct rate. We have considered a centralized placement phase, where the server knows the channel qualities of the links in the delivery phase in addition to the identity of the users. By allowing the users to have different cache capacities, we have defined the capacity region for a total cache capacity. We designed a placement phase through cache allocation across the users and the files' layers to maximize the rates allocated to different layers. We have proposed three achievable schemes, which deliver coded multicast packets through different channel coding techniques over the Gaussian BC. Although the coded multicast packets are intended for a set of users with distinct link capacities, channel coding techniques can be employed to deliver requested files such that the users with better channels achieve higher rates. We have also developed an outer bound on the capacity region assuming uncoded caching.

In Chapter 6 we have studied distributed computation of an arbitrary function over a dataset across workers with different random speeds. In contrast to the growing literature on coded computation to mitigate straggling servers, here we have studied uncoded computations and sequential communication to the master in order to benefit from all the computations carried out by the workers, including the slower ones. We have considered the assignment of data points to the workers with a predesigned computation order. Assuming a general statistics for the computation and communication delays of different workers, we have obtained closed-form expressions for the average completion time of two particular computation allocation schemes, called CS and SS. The results of the experiments carried out on Amazon EC2 cluster show that the CS and SS schemes provide significant reduction in the average completion time over the state-of-the-art coded computing schemes and an uncoded computing scheme with random scheduling of computations.

In Chapter 7 we have studied distributed ML at the wireless network edge, where the workers aim to minimize an empirical loss function collaboratively by performing

DSGD with the help of a remote PS. Workers have their own local datasets, and they communicate with the PS over a wireless MAC with limited bandwidth. As opposed to standard approach to distributed ML, which ignores the channel aspects, and simply aims at reducing the communication load by compressing the gradients at each iteration to a prefixed level, here we incorporate the wireless channel characteristics and constraints into the system design. We have considered both a digital approach (D-DSGD) that separates computation and communication, and an analog approach (A-DSGD) that exploits the superposition property of the wireless channel to have the average gradient computed over-the-air. Numerical results have shown significant improvement in performance with the analog approach, particularly in the low-power and low-bandwidth regimes.

Future Research Challenges

In this dissertation we have studied several problems related to distributed coded caching and computing. However, there are many open research questions that need to be addressed for a full understanding of the performance limits of coded caching and computing at the wireless network edge.

Despite the relaxations applied to the idealistic caching model introduced in [3] to make the model more realistic, there are still certain aspects of the coded caching and delivery techniques proposed in this dissertation that must be reconsidered carefully to make the proposed solutions practically relevant. Level of subpacketization is a crucial metric for making the caching schemes practical, in particular for the decentralized caching scenario. With the centralized and decentralized coded caching schemes proposed in [3] and [26], respectively, the subpacketization level grows exponentially with the number of workers, K. For example, for $K = 50$, the scheme in [3] requires a subpacketization level of approximately 10^{14}, which results in an impractical file size, and would introduce significant overhead in a practical implementation. Recently, there have been efforts to reduce the subpacketization level of caching networks while achieving global caching gain, focusing on error-free link for communication from the server to the users [168–174]. A potential research challenge is to develop techniques with relatively low subpacketization level taking into account the channel characteristics.

In the different caching models considered here, the placement phase is assumed to be performed without any cost. Instead, an appropriate cost in terms of delay or entropy can be introduced in the placement phase. The optimal transmission policy to minimize the total energy consumed by the server for a point-to-point fading channel, where the receiver is equipped with a finite capacity cache has been formulated as a water-filling solution in [9]. We argue here that one potential approach to incorporate error into the placement phase is to extend the technique presented in [9] to a multicasting scenario, in which the server has to satisfy the arbitrary demands of a group of users with limited caches through a fading broadcast channel, aiming to minimize the corresponding cost function, e.g., transmission energy.

In the distributed computing framework, we did not include the computation delay at the PS in the evaluations, while additional encoding and decoding complexities can introduce a relatively significant delay at the PS. This delay is more highlighted in the case of coded computing, in which the PS requires to encode the data points and/or decode the received computations. One possible research direction direction is to incorporate computation delay at the PS into the framework.

One potential direction to extend the scheme proposed for the ML over-the-air problem is to incorporate energy into the optimization framework, and investigate a communication-efficient approach with minimum energy consumption.

Finally, for the ML at the wireless edge problem, it is important to study the case where the data across the workers is dependent, which can happen in practice particularly for federated learning setting. In the extreme scenario, we can assume that all the workers have access to the same portion of the dataset, and develop techniques for transmission over a bandwidth limited wireless medium from the workers to the PS.

In conclusion, despite the considerable efforts for development of various coded caching techniques, the optimal delivery rate-cache capacity trade-off even for the conventional caching model in [3] is still an open problem. Also, the distributed ML over-the-air problem is still far from being optimal in terms of the speed of convergence for realistic settings. However, we hope that the results presented in this Ph.D. thesis have contributed towards our understanding of these problems and advanced the state-of-the-art towards the optimal solutions, and encouraged research and development in these important and challenging problems so that the remaining open problems will be solved and some of proposed ideas will be taken up for practical implementations.

Bibliography

[1] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE J. Sel. Areas in Commun.*, vol. 14, no. 6, pp. 1110–1122, Aug. 1996.

[2] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142–149, Apr. 2013.

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

[4] M. Gregori, J. Gomez-Vilardebo, J. Matamoros, and D. Gündüz, "Wireless content caching for small cell and D2D networks," *IEEE J. Sel. Areas in Commun.*, vol. 34, no. 5, pp. 1222–1234, Mar. 2016.

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

[6] I. Baev, R. Rajaraman, and C. Swamy, "Approximation algorithms for data placement problems," *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1411–1429, 2008.

[7] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, San Diego, CA, Mar. 2010, pp. 1–9.

[8] P. Blasco and D. Gunduz, "Multi-armed bandit optimization of cache content in wireless infostation networks," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Honolulu, HI, Jun. 2014, pp. 51–55.

[9] A. C. Güngör and D. Gündüz, “Proactive wireless caching at mobile user devices for energy efficiency,” in *Proc. IEEE Int. Symp. on Wireless Comm. Systems (ISWCS)*, Brussels, Belgium, Aug. 2015.

[10] K. Wan, D. Tuninetti, and P. Piantanida, “On the optimality of uncoded cache placement,” in *Proc. IEEE Inform. Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016, pp. 161–165.

[11] Z. Chen, P. Fan, and K. B. Letaief, “Fundamental limits of caching: Improved bounds for users with small buffers,” *IET Communications*, vol. 10, no. 17, pp. 2315–2318, Nov. 2016.

[12] K. Wan, D. Tuninetti, and P. Piantanida, “On caching with more users than files,” in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 135–139.

[13] S. Sahraei and M. Gastpar, “K users caching two files: An improved achievable rate,” in *Proc. Annual Conference on Information Science and Systems (CISS)*, Princeton, NJ, USA, Mar. 2016, pp. 620–624.

[14] M. Mohammadi Amiri and D. Gündüz, “Fundamental limits of coded caching: Improved delivery rate-cache capacity trade-off,” *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 806–815, Feb. 2017.

[15] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, “Coded caching for a large number of users,” in *Proc. IEEE Inform. Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016, pp. 171–175.

[16] J. Gomez-Vilardebo, “Fundamental limits of caching: Improved bounds with coded prefetching,” *arXiv: 1612.09071 [cs.IT]*, May 2017.

[17] C. Tian and J. Chen, “Caching and delivery via interference elimination,” *IEEE Trans. Inform. Theory*, vol. 64, no. 3, pp. 1548–1560, Mar. 2018.

[18] K. Zhang and C. Tian, “Fundamental limits of coded caching: From uncoded prefetching to coded prefetching,” *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 6, pp. 1153–1164, Jun. 2018.

[19] J. Gomez-Vilardebo, "A novel centralized coded caching scheme with coded prefetching," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 6, pp. 1165–1175, Jun. 2018.

[20] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," *IEEE Trans. Inform. Theory*, vol. 64, no. 2, pp. 1281–1296, Feb. 2018.

[21] A. Sengupta, R. Tandon, and T. C. Clancy, "Improved approximation of storage-rate tradeoff for caching via new outer bounds," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 1691–1695.

[22] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 1696–1700.

[23] C. Tian, "A note on the fundamental limits of coded caching," *arXiv:1503.00010 [cs.IT]*, Feb. 2015.

[24] C.-Y. Wang, S. H. Lim, and M. Gastpar, "A new converse bound for coded caching," in *Proc. Information Theory and Applications Workshop (ITA)*, La Jolla, CA, USA, Jan. 2016, pp. 1–6.

[25] C.-Y. Wang, S. Saeedi Bidokhti, and M. Wigger, "Improved converses and gap results for coded caching," *IEEE Trans. Inform. Theory*, vol. 64, no. 11, pp. 7051–7062, Nov. 2018.

[26] M. A. Maddah-Ali and U. Niesen, "Decentralized caching attains order optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw*, vol. 23, no. 4, pp. 1029–1040, Apr. 2014.

[27] Y. P. Wei and S. Ulukus, "Novel decentralized coded caching through coded prefetching," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017, pp. 1–5.

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

[29] K. Poularakis and L. Tassiulas, "On the complexity of optimal content placement in hierarchical caching networks," *IEEE Trans. Commun.*, vol. 64, no. 5, pp. 2092–2103, May 2016.

[30] J. Hachem, N. Karamchandani, and S. N. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Trans. Inform. Theory*, vol. 36, no. 5, pp. 3108–3141, May 2017.

[31] J. Zhang, X. Lin, C. C. Wang, and X. Wang, "Coded caching for files with distinct file sizes," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 1686–1690.

[32] S. Wang, W. Li, X. Tian, and H. Liu, "Coded caching with heterogeneous cache sizes," *arXiv:1504.01123 [cs.IT]*, Aug. 2015.

[33] A. Ibrahim, A. Zewail, and A. Yener, "Centralized coded caching with heterogeneous cache sizes," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, Mar. 2017, pp. 1–6.

[34] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Decentralized caching and coded delivery with distinct cache capacities," *IEEE Trans. Commun.*, vol. PP, no. 99, pp. 1–1, Aug. 2017.

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

[36] Q. Yang, M. Mohammadi Amiri, and D. Gündüz, "Audience-retention-rate-aware caching and coded video delivery with asynchronous demands," *arXiv:1808.04835 [cs.IT]*, Aug. 2018.

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

[38] S. H. Lim, C.-Y. Wang, and M. Gastpar, "Information theoretic caching: The multi-user case," *IEEE Trans. Inform. Theory*, vol. 63, no. 11, pp. 7018–7037, Nov. 2017.

[39] R. Timo, S. Saeedi Bidokhti, M. Wigger, and B. Gieger, "A rate-distortion approach to caching," *IEEE Trans. Inform. Theory*, vol. 64, no. 3, pp. 1957–1976, Mar. 2018.

[40] Q. Yang and D. Gündüz, "Coded caching and content delivery with heterogeneous distortion requirements," *IEEE Trans. Inform. Theory*, vol. 64, no. 6, pp. 4347–4364, Jun. 2018.

[41] P. Hassanzadeh, E. Erkip, J. Llorca, and A. Tulino, "Distortion-memory tradeoffs in cache-aided wireless video delivery," in *Proc. Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello, IL, USA, Sep. 2015, pp. 1150–1157.

[42] R. Timo, S. B. Bidokhti, M. Wigger, and B. Geiger, "A rate-distortion approach to caching," in *Proc. International Zurich Seminar on Communications*, Zurich, Switzerland, Mar. 2016.

[43] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Trans. Netw*, vol. 24, no. 2, pp. 836–845, Apr. 2016.

[44] S. O. Somuyiwa, A. György, and D. Gündüz, "A reinforcement-learning approach to proactive caching in wireless networks," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 6, pp. 1331–1344, Jun. 2018.

[45] K. Shanmugam, M. Ji, A. M. Tulino, J. Llorca, and A. G. Dimakis, "Finite-length analysis of caching-aided coded multicasting," *IEEE Trans. Inform. Theory*, vol. 62, no. 10, pp. 5524–5537, Oct. 2016.

[46] S. Jin, Y. Cui, H. Liu, and G. Caire, "Order-optimal decentralized coded caching schemes with good performance in finite file size regime," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Washington, DC, USA, Dec. 2016, pp. 1–7.

[47] K. Wan, D. Tuninetti, and P. Piantanida, "Novel delivery schemes for decentralized coded caching in the finite file size regime," in *Proc. IEEE Int.l Conf. on Commun. Workshops (ICC Workshops)*, Paris, France, May 2017, pp. 1183–1188.

[48] E. Lampiris and P. Elia, "Adding transmitters dramatically boosts coded-caching gains for finite file sizes," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 6, pp. 1176–1188, Jun. 2018.

[49] S. Jin, Y. Cui, H. Liu, and G. Caire, "A new order-optimal decentralized coded caching scheme with good performance in the finite file size regime," *to appear, IEEE Trans. Commun.*, Apr. 2019.

[50] P. Hassanzadeh, A. Tulino, J. Llorca, and E. Erkip, "Cache-aided coded multicast for correlated source," in *Proc. IEEE Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Brest, France, Sep. 2016, pp. 360–364.

[51] ——, "Correlation-aware distributed caching and coded delivery," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016, pp. 166–170.

[52] Q. Yang and D. Gündüz, "Centralized coded caching of correlated contents," in *Proc. IEEE Int. Conf. on Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[53] P. Hassanzadeh, A. Tulino, J. Llorca, and E. Erkip, "On coding for cache-aided delivery of dynamic correlated content," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 8, pp. 1666–1681, Jun. 2018.

[54] Q. Yang, P. Hassanzadeh, D. Gündüz, and E. Erkip, "Centralized caching and delivery of correlated contents over a Gaussian broadcast channel," in *Int. Symp. on Modeling and Opt. in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, Shanghai, China, May 2018, pp. 1–6.

[55] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inform. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

[56] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Ciare, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inform. Theory*, vol. 59, no. 12, pp. 8402–8413, Sep. 2013.

[57] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142–149, Apr. 2013.

[58] M. Ji, M. F. Wong, A. M. Tulino, J. Llorca, G. Caire, M. Effros, and M. Langberg, "On the fundamental limits of caching in combination networks," in *Proc. IEEE Int. Workshop on Signal Process. Advances in Wireless Commun. (SPAWC)*, Stockholm, Sweden, Jun.-Jul. 2015, pp. 695–699.

[59] L. Tang and A. Ramamoorthy, "Coded caching for networks with the resolvability property," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 420–424.

[60] N. Naderializadeh, M. A. Maddah-Ali, and S. Avestimeh, "On the optimality of separation between caching and delivery in general cache network," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 1232–1236.

[61] A. A. Zewail and A. Yener, "Coded caching for combination networks with cache-aided relays," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 2433–2437.

[62] K. Wan, D. Tuninetti, P. Piantanida, and M. Ji, "On combination networks with cache-aided relays and users," in *Proc. Int. ITG Workshop on Smart Antennas (WSA)*, Bochum, Germany, Mar. 2018, pp. 1–7.

[63] Q. Yan, M. Wigger, and S. Yang, "Placement delivery array design for combination networks with edge caching," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 1555–1559.

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

[65] M. Wigger, R. Timo, and S. Shamai, "Complete interference mitigation through receiver-caching in Wyner's networks," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016, pp. 335–339.

[66] Y. Ugur, Z. H. Awan, and A. Sezgin, "Cloud radio access networks with coded caching," in *Proc. Int. ITG Workshop on Smart Antennas (WSA)*, Munich, Germany, Mar. 2016, pp. 1–5.

[67] S.-H. Park, O. Simeone, and S. Shamai, "Joint optimization of cloud and edge processing for fog radio access networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7621–7632, Nov. 2016.

[68] R. Tandon and O. Simeone, "Cloud-aided wireless networks with edge caching: Fundamental latency trade-offs in fog radio access networks," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 2029–2033.

[69] B. Azari, O. Simeone, U. Spagnolini, and A. Tulino, "Hypergraph-based analysis of clustered cooperative beamforming with application to edge caching," *IEEE Wireless Commun. Lett.*, vol. 5, no. 1, pp. 84–87, Feb. 2016.

[70] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog computing based radio access networks: Issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, Jul.-Aug. 2016.

[71] H. Xu and C. G. X. Wang, "Efficient file delivery for coded prefetching in shared cache networks with multiple requests per user," *IEEE Trans. Commun.*, vol. 67, no. 4, pp. 2849–2865, Apr. 2019.

[72] S. Wang, X. Tian, and H. Liu, "Exploiting the unexploited of coded caching for wireless content distribution," in *Proc. IEEE Int. Conf. Comput. Commun. Netw. (ICCCN)*, Las Vegas, NV, USA, Feb. 2015, pp. 700–706.

[73] W. Huang, S. Wang, L. Ding, F. Yang, and W. Zhang, "The performance analysis of coded cache in wireless fading channel," *arXiv:1504.01452 [cs.IT]*, Apr. 2015.

[74] A. Ghorbel, K. H. Ngo, R. Combes, M. Kobayashi, and S. Yang, "Opportunistic content delivery in fading broadcast channels," in *Proc. IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–6.

[75] K.-H. Ngo, S. Yang, and M. Kobayashi, "Scalable content delivery with coded caching in multi-antenna fading channels," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 548–562, Jan. 2018.

[76] J. Zhang and P. Elia, "Fundamental limits of cache-aided wireless BC: Interplay of coded-caching and CSIT feedback," *IEEE Trans. Inform. Theory*, vol. 63, no. 5, pp. 3142–3160, May 2017.

[77] ——, "Wireless coded caching: A topological perspective," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 401–405.

[78] ——, "Feedback-aided coded caching for the MISO BC with small caches," in *Proc. Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.

[79] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, "Multi-antenna coded caching," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 2113–2117.

[80] ——, "Physical-layer schemes for wireless coded caching," *IEEE Trans. Inform. Theory*, vol. 65, no. 5, pp. 2792–2807, May 2019.

[81] E. Piovano, H. Joudeh, and B. Clerckx, "On coded caching in the overloaded MISO broadcast channel," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 2795–2799.

[82] ——, "Generalized degrees of freedom of the symmetric cache-aided miso broadcast channel with partial csit," *to appear, IEEE Trans. Inform. Theory*, Apr. 2019.

[83] J. Zhao, M. Mohammadi Amiri, and D. Gunduz, "A low-complexity cache-aided multi-antenna content delivery scheme," in *Proc. IEEE Int. Workshop on Signal Processing Advances in Wireless Commun. (SPAWC)*, Cannes, France, Jul. 2019.

[84] X. Yi and G. Caire, "Topological coded caching," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 2039–2043.

[85] A. Ghorbel, M. Kobayashi, and S. Yang, "Content delivery in erasure broadcast channels with cache and feedback," *IEEE Trans. Inform. Theory*, vol. 62, no. 11, pp. 6407–6422, Nov. 2016.

[86] M. Maddah-Ali and U. Niesen, "Cache-aided interference channels," *IEEE Trans. Inform. Theory*, vol. 65, no. 3, pp. 1714–1724, Mar. 2019.

[87] J. Pujol Roig, D. Gündüz, and F. Tosato, "Interference networks with caches at both ends," in *Proc. IEEE Int. Conf. on Commun. (ICC)*, Paris, France, May 2016, pp. 1–6.

[88] J. Pujol Roig, F. Tosato, and D. Gündüz, "Storage-latency trade-off in cache-aided fog radio access networks," in *Proc. IEEE Int. Conf. on Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[89] N. Naderializadeh, M. Maddah-Ali, and A. S. Avestimehr, "Fundamental limits of cache-aided interference management," *IEEE Trans. Inform. Theory*, vol. 63, no. 5, pp. 3092–3107, May 2017.

[90] R. Timo and M. Wigger, "Joint cache-channel coding over erasure broadcast channels," in *Proc. IEEE Int. Symp. Wireless Commun. Syst. (SWCS)*, Bruxelles, Belgium, Aug. 2015, pp. 201–205.

[91] S. Saeedi Bidokhti, M. Wigger, and R. Timo, "Noisy broadcast networks with receiver caching," *IEEE Trans. Inform. Theory*, vol. 64, no. 11, pp. 6996–7016, Nov. 2018.

[92] S. Saeedi Bidokhti, M. Wigger, and A. Yener, "Benefits of cache assignment on degraded broadcast channels," *arXiv:1702.08044 [cs.IT]*, Feb. 2017.

[93] M. Mohammadi Amiri and D. Gündüz, "Cache-aided content delivery over erasure broadcast channels," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 370–381, Jan. 2018.

[94] ——, "Caching and coded delivery over Gaussian broadcast channels for energy efficiency," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 8, pp. 1706–1720, Aug. 2018.

[95] ——, "On the capacity region of a cache-aided Gaussian broadcast channel with multi-layer messages," in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 1909–1913.

[96] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *CoRR*, vol. abs/1812.02858, 2018. [Online]. Available: http://arxiv.org/abs/1812.02858

[97] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inform. Theory*, vol. 64, no. 3, pp. 1514–1529, Mar. 2018.

[98] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proc. of ICML*, Sydney, Australia, Aug. 2017, pp. 3368–3376.

[99] W. Halbawi, N. A. Ruhi, F. Salehi, and B. Hassibi, "Improving distributed gradient descent using Reed-Solomon codes," *arXiv:1706.05436 [cs.IT]*, Jun. 2017.

[100] S. Dutta, G. Joshi, S. Ghosh, P. Dube, and P. Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed SGD," *arXiv:1803.01113 [cs.IT]*, May 2018.

[101] R. K. Maity, A. S. Rawat, and A. Mazumdar, "Robust gradient descent via moment encoding with LDPC codes," *arXiv:1805.08327 [cs.IT]*, Jan. 2019.

[102] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," *arXiv:1802.03475 [cs.IT]*, Feb. 2018.

[103] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *arXiv:1801.07487 [cs.IT]*, May 2018.

[104] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *arXiv:1801.10292 [cs.IT]*, May 2018.

[105] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," *arXiv:1705.10464 [cs.IT]*, Jan. 2018.

[106] Q. Yu, N. Raviv, J. So, and A. S. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security and privacy," *arXiv:1806.00939 [cs.IT]*, Jun. 2018.

[107] S. Li, S. M. Mousavi Kalan, Q. Yu, M. Soltanolkotabi, and A. S. Avestimehr, “Polynomially coded regression: Optimal straggler mitigation via data encoding,” *arXiv:1805.09934 [cs.IT]*, May 2018.

[108] N. Ferdinand and S. C. Draper, “Hierarchical coded computation,” in *Proc. IEEE Int. Symp. on Inform. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 1620–1624.

[109] S. Kiani, N. Ferdinand, and S. C. Draper, “Exploitation of stragglers in coded computation,” *arXiv:1806.10253 [cs.IT]*, Jun. 2018.

[110] A. Mallick, M. Chaudhari, and G. Joshi, “Rateless codes for near-perfect load balancing in distributed matrix-vector multiplication,” *arXiv:1804.10331 [cs.DC]*, Mar. 2018.

[111] E. Ozfatura, D. Gündüz, and S. Ulukus, “Speeding up distributed gradient descent by utilizing non-persistent stragglers,” *arXiv:1808.02240 [cs.IT]*, Aug. 2018.

[112] S. Li, S. M. Mousavi Kalan, A. S. Avestimehr, and M. Soltanolkotabi, “Near-optimal straggler mitigation for distributed gradient methods,” *arXiv:1710.09990 [cs.IT]*, Oct. 2017.

[113] M. A. Attia and R. Tandon, “Combating computational heterogeneity in large-scale distributed computing via work exchange,” *arXiv:1711.08452 [cs.DC]*, Nov. 2017.

[114] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” *arXiv:1610.05492 [cs.LG]*, Oct. 2017.

[115] F. Seide1, H. Fu, J. Droppo, G. Li, and D. Yu, “1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs,” in *Proc. INTERSPEECH*, Singapore, Sep. 2014, pp. 1058–1062.

[116] N. Strom, “Scalable distributed DNN training using commodity GPU cloud computing,” in *Proc. INTERSPEECH*, 2015, pp. 1488–1492.

[117] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "ATOMO: Communication-efficient learning via atomic sparsification," *arXiv:1806.04090 [stat.ML]*, Jun. 2018.

[118] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *arXiv:1712.01887 [cs.CV]*, Feb. 2018.

[119] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *International Conference on International Conference on Machine Learning (ICML)*, Jul. 2015, pp. 1737–1746.

[120] D. Alistarh, D. Grubic, J. Z. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via randomized quantization and encoding," in *Proc. Advances in Neural Information Processing Systems*, Long Beach, CA, USA, Dec. 2017, pp. 1709–1720.

[121] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "TernGrad: Ternary gradients to reduce communication in distributed deep learning," *arXiv:1705.07878 [cs.LG]*, Dec. 2017.

[122] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv:1606.06160 [cs.NE]*, Feb. 2018.

[123] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," *arXiv:1802.04434 [cs.LG]*, Aug. 2018.

[124] J. Wu, W. Huang, J. Huang, and T. Zhang, "Error compensated quantized SGD and its applications to large-scale distributed optimization," *arXiv:1806.08054 [cs.CV]*, Jun. 2018.

[125] B. Li, W. Wen, J. Mao, S. Li, Y. Chen, and H. Li, "Running sparse and low-precision neural network: When algorithm meets hardware," in *Proc. Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jeju, South Korea, Jan. 2018.

[126] N. Strom, "Scalable distributed DNN training using commodity gpu cloud computing," in *Proc. Conference of the International Speech Communication Association (INTERSPEECH)*, 2015.

[127] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," *arXiv:1704.05021 [cs.CL]*, Jul. 2017.

[128] X. Sun, X. Ren, S. Ma, and H. Wang, "meProp: Sparsified back propagation for accelerated deep learning with reduced overfitting," *arXiv:1706.06197 [cs.LG]*, Oct. 2017.

[129] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Sparse binary compression: Towards distributed deep learning with minimal communication," *arXiv:1805.08768 [cs.LG]*, May 2018.

[130] C. Renggli, D. Alistarh, T. Hoefler, and M. Aghagolzadeh, "SparCML: High-performance sparse communication for machine learning," *arXiv:1802.08021 [cs.DC]*, Oct. 2018.

[131] D. Alistarh, T. Hoefler, M. Johansson, S. Khirirat, N. Konstantinov, and C. Renggli, "The convergence of sparsified gradient methods," *arXiv:1809.10505 [cs.LG]*, Sep. 2018.

[132] Y. Tsuzuku, H. Imachi, and T. Akiba, "Variance-based gradient compression for efficient distributed deep learning," *arXiv:1802.06058 [cs.LG]*, Feb. 2018.

[133] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2017.

[134] S. U. Stich, "Local SGD converges fast and communicates little," *arXiv:1805.09767 [math.OC]*, Jun. 2018.

[135] T. Lin, S. U. Stich, and M. Jaggi, "Don't use large mini-batches, use local SGD," *arXiv:1808.07217 [cs.LG]*, Oct. 2018.

[136] T. Chen, G. B. Giannakis, T. Sun, and W. Yin, “LAG: Lazily aggregated gradient for communication-efficient distributed learning,” *arXiv:1805.09965 [stat.ML]*, May 2018.

[137] G. Zhu, Y. Wang, and K. Huang, “Low-latency broadband analog aggregation for federated edge learning,” *arXiv:1812.11494 [cs.IT]*, Jan. 2019.

[138] K. Yang, T. Jiang, Y. Shi, and Z. Ding, “Federated learning via over-the-air computation,” *arXiv:1812.11750 [cs.LG]*, Jan. 2019.

[139] M. Mohammadi Amiri and D. Gündüz, “Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air,” *arXiv:1901.00844 [cs.DC]*, Jan. 2019.

[140] B. McMahan and D. Ramage, “Federated learning: Collaborative machine learning without centralized training data,” *[online]. Available. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html*, Apr. 2017.

[141] M. Mohammadi Amiri and D. Gündüz, “Fundamental limits of coded caching: Improved delivery rate-cache capacity trade-off,” *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 806–815, Feb. 2017.

[142] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, “Coded caching for a large number of users,” *arXiv:1605.01993 [cs.IT]*, May 2016.

[143] ——, “Decentralized caching and coded delivery with distinct cache capacities,” *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 4657–4669, Nov. 2017.

[144] R. L. Urbanke and A. D. Wyner, “Packetizing for the erasure broadcast channel with an internet application,” in *Proc. Int.. Conf. Combinatorics, Information Theory and Statistics*, Portland, ME, May 1997, p. 93.

[145] E. Tuncel, “Slepian-Wolf coding over broadcast channels,” *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1469–1482, Apr. 2006.

[146] A. El Gamal and Y.-H. Kim, *Network information theory.* Cambridge, UK: Cambridge University Press, 2012.

[147] P. Bergmans, "Coding theorem for broadcast channels with degraded components," *IEEE Trans. Inform. Theory*, vol. 19, no. 2, pp. 197–207, Mar. 1973.

[148] M. Costa, "Writing on dirty paper," *IEEE Trans. Inform. Theory*, vol. 29, no. 3, pp. 439–441, May 1983.

[149] C.-Y. Chen, J. Choi, D. Brand, A. Agrawal, W. Zhang, and K. Gopalakrishnan, "AdaComp : Adaptive residual gradient compression for data-parallel distributed training," *arXiv:1712.02679 [cs.LG]*, Dec. 2017.

[150] Z. Tao and Q. Li, "eSGD: Communication efficient distributed deep learning on the edge," in *Workshop on Hot Topics in Edge Computing (HotEdge)*, Boston, MA, USA, Jul. 2018.

[151] R. L. Graham, "Bounds on multiprocessing anomalies and packing algorithms," in *Proc. SJCC*, 1972, pp. 205–218.

[152] J. D. Ullman, "NP-complete scheduling problems," *J. Comput. Syst. Sci.*, vol. 10, no. 3, pp. 384–393, Jun. 1975.

[153] S. Xue, M. Li, X. Xu, and J. Chen, "An ACO-LB algorithm for task scheduling in the cloud environment," *J. Software*, vol. 9, no. 2, pp. 466–473, Feb. 2014.

[154] S. Zhan and H. Huo, "Improved PSO-based task scheduling algorithm in cloud computing," *J. Inf. Comput. Sci.*, vol. 9, no. 13, pp. 3821–3829, Nov. 2012.

[155] G. Reddy, N. Reddy, and S. Phanikumar, "Multi objective task scheduling using modified ant colony optimization in cloud computing," *Int'l J. Intell. Eng. Sys.*, vol. 11, no. 3, pp. 242–250, 2018.

[156] T. Choudhari, M. Moh, and T.-S. Moh, "Prioritized task scheduling in fog computing," in *Proc. ACMSE*, Richmond, Kentucky, Mar. 2018, pp. 401–405.

[157] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Trans. Indus. Inform.*, vol. 14, no. 10, pp. 4712–4721, Oct. 2018.

[158] R. Pedarsani, J. Walrand, and Y. Zhong, "Scheduling tasks with precedence constraints on multiple servers," in *Proc. Allerton*, Monticello, IL, 2014, pp. 1196–1203.

[159] C.-S. Yang, R. Pedarsani, and A. S. Avestimehr, "Communication-aware scheduling of serial tasks for dispersed computing," *arXiv:1804.06468 [cs.DC]*, Apr. 2018.

[160] M. Mohammadi Amiri and D. Gündüz, "Computation scheduling for distributed machine learning with straggling workers," *arXiv:1810.09992 [cs.DC]*, Oct. 2018.

[161] M. Mohammadi Amiri and D. Gündüz, "Federated learning over wireless fading channels," *arXiv:1907.09769 [cs.IT]*, Jul. 2019.

[162] M. Goldenbaum and S. Stanczak, "Robust analog function computation via wireless multiple-access channels," *IEEE Trans. Commun.*, vol. 61, no. 9, pp. 3863–3877, Sep. 2013.

[163] T-Y. Tung and D. Gündüz, "SparseCast: Hybrid digital-analog wireless image transmission exploiting frequency-domain sparsity," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2451–2454, Dec. 2018.

[164] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 45, pp. 18 914–18 919, Nov. 2009.

[165] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Proc. International Conference on Neural Information Processing Systems (NIPS)*, Navada, USA, Dec. 2013.

[166] Y. LeCun, C. Cortes, and C. Burges, "The MNIST database of handwritten digits," *http://yann.lecun.com/exdb/mnist/*, 1998.

[167] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980 [cs.LG]*, Jan. 2017.

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

[169] L. Tang and A. Ramamoorthy, "Coded caching schemes with reduced subpacketization from linear block code," *IEEE Trans. Inform. Theory*, vol. 64, no. 4, pp. 3099–3120, Apr. 2018.

[170] J. Wang, M. Cheng, Q. Yan, and X. Tang, "Placement delivery array design for coded caching scheme in D2D networks," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3388–3395, May 2019.

[171] N. Woolsey, R.-R. Chen, and M. Ji, "Coded caching in wireless device-to-device networks using a hypercube approach," in *Proc. IEEE Int. Conf. on Commun. Workshops (ICC Workshops)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[172] M. B. H. H. Suthan and P. Krishnan, "Coded caching via projective geometry: A new low subpacketization scheme," *arXiv:1901.07823 [cs.IT]*, Feb. 2019.

[173] P. Krishna, "Coded caching via line graphs of bipartite graphs," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Guangzhou, China, Nov. 2018.

[174] Q. Yan, X. Tang, and Q. Chen, "Placement delivery array and its application," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Guangzhou, China, Nov. 2018.

Appendix A

Proofs for Chapter 2

A.1 Proof of Theorem 2.1

To find the delivery rate of the proposed scheme, the delivery rate for each part of the delivery phase is calculated separately. Having received the bits sent in the first part of the delivery phase presented in Algorithm 1, we would like each user to recover all the subfiles in its cache that have been cached in the XOR-ed form during the placement phase. However, to achieve this, we transmit pieces of the files that are not requested by that user. For example, for user j in group $\mathcal{G}_i$ with demand W_i, $i \in [N]$ and $j \in [S_{i-1}+1 : S_i]$, we deliver $(N-1)$ different pieces corresponding to $(N-1)$ different files (except file W_i) to retrieve all the subfiles $W_{l,j}$, for $l \in [N]$. Since there are K users, a total of $K(N-1)$ different pieces, each of length $\frac{F}{K(N-1)}$ bits, are sent over the shared link in the first part of the delivery phase. As a result, the delivery rate of part 1 of the delivery phase is $D_{\mathrm{PCC}_1} = 1$.

In part 2 of the proposed delivery phase provided in Algorithm 2, for the users in each group $\mathcal{G}_i$, $(K_i - 1)$ XOR-ed contents $\bigcup_{j=S_{i-1}+1}^{S_i-1} (W_{i,j} \oplus W_{i,j+1})$ are transmitted over the shared link, enabling all the users in group $\mathcal{G}_i$ to recover the subfiles $W_{i,S_{i-1}+1}, \ldots, W_{i,S_i}$. Hence, a total of $\sum_{i=1}^{N} (K_i - 1)$ XOR-ed contents, each of size F/K bits, are delivered over the shared link, which results in a delivery rate of

$$D_{\mathrm{PCC}_2} = \frac{1}{K} \sum\nolimits_{i=1}^{N} (K_i - 1) = 1 - \frac{N}{K} \tag{A.1}$$

for the second part of the delivery phase.

Finally, Algorithm 3 corresponds to the last part of the proposed delivery scheme, which enables file exchanges between the users in groups $\mathcal{G}_i$ and $\mathcal{G}_j$, for $i \in [N-1]$

and $j \in [i+1:N]$. There are $(N-2)$ missing pieces of the file requested by users in group $\mathcal{G}_i$ ($\mathcal{G}_j$) that are located in the cache of each of the users in group $\mathcal{G}_j$ ($\mathcal{G}_i$) with indexes $l_1 \in [N-1] \setminus \{m_{i,S_j}\}$ ($l_2 \in [N-1] \setminus \{m_{j,S_i}\}$). Note that, we have $(N-2)$ missing pieces rather than $(N-1)$ as one piece was delivered in part 1 of the delivery scheme. For the piece with index l_1 and the piece with index l_2, the server delivers $\bigcup_{n=S_{j-1}+1}^{S_j-1} \left(W_{i,n}^{(l_1)} \oplus W_{i,n+1}^{(l_1)} \right)$, $\bigcup_{n=S_{i-1}+1}^{S_i-1} \left(W_{j,n}^{(l_2)} \oplus W_{j,n+1}^{(l_2)} \right)$, and $W_{i,S_j}^{(l_1)} \oplus W_{j,S_i}^{(l_2)}$, which enables all the users in group $\mathcal{G}_i$ to recover the pieces $W_{i,S_{j-1}+1}^{(l_1)}, ..., W_{i,S_j}^{(l_1)}$, and also all the users in group $\mathcal{G}_j$ to recover the pieces $W_{j,S_{i-1}+1}^{(l_2)}, ..., W_{j,S_i}^{(l_2)}$, by delivering a total of $(K_i + K_j - 1)$ XOR-ed contents, each of size $\frac{F}{K(N-1)}$ bits. As a result, the delivery rate of the third part is given by

$$D_{\text{PCC}_3} = \frac{(N-2)}{K(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (K_i + K_j - 1) = (N-2)\left(1 - \frac{N}{2K}\right). \tag{A.2}$$

By adding up the delivery rate of the three parts, the following delivery rate is achieved:

$$D_{\text{PCC}}\left(\frac{N-1}{K}\right) = D_{\text{PCC}_1} + D_{\text{PCC}_2} + D_{\text{PCC}_3} = N\left(1 - \frac{N}{2K}\right), \tag{A.3}$$

which completes the proof of Theorem 2.1.

A.2 Proof of Theorem 2.3

We first go through the coded delivery phase presented in Algorithm 4, and show that all user requests are satisfied at the end of the delivery phase. First part of this algorithm enables each user to obtain the subfiles of its requested file which are in the cache of all other users in the same group. We consider the first group, i.e., $i = 1$ in line 2 of Algorithm 4, which includes the users that demand W_1. In this case, the XOR-ed contents $W_{1,j} \oplus W_{1,j+1}$, for $j \in [K_1 - 1]$, are delivered by the server. Having access to the subfile $W_{1,j}$ locally in its cache, each user j, for $j \in [K_1]$, can decode all the remaining subfiles $W_{1,l}$, for $l \in [K_1] \setminus \{j\}$. Thus, a total number of $(K_1 - 1)$ XOR-ed contents, each of size $\frac{F}{K}$ bits, are delivered by the server for the users in group $\mathcal{G}_1$. Similarly, the second group ($i = 2$ in line 2 of Algorithm 4), containing the users requesting file W_2, the XOR-ed contents $W_{2,j} \oplus W_{2,j+1}$, for $j \in [K_1+1 : K_1+K_2-1]$, are sent by the server. With subfile $W_{2,j}$ available locally at user j, for $j \in [K_1+1 : K_1+K_2]$, user j can obtain the missing subfiles $W_{2,l}$, $\forall l \in [K_1+1 : K_1+K_2] \setminus \{j\}$. Hence, a total of $(K_2-1)F/K$ bits are served for the users in $\mathcal{G}_2$, and so on so forth. Accordingly, for the users belonging to group $\mathcal{G}_i$, $(K_i-1)F/K$ bits are delivered by the server, for $i \in [N]$, and the total number of bits transmitted by the server in the first part of the coded delivery phase presented in Algorithm 4 is given by

$$D_{\mathrm{GBC}_1}\left(\frac{N}{K}\right) = \frac{F}{K}\sum\nolimits_{i=1}^{N}(K_i-1) = (K-N)\frac{F}{K}. \tag{A.4}$$

In the second part of Algorithm 4, each user in group $\mathcal{G}_i$, for $i \in [N]$, will decode the missing subfiles of its requested file, which are in the cache of users belonging to groups $j \in [N] \setminus \{i\}$. We first start with $i = 1$ and $j = 2$ in lines 7 and 8, respectively. The XOR-ed contents $W_{1,l} \oplus W_{1,l+1}$, for $l \in [K_1+1 : K_1+K_2-1]$, i.e., the subfiles of W_1 cached by users in group $\mathcal{G}_2$, are delivered in line 9. In line 10, the XOR-ed contents $W_{2,l} \oplus W_{2,l+1}$, for $l \in [K_1-1]$, i.e., the subfiles of W_2 cached by users in group $\mathcal{G}_1$, are delivered by the server. Finally, by delivering $W_{1,K_1+K_2} \oplus W_{2,K_1}$ in line 11, and having already decoded $W_{2,l}$ ($W_{1,l}$), each user l in $\mathcal{G}_1$ ($\mathcal{G}_2$) can recover the missing subfiles of its requested file W_1 (W_2) which are in the cache of users in $\mathcal{G}_2$ ($\mathcal{G}_1$), for $l \in [K_1]$ (for $l \in [K_1+1 : K_1+K_2]$). In this particular case, the number of bits delivered by the server in lines 9, 10, and 11 are $(K_2-1)F/K$, $(K_1-1)F/K$, and F/K, respectively, which adds up to a total number of $(K_1+K_2-1)F/K$ bits. In a similar manner, the subfiles can be exchanged between users in groups $\mathcal{G}_i$ and $\mathcal{G}_j$, for $i \in [N-1]$ and $j \in [i+1 : N]$, by delivering a total of $(K_i+K_j-1)F/K$ bits through sending the XOR-ed contents stated in lines 9, 10, and 11 of Algorithm 4. Hence, the total number of bits delivered by the server in the second part of the coded delivery phase is given by

$$D_{\mathrm{GBC}_2}\left(\frac{N}{K}\right) = \frac{F}{K}\sum\nolimits_{i=1}^{N-1}\sum\nolimits_{j=i+1}^{N}(K_i+K_j-1) = (N-1)\left(K-\frac{N}{2}\right)\frac{F}{K}. \tag{A.5}$$

By summing up (A.4) and (A.5), the delivery rate of the GBC scheme is given by

$$D_{\mathrm{GBC}}\left(\frac{N}{K}\right) = N - \frac{N\left(N+1\right)}{2K}. \tag{A.6}$$

A.3 Proof of Theorem 2.5

Consider first the CODED DELIVERY procedure in Algorithm 6. We note that, when $N < K$, the difference between the first procedure of the proposed delivery phase and the delivery phase presented in [32, Algorithm 1] lies in the first two parts, i.e., delivering the missing bits of the requested files, which either have not been cached by any user, or have been cached by only a single user. Hence, having the delivery rate of the scheme in [32, Algorithm 1], the delivery rate of the CODED DELIVERY procedure in Algorithm 6 can be determined by finding the difference in the delivery rates in these first two parts.

The delivery rate for Part 1 of the proposed CODED DELIVERY procedure, in which the bits of each request W_{d_i}, for $i \in [K]$, that have not been cached by any user are directly sent to the users requesting the file, is given by

$$D_{\mathrm{GBD}_1}\left(M_{[K]}\right) = N \prod\nolimits_{i=1}^{K} \left(1 - \frac{M_i}{N}\right). \tag{A.7}$$

We can see that the worst-case demand combination for this part of the CODED DELIVERY procedure is when each file is requested by at least one user, i.e., $K_i \geq 1$, $\forall i \in [N]$. The corresponding delivery rate of [32, Algorithm 1] is given by:

$$D_{\mathrm{WLTL}_1}\left(M_{[K]}\right) = K \prod\nolimits_{i=1}^{K} \left(1 - \frac{M_i}{N}\right). \tag{A.8}$$

The difference between these two delivery rates is

$$\Delta D_1\left(M_{[K]}\right) \triangleq D_{\mathrm{WLTL}_1}\left(M_{[K]}\right) - D_{\mathrm{GBD}_1}\left(M_{[K]}\right) = (K - N) \prod\nolimits_{i=1}^{K} \left(1 - \frac{M_i}{N}\right). \tag{A.9}$$

In Part 2 of the delivery phase of the GBD scheme, we deal with the bits of each requested file that have been cached by only a single user i, i.e., $W_{d_j,\{i\}}$, for some

$i, j \in [K]$. For any request W_{d_j}, the normalized number of bits that have been cached exclusively by user i will be denoted by Q_i. As $F \to \infty$, by the law of large numbers, Q_i can be approximated as [26]

$$Q_i \approx \left(\frac{M_i}{N}\right) \prod\nolimits_{l \in [K] \setminus \{k\}} \left(1 - \frac{M_l}{N}\right) = \left(\frac{M_i}{N - M_i}\right) \prod\nolimits_{l=1}^{K} \left(1 - \frac{M_l}{N}\right). \tag{A.10}$$

From (A.10) we can see that $Q_i \geq Q_j$, $i \neq j$, $\forall i, j \in [K]$, if and only if $M_i \geq M_j$; that is, the user with a larger cache size stores more bits of each file for F sufficiently large.

Next, we evaluate the delivery rate for Part 2 of the CODED DELIVERY procedure. We start with message $X_{2,1}^{(F)}$. For the users in $\mathcal{G}_i$, for $i \in [N]$, ordered in increasing cache capacities $M_{S_{i-1}+1} \leq M_{S_{i-1}+2} \leq \cdots \leq M_{S_i}$, a total number of $(K_i - 1)$ pieces, with the normalized sizes $Q_{S_{i-1}+2}, \dots, Q_{S_i}$ are delivered. Thus, the delivery rate of the common message $X_{2,1}^{(F)}$ is given by

$$D_{\mathrm{GBD}_2}^{1}\left(M_{[K]}\right) \triangleq \sum\nolimits_{i=1}^{N} \sum\nolimits_{j=S_{i-1}+2}^{S_i} Q_j. \tag{A.11}$$

In line 8 of Algorithm 6, $(K_j - 1)$ pieces, each of length $Q_{S_{j-1}+2}, \dots, Q_{S_j}$, and $(K_i - 1)$ pieces, each of length $Q_{S_{i-1}+2}, \dots, Q_{S_i}$ are delivered for users in $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively, and also the normalized length of the bits delivered with the last content of $X_{2,2}^{(F)}$ is $\max\left\{Q_{S_{j-1}+1}, Q_{S_{i-1}+1}\right\}$, for $i \in [N-1]$ and $j \in [i+1 : N]$. Hence, the rate of the common message $X_{2,2}^{(F)}$ is given by

$$\begin{aligned} D_{\mathrm{GBD}_2}^{2}\left(M_{[K]}\right) \triangleq \sum\nolimits_{i=1}^{N-1} \sum\nolimits_{j=i+1}^{N} \Bigg(& \sum\nolimits_{n=S_{j-1}+2}^{S_j} Q_n + \sum\nolimits_{n=S_{i-1}+2}^{S_i} Q_n \\ & + \max\left\{Q_{S_{j-1}+1}, Q_{S_{i-1}+1}\right\} \Bigg). \end{aligned} \tag{A.12}$$

To simplify the presentation, without loss of generality, let us assume that $M_1 \leq M_{S_1+1} \leq \cdots \leq M_{S_{N-1}+1}$. Then (A.12) can be rewritten as

$$D_{\mathrm{GBD}_2}^{2}\left(M_{[K]}\right) \triangleq \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left(\sum\nolimits_{n=S_{j-1}+2}^{S_j} Q_n + \sum\nolimits_{n=S_{i-1}+2}^{S_i} Q_n + Q_{S_{j-1}+1} \right). \tag{A.13}$$

The total delivery rate for the second part of the proposed coded delivery phase is found by summing up the rates of the two parts, i.e.,

$$D_{\text{GBD}_2}\left(M_{[K]}\right) \triangleq \sum\nolimits_{i=1}^{2} D^{i}_{\text{GBD}_2}\left(M_{[K]}\right). \tag{A.14}$$

By substituting (A.11) and (A.13) into (A.14), we obtain

$$D_{\text{GBD}_2}\left(M_{[K]}\right) = N\sum\nolimits_{i=1}^{N}\sum\nolimits_{j=S_{i-1}+2}^{S_i} Q_j + \sum\nolimits_{i=1}^{N-1} iQ_{S_i+1}. \tag{A.15}$$

Note that, in (A.15), the coefficient of Q_{S_i+1} is i, for $i \in [0:N-1]$, whereas the coefficient of all other Q_js, $\forall j \in [K]\backslash\mathcal{P}$, where $\mathcal{P} \triangleq \{1, S_1+1, ..., S_{N-1}+1\}$, is N. Since $N > K$, the achievable rate for Part 2 of the CODED DELIVERY procedure in Algorithm 6 is maximized (the worst-case user demands happens) if $Q_i \leq Q_j$, for $i \in \mathcal{P}$ and $j \in [K]\backslash\mathcal{P}$; or, equivalently, if $M_i \leq M_j$, for $i \in \mathcal{P}$ and $j \in [K]\backslash\mathcal{P}$. According to the definition of set $\mathcal{P}$, the above condition means that N users with the smallest cache sizes, i.e., users i, $\forall i \in \mathcal{P}$, will request different files, and belong to distinct groups in the worst-case scenario.

For simplification, without loss of generality, the users are ordered such that $M_1 \leq M_2 \leq \cdots \leq M_K$. Then, the delivery rate of Part 2 of the CODED DELIVERY procedure is

$$D_{\text{GBD}_2}\left(M_{[K]}\right) = \sum\nolimits_{i=1}^{N}(i-1)\,Q_i + N\sum\nolimits_{i=N+1}^{K} Q_i. \tag{A.16}$$

By substituting Q_i in (A.10), we have

$$D_{\text{GBD}_2}\left(M_{[K]}\right) = \left[\sum_{i=1}^{N}(i-1)\left(\frac{M_i}{N-M_i}\right) + N\sum_{i=N+1}^{K}\left(\frac{M_i}{N-M_i}\right)\right]\prod_{l=1}^{K}\left(1-\frac{M_l}{N}\right). \tag{A.17}$$

Now, we derive the delivery rate for the corresponding part in [32, Algorithm 1], i.e., when the server delivers the bits of the file requested by user i, having been cached only by user j, $\forall i,j \in [K]$, $i \neq j$. For this case, from [32, Algorithm 1], when $M_1 \leq M_2 \leq \cdots \leq M_K$, we have

$$D_{\text{WLTL}_2}\left(M_{[K]}\right) = \left[\sum\nolimits_{i=1}^{K}(i-1)\left(\frac{M_i}{N-M_i}\right)\right]\prod\nolimits_{j=1}^{K}\left(1-\frac{M_j}{N}\right). \tag{A.18}$$

Hence, the difference between the delivery rates for the second part of the proposed coded delivery phase and its counterpart in [32, Algorithm 1] is given by

$$\begin{aligned}\Delta D_2\left(M_{[1:K]}\right) &\triangleq D_{\mathrm{WLTL}_2}\left(M_{[K]}\right) - D_{\mathrm{GBD}_2}\left(M_{[K]}\right)\\ &= \left[\sum\nolimits_{i=1}^{K-N} (i-1)\left(\frac{M_{i+N}}{N-M_{i+N}}\right)\right]\prod\nolimits_{j=1}^{K}\left(1-\frac{M_j}{N}\right). \qquad \text{(A.19)}\end{aligned}$$

Part 3 of the CODED DELIVERY procedure in Algorithm 6 is the same as its counterpart in [32, Algorithm 1]; so, they achieve the same delivery rate. Based on [32, Theorem 3], assuming that $M_1 \leq M_2 \leq \cdots \leq M_K$, the delivery rate for the CODED DELIVERY procedure is

$$D_{\mathrm{CD}}\left(M_{[K]}\right) \triangleq \sum\nolimits_{i=1}^{K}\left[\prod\nolimits_{j=1}^{i}\left(1-\frac{M_j}{N}\right)\right] - \Delta D_1\left(M_{[K]}\right) - \Delta D_2\left(M_{[K]}\right), \qquad \text{(A.20)}$$

where $\Delta D_1\left(M_{[K]}\right)$ and $\Delta D_2\left(M_{[K]}\right)$ are as given in (A.9) and (A.19), respectively.

Now, consider the RANDOM DELIVERY procedure in Algorithm 6. Each delivered message in this procedure is directly targeted for the users in a group requesting the same file. It is assumed that the users in $\mathcal{G}_i$ are ordered to have increasing cache capacities, such that $M_{S_{i-1}+1} \leq M_{S_{i-1}+2} \leq \cdots \leq M_{S_i}$, for $i \in [N]$. Since each user in $\mathcal{G}_i$ requires at most $\left(1 - M_{S_{i-1}+1}/N\right)F$ bits to get its requested file, a total number of $\left(1 - M_{S_{i-1}+1}/N\right)F$ bits, obtained from random linear combinations of W_i, are sufficient to enable the users in $\mathcal{G}_i$ to decode their request W_i. Hence, the delivery rate for the RANDOM DELIVERY procedure in Algorithm 6 is

$$D_{\mathrm{RD}}\left(M_{[K]}\right) \triangleq \sum\nolimits_{i=1}^{N}\left(1-\frac{M_{S_{i-1}+1}}{N}\right). \qquad \text{(A.21)}$$

Observe that the worst-case user demand combination corresponding to delivery rate $D_{\mathrm{RD}}\left(M_{[K]}\right)$ happens (i.e., the delivery rate $D_{\mathrm{RD}}\left(M_{[K]}\right)$ is maximized) when M_j, $\forall j \in \mathcal{P}$ forms the set of N smallest cache capacities, i.e., the N users with the smallest cache capacities should request different files, which is consistent with the worst-case user demand combination corresponding to $D_{\mathrm{CD}}\left(M_{[K]}\right)$. If the users are labelled such that

$M_1 \leq M_2 \leq \cdots \leq M_K$, then we have

$$D_{\mathrm{RD}}\left(M_{[K]}\right) = \sum\nolimits_{i=1}^{N} \left(1 - \frac{M_i}{N}\right). \tag{A.22}$$

We emphasize here that, before starting the *delivery phase*, it is assumed that each user sends its demand together with its cache contents to the server. With this information, the server can perform the delivery procedure which requiers a smaller delivery rate (by comparing (A.20) and (A.22)), and the following delivery rate is achievable:

$$D_{\mathrm{GBD}}\left(M_{[K]}\right) \triangleq \min\left\{D_{\mathrm{CD}}\left(M_{[K]}\right), D_{\mathrm{RD}}\left(M_{[K]}\right)\right\}, \tag{A.23}$$

which completes the proof of Theorem 2.5.

Appendix B

Proofs for Chapter 3

B.1 Proof of Theorem 3.1

The rate of the coded content targeted to a group of weak receivers for each message of the delivery phase is allocated such that it can be decoded by the weakest receiver among the intended group of weak receivers.

With sub-message j of message 1 of length $\beta_{1,j}n$ channel uses, $W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],j}}$, is given in (3.40), a message of rate $R^{(q)}/\binom{K_w}{q}$ is transmitted to the weak receivers in $\mathcal{S}^{q+1}_{[K_w],j}$, for $j \in \left[\binom{K_w}{q+1}\right]$. The rate of $W^{(q)}_{\mathcal{S}^{q+1}_{[K_w],j}}$ is set such that the weakest receiver in $\mathcal{S}^{q+1}_{[K_w],j}$ can decode it, i.e.,

$$\frac{R^{(q)}}{\binom{K_w}{q}} \leq \beta_{1,j} \left(1 - \max_{r \in \mathcal{S}^{q+1}_{[K_w],j}} \{\delta_r\} \right) F, \quad \text{for } j \in \left[\tbinom{K_w}{q+1}\right]. \tag{B.1}$$

Summing over all the sets $\mathcal{S}^{q+1}_{[K_w],j}$, for $j \in \left[\binom{K_w}{q+1}\right]$, one can obtain

$$\frac{R^{(q)}}{\binom{K_w}{q}} \sum_{r=1}^{K_w - q} \frac{\binom{K_w - r}{q}}{1 - \delta_r} \leq \sum_{j=1}^{\binom{K_w}{q+1}} \beta_{1,j} F = \beta_1 F. \tag{B.2}$$

Note that with the codeword given in (3.45), $W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j},m}$, targeted for the receivers in $\mathcal{S}^{i+1}_{[K_w],j}$, is of rate $R^{(i)}/\left(K_s\binom{K_w}{i}\right)$, while $W^{(i+1)}_{d_{K_w+m},\mathcal{S}^{i+1}_{[K_w],j}}$, destined for receiver $K_w + m$, is of rate $R^{(i+1)}/\binom{K_w}{i+1}$, for $m \in [K_s]$, $i = q-1, \dots, p$ and $j \in \left[\binom{K_w}{i+1}\right]$. Proposition 3.3.2 suggests that the codeword in (3.45) can be decoded correctly by the intended receivers if, for $m \in [K_s]$,

$$\max\left\{ \frac{R^{(i)}/\left(K_s\binom{K_w}{i}\right)}{\left(1-\max\limits_{r\in\mathcal{S}^{i+1}_{[K_w],j}}\{\delta_r\}\right)F}, \frac{R^{(i)}/\left(K_s\binom{K_w}{i}\right)+R^{(i+1)}/\binom{K_w}{i+1}}{\left(1-\delta_{K_w+m}\right)F} \right\} \leq \beta_{q-i+1,j,m}, \quad \text{(B.3)}$$

where the rate of $W^{(i)}_{\mathcal{S}^{i+1}_{[K_w],j},m}$ is limited by the weakest receiver in $\mathcal{S}^{i+1}_{[K_w],j}$, for $i = q-1,\dots,p$, and $j \in \left[\binom{K_w}{i+1}\right]$. By summing up all the K_s inequalities in (B.3), we have, for $i = q-1,\dots,p$ and $j \in \left[\binom{K_w}{i+1}\right]$,

$$\max\left\{ \frac{R^{(i)}/\binom{K_w}{i}}{\left(1-\max\limits_{r\in\mathcal{S}^{i+1}_{[K_w],j}}\{\delta_r\}\right)F}, \left(\frac{R^{(i)}}{K_s\binom{K_w}{i}}+\frac{R^{(i+1)}}{\binom{K_w}{i+1}}\right)\sum_{m=1}^{K_s}\frac{1}{\left(1-\delta_{K_w+m}\right)F} \right\} \leq \beta_{q-i+1,j}. \quad \text{(B.4)}$$

By the choice of (3.35), and the fact that

$$\gamma\left(p,\boldsymbol{\delta},i+1\right) = \gamma\left(p,\boldsymbol{\delta},i\right)\frac{\binom{K_w}{i+1}}{\binom{K_w}{i}K_s}\left(\frac{K_s}{\left(1-\delta_{K_w-i}\right)\sum_{l=K_w+1}^{K}\frac{1}{1-\delta_l}}-1\right), \quad \text{(B.5)}$$

which follows from the definition in (3.36), the second term of the maximization in (B.4) is reduced to $\frac{R^{(i)}/\binom{K_w}{i}}{(1-\delta_{K_w-i})F}$. Thus, (B.4) is simplified as follows:

$$\max\left\{ \frac{R^{(i)}/\binom{K_w}{i}}{\left(1-\max_{r\in\mathcal{S}^{i+1}_{[K_w],j}}\{\delta_r\}\right)F}, \frac{R^{(i)}/\binom{K_w}{i}}{\left(1-\delta_{K_w-i}\right)F} \right\} \leq \beta_{q-i+1,j}. \quad \text{(B.6)}$$

Note that $\left|\mathcal{S}^{i+1}_{[K_w],j}\right| = i+1$; hence, for $i = q-1,...,p$,

$$\max_{r\in\mathcal{S}^{i+1}_{[K_w],j}}\{\delta_r\} \geq \delta_{K_w-i}, \quad \forall j \in \left[\binom{K_w}{i+1}\right]. \quad \text{(B.7)}$$

From (B.7), (B.6) is reduced to

$$\frac{R^{(i)}/\binom{K_w}{i}}{\left(1-\max\limits_{r\in\mathcal{S}^{i+1}_{[K_w],j}}\{\delta_r\}\right)F} \leq \beta_{q-i+1,j}, \quad \text{for } i = q-1,\dots,p,\ j \in \left[\binom{K_w}{i+1}\right], \quad \text{(B.8)}$$

which holds for every $j \in \left[\binom{K_w}{i+1}\right]$, each corresponding to a different $(i+1)$-element subset $\mathcal{S}^{i+1}_{[K_w],j}$. After summing up over all values of j, one can obtain

$$\frac{R^{(i)}}{\binom{K_w}{i}} \sum\nolimits_{r=1}^{K_w-i} \frac{\binom{K_w-r}{i}}{1-\delta_r} \leq \sum\nolimits_{j=1}^{\binom{K_w}{i+1}} \beta_{q-i+1,j} F = \beta_{q-i+1} F, \quad \text{for } i = q-1, \dots, p. \tag{B.9}$$

According to Proposition 3.3.1, each receiver l, $l \in [K_w+1 : K]$, can decode subfile $W^{(p)}_{d_k}$ of rate $R^{(p)}$, delivered by the last message, correctly, if

$$R^{(p)} \sum\nolimits_{l=K_w+1}^{K} \frac{1}{1-\delta_l} \leq \beta_{q-p+2} F. \tag{B.10}$$

By combining inequalities (B.2), (B.9) and (B.10), we have

$$\sum_{i=p}^{q} \left(\frac{R^{(i)}}{\binom{K_w}{i}} \sum_{j=1}^{K_w-i} \frac{\binom{K_w-j}{i}}{1-\delta_j} \right) + R^{(p)} \sum_{j=K_w+1}^{K} \frac{1}{1-\delta_j} \leq \sum_{i=p-1}^{q} \beta_{q-i+1} F = F. \tag{B.11}$$

Finally, by replacing $R^{(i)}$, for $i \in [p : q]$, with the expression in (3.35), one can obtain

$$R \leq \frac{F \sum_{i=p}^{q} \gamma\left(p, \boldsymbol{\delta}, i\right)}{\sum_{i=p}^{q} \left(\frac{\gamma(p,\boldsymbol{\delta},i)}{\binom{K_w}{i}} \sum_{j=1}^{K_w-i} \frac{\binom{K_w-j}{i}}{1-\delta_j} \right) + \sum_{j=K_w+1}^{K} \frac{1}{1-\delta_j}}, \tag{B.12}$$

which, together with the cache capacity of each weak receiver, M, given in (3.39), proves the achievability of the memory-rate pairs $\left(M_{(p,q)}, R_{(p,q)}\right)$ in (3.46).

Appendix C

Proofs for Chapter 4

C.1 Proof of Theorem 4.3

For any given $\mathcal{U}_{\mathbf{d}}$, let $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ denote the set of all demand vectors with the same $\mathcal{U}_{\mathbf{d}}$. The union $\bigcup_{\mathcal{U}_{\mathbf{d}}} \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ form the set of all possible demand vectors $[N]^K$. Therefore, the set of all possible demand vectors can be broken into classes $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ based on the $\mathcal{U}_{\mathbf{d}}$ set they correspond to.

For any given $\mathcal{U}_{\mathbf{d}}$, and an (n, R, M) code as defined in (4.1), (4.3) and (4.5) in Section 4.3, define the error probability as follows:

$$P_{e_{\mathcal{U}_{\mathbf{d}}}} \triangleq \Pr\left\{\bigcup\nolimits_{\mathbf{d}\in\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}} \bigcup\nolimits_{k\in\mathcal{U}_{\mathbf{d}}} \left\{\hat{W}_{d_k} \neq W_{d_k}\right\}\right\}. \tag{C.1}$$

Let $P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})$ denote the average power of the codeword this code generates for a demand vector $\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$. We say that an $\left(R, M, \bar{P}, \hat{P}\right)$ tuple is $\mathcal{U}_{\mathbf{d}}$-*achievable* if for every $\varepsilon > 0$, there exists an (n, R, M) code with sufficiently large n, which satisfies $P_{e_{\mathcal{U}_{\mathbf{d}}}} < \varepsilon$, $\mathrm{E}_{\mathbf{d}}\left[P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})\right] \leq \bar{P}$, and $P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d}) \leq \hat{P}$, $\forall \mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$. We can also define $\bar{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ and $\hat{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ as in (4.6a) and (4.6b), respectively, by considering $\mathcal{U}_{\mathbf{d}}$-achievable codes.

We note from (C.1) that, a $\mathcal{U}_{\mathbf{d}}$-achievable code satisfies only the demands of the users in set $\mathcal{U}_{\mathbf{d}}$. Accordingly, an achievable $\left(R, M, \bar{P}, \hat{P}\right)$ tuple is also $\mathcal{U}_{\mathbf{d}}$-achievable, since $P_e \geq P_{e_{\mathcal{U}_{\mathbf{d}}}}$, for any $\mathcal{U}_{\mathbf{d}}$ set. Thus, lower bounds on $\bar{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ and $\hat{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ also serve as lower bounds on $\bar{P}^*(R, M)$ and $\hat{P}^*(R, M)$, respectively. In the following, we provide lower bounds on $\bar{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ and $\hat{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$.

Let $\left(R, M, \bar{P}, \hat{P}\right)$ be any $\mathcal{U}_{\mathbf{d}}$-achievable tuple. For uniformly distributed demands, we have

$$\bar{P} \geq \mathrm{E}_{\mathbf{d}}\left[P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})\right] = \mathrm{E}_{\mathcal{U}_{\mathbf{d}}}\left[\frac{1}{N_{\mathcal{U}_{\mathbf{d}}}} \sum_{\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}} P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})\right], \tag{C.2}$$

where we used the fact that the probability of each demand vector in $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ is equal. We divide the set of demand vectors $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}$ into different subsets according to the demands of users in $\mathcal{U}_{\mathbf{d}}$, where each subset consists of the demand vectors for which the demands of all the users in $\mathcal{U}_{\mathbf{d}}$ are the same. Note that, there are $\mathfrak{N} \triangleq \binom{N}{N_{\mathbf{d}}} N_{\mathbf{d}}!$ such subsets[1], denoted by $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^l$, for $l = 1, ..., \mathfrak{N}$, i.e., $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}} = \bigcup_{l=1}^{\mathfrak{N}} \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^l$. We note that the number of demand vectors in each $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^l$, denoted by $N_{\mathcal{U}_{\mathbf{d}}}'$, is the same, and is given by

$$N_{\mathcal{U}_{\mathbf{d}}}' = \prod\nolimits_{j=2}^{N_{\mathbf{d}}} j^{u_{j+1}-u_j-1}, \tag{C.3}$$

where, we remind that $\mathcal{U}_{\mathbf{d}} = \{u_1, u_2, ..., u_{N_{\mathbf{d}}}\}$, where $1 = u_1 \leq u_2 \leq \cdots \leq u_{N_{\mathbf{d}}}$. Thus, we have $N_{\mathcal{U}_{\mathbf{d}}} = \mathfrak{N} N_{\mathcal{U}_{\mathbf{d}}}'$, and (C.2) can be rewritten as follows:

$$\bar{P} \geq \mathrm{E}_{\mathcal{U}_{\mathbf{d}}}\left[\frac{1}{N_{\mathcal{U}_{\mathbf{d}}}} \sum\nolimits_{\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}} P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})\right] = \mathrm{E}_{\mathcal{U}_{\mathbf{d}}}\left[\frac{1}{\mathfrak{N}} \sum_{l=1}^{\mathfrak{N}} \left(\frac{1}{N_{\mathcal{U}_{\mathbf{d}}}'} \sum\nolimits_{\mathbf{d} \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^l} P_{\mathcal{U}_{\mathbf{d}}}(\mathbf{d})\right)\right]. \tag{C.4}$$

For any arbitrary demand vector $\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l \in \mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^l$, for $l \in [\mathfrak{N}]$, it is proved in [92, Lemma 14] that there exist random variables[2] $X\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right)$, $Y_{\pi_{\mathcal{U}_{\mathbf{d}}(1)}}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right), \ldots, Y_{\pi_{\mathcal{U}_{\mathbf{d}}(N_{\mathbf{d}})}}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right)$, and $\left\{V_1\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right), ..., V_{N_{\mathbf{d}}-1}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right)\right\}$, where

$$V_1\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \to \cdots \to V_{N_{\mathbf{d}}-1}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \to X\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \to Y_{\pi_{\mathcal{U}_{\mathbf{d}}(N_{\mathbf{d}})}}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \to \cdots \to Y_{\pi_{\mathcal{U}_{\mathbf{d}}(1)}}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \tag{C.5}$$

forms a Markov chain, and satisfy

$$R - \varepsilon_n \leq \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}}; U_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}\right) + I\left(V_{\mathcal{U}_{\mathbf{d}},1}; Y_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right)\right), \tag{C.6a}$$

$$\begin{aligned} R - \varepsilon_n \leq & \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}}; U_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}, \ldots, U_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)} \middle| W_{d^l_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}}, \ldots, W_{d^l_{\pi_{\mathcal{U}_{\mathbf{d}}}(i-1)}}\right) \\ & + I\left(V_i\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right); Y_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right) \middle| V_{i-1}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^l\right)\right), \forall i \in [2 : N_{\mathbf{d}} - 1], \end{aligned} \tag{C.6b}$$

[1] For simplicity, we drop the dependence of $\mathfrak{N}$ on N and $N_{\mathbf{d}}$.

[2] For ease of presentation, we drop the dependence of the transmitted signal X^n, and the received signals Y_k^n, $\forall k \in [K]$, on the library $\mathbf{W}$.

$$\begin{aligned} R-\varepsilon_n \leq & \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)},\dots,U_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})} \middle| W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(1)}},\dots,W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d}-1)}}\right) \\ & + I\left(X\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right); Y_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \middle| V_{N_\mathbf{d}-1}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right), \end{aligned} \tag{C.6c}$$

where $d^l_{\pi_{\mathcal{U}_\mathbf{d}}(i)}$ is the $\pi_{\mathcal{U}_\mathbf{d}}(i)$-th element of demand vector $\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}$, $i \in [N_\mathbf{d}]$, and $\varepsilon_n > 0$ tends to zeros as $n \to \infty$. We note that, due to the independence of the files and the fact that the users in $\mathcal{U}_\mathbf{d}$ demand distinct files, for any uncoded cache placement phase and any $\mathcal{U}_\mathbf{d}$ set, we have

$$\begin{aligned} & I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(i)}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)},\dots,U_{\pi_{\mathcal{U}_\mathbf{d}}(i)} \middle| W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(1)}},\dots,W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(i-1)}}\right) \\ & = I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(i)}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)},\dots,U_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\right), \quad \forall i \in [2:N_\mathbf{d}], l \in [\mathfrak{N}]. \end{aligned} \tag{C.7}$$

Thus, for an uncoded cache placement phase, (C.6) is equivalent to

$$R-\varepsilon_n \leq \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(1)}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)}\right) + I\left(V_1\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right); Y_{\pi_{\mathcal{U}_\mathbf{d}}(1)}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right), \tag{C.8a}$$

$$\begin{aligned} R-\varepsilon_n \leq & \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(i)}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)},\dots,U_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\right) \\ & + I\left(V_i\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right); Y_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \middle| V_{i-1}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right), \forall i \in [2:N_\mathbf{d}-1], \end{aligned} \tag{C.8b}$$

$$\begin{aligned} R-\varepsilon_n \leq & \frac{1}{n} I\left(W_{d^l_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}}; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)},\dots,U_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}\right) \\ & + I\left(X\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right); Y_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \middle| V_{N_\mathbf{d}-1}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right), \end{aligned} \tag{C.8c}$$

For the Gaussian channel (4.2), for $i = 1,...,N_\mathbf{d}$, we have [147]

$$\begin{aligned} & I\left(V_i\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right); Y_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \middle| V_{i-1}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right) \leq \\ & \frac{1}{2}\log_2\left(1+\frac{\beta_i h^2_{\pi_{\mathcal{U}_\mathbf{d}}(i)} P_{\mathcal{U}_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)}{h^2_{\pi_{\mathcal{U}_\mathbf{d}}(i)} \sum_{j=i+1}^{N_\mathbf{d}} \beta_j P_{\mathcal{U}_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)+1}\right), \end{aligned} \tag{C.9}$$

for some $\beta_i \geq 0$, for $i = 1,...,N_\mathbf{d}$, such that $\sum_{i=1}^{N_\mathbf{d}} \beta_i = 1$, where we set $V_0\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \triangleq 0$, and $V_{N_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right) \triangleq X\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)$. From (C.8) and (C.9), for n sufficiently large, the average power $P_{\mathcal{U}_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)$ to satisfy any demand vector $\mathbf{d}^l_{\mathcal{U}_\mathbf{d}} \in \mathcal{D}^l_{\mathcal{U}_\mathbf{d}}$, for $l \in [D]$, is lower bounded by

$$P_{\mathcal{U}_{\mathbf{d}}}\left(\mathbf{d}_{\mathcal{U}_{\mathbf{d}}}^{l}\right) \geq \mathfrak{F}\left(c_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}^{l}, \ldots, c_{\pi_{\mathcal{U}_{\mathbf{d}}}(N_{\mathbf{d}})}^{l}\right) \triangleq \sum_{i=1}^{N_{\mathbf{d}}}\left(\frac{2^{2c_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l}}-1}{h_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{2}}\right)\prod_{j=1}^{i-1} 2^{2c_{\pi_{\mathcal{U}_{\mathbf{d}}}(j)}^{l}}, \quad \text{(C.10a)}$$

where, for $i = 1, ..., N_{\mathbf{d}}$ and $l = 1, ..., \mathfrak{N}$,

$$c_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l} \triangleq R - \frac{1}{n} I\left(W_{d_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l}}; U_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}, \ldots, U_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}\right). \quad \text{(C.10b)}$$

Note that the lower bound in (C.10a) does not depend on any particular demand in $\mathcal{D}_{\mathcal{U}_{\mathbf{d}}}^{l}$, $l \in [\mathfrak{N}]$. Thus, from (C.4) and (C.10), we have

$$\bar{P} \geq \mathrm{E}_{\mathcal{U}_{\mathbf{d}}}\left[\frac{1}{\mathfrak{N}}\sum\nolimits_{l=1}^{\mathfrak{N}} f\left(c_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}^{l}, \ldots, c_{\pi_{\mathcal{U}_{\mathbf{d}}}(N_{\mathbf{d}})}^{l}\right)\right]. \quad \text{(C.11)}$$

Lemma C.1. *Given a set of users $\mathcal{U}_{\mathbf{d}}$ of size $N_{\mathbf{d}}$ with distinct demands, we have*

$$\frac{1}{\mathfrak{N}}\sum\nolimits_{l=1}^{\mathfrak{N}} f\left(c_{\pi_{\mathcal{U}_{d}}(1)}^{l}, \ldots, c_{\pi_{\mathcal{U}_{d}}(N_{d})}^{l}\right) \geq$$
$$\sum\nolimits_{i=1}^{N_{\mathbf{d}}}\left(\frac{2^{2R(1-\min\{iM/N,1\})}-1}{h_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{2}}\right)\prod\nolimits_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})}. \quad \text{(C.12)}$$

Proof. It is proved in Appendix C.2 that $\mathfrak{F}(\cdot)$ is a convex function of its arguments. Thus,

$$\frac{1}{\mathfrak{N}}\sum_{l=1}^{\mathfrak{N}} \mathfrak{F}\left(c_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}^{l}, \ldots, c_{\pi_{\mathcal{U}_{\mathbf{d}}}(N_{\mathbf{d}})}^{l}\right) \geq \mathfrak{F}\left(\frac{1}{\mathfrak{N}}\sum_{l=1}^{\mathfrak{N}} c_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}^{l}, ..., \frac{1}{\mathfrak{N}}\sum_{l=1}^{\mathfrak{N}} c_{\pi_{\mathcal{U}_{\mathbf{d}}}(N_{\mathbf{d}})}^{l}\right). \quad \text{(C.13)}$$

From the definition, we have, for $i = 1, ..., N_{\mathbf{d}}$,

$$\frac{1}{\mathfrak{N}}\sum\nolimits_{l=1}^{\mathfrak{N}} c_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l} = R - \frac{1}{n\mathfrak{N}}\sum\nolimits_{l=1}^{\mathfrak{N}} I\left(W_{d_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l}}; U_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}, \ldots, U_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}\right). \quad \text{(C.14)}$$

where, due to the symmetry, each file W_k, for $k = 1, ..., N$, appears $\binom{N-1}{N_{\mathbf{d}}-1}(N_{\mathbf{d}}-1)!$ times in the sum on the right hand side of (C.14) for each i value. Thus, for $i = 1, ..., N_{\mathbf{d}}$, we have

$$\frac{1}{\mathfrak{N}}\sum\nolimits_{l=1}^{\mathfrak{N}} c_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}^{l} = R - \frac{\binom{N-1}{N_{\mathbf{d}}-1}(N_{\mathbf{d}}-1)!}{n\mathfrak{N}}\sum\nolimits_{k=1}^{N} I\left(W_k; U_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}, \ldots, U_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}\right)$$

$$\begin{aligned}
&= R - \frac{1}{nN} \sum\nolimits_{k=1}^{N} I\left(W_k; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)}, \dots, U_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\right) \\
&\overset{\text{(a)}}{\geq} R - \frac{1}{nN} I\left(W_1, ..., W_N; U_{\pi_{\mathcal{U}_\mathbf{d}}(1)}, \dots, U_{\pi_{\mathcal{U}_\mathbf{d}}(i)}\right) \\
&= R - \frac{R}{N} \min\{iM, N\}, \qquad \text{(C.15)}
\end{aligned}$$

where (a) follows from the independence of the files. From (C.13) and (C.15), and the definition of function $\mathfrak{F}$, we have

$$\begin{aligned}
&\frac{1}{\mathfrak{N}} \sum\nolimits_{l=1}^{\mathfrak{N}} \mathfrak{F}\left(c^l_{\pi_{\mathcal{U}_\mathbf{d}}(1)}, \dots, c^l_{\pi_{\mathcal{U}_\mathbf{d}}(N_\mathbf{d})}\right) \geq \\
&\mathfrak{F}\left(R\left(1 - \min\left\{\frac{iM}{N}, 1\right\}\right), \dots, R\left(1 - \min\left\{\frac{iM}{N}, 1\right\}\right)\right), \qquad \text{(C.16)}
\end{aligned}$$

which concludes the proof of Lemma C.1. □

According to (C.11) and Lemma C.1, $\bar{P}$ is lower bounded by $\bar{P}_{\text{LB}}(R, M)$ defined in (4.52). Thus, $\bar{P}_{\text{LB}}(R, M)$ is a lower bound on $\bar{P}^*_{\mathcal{U}_\mathbf{d}}(R, M)$ as well as $\bar{P}^*(R, M)$.

Next, we prove the lower bound on $\hat{P}^*(R, M)$ stated in Theorem 4.3. For any $\mathcal{U}_\mathbf{d}$ set, let $\left(R, M, \bar{P}, \hat{P}\right)$ tuple be $\mathcal{U}_\mathbf{d}$-achievable. We have $\hat{P} \geq P_{\mathcal{U}_\mathbf{d}}(\mathbf{d})$, $\forall \mathbf{d} \in \mathcal{D}_{\mathcal{U}_\mathbf{d}}$, which is equivalent to

$$\hat{P} \geq P_{\mathcal{U}_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right), \quad \forall \mathbf{d}^l_{\mathcal{U}_\mathbf{d}} \in \mathcal{D}^l_{\mathcal{U}_\mathbf{d}}, \text{for } l = 1, ..., \mathfrak{N}. \qquad \text{(C.17)}$$

Averaging over all $N_{\mathcal{U}_\mathbf{d}} = \mathfrak{N} N'_{\mathcal{U}_\mathbf{d}}$ possible demands with the same $\mathcal{U}_\mathbf{d}$ set, we have

$$\hat{P} \geq \frac{1}{\mathfrak{N}} \sum\nolimits_{l=1}^{\mathfrak{N}} \left(\frac{1}{N'_{\mathcal{U}_\mathbf{d}}} \sum\nolimits_{\mathbf{d}^l_{\mathcal{U}_\mathbf{d}} \in \mathcal{D}^l_{\mathcal{U}_\mathbf{d}}} P_{\mathcal{U}_\mathbf{d}}\left(\mathbf{d}^l_{\mathcal{U}_\mathbf{d}}\right)\right). \qquad \text{(C.18)}$$

According to (C.10) and Lemma C.1, $\hat{P}$ is lower bounded as follows:

$$\hat{P} \geq \sum\nolimits_{i=1}^{N_\mathbf{d}} \left(\frac{2^{2R(1-\min\{iM/N,1\})} - 1}{h^2_{\pi_{\mathcal{U}_\mathbf{d}}(i)}}\right) \prod\nolimits_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})}, \quad \forall \mathcal{U}_\mathbf{d}. \qquad \text{(C.19)}$$

Thus, we have

$$\hat{P} \geq \max_{\mathcal{U}_\mathbf{d}} \left\{\sum\nolimits_{i=1}^{N_\mathbf{d}} \left(\frac{2^{2R(1-\min\{iM/N,1\})} - 1}{h^2_{\pi_{\mathcal{U}_\mathbf{d}}(i)}}\right) \prod\nolimits_{j=1}^{i-1} 2^{2R(1-\min\{jM/N,1\})}\right\}. \qquad \text{(C.20)}$$

We note that the term on the right hand side of the inequality in (C.20) is equivalent to $\hat{P}_{\mathrm{LB}}(R, M)$ defined in (4.53). Thus, $\hat{P}_{\mathrm{LB}}(R, M)$ is a lower bound on $\hat{P}^*_{\mathcal{U}_{\mathbf{d}}}(R, M)$ as well as $\hat{P}^*(R, M)$.

C.2 Proof of Convexity of Function $\mathfrak{F}(\cdot)$

We show that, for $1 \leq N_{\mathbf{d}} \leq K$, function $\mathfrak{F} : \mathbb{R}^{N_{\mathbf{d}}} \rightarrow \mathbb{R}$ is a convex function of $(s_1, ..., s_{N_{\mathbf{d}}})$:

$$\mathfrak{F}(s_1, \ldots, s_{N_{\mathbf{d}}}) = \sum_{i=1}^{N_{\mathbf{d}}} \left(\frac{2^{2s_k} - 1}{h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}} \right) \prod_{j=1}^{i-1} 2^{2s_j}. \tag{C.21}$$

After mathematical manipulation, one can obtain

$$\mathfrak{F}(s_1, \ldots, s_{N_{\mathbf{d}}}) = \sum_{i=1}^{N_{\mathbf{d}}} \left(\left(\frac{1}{h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)}} - \frac{1}{h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(i+1)}} \right) 2^{2\mathbf{a}_i^T \mathbf{s}} \right) - \frac{1}{h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(1)}}, \tag{C.22}$$

where $h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(N_{\mathbf{d}}+1)} \triangleq \infty$, $\mathbf{s} \triangleq [s_1 \; s_2 \; \cdots \; s_{N_{\mathbf{d}}}]^T$, and

$$\mathbf{a}_i \triangleq \Big[\underbrace{1\,1 \cdots 1}_{1 \times i} \, \underbrace{0\,0 \cdots 0}_{1 \times (N_{\mathbf{d}} - i)} \Big]^T, \quad \text{for } i = 1, ..., N_{\mathbf{d}}. \tag{C.23}$$

We note that $h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(i)} \leq h^2_{\pi_{\mathcal{U}_{\mathbf{d}}}(i+1)}$, $\forall i \in [N_{\mathbf{d}}]$, and function 2^{2s}, $s \in \mathbb{R}$, is a convex function of s. Thus, all the functions $2^{2\mathbf{a}_i^T \mathbf{s}}$, $\forall i \in [N_{\mathbf{d}}]$, are convex since the affine substitution of the arguments preserves convexity. Hence, function $\mathfrak{F}$ is convex with respect to $(s_1, ..., s_{N_{\mathbf{d}}})$ since any linear combination of convex functions with non-negative coefficients is convex.

Appendix D

Proofs for Chapter 5

D.1 Proof of Theorem 5.2

For ease of presentation, we prove the outer bound for $\mathcal{G} = [K]$, and the proof of general case follows similarly. By an abuse of the notation, for a demand vector $\mathbf{d}$ with distinct entries and noise variances $\boldsymbol{\sigma}$ in the delivery phase, we denote the channel input, generated by function $\psi_{\boldsymbol{\sigma},\mathbf{d}}$, by $X_{\mathbf{d}}^n$, and the channel output at user k by $Y_{\mathbf{d},k}^n$, where

$$Y_{\mathbf{d},k}^n = X_{\mathbf{d}}^n + Z_k^n, \quad \text{for } k \in [K]. \tag{D.1}$$

Lemma D.1. *Let $(R_1, \dots, R_K)$ be an achievable rate tuple. For a demand vector $\mathbf{d} = (d_1, \dots, d_K)$ with all distinct entries, there exist random variables $X_{\mathbf{d}}$, $Y_{\mathbf{d},1}, \dots, Y_{\mathbf{d},K}$, and $V_{\mathbf{d},1}, \dots, V_{\mathbf{d},K-1}$, where*

$$V_{\mathbf{d},1} \to \cdots \to V_{\mathbf{d},K-1} \to X_{\mathbf{d}} \to Y_{\mathbf{d},K} \to \cdots \to Y_{\mathbf{d},1} \tag{D.2}$$

forms a Markov chain, that satisfy

$$R_1 - \varepsilon \leq I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{1}{n} I\left(W_{d_1}^{(1)}; U_1\right), \tag{D.3a}$$

$$R_k - \varepsilon \leq I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \middle| V_{\mathbf{d},k-1}\right) + \frac{1}{n} I\left(\bigcup_{l=1}^{k} W_{d_k}^{(l)}; U_1, \dots, U_k \middle| \bigcup_{m=1}^{k-1} \bigcup_{l=1}^{m} W_{d_m}^{(l)}\right),$$
$$\forall k \in [2 : K-1], \tag{D.3b}$$

$$R_K - \varepsilon \leq I\left(X_{\mathbf{d}}; Y_{\mathbf{d},K} \middle| V_{\mathbf{d},K-1}\right) + \frac{1}{n} I\left(\bigcup_{l=1}^{K} W_{d_k}^{(l)}; U_1, \dots, U_K \middle| \bigcup_{m=1}^{K-1} \bigcup_{l=1}^{m} W_{d_m}^{(l)}\right), \tag{D.3c}$$

where $\varepsilon > 0$ tends to zero as $n \to \infty$.

Proof. See Appendix D.2. □

Assuming $N \geq K$, let $\mathcal{D}_k$ be the set of all $\binom{N}{k}k!$ k-dimensional vectors, where all entries of each vector are distinct, and each entry of every vector takes a value in $[N]$, for $k \in [K]$. We note that $\mathcal{D}_K$ is the set of all demand vectors, each with all different entries. By averaging over all demand vectors with different entries, we can obtain from Lemma D.1 that

$$R_1 - \varepsilon \leq I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{1}{\binom{N}{K}K!}\sum_{\mathbf{d}\in\mathcal{D}_K}\frac{1}{n}I\left(W_{d_1}^{(1)}; U_1\right) \tag{D.4a}$$

$$= I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{1}{\binom{N}{K}K!}\binom{N-1}{K-1}(K-1)!\sum\nolimits_{j=1}^{N}\frac{1}{n}I\left(W_j^{(1)}; U_1\right) \tag{D.4b}$$

$$= I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{1}{N}\sum\nolimits_{j=1}^{N}\frac{1}{n}I\left(W_j^{(1)}; U_1\right) \tag{D.4c}$$

$$\leq I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{1}{nN}I\left(\mathbf{W}^{(1)}; U_1\right) \tag{D.4d}$$

$$\leq I\left(V_{\mathbf{d},1}; Y_{\mathbf{d},1}\right) + \frac{M_1}{N}, \tag{D.4e}$$

where (D.4d) follows from the independence of the files, and, for $k \in [2:K]$,

$$R_k - \varepsilon \leq I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \left| V_{\mathbf{d},k-1}\right.\right) + \frac{1}{\binom{N}{K}K!}\sum_{\mathbf{d}\in\mathcal{D}_K}\frac{1}{n}I\left(\bigcup_{l=1}^{k}W_{d_k}^{(l)}; U_1,\ldots,U_k \middle| \bigcup_{m=1}^{k-1}\bigcup_{l=1}^{m}W_{d_m}^{(l)}\right) \tag{D.5a}$$

$$= I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \left| V_{\mathbf{d},k-1}\right.\right) + \frac{1}{\binom{N}{K}K!}\sum_{\tilde{\mathbf{d}}\in\mathcal{D}_{k-1}}\sum_{\mathbf{d}\in\mathcal{D}_K:(d_1,\ldots,d_{k-1})=\tilde{\mathbf{d}}}\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)}; U_1,\ldots,U_k \middle| \bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right) \tag{D.5b}$$

$$= I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \left| V_{\mathbf{d},k-1}\right.\right) + \frac{1}{\binom{N}{K}K!}\sum_{\tilde{\mathbf{d}}\in\mathcal{D}_{k-1}}\sum_{\mathbf{d}\in\mathcal{D}_K:(d_1,\ldots,d_{k-1})=\tilde{\mathbf{d}}}\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)}; U_1,\ldots,U_k\right) \tag{D.5c}$$

$$= I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \left| V_{\mathbf{d},k-1}\right.\right) + \frac{1}{\binom{N}{K}K!}\sum_{\tilde{\mathbf{d}}\in\mathcal{D}_{k-1}}\sum_{j\in[N]\setminus\left\{\tilde{d}_1,\ldots,\tilde{d}_{k-1}\right\}}\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{j}^{(l)}; U_1,\ldots,U_k\right)\binom{N-k}{K-k}(K-k)! \tag{D.5d}$$

$$= I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \left| V_{\mathbf{d},k-1}\right.\right) +$$

$$\frac{1}{\binom{N}{K}K!}\sum_{j=1}^{N}\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_j^{(l)};U_1,\dots,U_k\right)\binom{N-k}{K-k}(K-k)!\binom{N-1}{k-1}(k-1)! \quad \text{(D.5e)}$$

$$=I\left(V_{\mathbf{d},k};Y_{\mathbf{d},k}\left|V_{\mathbf{d},k-1}\right.\right)+\frac{1}{N}\sum\nolimits_{j=1}^{N}\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_j^{(l)};U_1,\dots,U_k\right) \quad \text{(D.5f)}$$

$$\leq I\left(V_{\mathbf{d},k};Y_{\mathbf{d},k}\left|V_{\mathbf{d},k-1}\right.\right)+\frac{1}{nN}I\left(\bigcup\nolimits_{l=1}^{k}\mathbf{W}^{(l)};U_1,\dots,U_k\right) \quad \text{(D.5g)}$$

$$\leq I\left(V_{\mathbf{d},k};Y_{\mathbf{d},k}\left|V_{\mathbf{d},k-1}\right.\right)+\frac{1}{N}\sum\nolimits_{i=1}^{k}M_k, \quad \text{(D.5h)}$$

where (D.5c) follows from the assumption of uncoded caching and the independence of the files, $\tilde{d}_i$ in (D.5d), for $i\in[k-1]$, returns the i-th element of vector $\tilde{\mathbf{d}}$, (D.5g) follows from the the independence of the files, and we define $V_{\mathbf{d},K}\triangleq X$. For the Gaussian channel, we have [147]

$$I\left(V_{\mathbf{d},k};Y_{\mathbf{d},k}\left|V_{\mathbf{d},k-1}\right.\right)\leq C_{\sum_{i=k+1}^{K}\eta_i P+\sigma_k^2}^{\eta_k P}, \quad \text{for } k\in[K], \quad \text{(D.6)}$$

for some non-negative coefficients $\eta_1,\dots,\eta_K$, such that $\sum_{i=1}^{K}\eta_i=1$, where we set $V_{\mathbf{d},0}\triangleq 0$. This completes the proof of Theorem 5.2 for $\mathcal{G}=[K]$. The proof can be extended to the general case by taking similar steps.

D.2 Proof of Lemma D.1

We follow the same steps as in [92, Lemma 14], but for multi-layer massages. Given a demand vector $\mathbf{d}$ with all different entries and $\boldsymbol{\sigma}$ in the delivery phase, consider an achievable rate tuple $(R_1,\dots,R_K)$. Thus, there exist K caching functions $\phi_{\boldsymbol{\sigma},1},\dots,\phi_{\boldsymbol{\sigma},K}$, an encoding function $\psi_{\boldsymbol{\sigma},\mathbf{d}}$, and K decoding functions $\mu_{\mathbf{d},1},\dots,\mu_{\mathbf{d},K}$, which, for large enough n, $P_e<\varepsilon$, where ε tends to 0 as $n\to\infty$. From Fano's inequality, we have

$$R_k-\varepsilon\leq\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n,U_k\right), \quad \text{for } k\in[K]. \quad \text{(D.7)}$$

Accordingly,

$$R_1-\varepsilon\leq\frac{1}{n}I\left(W_{d_1}^{(1)};Y_{\mathbf{d},1}^n,U_1\right) \quad \text{(D.8a)}$$

$$=\frac{1}{n}I\left(W_{d_1}^{(1)};U_1\right)+\frac{1}{n}I\left(W_{d_1}^{(1)};Y_{\mathbf{d},1}^n\left|U_1\right.\right), \tag{D.8b}$$

where the second term in (D.8b) can be bounded as follows:

$$\frac{1}{n}I\left(W_{d_1}^{(1)};Y_{\mathbf{d},1}^n\left|U_1\right.\right)=\frac{1}{n}\sum\nolimits_{i=1}^{n}I\left(W_{d_1}^{(1)};Y_{\mathbf{d},1,i}\left|U_1,Y_{\mathbf{d},1}^{i-1}\right.\right) \tag{D.9a}$$

$$\leq\frac{1}{n}\sum\nolimits_{i=1}^{n}I\left(W_{d_1}^{(1)},Y_{\mathbf{d},1}^{i-1};Y_{\mathbf{d},1,i}\left|U_1\right.\right), \tag{D.9b}$$

where we define $Y_{\mathbf{d},k}^i\triangleq(Y_{\mathbf{d},k,1},\dots,Y_{\mathbf{d},k,i})$, for $k\in[K]$ and $i\in[n]$. Let $\mathfrak{E}$ be a random variable uniformly distributed over $[n]$ and independent from all other random variables. We have

$$\frac{1}{n}\sum\nolimits_{i=1}^{n}I\left(W_{d_1}^{(1)},Y_{\mathbf{d},1}^{i-1};Y_{\mathbf{d},1,i}\left|U_1\right.\right)=I\left(W_{d_1}^{(1)},Y_{\mathbf{d},1}^{\mathfrak{E}-1};Y_{\mathbf{d},1,\mathfrak{E}}\left|U_1,\mathfrak{E}\right.\right) \tag{D.10a}$$

$$\leq I\left(W_{d_1}^{(1)},Y_{\mathbf{d},1}^{\mathfrak{E}-1},U_1,\mathfrak{E};Y_{\mathbf{d},1,\mathfrak{E}}\right) \tag{D.10b}$$

$$=I\left(V_{\mathbf{d},1};Y_{\mathbf{d},1}\right), \tag{D.10c}$$

where we define $V_{\mathbf{d},1}\triangleq\left(W_{d_1}^{(1)},Y_{\mathbf{d},1}^{\mathfrak{E}-1},U_1,\mathfrak{E}\right)$, and $Y_{\mathbf{d},1}\triangleq(Y_{\mathbf{d},1,T})$. From (D.8)-(D.10), (D.3a) is proved. We also have, for $k\in[2:K]$,

$$R_k-\varepsilon\leq\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n,U_k\right) \tag{D.11a}$$

$$\leq\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n,U_k\left|\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right.\right) \tag{D.11b}$$

$$\leq\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n,U_1,\dots,U_k\left|\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right.\right) \tag{D.11c}$$

$$=\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};U_1,\dots,U_k\left|\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right.\right)+$$
$$\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n\left|U_1,\dots,U_k,\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right.\right), \tag{D.11d}$$

where (D.11b) follows from the independence of the files. We now bound the second term in (D.11d) as follows:

$$\frac{1}{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k}^n\left|U_1,\dots,U_k,\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)}\right.\right)$$
$$=\frac{1}{n}\sum\nolimits_{i=1}^{n}I\left(\bigcup\nolimits_{l=1}^{k}W_{d_k}^{(l)};Y_{\mathbf{d},k,i}\left|U_1,\dots,U_k,\bigcup\nolimits_{m=1}^{k-1}\bigcup\nolimits_{l=1}^{m}W_{d_m}^{(l)},Y_{\mathbf{d},k}^{i-1}\right.\right) \tag{D.12a}$$

$$\leq \frac{1}{n}\sum_{i=1}^{n} I\left(\bigcup\nolimits_{l=1}^{k} W_{d_k}^{(l)}; Y_{\mathbf{d},k,i} \middle| U_1,\ldots,U_k, \bigcup_{m=1}^{k-1}\bigcup_{l=1}^{m} W_{d_m}^{(l)}, Y_{\mathbf{d},k}^{i-1}, Y_{\mathbf{d},k-1}^{i-1},\ldots,Y_{\mathbf{d},1}^{i-1}\right) \tag{D.12b}$$

$$\leq \frac{1}{n}\sum_{i=1}^{n} I\left(\bigcup_{l=1}^{k} W_{d_k}^{(l)}, Y_{\mathbf{d},k}^{i-1}; Y_{\mathbf{d},k,i} \middle| U_1,\ldots,U_k, \bigcup_{m=1}^{k-1}\bigcup_{l=1}^{m} W_{d_m}^{(l)}, Y_{\mathbf{d},k-1}^{i-1},\ldots,Y_{\mathbf{d},1}^{i-1}\right) \tag{D.12c}$$

$$= I\left(\bigcup_{l=1}^{k} W_{d_k}^{(l)}, Y_{\mathbf{d},k}^{\mathfrak{E}-1}; Y_{\mathbf{d},k,\mathfrak{E}} \middle| U_1,\ldots,U_k, \bigcup_{m=1}^{k-1}\bigcup_{l=1}^{m} W_{d_m}^{(l)}, Y_{\mathbf{d},k-1}^{\mathfrak{E}-1},\ldots,Y_{\mathbf{d},1}^{\mathfrak{E}-1}, \mathfrak{E}\right) \tag{D.12d}$$

$$\leq I\left(\bigcup_{l=1}^{k} W_{d_k}^{(l)}, Y_{\mathbf{d},k}^{\mathfrak{E}-1}, U_k; Y_{\mathbf{d},k,\mathfrak{E}} \middle| U_1,\ldots,U_{k-1}, \bigcup_{m=1}^{k-1}\bigcup_{l=1}^{m} W_{d_m}^{(l)}, Y_{\mathbf{d},k-1}^{\mathfrak{E}-1},\ldots,Y_{\mathbf{d},1}^{\mathfrak{E}-1}, \mathfrak{E}\right) \tag{D.12e}$$

$$= I\left(V_{\mathbf{d},k}; Y_{\mathbf{d},k} \middle| V_{\mathbf{d},k-1}\right), \tag{D.12f}$$

where $V_{\mathbf{d},k} \triangleq \left(V_{\mathbf{d},k-1}, \bigcup_{l=1}^{k} W_{d_k}^{(l)}, Y_{\mathbf{d},k}^{\mathfrak{E}-1}, U_k\right)$, and $Y_{\mathbf{d},k} \triangleq (Y_{\mathbf{d},k,\mathfrak{E}})$, for $k \in [2:K]$. We also note that $V_{\mathbf{d},K} = X_{\mathbf{d}}$. By plugging (D.12) into (D.11), the proof of Lemma D.1 is completed.