UNIVERSITY OF CALIFORNIA,
IRVINE

Privacy, Security, and Flexibility in Distributed Computing

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical and Computer Engineering

by

Zhen Chen

2021

DEDICATION

To my wife and parents, for their unwavering support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I have been truly lucky to be advised by Professor Syed Jafar and Professor Zhiying Wang, from whom I not only learned everything about research, but also the wisdom of life. I have benefited tremendously from their guidance and countless discussions with them. They provide me enough freedom to explore my interests and encourage me to explore the fundamental and conceptually novel problems. Their insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. Without their insights and immense knowledge, this dissertation would not have been possible. I would always remember and learn from their insight, vision, commitment to excellence in my career.

I would like to thank Professor Hamid Jafarkhani for serving on my dissertation committee, and Professor A. Lee Swindlehurst, and Professor Chen Li for serving on my qualifying examination committee. I appreciate all these valuable suggestions from my committee members. I am thankful to all the professors I engaged with, either through courses or personally, over the past five years at UCI. I specifically want to thank Professor Xiaohui Xie, who gave me a chance to exposure to deep learning.

It has been a great pleasure to have co-authored with Zhuqing Jia, Weiqi Li and Marwen Zorgui. I really enjoy the time working with them. My thanks extend to former and current colleagues including Hua Sun, Arash Gholami Davoodi, Bofeng Yuan, Yao-Chia Chan, Junge Wang, Yuxiang Lu, Nilab İsmailoğlu, Alireza Javani, Peng Fei, Wenkai Zhang, Xiaoran Li and Keqing Fu. I am proud of being a part of such a wonderful group.

Next, I would like to thank my friends, Chuan Di, Ting Tang, Guoliang Yan, Kunpeng Zhang, Yubo Wang, Chang Liu, Yang Li, Jin Dang and Jialie Yan. Thanks for their accompanies and friendships, which makes my life more colorful. They were always there for me when I needed them.

Finally, I must express my profound gratitude to my family. Their constant support is endless. I would not reach this far without their guidance. Their endurance of my absence was a true sacrifice that I cannot repay. I especially want to thank my wife and friend, Ruoxi. Saying that she was supportive would be an understatement. She never lost faith in me even in my darkest hours. Ruoxi took care of countless responsibilities, that could distract me from my work, willingly and with sincere love. In addition, our daily discussions about nearly everything provide me with warmth and comfort. This dissertation is dedicated to all of them.

VITA

Zhen Chen

EDUCATION

Doctor of Philosophy in Electrical Engineering **2021**
University of California, Irvine *Irvine, California*

Master of Science in Electronics and Information Engineering **2016**
Beihang University *Beijing, China*

Bachelor of Science in ShenYuan Honors College **2013**
Beihang University *Beijing, China*

RESEARCH EXPERIENCE

Graduate Research Assistant **2016–2021**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2014–2015**
Beihang University *Beijing, China*

ABSTRACT OF THE DISSERTATION

Privacy, Security, and Flexibility in Distributed Computing

By

Zhen Chen

Doctor of Philosophy in Electrical and Computer Engineering

University of California, Irvine, 2021

Assistant Professor Zhiying Wang, Chair
Chancellor's Professor Syed Ali Jafar, Chair

The modern information age is heralded by exciting paradigms that generate a tremendous amount of data, such as health records, financial transactions, and social media user information. As information becomes increasingly available, distributed computing becomes more and more important, especially in information retrieval, search, and computation. The high demand for distributed computing brings many new challenges and concerns. This dissertation focuses on privacy, security, and flexibility issues in distributed computing from an information-theoretic perspective. Specifically, we study the privacy problem via private information retrieval (PIR) with a focus on the scenarios with available side information and private search. The security and flexibility problems are investigated via coded distributed computing. Two settings for matrix multiplication are studied: the secure multi-party computation and the distributed computation with a flexible number of available servers.

PIR allows a user to retrieve one message from databases while preserving the privacy of the desired message index. We investigate the role of side information in PIR, meaning a subset of the messages known by the user. Assume T is the number of possible colluding databases, despite which the privacy of the desired message and the side information should still be maintained. We focus on T-Private Information Retrieval with Private Side Information (TPIR-PSI) and characterize its capacity. A novel achievable scheme is proposed. As a special case obtained by setting T = 1, our result settles the capacity of PIR-PSI, an open problem previously noted by Kadhe et al. We extend our results to the problem of symmetric-TPIR with private side information (STPIR-PSI).

Then, we consider private search, where a user searches for all records matching a symbol from the servers, without revealing any information about the queried symbol. Private search is shown to be highly related to PIR with arbitrarily dependent messages. A new converse bound is presented for PIR with arbitrary dependence. Based on the asymptotic behavior of the new converse bound, the asymptotic capacity of private search is characterized. Our result is further generalized to OR search, AND search, NOT search and sequence search.

As the next step, we consider secure multi-party batch matrix multiplication (SMBMM). Two batches of matrices are individually coded and transmitted to the computation servers, who communicate with each other and a master such that the product of the matrices is decoded at the master. The matrices are kept secure from the servers and the master, except that the master obtains the final product. A solution called Generalized Cross Subspace Alignment codes with Noise Alignment (GCSA-NA) is proposed based on cross-subspace alignment codes. Compared to the state-of-art solution, polynomial sharing, GCSA-NA outperforms it in several key aspects — more efficient and secure inter-server communication, lower latency, flexible inter-server network topology, efficient batch processing, and tolerance to stragglers. Moreover, the idea of noise alignment can be applied to symmetric secure private information retrieval to achieve the asymptotic capacity.

Finally, we focus on the flexibility of the coded distributed matrix multiplication. The goal is also to compute the matrix product from distributed servers. However, servers do not communicate among themselves and the number of stragglers (slow servers) is not known a priori. A novel flexible construction is proposed to fully utilize the computation capability of available servers. The main idea is to require non-straggler servers to finish more sub-tasks to compensate for the effect of the stragglers. The fundamental trade-off between server storage capacity and computation load is investigated.

Chapter 1

Introduction

1.1 Background

In the era of big data, large-scale learning, and the Internet of things, the rapid increase in the amount of information and computation motivates distributed computing to take center-stage. In distributed computing, a large number of devices are involved, and a problem is divided into many tasks, each solved by one or more devices which communicate with each other. Compared to large centralized systems, distributed computing has many advantages, such as scalability growth, fault tolerance, and cost effectiveness.

However, distributed computing also brings many new challenges and concerns, including user privacy and data security for sensitive data such as medical and financial records, computation efficiency under limited resources and/or massive computation load, and flexibility with unknown system parameters. Efficient solutions to the above challenges are essential in promoting distributed computing in the above scenarios, and have been extensively studied in recent decades. For example, introduced in 1995 by Chor, Kushilevitz, Goldreich and Sudan [22, 23], the private information retrieval problem seeks the most efficient way for a user to retrieve a desired message out of K

messages from N distributed servers, each of which stores all the messages, without revealing any information about the identity of the desired message to any individual server. For another example, private search aims to find a solution that efficiently searches for all records, stored in distributed servers, that match a user's privately chosen value such that any server learns nothing about the chosen value. Secure multi-party computation [121] and secure coded distributed computing (CDC) [125] are examples with the goal of creating methods for servers to jointly compute a function while keeping input data secure. And flexible schemes for varying number of available servers are investigated ranging from communication efficient secret sharing [51], adaptive gradient codes [78], private information retrieval [12] to coded distributed computing [91].

The focus of this dissertation is the fundamental understanding of the trade-offs among privacy, security, flexibility, and efficiency of distributed computing. The schemes and converse of several distributed computing scenarios are investigated, including private information retrieval, private search, secure and flexible distributed matrix multiplication.

1.2 Overview of the Dissertation

Chapter 2 considers the problem of T-Private Information Retrieval with private side information. In this problem, N replicated databases store K independent messages, and a user, equipped with a local cache that holds M messages as side information, wishes to retrieve one of the other $K-M$ messages. The desired message index and the side information must remain jointly private even if any T of the N databases collude. The capacity is defined as the maximum number of bits of the desired message per downloaded bit from the servers to the user. We show that the capacity of TPIR-PSI is $\left(1+\frac{T}{N}+\cdots+\left(\frac{T}{N}\right)^{K-M-1}\right)^{-1}$. As a special case obtained by setting $T=1$, this result settles the capacity of PIR-PSI, an open problem proposed by Kadhe et al. We also consider the problem of symmetric-TPIR with private side information, where the answers from all N databases reveal no information about any other message besides the desired message. We show that the capacity of STPIR-PSI is $1 - \frac{T}{N}$ if the databases have access to common randomness (not available to the user) that is independent of the messages, in an amount that is at least $\frac{T}{N-T}$ bits per desired message bit. Otherwise, the capacity of STPIR-PSI is zero.

In Chapter 3, the private search problem is introduced, where a dataset comprised of L i.i.d. records is replicated across N non-colluding servers, and a user wishes to search for all records that match a privately chosen value, without revealing any information about the chosen value to any individual server. Each record contains P symbols, and each symbol takes values uniformly and independently from an alphabet of size K. Considering the large number of records in modern datasets, it is assumed that L is much larger than the alphabet size K. The capacity of private search is the maximum number of bits of desired information that can be retrieved per bit of download. The asymptotic (large K) capacity of private search is shown to be $1 - 1/N$, even when the scope of private search is further generalized to allow OR search, AND search, NOT search and sequence search. The results are based on the asymptotic behavior of a new converse bound for private information retrieval with arbitrarily dependent messages. The asymptotic behavior is also applicable to T-colluding servers or (N, T)-MDS coded servers.

As the next step in expanding the scope of distributed computing, in Chapter 4 a secure multi-party batch matrix multiplication problem is considered, where the goal is to allow a master to efficiently compute the pairwise products of two batches of massive matrices, by distributing the computation across S servers. Any X colluding servers gain no information about the input, and the master gains no additional information about the input beyond the product. A solution called Generalized Cross Subspace Alignment codes with Noise Alignment is proposed in this chapter, based on cross-subspace alignment codes. The state of the art solution to SMBMM is a coding scheme called polynomial sharing (PS) that was proposed by Nodehi and Maddah-Ali [83]. GCSA-NA outperforms PS codes in several key aspects — more efficient and secure inter-server communication, lower latency, flexible inter-server network topology, efficient batch processing, and tolerance to stragglers.

Chapter 5 concentrates on the flexibility of distributed computing. The distributed matrix multiplication problem with an unknown number of stragglers is considered, where the goal is to efficiently and flexibly obtain the product of two massive matrices by distributing the computation across N servers. There are up to $N - R$ stragglers but the exact number is not known a priori. Motivated by reducing the computation load of each server, a flexible solution is proposed to fully utilize the computation capability of available servers. The computing task for each server is separated into several subtasks, constructed based on Entangled Polynomial codes by Yu et al [128]. The final results can be obtained from either a larger number of servers with a smaller amount of computation completed per server or a smaller number of servers with a larger amount of computation completed per server. The required finite field size of the proposed solution is less than $2N$. Moreover, the optimal design parameters such as the partitioning of the input matrices is discussed. Our constructions can also be generalized to other settings such as batch distributed matrix multiplication and secure distributed matrix multiplication. Finally, we conclude this dissertation in Chapter 6.

Chapter 2

The Capacity of T-Private Information Retrieval with Private Side Information

2.1 Introduction

The private information retrieval (PIR) problem investigates the privacy of the contents downloaded from public databases. In the classical form of PIR [23], a user wishes to, as efficiently as possible, retrieve one of K messages that are replicated across N non-colluding databases while preserving the privacy of the desired message index. Since its first formulation by Chor et al. in [23], the PIR problem has been studied extensively in computer science and cryptography under both information-theoretic and computational privacy constraints [9, 38, 41, 89, 123]. While studies of PIR typically seek to optimize both the upload and download costs, recently there has been a burst of activity aimed at *capacity* characterizations for information-theoretic PIR under the assumption of large message sizes, so that the communication cost is dominated by the download cost [7, 16, 96, 101, 102, 107]. The capacity of PIR was defined in [102] as the maximum number of bits of the desired message that can be privately obtained per bit of total downloaded information from all the servers. In order to summarize some of the capacity results for PIR, let us define the function $\Psi(A,B) = \left(1 + A + A^2 + \cdots + A^{B-1}\right)^{-1}$ for positive real number A and positive integer B. Correspondingly, $\Psi(A,\infty) = 1 - A$ for $A < 1$. The capacity of PIR was characterized in [102] as $C_{\text{PIR}} = \Psi(1/N, K)$. The capacity of T-PIR, where the privacy of the user's desired message index must be protected against collusion among any set of up to T servers, was characterized in [105] as $C_{\text{TPIR}} = \Psi(T/N, K)$. The capacity of symmetric PIR (SPIR), where the user learns nothing about the database besides his desired message, was shown in [103] to be $C_{\text{SPIR}} = \Psi(1/N, \infty)$, and the capacity of STPIR, with both symmetric privacy and robustness against collusion among any T servers, was characterized in [114] as $C_{\text{STPIR}} = \Psi(T/N, \infty)$. A number of other variants of PIR have also been investigated, such as PIR with MDS coded storage [7], multi-message PIR [5], multi-round PIR [104], secure PIR [59], and PIR with side information [47, 48, 49, 50, 61, 64, 73, 74, 108, 117, 119]. Especially relevant to the work in this chapter is the problem of PIR with side information.

The recent focus on the capacity of PIR with side information started with the work on cache-aided PIR by Tandon [108], where the user has enough local cache memory to store a fraction r of all messages as side information. In this model, the side information can be any function of the K messages (subject to the storage constraint) and is globally known to both the user and all the databases. The capacity for this setting is characterized in [108] as $\Psi(1/N, K)/(1 - r)$.

Different from [108] which allows side information to be an arbitrary function of the messages, the side information in [61] (and in this paper) can only take the form of M *full messages* cached by the user. Within this model there are several interesting variations depending on the constraints on the privacy of the side information.

- PIR-GSI, or PIR with global side information, implies that the side information is globally known.
- PIR-SI, i.e., PIR with (non-private) side information, corresponds to the case that the side

information is not globally known, but the privacy of the side information need not be preserved.

- PIR-PSI, or PIR with private side information, refers to the setting where the *joint* privacy of both the desired message and the side information must be preserved. This is the focus of the paper.

- PIR-SPSI, or PIR with *separately* private side information, refers to the setting where the privacy of the desired message and the privacy of side information must each be separately preserved (although their joint privacy need not be preserved). In Appendix 2.6 we provide some insights into the capacity of PIR-SPSI.

Out of these four settings, PIR-GSI is rather trivial, and PIR-SPSI has not been studied at all, perhaps because there is insufficient practical motivation for such an assumption. However, the remaining two variants, PIR-PSI and PIR-SI, have indeed drawn much attention, starting with the work of Kadhe et al. in [61].

For PIR-SI with a single database ($N = 1$), Kadhe et al. showed in [61] that the capacity is $\lceil \frac{K}{M+1} \rceil^{-1}$. The single-database setting has seen rapid progress in various directions [47, 48, 49, 50, 64, 73, 74]. However, PIR-SI with *multiple* databases turns out to be considerably more challenging. In [61], Kadhe et al. provided an achievable scheme for PIR-SI with multiple databases ($N > 1$), which achieves the rate $\Psi(1/N, \lceil K/(M+1) \rceil)$. In spite of some progress in this direction [73], the capacity of PIR-SI generally remains open[1] for multiple databases. In addition, the works in [117, 119] consider a different form of side information instead of full messages.

For PIR-PSI with a single database, Kadhe et al. found in [61] that the capacity is $(K-M)^{-1}$. The capacity of PIR-PSI with more than one database was left as an open problem in [61]. Remarkably, neither a general achievable scheme nor a converse was known in this case. It is this open problem

[1]The converse in [73] does not cover the scope of PIR-SI, because the privacy condition assumed in [73] is not a necessary condition for PIR-SI schemes.

that motivates this work.

The first contribution of this chapter is to show that the capacity of PIR-PSI is $C_{\text{PIR-PSI}} = \Psi(1/N, K-M)$, for an arbitrary number of databases N, thus settling this open problem. This allows us to completely order[2] the four variants of PIR with side information that are listed above, in terms of their capacities as PIR-SI $\geq$ PIR-SPSI $\geq$ PIR-PSI $=$ PIR-GSI. Remarkably, all the inequalities can be strict for certain parameters.

As a generalization, we show that the capacity of TPIR-PSI, i.e., PIR-PSI where up to T databases may collude, is $C_{\text{TPIR-PSI}} = \Psi(T/N, K-M)$. Evidently, the effect of private side information on capacity is the same as if the number of messages in TPIR was reduced from K to $K-M$ [105]. Similar to the case with non-colluding databases, this is also the capacity if the side information is globally known to all databases as well.

As the second contribution of this chapter, we characterize the capacity of STPIR-PSI, i.e., PIR with private side information with symmetric privacy and robustness against any T-colluding servers. We show $C_{\text{TPIR-PSI}} = \Psi(T/N, \infty)$, provided that the databases have access to common randomness (not available to the user) in the amount that is at least $T/(N-T)$ bits per queried message bit. Otherwise, the capacity of STPIR-PSI is zero. Note that this is identical to the capacity of STPIR with no side information [114].

The remainder of this chapter is organized as follows. Section 2.2 presents the problem statements. Section 3.3 presents the main results, i.e., the capacity characterizations of TPIR-PSI and STPIR-PSI. The proofs of the capacity results are presented in Section 2.4 and Section 2.5, and we conclude with Section 3.5.

Notation: We use bold font for random variables to distinguish them from deterministic variables, that are shown in normal font. For integers $z_1 < z_2$, $[z_1 : z_2]$ represents the set $\{z_1, z_1+1, \cdots, z_2\}$

[2]Based on progressively tighter privacy constraints, it is already immediately obvious that in terms of their capacities, the settings can be partially ordered as PIR-SI $\geq$ PIR-SPSI $\geq$ PIR-PSI, and PIR-SI$\geq$ PIR-GSI. The main result of this chapter shows that PIR-PSI has the same capacity as PIR-GSI, thus allowing a complete ordering.

and $(z_1 : z_2)$ represents the vector $(z_1, z_1+1, \cdots, z_2)$. The compact notation $[z]$ represents $[1 : z]$ for positive integer z. For random variables $\boldsymbol{W}_i, i = 1, 2, \ldots,$ and a set of positive integers $S = \{s_1, s_2, \cdots, s_n\}$, where $s_1 < s_2 < \cdots < s_n$, the notation $\boldsymbol{W}_S$ represents the vector $(\boldsymbol{W}_{s_1}, \boldsymbol{W}_{s_2}, \cdots, \boldsymbol{W}_{s_n})$. For a matrix G and a vector S, the notation $G[S, :]$ represents the submatrix of G formed by retaining only the rows corresponding to the elements of the vector S. For a matrix G, its transpose is denoted as G'. $\mathbb{F}_q$ represents the finite field of size q.

2.2 Problem Statements

2.2.1 TPIR-PSI: T-Private Information Retrieval with Private Side Information

The TPIR-PSI problem is parametrized by (K, M, N, T). Consider K independent messages $\boldsymbol{W}_{[K]} = (\boldsymbol{W}_1, \cdots, \boldsymbol{W}_K)$, each containing L independent and uniform bits, i.e., their entropy satisfies

$$H(\boldsymbol{W}_1, \cdots, \boldsymbol{W}_K) = H(\boldsymbol{W}_1) + \cdots + H(\boldsymbol{W}_K), \tag{2.1}$$

$$H(\boldsymbol{W}_1) = \cdots = H(\boldsymbol{W}_K) = L. \tag{2.2}$$

There are N databases and each database stores all K messages $\boldsymbol{W}_1, \cdots, \boldsymbol{W}_K$. A user is equipped with a local cache and has M $(M < K)$ messages as side information. Let $\boldsymbol{S} = \{\boldsymbol{i}_1, \boldsymbol{i}_2, \cdots, \boldsymbol{i}_M\}$ be M distinct indices chosen uniformly from $[K]$. These M cached messages are represented as $\boldsymbol{W}_{\boldsymbol{S}} = (\boldsymbol{W}_{\boldsymbol{i}_1}, \cdots, \boldsymbol{W}_{\boldsymbol{i}_M})$. $\boldsymbol{S}$ is not known to the databases. A user wishes to retrieve $\boldsymbol{W}_{\boldsymbol{\Theta}}$, where $\boldsymbol{\Theta}$ is a message index uniformly chosen from $[K] \setminus \boldsymbol{S}$, as efficiently as possible, while revealing no information about $(\boldsymbol{\Theta}, \boldsymbol{S})$ to any colluding subsets of up to T out of the N databases. Note the following independence,

$$H(\boldsymbol{\Theta}, \boldsymbol{S}, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_K) = H(\boldsymbol{\Theta}, \boldsymbol{S}) + \sum_{i=1}^{K} H(\boldsymbol{W}_i). \tag{2.3}$$

In order to retrieve $\boldsymbol{W}_{\boldsymbol{\Theta}}$, the user generates N queries $\boldsymbol{Q}_1^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \cdots, \boldsymbol{Q}_N^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ with the knowledge of $(\boldsymbol{\Theta}, \boldsymbol{S}, \boldsymbol{W}_{\boldsymbol{S}})$. Since the queries are generated with no knowledge of the other $K - M$ messages, the queries must be independent of them,

$$I\left(\boldsymbol{\Theta}, \boldsymbol{S}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_1^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \cdots, \boldsymbol{Q}_N^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{W}_{[K]\setminus \boldsymbol{S}}\right) = 0. \tag{2.4}$$

The user sends query $\boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ to the n^{th} database and in response, the n^{th} database returns an answer $\boldsymbol{A}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ which is a deterministic function of $\boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ and $\boldsymbol{W}_{[K]}$,

$$H\left(\boldsymbol{A}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_K\right) = 0. \tag{2.5}$$

Upon collecting the answers from all N databases, the user must be able to decode the desired message $\boldsymbol{W}_{\boldsymbol{\Theta}}$ based on the queries and side information,

$$\text{[Correctness]}\ H\left(\boldsymbol{W}_{\boldsymbol{\Theta}} \mid \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{S}, \boldsymbol{\Theta}\right) = 0. \tag{2.6}$$

To satisfy the user-privacy constraint that any T colluding databases learn nothing about $(\boldsymbol{\Theta}, \boldsymbol{S})$, the information available to any T databases (queries, answers and stored messages) must be independent of $(\boldsymbol{\Theta}, \boldsymbol{S})$. [3] Let $\mathcal{T}$ be any subset of $[1 : N]$, of cardinality $|\mathcal{T}|= T$. $\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ represents the vector of queries corresponding to $\boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, n \in \mathcal{T}$. $\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ is defined as the answer vector corresponding to $\boldsymbol{A}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, n \in \mathcal{T}$. To satisfy the T-privacy requirement we must have

[3]Note that the joint privacy of $(\boldsymbol{\Theta}, \boldsymbol{S})$ is a stronger constraint than the marginal privacy of each of $\boldsymbol{\Theta}$ and $\boldsymbol{S}$, i.e., $I(\boldsymbol{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$ implies both $I(\boldsymbol{\Theta}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$ and $I(\boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$. However, the reverse is not true, i.e., even if both $I(\boldsymbol{\Theta}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$ and $I(\boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$, this does not imply that $I(\boldsymbol{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}) = 0$.

$\forall \mathcal{T} \subset [1:N], |\mathcal{T}| = T,$

$$\text{[User privacy]}\ I\left(\boldsymbol{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = 0. \tag{2.7}$$

A TPIR-PSI scheme is called *feasible* if it satisfies the correctness constraint (2.6) and the user-privacy constraint (2.7). For a feasible scheme, the TPIR-PSI rate indicates asymptotically how many bits of desired information are retrieved per downloaded bit, and is defined as follows.

$$R_{\text{TPIR-PSI}} \triangleq \lim_{L\to\infty} \frac{L}{D}, \tag{2.8}$$

where D is the expected (over all $\boldsymbol{\Theta}$, $\boldsymbol{S}$, $\boldsymbol{W}_{[K]}$ and random queries) total number of bits downloaded by the user from all the databases. The *capacity*, $C_{\text{TPIR-PSI}}$, is the supremum of $R_{\text{TPIR-PSI}}$ over all feasible schemes.

2.2.2 STPIR-PSI: Symmetric T-Private Information Retrieval with Private Side Information

In symmetric T-colluding private information retrieval, an additional constraint is imposed: database privacy, which means that the user does not learn any information about $\boldsymbol{W}_{[K]}$ beyond the retrieved message, $\boldsymbol{W}_{\boldsymbol{\Theta}}$, and the side information, $\boldsymbol{W}_{\boldsymbol{S}}$. To facilitate database privacy, suppose the databases share a common random variable $\boldsymbol{U}$ that is not known to the user. It has been shown that without such common randomness, symmetric PIR is not feasible when there is more than one message [41, 103]. The common randomness is independent of the messages, the desired messages index, and the side information index, so that

$$H\left(\boldsymbol{\Theta}, \boldsymbol{S}, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_K, \boldsymbol{U}\right) = H\left(\boldsymbol{\Theta}, \boldsymbol{S}\right) + \sum_{i=1}^{K} H\left(\boldsymbol{W}_i\right) + H(\boldsymbol{U}). \tag{2.9}$$

The answering string $\boldsymbol{A}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ is a deterministic function of $\boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}$, $\boldsymbol{W}_{[K]}$ and common randomness $\boldsymbol{U}$,

$$H\left(\boldsymbol{A}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_K, \boldsymbol{U}\right) = 0. \tag{2.10}$$

The correctness condition is the same as (2.6). The user-privacy condition is $\forall \mathcal{T} \subset [1:N], |\mathcal{T}|= T$,

$$\text{[User privacy] } I\left(\boldsymbol{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}, \boldsymbol{U}\right) = 0. \tag{2.11}$$

Database privacy requires that the user learns nothing about $\boldsymbol{W}_{\overline{(\boldsymbol{\Theta},\boldsymbol{S})}} = \boldsymbol{W}_{[K]\setminus(\{\boldsymbol{\Theta}\}\cup\boldsymbol{S})}$, i.e., messages other than his desired message and the side information. Therefore,

$$\text{[DB privacy] } I\left(\boldsymbol{W}_{\overline{(\boldsymbol{\Theta},\boldsymbol{S})}}; \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}, \boldsymbol{S}, \boldsymbol{W}_{\boldsymbol{S}}\right) = 0. \tag{2.12}$$

An STPIR-PSI scheme is called *feasible* if it satisifes the correctness constraint (2.6), the user-privacy constraint (2.11) and the database-privacy constraint (2.12). For a feasible scheme, the STPIR-PSI rate indicates how many bits of desired information are retrieved per downloaded bit. The *capacity*, $C_{\text{STPIR-PSI}}$, is the supremum of rates over all feasible STPIR-PSI schemes.

2.3 Main Results

The following theorem presents our first result, the capacity of TPIR-PSI.

THEOREM 2.1. *For the TPIR-PSI problem with K messages, N databases and M $(M < K)$ side information messages, the capacity is*

$$C_{\textit{TPIR-PSI}} = \left(1 + \frac{T}{N} + \left(\frac{T}{N}\right)^2 + \cdots + \left(\frac{T}{N}\right)^{K-M-1}\right)^{-1}$$

$$= \Psi(T/N, K-M), \tag{2.13}$$

where $\Psi(A,B) = \left(1 + A + A^2 + \cdots + A^{B-1}\right)^{-1}$.

The following observations place Theorem 2.1 in perspective.

REMARK 2.1. *The expression $C_{\text{TPIR-PSI}}$ equals the capacity of TPIR with $K-M$ messages [105]. Evidently, the impact of private side information is equivalent to reducing the effective number of messages from K to $K-M$.*

REMARK 2.2. *Remarkably, the capacity expression in* (2.13) *matches the capacity for the setting where the side information is assumed to be globally known, i.e., if the M side information messages are globally known, then the capacity is also $C_{\text{TPIR-GSI}} = \Psi(T/N, K-M)$. This can be seen as follows. The achievable scheme is the TPIR scheme of [105] after the cached messages are eliminated. To prove the converse by contradiction, suppose the capacity is greater than $\Psi(T/N, K-M)$. Then there is a scheme Π that achieves a larger rate than $\Psi(T/N, K-M)$ in the presence of the M globally known messages. Consider a TPIR problem with $K-M$ messages and no side information. From [105] we know that its capacity is $\Psi(T/N, K-M)$. It can be assumed that there are M globally known dummy messages. With this globally known side information, the user can use scheme Π to retrieve the desired message while achieving a rate larger than $\Psi(T/N, K-M)$, thus exceeding the capacity of TPIR, i.e., creating a contradiction. Therefore, the capacity of TPIR with globally known side information is $\Psi(T/N, K-M)$.*

REMARK 2.3. *It is worthwhile to place the previous remark in perspective with the capacity results in [108], where it is also assumed that the side information is globally available. $C_{\text{TPIR-GSI}}$ is in general less than the capacity expression found in [108]. The reason is that $C_{\text{TPIR-GSI}}$ is the capacity for a setting where the side information can only be M full messages (excluding the desired one). However, in [108], the side information is allowed to be* any *function of all messages. The relaxed setting of [108] should allow a higher capacity in general. For example, if $T=1$ and the amount of side information is ML bits, then the capacity result of [108] corresponds to the expression*

$\Psi(1/N, K)/\big(1 - \frac{M}{K}\big)$. *It is easy to verify that* $C_{\textit{TPIR-GSI}} = \Psi(1/N, K-M) < \Psi(1/N, K)/\big(1 - \frac{M}{K}\big)$ *when* $N \geq 2, K \geq 2, M \in [K-1]$. *Aside from this superficial distinction, it is notable that the essential insight in both settings is the same. The best strategy in the setting of [108] is to cache* $\frac{M}{K}$ *portion of each message and use the protocol of the original PIR scheme [102] to download the uncached portion. What this means is that if the side information is globally known, then there is nothing better than removing the side information from the effective messages. The expression for* $C_{\textit{TPIR-GSI}}$ *reflects the same insight — the role of globally known side information is to reduce the effective number of messages by* M. *The authors of [117] also give a similar explanation for the scheme in [108].*

REMARK 2.4. *Now we can completely order the four variants of PIR with side information, in terms of their capacities as PIR-SI $\geq$ PIR-SPSI $\geq$ PIR-PSI $=$ PIR-GSI. Remarkably, all the inequalities can be strict for certain parameters. For example, as will be shown in Appendix* 2.6, *suppose we have* $K = 6$ *messages stored at* $N = 1$ *database, and* $M = 2$ *of these messages are available to the user as side-information. Then for this example, the capacity of PIR-SI is* $1/2$ *while the capacity of PIR-SPSI is no more than* $1/3$, *so that PIR-SI > PIR-SPSI. Now suppose we have* $K = 6$ *messages stored at* $N = 1$ *database, and* $M = 1$ *of these messages is available to the user as side-information. Then for this example, the capacity of PIR-SPSI is* $1/3$ *while the capacity of PIR-PSI is only* $1/5$, *so that PIR-SPSI > PIR-PSI.*

Our second result is the capacity of STPIR-PSI, presented in the following theorem.

THEOREM 2.2. *For the STPIR-PSI problem with* $K \geq 2$ *messages,* N *databases and* M $(M < K)$ *side information messages, the capacity is*

$$C_{\textit{STPIR-PSI}} = \begin{cases} 1, & \text{if } M = K-1, \\ 1 - \frac{T}{N}, & \text{if } M < K-1 \text{ and } \rho \geq \frac{T}{N-T}, \\ 0, & \text{otherwise}, \end{cases} \tag{2.14}$$

where $\rho = \frac{H(\boldsymbol{U})}{L}$ is the amount of common randomness available to the databases, normalized by the message size.

The following observations are in order.

REMARK 2.5. *When there is only $K = 1$ message, or when there are $M = K-1$ side information messages, the database-privacy constraint is satisfied trivially, so STPIR reduces to the TPIR setting and the capacity is 1. Note that for symmetric PIR without side information, when $K \geq 2$, the common randomness is necessary for feasibility. However, for STPIR-PSI, if there are $M = K-1$ side information messages, then common randomness is not needed.*

REMARK 2.6. *When $K \geq 2$ and $M < K-1$, then $C_{\text{STPIR-PSI}}$ only depends on the number of databases N, the colluding parameter T, and the amount of common randomness. It is independent of the number of messages K and the number of side information messages M.*

REMARK 2.7. *The capacity of STPIR-PSI is strictly smaller than the capacity of TPIR-PSI, which means that the additional requirement of preserving database privacy strictly penalizes the capacity. However, the penalty vanishes in the regime of large number of messages, i.e., $C_{\text{TPIR-PSI}} > C_{\text{STPIR-PSI}}$ for any finite K and $C_{\text{TPIR-PSI}} \to C_{\text{STPIR-PSI}}$ when $K \to \infty$. This observation also holds for the case without side information.*

REMARK 2.8. *$C_{\text{STPIR-PSI}}$ is equal to the capacity of STPIR without side information, which is characterized in [112]. Furthermore, the capacity result remains the same even if the side information is globally known.[4] Thus, utilizing the private or globally known side information does not help improve the capacity.*

2.4 Proof of Theorem 2.1

[4]The explanation is similar to that for TPIR with globally known side information as in Remark 2.

2.4.1 Achievability

The backbone of the achievable scheme for TPIR-PSI with parameters (K, M, N, T) is the achievable scheme of TPIR [105]. We inherit the steps of the query structure construction and query specialization. The novel element of the achievable scheme is query redundancy removal based on the side information. To illustrate how this idea works, we present one toy example with $(K, M, N, T) = (3, 2, 3, 2)$, and then generalize it to arbitrary (K, M, N, T).

Example with $(K, M, N, T) = (3, 2, 3, 2)$

Let us start with the case without side information $(K, M, N, T) = (3, 0, 3, 2)$, i.e., there are 3 messages, 3 databases and any 2 of them can collude. Following the construction of [105], let each message consist of $L = N^K = 27$ symbols from a finite field $\mathbb{F}_q$ that is large enough so that a systematic $(28, 19)$ maximum distance separable (MDS) code exists. The MDS property implies that any 19 out of the 28 codeword symbols is sufficient to recover all 19 information symbols. A systematic code is a code in which the information symbols are embedded in the codeword symbols [77]. According to the query structure construction and query specialization for TPIR [105], the messages $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3 \in \mathbb{F}_q^{27}$ are 27×1 column vectors and let $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \boldsymbol{Y}_3 \in \mathbb{F}_q^{27\times 27}$ represent random matrices chosen privately by the user, independently and uniformly from all 27×27 full-rank matrices over $\mathbb{F}_q$. Let $G_{e\times f}$ denote the generator matrix of an (e, f) MDS code (e.g., a Reed Solomon code), for some integers e, f. The generator matrices need not be systematic or random, and may be globally known. Define the 27×1 column vectors $\boldsymbol{a}_{(1:27)}, \boldsymbol{b}_{(1:27)}, \boldsymbol{c}_{(1:27)} \in \mathbb{F}_q^{27}$ as follows.

$$\boldsymbol{a}_{(1:27)} = \boldsymbol{Y}_1 \boldsymbol{W}_1, \tag{2.15}$$

$$\boldsymbol{b}_{(1:18)} = G_{18\times 12} \boldsymbol{Y}_2[(1:12), :] \boldsymbol{W}_2, \tag{2.16}$$

$$\boldsymbol{c}_{(1:18)} = G_{18\times 12} \boldsymbol{Y}_3[(1:12), :] \boldsymbol{W}_3, \tag{2.17}$$

$$\boldsymbol{b}_{(19:27)} = G_{9\times 6}\boldsymbol{Y}_2[(13:18),:]\boldsymbol{W}_2, \tag{2.18}$$

$$\boldsymbol{c}_{(19:27)} = G_{9\times 6}\boldsymbol{Y}_3[(13:18),:]\boldsymbol{W}_3, \tag{2.19}$$

where $\boldsymbol{Y}_2[(1:18),:]$ and $\boldsymbol{Y}_3[(1:18),:]$ are 18×27 matrices comprised of the first 18 rows of $\boldsymbol{Y}_2$ and $\boldsymbol{Y}_3$, respectively. Note that the same generator matrix $G_{18\times 12}$ is used in (2.16) and (2.17), and the same generator matrix $G_{9\times 6}$ is used in (2.18) and (2.19).

The downloaded symbols from each database are represented in Table 2.1. We use DB$_i$ to represent the i^{th} database. It correctly recovers the queried message and maintains user privacy even if 2 databases collude. The achieved rate is $R_{\text{TPIR}} = 9/19$, namely, in this scheme the user recovers 9 desired symbols from a total of 19 downloads symbols from each database.

Table 2.1: Achievable scheme of TPIR [105]

DB$_1$	DB$_2$	DB$_3$
$\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3, \boldsymbol{a}_4$	$\boldsymbol{a}_5, \boldsymbol{a}_6, \boldsymbol{a}_7, \boldsymbol{a}_8$	$\boldsymbol{a}_9, \boldsymbol{a}_{10}, \boldsymbol{a}_{11}, \boldsymbol{a}_{12}$
$\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_4$	$\boldsymbol{b}_5, \boldsymbol{b}_6, \boldsymbol{b}_7, \boldsymbol{b}_8$	$\boldsymbol{b}_9, \boldsymbol{b}_{10}, \boldsymbol{b}_{11}, \boldsymbol{b}_{12}$
$\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \boldsymbol{c}_4$	$\boldsymbol{c}_5, \boldsymbol{c}_6, \boldsymbol{c}_7, \boldsymbol{c}_8$	$\boldsymbol{c}_9, \boldsymbol{c}_{10}, \boldsymbol{c}_{11}, \boldsymbol{c}_{12}$
$\boldsymbol{a}_{13} + \boldsymbol{b}_{13}$	$\boldsymbol{a}_{15} + \boldsymbol{b}_{15}$	$\boldsymbol{a}_{21} + \boldsymbol{b}_{17}$
$\boldsymbol{a}_{14} + \boldsymbol{b}_{14}$	$\boldsymbol{a}_{16} + \boldsymbol{b}_{16}$	$\boldsymbol{a}_{22} + \boldsymbol{b}_{18}$
$\boldsymbol{a}_{17} + \boldsymbol{c}_{13}$	$\boldsymbol{a}_{19} + \boldsymbol{c}_{15}$	$\boldsymbol{a}_{23} + \boldsymbol{c}_{17}$
$\boldsymbol{a}_{18} + \boldsymbol{c}_{14}$	$\boldsymbol{a}_{20} + \boldsymbol{c}_{16}$	$\boldsymbol{a}_{24} + \boldsymbol{c}_{18}$
$\boldsymbol{b}_{19} + \boldsymbol{c}_{19}$	$\boldsymbol{b}_{21} + \boldsymbol{c}_{21}$	$\boldsymbol{b}_{23} + \boldsymbol{c}_{23}$
$\boldsymbol{b}_{20} + \boldsymbol{c}_{20}$	$\boldsymbol{b}_{22} + \boldsymbol{c}_{22}$	$\boldsymbol{b}_{24} + \boldsymbol{c}_{24}$
$\boldsymbol{a}_{25} + \boldsymbol{b}_{25} + \boldsymbol{c}_{25}$	$\boldsymbol{a}_{26} + \boldsymbol{b}_{26} + \boldsymbol{c}_{26}$	$\boldsymbol{a}_{27} + \boldsymbol{b}_{27} + \boldsymbol{c}_{27}$

Now let us consider the case with side information $(K, M, N, T) = (3, 2, 3, 2)$, i.e., 2 of the messages are known to the user as side information. Assume the user knows $\boldsymbol{W}_2$ and $\boldsymbol{W}_3$ as side information and wishes to retrieve $\boldsymbol{W}_1$. He does not need to download individual symbols of $\boldsymbol{W}_2, \boldsymbol{W}_3$, or the linear combinations comprised of only $\boldsymbol{W}_2, \boldsymbol{W}_3$ symbols, i.e., $\boldsymbol{b}_i, \boldsymbol{c}_i, 1 \le i \le 12$ and $\boldsymbol{b}_j + \boldsymbol{c}_j, 19 \le j \le 24$ in Table 2.1. Therefore, 10 redundant symbols may be reduced from each database. Let us take the step of query redundancy removal. The idea is that the user asks each database to encode the 19 original downloaded symbols with a systematic $(28, 19)$ MDS code and downloads only the 9 linear combinations corresponding to the non-systematic part, called parity symbols. Formally, let $G^s_{e\times f}$ denote the generator matrix of a systematic (e,f) MDS code. The generator matrix does not need to be random, and it may be globally known. For $i=1,2,3$, denote by vector $\boldsymbol{X}_i\in\mathbb{F}_q^{19}$ the symbols downloaded from DB$_i$ after the query structure construction and query specialization (symbols in the DB$_i$ column in Table 2.1). The user asks each database to encode $\boldsymbol{X}_i$ with a systematic $(28,19)$ MDS code generator matrix $G^s_{28\times 19}=[V_{19\times 9}\mid I_{19\times 19}]'$, where $I_{19\times 19}$ is the identity matrix, and downloads only the 9 linear combinations corresponding to the parity part, $V'_{19\times 9}\boldsymbol{X}_i$.

The correctness constraint is satisfied because of the property of MDS code and the correctness of the original TPIR scheme. Given $(\boldsymbol{b}_i)_{i\in[12]}$, $(\boldsymbol{c}_i)_{i\in[12]}$, $(\boldsymbol{b}_i+\boldsymbol{c}_i)_{i\in[19:24]}$, $V'_{19\times 9}\boldsymbol{X}_1$, $V'_{19\times 9}\boldsymbol{X}_2$ and $V'_{19\times 9}\boldsymbol{X}_3$, the user is able to decode $\boldsymbol{X}_1$, $\boldsymbol{X}_2$ and $\boldsymbol{X}_3$, which constitute the original TPIR scheme. The privacy is essentially inherited from the original PIR scheme and the fact that the MDS code is fixed *a priori*, i.e., it does not depend on $(\boldsymbol{\Theta},\boldsymbol{S})$. Thus, the rate achieved with private side information is $R_{\text{TPIR-PSI}}=27/27=1$ which gives a lower bound on the capacity.

Arbitrary (K,M,N,T)

Scheme description. For the sake of completeness, let us briefly introduce the original TPIR achievable scheme in [105]. In this scheme, the message is $L=N^K$ symbols from a large enough finite field $\mathbb{F}_q$, and the normalized total download is $1+\frac{T}{N}+\cdots+(\frac{T}{N})^{K-1}$. It has two key steps: 1) query structure construction and 2) query specialization.

1) *Query Structure Construction:* Construct the query structure. After this step, the query of each database is comprised of K layers. Over the k^{th} layer, the query symbols are in the form of sums of k message symbols, each from one distinct message, called k-sum. There are $\binom{K}{k}$ possible "types" of k-sums and $(N-T)^{k-1}T^{K-k}$ distinct instances[5] of each type of k-sum in k^{th} layer. So, the

[5]The term $(N-T)^{k-1}T^{K-k}$ comes from the undesired message exploitation step (Step 4) of achievability in [105] and can be verified recursively. A detailed analysis of a similar flavor can be found in [102].

total number of elements contained in layer k is $\binom{K}{k}(N-T)^{k-1}T^{K-k}$. Therefore, the total number of symbols to be downloaded from each database is $\sum_{k=1}^{K}\binom{K}{k}(N-T)^{k-1}T^{K-k}$. This structure has two properties: symmetry across databases and message symmetry within the query from each database. Symmetry across databases means that the queries among the databases have the same structure (i.e., the same form of k-sums). Message symmetry implies that within the query of each database, any set of M messages determines the same number of k-sums, $1 \leq k \leq M$.

$2)$ *Query Specialization:* Map the message symbols to the symbols in the query structure. This step is to ensure the correctness and privacy.

Now we are ready to present the achievable scheme for arbitrary (K, M, N, T). First do query structure construction and query specialization without considering the side information, and denote the scheme by Π. Then do query redundancy removal based on the side information. Due to symmetry across databases and message symmetry within the query from each database, regardless of the realization of side information, the number of queried symbols and the number of known symbols (based on the side information) in each query are constants. For each database, let p_1 denote the number of symbols to be downloaded in Π. Out of these p_1 symbols, let p_2 $(p_2 < p_1)$ denote the number of user known symbols. Denote by vector $\boldsymbol{X}_i \in \mathbb{F}_q^{p_1}$ the symbols downloaded from DB$_i$ in Π. For each database, use a systematic $(2p_1-p_2, p_1)$ MDS code with generator matrix $G^s_{(2p_1-p_2)\times p_1} = \left[V_{p_1\times(p_1-p_2)} \mid I_{p_1\times p_1}\right]'$ to encode the p_1 symbols into $2p_1-p_2$ symbols, of which p_1 are systematic, and download only the p_1-p_2 parity symbols, $V'_{p_1\times(p_1-p_2)}\boldsymbol{X}_i$.

Note that the user does not need to know the realization of side information $\boldsymbol{S}$ or $\boldsymbol{W_S}$ in order to construct the queries. This is because the systematic MDS code in the query redundancy removal does not depend on $\boldsymbol{S}$ or $\boldsymbol{W_S}$. During the decoding, $\boldsymbol{S}$ and $\boldsymbol{W_S}$ are only used after the answers from the databases are collected. Therefore, the privacy of this TPIR-PSI scheme is inherited from the privacy of the original TPIR scheme. Correctness follows from the MDS property because in addition to the p_1-p_2 downloaded symbols from DB$_i$, i.e., $V'_{p_1\times(2p_1-p_2)}\boldsymbol{X}_i$, the user provides the p_2 symbols that he already knows, to obtain a total of p_1 symbols from the MDS code. Since any p_1

symbols from an MDS code suffice to recover the original p_1 symbols, the user recovers $\boldsymbol{X}_i$. Then the correctness is inherited from the correctness of the original TPIR scheme. All that remains is to calculate the rate achieved by this scheme.

Rate calculation. Consider the scheme Π, the total downloaded symbols from each database $p_1 = \sum_{k=1}^{K} \binom{K}{k}(N-T)^{k-1}T^{K-k}$. The next step is to calculate, out of these p_1 symbols, how many are already known to the user based on his side information. Suppose the user knows the M messages $\boldsymbol{W}_{\boldsymbol{i}_1}, \cdots, \boldsymbol{W}_{\boldsymbol{i}_M}, \{\boldsymbol{i}_1, \cdots, \boldsymbol{i}_M\} \in [K]$ as side information beforehand. Thus the user knows all linear combinations that are comprised of symbols from these M messages. In terms of layer k $(k \leq M)$, the user knows all the instances of k-sum that contain only symbols $\boldsymbol{W}_{\boldsymbol{j}_1}, \boldsymbol{W}_{\boldsymbol{j}_2}, \cdots, \boldsymbol{W}_{\boldsymbol{j}_k}$, where $\{\boldsymbol{j}_1, \boldsymbol{j}_2, \cdots, \boldsymbol{j}_k\} \subset \{\boldsymbol{i}_1, \cdots, \boldsymbol{i}_M\}$. So the total number of symbols known to the user corresponding to each database is $p_2 = \sum_{k=1}^{M} \binom{M}{k}(N-T)^{k-1}T^{K-k}$. Notice that p_1 can be simplified as,

$$p_1 = \sum_{k=1}^{K}(N-T)^{k-1}T^{K-k}\binom{K}{k} \tag{2.20}$$

$$= \frac{\sum_{k=0}^{K}(N-T)^{k}T^{K-k}\binom{K}{k} - T^K}{N-T} \tag{2.21}$$

$$= \frac{N^K - T^K}{N-T}. \tag{2.22}$$

And p_2 can be simplified as,

$$p_2 = \sum_{k=1}^{M}(N-T)^{k-1}T^{K-k}\binom{M}{k} \tag{2.23}$$

$$= T^{K-M}\sum_{k=1}^{M}(N-T)^{k-1}T^{M-k}\binom{M}{k} \tag{2.24}$$

$$= \frac{T^{K-M}(N^M - T^M)}{N-T}. \tag{2.25}$$

From each database the number of downloaded symbols of desired messages can be calculated as,

$$m = \sum_{k=1}^{K}(N-T)^{k-1}T^{K-k}\binom{K-1}{k-1} = N^{K-1}. \tag{2.26}$$

Therefore, the rate achieved is

$$
\begin{aligned}
R_{\text{TPIR-PSI}} &= \frac{Nm}{N(p_1 - p_2)} && (2.27)\\
&= \frac{N^{K-1}(N-T)}{(N^K - T^K) - T^{K-M}(N^M - T^M)} && (2.28)\\
&= \frac{1 - \frac{T}{N}}{1 - (\frac{T}{N})^{K-M}} && (2.29)\\
&= \left(1 + \frac{T}{N} + \cdots + \left(\frac{T}{N}\right)^{K-M-1}\right)^{-1}. && (2.30)
\end{aligned}
$$

This gives a lower bound on the capacity of TPIR-PSI, thus completing the proof of achievability for Theorem 2.1.

2.4.2 Converse

Let $\mathcal{S}$ be a set whose elements are all possible realizations of $\boldsymbol{S}$, i.e., $\mathcal{S} = \{S \mid S \subset [K], |S|= M\}$. We will need the following lemmas.

Lemma 2.1. *For all $S_1 \in \mathcal{S}$, $\theta \in [K] \setminus S_1$, $S_2 \subseteq [K] \setminus S_1$, and $\mathcal{T} \subset [N], |\mathcal{T}|= T$, given $\boldsymbol{S} = S_1, \boldsymbol{\Theta} = \theta$, $\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \leftrightarrow \left(\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{S_1 \cup S_2}\right) \leftrightarrow \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ is a Markov chain.*

Proof. In this proof, to be convenient, denote $\mathcal{E}_1 = S_1 \cup S_2$ and $\mathcal{E}_2 = [K] \setminus (S_1 \cup S_2)$. It is equivalent to prove

$$I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta} = \theta, \boldsymbol{S} = S_1\right) = 0.$$

By the chain rule of mutual information,

$$
\begin{aligned}
&I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta} = \theta, \boldsymbol{S} = S_1\right)\\
&= I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta} = \theta, \boldsymbol{S} = S_1\right)
\end{aligned}
$$

$$
\begin{aligned}
&\quad + I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&= I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&\quad + I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right).
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&= I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&+ I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&- I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right). \qquad (2.31)
\end{aligned}
$$

Consider the first RHS mutual information term in (2.31),

$$
\begin{aligned}
&I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&= I\left(\boldsymbol{W}_{\mathcal{E}_2}; \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{S_1\cup S_2}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right)\\
&\quad - I\left(\boldsymbol{W}_{[K]\setminus(S_1\cup S_2)}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right) \qquad (2.32)\\
&= 0, \qquad (2.33)
\end{aligned}
$$

where (2.33) holds because of (2.1) and (2.4). The second RHS mutual information term in (2.31) satisfies

$$
I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right) = 0
$$

because of (2.5). At last, the RHS negative mutual information term in (2.31) must also be zero because the LHS mutual information cannot be negative. Thus

$$
I\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}; \boldsymbol{Q}_{[N]\setminus\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\mathcal{E}_1}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S_1\right) = 0.
$$

■

Lemma 2.2. *For all $S \in \mathcal{S}$, $\theta, \theta' \in [K] \setminus S$, and $\mathcal{T} \subset [N], |\mathcal{T}|= T$,*

$$\begin{aligned}
&H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{\Theta}}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) \\
=&H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{\Theta}}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right), \qquad (2.34)\\
&H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) \\
=&H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right). \qquad (2.35)
\end{aligned}$$

Proof. It follows from the user-privacy constraint (2.11) and the non-negativity of mutual information, that for all $S \in \mathcal{S}$, $\mathcal{T} \subset [N], |\mathcal{T}|= T$

$$I\left(\boldsymbol{\Theta}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]} \mid \boldsymbol{S}=S\right)=0, \qquad (2.36)$$

which implies that $\forall \theta, \theta' \in [K] \setminus S$,

$$\begin{aligned}
&H\left(\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}} \mid \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) \\
=&H\left(\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}} \mid \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right), \qquad (2.37)\\
&H\left(\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}} \mid \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) \\
=&H\left(\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}} \mid \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right). \qquad (2.38)
\end{aligned}$$

Subtracting (2.38) from (2.37) yields (2.34). Equation (2.35) is similarly obtained. ■

Before presenting the general converse, let us start with a simple example $(K, M, N, T) = (3, 1, 3, 2)$ for ease of exposition.

Converse for $(K, M, N, T) = (3, 1, 3, 2)$

The total download is bounded as,

$$
\begin{aligned}
D &\geq H(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\mathbf{S}]} \mid \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}, \boldsymbol{S}) && (2.39)\\
&\geq \min_{\substack{S\in\mathcal{S}\\ \theta\in[K]\setminus S}} H(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\mathbf{S}]} \mid \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S). && (2.40)
\end{aligned}
$$

We will derive a lower bound on the entropy in (2.40) that holds for all (θ, S).

For $(K, M, N, T) = (3, 1, 3, 2)$, without loss of generality suppose message $\boldsymbol{W}_1$ is known as side information and $\boldsymbol{W}_2$ is desired. Let $S = \{1\}$. We bound the total download as,

$$
\begin{aligned}
D &\geq H\left(\boldsymbol{A}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_1, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right) && (2.41)\\
&\overset{(2.6)}{=} H\left(\boldsymbol{A}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_2 \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_1, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right) && (2.42)\\
&= H\left(\boldsymbol{W}_2 \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_1, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right)\\
&\quad + H\left(\boldsymbol{A}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right) && (2.43)\\
&\geq L + H\left(\boldsymbol{A}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right) && (2.44)\\
&= L + H\left(\boldsymbol{A}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=2, \boldsymbol{S}=S\right) && (2.45)\\
&= L + H\left(\boldsymbol{A}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) && (2.46)\\
&\geq L + H\left(\boldsymbol{A}_{[2]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) && (2.47)
\end{aligned}
$$

where (2.44) holds because of (2.2), (2.4), the chain rule and non-negativity of entropy. Equation (2.45) holds due to Lemma 2.1. Equation (2.46) holds because of Lemma 2.2. Similarly,

$$
\begin{aligned}
D &\geq L + H\left(\boldsymbol{A}_{\{2,3\}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right), && (2.48)\\
D &\geq L + H\left(\boldsymbol{A}_{\{1,3\}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right). && (2.49)
\end{aligned}
$$

Adding (2.47), (3.39), (2.49) and divided by 3 we have

$$\begin{aligned}
D &\geq L + \frac{1}{3} H\left(\boldsymbol{A}_{\{1,2\}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) \\
&\quad + \frac{1}{3} H\left(\boldsymbol{A}_{\{2,3\}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) \\
&\quad + \frac{1}{3} H\left(\boldsymbol{A}_{\{1,3\}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) && (2.50) \\
&\geq L + \frac{2}{3} H\left(\boldsymbol{A}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[2]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right) && (2.51) \\
&= L + \frac{2}{3} L && (2.52) \\
&= \frac{5}{3} L. && (2.53)
\end{aligned}$$

Here (2.51) follows from Han's inequality, and (2.52) holds because from $\left(\boldsymbol{W}_{[2]}, \boldsymbol{A}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{Q}_{[3]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=3, \boldsymbol{S}=S\right)$ one can recover $\boldsymbol{W}_3$ with vanishing probability of error. Since the same argument holds for all realizations $(\boldsymbol{\Theta}, \boldsymbol{S}) = (\theta, S)$, this gives us the upper bound on the capacity of TPIR-PSI with $(K, M, N, T) = (3, 1, 3, 2)$ as $C_{\text{TPIR-PSI}} \leq \frac{3}{5}$.

Converse for Arbitrary (K, M, N, T)

If $M = K - 1$, it is trivial that 1 is an upper bound, since any rates cannot be larger than 1. So let us assume that $M < K - 1$. For compact notation, let us define

$$D(K, S, \theta, V) \triangleq H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[V]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right).$$

Here $\boldsymbol{W}_{[V]} = (\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_V)$ represents the messages that appear in the conditioning. Also, define an arbitrary $\mathcal{T} \subset [N]$ with cardinality $|\mathcal{T}| = T$ which represents the set of indices of colluding databases.

Without loss of generality, suppose messages $\boldsymbol{W}_1, \cdots, \boldsymbol{W}_M$ are known as side information and

$\boldsymbol{W}_{M+1}$ is desired. Then, we have

$$\begin{aligned}
&D(K,[M],M+1,M)\\
&=H(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}|\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M])\\
&\overset{(2.6)}{=}H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{\boldsymbol{\Theta}}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right)\\
&=H\left(\boldsymbol{W}_{\boldsymbol{\Theta}}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right)\\
&\quad+H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right).
\end{aligned}$$

Note that

$$H\left(\boldsymbol{W}_{\boldsymbol{\Theta}}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right)=L$$

since messages are independent, and queries are independent of the messages. And

$$\begin{aligned}
&H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right)\\
&\geq H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right) &(2.54)\\
&=H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+1,\boldsymbol{S}=[M]\right) &(2.55)\\
&=H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+2,\boldsymbol{S}=[M]\right) &(2.56)\\
&\geq H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+2,\boldsymbol{S}=[M]\right), &(2.57)
\end{aligned}$$

where equation (2.55) holds because of Lemma 2.1. Equation (2.56) holds because of Lemma 2.2. There are a total of $\binom{N}{T}$ such subsets $\mathcal{T}$. Writing (2.57) for all such subsets, adding those inequalities and divided by $\binom{N}{T}$, we obtain

$$\begin{aligned}
&D(K,[M],M+1,M)\\
&\geq\frac{T}{N}H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\mid\boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]},\boldsymbol{W}_{[M+1]},\boldsymbol{\Theta}=M+2,\boldsymbol{S}=[M]\right)\\
&\quad+L &(2.58)
\end{aligned}$$

$$=L+\frac{T}{N}D(K,[M],M+2,M+1), \tag{2.59}$$

where (2.58) follows from Han's inequality. Proceeding along these lines, we have

$$\begin{aligned}
&D(K,[M],M+1,M)\\
&\geq L+\frac{T}{N}D(K,[M],M+2,M+1) &(2.60)\\
&\geq L+\frac{T}{N}\left(L+\frac{T}{N}D(K,[M],M+3,M+2)\right) &(2.61)\\
&\geq\cdots &(2.62)\\
&\geq L+\frac{T}{N}\left(L+\cdots+\frac{T}{N}\left(L+\frac{T}{N}D(K,[M],K,K-1)\right)\right) &(2.63)
\end{aligned}$$

where $D(K,[M],K,K-1)\geq L$. Therefore,

$$\begin{aligned}
&D(K,[M],M+1,M)\\
&\geq L+\frac{T}{N}L+\cdots+\left(\frac{T}{N}\right)^{K-M-1}L &(2.64)\\
&=L\left(1+\frac{T}{N}+\cdots+\left(\frac{T}{N}\right)^{K-M-1}\right). &(2.65)
\end{aligned}$$

The above argument holds similarly for any (θ,S), and hence the upper bound on the rate of TPIR-PSI is

$$R=\lim_{L\to\infty}\frac{L}{D}\leq\left(1+\frac{T}{N}+\left(\frac{T}{N}\right)^2+\cdots+\left(\frac{T}{N}\right)^{K-M-1}\right)^{-1}.$$

Thus, the proof of converse for Theorem 2.1 is complete.

REMARK 2.9. *The converse can also be proved alternatively by a genie-aided approach using the capacity of TPIR-GSI of Remark* 2 *as follows. Starting from the TPIR-PSI problem, suppose we provide the indices of the side information* $\boldsymbol{S}$ *to all the databases, so the side information is now globally known and only the privacy of the desired message needs to be preserved.*

Any schemes for TPIR-PSI are applicable to this TPIR-GSI setting, because they preserve the privacy of the desired message index even after *the side-information is revealed. This is because TPIR-PSI schemes satisfy* $I\left(\mathbf{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_{\mathcal{T}}^{[\mathbf{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\mathbf{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = 0$*, which in turn implies that* $I\left(\mathbf{\Theta}; \boldsymbol{Q}_{\mathcal{T}}^{[\mathbf{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{\mathcal{T}}^{[\mathbf{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]} \mid \boldsymbol{S}\right) = 0$*. Therefore,*

$$
\begin{aligned}
C_{\text{TPIR-PSI}} &\leq C_{\text{TPIR-GSI}} \\
&= \left(1 + \frac{T}{N} + \left(\frac{T}{N}\right)^2 + \cdots + \left(\frac{T}{N}\right)^{K-M-1}\right)^{-1}.
\end{aligned}
$$

2.5 Proof of Theorem 2.2

2.5.1 Achievability

When $M = K - 1$, the user can download the sum of all the messages from one database and get the desired message with side information. The rate is 1, achieving the capacity. Note that in this case, common randomness among databases is not required. When $M < K - 1$, the achievable scheme can directly use the scheme of STPIR [103, 114], and the side information is simply not used.

2.5.2 Converse

When $M = K - 1$, it is obvious that 1 is an upper bound. When $M < K - 1$, we show that $1 - \frac{T}{N}$ is an upper bound.

Proof of the bound $R \leq 1 - T/N$ Let us start with an intuitive understanding of the upper bound, $R \leq 1 - T/N$. Due to database privacy, given the side information, the answers from all

N databases should be independent of the non-queried messages. At the same time, the answers from any T databases should contain no information about the queried message index since the user privacy must be preserved. Combining these two facts, given the side information, the answers from any T databases should contain no information about *any* individual message, whether desired or undesired. As a result, the useful information about the desired message must come from the remaining $N-T$ databases. Thus, the download per database must be at least $1/(N-T)$ times the entropy of the desired message.

The formal proof is as follows. Since $M < K-1$, for any $S \in \mathcal{S}$, there exist at least 2 messages that are not in the set S. Any feasible STPIR-PSI scheme must satisfy the database-privacy constraint (2.12),

$$0 = I\left(\boldsymbol{W}_{\overline{(\boldsymbol{\Theta},\boldsymbol{S})}}; \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{S}, \boldsymbol{\Theta}\right) \tag{2.66}$$

Therefore, $\forall \mathcal{T} \subset [N], |\mathcal{T}|= T, \forall S \in \mathcal{S}$, and for all distinct $\theta, \theta' \in [K] \setminus S$,

$$\begin{aligned}
0 &= I\left(\boldsymbol{W}_{\theta'}; \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.67)\\
&= I\left(\boldsymbol{W}_{\theta'}; \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad + I\left(\boldsymbol{W}_{\theta'}; \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.68)\\
&= H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{W}_{\theta'}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.69)\\
&\overset{(2.34)}{=} H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{W}_{\theta'}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.70)
\end{aligned}$$

where (2.67) holds because $\mathcal{T}$ is a subset of $[N]$ and (2.69) holds due to (2.4). According to the correctness condition,

$$
\begin{aligned}
L &= H\left(\boldsymbol{W}_{\theta'}\right) \\
&\overset{(2.6)}{=} I\left(\boldsymbol{W}_{\theta'}; \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.71) \\
&= H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta'}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.72) \\
&\leq H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta'}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.73) \\
&= H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta'}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.74) \\
&\overset{(2.70)}{=} H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.75) \\
&\overset{(2.35)}{=} H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.76) \\
&\leq H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right), && (2.77)
\end{aligned}
$$

where (2.74) follows due to Lemma 2.1. Writing (2.77) for all $\mathcal{T} \subset [1:N], |\mathcal{T}|= T$, adding those inequalities and divided by $\binom{N}{T}$ we obtain,

$$
\begin{aligned}
L &\leq H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - \frac{1}{\binom{N}{T}} \sum_{\mathcal{T}} H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.78) \\
&\leq H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) \\
&\quad - \frac{T}{N} H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) && (2.79)
\end{aligned}
$$

$$= \left(1 - \frac{T}{N}\right) H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta} = \theta', \boldsymbol{S} = S\right) \tag{2.80}$$

where (2.79) is due to Han's inequality. Since this inequality is true for all $S \in \mathcal{S}, \theta' \in [K] \setminus S$, it is also true when averaged across them, so,

$$L \leq \left(1 - \frac{T}{N}\right) H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}, \boldsymbol{S}\right) \tag{2.81}$$

$$\leq \left(1 - \frac{T}{N}\right) H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}\right) \tag{2.82}$$

$$\leq \left(1 - \frac{T}{N}\right) D, \tag{2.83}$$

where (2.82) holds because dropping conditioning does not reduce entropy. Therefore, $R = \lim_{L\to\infty} \frac{L}{D} \leq 1 - \frac{T}{N}$, and we have shown that the rate of any feasible STPIR-SI scheme cannot be more than $1 - \frac{T}{N}$.

Proof of the bound $\rho \geq T/(N-T)$ Let us first explain the intuition behind this bound on the size of the common randomness $\boldsymbol{U}$ that should be available to all databases but not to the user. We have already shown that the normalized size of the answer from any individual database must be at least $L/(N-T)$. Due to the user and database privacy constraints, the answers from any T databases are independent of the messages. Therefore, to ensure database privacy, the amount of common randomness must be no smaller than the size of the answers from T databases.

The formal proof is as follows. Suppose a feasible STPIR-PSI scheme exists that achieves a non-zero rate. Then we show that it must satisfy $\rho \geq T/(N-T)$. For $\boldsymbol{S} = S \in \mathcal{S}$ and for $\Theta = \theta \in [K] \setminus S$, consider the answering strings $\boldsymbol{A}_1^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \cdots, \boldsymbol{A}_N^{[\boldsymbol{\Theta},\boldsymbol{S}]}$ and the side information $\boldsymbol{W}_{\boldsymbol{S}}$, from which the user can retrieve $\boldsymbol{W}_\theta$. According to the database-privacy constraint, we have

$$0 = I\left(\boldsymbol{W}_{\overline{(\theta,S)}}; \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta} = \theta, \boldsymbol{S} = S\right)$$

$$\overset{(2.6)}{=} I\left(\boldsymbol{W}_{\overline{(\theta,S)}}; \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_\theta \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta} = \theta, \boldsymbol{S} = S\right)$$

$$
\begin{aligned}
&\overset{(2.9)}{=} I\left(\boldsymbol{W}_{\overline{(\theta,S)}}\,; \boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\geq I\left(\boldsymbol{W}_{\overline{(\theta,S)}}\,; \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&= H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{[K]}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\overset{(2.10)}{=} H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad - H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{[K]}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad + H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{[K]}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{U}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&= H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\quad - I\left(\boldsymbol{U}; \boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{[K]}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right)\\
&\geq H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\theta}, \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) - H(\boldsymbol{U})\\
&\overset{(2.70)}{=} H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta', \boldsymbol{S}=S\right) - H(\boldsymbol{U})\\
&\overset{(2.35)}{=} H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) - H(\boldsymbol{U}).
\end{aligned}
$$

Therefore,

$$H(\boldsymbol{U}) \geq H\left(\boldsymbol{A}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right). \tag{2.84}$$

Adding (2.84) for all $\mathcal{T} \subset [N], |\mathcal{T}|= T$ and divided by $\binom{N}{T}$, we obtain,

$$
\begin{aligned}
H(\boldsymbol{U}) &\geq \frac{T}{N} H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{\mathcal{T}}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.85)\\
&\geq \frac{T}{N} H\left(\boldsymbol{A}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]} \mid \boldsymbol{W}_{\boldsymbol{S}}, \boldsymbol{Q}_{[N]}^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{\Theta}=\theta, \boldsymbol{S}=S\right) && (2.86)\\
&\overset{(2.80)}{\geq} \frac{T}{N}\left(\frac{N}{N-T}\right) L = \left(\frac{T}{N-T}\right) L. && (2.87)
\end{aligned}
$$

$$\Rightarrow \quad \rho = \frac{H(\boldsymbol{U})}{L} \geq \frac{T}{N-T} \quad \text{(letting } L \to \infty\text{)}. \tag{2.88}$$

Note that (2.85) is due to Han's inequality. Thus the amount of common randomness normalized by the message size for any feasible STPIR-PSI scheme cannot be less than $T/(N-T)$.

2.6 Some Insights on the Capacity of PIR-SPSI

The four variants of PIR with side information are defined as follows.

- **PIR-SI**, or PIR with (non-private) side information. Only the privacy of the desired message is preserved, i.e., $I\left(\boldsymbol{\Theta}; \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = 0, \forall n \in [N]$.
- **PIR-SPSI**, or PIR with separately private side information. The privacy of the desired message and the privacy of the side information are preserved individually, i.e., $I\left(\boldsymbol{\Theta}; \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = I\left(\boldsymbol{S}; \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = 0, \forall n \in [N]$.
- **PIR-PSI**, or PIR with jointly private side information. The privacy of the desired message and the privacy of the side information are preserved jointly, i.e., $I\left(\boldsymbol{\Theta}, \boldsymbol{S}; \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]}\right) = 0, \forall n \in [N]$.
- **PIR-GSI**, or PIR with global side information. The side information is globally known, i.e., the databases are also fully knowledgeable about the side information. In this case, the privacy of the desired message index must be preserved in spite of the globally known side information, $I\left(\boldsymbol{\Theta}; \boldsymbol{Q}_n^{[\boldsymbol{\Theta},\boldsymbol{S}]}, \boldsymbol{W}_{[K]} \mid \boldsymbol{S}\right) = 0, \forall n \in [N]$.

From the result of Theorem 2.1 we know the capacity of PIR-PSI is $\Psi(1/N, K-M)$, and from Remark 2 that follows Theorem 2.1 we also know the capacity of PIR-GSI is $\Psi(1/N, K-M)$. The capacity of PIR-SI is known to be $\lceil \frac{K}{M+1} \rceil^{-1}$ for $N=1$ database from [61]. In spite of various attempts the capacity of PIR-SI remains in general an open problem for multiple databases. The remaining setting of PIR-SPSI has not been studied, perhaps due to lack of practical motivation for this setting. Nevertheless, out of technical curiosity, let us present some insights into the capacity of PIR-SPSI. We will focus only on the single database setting, i.e., $N = 1$ in this section.

2.6.1 PIR-SPSI: $N = 1$, $M = 1$, K even

For this setting the capacity of PIR-SPSI is $\left(\frac{K}{2}\right)^{-1} = \lceil\frac{K}{2}\rceil^{-1}$, i.e., the same as the capacity of PIR-SI. Since PIR-SPSI is a more constrained version of PIR-SI, its capacity cannot be higher than that of PIR-SI. Thus, the converse is trivial. It turns out that the achievability is also straightforward because the Partition and Code scheme in [61] already preserves the separate privacy of side information. Let us present just an example to illustrate this. Suppose $N = 1, M = 1, K = 6$, and suppose each message is comprised of one bit. Let θ denote the desired message index and s denote the index of the message available as side information to the user. The user asks the database for three bits, corresponding to the three partitions: $P_1 = W_{i_1} + W_{i_2}, P_2 = W_{i_3} + W_{i_4}, P_3 = W_{i_5} + W_{i_6}$. The indices $(i_1, i_2, \cdots, i_6)$ are obtained by first randomly permuting $(1, 2, \cdots, 6)$ and then switching the position of the side information index s with another index (if needed) so that it appears within the same partition as θ, i.e., one of the partitions must contain $W_\theta + W_s$. The scheme is correct because the user can recover W_θ from the sum $W_\theta + W_s$ (because W_s is already available to the user as side information). It is easily verified that θ and s are each uniformly distributed over $(i_1, i_2, \cdots, i_6)$, so the scheme preserves their separate privacy. However, since θ, s must appear in the same partition, it is also clear that their *joint* privacy is not preserved. For example, (θ, s) cannot be equal to (i_1, i_3). The general scheme in [61] works for any even K by partitioning the messages into sets of size 2, one of which contains both θ and s. Each of θ and s is uniformly distributed over the indices but they are not jointly uniform.

2.6.2 PIR-SPSI: $N = 1$, $M = 1$, K odd

For this setting also the capacity of PIR-SPSI is $\left(\frac{K+1}{2}\right)^{-1} = \lceil\frac{K}{2}\rceil^{-1}$, the same as the capacity of PIR-SI. Once again, the converse is trivially inherited from PIR-SI. Achievability requires a small modification to the Partition and Code scheme of [61], as explained next. Let us also illustrate this through an example. Suppose $N = 1, M = 1, K = 7$ and each message is comprised of one symbol from, say $\mathbb{F}_5$. The user asks the database for 4 symbols, corresponding to $P_1 = W_{i_1} + W_{i_2}$, $P_2 = W_{i_3} + W_{i_4}$, $P_3 = W_{i_5} + W_{i_6} + W_{i_7}$, and $P_4 = W_{i_5} + 2W_{i_6} + 3W_{i_7}$. In fact, P_3, P_4 can be the non-systematic symbols of any $(5, 3)$ systematic MDS code applied to $W_{i_5}, W_{i_6}, W_{i_7}$. Once again, the indices $(i_1, i_2, \cdots, i_7)$ are obtained by first randomly permuting $(1, 2, \cdots, 7)$ and then switching the position of the side information index s with another index (if needed) so that it appears within the same partition as θ. If W_θ and W_s appear in P_1 or P_2 then W_θ is decoded by subtracting the side-information, while if W_θ and W_s appear in partitions P_3, P_4 with interfering message W_i, then after eliminating the known side information W_s, the two equations can be solved for the remaining two variables W_θ, W_i (equivalently, the MDS property guarantees decodability). Once again, it is easily verified that θ and s are each uniformly distributed over $(i_1, i_2, \cdots, i_7)$, so the scheme preserves their separate privacy. However, since θ, s must appear in the same partition, it is also clear that their *joint* privacy is not preserved. The example generalizes to any odd value of K, by constructing $(K + 1)/2$ partitions of the form $W_{i_1} + W_{i_2}$, $W_{i_3} + W_{i_4}$, $\cdots$, $W_{i_{K-4}} + W_{i_{K-3}}$, $W_{i_{K-2}} + W_{i_{K-1}} + W_{i_K}$ and $W_{i_{K-2}} + 2W_{i_{K-1}} + 3W_{i_K}$, and generating the indices $(i_1, i_2, \cdots, i_K)$ by first randomly permuting $(1, 2, \cdots, K)$ and then switching the position of the side information index s with another index (if needed) so that it appears within the same partition as θ. This ensures that θ and s are each uniformly distributed over $(i_1, i_2, \cdots, i_K)$, so the scheme preserves their separate privacy. However, since θ, s must appear in the same partition, it is also clear that their *joint* privacy is not preserved.

2.6.3 PIR-SPSI: $N = 1$, $M = 2$, $K = 6$

The preceding discussion shows that PIR-SI and PIR-SPSI have the same capacity for $N = 1, M = 1$. Let us now present an example to show that the capacity of PIR-SPSI can be strictly less than the capacity of PIR-SI in general. For this example, let us consider $K = 6$ messages stored at $N = 1$ database, out of which $M = 2$ messages are available to the user as side information. From [61] we know that the capacity of PIR-SI for this example is $1/2$. Incidentally, this is achieved by downloading two partitions, namely $W_{i_1} + W_{i_2} + W_{i_3}$ and $W_{i_4} + W_{i_5} + W_{i_6}$, where the indices $(i_1, i_2, \cdots, i_6)$ are generated by first randomly permuting $(1, 2, \cdots, 6)$ and then switching indices if necessary to place the two side information indices into the same partition as θ. Note that this scheme does not preserve the privacy of side information indices, e.g., (i_1, i_4) cannot be both side information indices (because side information indices must be within the same partition). We will show that for this example the capacity of PIR-SPSI is no more than $1/3$, i.e., strictly smaller than the capacity of PIR-SI.

Let us denote the entropy of each message as L bits. We will show that conditioned on each realization of the query, the download from the database must be at least $3L$ bits, which also proves that the average download must be at least $3L$ bits. To set up a proof by contradiction, let us assume that conditioned on the query realization $\mathbf{Q} = q$, the download $\mathbf{A}$ from the database is less than $3L$ bits. This assumption implies that,

$$H(\mathbf{A} \mid \mathbf{Q} = q) < 3L. \tag{2.89}$$

The conditioning on $\mathbf{Q} = q$ will be assumed throughout the remainder of this proof.

We need some preliminary work before we start the core of the converse proof. To have compact notation, for any subset $P \subset [K]$, let us define

$$H_{\mathbf{A}}(W_P) \triangleq H\left(\mathbf{A} \mid \mathbf{Q} = q, W_{[K]\setminus P}\right). \tag{2.90}$$

Intuitively, $H_{\mathbf{A}}(W_P)$ represents the entropy that remains in the answer $\mathbf{A}$ due to messages W_P (after all other messages are known), i.e., the 'space' occupied by the messages W_P in $\mathbf{A}$. We need the following facts.

LEMMA 2.3. *The following facts hold for PIR-SPSI with $N = 1, M = 2, K = 6$.*

1. *If P is a singleton set, e.g., $P = \{k\}$, then we must have*

$$H_{\mathbf{A}}(W_k) \geq L,\ \forall k \in [K]. \tag{2.91}$$

2. *If $P_1 \subset P_2 \subset [K]$, then*

$$H_{\mathbf{A}}(P_1) \leq H_{\mathbf{A}}(P_2). \tag{2.92}$$

3. *If $\mathbf{\Theta} = \theta$ is the desired message index, $\mathbf{S} = (s_1, s_2)$ are the $M = 2$ side information indices, and l, m, n are the 3 remaining indices representing* interfering *messages, then we must have,*

$$H_{\mathbf{A}}(W_l, W_m, W_n) < 2L, \tag{2.93}$$

$$H_{\mathbf{A}}(W_\theta, W_i) \geq 2L,\ \forall i \in \{l, m, n\}. \tag{2.94}$$

Proof. The first fact, (2.91) holds because given the answer $\mathbf{A}$ and all messages except W_k (which must include the side information), the user must be able to decode W_k, therefore $L = I(W_k; \mathbf{A} \mid \mathbf{Q} = q, W_{[K]\setminus\{k\}}) \leq H_{\mathbf{A}}(W_k)$. The next fact, (2.92) is simply the statement that conditioning reduces entropy. The third fact, (2.93) is quite intuitive, as it says that the space occupied by interference must be less than $2L$ bits because the overall download is less than $3L$ bits, out of which L bits are needed for the desired message. Formally, this can be seen as follows.

$$L = I(W_\theta; \mathbf{A} \mid \mathbf{Q} = q, W_{s_1}, W_{s_2}) \tag{2.95}$$

$$= H(\mathbf{A} \mid \mathbf{Q} = q, W_{s_1}, W_{s_2}) - H(\mathbf{A} \mid \mathbf{Q} = q, W_{s_1}, W_{s_2}, W_\theta) \tag{2.96}$$

$$\leq H(\mathbf{A} \mid \mathbf{Q} = q) - H_{\mathbf{A}}(W_l, W_m, W_n) \tag{2.97}$$

$$< 3L - H_{\mathbf{A}}(W_l, W_m, W_n) \tag{2.98}$$

which implies (2.93). Finally, the last fact, (2.94) is also quite intuitive. It says that the desired information must not align with interference so that the user is able to resolve the two. Formally, for any $i \in \{l, m, n\}$, because the user must be able to decode his desired message from $\mathbf{A}$ and his side information,

$$L = I(W_\theta; \mathbf{A} \mid \mathbf{Q} = q, W_{[K]\setminus\{\theta,i\}}) \tag{2.99}$$

$$= H_{\mathbf{A}}(W_\theta, W_i) - H_{\mathbf{A}}(W_i) \tag{2.100}$$

$$\leq H_{\mathbf{A}}(W_\theta, W_i) - L \tag{2.101}$$

which implies (2.94). Note that we used (2.91) to obtain (2.101). ■

With these preliminary facts established, let us now proceed with the core of the converse argument. Since the query preserves the privacy of the side information, all choices of (s_1, s_2) must be equally likely. In particular they must all be feasible (have non-zero probability) from the database's perspective. Note that because the database knows $\mathbf{Q} = q$, it can evaluate $H(W_P)$ for all $P \subset [K]$. Let (a, b, c, d, e, f) represent some permutation of $(1, 2, \cdots, 6)$. The main reasoning now proceeds through the following steps.

1. Consider $(s_1, s_2) = (a, b)$. Since this must be feasible, there must exist another index in $[K]$ that could be a desired message, i.e., that satisfies facts (2.93), (2.94). Without loss of generality, let c be this index, so that,

$$H_{\mathbf{A}}(W_d, W_e, W_f) < 2L, \tag{2.102}$$

$$H_{\mathbf{A}}(W_c, W_i) \geq 2L, \ \forall i \in \{d, e, f\}. \tag{2.103}$$

2. Now consider $(s_1, s_2) = (b, c)$. This must also be feasible, so there must exist an index in $[K]$ which can be a desired message. Based on (2.102), and the fact (2.94) the database can conclude that this index must be a. This is because all other indices lead to contradictions. For example, if the desired message is W_d, then from (2.94) we must have $H_{\mathbf{A}}(W_d, W_e) \geq 2L$, which contradicts the fact that $H_{\mathbf{A}}(W_d, W_e) \leq H_{\mathbf{A}}(W_d, W_e, W_f) < 2L$ according to (2.92) and (2.102). Similarly, the desired message index cannot be e or f either, leaving a as the only possibility. Now (2.94) implies that we must have

$$H_{\mathbf{A}}(W_a, W_i) \geq 2L, \ \forall i \in \{d, e, f\}. \tag{2.104}$$

3. Next, consider $(s_1, s_2) = (e, f)$. This must also be feasible, so there must exist an index in $[K]$ which can be a desired message. Based on (2.103), (2.104) and the fact (2.93) the database can conclude that this index must be d. This is because all other indices lead to contradictions. For example, if the desired message is a, then from (2.93) we must have $H_{\mathbf{A}}(W_b, W_c, W_d) < 2L$. Along with (2.92) this implies that $H_{\mathbf{A}}(W_c, W_d) < 2L$ which contradicts (2.103). Similarly, the desired message index cannot be b or c either, leaving d as the only possibility. Now (2.93) implies that we must have

$$H_{\mathbf{A}}(W_a, W_b, W_c) < 2L. \tag{2.105}$$

4. Finally, consider $(s_1, s_2) = (a, d)$. This must also be feasible, so there must exist an index in $[K]$ which can be a desired message. However, it turns out that every choice of this desired message index leads to a contradiction. For example, suppose the desired message index is b. Then according to (2.94) we must have $H_{\mathbf{A}}(W_b, W_c) \geq 2L$, which contradicts with the combination of (2.105) and (2.92). All other indices are similarly ruled out, leaving us with an unavoidable contradiction.

The contradiction proves that the download must be at least $3L$ bits, which in turn implies that the average download must be at least $3L$ bits, and therefore the capacity cannot be more than $1/3$. The exact capacity even for this simple setting remains an intriguing open problem. Remarkably, if the capacity is less than $1/3$ then that would imply that having more side-information is counter-productive for PIR-SPSI (because if M is reduced from 2 to 1 then we do know from the preceding discussion in this section that the capacity of PIR-SPSI is $1/3$).

2.7 Conclusion

In this chapter, the capacity of TPIR-PSI and the capacity of STPIR-PSI are characterized. Some insights on the capacity of PIR-SPSI are discussed. As a special case of TPIR-PSI obtained by setting $T = 1$, the result settles the capacity of PIR-PSI, an open problem highlighted by Kadhe et al. in [61]. Notably, the results of our work (initially limited to capacity of PIR-PSI for $T = 1$ as reported in our original ArXiv posting in 2017 [19]) have subsequently been generalized to multi-message PIR-PSI in [97]. Other generalizations, e.g., PIR-PSI with multi-round communication, secure and/or coded storage, remain promising directions for future work, as are the capacity characterizations for PIR-SI (multiple databases) and PIR-SPSI which remain open.

Chapter 3

The Asymptotic Capacity of Private Search

3.1 Introduction

Search is among the most frequent operations performed on large online datasets. With privacy concerns increasingly taking center stage in online interactions, a private search functionality is highly desirable. As a basic formulation of the information-theoretically private search problem, consider a *dataset* that is replicated across N non-colluding servers. There are L i.i.d. records in the dataset, each record is comprised of P symbols, and each symbol is from an alphabet of size K. A basic form of private search, called *exact* private search, allows a user to privately choose one symbol from the alphabet, and then search for all records that contain this symbol, without revealing any information about the queried symbol. Suppose the record length P is a constant, and $L \gg K \gg 1$, i.e., the alphabet size K is large, but the number of records in the dataset is much larger. This is not an uncommon scenario. For example, consider datasets of DNA sequences. When searching for a DNA pattern of length ℓ (e.g., $\ell = 10$), the alphabet size is $K = 4^\ell$, while current DNA sequencing machines produce millions of records (called reads) per run. Since the upload cost of private search can be made independent of L while the download cost scales linearly with L, the communication cost of private search for large L is dominated by the download cost. The *capacity* of private search is therefore defined as the maximum number of bits of desired information that can be retrieved per bit of download. Furthermore, since $K \gg 1$, the *asymptotic capacity* of private search, i.e., the capacity for large K is of particular interest. Characterizing the asymptotic capacity of private search is our main goal in this chapter.

Private search (PS) has been studied in computer science for decades. One branch focuses on designing searchable encryption schemes, which enable users to store encrypted data at the servers and execute search over ciphertext domain [15, 42]. Encryption preserves the user privacy computationally. Various models of search functionality have been explored in this framework, such as keyword search [15, 42], similarity search [69, 72], OR and AND search [43, 63] and ranked search [111]. Another branch allows servers to store unencrypted data, and relies on private information retrieval (PIR) [23] schemes to guarantee the privacy of the user's query. Problems investigated in this framework include keyword search [21], streaming data search [10, 37, 88], and media search [34, 35]. Our work is along the latter line and tries to characterize the asymptotic capacity of private search. Recall that in its original form as introduced by Chor et al. in [23], the goal of PIR is to allow a user to retrieve an arbitrary desired message out of μ *independent* messages that are replicated across N distributed and non-colluding servers, without revealing any information about the identity of the desired message to any individual server. The capacity of PIR is the maximum number of bits of desired information that can be retrieved per bit of download, and was shown in [102] to be $\left(1 + \frac{1}{N} + \cdots + \frac{1}{N^{\mu-1}}\right)^{-1}$. The capacity of many variants of PIR has since been characterized, such as PIR with colluding servers [105], PIR with coded servers [7, 107, 110], symmetric PIR [103, 114], PIR with side information [20, 60, 109, 118] and multi-message PIR [6].

Particularly relevant to this paper is the generalized form of PIR introduced in [80, 106], known as the private computation problem [106] or the private function retrieval problem [80, 85, 86]. As its main result, [106] establishes the capacity of PIR when the messages have arbitrary linear dependencies. A supplementary result of [106] shows that even if non-linear dependencies are allowed, the asymptotic capacity of private computation approaches $1 - 1/N$ provided that the message set includes an unbounded number of independent messages. Some other types of private computations are also investigated, i.e., private polynomial computation [92] which allows polynomial relationships among messages, and private inner product retrieval [81] which considers the inner product of messages. Private search is a form of PIR with a specific form of dependency among messages, that is not covered by these prior works. This is because in private search the dependencies among messages are neither linear nor of a polynomial form, and no two messages are independent. To see this clearly, consider exact search with alphabet set $\{A, B, C\}$ (which implies $K = 3$). Assume there are $L = 4$ records, A, A, B, C, each of size $P = 1$. We search for all records that match a queried symbol. Denote the retrieved message for a query by W_θ, for some $\theta \in [3]$, which is comprised of 4 i.i.d. bits, i.e., $W_\theta = (W_\theta(1), W_\theta(2), W_\theta(3), W_\theta(4))$, such that $W_\theta(l) = 1$ if the l-th record matches the queried symbol, and $W_\theta(l) = 0$ otherwise. For example, if A is queried, the corresponding message $W_1 = (1, 1, 0, 0)$. If B is queried, the corresponding message $W_2 = (0, 0, 1, 0)$. If C is queried, the corresponding message $W_3 = (0, 0, 0, 1)$. It is easily seen that any two messages, W_i, W_j, $i \neq j$, are identically distributed but not independent, e.g., $W_i(l) = 1$ implies $W_j(l) = 0$. This dependency is neither linear nor in a polynomial form. To approach the private search problem, we first consider a broader generalization of PIR to include messages with arbitrary dependencies (DPIR in short). Then we consider private search as a special case of DPIR.

Since simple keyword search often yields far too coarse results, almost all the search engines such as Google, Bing, Yahoo, Linkedin and Facebook, and large database management systems like MySQL and PostgreSQL support OR search, AND search and NOT search. These searches allow a broader range of search operations by connecting various pieces of information with OR, AND or NOT operators to make the search more precise. For example, instead of retrieving all emails from “Alice”, a user might only want those emails from Alice that are marked urgent and pertain to finance, in which case what is needed is the ability to search on the conjunction of the keywords, “Alice” and “urgent” and ”finance” [43]. In other cases, it is desirable to search for symbols that appear in consecutive positions in a record, e.g., to search for a phrase. Therefore as natural generalizations of exact private search, we also consider OR private search, AND private search, NOT private search and sequence private search. OR private search looks for all records which contain any of M symbols, AND private search looks for all records which contain all of M symbols, and NOT search looks for all records which do not contain the chosen symbol. Finally, sequence private search allows the user to search for all records that contain an M-symbol long sequence.

Our main contributions are as follows.

- We start with a general non-asymptotic converse for dependent private information retrieval or DPIR (Theorem 3.1). Converse here denotes a lower bound on the download cost, or equivalently, an upper bound on the capacity.
- The converse combined with a general achievability result for DPIR that was established in [106], leads us to a sufficient condition under which the asymptotic capacity of DPIR is characterized to be $1 - 1/N$ (Theorem 3.3).
- The sufficient condition of Theorem 3.3 is shown to hold for exact private search, thus establishing the asymptotic capacity of private search as $1 - 1/N$ (Theorem 3.4).
- We show that the sufficient condition of Theorem 3.3 also holds for OR search, AND search, NOT search and sequence search, so that the asymptotic capacity for these generalizations is also equal to $1 - 1/N$ (Theorem 3.4). Remarkably, for OR search, the asymptotic capacity characterization holds even when M itself grows with K.
- Finally, to illustrate the difficulty of finding general asymptotic capacity results for DPIR, we consider an example of OR private search with special restricted search patterns. For this example, we show that either the new converse bound is not tight, or the asymptotic capacity is not $1 - 1/N$ (Proposition 3.1). The asymptotic capacity for this example remains open.

The paper is organized as follows. Section 3.2 presents the problem statement. The download lower bound of DPIR and the asymptotic capacity of various forms of private search are characterized in Section 3.3. Section 3.4 presents the proofs of the results. Section 3.5 concludes the paper with a discussion of generalization of our settings, including extending Theorem 3.3 to T-colluding DPIR and MDS-coded DPIR, and the non-asymptotic capacity of private search.

Notation: We use parentheses $(a_1, a_2, \ldots, a_n)$ to represent a vector or a tuple (sequence) and braces $\{s_1, s_2, \ldots, s_N\}$ to represent a set. $[z_1 : z_2]$ represents the set $\{z_1, z_1 + 1, \cdots, z_2\}$, for $z_1, z_2 \in \mathbb{N}$, $z_1 < z_2$, $[z]$ represents $[1 : z]$ for $z \in \mathbb{N}$. Let $W_1, W_2, \ldots$ be random variables, and $S = \{s_1, s_2, \ldots, s_N\}$ be a subset of indices where $s_1 < s_2 < \cdots < s_N$. The random vector $(W_{s_1}, W_{s_2}, \ldots, W_{s_N})$ is represented by W_S. $A \sim B$ means that random vectors A and B are identically distributed. A function $f(L) = o(L)$ means that $\lim_{L\to\infty} f(L)/L = 0$. $o(L)$ can be positive or negative. A function $f(L) = O(L)$ means that $\lim_{L\to\infty} |f(L)|/L \leq c$, for some constant $c > 0$.

3.2 Problem Statement

3.2.1 Dependent private information retrieval (DPIR)

Consider $\mu \in \mathbb{N}$ messages, $W_m, m \in [\mu]$, each comprised of L symbols, $W_m = (W_m(1), W_m(2), \cdots, W_m(L))$. The random vectors $(W_1(l), W_2(l), \cdots, W_\mu(l))$, for all $l \in [L]$, are i.i.d., and have a distribution that is identical to the distribution of the random vector $(w_1, w_2, \cdots, w_\mu)$. Namely, for different l, the vectors $(W_1(l), W_2(l), \cdots, W_\mu(l))$ are independent; but for any particular l, the variables $W_m(l), m \in [\mu]$ have dependencies defined by the joint distribution of $w_m, m \in [\mu]$.

EXAMPLE 3.1. *For $L = 2, \mu = 3$, let (w_1, w_2, w_3) be a binary random vector and the distribution* *be* $p(w_1, w_2, w_3) = 1/3$ *for* $(w_1, w_2, w_3) \in \{(1,0,0), (0,1,0), (0,0,1)\}$*, and* $p(w_1, w_2, w_3) = 0$ *for all other cases. One realization of the messages can be:*

W_1	*1*	*0*
W_2	*0*	*0*
W_3	*0*	*1*

The first bit (column) and the second bit (column) of the messages are independent. Within the first bit (column), only one entry can be 1.

The amount of information carried by the m-th message, $m \in [\mu]$, is

$$H(W_m) = LH(w_m). \tag{3.1}$$

We say that the DPIR problem is *balanced* if all messages $W_m, \forall m \in [\mu]$ carry the same amount of information,

$$H(W_1) = H(W_2) = \cdots = H(W_\mu) \triangleq LH(w), \tag{3.2}$$

i.e. $\forall m \in [\mu], H(w_m) = H(w)$.

We note here that the random variable w may depend on the number of messages μ, especially in the context of private search, i.e., $H(w) = H(w(\mu))$. For compact notation, we will not explicitly show the dependence on μ.

The problem of DPIR is as follows. There are N servers and each server stores all μ messages. A user privately generates $\theta \in [\mu]$ and wishes to retrieve W_θ while keeping θ private from each server. Depending on θ, the user employs N queries $Q_1^{[\theta]}, \cdots, Q_N^{[\theta]}$ and sends $Q_n^{[\theta]}$ to the n-th server.

The n-th server returns a response string $A_n^{[\theta]}$ which is a function of $Q_n^{[\theta]}$ and $W_{[\mu]}$, i.e.,

$$\forall \theta \in [\mu], \forall n \in [N],\ H(A_n^{[\theta]} \mid Q_n^{[\theta]}, W_{[\mu]}) = 0. \tag{3.3}$$

From all the information that is now available to the user, he must be able to decode the desired message W_θ, with probability of error $P_e \to 0$ as $L \to \infty$. This is called the "correctness" constraint. From Fano's inequality, we have

$$\text{[Correctness] } H\left(W_\theta \mid A_{[N]}^{[\theta]}, Q_{[N]}^{[\theta]}\right) = o(L). \tag{3.4}$$

To protect the user's privacy, θ must be indistinguishable from θ', from the perspective of each server, $\forall \theta, \theta' \in [\mu]$, i.e.,

$$\text{[Privacy] } (Q_n^{[\theta]}, A_n^{[\theta]}, W_{[\mu]}) \sim (Q_n^{[\theta']}, A_n^{[\theta']}, W_{[\mu]}). \tag{3.5}$$

The DPIR *rate* characterizes how many bits of desired information are retrieved per downloaded bit, and is limited by the worst case as,

$$R \triangleq \frac{\min_{m\in[\mu]} LH(w_m)}{D}, \tag{3.6}$$

where D is the expected total number of bits downloaded by the user from all the servers. If the DPIR problem is balanced, then the minimum over m may be ignored. The supremum of achievable rates R is the *capacity* $C_{\textit{DPIR}}(\mu, N)$.

3.2.2 Private search

We first define exact search and OR search. Later we define AND search, NOT search and sequence search. Examples of different kinds of search are given in Table 3.1.

Exact Search and OR Search

Consider a dataset Δ comprised of L i.i.d. records: $\Delta = (\Delta_1, \Delta_2, \cdots, \Delta_L)$. Each record Δ_l, $l \in [L]$, is a sequence of length P, where P is constant, denoted by $(\delta_{l1}, \delta_{l2}, \cdots, \delta_{lP})$, and each symbol δ_{li} takes values uniformly and independently from the alphabet set $\mathcal{U} = \{U_1, U_2, \cdots, U_K\}$. The dataset is replicated across N non-colluding servers.

For all $l \in [L]$, $\delta_{l_i} \in \mathcal{U}$, $i \in [P]$,

$$P(\Delta_l = (\delta_{l1}, \delta_{l2}, \cdots, \delta_{lP})) = \frac{1}{K^P}, \tag{3.7}$$

$$H(\Delta) = LH(\Delta_l) = L \log_2(K^P) = LP \log_2 K \text{ bits.} \tag{3.8}$$

A user privately chooses a set (search pattern), $S = \{U_{\theta_1}, U_{\theta_2}, \cdots, U_{\theta_M}\}$, $S \subset \mathcal{U}$, $M < K$, and searches for all records in Δ that contain at least one element of S. Note that even though each record is of length P, given a search pattern S the search result for a record is only a single bit, indicating whether the P symbols in the record contain an element in S or not. The overall search result for the dataset is L (independent) bits. We refer to the $M = 1$ setting as *exact* private search, and to the $M > 1$ setting as ***OR*** private search, because the output of the search reveals the exact value of a matching record if $M = 1$, but not if $M > 1$. In general, for OR search we allow M to grow with K (either $o(K)$ or $\Omega(K)$) in the asymptotic regime $K \to \infty$.

To view private search as a special case of DPIR, we treat the result of each possible search pattern as one message. A similar technique has been also used in the work of Fanti [35]. There are a total of $\mu = \binom{K}{M}$ search patterns. Let us arbitrarily label them $S_m, m \in [\mu]$. Correspondingly, there are a total of μ messages. Label these messages W_m, so that $\forall m \in [\mu]$,

$$W_m = (W_m(1), W_m(2), \cdots, W_m(L)), \tag{3.9}$$

and

$$W_m(l) = \begin{cases} 1, & \text{if } \exists i \in [M],\ U_{\theta_i} \in \{\delta_{l1}, \cdots, \delta_{lP}\}, \\ 0, & \text{otherwise.} \end{cases}$$

Note that each message is comprised of L i.i.d. bits. $\forall l \in [L]$,

$$H(w) = H(W_m(l)) = H_2\left(\frac{(K-M)^P}{K^P}\right), \forall m \in [\mu], \tag{3.10}$$

where the binary entropy function is defined as follows.

$$H_2(p) = -p\log_2(p) - (1-p)\log_2(1-p), \tag{3.11}$$

$H_2(0) = H_2(1) = 0$. The second equation in (3.10) is based on the facts that for each record there are a total of K^P possible realizations and $(K-M)^P$ of those do not match. Fig. 3.1 shows the relationship between private search and DPIR. For example, suppose there are $L = 2$ records of length $P = 1$, the alphabet is $\mathcal{U} = \{A, B, C\}$ of size $K = 3$, and we do exact search ($M = 1$). Let the records be $\Delta_1 = A$, $\Delta_2 = C$. Then the $\mu = 3$ messages are shown as in Example 3.1. See Table 3.1 for additional examples.

AND Search

For AND private search, a set $S_m = \{U_{\theta_1}, U_{\theta_2}, \cdots, U_{\theta_M}\}$ is chosen out of a total of $\mu = \binom{K}{M}$ possibilities. In general, M can be arbitrary. However, if $M > P$ the problem degenerates into a trivial case where no record can contain all of the chosen symbols. Therefore, we only consider the non-trivial case where $1 \leq M \leq P$. For all $m \in [\mu], l \in [L]$, the l^{th} bit of the corresponding

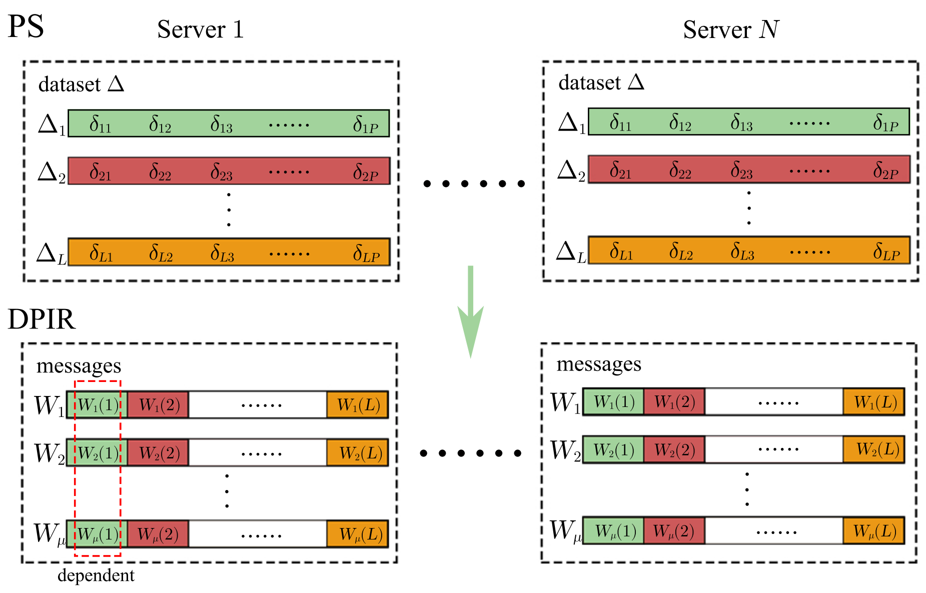

PS
Server 1
Server N
dataset Δ
Δ_1 δ_{11} δ_{12} δ_{13} …… δ_{1P}
Δ_2 δ_{21} δ_{22} δ_{23} …… δ_{2P}
Δ_L δ_{L1} δ_{L2} δ_{L3} …… δ_{LP}
DPIR
messages
W_1 $W_1(1)$ $W_1(2)$ …… $W_1(L)$
W_2 $W_2(1)$ $W_2(2)$ …… $W_2(L)$
W_μ $W_\mu(1)$ $W_\mu(2)$ …… $W_\mu(L)$
dependent

Figure 3.1: Relationship between PS and DPIR. The i-th record (row) Δ_i in PS corresponds to the i-th symbol (column) of each message in DPIR. Different messages (rows) in DPIR correspond to different search patterns S.

message W_m is defined as,

$$W_m(l) = \begin{cases} 1, & \text{if } \forall i \in [M],\ U_{\theta_i} \in \{\delta_{l1}, \cdots, \delta_{lP}\}, \\ 0, & \text{otherwise.} \end{cases}$$

The L-bits of each message are i.i.d., and $\forall l \in [L], \forall m \in [\mu], H(w) = H(W_m(l))$. See Table 3.1 for an example.

NOT Search

For NOT private search, a user privately chooses a value $S_m = \{U_\theta\}$ out of $\mu = K$ possibilities. The l^{th} symbol of the corresponding message W_m is defined as

$$W_m(l) = \begin{cases} 1, & \text{if } U_\theta \notin \{\delta_{l1}, \cdots, \delta_{lP}\}, \\ 0, & \text{otherwise.} \end{cases}$$

Table 3.1: Example of different types of private search. The dataset contains $L = 3$ records (A, B, C), (A, C, B) and (B, B, B). Each record contains $P = 3$ symbols.

Type	Pattern	(A, B, C)	(A, C, B)	(B, B, B)	W_m
exact	A	1	1	0	110
OR	A or B	1	1	1	111
AND	A and B	1	1	0	110
NOT	not A	0	0	1	001
sequence	(A, B)	1	0	0	100

The L bits of each message are i.i.d. and $\forall l \in [L], \forall m \in [\mu], H(w) = H(W_m(l))$. Essentially NOT search is the complement of exact search. For example, in terms of the same chosen symbol, if $W_m(l) = 1$ in exact search, then $W_m(l) = 0$ in NOT search, and vice versa. See Table 3.1 for an example.

Sequence Search

Sequence private search is similar to AND search, the difference is that the order of symbols matters in sequence search. Specifically, a sequence $S_m = (U_{\theta_1}, U_{\theta_2}, \cdots, U_{\theta_M})$ is chosen, out of $\mu = K^M$ possibilities. For the same reason as AND search, we only consider the non-trivial scenario where $1 \leq M \leq P$. For all $m \in [\mu], l \in [L]$, the l^{th} symbol of the corresponding message W_m,

$$W_m(l) = \begin{cases} 1, & \text{if } S_m \in \{(\delta_{l_{i+1}}, \cdots, \delta_{l_{i+M}}), i \in [0 : P - M]\}, \\ 0, & \text{otherwise.} \end{cases}$$

The L-bits of each message are i.i.d., and $\forall l \in [L], \forall m \in [\mu], H(w) = H(W_m(l))$. See Table I for an example.

Even though in our definitions of private search, we assume that all search sets S (or search sequence S) of size M are allowed, one can generalize the definition to restricted search patterns.

One example of such a setting is discussed in Section 3.3.5.

The queries and answers, privacy and correctness constraints, rate and capacity definitions for private search are inherited from DPIR. The *capacity of private search* is denoted $C_{PS}(K, M, P, N)$, and the asymptotic capacity of private search is denoted $\lim_{K\to\infty} C_{PS}(K, M, P, N)$.

3.3 Results

We present the main results in this section. All proofs appear in Section 3.4.

3.3.1 A General Converse for DPIR

The download cost (expected number of bits of download) for DPIR is bounded as follows.

THEOREM 3.1. *For DPIR, denote by $W_1, W_2, \ldots, W_\mu$ an arbitrary permutation of the μ messages. Then*

$$D \geq H(W_1) + \frac{H(W_2|W_1)}{N} + \frac{H(W_3|W_{[1:2]})}{N^2} + \cdots + \frac{H(W_\mu|W_{[1:\mu-1]})}{N^{\mu-1}}. \tag{3.12}$$

Note that the bound depends on the chosen permutation of message indices, so finding the best bound from Theorem 3.1 requires a further optimization of the permutation. Substituting (3.12) into (3.6), we obtain an equivalent bound on capacity. If the messages are independent, we recover the converse bound of [102]. However, Theorem 3.1 is more broadly useful since it allows arbitrary dependencies. Also note that Theorem 3.1 is not limited to balanced DPIR.

3.3.2 General Achievable Rate for DPIR [106]

A PIR achievable scheme for independent messages is also a DPIR achievable scheme. We use the achievable PIR scheme with $\mu \to \infty$ messages from [106, Theorem 2] (also see [23, 96, 103]), from which we obtain the following lower bound on the capacity of DPIR.

$$\begin{aligned} C_{\text{DPIR}}(\mu, N) &\geq \lim_{\mu\to\infty} C_{\text{DPIR}}(\mu, N) \geq \lim_{\mu\to\infty} C_{\text{PIR}}(\mu, N) \\ &= \left(1 - \frac{1}{N}\right) \frac{\min_{m\in[\mu]} H(w_m)}{\max_{m\in[\mu]} H(w_m)}. \end{aligned} \tag{3.13}$$

THEOREM 3.2. *The capacity of DPIR satisfies*

$$C_{\text{DPIR}}(\mu, N) \geq \left(1 - \frac{1}{N}\right) \frac{H_{\min}}{H_{\max}}, \tag{3.14}$$

where $H_{\min} = \min_{m\in[\mu]} H(w_m)$ *and* $H_{\max} = \max_{m\in[\mu]} H(w_m)$.

For balanced DPIR, this gives us $1 - 1/N$ as a lower bound on capacity. As a simple example of the achievable scheme, assume there are $N = 2$ servers and $\mu = 2$ messages with the size of 1 bit. A user wishes to retrieve W_1. He generates two binary random variables α and β independently, and sends (α, β) to server 1 and $(\alpha + 1, \beta)$ to server 2. Here "+" denotes XOR. He downloads $\alpha W_1 + \beta W_2$ and $(\alpha + 1)W_1 + \beta W_2$ from server 1 and server 2, respectively, which allows him to retrieve W_1 privately. Therefore the total download is 2 bits, and $R = 1/2 = 1 - 1/N$. This achievable scheme requires one multiplication for each symbol of each message. In general, for each server, the computation complexity is $O(\mu L)$.

3.3.3 Asymptotic Optimality of Rate $1 - 1/N$ for Balanced DPIR

For balanced DPIR, as the number of messages $\mu \to \infty$, the asymptotic behavior of (3.12) gives us the following sufficient condition. Here we define $W_k = 0$ if $k > \mu$, and define a sequence function to be a sequence of functions $k_1(\mu), k_2(\mu), \cdots$, where every $k_i, i \geq 1$, is a mapping $\mathbb{N} \to \mathbb{N}$. When it is clear from the context, we drop the variable μ and simply use k_i to denote $k_i(\mu)$. But one should keep in mind that k_i depends on the number of messages μ.

THEOREM 3.3. *For balanced DPIR, if there exists an increasing sequence* $k_i \in \mathbb{N}, \forall i \in \mathbb{N}$, *such that* $\forall l \in \mathbb{N}$,

$$\lim_{\mu\to\infty} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)} = 0, \tag{3.15}$$

then the asymptotic capacity is

$$\lim_{\mu\to\infty} C_{\textit{DPIR}}(\mu, N) = 1 - \frac{1}{N}. \tag{3.16}$$

Note since $H(w)$ may depend on μ, the sufficient condition is in general not equivalent to $\lim_{\mu\to\infty} I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right) = 0$. In particular, (3.15) provides a measure of "weak" dependency among the messages in the asymptotic regime, such that the capacity of DPIR is $1 - 1/N$. Intuitively, if we find an infinite sequence of messages that have this weak dependency in DPIR, we know the asymptotic capacity is $1 - 1/N$. If the dependence among messages are all linear, (3.16) $\Rightarrow$ (3.15) is also correct. Please see Section 3.4.5.

3.3.4 Asymptotic Capacity of Private Search

THEOREM 3.4. *The asymptotic capacity of private search is*

$$\lim_{K\to\infty} C_{\textit{PS}}(K, M, P, N) = 1 - \frac{1}{N}, \tag{3.17}$$

for exact search ($M = 1$), NOT search ($M = 1$), OR search ($M > 1$), AND search ($1 \leq M \leq P$) and sequence private search ($1 \leq M \leq P$). For OR search, M can even grow with K, satisfying either $M = o(K)$ or $M = \Omega(K)$.

Theorem 3.4 is proved by showing that the sufficient condition (3.15) is satisfied for private search. Note that condition (3.15) is explicitly proven to be true for balanced DPIR, and private search indeed has balanced messages. Notably, for exact private search, as $K \to \infty$, both $I(W_{k_{l+1}}; W_{k_{[1:l]}})$ and $H(w)$ approach zero. The key to the asymptotic capacity result is that $I(W_{k_{l+1}}; W_{k_{[1:l]}})$ approaches zero much faster than $H(w)$. Furthermore, as shown in Fig. 3.2, convergence of capacity to its asymptotic value is quite fast, and the larger the value of N, the faster the convergence. For example, when the record size $P = 1$ and the number of servers $N = 5$, the bound (3.12) for $K = 10$ messages is already within 1% gap from the asymptotic value.

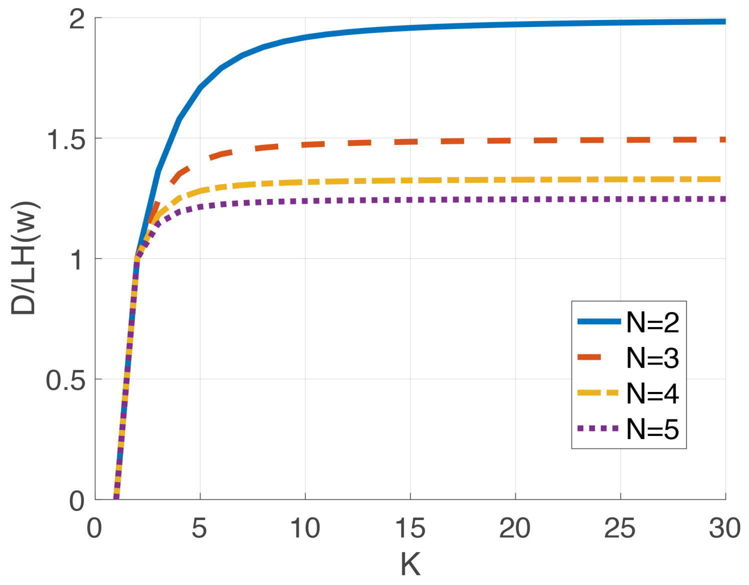


Figure 3.2: Normalized download lower bound of exact search ($P = 1$) based on Theorem 1 versus alphabet size K. The asymptotic value $(1 - 1/N)^{-1}$ is the upper bound.

3.3.5 Difficulty of Private Search over Restricted Search Patterns

Finding the capacity of DPIR with arbitrary dependency structures is in general a difficult problem. The difficulty remains even when the problem is limited to asymptotic capacity. To highlight this aspect, we present an example of approximate private search over restricted search patterns where the asymptotic capacity remains an open problem.

PROPOSITION 3.1. *Consider OR private search, with $P = 1$ and $M = \lfloor \frac{K}{2} \rfloor$, where the only search sets allowed are*

$$S_k = \{U_{\langle k+1 \rangle}, U_{\langle k+2 \rangle}, \cdots, U_{\langle k+M \rangle}\}, \ \ \forall k \in [K], \tag{3.18}$$

and $\langle m \rangle \triangleq (m \bmod K) + 1$. *As* $K \to \infty$, *either the bound* (3.12) *is not tight, or* $\lim_{K\to\infty} C_{PS}(K, M, P, N) \neq 1 - \frac{1}{N}$.

Here privacy is required only within the $\mu = K$ choices of search sets.

3.4 Proofs

3.4.1 Proof of Theorem 3.1

For the DPIR problem, the total download is bounded as,

$$D \geq H\left(A_{[N]}^{[1]} \mid Q_{[N]}^{[1]}\right) \tag{3.19}$$

$$\overset{(3.4)}{=} H\left(A_{[N]}^{[1]}, W_1 \mid Q_{[N]}^{[1]}\right) + o(L) \tag{3.20}$$

$$= H\left(W_1 \mid Q_{[N]}^{[1]}\right) + H\left(A_{[N]}^{[1]} \mid Q_{[N]}^{[1]}, W_1\right) + o(L) \tag{3.21}$$

$$\geq H(W_1) + H\left(A_1^{[1]} \mid Q_{[N]}^{[1]}, W_1\right) + o(L) \tag{3.22}$$

$$= H\left(W_1\right) + H\left(A_1^{[1]} \mid Q_1^{[1]}, W_1\right) + o(L) \tag{3.23}$$

$$\overset{(3.5)}{=} H(W_1) + H\left(A_1^{[2]} \mid Q_1^{[2]}, W_1\right) + o(L) \tag{3.24}$$

$$= H(W_1) + H\left(A_1^{[2]} \mid Q_{[N]}^{[2]}, W_1\right) + o(L), \tag{3.25}$$

where (3.22) follows because the queries are independent of W_1, and $A_1^{[1]}$ is an element of $A_{[N]}^{[1]}$, and (3.23) follows from the fact $A_1^{[1]}$ and $Q_{[2:N]}^{[1]}$ are independent given $Q_1^{[1]}$. Similarly, for all $n \in [2:N]$ we have,

$$D \geq H(W_1) + H\left(A_n^{[2]} \mid Q_{[N]}^{[2]}, W_1\right). \tag{3.26}$$

Adding all of these N inequalities we obtain,

$$ND \geq NH(W_1) + H\left(A_{[N]}^{[2]} \mid Q_{[N]}^{[2]}, W_1\right) \tag{3.27}$$

$$\Rightarrow D \geq H(W_1) + \frac{H\left(A_{[N]}^{[2]} \mid Q_{[N]}^{[2]}, W_1\right)}{N}. \tag{3.28}$$

Proceeding recursively in a similar manner as (3.28), $\forall m \in [2:\mu-1]$, we have

$$\begin{aligned} H\left(A_{[N]}^{[m]} \mid Q_{[N]}^{[m]}, W_1, \cdots, W_{m-1}\right) \geq{}& H\left(W_m \mid W_1, \cdots, W_{m-1}\right) \\ &+ \frac{H\left(A_{[N]}^{[m+1]} \mid Q_{[N]}^{[m+1]}, W_1, \cdots, W_m\right)}{N} \end{aligned} \tag{3.29}$$

and when $m = \mu$,

$$H\left(A_{[N]}^{[\mu]} \mid Q_{[N]}^{[\mu]}, W_1, \cdots, W_{\mu-1}\right) \geq H\left(W_\mu \mid W_1, \cdots, W_{\mu-1}\right). \tag{3.30}$$

Therefore,

$$D \geq H(W_1) + \frac{H\left(A_{[N]}^{[2]} \mid Q_{[N]}^{[2]}, W_1\right)}{N} \tag{3.31}$$

$$\geq H(W_1) + \frac{H(W_2 \mid W_1)}{N} + \frac{H\left(A_{[N]}^{[3]} \mid Q_{[N]}^{[3]}, W_{[1:2]}\right)}{N^2} \tag{3.32}$$

$$\geq \cdots$$

$$\geq H(W_1) + \frac{H(W_2 \mid W_1)}{N} + \frac{H(W_3 \mid W_{[1:2]})}{N^2}$$
$$+ \cdots + \frac{H(W_\mu \mid W_{[1:\mu-1]})}{N^{\mu-1}}. \tag{3.33}$$

3.4.2 Proof for Theorem 3.3

Define m such that $k_m \leq \mu < k_{m+1}$. Note that m is a function of μ and as $\mu \to \infty$, $m \to \infty$. Based on Theorem 3.1 and equations (3.1), (3.2),

$$\begin{aligned} D &\geq H(W_{k_1}) + \frac{H(W_{k_2} \mid W_{k_1})}{N} + \\ &\quad + \frac{H(W_{k_3} \mid W_{k_1}, W_{k_2})}{N^2} + \cdots + \frac{H\left(W_{k_m} \mid W_{k_{[1:m-1]}}\right)}{N^{m-1}} \\ &= H(W_{k_1}) + \frac{H(W_{k_2})}{N} + \cdots + \frac{H(W_{k_m})}{N^{m-1}} \\ &\quad - \frac{I(W_{k_2}; W_{k_1})}{N} - \cdots - \frac{I\left(W_{k_m}; W_{k_{[1:m-1]}}\right)}{N^{m-1}} \\ &= (1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{m-1}})LH(w) \\ &\quad - \frac{I(W_{k_2}; W_{k_1})}{N} - \cdots - \frac{I\left(W_{k_m}; W_{k_{[1:m-1]}}\right)}{N^{m-1}}. \end{aligned} \tag{3.34}$$

Normalizing both sides by $LH(w)$ we have

$$\begin{aligned} \frac{D}{LH(w)} &= \left(1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{m-1}}\right) \\ &\quad - \frac{I(W_{k_2}; W_{k_1})}{NLH(w)} - \cdots - \frac{I\left(W_{k_m}; W_{k_{[1:m-1]}}\right)}{N^{m-1}LH(w)}. \end{aligned} \tag{3.35}$$

Applying limit $\mu \to \infty$, the reciprocal of rate is bounded as

$$\lim_{\mu\to\infty} \frac{D}{LH(w)} \geq \left(1 - \frac{1}{N}\right)^{-1} - \lim_{\mu\to\infty} \sum_{l=1}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l}.$$

Now, we need to show that

$$\lim_{\mu\to\infty} \sum_{l=1}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l} = 0. \tag{3.36}$$

Equivalently, for every $\epsilon > 0$ we will show that

$$\lim_{\mu\to\infty} \sum_{l=1}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l} \leq \epsilon. \tag{3.37}$$

Choose a finite l^* such that

$$\frac{1}{N^{l^*}} \left(1 - \frac{1}{N}\right)^{-1} \leq \epsilon. \tag{3.38}$$

Note that l^* depends only on N and ϵ. More importantly, it is not a function of μ. Now partition the sum as follows

$$\begin{aligned} &\lim_{\mu\to\infty} \sum_{l=1}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l} \\ &= \lim_{\mu\to\infty} \sum_{l=1}^{l^*-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l} + \lim_{\mu\to\infty} \sum_{l=l^*}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l}. \end{aligned} \tag{3.39}$$

The first term on the RHS of (3.39) is zero because it is a sum of finitely many terms (l^* is finite), each of which is zero because (3.15) holds by assumption. For the second term in (3.39),

$$\lim_{\mu\to\infty} \sum_{l=l^*}^{m-1} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)N^l} \leq \lim_{\mu\to\infty} \sum_{l=l^*}^{m-1} \frac{1}{N^l} \tag{3.40}$$

$$\leq \frac{1}{N^{l^*}} \lim_{\mu\to\infty} \sum_{l=0}^{m-1-l^*} \frac{1}{N^l} \tag{3.41}$$

$$\leq \frac{1}{N^{l^*}} \left(1 - \frac{1}{N}\right)^{-1} \leq \epsilon. \tag{3.42}$$

Thus, the reciprocal of rate is bounded as $1/R \geq (1 - 1/N)^{-1}$, i.e., the rate is bounded as $R \leq 1 - 1/N$. By Theorem 3.2 this rate is achievable. Hence proved.

3.4.3 Proof of Theorem 3.4

We treat private search as a balanced DPIR problem. As an application of Theorem 3.3, we show that (3.15) is satisfied. Therefore the asymptotic capacity must be $1 - 1/N$. Note that for all the private search variations, the number of messages $\mu \to \infty$ if and only if the alphabet size $K \to \infty$. So in our proofs we let K grow to infinity.

In the following proofs, we consider a subset of the possible messages, $W_1, W_2, \dots, W_{f(\mu)}$, for some $f(\mu) \leq \mu$ that grows with μ. We use the identity sequence functions $k_i = i$ for $1 \leq i \leq f(\mu)$, and map i to some $k_i > \mu$ for $i > f(\mu)$. In other words, we only use the first $f(\mu)$ messages in our proofs. Then (3.15) becomes

$$\lim_{\mu\to\infty} \frac{I\left(W_{l+1}; W_{[1:l]}\right)}{LH(w)} = 0 \tag{3.43}$$

for all $1 \leq l \leq f(\mu)$.

Exact private search

We start with the exact private search problem $(M = 1)$.

Firstly, consider the case where the record length $P = 1$, note that

$$\lim_{K\to\infty} H(w) = \lim_{K\to\infty} H_2\left(\frac{1}{K}\right) = 0. \tag{3.44}$$

According to L′Hôpital′s rule,

$$\lim_{K\to\infty} \frac{H_2\left(\frac{1}{K-t}\right)}{H_2\left(\frac{1}{K}\right)} = 1, \tag{3.45}$$

where t is a constant. The detailed proof of (3.45) is shown below. Let $f(p) = H_2(p) = -p\log_2(p) - (1-p)\log_2(1-p)$. When $K \to \infty$, $\frac{1}{K-t} \to 0$ and $\frac{1}{K} \to 0$. Since $\lim_{K\to\infty} f\left(\frac{1}{K-t}\right) = 0$, $\lim_{K\to\infty} f\left(\frac{1}{K}\right) = 0$ and both of them are differentiable, L′Hôpital′s rule is applicable. Consider the derivative of $f(p)$,

$$\frac{\mathrm{d}f}{\mathrm{d}p} = -\log_2 p - \frac{p}{p\ln 2} + \log_2(1-p) + \frac{1-p}{(1-p)\ln 2} \tag{3.46}$$

$$= \log_2(1-p) - \log_2 p = \log_2 \frac{1-p}{p}. \tag{3.47}$$

Let $p = \frac{1}{K-t}$ and $q = \frac{1}{K}$. According to L′Hôpital′s rule,

$$\lim_{K\to\infty} \frac{f\left(\frac{1}{K-t}\right)}{f\left(\frac{1}{K}\right)} = \lim_{K\to\infty} \frac{\frac{\mathrm{d}f}{\mathrm{d}p}\frac{\mathrm{d}p}{\mathrm{d}K}}{\frac{\mathrm{d}f}{\mathrm{d}q}\frac{\mathrm{d}q}{\mathrm{d}K}} \tag{3.48}$$

$$= \lim_{K\to\infty} \frac{\frac{-1}{(K-t)^2}\log_2\frac{1-p}{p}}{\frac{-1}{K^2}\log_2\frac{1-q}{q}} \tag{3.49}$$

$$= \lim_{K\to\infty} \frac{\frac{-1}{(K-t)^2}\log_2(K-t-1)}{\frac{-1}{K^2}\log_2(K-1)}. \tag{3.50}$$

Since $\lim_{K\to\infty} \frac{\frac{-1}{(K-t)^2}}{\frac{-1}{K^2}} = 1$ and $\lim_{K\to\infty} \frac{\log_2(K-t-1)}{\log_2(K-1)} = 1$, we obtain

$$\lim_{K\to\infty} \frac{f\left(\frac{1}{K-t}\right)}{f\left(\frac{1}{K}\right)} = 1. \tag{3.51}$$

Since $\forall l \in [K], W_l(1), \cdots, W_l(L)$ are i.i.d., $H(W_l) = LH\left(W_l(\eta)\right)$, η can be any integer between 1 to L. Consider the dependence among the messages,

$$\frac{H\left(W_{l+1} \mid W_{[1:l]}\right)}{L} = H\left(W_{l+1}(\eta) \mid W_{[1:l]}(\eta)\right) \tag{3.52}$$

$$= Pr\left(W_{[1:l]}(\eta) = \vec{0}\right) \cdot H\left(W_{l+1}(\eta) \mid W_{[1:l]}(\eta) = \vec{0}\right)$$
$$+ \sum_{i=1}^{l} Pr\left(W_i(\eta) = 1, W_{[1:l]\setminus\{i\}}(\eta) = \vec{0}\right)$$
$$\cdot H\left(W_{l+1}(\eta) \mid W_i(\eta) = 1, W_{[1:l]\setminus\{i\}}(\eta) = \vec{0}\right) \tag{3.53}$$

$$= \left(1 - \frac{l}{K}\right) H_2\left(\frac{1}{K-l}\right) + \sum_{i=1}^{l} \frac{1}{K} \cdot 0 \tag{3.54}$$

$$= \left(1 - \frac{l}{K}\right) H_2\left(\frac{1}{K-l}\right), \tag{3.55}$$

where $Pr(e)$ is the probability of event e and bold $\vec{0}$ is the zero vector. The above equalities are explained as below. The only possible values for $W_{[1:l]}(\eta)$ are either all zeros or 1 one and $l-1$ zeros. Note that the probability $Pr\left(W_{[1:l]}(\eta) = \vec{0}\right) = 1 - l/K$. If $W_i(\eta) = 0, \forall i \in [l]$, then $\Delta_\eta \neq U_1, \cdots, U_l$ and Δ_η can only take values from $\{U_{l+1}, U_{l+2}, \cdots, U_K\}$, each with probability $1/(K-l)$. Therefore, conditioning on $W_{[1:l]}(\eta) = \vec{0}$, we have $\Delta_\eta = U_{l+1}$, i.e., $W_{l+1}(\eta) = 1$, with probability $1/(K-l)$, and $\Delta_\eta \neq U_{l+1}$, i.e., $W_{l+1}(\eta) = 0$, with probability $1 - 1/(K-l)$. If $W_1(\eta) = 1$, then $\Delta_\eta = U_1$ and $W_2(\eta), \cdots, W_K(\eta)$ must be equal to zero. Thus there is at most one $W_i(\eta) = 1$ and each $W_i(\eta) = 1$ with probability $1/K$. If any $W_i(\eta) = 1$, then $H\left(W_j(\eta) \mid W_i(\eta)\right) = 0, \forall j \neq i$.

Substituting $\mu = K$ into the LHS of (3.43), we have for any fixed $l \in \mathbb{N}$,

$$\lim_{K\to\infty} \frac{I(W_{l+1}; W_1, W_2, \cdots, W_l)}{LH_2\left(\frac{1}{K}\right)} \tag{3.56}$$

$$= \lim_{K\to\infty} \frac{H(W_{l+1}) - H(W_{l+1} \mid W_{[1:l]})}{LH_2(\frac{1}{K})} \tag{3.57}$$

$$= \lim_{K\to\infty} \frac{H_2\left(\frac{1}{K}\right) - \left(1 - \frac{l}{K}\right) H_2\left(\frac{1}{K-l}\right)}{H_2\left(\frac{1}{K}\right)} \tag{3.58}$$

$$= 1 - \lim_{K\to\infty} \frac{\left(1-\frac{l}{K}\right) H_2\left(\frac{1}{K-l}\right)}{H_2\left(\frac{1}{K}\right)} = 1 - 1 = 0. \tag{3.59}$$

Here (3.59) follows from (3.45). Therefore, (3.43) is satisfied, and based on Theorem 3.3, the asymptotic capacity is $1 - 1/N$.

Then consider the case where $P > 1$ is a constant, note that

$$\lim_{K\to\infty} H(w) = \lim_{K\to\infty} H_2\left(\frac{(K-1)^P}{K^P}\right) = 0. \tag{3.60}$$

According to L′Hôpital′s rule,

$$\lim_{K\to\infty} \frac{H_2\left(\frac{(K-t-1)^P}{(K-t)^P}\right)}{H_2\left(\frac{(K-1)^P}{K^P}\right)} = 1, \tag{3.61}$$

where t is a constant. The detailed proof of (3.61) is shown below. Let $f(p) = H_2(p) = -p\log_2(p) - (1-p)\log_2(1-p)$. When $K \to \infty$, $\frac{(K-t-1)^P}{(K-t)^P} \to 1$ and $\frac{(K-1)^P}{K^P} \to 1$. Since $\lim_{K\to\infty} f\left(\frac{(K-t-1)^P}{(K-t)^P}\right) = 0$, $\lim_{K\to\infty} f\left(\frac{(K-1)^P}{K^P}\right) = 0$ and both of them are differentiable, L′Hôpital′s rule is applicable. Let $p = \frac{(K-t-1)^P}{(K-t)^P}$ and $q = \frac{(K-1)^P}{K^P}$. According to L′Hôpital′s rule,

$$\lim_{K\to\infty} \frac{f\left(\frac{(K-t-1)^P}{(K-t)^P}\right)}{f\left(\frac{(K-1)^P}{K^P}\right)} = \lim_{K\to\infty} \frac{\frac{\mathrm{d}f}{\mathrm{d}p}\frac{\mathrm{d}p}{\mathrm{d}K}}{\frac{\mathrm{d}f}{\mathrm{d}q}\frac{\mathrm{d}q}{\mathrm{d}K}} \tag{3.62}$$

$$= \lim_{K\to\infty} \frac{\frac{P(K-t-1)^{P-1}(K-t)^{P-1}}{(K-t)^{2P}} \log_2 \frac{1-p}{p}}{\frac{P(K-1)^{P-1}K^{P-1}}{K^{2P}} \log_2 \frac{1-q}{q}} \tag{3.63}$$

$$= \lim_{K\to\infty} \frac{\frac{P(K-t-1)^{P-1}(K-t)^{P-1}}{(K-t)^{2P}} \log_2 \left(\frac{(K-t)^P}{(K-t-1)^P} - 1\right)}{\frac{P(K-1)^{P-1}K^{P-1}}{K^{2P}} \log_2 \left(\frac{K^P}{(K-1)^P} - 1\right)}. \tag{3.64}$$

Since

$$\lim_{K\to\infty} \frac{\frac{P(K-t-1)^{P-1}(K-t)^{P-1}}{(K-t)^{2P}}}{\frac{P(K-1)^{P-1}K^{P-1}}{K^{2P}}} = 1 \tag{3.65}$$

and

$$\lim_{K\to\infty} \frac{\log_2\left(\frac{(K-t)^P}{(K-t-1)^P} - 1\right)}{\log_2\left(\frac{K^P}{(K-1)^P} - 1\right)} \tag{3.66}$$

$$\overset{\text{L'Hôpital's rule}}{=} \lim_{K\to\infty} \frac{\frac{1}{\left(\frac{(K-t)^P}{(K-t-1)^P}-1\right)\ln 2} \cdot \frac{-P(K-t)^{P-1}}{(K-t-1)^{P+1}}}{\frac{1}{\left(\frac{K^P}{(K-1)^P}-1\right)\ln 2} \cdot \frac{-PK^{P-1}}{(K-1)^{P+1}}} = 1, \tag{3.67}$$

we obtain

$$\lim_{K\to\infty} \frac{f\left(\frac{(K-t-1)^P}{(K-t)^P}\right)}{f\left(\frac{(K-1)^P}{K^P}\right)} = 1. \tag{3.68}$$

For any fixed $l \in \mathbb{N}$, we denote by E_i the event that i entries out of $W_1(\eta), \cdots, W_l(\eta)$ are 1, and the remaining $l - i$ entries are 0. Let its probability be $\tau_i = Pr(E_i)$. Since each record size is P, there are at most P of $W_i(\eta)$ equal to 1, hence $0 \le i \le \min(P, l)$.

$$H(W_{l+1} \mid W_l, \cdots, W_1) \tag{3.69}$$

$$= LH(W_{l+1}(\eta) \mid W_l(\eta), \cdots, W_1(\eta)) \tag{3.70}$$

$$= \sum_{i=0}^{\min(P,l)} \binom{l}{i} \tau_i LH(W_{l+1}(\eta) \mid E_i) \tag{3.71}$$

$$\ge \binom{l}{0} \tau_0 LH(W_{l+1}(\eta) \mid E_0) \tag{3.72}$$

$$= \tau_0 LH(W_{l+1}(\eta) \mid E_0). \tag{3.73}$$

Note that

$$\tau_0 = Pr(E_0) = \frac{(K-l)^P}{K^P}, \tag{3.74}$$

and conditioned on E_0, $W_{l+1}(\eta)$ is 0 with probability $(K-l-1)^P/(K-l)^P$, thus

$$H(W_{l+1}(\eta) \mid E_0) = H_2\left(\frac{(K-l-1)^P}{(K-l)^P}\right). \tag{3.75}$$

Therefore,

$$\begin{align} H(W_{l+1} \mid W_l, \cdots, W_1) &\geq \tau_0 L H(W_{l+1}(\eta) \mid E_0) \tag{3.76} \\ &= \frac{L(K-l)^P}{K^P} H_2\left(\frac{(K-l-1)^P}{(K-l)^P}\right). \tag{3.77} \end{align}$$

Substituting $\mu = K$ into the LHS of (3.43), we have

$$\begin{align} &\lim_{K\to\infty} \frac{I(W_{l+1}; W_1, W_2, \cdots, W_l)}{L H_2\left(\frac{(K-1)^P}{K^P}\right)} \tag{3.78} \\ =& \lim_{K\to\infty} \frac{H(W_{l+1}) - H(W_{l+1} \mid W_{[1:l]})}{L H_2(\frac{(K-1)^P}{K^P})} \tag{3.79} \\ \leq& \lim_{K\to\infty} \frac{H_2(\frac{(K-1)^P}{K^P}) - \frac{(K-l)^P}{K^P} H_2\left(\frac{(K-l-1)^P}{(K-l)^P}\right)}{H_2(\frac{(K-1)^P}{K^P})} \tag{3.80} \\ =& 1 - \lim_{K\to\infty} \frac{\frac{(K-l)^P}{K^P} H_2\left(\frac{(K-l-1)^P}{(K-l)^P}\right)}{H_2(\frac{(K-1)^P}{K^P})} = 1 - 1 = 0. \tag{3.81} \end{align}$$

Here (3.81) follows from (3.61). Therefore, (3.43) is satisfied, and based on Theorem 3.3, the asymptotic capacity is $1 - 1/N$.

In summary, for arbitrary constant P the asymptotic capacity of exact private search is $1 - 1/N$.

NOT search

Since NOT search essentially is the complement of exact search, the asymptotic capacity of NOT search is $1 - 1/N$.

OR search

For OR search, first we show that any $P > 1$ OR search problems can be viewed as $P = 1$ OR search problem. For example, suppose the alphabet set is $\{A, B, C\}$, i.e., $K = 3$ and record length $P = 2$. A user wishes to search for A or B. It is equivalent to the problem that the alphabet set is all ordered tuples consisting of $\{A, B, C\}$, i.e. $\{AA, AB, AC, BA, BB, BC, CA, CB, CC\}$ and the record length is $P' = 1$. Notice that every character in the new alphabet set is searched for with the same probability. The user wishes to search for AA or AB or AC or BA or BB or BC or CA or CB. In general, for OR search problem with alphabet size K, record length $P > 1$ and search set size M, it is equivalent to the OR search problem with alphabet size $K' = K^P$, record length $P' = 1$ and search set size $M' = K' - (K - M)^P$. Note that if $M = o(K)$,

$$\lim_{K\to\infty} \frac{M'}{K'} = \lim_{K\to\infty} \frac{K^P - (K-M)^P}{K^P} = 0, \tag{3.82}$$

i.e. $M' = o(K')$. Similarly, if $M = \Omega(K), M' = \Omega(K')$. Therefore in the following proof of OR private search, we only consider the case $P = 1$.

Define $\gamma \triangleq M/K < 1$. When $M = o(K)$, regard each M-element set as one new symbol and consider messages corresponding to disjoint search patterns. For example, suppose the alphabet set is $\{1, 2, \cdots, K\}$, $M = 2$, regard $\{1, 2\}, \{1, 3\}, \{2, 3\}, \cdots$ as new symbols. Consider the messages corresponding to $\{1, 2\}, \{3, 4\}, \{5, 6\}, \cdots$. There are $K' = \lfloor K/2 \rfloor$ such messages. As $K \to \infty$, $K' \to \infty$. Based on the proof for the exact search setting, these messages satisfy (3.15). Therefore the asymptotic capacity is $1 - 1/N$. For the general case, consider the $K' = \lfloor K/M \rfloor$ messages corresponding to disjoint search patterns. Invoking Theorem 3.3 we conclude that the asymptotic capacity is $1 - 1/N$.

For $M = \Omega(K)$, by symmetry of the truth function, searching for a given set is the same as searching for its complement. The entropy $H_2(\gamma) = H_2(1-\gamma)$ and the capacity as a function of γ, is symmetric around $\gamma = 1/2$. Thus we only need to consider $\gamma = M/K \leq 1/2$. Let us find a sequence of dependent messages such that (3.43) is satisfied. Choose W_1 corresponding to $S_1 = \{U_1, U_2, \cdots, U_M\}$. It separates the alphabet set $\mathcal{U}$ into 2 parts: S_1 of size γK, and $\mathcal{U}\backslash S_1$ of size $(1-\gamma)K$. Note that $\gamma K = M$ is an integer. Choose the second message W_2 so that it is comprised of $\lfloor \gamma M \rfloor$ elements of S_1 and $M - \lfloor \gamma M \rfloor$ elements of $\mathcal{U}\backslash S_1$. Repeating this step we get a series of dependent messages, as in Fig. 3.3.

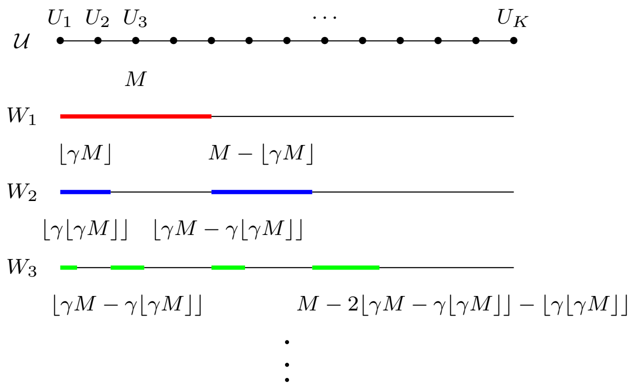

U_1 U_2 U_3
$\cdots$
U_K
$\mathcal{U}$
M
W_1
$\lfloor \gamma M \rfloor$
$M - \lfloor \gamma M \rfloor$
W_2
$\lfloor \gamma \lfloor \gamma M \rfloor \rfloor$
$\lfloor \gamma M - \gamma \lfloor \gamma M \rfloor \rfloor$
W_3
$\lfloor \gamma M - \gamma \lfloor \gamma M \rfloor \rfloor$
$M - 2\lfloor \gamma M - \gamma \lfloor \gamma M \rfloor \rfloor - \lfloor \gamma \lfloor \gamma M \rfloor \rfloor$

Figure 3.3: Partition of the alphabet to obtain a sequence of dependent messages for OR search, $M = \Omega(K), P = 1$. Here $\gamma = M/K$. The alphabet $\mathcal{U} = \{U_1, U_2, \cdots, U_K\}$ is represented on a straight line.

Note that

$$H(W_l) = LH_2(\gamma), \ \forall l. \tag{3.83}$$

Since $\gamma \leq 1/2$, $\frac{\gamma M - 1}{M} \leq \frac{\lfloor \gamma M \rfloor}{M} \leq 1/2$. In terms of function $H_2(x)$, $\frac{\gamma M - 1}{M}$ and $\frac{\lfloor \gamma M \rfloor}{M}$ are both in the monotonically increasing range. When $\frac{M-\lfloor\gamma M\rfloor}{K-M} \leq 1/2$,

$$
\begin{aligned}
& H(W_2|W_1) \\
&= LH_2\left(\frac{\lfloor\gamma M\rfloor}{M}\right)\frac{M}{K} + LH_2\left(\frac{M-\lfloor\gamma M\rfloor}{K-M}\right)\frac{K-M}{K} && (3.84)\\
&\geq LH_2\left(\frac{\gamma M-1}{M}\right)\frac{M}{K} + LH_2\left(\frac{\gamma(K-M)-1}{K-M}\right)\frac{K-M}{K} && (3.85)\\
&= LH_2\left(\frac{\gamma^2 K-1}{\gamma K}\right)\frac{M}{K} + LH_2\left(\frac{\gamma(1-\gamma)K-1}{(1-\gamma)K}\right)\frac{K-M}{K}. && (3.86)
\end{aligned}
$$

Then we have

$$
\begin{aligned}
& \lim_{K\to\infty} H(W_2|W_1) \geq LH_2(\gamma) = H(W_1), && (3.87)\\
\Rightarrow & \lim_{K\to\infty} H(W_2|W_1) = H(W_1). && (3.88)
\end{aligned}
$$

When $\frac{M-\lfloor\gamma M\rfloor}{K-M} \geq 1/2$, $\frac{M-\lfloor\gamma M\rfloor}{K-M}$ and $\frac{\gamma(K-M)-1}{K-M}$ are in non-monotonic range, (3.85) is still true. Due to the symmetry, we only need to show $\frac{M-\lfloor\gamma M\rfloor}{K-M}$ is closer to $1/2$ than $\frac{\gamma(K-M)-1}{K-M}$. We have

$$
\begin{aligned}
& \frac{M-\lfloor\gamma M\rfloor}{K-M} - \frac{1}{2} \\
&= \frac{M-\lfloor\gamma M\rfloor}{K-M} - \gamma + \gamma - \frac{1}{2} && (3.89)\\
&= \frac{M-\lfloor\gamma M\rfloor}{K-M} - \frac{M-\gamma M}{K-M} + \gamma - \frac{1}{2} && (3.90)\\
&= \frac{\gamma M-\lfloor\gamma M\rfloor}{K-M} - \frac{1}{2} + \gamma && (3.91)\\
&\leq \frac{1}{K-M} - \frac{1}{2} + \gamma && (3.92)\\
&\leq \frac{1}{K-M}. && (3.93)
\end{aligned}
$$

And

$$
\frac{1}{2} - \frac{M-\gamma M-1}{K-M}
$$

$$= \frac{1}{2} - \frac{M - \gamma M - 1}{K - M} + \gamma - \gamma \tag{3.94}$$
$$= \frac{1}{2} - \frac{M - \gamma M - 1}{K - M} + \frac{M - \gamma M}{K - M} - \gamma \tag{3.95}$$
$$= \frac{1}{2} + \frac{1}{K - M} + \gamma \tag{3.96}$$
$$\geq \frac{1}{K - M}. \tag{3.97}$$

Combining (3.93) and (3.97), (3.85) is satisfied.

Since $M = \Omega(K)$, there exists a constant $0 < c < 1$ such that $\gamma = M/K \geq c$ for sufficiently large K. For a given K, consider the search of only the restricted messages $\{W_l : l \leq \log_{1/c} \sqrt{K}\}$. Note that the number of the restricted messages goes to infinity as $K \to \infty$. Next we prove

$$\lim_{K \to \infty} \frac{H(W_l | W_{[l-1]})}{L H_2(\gamma)} = 1, \ \forall l \leq \log_{1/c} \sqrt{K}. \tag{3.98}$$

According to our choice of the message W_l in Fig. 3.3, we partition the alphabet into 2^l parts at step l. Thus there are 2^{l-1} terms in $H(W_l | W_{[l-1]})$. In particular, $\forall i \in [2^{l-1}]$, the i-th term corresponds to the event that the record symbol is in the i-th part, and we use ξ_i to denote its probability. To bound the i-th term, let us use a binary number to represent $i - 1$. Let the number of "1"s in the binary number be m_i and $m_i \in [l-1]$. For example, if $l = 4$ and $i = 2$, then $i - 1 = (001)_2$, and $m_i = 1$. The size of the i-th part is between $\gamma^{l-m_i}(1-\gamma)^{m_i} K - l + 1$ and $\gamma^{l-m_i}(1-\gamma)^{m_i} K + l - 1$. Then the i-th term of $H(W_l | W_1, \cdots, W_{l-1})$ is greater than or equal to

$$L H_2 \left(\frac{\gamma^{l-m_i+1}(1-\gamma)^{m_i} K - l + 1}{\gamma^{l-m_i}(1-\gamma)^{m_i} K + l - 1} \right) \cdot \xi_i \tag{3.99}$$
$$= L H_2 \left(\frac{\gamma - \frac{l-1}{\gamma^{l-m_i}(1-\gamma)^{m_i} K}}{1 + \frac{l-1}{\gamma^{l-m_i}(1-\gamma)^{m_i} K}} \right) \cdot \xi_i. \tag{3.100}$$

When $K \to \infty, \forall i \in [2^{l-1}], l \leq \log_{1/c} \sqrt{K}$,

$$\lim_{K\to\infty} \frac{l-1}{\gamma^{l-m_i}(1-\gamma)^{m_i}K} \leq \lim_{K\to\infty} \frac{l-1}{\gamma^l K} = 0. \tag{3.101}$$

Therefore,

$$\lim_{K\to\infty} LH_2\left(\frac{\gamma - \frac{l-1}{\gamma^{l-m_i}(1-\gamma)^{m_i}K}}{1+\frac{l-1}{\gamma^{l-m_i}(1-\gamma)^{m_i}K}}\right) = \lim_{K\to\infty} LH_2(\gamma). \tag{3.102}$$

Summing up all the terms, we obtain

$$\lim_{K\to\infty} H(W_l|W_1,\cdots,W_{l-1}) \geq \lim_{K\to\infty} LH_2(\gamma), \tag{3.103}$$

$$\Rightarrow \lim_{K\to\infty} H(W_l|W_1,\cdots,W_{l-1}) = \lim_{K\to\infty} H(W_l). \tag{3.104}$$

Invoking Theorem 3.3 at this point, we conclude that the asymptotic capacity is $1 - 1/N$.

AND private search

For AND search, the record size $P \geq M$, otherwise no record matches. Similar to OR search, we translate it to OR search with the record size $P' = 1$. For example, consider the alphabet set is $\{A, B, C\}$, i.e., $K = 3$ and record length $P = 2$. A user wishes to search for A and B. It is equivalent to the problem that the alphabet set is all ordered tuples consisting of $\{A, B, C\}$, i.e. $\{AA, AB, AC, BA, BB, BC, CA, CB, CC\}$ and the record length is $P' = 1$. The user wishes to search for AB or BA.

In general case, for AND search problem with alphabet size K, search set size M and record length $P \geq M$, it is equivalent to the OR search problem with alphabet size $K' = K^P$, record length $P' = 1$ and search set size $M' = o(K')$. Here M' is the number of matching cases for an AND search, which is a function of P, M, K, i.e. $M' = \Gamma(P, M, K)$. To show $M' = o(K')$, we first calculate the value of $\Gamma(P, M, K)$. Notice the fact that there are two cases where a P-symbol length record matches: $1)$ The first $P-1$ symbols already contain all of the M chosen symbols, and $2)$ The first $P-1$ symbols only contain $M-1$ chosen symbols. For the first case, the last symbol can be any one in the alphabet while for the second case, the last symbol must be the missing symbol and any one of the M symbols could be missing. Thus, the value of $\Gamma(P, M, K)$ can be calculated by recursive equation

$$\Gamma(P, M, K) = K\Gamma(P-1, M, K) + M\Gamma(P-1, M-1, K-1), \tag{3.105}$$

with base cases

$$\Gamma(P, 1, K) = K^P - (K-1)^P, \tag{3.106}$$

$$\Gamma(M, M, K) = M!\,. \tag{3.107}$$

Recall that P and M are constants, which do not grow with K. When $K \to \infty$, $\Gamma(P, 1, K) = o(K^P)$ and $\Gamma(M, M, K) = O(1)$. Suppose $\Gamma(P-1, M, K) = o(K^{P-1})$ and $\Gamma(P-1, M-1, K-1) = o((K-1)^{P-1})$,

$$\Gamma(P, M, K) = K \cdot o\left(K^{P-1}\right) + M \cdot o\left((K-1)^{P-1}\right) = o\left(K^P\right). \tag{3.108}$$

Based on mathematical induction, $\forall M$ and $\forall P \geq M$,

$$\Gamma(P, M, K) = o(K^P) = o(K'). \tag{3.109}$$

Since the asymptotic capacity of OR search is $1 - 1/N$, the asymptotic capacity of AND search is $1 - 1/N$.

Sequence private search

For sequence search, the non-trival case is under the condition $P \geq M$, otherwise no record matches. Suppose the chosen sequence is the tuple $S = (U_{\theta_1}, \cdots, U_{\theta_M})$. Note that here U_{θ_i} and U_{θ_j} can be the same symbol even through $i \neq j$. For sequence search, again we translate it to OR search with the record size $P' = 1$. Consider the same example where alphabet set is $\{A, B, C\}$, i.e., $K = 3$ and record length $P = 2$. A user wishes to search for a sequence AB. It is equivalent to the problem that the alphabet set is all ordered tuples consisting of $\{A, B, C\}$, i.e. $\{AA, AB, AC, BA, BB, BC, CA, CB, CC\}$ and the record length is $P' = 1$. The user wishes to search for AB.

In general case, for sequence search problem with alphabet size K, search set size M and record length $P \geq M$, it is equivalent to the OR search problem with alphabet size $K' = K^P$, record length $P' = 1$ and search set size $M' = o(K')$. Here M' is the number of matching cases for a sequence search, which is a function of P, M, K, i.e. $M' = \Psi(P, M, K)$. To show $M' = o(K')$, note that if a sequence is contained in a record, every character in the sequence must be contained in that record,

$$\Psi(P, M, K) \leq \Gamma(P, m, K) = o(K^P) = o(K'), \tag{3.110}$$

where $m = |\{U_{\theta_1}, \cdots, U_{\theta_M}\}| \leq M$ is the number of distinct searched symbols. Therefore the asymptotic capacity of sequence search is $1 - 1/N$.

3.4.4 Proof of Proposition 3.1

Consider the even values of K as it approaches infinity so that we have $H(W_k(\eta)) = H(1/2) = 1$ bit, i.e., each message bit is marginally uniform. We prove the proposition by contradiction. Suppose the asymptotic capacity is $1 - \frac{1}{N}$, namely, $\lim_{K\to\infty} \frac{D}{LH(1/2)} = \left(1 - \frac{1}{N}\right)^{-1}$, and suppose the bound (3.12) is tight for some sequence $k_1, k_2, \cdots$. Note that message W_{k_i} corresponds to search set S_{k_i}. Then we have the following equation.

$$\begin{aligned}
&\lim_{K\to\infty} 1 + \frac{1}{N} + \frac{1}{N^2} + \cdots = \lim_{K\to\infty} \frac{D}{LH(\frac{1}{2})} = \lim_{K\to\infty} \frac{D}{L} \\
&\overset{(3.12)}{=} \lim_{K\to\infty} H(W_{k_1}(\eta)) + \frac{1}{N} H(W_{k_2}(\eta) \mid W_{k_1}(\eta)) \\
&+ \frac{1}{N^2} H\left(W_{k_3}(\eta) \mid W_{k_1}(\eta), W_{k_2}(\eta)\right) + \dots
\end{aligned} \tag{3.111}$$

Therefore,

$$\begin{aligned}
0 = &\lim_{K\to\infty} \frac{1}{N} \left(1 - H(W_{k_2}(\eta) \mid W_{k_1}(\eta))\right) \\
&+ \frac{1}{N^2} \left(1 - H\left(W_{k_3}(\eta) \mid W_{k_1}(\eta), W_{k_2}(\eta)\right)\right) + \cdots,
\end{aligned}$$

which implies that

$$\lim_{K\to\infty} H\left(W_{k_2}(\eta) \mid W_{k_1}(\eta)\right) = 1, \tag{3.112}$$

$$\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid W_{k_1}(\eta), W_{k_2}(\eta)\right) = 1. \tag{3.113}$$

Let us represent $U_1, U_2, \cdots, U_K$ on an alphabet circle $\mathcal{U}$ shown in Fig. 3.4.

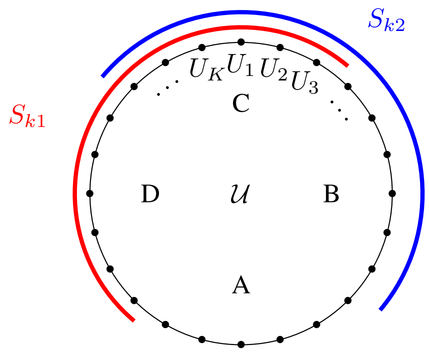


Figure 3.4: Alphabet circle

Since S_{k_1} is a contiguous set of $K/2$ points on the circle, without loss of generality it may be represented by the red semi-circle. $W_{k_1}(\eta)$ and $W_{k_2}(\eta)$ are binary random variables. So if $\lim_{K\to\infty} H\left(W_{k_2}(\eta) \mid W_{k_1}(\eta)\right) = 1$, then

$$\lim_{K\to\infty} H\left(W_{k_2}(\eta) \mid W_{k_1}(\eta) = 0\right) = 1, \tag{3.114}$$

$$\lim_{K\to\infty} H\left(W_{k_2}(\eta) \mid W_{k_1}(\eta) = 1\right) = 1. \tag{3.115}$$

This is equivalent to, within S_{k_1} half of the points must be in S_{k_2} and half of the points must be outside S_{k_2}, when K approaches infinity. Similar for the points outside S_{k_1}. Therefore, without loss of generality, S_{k_2} is represented by the blue semi-circle on the alphabet circle. Note that this divides the alphabet circle into 4 parts, labeled as A, B, C, D, corresponding to $(W_{k_1}(\eta), W_{k_2}(\eta)) = (0,0), (0,1), (1,1), (1,0)$, respectively. Note that each of these spans $K/4 + o(K)$ points.

Since $\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid W_{k_1}(\eta), W_{k_2}(\eta)\right) = 1$, then

$$\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid (W_{k_1}(\eta), W_{k_2}(\eta)) = (0,0)\right) = 1, \tag{3.116}$$

$$\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid (W_{k_1}(\eta), W_{k_2}(\eta)) = (0,1)\right) = 1, \tag{3.117}$$

$$\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid (W_{k_1}(\eta), W_{k_2}(\eta)) = (1,1)\right) = 1, \tag{3.118}$$

$$\lim_{K\to\infty} H\left(W_{k_3}(\eta) \mid (W_{k_1}(\eta), W_{k_2}(\eta)) = (1,0)\right) = 1. \tag{3.119}$$

Consider (3.116), it is the sector of the $\mathcal{U}$ circle labeled A. Within this sector $W_{k_3}(\eta)$ must be uniform, i.e., half of A must be in S_{k_3} and half of A must be outside S_{k_3}. Similarly, conditions (3.117), (3.118) and (3.119) imply that half of B, C, D must be in S_{k_3} and half of B, C, D must be outside S_{k_3}. But S_{k_3} is a contiguous semicircle, a continuous semi-circle cannot overlap with half of each of A, B, C, D. Therefore we have a contradiction. The contradiction means that either the asymptotic capacity of OR search with special patterns is not $1 - 1/N$ or Theorem 3.1 is not tight for this OR private search.

3.4.5 Proof of (3.16) $\Rightarrow$ (3.15) under Block model with Linear Dependencies.

For balanced DPIR, each message is comprised of L uniformly distributed symbols from the finite field $\mathbb{F}_q$, where $q = \lceil H(w) \rceil$. Both L and q are fixed. If assume all dependencies among messages are linear, Theorem 3.3.3 can be stated more explicitly as

LEMMA 3.1. *For balanced DPIR with linear dependencies, then the asymptotic capacity is*

$$\lim_{\mu\to\infty} C_{DPIR}(\mu, N) = \lim_{\mu\to\infty} C_{PIR}(\mu, N) = 1 - \frac{1}{N}. \tag{3.120}$$

if and only if there exists an increasing sequence $k_i \in \mathbb{N}, \forall i \in \mathbb{N}$*, such that* $\forall l \in \mathbb{N}$,

$$\lim_{\mu\to\infty} \frac{I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w)} = 0, \tag{3.121}$$

For convenience, denote $C_{\text{DPIR}}(\infty, N) = \lim_{\mu\to\infty} C_{\text{DPIR}}(\mu, N)$ and $C_{\text{PIR}}(\infty, N) = \lim_{\mu\to\infty} C_{\text{PIR}}(\mu, N)$.

The proof of (3.121) $\Rightarrow$ (3.120) is shown in 3.4.2. To prove that (3.120) $\Rightarrow$ (3.121), it suffices to assume that there does not exist an infinite subsequence of independent messages, and then based on this assumption, prove that $C_{\text{DPIR}}(\infty, N) > C_{\text{PIR}}(\infty, N) = 1 - 1/N$. So, henceforth we assume that there does not exist an infinite subsequence of independent messages. This implies that there exists a finite m, such that $\forall k \in \mathbb{N}$,

$$H(W_k | W_1, W_2, \cdots, W_m) \quad < \quad LH(w) \tag{3.122}$$

Since q, L, m are fixed, there are only finitely many different linear combinations of symbols from these m messages in $\mathbb{F}_q$. Let this number be s and denote the symbols corresponding to the distinct linear combinations as $a_1, a_2, \cdots, a_s$. Since dependencies are linear, every one of the messages $W_{m+1}, W_{m+2}, \cdots$, has at least one symbol in common with these s symbols. Partition all the messages into s disjoint groups, such that every message in the i^{th} group has the symbol a_i as one of its L symbols. Now, use PIR to privately recover one of the symbols $a_1, a_2, \cdots, a_s$ depending on whichever group the desired symbol falls in at a rate $C_{\text{PIR}}(s, N)$. Since s is finite $C_{\text{PIR}}(s, N) > C_{\text{PIR}}(\infty, N)$. For the remaining $L-1$ desired symbols, use asymptotic PIR at rate $C_{\text{PIR}}(N, \infty) = 1 - 1/N$. Thus, the rate achieved is

$$R = \frac{L}{\frac{1}{C_{\text{PIR}}(s,N)} + \frac{L-1}{C_{\text{PIR}}(\infty,N)}} \tag{3.123}$$

$$> \frac{L}{\frac{1}{C_{\text{PIR}}(\infty,N)} + \frac{L-1}{C_{\text{PIR}}(\infty,N)}} \tag{3.124}$$

$$= C_{\text{PIR}}(\infty, N) \tag{3.125}$$

Thus, we can achieve a rate strictly higher than $C(\infty, N)$ and the proof is complete.

REMARK 3.1. *Note that the proof is not limited to finite length messages. It also applies to a block extension of this setting, where each message is comprised of blocks, each block has L symbols from $\mathbb{F}_q$, dependencies exist across messages within the same block, but across blocks all realizations are i.i.d.*

3.5 Concluding Remarks

We introduced the private search problem, which requires PIR with dependent messages (DPIR). We derived a general converse bound for DPIR, studied its asymptotic behavior, and combined it with a known general achievability result in order to characterize the asymptotic capacity of various forms of private search, which include exact search, OR search, AND search, NOT search and sequence search. We also showed through an example that even asymptotic capacity characterizations for private search are difficult for additionally constrained message structures.

We note that the sufficient condition in Theorem 3.3 is applicable to T-colluding servers [105] or (N, T)-MDS coded servers $(T < N)$ [7], i.e., for DPIR with T-colluding servers or (N, T)-MDS coded servers, if there exists an increasing sequence k_i that satisfies (3.15), then the asymptotic capacity is $1 - \frac{T}{N}$. The converse proof is similar to the proof of Theorem 3.3. Following [7, 105], we can obtain a download lower bound similar to Equation (3.34),

$$\begin{aligned} D \geq & \left(1 + \frac{T}{N} + \frac{T^2}{N^2} + \cdots + \frac{T^{m-1}}{N^{m-1}}\right) LH(w) \\ & - \frac{TI(W_{k_2}; W_{k_1})}{N} - \cdots - \frac{T^{m-1} I\left(W_{k_m}; W_{k_{[1:m-1]}}\right)}{N^{m-1}}. \end{aligned} \tag{3.126}$$

With the method in Section 3.4.2, it is easily proven that $\lim_{\mu \to \infty} \sum_{l=1}^{m-1} \frac{T^l I\left(W_{k_{l+1}}; W_{k_{[1:l]}}\right)}{LH(w) N^l} = 0$. In terms of the achievable scheme, one can use the scheme of [58] and set the parameters $K_c = 1, X = 0, T = T, B = 0, U = 0$ for T-colluding servers, and set parameters $K_c = T, X = 0, T = 1, B = 0, U = 0$ for (N, T)-MDS coded servers.

One future direction is the capacity of private search over restricted search patterns discussed in section 3.4.4. Another future direction is the capacity in non-asymptotic regime. In contrast to the outer bound matching the achieving rate in the asymptotic regime, there is a gap between the outer bound and the achieving rate in the non-asymptotic regime. Take exact search with $P = 1$ as an example, when there are only $K = 2$ messages, the result is trivial because W_1 is a function of W_2. So there is no privacy for $K = 2$. Consider $K = 3$ and $N = 2$, suppose the desired message is W_1, an achievable scheme is shown in Table 3.2.

The "+" in the scheme means XOR operation. a_i notates $W_1(i)$, which is the i-th symbol of the first message. Similarly, b_i, c_i notate $W_2(i)$ and $W_3(i)$. Based on the problem setting, dependency

Table 3.2: Achievable scheme for DPIR with $K = 3$, $N = 2$

Server 1	Server 2
a_1, b_1	a_2, b_2
$a_3 + b_2$	$a_5 + b_1$
$a_4 + c_2$	$a_6 + c_2$
$b_4 + c_3$	$b_6 + c_5$
$a_7 + b_6 + c_5$	$a_8 + b_4 + c_3$

only exists among a_i, b_i, c_i. The correctness and privacy of this scheme are inherited from the achievable scheme of PIR [102] and the dependence.

In this scheme, on one hand due to the dependency among a_i, b_i, c_i, we achieve the rate $R = 0.6617$. On the other hand, according to our outer bound (3.12), $C_{PS}(3, 1, 1, 2) \leq 0.7337$. There is a gap between the achievable rate and the outer bound. Bound (3.12) is an outer bound for general DPIR problems, and Proposition 3.1 and this example show that the outer bound may not be tight for private search. To close the gap, one might need to improve the converse bound in the future. Note that for $N = 2, K = 3$, the PIR capacity is $4/7 < 0.6617$. It shows that in the non-asymptotic regime, dependency among messages can increase the capacity, which is different from that in the asymptotic case.

Chapter 4

GCSA Codes with Noise Alignment for Secure Coded Multi-Party Batch Matrix Multiplication

4.1 Introduction

Recent interest in coding for secure, private, and distributed computing combines a variety of elements such as coded distributed massive matrix multiplication, straggler tolerance, batch computing and private information retrieval [1, 4, 7, 8, 17, 27, 28, 29, 30, 31, 44, 53, 54, 55, 56, 57, 59, 61, 62, 67, 68, 70, 71, 75, 79, 90, 93, 95, 98, 100, 102, 105, 113, 115, 116, 124, 125, 126, 127, 128]. These related ideas intersected recently in Generalized Cross Subspace Alignment (GCSA) codes presented in [55]. GCSA codes originated in the setting of secure private information retrieval [59] and have recently been developed further in [55] for applications to coded distributed batch computation problems. GCSA codes generalize and improve upon the state of art distributed computing schemes such as Polynomials codes [127], MatDot codes and PolyDot codes [30], Generalized PolyDot codes [27] and Entangled Polynomial (EP) Codes [128] that partition matrices into submatrices, as well as Lagrange Coded Computing [124, 125] that allows batch processing of multiple computations.

As the next step in the expanding scope of coding for distributed computing, recently in [83] Nodehi and Maddah-Ali explored its application to secure multiparty computation [122]. Specifically, Nodehi et al. consider a system including N sources, S servers and one master. Each source sends a coded function of its data (called a share) to each server. The servers process their inputs and while doing so, may communicate with each other. After that each server sends a message to the master, such that the master can recover the required function of the source inputs. The input data must be kept perfectly secure from the servers even if up to X of the servers collude among themselves. The master must not gain any information about the input data beyond the result. Nodehi et al. propose a scheme called polynomial sharing (PS), which admits basic matrix operations such as addition and multiplication. By concatenating basic operations, arbitrary polynomial function can be calculated. The PS scheme has a few key limitations. It needs multiple rounds of communication among servers where every server needs to send messages to every other server. This is a concern because communication increases the risk for collusion. Furthermore, PS carries a high communication cost and requires the network topology among servers to be a complete graph (otherwise data security may be compromised), does not tolerate stragglers, and does not lend itself to batch processing. These aspects (batch processing, improved inter-server communication efficiency, various network topologies) are highlighted as open problems by Nodehi et al. in [83].

Since GCSA codes are particularly efficient at batch processing and already encompass prior approaches to coded distributed computing, in this chapter we explore whether GCSA codes can also be applied to the problem identified by Nodehi et al. In particular, we focus on the problem of secure multiplication of two matrices. Such a problem may arise, e.g., in correlation analysis between privately held genomic datasets to determine genetic connections without revealing anything else. As it turns out, in this context the answer is in the affirmative. Securing the data against any X colluding servers is already possible with GCSA codes as shown in [55]. The only remaining challenge is how to prevent the master from learning anything about the inputs besides the result of the computation. Let us refer to the additional terms that are contained in the answers sent by the servers to the master, which may collectively reveal information about the inputs beyond the result of the computation, as *interference* terms. To secure these interference terms, we use the idea of Noise Alignment (NA) – the workers communicate among themselves to share noise terms (unknown to the master) that are structured in the same manner as the interfering terms. Because of their matching structures, when added to the answer, the noise terms align perfectly with the interference terms and as a result no information is leaked to the master about the input data besides the result of the computation. Notably, the idea of noise alignment is not novel. While there are superficial distinctions, noise alignment is used essentially in the same manner in [129].

The combination of GCSA codes with noise alignment, GCSA-NA in short, leads to significant advantages over PS schemes. Foremost, because it uses GCSA codes, it allows the benefits of batch processing as well as straggler robustness, neither of which are available in the PS scheme of [83]. The only reason any inter-server communication is needed in a GCSA-NA scheme is to share the aligned noise terms among the servers. Since these terms do not depend on the data inputs, the inter-server communication in a GCSA-NA scheme is secure in a stronger sense than possible with PS, i.e., even if all inter-server communication is leaked, it can reveal nothing about the data inputs. The inter-server communication can take place *before* the input data is determined, say during off-peak hours. This directly leads to another advantage. The GCSA-NA scheme allows the inter-server communication network graph to be any connected graph unlike PS schemes which require a complete graph. In fact, the GCSA-NA scheme works even if inter-server communication is *entirely* disallowed, because the aligned noise can be equivalently generated by either of the source nodes and sent to the servers. By disallowing communication among servers, GCSA-NA may reduce the probability of collusion among servers relative to PS where all servers must communicate with each other.

The rest of the paper is organized as follows. Section 4.2 presents the problem statement. In Section 4.3 we state the main result and compare it with previous approaches. A toy example is presented in Section 4.4. The construction and proof of GCSA-NA are shown in Section 4.5. Section 4.6 concludes the paper.

Notation: For positive integers M, N ($M < N$), $[N]$ stands for the set $\{1, 2, \ldots, N\}$ and $[M : N]$ stands for the set $\{M, M+1, \ldots, N\}$. For a set $\mathcal{I} = \{i_1, i_2, \ldots, i_N\}$, $X_{\mathcal{I}}$ denotes the set $\{X_{i_1}, X_{i_2}, \ldots, X_{i_N}\}$. The notation $\otimes$ denotes the Kronecker product of two matrices. $\mathbf{I}_N$ denotes the $N \times N$ identity matrix. $\mathbf{T}(X_1, X_2, \cdots, X_N)$ denotes the $N \times N$ lower triangular Toeplitz matrix, i.e.,

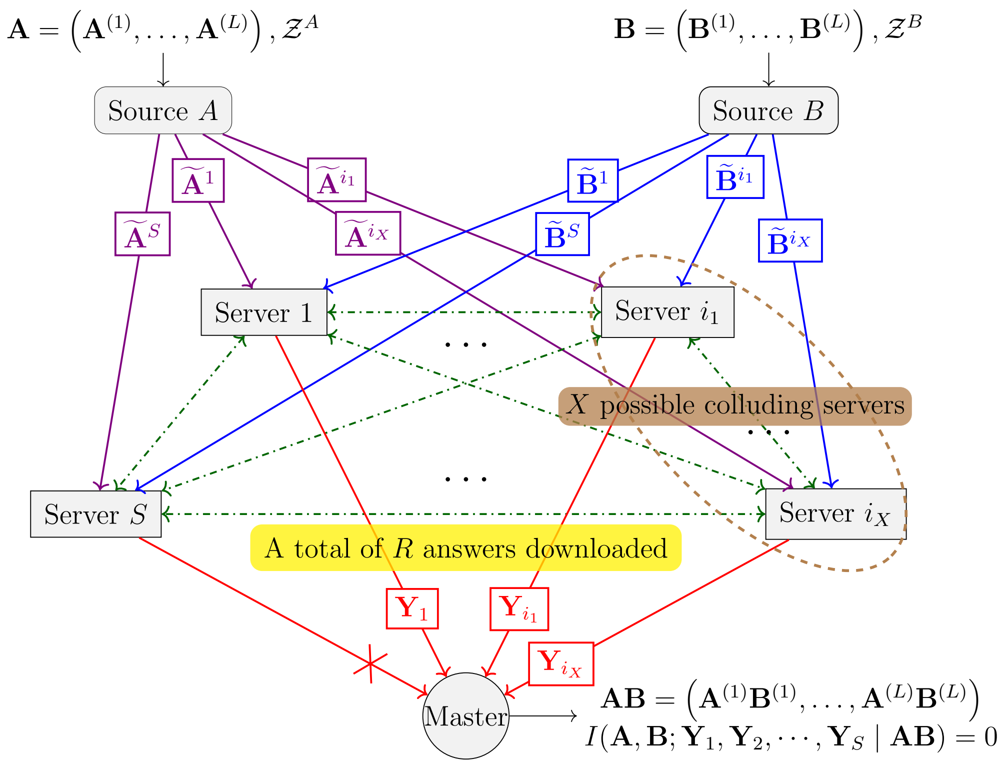
$$\mathbf{T}(X_1, X_2, \cdots, X_N) = \begin{bmatrix} X_1 & & & \\ X_2 & X_1 & & \\ \vdots & \ddots & \ddots & \\ X_N & \cdots & X_2 & X_1 \end{bmatrix}.$$

For a matrix M, $|M|$ denotes the number of elements in M. For a polynomial P, $deg_\alpha(P)$ denotes the degree with respect to a variable α. Define the degree of the zero polynomial as -1. The notation $\tilde{\mathcal{O}}(a \log^2 b)$ suppresses polylog terms for computation complexity. It may be replaced with $\mathcal{O}(a \log^2 b)$ if the field $\mathbb{F}$ supports the Fast Fourier Transform (FFT), and with $\mathcal{O}(a \log^2 b \log\log(b))$ if it does not.

4.2 Problem Statement

Consider a system including 2 sources (A and B), S servers (workers) and one master, as illustrated in Fig. 4.1. Each source is connected to every single server. Servers are connected to each other,[1]

[1]While we *allow* these links (dash-dotted lines in Figure 4.1) for the sake of consistency with the original formulation in [83], these links are not necessary for our solution. See the remark following the definition of security & strong security.



$\mathbf{A} = \left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(L)}\right), \mathcal{Z}^A$
$\mathbf{B} = \left(\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(L)}\right), \mathcal{Z}^B$
Source A
Source B
$\widetilde{\mathbf{A}}^1$
$\widetilde{\mathbf{A}}^{i_1}$
$\widetilde{\mathbf{A}}^S$
$\widetilde{\mathbf{A}}^{i_X}$
$\widetilde{\mathbf{B}}^1$
$\widetilde{\mathbf{B}}^{i_1}$
$\widetilde{\mathbf{B}}^S$
$\widetilde{\mathbf{B}}^{i_X}$
Server 1
Server i_1
Server S
Server i_X
X possible colluding servers
A total of R answers downloaded
$\mathbf{Y}_1$
$\mathbf{Y}_{i_1}$
$\mathbf{Y}_{i_X}$
Master
$\mathbf{AB} = \left(\mathbf{A}^{(1)}\mathbf{B}^{(1)}, \ldots, \mathbf{A}^{(L)}\mathbf{B}^{(L)}\right)$
$I(\mathbf{A}, \mathbf{B}; \mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_S \mid \mathbf{AB}) = 0$

Figure 4.1: The SMBMM problem. Sources generate matrices $\mathbf{A} = (\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(L)})$ with separate noise $\mathcal{Z}^A$ and $\mathbf{B} = (\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \cdots, \mathbf{B}^{(L)})$ with separate noise $\mathcal{Z}^B$, and upload information to S distributed servers in coded form $\tilde{\mathbf{A}}^{[S]}$, $\tilde{\mathbf{B}}^{[S]}$, respectively. Servers may communicate with each other via dash-dotted links. For security, any X colluding servers (e.g., Servers i_1 to i_X in the figure) learn nothing about $\mathbf{A}, \mathbf{B}$. The s^{th} server computes the answer $\mathbf{Y}_s$, which is a function of all information available to it. For effective straggler (e.g., Server S in the figure) mitigation, upon downloading answers from any R servers, where $R < S$, the master must be able to recover the product $\mathbf{AB} = (\mathbf{A}^{(1)}\mathbf{B}^{(1)}, \mathbf{A}^{(2)}\mathbf{B}^{(2)}, \ldots, \mathbf{A}^{(L)}\mathbf{B}^{(L)})$. For privacy, the master must not gain any additional information about $\mathbf{A}, \mathbf{B}$ beyond the desired product $\mathbf{AB}$.

and all of the servers are connected to the master. All of these links are secure and error free.

Source A and B independently generate sequences[2] of L matrices, denoted as $\mathbf{A} = \big(\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \dots, \mathbf{A}^{(L)}\big)$, and $\mathbf{B} = \big(\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \dots, \mathbf{B}^{(L)}\big)$, respectively, such that $\forall l \in [L]$, $\mathbf{A}^{(l)} \in \mathbb{F}^{\lambda \times \kappa}$ and $\mathbf{B}^{(l)} \in \mathbb{F}^{\kappa \times \mu}$. The master is interested in the sequence of product matrices, $\mathbf{AB} = \big(\mathbf{A}^{(1)}\mathbf{B}^{(1)}, \mathbf{A}^{(2)}\mathbf{B}^{(2)}, \dots, \mathbf{A}^{(L)}\mathbf{B}^{(L)}\big)$. The system operates in three phases: $1)$ sharing, $2)$ computation and communication, and $3)$ reconstruction.

1) Sharing: Each source encodes (encrypts) its matrices for the s^{th} server as $\tilde{\mathbf{A}}^s$ and $\tilde{\mathbf{B}}^s$, so $\tilde{\mathbf{A}}^s = f_s(\mathbf{A}, \mathcal{Z}^A), \tilde{\mathbf{B}}^s = g_s(\mathbf{B}, \mathcal{Z}^B)$, where $\mathcal{Z}^A$ and $\mathcal{Z}^B$ represent private randomness (noise) generated by the source. The encoded matrices, $\tilde{\mathbf{A}}^s, \tilde{\mathbf{B}}^s$, are sent to the s^{th} server.

2) Computation and Communication: Servers may send messages to other servers, and process what they received from both the sources and other servers. Denote the communication from Server s to Server s' as $\mathbf{M}_{s \to s'}$. Define $\mathcal{M}_s \triangleq \{\mathbf{M}_{s' \to s} \mid s' \in [S] \setminus \{s\}\}$ as the messages that Server s receives from other servers, and $\mathcal{M} \triangleq \{\mathcal{M}_s \mid s \in [S]\}$ as the total messages that all servers receive. After the communication among servers, each server s computes a response $\mathbf{Y}_s$ and sends it to the master. $\mathbf{Y}_s$ is a function of $\tilde{\mathbf{A}}^s$, $\tilde{\mathbf{B}}^s$ and $\mathcal{M}_s$, i.e., $\mathbf{Y}_s = h_s(\tilde{\mathbf{A}}^s, \tilde{\mathbf{B}}^s, \mathcal{M}_s)$, where $h_s, s \in [S]$ are the functions used to produce the answer, and we denote them collectively as $\boldsymbol{h} = (h_1, h_2, \dots, h_S)$.

3) Reconstruction: The master downloads information from servers. Some servers may fail to respond (or respond after the master executes the reconstruction), such servers are called stragglers. The master decodes the sequence of product matrices $\mathbf{AB}$ based on the information from the responsive servers, using a class of decoding functions $\boldsymbol{d} = \{d_{\mathcal{R}} \mid \mathcal{R} \subset [S]\}$ where $d_{\mathcal{R}}$ is the decoding function used when the set of responsive servers is $\mathcal{R}$.

This scheme must satisfy three constraints.

[2]The batch size L can be chosen to be arbitrarily large by the coding algorithm.

Correctness: The master must be able to recover the desired products $\mathbf{AB}$, i.e.,

$$H(\mathbf{AB} \mid \mathbf{Y}_{\mathcal{R}}) = 0, \tag{4.1}$$

or equivalently $\mathbf{AB} = d_{\mathcal{R}}(\mathbf{Y}_{\mathcal{R}})$, for some $\mathcal{R}$.

Security & Strong Security: We first define *security* which is called privacy for workers in [83]. The servers must remain oblivious to the content of the data $\mathbf{A}, \mathbf{B}$, even if X of them collude. Formally, $\forall \mathcal{X} \subset [S], |\mathcal{X}| \leq X$,

$$I(\mathbf{A}, \mathbf{B}; \tilde{\mathbf{A}}^{\mathcal{X}}, \tilde{\mathbf{B}}^{\mathcal{X}}, \mathcal{M}_{\mathcal{X}}) = 0. \tag{4.2}$$

In this paper, *strong security* is also considered. It requires that the information transmitted among servers is independent of data $\mathbf{A}, \mathbf{B}$ and all the shares $\tilde{\mathbf{A}}^{[S]}, \tilde{\mathbf{B}}^{[S]}$, i.e.,

$$I(\mathbf{A}, \mathbf{B}, \tilde{\mathbf{A}}^{[S]}, \tilde{\mathbf{B}}^{[S]}; \mathcal{M}) = 0. \tag{4.3}$$

This property makes it possible that inter-server communications happen before receiving data from sources, and makes the server communication network topology more flexible. Note that PS does not satisfy strong security because $H(\mathbf{AB} \mid \mathcal{M}) = 0$ in the PS scheme.

REMARK 4.1. *$\mathcal{M}$ can be shared among servers in various ways that satisfy strong security. For example, the servers can share $\mathcal{M}$* a-priori *during an initial setup phase, so that there is no communication among servers during the actual computation phase. Alternatively, $\mathcal{M}$ can come from a service provider whose sole purpose is to generate structured noise and send it to each server. Finally, $\mathcal{M}$ can also be separately generated by either of the source nodes (independent of the input matrices and their coded shares) and sent to each server. This only makes uses of existing communication links between the source and server nodes, and requires no communication between servers.*

Privacy: The master must not gain any additional information about $\mathbf{A}, \mathbf{B}$, beyond the required product. Precisely,

$$I(\mathbf{A}, \mathbf{B}; \mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_S \mid \mathbf{AB}) = 0. \tag{4.4}$$

This is the privacy for the master in [83].

REMARK 4.2. *There is another setting for secure distributed matrix multiplication that appears in the literature, where the input matrices originate at the master node itself [1, 17, 31, 32, 62, 67]. In that case, while the solution presented in this chapter will still apply, the privacy aspect would be irrelevant because the master already knows both* $\mathbf{A}$ *and* $\mathbf{B}$*. Correspondingly, our solution degenerates to a special case called* X*-secure GCSA codes (see Remark 2 of Appendix A in [55]). Since privacy is an important aspect of this chapter, we assume that the source nodes are distinct from the master node as shown in Figure* 4.1.

We say that $(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{d})$ form an SMBMM (**S**ecure coded **M**ulti-party **B**atch **M**atrix **M**ultiplication) code if it satisfies these three constraints. An SMBMM code is said to be r-recoverable if the master is able to recover the desired products from the answers obtained from any r servers. In particular, an SMBMM code $(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{d})$ is r-recoverable if for any $\mathcal{R} \subset [S]$, $|\mathcal{R}| = r$, and for any realization of $\mathbf{A}$, $\mathbf{B}$, we have $\mathbf{AB} = d_{\mathcal{R}}(Y_{\mathcal{R}})$. Define the recovery threshold R of an SMBMM code $(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{d})$ to be the minimum integer r such that the SMBMM code is r-recoverable.

The communication cost of an SMBMM code is comprised of these parts: upload cost of the sources, communication cost among the servers, and download cost of the master. The (normalized)[3] upload costs U_A and U_B are defined as follows.

$$U_A = \frac{\sum_{s \in [S]} |\tilde{\mathbf{A}}^s|}{L\lambda\kappa}, \quad U_B = \frac{\sum_{s \in [S]} |\tilde{\mathbf{B}}^s|}{L\kappa\mu}. \tag{4.5}$$

[3]We normalize source upload cost with the number of elements contained in the constituent matrices $\mathbf{A}, \mathbf{B}$. The server communication cost and master download cost are normalized by the number of elements contained in the desired product $\mathbf{AB}$.

Similarly, the (normalized) server communication cost CC and download cost D are defined as follows.

$$CC = \frac{\sum_{\mathbf{M}\in\mathcal{M}}|\mathbf{M}|}{L\lambda\mu}, \ \ D = \max_{\mathcal{R},\mathcal{R}\subset[S],|\mathcal{R}|=R} \frac{\sum_{s\in\mathcal{R}}|\mathbf{Y}_s|}{L\lambda\mu}. \tag{4.6}$$

Next let us consider the complexity of encoding, decoding and server computation. Define the (normalized) computational complexity at each server, $\mathcal{C}_s$, to be the order of the number of arithmetic operations required to compute the function h_s at each server, normalized by L. Similarly, define the (normalized) encoding computational complexity $\mathcal{C}_{eA}$ for $\tilde{\mathbf{A}}^{[S]}$ and $\mathcal{C}_{eB}$ for $\tilde{\mathbf{B}}^{[S]}$ as the order of the number of arithmetic operations required to compute the functions $\boldsymbol{f}$ and $\boldsymbol{g}$, respectively, each normalized by L. Finally, define the (normalized) decoding computational complexity $\mathcal{C}_d$ to be the order of the number of arithmetic operations required to compute $d_{\mathcal{R}}(\mathbf{Y}_{\mathcal{R}})$, maximized over $\mathcal{R},\mathcal{R}\subset[S],|\mathcal{R}|=R$, and normalized by L. Note that normalization by batch-size L is needed to have fair comparisons between batch processing approaches and individual matrix-partitioning solutions *per matrix multiplication*.

4.3 Main Result

Our main result appears in the following theorem.

THEOREM 4.1. *For SMBMM over a field $\mathbb{F}$ with S servers, X-security, and positive integers (ℓ, K_c, p, m, n) such that $m \mid \lambda$, $p \mid \kappa$, $n \mid \mu$ and $L = \ell K_c \leq |\mathbb{F}|-S$, the GCSA-NA scheme presented in Section* 4.5 *is a solution, and its recovery threshold, cost, and complexity are listed as follows.*

Recovery Threshold: $R = pmn(\ell+1)K_c + 2X - 1,$

Source Upload Cost of $\tilde{A}^{[S]}, \tilde{B}^{[S]}$*:* $(U_A, U_B) = \left(\frac{S}{K_c pm}, \frac{S}{K_c pn}\right),$

Server Communication Cost:$CC = \frac{S-1}{\ell K_c mn}$,

Master Download Cost:$D = \frac{R}{\ell K_c mn}$,

Source Encoding Complexity for $\tilde{A}^{[S]}, \tilde{B}^{[S]}$*:*

$$(\mathcal{C}_{eA}, \mathcal{C}_{eB}) = \left(\tilde{\mathcal{O}}\left(\frac{\lambda\kappa S\log^2 S}{K_c pm}\right), \tilde{\mathcal{O}}\left(\frac{\kappa\mu S\log^2 S}{K_c pn}\right)\right),$$

Server Computation Complexity:$\mathcal{C}_s = \mathcal{O}\left(\frac{\lambda\kappa\mu}{K_c pmn}\right)$,

Master Decoding Complexity:$\mathcal{C}_d = \tilde{\mathcal{O}}\left(\lambda\mu p\log^2 R\right)$.

The following observations place the result of Theorem 4.1 in perspective.

1. GCSA-NA codes are based on the construction of GCSA codes from [55], combined with the idea of noise-alignment (e.g., [129]). In turn, GCSA codes are based on a combination of CSA codes for batch processing [55] and EP codes for matrix partitioning [128]. CSA codes are themselves based on the idea of Cross-Subspace Alignment (CSA) that was introduced in the context of secure Private Information Retrieval (PIR) [59]. It is a remarkable coincidence that while the idea of CSA originated in the context of PIR [59], and Lagrange Coded Computing (LCC) was introduced in parallel independently in [125] for the context of coded computing, the two approaches are essentially identical, with CSA codes being slightly more powerful in the context of coded distributed matrix multiplication (CSA codes offer additional improvements over LCC codes in terms of download cost [55]). Indeed, LCC codes for batch matrix multiplication are recovered as a special case of CSA codes.

2. The idea of noise alignment can be applied to the N-CSA codes [55], for N-source secure coded multi-party batch matrix computation. In [124], Strassen's construction [99], combined with LCC, are introduced for batch distributed matrix multiplication for better computation complexity. Noise alignment is also applicable to Strassen's constructions (see Section 4.6). By setting $K_c = 1$, $\ell = L$ and $S = R$, the construction of GCSA-NA codes, with a straightforward generalization, can be further modified to settle the asymptotic (the number of messages go to infinity) capacity of

symmetric X-secure T-private computation (and also symmetric X-secure T-private information retrieval XSTPIR) [59]. However, the amount of randomness required by the construction is not necessarily optimal. For example, it is shown in [59] that by the achievable scheme for XSTPIR, symmetric security (privacy) is automatically satisfied when $T = 1$, i.e., no randomness among servers is required.

	Polynomial Sharing (PS [83])	GCSA-NA
Strong Security	No	Yes
Recovery Threshold (R)	$2pmn + 2X - 1$	$pmn(\ell + 1)K_c + 2X - 1$
Straggler Tolerance	No $(S = R)$	Yes. Tolerates $S - R$ stragglers
Server Network Topology	Complete Graph	Any Connected Graph
Source Encoding Complexity $(\mathcal{C}_{eA}, \mathcal{C}_{eB})$	$\left(\tilde{\mathcal{O}}\left(\frac{\lambda\kappa S \log^2 S}{pm}\right), \tilde{\mathcal{O}}\left(\frac{\kappa\mu S \log^2 S}{pn}\right)\right)$	$\left(\tilde{\mathcal{O}}\left(\frac{\lambda\kappa S \log^2 S}{K_c pm}\right), \tilde{\mathcal{O}}\left(\frac{\kappa\mu S \log^2 S}{K_c pn}\right)\right)$
Source Upload Cost (U_A, U_B)	$\left(\frac{S}{pm}, \frac{S}{pn}\right)$	$\left(\frac{S}{K_c pm}, \frac{S}{K_c pn}\right)$
Server Communication Cost (CC)	$\frac{S(S-1)}{mn}$	$\frac{S-1}{\ell K_c mn}$
Server Computation Complexity $(\mathcal{C}_s)$	$\mathcal{O}\left(\frac{\lambda\kappa\mu}{pmn}\right) + \mathcal{O}(\lambda\mu) + \tilde{\mathcal{O}}\left(\frac{S \log^2 S \lambda\mu}{mn}\right)$ $+\mathcal{O}\left(\frac{(S-1)\lambda\mu}{mn}\right) \approx \mathcal{O}\left(\frac{\lambda\kappa\mu}{pmn}\right)$ if $\frac{\kappa}{p} \gg S$	$\mathcal{O}\left(\frac{\lambda\kappa\mu}{K_c pmn}\right) + \mathcal{O}\left(\frac{\lambda\mu}{K_c mn}\right) + \tilde{\mathcal{O}}\left(\frac{\lambda\mu \log^2 S}{\ell K_c mn}\right)$ $\approx \mathcal{O}\left(\frac{\lambda\kappa\mu}{K_c pmn}\right)$ if $\frac{\kappa}{p} \gg S$
Master Download Cost (D)	$\frac{mn+X}{mn}$	$\frac{R}{\ell K_c mn}$
Master Decoding Complexity $(\mathcal{C}_d)$	$\tilde{\mathcal{O}}\left(\lambda\mu \log^2(mn + X)\right)$	$\tilde{\mathcal{O}}\left(\lambda\mu p \log^2(R)\right)$

Table 4.1: Performance Comparison of PS and GCSA-NA.

3. A side-by-side comparison of the GCSA-NA solution with polynomial sharing (PS) appears in Table 4.1. Because all inter-server communication is independent of input data, GCSA-NA schemes are strongly secure, i.e., even if all inter-server communication is leaked it does not compromise the security of input data. In GCSA-NA the inter-server network graph can be any connected graph. This is not possible with PS. For example, if the inter-server network graph is a star graph, then the hub server can decode $\mathbf{AB}$ by monitoring all the inter-server communication in a PS scheme, violating the security constraint. Unlike the PS scheme, in GCSA-NA, all inter-server communication can take place during off-peak hours, even before the input data is generated, giving GCSA-NA a significant latency advantage. Unlike PS where every server must communicate with every server, i.e., $S(S-1)$ such inter-server communications must take place, GCSA-NA only requires $S - 1$ inter-server communications to propagate structured noise terms across all servers. This improvement is shown numerically in Fig. 4.2a. The server computation complexity is also lower for the GCSA-NA scheme than the PS scheme. This is because in PS, each server needs to multiply the two shares received from the sources, calculate the shares for *every other server* and

sum up all the shares from *every other server*. However, in GCSA-NA, each server only needs to multiply the two shares received from the sources and add noise (which can be precomputed during off-peak hours). This advantage is particularly significant for large number of servers. The GCSA-NA scheme naturally allows robustness to stragglers, which is particularly important for massive matrix multiplications. Stragglers can be an especially significant concern for PS because of the strongly sequential nature of multi-round computation that is central to PS. This is because server failures between computation rounds disrupt the computation sequence. Remarkably, Fig. 4.2a shows that the inter-server communication cost of GCSA-NA is significantly better than PS even when GCSA-NA accommodates stragglers (while PS does not).

When restricted to batch size 1, i.e., with $\ell = K_c = 1$, GCSA-NA has the same recovery threshold as PS. Now consider batch processing, i.e., batch size $L > 1$, e.g., with $L = K_c, \ell = 1$. PS can be applied to batch processing by repeating the scheme L times. Fig. 4.2b shows that the normalized server communication cost of GCSA-NA decreases as L increases and is significantly less than that in PS. For the same number of servers S, the upload cost of GCSA-NA is smaller by a factor of $1/K_c$ compared to PS. GCSA-NA does have higher download cost and decoding complexity than PS by approximately a factor of p, which depends on how the matrices are partitioned. If p is a small value, e.g., $p = 1$, then the costs are quite similar. The improvement in download cost and decoding complexity of PS by a factor of $1/p$ comes at the penalty of increased inter-server communication cost by a factor of S. But since $S \geq R \geq 2pmn + 2X - 1 \geq p$, and typically $S \gg p$, the improvement is dominated by the penalty, so that overall the communication cost of PS is still significantly higher.

4.4 Toy Example

Let us consider a toy example with parameters $\lambda = \kappa = \mu, m = n = 1, p = 2, l = 1, K_c = 2, X = 1$ and $S = R$. Suppose matrices $\mathbf{A}, \mathbf{B} \in \mathbb{F}^{\lambda \times \lambda}$, and we wish to multiply matrix $\mathbf{A} = [\mathbf{A}_1 \ \mathbf{A}_2]$

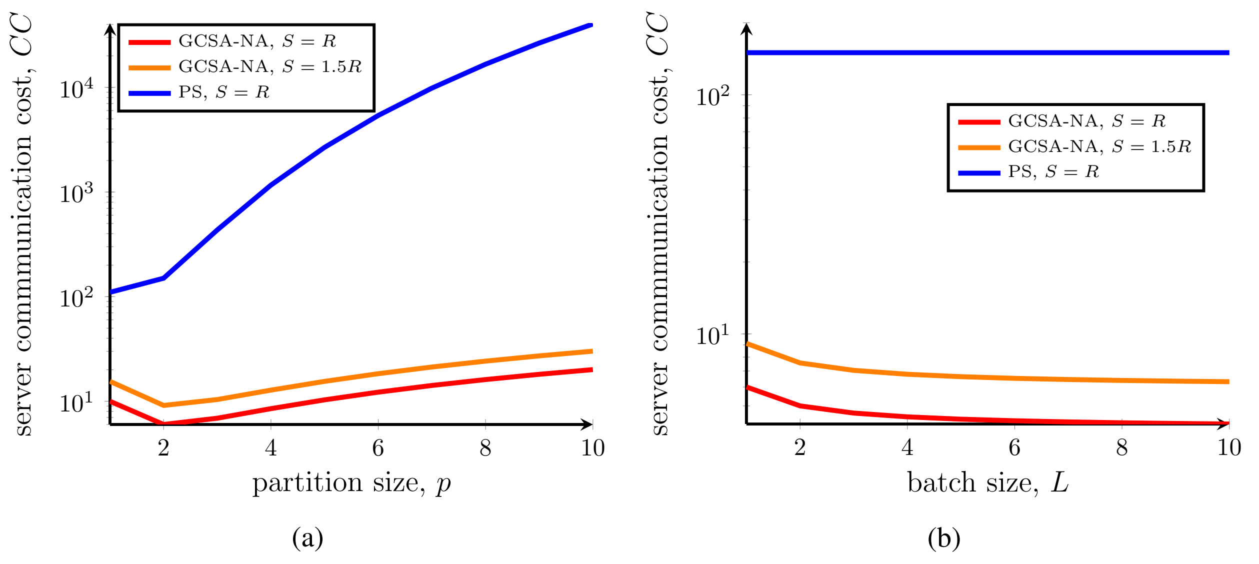


Figure 4.2: $\lambda = \kappa = \mu$, $p = m = n$. (a) Server communication cost vs. partition size, given $L = 1, X = 5$. (b) Server communication cost vs. batch size, given $p = 2, X = 5$.

with matrix $\mathbf{B} = \begin{bmatrix} \mathbf{B}_1^T \ \mathbf{B}_2^T \end{bmatrix}^T$ to compute the product $\mathbf{AB} = \mathbf{A}_1\mathbf{B}_1 + \mathbf{A}_2\mathbf{B}_2$, where $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{F}^{\lambda \times \frac{\lambda}{2}}, \mathbf{B}_1, \mathbf{B}_2 \in \mathbb{F}^{\frac{\lambda}{2} \times \lambda}$. For this toy example we summarize both the Polynomial Sharing approach [82, 83, 84], and our GCSA-NA approach.

4.4.1 Polynomial Sharing Solution

Polynomial sharing is based on EP code [128] . The given partitioning corresponds to EP code construction for $m = n = 1, p = 2$, and we have

$$\mathbf{P} = \mathbf{A}_1 + \alpha\mathbf{A}_2, \quad \mathbf{Q} = \alpha\mathbf{B}_1 + \mathbf{B}_2. \tag{4.7}$$

$$\implies \mathbf{PQ} = \mathbf{A}_1\mathbf{B}_2 + \alpha(\mathbf{A}_1\mathbf{B}_1 + \mathbf{A}_2\mathbf{B}_2) + \alpha^2\mathbf{A}_2\mathbf{B}_1. \tag{4.8}$$

To satisfy $X = 1$ security, PS includes noise with each share, i.e., $\tilde{\mathbf{A}} = \mathbf{P} + \alpha^2\mathbf{Z}^A, \tilde{\mathbf{B}} = \mathbf{Q} + \alpha^2\mathbf{Z}^B$, where $\alpha, \tilde{\mathbf{A}}, \tilde{\mathbf{B}}$ are generic variables that should be replaced with $\alpha_s, \tilde{\mathbf{A}}^s, \tilde{\mathbf{B}}^s$ for Server s, and $\alpha_1, \cdots, \alpha_S$ are distinct elements. Each server computes the product of the shares that it receives, i.e.,

$$\tilde{\mathbf{A}}\tilde{\mathbf{B}} = \mathbf{P}\mathbf{Q} + \alpha^2 P\mathbf{Z}^B + \alpha^2\mathbf{Z}^A\mathbf{Q} + \alpha^4\mathbf{Z}^A\mathbf{Z}^B \tag{4.9}$$

$$= \mathbf{A}_1\mathbf{B}_2 + \alpha(\mathbf{A}_1\mathbf{B}_1 + \mathbf{A}_2\mathbf{B}_2) + \alpha^2(\mathbf{A}_2\mathbf{B}_1 + \mathbf{A}_1\mathbf{Z}^B + \mathbf{Z}^A\mathbf{B}_2)$$
$$+ \alpha^3(\mathbf{A}_2\mathbf{Z}^B + \mathbf{Z}^A\mathbf{B}_1) + \alpha^4\mathbf{Z}^A\mathbf{Z}^B. \tag{4.10}$$

To secure inputs from the master, PS requires that every server sends to the master only the desired term $\mathbf{A}_1\mathbf{B}_1+\mathbf{A}_2\mathbf{B}_2$ by using secret sharing scheme among servers. Since $deg_\alpha(\tilde{\mathbf{A}}\tilde{\mathbf{B}}) = 4$, $\mathbf{A}_1\mathbf{B}_1+\mathbf{A}_2\mathbf{B}_2$ can be calculated from 5 distinct $\tilde{\mathbf{A}}\tilde{\mathbf{B}}$ according to the Lagrange interpolation rules. In particular, there exist 5 constants $r_1, \cdots, r_5$, such that $\mathbf{A}_1\mathbf{B}_1 + \mathbf{A}_2\mathbf{B}_2 = \sum_{s\in[5]} r_s\tilde{\mathbf{A}}^s\tilde{\mathbf{B}}^s$. Consider Server s, it sends $\mathbf{M}_{s\to j} = r_s\tilde{\mathbf{A}}^s\tilde{\mathbf{B}}^s + \alpha_j\mathbf{Z}_s$ to Server j, where $\mathbf{Z}_1, \cdots, \mathbf{Z}_5$ are i.i.d. uniform noise matrices. After Server s collects all the shares $M_{j\to s}$, it sums them up

$$\mathbf{Y}_s = \sum_{j\in[5]}\mathbf{M}_{j\to s} = \mathbf{A}_1\mathbf{B}_1 + \mathbf{A}_2\mathbf{B}_2 + \alpha_s\sum_{j\in[5]}\mathbf{Z}_j \tag{4.11}$$

and sends $\mathbf{Y}_s$ to the master. Note that after receiving $\mathbf{M}_{j\to s}$ for all $j \in [5]$, Server s still gains no information about the input data, which guarantees the security. However, it does not satisfy strong security, because $\mathbf{A}\mathbf{B}$ can be decoded based on $\mathbf{M}_{j\to s}, j, s \in [5]$.

The master can decode $\mathbf{A}\mathbf{B}$ after collecting 2 responses from servers.[4] Note that PS needs at least $S = R = 5$ servers, since 5 distinct $\tilde{\mathbf{A}}\tilde{\mathbf{B}}$ are required to obtain $\mathbf{Y}_s$.

[4]In [84], for arbitrary polynomials, $\mathbf{M}_{s\to j} = r_s\tilde{\mathbf{A}}^s\tilde{\mathbf{B}}^s + \alpha_j^2\mathbf{Z}_s$ because $\mathbf{Y}_s$ is forced to be casted in the form of entangled polynomial sharing.

4.4.2 GCSA-NA Solution

GCSA codes [55] can handle batch processing, therefore let us consider batch size 2 ($\ell = 1, K_c = 2$). Denote the second instance by $\mathbf{A}', \mathbf{B}'$. Using CSA code,

$$\mathbf{P} = \mathbf{A}_1 + (f-\alpha)\mathbf{A}_2, \quad \mathbf{Q} = (f-\alpha)\mathbf{B}_1 + \mathbf{B}_2. \tag{4.12}$$

$$\mathbf{P}' = \mathbf{A}_1' + (f'-\alpha)\mathbf{A}_2', \quad \mathbf{Q}' = (f'-\alpha)\mathbf{B}_1' + \mathbf{B}_2'. \tag{4.13}$$

And the shares are constructed as follows,

$$\tilde{\mathbf{A}} = \Delta\left(\frac{\mathbf{P}}{(f-\alpha)^2} + \frac{\mathbf{P}'}{(f'-\alpha)^2}\right), \tilde{\mathbf{B}} = \frac{\mathbf{Q}}{(f-\alpha)^2} + \frac{\mathbf{Q}'}{(f'-\alpha)^2},$$

where $\Delta = (f-\alpha)^2(f'-\alpha)^2$, and $\alpha, \tilde{\mathbf{A}}, \tilde{\mathbf{B}}$ are generic variables that should be replaced with $\alpha_s, \tilde{\mathbf{A}}^s, \tilde{\mathbf{B}}^s$ for Server s. Furthermore, $f, f', \alpha_1, \alpha_2, \cdots, \alpha_S$ are distinct elements. Each server computes the product of the shares that it receives, i.e.,

$$\begin{aligned}
\tilde{\mathbf{A}}\tilde{\mathbf{B}} &= \frac{(f'-\alpha)^2}{(f-\alpha)^2}\mathbf{P}\mathbf{Q} + \frac{(f-\alpha)^2}{(f'-\alpha)^2}\mathbf{P}'\mathbf{Q}' + \mathbf{P}'\mathbf{Q} + \mathbf{P}\mathbf{Q}' && (4.14)\\
&= \frac{((f'-f)+(f-\alpha))^2}{(f-\alpha)^2}\mathbf{P}\mathbf{Q} + \frac{((f-f')+(f'-\alpha))^2}{(f'-\alpha)^2}\mathbf{P}'\mathbf{Q}'\\
&\quad + \mathbf{P}'\mathbf{Q} + \mathbf{P}\mathbf{Q}' && (4.15)\\
&= \frac{(f'-f)^2 + 2(f'-f)(f-\alpha) + (f-\alpha)^2}{(f-\alpha)^2}\mathbf{P}\mathbf{Q}\\
&\quad + \frac{(f-f')^2 + 2(f-f')(f'-\alpha) + (f'-\alpha)^2}{(f'-\alpha)^2}\mathbf{P}'\mathbf{Q}'\\
&\quad + \mathbf{P}'\mathbf{Q} + \mathbf{P}\mathbf{Q}' && (4.16)\\
&= \frac{(f'-f)^2}{(f-\alpha)^2}\mathbf{P}\mathbf{Q} + \frac{2(f'-f)}{f-\alpha}\mathbf{P}\mathbf{Q} + \frac{(f-f')^2}{(f'-\alpha)^2}\mathbf{P}'\mathbf{Q}'\\
&\quad + \frac{2(f-f')}{f'-\alpha}\mathbf{P}'\mathbf{Q}' + \mathbf{P}\mathbf{Q} + \mathbf{P}'\mathbf{Q}' + \mathbf{P}'\mathbf{Q} + \mathbf{P}\mathbf{Q}' && (4.17)\\
&= \frac{c_0}{(f-\alpha)^2}\mathbf{P}\mathbf{Q} + \frac{c_1}{f-\alpha}\mathbf{P}\mathbf{Q} + \frac{c_0'}{(f'-\alpha)^2}\mathbf{P}'\mathbf{Q}'\\
&\quad + \frac{c_1'}{f'-\alpha}\mathbf{P}'\mathbf{Q}' + \mathbf{I}_0 + \alpha\mathbf{I}_1 + \alpha^2\mathbf{I}_2 && (4.18)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{c_0\mathbf{A}_1\mathbf{B}_2}{(f-\alpha)^2} + \frac{c_0\mathbf{A}_1\mathbf{B}_1 + c_0\mathbf{A}_2\mathbf{B}_2 + c_1\mathbf{A}_1\mathbf{B}_2}{f-\alpha} + \frac{c_0'\mathbf{A}_1'\mathbf{B}_2'}{(f'-\alpha)^2} \\
&\quad + \frac{c_0'\mathbf{A}_1'\mathbf{B}_1' + c_0'\mathbf{A}_2'\mathbf{B}_2' + c_1'\mathbf{A}_1'\mathbf{B}_2'}{f'-\alpha} + \mathbf{I}_0 + \alpha\mathbf{I}_1 + \alpha^2\mathbf{I}_2, \qquad (4.19)
\end{aligned}
$$

where $\mathbf{I}_0, \mathbf{I}_1, \mathbf{I}_2$ are various linear combinations of $\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_1, \mathbf{B}_2, \mathbf{A}'_1, \mathbf{A}'_2, \mathbf{B}'_1, \mathbf{B}'_2$ and c_0, c_1, c_0', c_1' are constants. Their exact forms can be found by performing partial fraction decomposition. This is the original GCSA code [55], and we need $R = pmn((\ell+1)K_c - 1) + p - 1 = 7$ responses to recover the desired product.

Next, let us modify the scheme to make it $X = 1$ secure by including noise with each share, i.e.,

$$
\begin{aligned}
\tilde{\mathbf{A}} &= \Delta\left(\frac{\mathbf{P}}{(f-\alpha)^2} + \frac{\mathbf{P}'}{(f'-\alpha)^2} + \mathbf{Z}^A\right), \\
\tilde{\mathbf{B}} &= \frac{\mathbf{Q}}{(f-\alpha)^2} + \frac{\mathbf{Q}'}{(f'-\alpha)^2} + \mathbf{Z}^B. \\
\tilde{\mathbf{A}}\tilde{\mathbf{B}} &= \frac{c_0\mathbf{P}\mathbf{Q}}{(f-\alpha)^2} + \frac{c_1\mathbf{P}\mathbf{Q}}{f-\alpha} + \frac{c_0'\mathbf{P}'\mathbf{Q}'}{(f'-\alpha)^2} + \frac{c_1'\mathbf{P}'\mathbf{Q}'}{f'-\alpha} + \sum_{i=0}^{4}\alpha^i\mathbf{I}_i.
\end{aligned}
$$

As a result of the added noise terms, the recovery threshold is now increased to 9. Note that the term $\mathbf{I}_4$ contains only contributions from $\Delta\mathbf{Z}^A\mathbf{Z}^B$, i.e., this term leaks no information about $\mathbf{A}, \mathbf{B}$ matrices.

If the servers directly return their computed values of $\tilde{\mathbf{A}}\tilde{\mathbf{B}}$ to the master, then besides the result of the computation some additional information about the input matrices $\mathbf{A}, \mathbf{B}$ may be leaked by the interference terms $\left(\frac{c_0}{(f-\alpha)^2} + \frac{c_1}{f-\alpha}\right)\mathbf{A}_1\mathbf{B}_2 + \left(\frac{c_0'}{(f'-\alpha)^2} + \frac{c_1'}{f'-\alpha}\right)\mathbf{A}_1'\mathbf{B}_2' + \sum_{i=0}^{3}\alpha^i\mathbf{I}_i$, which can be secured by the addition of *aligned noise* terms $\tilde{\mathbf{Z}} = \left(\frac{c_0}{(f-\alpha)^2} + \frac{c_1}{f-\alpha}\right)\mathbf{Z} + \left(\frac{c_0'}{(f'-\alpha)^2} + \frac{c_1'}{f'-\alpha}\right)\mathbf{Z}' + \sum_{i=0}^{3}\alpha^i\mathbf{Z}_i$ at each server so that the answer returned by each server to the master is $\tilde{\mathbf{A}}\tilde{\mathbf{B}} + \tilde{\mathbf{Z}}$. Here $\mathbf{Z}, \mathbf{Z}', \mathbf{Z}_0, \mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$ are i.i.d. uniform noise matrices, that can all be privately generated by one server, who can then share their aligned form $\tilde{\mathbf{Z}}$ with all other servers. This sharing of $\tilde{\mathbf{Z}}$ is the only inter-server communication needed in GCSA-NA. Since it is independent of the inputs, it can be done during off-peak hours, thereby reducing the latency of server computation. The strong security is also automatically satisfied.

4.5 Construction of GCSA-NA

Now let us present the general construction. $L = \ell K_c$ instances of $\mathbf{A}$ and $\mathbf{B}$ matrices are split into ℓ groups. $\forall l \in [\ell], \forall k \in [K_c]$, denote

$$\mathbf{A}^{l,k} = \mathbf{A}^{(K_c(l-1)+k)},\ \mathbf{B}^{l,k} = \mathbf{B}^{(K_c(l-1)+k)}. \tag{4.20}$$

Further, each matrix $\mathbf{A}^{l,k}$ is partitioned into $m \times p$ blocks and each matrix $\mathbf{B}^{l,k}$ is partitioned into $p \times n$ blocks, i.e.,

$$\mathbf{A}^{l,k} = \begin{bmatrix} \mathbf{A}^{l,k}_{1,1} & \cdots & \mathbf{A}^{l,k}_{1,p} \\ \mathbf{A}^{l,k}_{2,1} & \cdots & \mathbf{A}^{l,k}_{2,p} \\ \vdots & \vdots & \vdots \\ \mathbf{A}^{l,k}_{m,1} & \cdots & \mathbf{A}^{l,k}_{m,p} \end{bmatrix}, \mathbf{B}^{l,k} = \begin{bmatrix} \mathbf{B}^{l,k}_{1,1} & \cdots & \mathbf{B}^{l,k}_{1,n} \\ \mathbf{B}^{l,k}_{2,1} & \cdots & \mathbf{B}^{l,k}_{2,n} \\ \vdots & \vdots & \vdots \\ \mathbf{B}^{l,k}_{p,1} & \cdots & \mathbf{B}^{l,k}_{p,n} \end{bmatrix},$$

where $\left(\mathbf{A}^{l,k}_{i,j}\right)_{i\in[m],j\in[p]} \in \mathbb{F}^{\frac{\lambda}{m}\times\frac{\kappa}{p}}$ and $\left(\mathbf{B}^{l,k}_{i,j}\right)_{i\in[m],j\in[p]} \in \mathbb{F}^{\frac{\kappa}{p}\times\frac{\mu}{n}}$.

Let $f_{1,1}, f_{1,2}, \cdots, f_{\ell,K_c}, \alpha_1, \alpha_2, \cdots, \alpha_S$ be $(S+L)$ distinct elements from the field $\mathbb{F}$. For convenience, define

$$R' = pmn, \quad D_E = \max(pm, pmn - pm + p) - 1, \tag{4.21}$$

$$\mathcal{E} = \{p + p(m'-1) + pm(n''-1) \mid m' \in [m], n'' \in [n]\}, \tag{4.22}$$

$$\Delta_s^{l,K_c} = \prod_{k\in[K_c]} (f_{l,k} - \alpha_s)^{R'}, \forall l \in [\ell], \forall s \in [S]. \tag{4.23}$$

Define $c_{l,k,i}, i \in \{0, 1, \cdots, R'(K_c-1)\}$ to be the coefficients satisfying

$$\Psi_{l,k}(\alpha) = \prod_{k'\in[K_c]\backslash\{k\}} (\alpha + (f_{l,k'} - f_{l,k}))^{R'}$$

$$= \sum_{i=0}^{R'(K_c-1)} c_{l,k,i}\alpha^i, \forall l \in [\ell], \forall k \in [K_c], \tag{4.24}$$

i.e., they are the coefficients of the polynomial $\Psi_{l,k}(\alpha) = \prod_{k' \in [K_c]\backslash\{k\}} (\alpha + (f_{l,k'} - f_{l,k}))^{R'}$, which is defined by its roots. Note that all the coefficients $(c_{l,k,i})_{l\in[L],k\in[K_c],i\in\{0,1,\cdots,R'(K_c-1)\}}$, $\alpha_{[S]}$, $(f_{l,k})_{l\in[L],k\in[K]}$ are globally known.

4.5.1 Sharing

Firstly, each source encodes each constituent matrix blocks $\mathbf{A}^{l,k}$ and $\mathbf{B}^{l,k}$ with Entangled Polynomial code [128]. For all $l \in [\ell], k \in [K_c]$, define

$$\mathbf{P}_s^{l,k} = \sum_{m'\in[m]} \sum_{p'\in[p]} \mathbf{A}_{m',p'}^{l,k} (f_{l,k} - \alpha_s)^{p'-1+p(m'-1)}, \tag{4.25}$$

$$\mathbf{Q}_s^{l,k} = \sum_{p''\in[p]} \sum_{n''\in[n]} \mathbf{B}_{p'',n''}^{l,k} (f_{l,k} - \alpha_s)^{p-p''+pm(n''-1)}. \tag{4.26}$$

Note that the original Entangled Polynomial code can be regarded as polynomials of α_s, and here for each (l,k), Entangled Polynomial code is constructed as polynomials of $(f_{l,k} - \alpha_s)$.

Each source generates ℓX independent random matrices, $\mathcal{Z}^A = \left\{\mathbf{Z}_{1,1}^A, \cdots, \mathbf{Z}_{\ell,X}^A\right\}$ and $\mathcal{Z}^B = \left\{\mathbf{Z}_{1,1}^B, \cdots, \mathbf{Z}_{\ell,X}^B\right\}$. The independence is established as follows.

$$\begin{aligned} H(\mathcal{Z}^A, \mathcal{Z}^B, \mathbf{A}, \mathbf{B}) &= H(\mathbf{A}) + H(\mathbf{B}) \\ &+ \sum_{l\in[\ell],x\in[X]} H\left(\mathbf{Z}_{l,x}^A\right) + \sum_{l\in[\ell],x\in[X]} H\left(\mathbf{Z}_{l,x}^B\right). \end{aligned} \tag{4.27}$$

For all $s \in [S]$, the shares of matrices $\mathbf{A}$ and $\mathbf{B}$ at the s^{th} server are constructed as $\tilde{\mathbf{A}}^s =$

$\left(\tilde{\mathbf{A}}_1^s, \tilde{\mathbf{A}}_2^s, \ldots, \tilde{\mathbf{A}}_\ell^s\right), \tilde{\mathbf{B}}^s = \left(\tilde{\mathbf{B}}_1^s, \tilde{\mathbf{B}}_2^s, \ldots, \tilde{\mathbf{B}}_\ell^s\right)$, where for all $l \in [\ell]$,

$$\tilde{\mathbf{A}}_l^s = \Delta_s^{l,K_c} \left(\sum_{k\in[K_c]} \frac{\mathbf{P}_s^{l,k}}{(f_{l,k} - \alpha_s)^{R'}} + \sum_{x\in[X]} \alpha_s^{x-1} \mathbf{Z}_{l,x}^A \right),$$
$$\tilde{\mathbf{B}}_l^s = \sum_{k\in[K_c]} \frac{\mathbf{Q}_s^{l,k}}{(f_{l,k} - \alpha_s)^{R'}} + \sum_{x\in[X]} \alpha_s^{x-1} \mathbf{Z}_{l,x}^B.$$

Then each pair of shares $\tilde{\mathbf{A}}^s, \tilde{\mathbf{B}}^s$ is sent to the corresponding server.

4.5.2 Computation and Communication

One of the servers generates a set of $\frac{\lambda}{m} \times \frac{\mu}{n}$ matrices $\mathcal{Z}^{server}$, which contains $R'(K_c - 1) + X + D_E + \ell K_c(p-1)mn$ independent random matrices and $\ell K_c mn$ zero matrices. In particular, $\mathcal{Z}^{server} = \{\mathcal{Z}_1^{server}, \mathcal{Z}_2^{server}\}$, $\mathcal{Z}_1^{server} = \{\mathbf{Z}_i' \mid i \in [R'(K_c-1) + X + D_E]\}$, and $\mathcal{Z}_2^{server} = \left\{\mathbf{Z}_{l,k,i}'' \mid l \in [\ell], k \in [K_c], i \in [R']\right\}$. Here,

$$\mathbf{Z}_{l,k,i}'' = \begin{cases} \mathbf{0}, & \text{if } i \in \mathcal{E} \\ \mathbf{Z}_{l,k,i}''', & \text{otherwise,} \end{cases} \quad \forall l \in [\ell], \forall k \in [K_c].$$

Here $\mathbf{Z}_i'$ and $\mathbf{Z}_{l,k,i}'''$ are the independent random matrices. The independence is established as follows.

$$H(\mathcal{Z}^{server}, \mathbf{A}, \mathbf{B}) = H(\mathbf{A}) + H(\mathbf{B}) + \sum_{i=1}^{R'(K_c-1)+X+D_E} H(\mathbf{Z}_i') + \sum_{\substack{l\in[\ell],k\in[K_c],\\ i\in[R']}} H(\mathbf{Z}_{l,k,i}''). \tag{4.28}$$

Without loss of generality, assume the first server generates $\mathcal{Z}^{server}$, encodes them into

$$\tilde{\mathbf{M}}_s = \sum_{x=1}^{R'(K_c-1)+X+D_E} \alpha_s^{x-1}\mathbf{Z}'_x + \sum_{l\in[\ell]}\sum_{k\in[K_c]}\sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k.i-i'}\mathbf{Z}''_{l,k,i'+1}}{(f_{l,k}-\alpha_s)^{R'-i}} \tag{4.29}$$

and sends $\tilde{\mathbf{M}}_s$ to server $s, s \in [S]\setminus\{1\}$, where $c_{l,k,i}$ is defined in (4.24). The answer returned by the s^{th} server to the master is constructed as $\mathbf{Y}_s = \sum_{l\in[\ell]} \tilde{\mathbf{A}}_l^s \tilde{\mathbf{B}}_l^s + \tilde{\mathbf{M}}_s$.

4.5.3 Reconstruction

After the master collects any R answers, it decodes the desired products $\mathbf{AB}$.

4.5.4 Proof of Theorem 1

To begin, let us recall the standard result for Confluent Cauchy-Vandermonde matrices [39], replicated here for the sake of completeness.

LEMMA 4.1. *If $f_{1,1}, f_{1,2}, \cdots, f_{\ell,K_c}, \alpha_1, \alpha_2, \cdots, \alpha_R$ are $R+L$ distinct elements of $\mathbb{F}$, with $|\mathbb{F}|\geq R+L$, $L=\ell K_c$ and $R = R'(\ell+1)K_c + 2X - 1$, then the $R\times R$ Confluent Cauchy-Vandermonde matrix* (4.30) *is invertible over $\mathbb{F}$.*

$$\hat{\mathbf{V}}_{\ell,K_c,R',X,R} \triangleq \begin{bmatrix} \frac{1}{(f_{1,1}-\alpha_1)^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_1} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_1)^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_1} & 1 & \cdots & \alpha_1^{R'K_c+2X-2} \\ \frac{1}{(f_{1,1}-\alpha_2)^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_2} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_2)^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_2} & 1 & \cdots & \alpha_2^{R'K_c+2X-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{(f_{1,1}-\alpha_R)^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_R} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_R)^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_R} & 1 & \cdots & \alpha_R^{R'K_c+2X-2} \end{bmatrix} \tag{4.30}$$

Firstly, let us prove that the GCSA-NA codes are $R = pmn(\ell+1)K_c + 2X - 1$ recoverable. Rewrite $\mathbf{Y}_s$ as follows.

$$\mathbf{Y}_s = \tilde{\mathbf{A}}_1^s\tilde{\mathbf{B}}_1^s + \tilde{\mathbf{A}}_2^s\tilde{\mathbf{B}}_2^s + \cdots + \tilde{\mathbf{A}}_\ell^s\tilde{\mathbf{B}}_\ell^s + \tilde{\mathbf{M}}_s \tag{4.31}$$

$$= \sum_{l\in[\ell]} \Delta_s^{l,K_c} \left(\sum_{k\in[K_c]} \frac{\mathbf{P}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} + \sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{A} \right)$$

$$\cdot \left(\sum_{k\in[K_c]} \frac{\mathbf{Q}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} + \sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{B} \right) + \tilde{\mathbf{M}}_s \tag{4.32}$$

$$= \sum_{l\in[\ell]} \Delta_s^{l,K_c} \left(\sum_{k\in[K_c]} \frac{\mathbf{P}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} \right) \left(\sum_{k\in[K_c]} \frac{\mathbf{Q}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} \right)$$

$$+ \underbrace{\sum_{l\in[\ell]} \Delta_s^{l,K_c} \left(\sum_{k\in[K_c]} \frac{\mathbf{P}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} \right) \left(\sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{B} \right)}_{\mathbf{\Gamma}_2}$$

$$+ \underbrace{\sum_{l\in[\ell]} \Delta_s^{l,K_c} \left(\sum_{k\in[K_c]} \frac{\mathbf{Q}_s^{l,k}}{(f_{l,k}-\alpha_s)^{R'}} \right) \left(\sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{A} \right)}_{\mathbf{\Gamma}_3}$$

$$+ \underbrace{\sum_{l\in[\ell]} \Delta_s^{l,K_c} \left(\sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{A} \right) \left(\sum_{x\in[X]} \alpha_s^{x-1}\mathbf{Z}_{l,x}^{B} \right)}_{\mathbf{\Gamma}_4} + \tilde{\mathbf{M}}_s \tag{4.33}$$

$$= \sum_{l\in[\ell]} \sum_{k\in[K_c]} \frac{\prod_{k'\in[K_c]\setminus\{k\}} (f_{l,k'}-\alpha_s)^{R'}}{(f_{l,k}-\alpha_s)^{R'}} \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k}$$

$$+ \underbrace{\sum_{l\in[\ell]} \sum_{\substack{k,k'\in[K_c]\\ k\neq k'}} \left(\prod_{k''\in[K_c]\setminus\{k,k'\}} (f_{l,k''}-\alpha_s)^{R'} \right) \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k'}}_{\mathbf{\Gamma}_1}$$

$$+ \mathbf{\Gamma}_2 + \mathbf{\Gamma}_3 + \mathbf{\Gamma}_4 + \tilde{\mathbf{M}}_s. \tag{4.34}$$

Consider the first term in (4.34). For each $l \in [\ell], k \in [K_c]$, we have

$$\frac{\prod_{k'\in[K_c]\setminus\{k\}} (f_{l,k'}-\alpha_s)^{R'}}{(f_{l,k}-\alpha_s)^{R'}} \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k}$$

$$= \frac{\prod_{k'\in[K_c]\setminus\{k\}} \left((f_{l,k}-\alpha_s) + (f_{l,k'}-f_{l,k})\right)^{R'}}{(f_{l,k}-\alpha_s)^{R'}} \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k} \tag{4.35}$$

$$= \frac{\Psi_{l,k}(f_{l,k}-\alpha_s)}{(f_{l,k}-\alpha_s)^{R'}} \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k} \tag{4.36}$$

$$= \left(\frac{c_{l,k,0}}{(f_{l,k}-\alpha_s)^{R'}} + \frac{c_{l,k,1}}{(f_{l,k}-\alpha_s)^{R'-1}} + \cdots + \frac{c_{l,k,R'-1}}{f_{l,k}-\alpha_s} \right) \mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k}$$

$$+\underbrace{\left(\sum_{i=R'}^{R'(K_c-1)} c_{l,k,i}(f_{l,k}-\alpha_s)^{i-R'}\right)\mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k}}_{\mathbf{\Gamma}_5}, \tag{4.37}$$

where (4.36) results from the definition of $\Psi_{l,k}(\cdot)$ as in (4.24) and in (4.37) the polynomial $\Psi_{l,k}(f_{l,k}-\alpha_s)$ is rewritten in terms of its coefficients.

By the construction of Entangled Polynomial code (4.25) (4.26), the product $\mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k}$ can be written as weighted sums of the terms $1, (f_{l,k}-\alpha_s), \cdots, (f_{l,k}-\alpha_s)^{R'+p-2}$, i.e.,

$$\mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k} = \sum_{i=0}^{R'+p-2} \mathbf{C}_{i+1}^{l,k}(f_{l,k}-\alpha_s)^i, \tag{4.38}$$

where $\mathbf{C}_1^{l,k}, \mathbf{C}_2^{l,k}, \cdots, \mathbf{C}_{R'+p-1}^{l,k}$ are various linear combinations of products of blocks of $\mathbf{A}^{l,k}$ and blocks of $\mathbf{B}^{l,k}$. Consider the first term in (4.37).

$$\begin{aligned}
&\left(\frac{c_{l,k,0}}{(f_{l,k}-\alpha_s)^{R'}}+\cdots+\frac{c_{l,k,R'-1}}{f_{l,k}-\alpha_s}\right)\mathbf{P}_s^{l,k}\mathbf{Q}_s^{l,k} \\
\overset{(4.38)}{=}&\left(\frac{c_{l,k,0}}{(f_{l,k}-\alpha_s)^{R'}}+\cdots+\frac{c_{l,k,R'-1}}{f_{l,k}-\alpha_s}\right)\sum_{i=0}^{R'+p-2}\mathbf{C}_{i+1}^{l,k}(f_{l,k}-\alpha_s)^i \qquad (4.39)\\
=&\sum_{i=0}^{R'-1}\frac{\sum_{i'=0}^{i} c_{l,k,i-i'}\mathbf{C}_{i'+1}^{l,k}}{(f_{l,k}-\alpha_s)^{R'-i}} \\
&+\underbrace{\sum_{i=0}^{p-2}(f_{l,k}-\alpha_s)^i\left(\sum_{i'=i+1}^{R'+i} c_{l,k,R'-i'+i}\mathbf{C}_{i'+1}^{l,k}\right)}_{\mathbf{\Gamma}_6} \\
&+\underbrace{\sum_{i=p-1}^{R'+p-3}(f_{l,k}-\alpha_s)^i\left(\sum_{i'=i+1}^{R'+p-2} c_{l,k,R'-i'+i}\mathbf{C}_{i'+1}^{l,k}\right)}_{\mathbf{\Gamma}_7}. \qquad (4.40)
\end{aligned}$$

Note that if $K_c = 1, \forall i \neq 0, c_{l,k,i} = 0$, then $\mathbf{\Gamma}_5$ and $\mathbf{\Gamma}_7$ are zero polynomials. Now let us consider the degree with respect to α_s of $\mathbf{\Gamma}_1, \cdots, \mathbf{\Gamma}_7$.

$$deg_{\alpha_s}(\mathbf{\Gamma}_1) = \begin{cases} R'(K_c-1)+p-2, & \text{if } K_c > 1 \\ -1, & \text{otherwise} \end{cases},$$

$$deg_{\alpha_s}(\mathbf{\Gamma}_2) = R'(K_c-1)+pm+X-2,$$

$$deg_{\alpha_s}(\mathbf{\Gamma}_3) = R'(K_c-1)+pmn-pm+p+X-2,$$

$$deg_{\alpha_s}(\mathbf{\Gamma}_4) = R'K_c+2X-2, \;\; deg_{\alpha_s}(\mathbf{\Gamma}_6) = p-2,$$

$$deg_{\alpha_s}(\mathbf{\Gamma}_5) = \begin{cases} R'(K_c-1)+p-2, & \text{if } K_c > 1 \\ -1, & \text{otherwise} \end{cases},$$

$$deg_{\alpha_s}(\mathbf{\Gamma}_7) = \begin{cases} R'+p-3, & \text{if } K_c > 1 \\ -1, & \text{otherwise} \end{cases}.$$

Recall X, p, m, n, K_c are positive integers. If $K_c > 1$, it is easy to see that $R'K_c + 2X - 2$ is the largest. If $K_c = 1$, $R' = pmn \geq p > p-2$, $R'K_c + 2X - 2$ is also the largest. Therefore the sum of $\Gamma_1, \cdots, \Gamma_7$ can be expanded into weighted sums of the terms $1, \alpha_s, \cdots, \alpha_s^{R'K_c+2X-2}$. Note that the weights of terms $\alpha_s^{R'(K_c-1)+X+D_E+1}, \cdots, \alpha_s^{R'K_c+2X-2}$ are functions of $\mathcal{Z}^A, \mathcal{Z}^B$. $\mathbf{Y}_s$ can be rewritten as

$$\mathbf{Y}_s = \sum_{l\in[\ell]} \sum_{k\in[K_c]} \sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k,i-i'} \mathbf{C}_{i'+1}^{l,k}}{(f_{l,k}-\alpha_s)^{R'-i}} + \sum_{x=1}^{R'K_c+2X-1} \alpha_s^{x-1}\mathbf{I}_x + \tilde{\mathbf{M}}_s \tag{4.41}$$

$$\begin{aligned} &\overset{(4.29)}{=} \sum_{l\in[\ell]} \sum_{k\in[K_c]} \sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k,i-i'} \mathbf{C}_{i'+1}^{l,k}}{(f_{l,k}-\alpha_s)^{R'-i}} \\ &+ \sum_{x=1}^{R'K_c+2X-1} \alpha_s^{x-1}\mathbf{I}_x + \sum_{x=1}^{R'(K_c-1)+X+D_E} \alpha_s^{x-1}\mathbf{Z}'_x \\ &+ \sum_{l\in[\ell]} \sum_{k\in[K_c]} \sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k.i-i'} \mathbf{Z}''_{l,k,i'+1}}{(f_{l,k}-\alpha_s)^{R'-i}} \end{aligned} \tag{4.42}$$

$$
\begin{aligned}
&= \sum_{l\in[\ell]} \sum_{k\in[K_c]} \sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k,i-i'} \left(\mathbf{C}_{i'+1}^{l,k} + \mathbf{Z}''_{l,k,i'+1}\right)}{(f_{l,k}-\alpha_s)^{R'-i}} \\
&\quad + \sum_{x=1}^{R'(K_c-1)+X+D_E} \alpha_s^{x-1}\left(\mathbf{I}_x + \mathbf{Z}'_x\right) + \sum_{x=R'(K_c-1)+X+D_E+1}^{R'K_c+2X-1} \alpha_s^{x-1}\mathbf{I}_x \qquad (4.43) \\
&= \sum_{l\in[\ell]} \sum_{k\in[K_c]} \sum_{i=0}^{R'-1} \frac{\sum_{i'=0}^{i} c_{l,k,i-i'}\mathbf{D}_{i'+1}^{l,k}}{(f_{l,k}-\alpha_s)^{R'-i}} + \sum_{x\in[R'K_c+2X-1]} \alpha_s^{x-1}\mathbf{J}_x, \qquad (4.44)
\end{aligned}
$$

where $\mathbf{D}_i^{l,k} = \mathbf{C}_i^{l,k} + \mathbf{Z}''_{l,k,i}, l \in [\ell], k \in [K_c], i \in [R'], \mathbf{J}_x = \mathbf{I}_x + \mathbf{Z}'_x, x \in [R'(K_c-1)+X+D_E]$ and $\mathbf{J}_x = \mathbf{I}_x, x \in [R'(K_c-1)+X+D_E+1 : R'K_c+2X-1]$. In the matrix form, answers from any $R = R'K_c + 2X - 1 + R'L = pmn(\ell+1)K_c + 2X - 1$ servers, whose indices are denoted as $s_1, s_2, \cdots, s_R$, can be written as (4.45)

$$
\begin{bmatrix} \mathbf{Y}_{s_1} \\ \mathbf{Y}_{s_2} \\ \vdots \\ \mathbf{Y}_{s_R} \end{bmatrix}
= \underbrace{\left[\begin{array}{ccc|c|ccc|ccc}
\frac{1}{(f_{1,1}-\alpha_{s_1})^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_{s_1}} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_{s_1})^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_{s_1}} & 1 & \cdots & \alpha_{s_1}^{R'K_c+2X-2} \\
\frac{1}{(f_{1,1}-\alpha_{s_2})^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_{s_2}} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_{s_2})^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_{s_2}} & 1 & \cdots & \alpha_{s_2}^{R'K_c+2X-2} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{1}{(f_{1,1}-\alpha_{s_R})^{R'}} & \cdots & \frac{1}{f_{1,1}-\alpha_{s_R}} & \cdots & \frac{1}{(f_{\ell,K_c}-\alpha_{s_R})^{R'}} & \cdots & \frac{1}{f_{\ell,K_c}-\alpha_{s_R}} & 1 & \cdots & \alpha_{s_R}^{R'K_c+2X-2}
\end{array}\right]}_{\hat{\mathbf{V}}_{\ell,K_c,R',X,R}}
$$

$$
\underbrace{\left[\begin{array}{c|c|c|c}
\mathbf{T}(c_{1,1,0},\cdots,c_{1,1,R'-1}) & & & \\ \hline
 & \ddots & & \\ \hline
 & & \mathbf{T}(c_{\ell,K_c,0},\cdots,c_{\ell,K_c,R'-1}) & \\ \hline
 & & & \mathbf{I}_{R-R'L}
\end{array}\right]}_{\hat{\mathbf{V}}'_{\ell,K_c,R',X,R}} \otimes \mathbf{I}_{\lambda/m}
\left[\begin{array}{c}
\mathbf{D}_1^{1,1} \\ \vdots \\ \mathbf{D}_{R'}^{1,1} \\ \hline \vdots \\ \hline \mathbf{D}_1^{\ell,K_c} \\ \vdots \\ \mathbf{D}_{R'}^{\ell,K_c} \\ \hline \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_{R'K_c+2X-1}
\end{array}\right]. \qquad (4.45)
$$

Since $f_{1,1}, f_{1,2}, \cdots, f_{\ell,K_c}$ are distinct, for all $l \in [\ell], k \in [K_c], c_{l,k,0} = \prod_{k'\in[K_c]\setminus\{k\}}(f_{l,k'} - f_{l,k})^{R'}$ are non-zero. Hence, the lower triangular toeplitz matrices $\mathbf{T}(c_{1,1,0},\cdots,c_{1,1,R'-1})$, $\cdots$, $\mathbf{T}(c_{\ell,K_c,0},\cdots,c_{\ell,K_c,R'-1})$ are non-singular, and the block diagonal matrix $\hat{\mathbf{V}}'_{\ell,K_c,R',X,R}$ is invertible. Guaranteed by Lemma 4.1 and the fact that the Kronecker product of non-singular matrices is non-singular, the matrix $(\hat{\mathbf{V}}_{\ell,K_c,R',X,R}\hat{\mathbf{V}}'_{\ell,K_c,R',X,R}) \otimes \mathbf{I}_{\lambda/m}$ is invertible. Therefore, the master is able to recover $\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}$ by inverting the matrix. Note that $\mathbf{Z}''_{l,k,i}=\mathbf{0}, l\in[\ell], k\in[K_c], i\in\mathcal{E}$, therefore $\left(\mathbf{C}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in\mathcal{E}}=\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in\mathcal{E}}$. The desired products $(\mathbf{A}^{(l)}\mathbf{B}^{(l)})_{l\in[L]}$ are recoverable from $\left(\mathbf{C}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in\mathcal{E}}$, guaranteed by the correctness of Entangled Polynomial code [128]. This completes the proof of recovery threshold $R=pmn(\ell+1)K_c+2X-1$.

Consider the strong security property. According to the construction, $\mathcal{M}_1=0$, $\mathcal{M}_s=\tilde{\mathbf{M}}_s, s\in[S]\setminus\{1\}$, and $\mathcal{M}=\{\tilde{\mathbf{M}}_s\mid s\in[S]\setminus\{1\}\}$. Since $\tilde{\mathbf{M}}_s$ is a function of $\mathcal{Z}^{server}$,

$$I\left(\mathbf{A},\mathbf{B},\tilde{\mathbf{A}}^{[S]},\tilde{\mathbf{B}}^{[S]};\mathcal{M})\le I(\mathbf{A},\mathbf{B},\tilde{\mathbf{A}}^{[S]},\tilde{\mathbf{B}}^{[S]};\mathcal{Z}^{server}\right)=0.$$

Strong security is satisfied. Security is guaranteed because $\forall\mathcal{X}\subset[S], |\mathcal{X}|=X$,

$$\begin{aligned}
&I\left(\mathbf{A},\mathbf{B};\tilde{\mathbf{A}}^{\mathcal{X}},\tilde{\mathbf{B}}^{\mathcal{X}},\mathcal{M}_{\mathcal{X}}\right)\\
&=I\left(\mathbf{A},\mathbf{B};\mathcal{M}_{\mathcal{X}})+I(\mathbf{A},\mathbf{B};\tilde{\mathbf{A}}^{\mathcal{X}},\tilde{\mathbf{B}}^{\mathcal{X}}\mid\mathcal{M}_{\mathcal{X}}\right) &(4.46)\\
&=I\left(\mathbf{A},\mathbf{B};\mathcal{M}_{\mathcal{X}}\right)+I\left(\mathbf{A},\mathbf{B};\tilde{\mathbf{A}}^{\mathcal{X}},\tilde{\mathbf{B}}^{\mathcal{X}}\right)=0, &(4.47)
\end{aligned}$$

where (4.47) is due to (4.27), (4.28) and the facts that each share is encoded with (X,S) Reed-Solomon code with uniformly and independently distributed noise.

Consider the privacy property,

$$\begin{aligned}
&I\left(\mathbf{Y}_1,\mathbf{Y}_2,\cdots,\mathbf{Y}_S;\mathbf{A},\mathbf{B}\mid\mathbf{AB}\right)\\
&=I\left(\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']},(\mathbf{J}_x)_{x\in[R'K_c+2X-1]};\mathbf{A},\mathbf{B}\mid\mathbf{AB}\right) &(4.48)\\
&=I\left(\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']};\mathbf{A},\mathbf{B}\mid\mathbf{AB}\right)\\
&\quad+I\left((\mathbf{J}_x)_{x\in[R'K_c+2X-1]};\mathbf{A},\mathbf{B}\mid\mathbf{AB},\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}\right) &(4.49)\\
&=I\left((\mathbf{J}_x)_{x\in[R'K_c+2X-1]};\mathbf{A},\mathbf{B}\mid\mathbf{AB},\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}\right) &(4.50)\\
&\le I\left((\mathbf{J}_x)_{x\in[R'K_c+2X-1]};\mathbf{A},\mathbf{B},\mathbf{AB},\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}\right) &(4.51)
\end{aligned}$$

$$\leq I\left(\mathcal{Z}_1^{server}, \mathcal{Z}^A, \mathcal{Z}^B; \mathbf{A}, \mathbf{B}, \left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}\right) = 0, \tag{4.52}$$

where (4.48) holds because the mapping from $\left(\left(\mathbf{D}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in[R']}, (\mathbf{J}_x)_{x\in[R'K_c+2X-1]}\right)$ to $(\mathbf{Y}_1, \cdots, \mathbf{Y}_S)$ is bijective. Equation (4.50) holds due to (4.28) and the fact $\left(\mathbf{C}_i^{l,k}\right)_{l\in[\ell],k\in[K_c],i\in\mathcal{E}}$ are functions of $\mathbf{AB}$.

Consider the communication cost. The source upload cost $U_A = \frac{S}{K_cpm}$ and $U_B = \frac{S}{K_cpn}$. The server communication cost $CC = \frac{S-1}{\ell K_cmn}$. Note that the master is able to recover Lmn desired symbols from R downloaded symbols, the master download cost is $D = \frac{R}{Lmn} = \frac{pmn(\ell+1)K_c+2X-1}{\ell K_cmn}$. Thus the desired costs are achievable.

Now let us consider the computation complexity. Note that the source encoding procedure can be regarded as products of confluent Cauchy matrices by vectors. So by fast algorithms [87], the encoding complexity of $(\mathcal{C}_{eA}, \mathcal{C}_{eB}) = \left(\tilde{\mathcal{O}}\left(\frac{\lambda\kappa S\log^2 S}{K_cpm}\right), \tilde{\mathcal{O}}\left(\frac{\kappa\mu S\log^2 S}{K_cpn}\right)\right)$ is achievable. For the server computation complexity, each server multiplies the ℓ pairs of shares $\tilde{A}_l^s, \tilde{B}_l^s, l \in [\ell]$, and returns the sum of these ℓ products and structured noise $\tilde{M}_s$. With straightforward matrix multiplication algorithms, each of the ℓ matrix products has a computation complexity of $\mathcal{O}\left(\frac{\lambda\kappa\mu}{pmn}\right)$ for a total of $\mathcal{O}\left(\frac{\ell\lambda\kappa\mu}{pmn}\right)$. The complexity of summation over the products and noise is $\mathcal{O}\left(\frac{\ell\lambda\mu}{mn}\right)$. To construct the noise, one server needs to encode the noise, whose complexity is $\tilde{\mathcal{O}}\left(\frac{\lambda\mu S\log^2 S}{mn}\right)$ by fast algorithms [87]. Normalized by the number of servers, it is $\tilde{\mathcal{O}}\left(\frac{\lambda\mu\log^2 S}{mn}\right)$. Considering these 3 procedures, upon normalization by $L = \ell K_c$, it yields a complexity of $\mathcal{O}\left(\frac{\lambda\kappa\mu}{K_cpmn}\right) + \mathcal{O}\left(\frac{\lambda\mu}{K_cmn}\right) + \tilde{\mathcal{O}}\left(\frac{\lambda\mu\log^2 S}{\ell K_cmn}\right)$ per server. The master decoding complexity is inherited from that of GCSA codes [55], which is at most $\tilde{\mathcal{O}}(\lambda\mu p\log^2 R)$. This completes the proof of Theorem 4.1.

REMARK 4.3. *When $L = \ell = K_c = 1$, $S = R$, by setting $f_{1,1} = 0$, our construction of shares of $\tilde{A}^s$ and $\tilde{B}^s$ essentially recovers the construction of shares in [83].*

4.6 Discussion and Conclusion

In this paper, the class of GCSA codes is expanded by including noise-alignment, so that the resulting GCSA-NA code is a solution for secure coded multi-party computation of massive matrix multiplication. For two sources and matrix multiplication, GCSA-NA strictly generalizes PS [83] and outperforms it in several key aspects. This construction also settles the asymptotic capacity of symmetric X-secure T-private information retrieval. The idea of noise-alignment can be applied to construct a scheme for N sources based on N-CSA codes, and be combined with Strassen's construction. As open problems, exploring the optimal amount of randomness and finding the communication efficient schemes for arbitrary polynomial are interesting directions.

Since Strassen's algorithm [99] is an important fast matrix multiplication approach, it is interesting to show noise alignment can be combined with it for secure multi-party matrix multiplication. Consider an example with two 2×2 block matrices $\mathbf{A}, \mathbf{B}$ and $X = 1$. It can be shown that the general recursive Strassen's algorithm also works similarly. The desired product is denoted by $\mathbf{C} = \begin{bmatrix} \mathbf{C}_{1,1} & \mathbf{C}_{1,2} \\ \mathbf{C}_{2,1} & \mathbf{C}_{2,2} \end{bmatrix}$. The Strassen's constuction constructs 14 matrices $\mathbf{P}_i, \mathbf{Q}_i, i \in [7]$ ($\mathbf{P}_i$ only depends on $\mathbf{A}$ and $\mathbf{Q}_i$ only depends on $\mathbf{B}$) and

$$\begin{bmatrix} \mathbf{C}_{1,1} \\ \mathbf{C}_{1,2} \\ \mathbf{C}_{2,1} \\ \mathbf{C}_{2,2} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_1\mathbf{Q}_1 \\ \mathbf{P}_2\mathbf{Q}_2 \\ \vdots \\ \mathbf{P}_7\mathbf{Q}_7. \end{bmatrix}. \tag{4.53}$$

This is the basic Strassen algorithm. Now let us see how we apply CSA and noise alignment to it. Each share is constructed based on CSA code principles with noise, i.e., $\tilde{\mathbf{A}} = \Delta\left(\sum_{i\in[7]} \frac{\mathbf{P}_i}{f_i-\alpha} + \mathbf{Z}^A\right), \tilde{\mathbf{B}} = \sum_{i\in[7]} \frac{\mathbf{Q}_i}{f_i-\alpha} + \mathbf{Z}^B, \tilde{\mathbf{A}}\tilde{\mathbf{B}} = \sum_{i\in[7]} \frac{c_i}{f_i-\alpha}\mathbf{P}_i\mathbf{Q}_i + \sum_{i=0}^{7} \alpha^i \mathbf{I}_i$.

If the servers directly return $\tilde{\mathbf{A}}\tilde{\mathbf{B}}$ to the master, additional information about the input may be leaked due to interference terms $\mathbf{P}_1\mathbf{Q}_1, \cdots, \mathbf{P}_7\mathbf{Q}_7$ and $\sum_{i=0}^{6} \alpha^i I_i$. We secure the scheme by the addition of noise. The idea is that we want the master to decode $\mathbf{T}_1, \cdots, \mathbf{T}_7$ instead of $\mathbf{P}_1\mathbf{Q}_1, \cdots, \mathbf{P}_7\mathbf{Q}_7$, such that

$$H(\mathbf{C} \mid \mathbf{T}_1, \cdots, \mathbf{T}_7) = I(\mathbf{A}, \mathbf{B}; \mathbf{T}_1, \cdots, \mathbf{T}_7 \mid \mathbf{C}) = 0. \tag{4.54}$$

$\mathbf{T}_1, \cdots, \mathbf{T}_v$ are constructed as follows.

$$\begin{aligned}
&\mathbf{T}_1 = \mathbf{P}_1\mathbf{Q}_1 - \mathbf{Z}_1 - \mathbf{Z}_2 + \mathbf{Z}_3,\ \ \mathbf{T}_2 = \mathbf{P}_2\mathbf{Q}_2 - \mathbf{Z}_1 + \mathbf{Z}_2 - \mathbf{Z}_3,\\
&\mathbf{T}_3 = \mathbf{P}_3\mathbf{Q}_3 - \mathbf{Z}_1,\ \ \mathbf{T}_4 = \mathbf{P}_4\mathbf{Q}_4 + \mathbf{Z}_1,\ \ \mathbf{T}_5 = \mathbf{P}_5\mathbf{Q}_5 + \mathbf{Z}_2,\\
&\mathbf{T}_6 = \mathbf{P}_6\mathbf{Q}_6 - \mathbf{Z}_3,\ \ \mathbf{T}_7 = \mathbf{P}_7\mathbf{Q}_7 + \mathbf{Z}_3,
\end{aligned}$$

where $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$ are i.i.d. uniform noise matrices. To align the noise, we construct $\tilde{\mathbf{Z}}$,

$$\begin{aligned}
\tilde{\mathbf{Z}} = &\left(-\frac{c_1}{f_1-\alpha} - \frac{c_2}{f_2-\alpha} - \frac{c_3}{f_3-\alpha} + \frac{c_4}{f_4-\alpha}\right)\mathbf{Z}_1\\
&+ \left(-\frac{c_1}{f_1-\alpha} + \frac{c_2}{f_2-\alpha} + \frac{c_5}{f_5-\alpha}\right)\mathbf{Z}_2\\
&+ \left(\frac{c_1}{f_1-\alpha} - \frac{c_2}{f_2-\alpha} - \frac{c_6}{f_6-\alpha} + \frac{c_7}{f_7-\alpha}\right)\mathbf{Z}_3 + \sum_{i=0}^{6} \alpha^i \mathbf{Z}_{i+4},
\end{aligned}$$

where $\mathbf{Z}_4, \cdots, \mathbf{Z}_{10}$ are i.i.d. uniform noise matrices. The answer returned by each server to the master is $\tilde{\mathbf{A}}\tilde{\mathbf{B}} + \tilde{\mathbf{Z}}$. The correctness and privacy are easily proved.

Chapter 5

Flexible Distributed Matrix Multiplication

5.1 Introduction

Distributed matrix multiplication has received wide interest because of the huge amount of data computation required by many popular applications like federated learning, cloud computing, and the Internet of things. In particular, the multiplication of two massive input matrices $A \in \mathbb{F}^{\lambda\times\kappa}$ and $B \in \mathbb{F}^{\kappa\times\mu}$, where $\mathbb{F}$ is some finite field is considered. Each matrix is encoded into N *shares* and distributed to N servers. Each server performs computation on its own shares and sends the *results* to the central computational node, e.g., the cloud. After collecting enough results, the desired product AB can be calculated. However, stragglers (servers that fail to respond or respond after the the reconstruction is executed) are inevitable in distributed systems, due to various reasons [3, 26] including network latency, resource contention, workload imbalance, failures of hardware or software, etc. To reduce the overall system latency caused by stragglers, distributed matrix computing schemes with straggler tolerance are provided in [1, 4, 17, 18, 27, 28, 29, 30, 31, 44, 53, 54, 55, 62, 67, 68, 70, 71, 75, 79, 90, 93, 95, 98, 100, 115, 116, 125, 126, 127, 128] with a predetermined *recovery threshold* R such that the final product can be obtained using computation results from any R out of N servers. Among the state-of-the-art schemes, some are based on matrix partitioning such as Polynomial codes [127], MatDot codes and PolyDot codes [30], Generalized PolyDot codes [27] and Entangled Polynomial (EP) codes [128], and others are based on batch processing such as Lagrange Coded Computing [125], Cross Subspace Alignment (CSA) codes and Generalized Cross Subspace Alignment (GCSA) codes [55].

The above literature assumes there are a fixed number $N-R$ of stragglers. However, the number of stragglers is unpredictable in practical systems. When the number of stragglers is smaller than $N-R$, each non-straggler server still needs to do the same amount of computation as if there are $N-R$ stragglers and the central node still only uses the results from R servers. A significant amount of computation power is wasted. To handle this situation, a setting in which the number of stragglers is not known a priori has been considered in [2, 11, 13, 14, 24, 25, 33, 36, 45, 46, 65, 66, 91] and schemes that can cope with such a setting have been designed. The underlying idea is to assign a sequence of small tasks to each server instead of assigning a single large task. Therefore, besides the scenario that the fastest R servers finish all their tasks, there are other scenarios that make the computation complete. References [2, 36] focus on the task scheduling for general distributed computing. The matrix-vector multiplication setting is considered in [11, 14, 25, 91]. In these works, only the input matrix is partitioned. References [13, 24, 33, 45, 46, 65] consider matrix-matrix multiplication, but they can only handle a special partitioning, i.e., A and B are row-wisely and column-wisely split, respectively, or only A is row-wisely split. In [66], the authors propose 3 hierarchical schemes for matrix multiplication to leverage partial stragglers. The main idea is that the task is first divided into several small subtasks, i.e., the multiplication of several pairs of small matrices, and each subtask is coded separately with existing schemes.

Arbitrary partitioning of input matrices is important in massive matrix multiplication since it enables different utilization of system resources, e.g., the required amount of storage at each server and the amount of communication from servers to the central node. When the number of stragglers is fixed, many codes such as PolyDot codes [30], EP codes [128] and GCSA codes [55] provide elegant solutions for arbitrary partitioning by encoding the input matrix blocks into a carefully designed polynomial. In particular, EP codes effectively align the servers' computation with the terms that the central node needs and achieve the optimal recovery threshold among all linear coding strategies in some cases.

A naive solution to achieve flexibility for distributed matrix multiplication with arbitrary partitioning is simply applying a fixed EP code with a recovery threshold of PR, where each server gets P pairs of shares instead of one pair of shares. The central node can calculate the final results with any PR out of the PN computing results. Thus, each server only needs to compute RP/N results when there is no straggler, and in general, the number of results computed in each server can be adjusted based on the number of stragglers. However, by doing so, the computation needs to be done in a field with a minimum size of PN and operations over a larger field result in a much bigger delay [40].

In this chapter, we present a flexible coding scheme for distributed matrix multiplication that allows a flexible number of stragglers and arbitrary matrix partitioning while only requiring a much smaller field size. The main idea is that non-straggler servers can finish more tasks to compensate for the effect of the stragglers without knowing the stragglers a priori. Specifically, the computation is encoded into several tasks for each server, and each server keeps calculating and sending results to the central node until enough results are obtained. Enough results can be either a larger number of servers with a smaller amount of completed computation by each server or a smaller number of servers with a larger amount of completed computation by each server. Therefore, the number of available servers is flexible and the number of required tasks is adjusted to the number of available servers. Our scheme is different from those that leverage partial stragglers [24, 25, 33, 65, 66, 91]. In our construction, the computation load (the number of multiplication operations) of each non-straggler server is the same and the computation by stragglers is neglected, while in schemes with partial stragglers, the computation load varies in different servers including stragglers.

The main contributions of the chapter are as follows. We present a coding framework of flexible distributed matrix multiplication schemes, and one-round and multi-round communication models. A construction with multiple layers of computation tasks is proposed, which only requires a field size of less than $2N$ and the computation load of each server is reduced significantly when there are fewer stragglers than $N-R$. We also demonstrate the optimization of the parameters to obtain the lowest computation load. We show that the two-layer construction outperforms the fixed scheme under the one-round model as long as the server storage is above a threshold, and the maximum number of layers is preferred under the multi-round model.

The rest of the chapter is organized as follows. Section 5.2 presents the problem statement. In Section 5.3, we present our construction and its performance. The choice of parameters to optimize the computation load given the storage capacity is discussed in Section 5.4. Section 5.5 concludes the chapter.

Notation: We use calligraphic characters to denote sets. For positive integer N, $[N]$ stands for the set $\{1, 2, \ldots, N\}$. For a matrix M, $|M|$ denotes its number of entries. For a set of matrices $\mathcal{M}$, $|\mathcal{M}|$ represents the sum of the number of entries in all its matrices. When M is partitioned into sub-block matrices, $M_{(u,v)}$ denotes the block in the u-th row and the v-th column.

5.2 Problem Statement

We consider a problem of matrix multiplication (see Fig. 5.1) with two input matrices $A \in \mathbb{F}^{\lambda\times\kappa}$ and $B \in \mathbb{F}^{\kappa\times\mu}$, for some integers λ, κ, μ and a field $\mathbb{F}$. We are interested in computing the product $\Gamma = AB$ in a distributed computing environment with 2 sources, a central node, and N servers. Sources 1 and 2 hold matrices A and B, respectively. It is assumed that there are up to $N-R$ stragglers among the servers. In non-flexible distributed matrix multiplication, R is called the *recovery threshold*. The shares (coded matrix sets) $\tilde{\mathcal{A}}_i$ and $\tilde{\mathcal{B}}_i$ are generated by sources for Server i, $i \in [N]$. Each share consists of some coded matrices, denoted by $\left\{\tilde{A}_{i,1}, \cdots, \tilde{A}_{i,\tilde{\gamma}}\right\}$, or $\left\{\tilde{B}_{i,1}, \cdots, \tilde{B}_{i,\tilde{\gamma}}\right\}$,

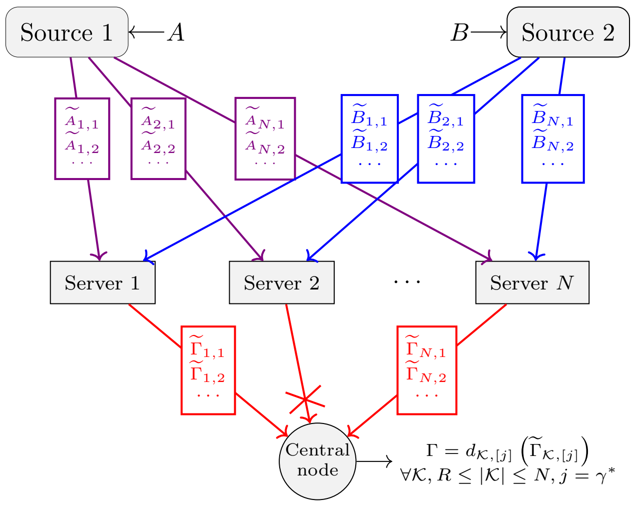

Source 1
A
B
Source 2
$\widetilde{A}_{1,1}$ $\widetilde{A}_{1,2}$ $\cdots$
$\widetilde{A}_{2,1}$ $\widetilde{A}_{2,2}$ $\cdots$
$\widetilde{A}_{N,1}$ $\widetilde{A}_{N,2}$ $\cdots$
$\widetilde{B}_{1,1}$ $\widetilde{B}_{1,2}$ $\cdots$
$\widetilde{B}_{2,1}$ $\widetilde{B}_{2,2}$ $\cdots$
$\widetilde{B}_{N,1}$ $\widetilde{B}_{N,2}$ $\cdots$
Server 1
Server 2
$\cdots$
Server N
$\widetilde{\Gamma}_{1,1}$ $\widetilde{\Gamma}_{1,2}$ $\cdots$
$\widetilde{\Gamma}_{N,1}$ $\widetilde{\Gamma}_{N,2}$ $\cdots$
Central node
$\Gamma = d_{\mathcal{K},[j]}\left(\widetilde{\Gamma}_{\mathcal{K},[j]}\right)$
$\forall \mathcal{K}, R \le |\mathcal{K}| \le N, j = \gamma^*$

Figure 5.1: The flexible distributed matrix multiplication problem.

where $\tilde{\gamma}$ is a function of N and R. For $i \in [N]$, the shares and the encoding functions are

$$\tilde{\mathcal{A}}_i = \left\{\tilde{A}_{i,j} \mid j \in [\tilde{\gamma}]\right\} = u_i(A), \tag{5.1}$$

$$\tilde{\mathcal{B}}_i = \left\{\tilde{B}_{i,j} \mid j \in [\tilde{\gamma}]\right\} = v_i(B). \tag{5.2}$$

Then, $\tilde{\mathcal{A}}_i$ and $\tilde{\mathcal{B}}_i$ are sent to Server i from the sources. Server i sequentially computes $\tilde{\gamma}$ *tasks* in order:

$$\tilde{\Gamma}_{i,j} = \tilde{A}_{i,j} \cdot \tilde{B}_{i,j},\ j \in [\tilde{\gamma}], \tag{5.3}$$

and sends $\tilde{\Gamma}_{i,j}$ to the central node once its computation is finished. Due to the sequential processing nature of the servers, the central node receives $\tilde{\Gamma}_{i,j_1}$ before $\tilde{\Gamma}_{i,j_2}$ for $\forall i \in [N], j_1 < j_2$. Denote $\tilde{\Gamma}_{i,[j]} = \left\{\tilde{\Gamma}_{i,t} \mid t \in [j]\right\}$ and $\tilde{\Gamma}_{\mathcal{K},[j]} = \left\{\tilde{\Gamma}_{i,[j]} \mid i \in \mathcal{K}\right\}, \forall \mathcal{K} \subset [N]$.

We require that the central node be able to decode the desired product Γ from arbitrary $\hat{R} \geq R$ servers, where each server calculates γ^* (a function of $\hat{R}$) tasks. Equivalently, the decoding function $d_{\mathcal{K},[j]}$ of the central node for recovering Γ satisfies

$$\Gamma = d_{\mathcal{K},[j]}\left(\tilde{\Gamma}_{\mathcal{K},[j]}\right), \forall \mathcal{K}, R \leq |\mathcal{K}| = \hat{R} \leq N, j = \gamma^*. \tag{5.4}$$

The function set $\{u_i, v_i, d_{\mathcal{K},[j]} \mid 1 \leq i \leq N, R \leq |\mathcal{K}| = \hat{R} \leq N, j = \gamma^*\}$ is called the *flexible constructions for distributed matrix multiplication*.

In other words, the sources send coded matrices to each server. Each server keeps calculating and sending results to the central node until it obtains enough results – when the quickest $\hat{R}$ servers complete the first γ^* tasks. The remaining servers are viewed as stragglers and the computation results from stragglers are ignored.

In this chapter, we consider two communication models: the one-round communication model and the multi-round communication model. For the *one-round communication model*, the sources send *all* $\tilde{\gamma}$ coded matrices to the server at one time. After that there are no communications between sources and servers. For the *multi-round communication model*, first, the sources send one pair of coded matrices to the servers. Once a server finishes its tasks, it will ask the sources to send another pair of coded matrices. It is not necessary for the sources to know which servers are the stragglers. This procedure lasts until the central node obtains enough results. Note that there are no communications among servers in either models.

The *computation load* L is defined as the number of multiplication operations per server. Moreover, each server has a *storage capacity* C [1]. At any time, any server cannot store more than C. Specifically, for the one-round communication model, $\max_{i\in[N]} \left(\left|\tilde{\mathcal{A}}_i\right| + \left|\tilde{\mathcal{B}}_i\right|\right) \leq C$. For the multi-round communication model, $\max_{i\in[N],j\in[\tilde{\gamma}]} \left(\left|\tilde{A}_{i,j}\right| + \left|\tilde{B}_{i,j}\right|\right) \leq C$. This is because once a server finishes a task and sends the result to the central node, it can refresh the storage and delete the coded matrices related to this task. In general, the storage constraint is stricter in the one-round communication model. We want to find flexible constructions with the *storage capacity* C and the *computation load* L at each server as small as possible.

[1]The maximum storage size C is usually smaller than $|A|+|B|$, otherwise the sources can send A and B to the servers.

5.3 Construction

In this section, we present our flexible constructions. The scheme is based on EP code [128] and the computation tasks are divided into several layers to provide flexibility. We start with a motivating example. The general construction and its storage and computation load are presented afterwards.

EXAMPLE 5.1. *Consider the matrix multiplication of A and B, for $A \in \mathbb{F}^{\lambda\times\kappa}, B \in \mathbb{F}^{\kappa\times\mu}$, using $N = 5$ servers with at most $N - R = 2$ stragglers. Suppose A is column-wisely partitioned as $A = [A_1, A_2]$, each submatrix is of size $\lambda \times \frac{\kappa}{2}$, and B is row-wisely partitioned as $B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$, each submatrix is of size $\frac{\kappa}{2} \times \mu$. The central node requires $AB = A_1B_1 + A_2B_2$. Applying the EP code, Server $i, i \in [5]$ receives coded matrices $A_1 + \alpha_i A_2$ and $\alpha_i B_1 + B_2$, for $\alpha_i \in \mathbb{F}$, and calculates*

$$\begin{aligned} &(A_1 + \alpha_i A_2) \cdot (\alpha_i B_1 + B_2) \\ =&A_1B_2 + \alpha_i(A_1B_1 + A_2B_2) + \alpha_i^2 A_2B_1, \end{aligned} \tag{5.5}$$

which is a degree 2 polynomial with respect to α_i. Thus, $A_1B_1 + A_2B_2$ can be calculated by 3 distinct evaluations from $\{\alpha_i \mid i \in [5]\}$ using Lagrange interpolation. The total computation load of directly multiplying A and B is $L = \lambda\kappa\mu$, while using the EP code the computation load of each server is $L/2$. However, when there is no straggler, the computation of 2 servers are wasted.

Alternatively, we can use a flexible scheme to calculate AB, such that any $\hat{R}$ available servers can complete the computation, $3 = R \leq \hat{R} \leq N = 5$. First, we partition the matrices and get $A = [A_1, A_2, A_3]$, each submatrix is of size $\lambda \times \frac{\kappa}{3}$, and $B = [B_1^T, B_2^T, B_3^T]^T$, each submatrix is of size $\frac{\kappa}{3} \times \mu$. The central node requires $AB = A_1B_1 + A_2B_2 + A_3B_3$. Let $\{\alpha_i | i \in [7]\}$ be distinct elements in $\mathbb{F}$. The calculation will be divided into 2 layers.

Layer 1: Server $i, i \in [5]$, calculates $\gamma_1 = 1$ task

$$\begin{aligned}\Gamma_{i,1} =&(A_1 + \alpha_i A_2 + \alpha_i^2 A_3) \cdot (\alpha_i^2 B_1 + \alpha_i B_2 + B_3) \\ =&A_1B_3 + \alpha_i(A_2B_3 + A_1B_2) + \alpha_i^2(A_1B_1 + A_2B_2 + A_3B_3) \\ &+ \alpha_i^3(A_2B_1 + A_3B_2) + \alpha_i^4 A_3B_1. \end{aligned} \tag{5.6}$$

It is a degree 4 polynomial with respect to α_i and the final product can be obtained from all 5 servers. If there is no straggler, we stop here. In this layer, matrices A and B are divided into smaller pieces compared to the fixed EP code and the computation load of each server is $L/3$. If there are stragglers, the servers continue the calculation in Layer 2.

Layer 2: We set $A_{\alpha_i} = A_1 + \alpha_i A_2 + \alpha_i^2 A_3, B_{\alpha_i} = \alpha_i^2 B_1 + \alpha_i B_2 + B_3, i \in \{6, 7\}$ and partition them into 2 parts,

$$A_{\alpha_i} = [A_{\alpha_i,1}, A_{\alpha_i,2}], B_{\alpha_i} = \begin{bmatrix} B_{\alpha_i,1} \\ B_{\alpha_i,2} \end{bmatrix}. \tag{5.7}$$

Server i has $\gamma_2 = 2$ computation tasks:

$$\Gamma_{i,2} = (A_{\alpha_6,1} + \alpha_i A_{\alpha_6,2}) \cdot (\alpha_i B_{\alpha_6,1} + B_{\alpha_6,2}), \tag{5.8}$$

$$\Gamma_{i,3} = (A_{\alpha_7,1} + \alpha_i A_{\alpha_7,2}) \cdot (\alpha_i B_{\alpha_7,1} + B_{\alpha_7,2}). \tag{5.9}$$

The detailed calculation of each server is shown in Table 5.1.

Since Layer 2 has a similar structure as (5.5)*, from any 3 of the servers, we can get $A_{\alpha_6} \cdot B_{\alpha_6}$ and/or $A_{\alpha_7} \cdot B_{\alpha_7}$. If there is one straggler, the central node obtains $A_{\alpha_6} \cdot B_{\alpha_6}$ from Layer 2, which causes the additional computation load of $L/6$ in a server. If there are 2 stragglers, the central node obtains both $A_{\alpha_6} \cdot B_{\alpha_6}$ and $A_{\alpha_7} \cdot B_{\alpha_7}$, which causes the computation load of $L/3$ in Layer 2 for each server.*

In this example, there are two recovery thresholds $R_1 = 5$ and $R_2 = 3$, corresponding to two layers, respectively. We term the choice of per-layer recovery thresholds as recovery profile. *There are totally $\tilde{\gamma} = \gamma_1 + \gamma_2 = 3$ coded matrices in a share where γ_1 coded matrices correspond to Layer 1 and γ_2 coded matrices correspond to Layer 2. Specifically, $\forall i \in [N]$, the shares $\tilde{\mathcal{A}}_i$ and $\tilde{\mathcal{B}}_i$ contain*

$$\tilde{A}_{i,1} = A_{\alpha_i}, \;\; \tilde{A}_{i,2} = A_{\alpha_6,1} + \alpha_i A_{\alpha_6,2}, \;\; \tilde{A}_{i,3} = A_{\alpha_7,1} + \alpha_i A_{\alpha_7,2}, \tag{5.10}$$

$$\tilde{B}_{i,1} = B_{\alpha_i}, \;\; \tilde{B}_{i,2} = B_{\alpha_6,1} + \alpha_i B_{\alpha_6,2}, \;\; \tilde{B}_{i,3} = B_{\alpha_7,1} + \alpha_i B_{\alpha_7,2}, \tag{5.11}$$

respectively. Each server needs to store all the above 6 coded matrices under the one-round communication model, but only 2 coded matrices at a time under the multi-round communication. Each server computes up to $\tilde{\gamma} = 3$ tasks in order, independent of the progress of the other servers.

Table 5.1: Calculation tasks in each server for Example 1.

	Server 1	Server 2	Server 3	Server 4	Server 5
Layer 1	$A_{\alpha_1} \cdot B_{\alpha_1}$	$A_{\alpha_2} \cdot B_{\alpha_2}$	$A_{\alpha_3} \cdot B_{\alpha_3}$	$A_{\alpha_4} \cdot B_{\alpha_4}$	$A_{\alpha_5} \cdot B_{\alpha_5}$
Layer 2	$(A_{\alpha_6,1} + \alpha_1 A_{\alpha_6,2})$ $\cdot(\alpha_1 B_{\alpha_6,1} + B_{\alpha_6,2})$, $(A_{\alpha_7,1} + \alpha_1 A_{\alpha_7,2})$ $\cdot(\alpha_1 B_{\alpha_7,1} + B_{\alpha_7,2})$	$(A_{\alpha_6,1} + \alpha_2 A_{\alpha_6,2})$ $\cdot(\alpha_2 B_{\alpha_6,1} + B_{\alpha_6,2})$, $(A_{\alpha_7,1} + \alpha_2 A_{\alpha_7,2})$ $\cdot(\alpha_2 B_{\alpha_7,1} + B_{\alpha_7,2})$	$(A_{\alpha_6,1} + \alpha_3 A_{\alpha_6,2})$ $\cdot(\alpha_3 B_{\alpha_6,1} + B_{\alpha_6,2})$, $(A_{\alpha_7,1} + \alpha_3 A_{\alpha_7,2})$ $\cdot(\alpha_3 B_{\alpha_7,1} + B_{\alpha_7,2})$	$(A_{\alpha_6,1} + \alpha_4 A_{\alpha_6,2})$ $\cdot(\alpha_4 B_{\alpha_6,1} + B_{\alpha_6,2})$, $(A_{\alpha_7,1} + \alpha_4 A_{\alpha_7,2})$ $\cdot(\alpha_4 B_{\alpha_7,1} + B_{\alpha_7,2})$	$(A_{\alpha_6,1} + \alpha_5 A_{\alpha_6,2})$ $\cdot(\alpha_5 B_{\alpha_6,1} + B_{\alpha_6,2})$, $(A_{\alpha_7,1} + \alpha_5 A_{\alpha_7,2})$ $\cdot(\alpha_5 B_{\alpha_7,1} + B_{\alpha_7,2})$

For Example 5.1, the computation load of each server is $L/3, L/2, 2L/3$ for the cases of no stragglers, 1 straggler and 2 stragglers, respectively. When there is no straggler (which is more likely in most practical systems), the computation load of each server is reduced 33%, from $L/2$ to $L/3$. The resulting computation latency under an exponential model is plotted in Fig. 5.2.

In this example, if there is only one communication round from the sources to the servers, the storage size required for each server is $\frac{2\lambda\kappa}{3} + \frac{2\kappa\mu}{3}$ for our flexible construction and $\frac{\lambda\kappa}{2} + \frac{\kappa\mu}{2}$ for the EP code. We will discuss how to partition the matrices to obtain an advantageous computation load while maintaining the same storage size in Section 5.4.

Next, we present the general definitions and constructions of our flexible schemes. The key component is to generate extra parities during the encoding in each layer that will correspond to extra

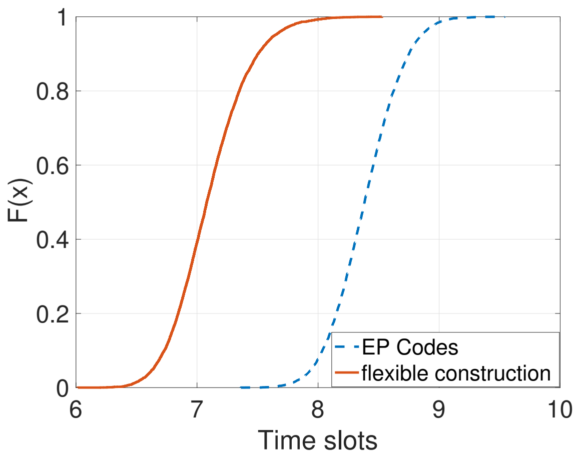


Figure 5.2: CDF of computation latency for flexible construction and EP code in Example 1 of Section 5.3. $N = R_1 = 5, R_2 = R = 3$. We assume $\lambda = \kappa = \mu = 6U$, for some integer U, and the computation delay for multiplication of two $U \times U$ matrices in each server satisfy the exponential distribution with parameter 0.1. The latency of the EP code is the delay of the 3rd quickest server, and the slowest 2 servers are viewed as stragglers. For the flexible construction, the computation is completed in the cases of 5 servers complete 1 task (no straggler), or 4 servers complete 2 tasks (1 straggler), or 3 servers complete 3 tasks (2 stragglers). The overall latency is the smallest latency of these 3 cases. The expected latency is 10.79 for EP code, and 8.20 for the flexible construction. Hence we save 24%.

tasks to be completed by higher layers to compensate for more stragglers.

Define the *recovery profile* as a tuple of integers $(R_1, R_2, \cdots, R_a)$, where $N \geq R_1 > R_2 > ... > R_a = R$ and a is some integer termed the *number of layers*. Denote

$$\gamma_j = \begin{cases} 1, & j = 1, \\ (R_{j-1} - R_j) \sum_{J=1}^{j-1} \gamma_J, & 2 \leq j \leq a, \end{cases} \tag{5.12}$$

which will be shown to be the number of tasks in each layer. For two matrices Φ, Ψ and partition parameters p_j, m_j, n_j, define functions $f_j, g_j, j \in [a]$, as

$$f_j(\alpha_i; \Phi) = \sum_{u=1}^{m_j} \sum_{v=1}^{p_j} \Phi_{(u,v)} \alpha_i^{v-1+p_j(u-1)}, \tag{5.13}$$

$$g_j(\alpha_i;\Psi) = \sum_{u=1}^{p_j}\sum_{v=1}^{n_j} \Psi_{(u,v)}\alpha_i^{p_j-u+p_jm_j(v-1)}, \tag{5.14}$$

where

$$\Phi = \begin{bmatrix} \Phi_{(1,1)} & \cdots & \Phi_{(1,p_j)} \\ \Phi_{(2,1)} & \cdots & \Phi_{(2,p_j)} \\ \vdots & \vdots & \vdots \\ \Phi_{(m_j,1)} & \cdots & \Phi_{(m_j,p_j)} \end{bmatrix}, \Psi = \begin{bmatrix} \Psi_{(1,1)} & \cdots & \Psi_{(1,n_j)} \\ \Psi_{(2,1)} & \cdots & \Psi_{(2,n_j)} \\ \vdots & \vdots & \vdots \\ \Psi_{(p_j,1)} & \cdots & \Psi_{(p_j,n_j)} \end{bmatrix}. \tag{5.15}$$

Note that (5.13) and (5.14) are the encoding functions of the EP codes [128] used in Layer j.

CONSTRUCTION 5.1. *Given recovery profile $(R_1, R_2, \cdots, R_a)$ and partitioning parameters p_j, m_j, n_j such that $R_j = p_jm_jn_j + p_j - 1, j \in [a]$, the construction consists of a layers. Fix $N + R_1 - R_a$ distinct elements $\alpha_i, i \in [N + R_1 - R_a]$, in a finite field $\mathbb{F}$.*

In Layer 1, set $A^{(1,1)} = A$ and $B^{(1,1)} = B$. A pair of coded matrices $f_1\left(\alpha_t; A^{(1,1)}\right)$ and $g_1\left(\alpha_t; B^{(1,1)}\right)$ are generated for Server t, $t \in [N]$. Moreover, extra $R_1 - R_a$ pairs of parities will be generated, i.e., $f_1\left(\alpha_{N+t}; A^{(1,1)}\right)$ and $g_1\left(\alpha_{N+t}; B^{(1,1)}\right)$, $t \in [R_1 - R_a]$. They will be used in higher layers.

In Layer $j, 2 \le j \le a$, the number of pairs of coded matrices is γ_j given by (5.12). For each $\delta_j \in [\gamma_j]$, a pair of coded matrices $f_j\left(\alpha_t; A^{(j,\delta_j)}\right)$ and $g_j\left(\alpha_t; B^{(j,\delta_j)}\right)$ are generated for Server t, $t \in [N]$. Besides, extra parities $f_j(\alpha_{N+t}; A^{(j,\delta_j)})$ and $f_j(\alpha_{N+t}; B^{(j,\delta_j)})$, $t \in [R_j - R_a]$, are produced for higher layers. Here, $A^{(j,\delta_j)}$ and $B^{(j,\delta_j)}$, $\delta_j \in [\gamma_j]$, are from the extra parities $f_J(\alpha_{N+t}; A^{(J,\delta_J)}), g_J(\alpha_{N+t}; B^{(J,\delta_J)})$ in Layer J for all $J \in [j-1]$ and

$$R_j - R_a + 1 \le t \le R_{j-1} - R_a, \delta_J \in [\gamma_J]. \tag{5.16}$$

Specifically, given j and δ_j, $A^{(j,\delta_j)}$ and $B^{(j,\delta_j)}$ are set as

$$A^{(j,\delta_j)} = f_J\left(\alpha_{N+t} : A^{(J,\delta_J)}\right), \tag{5.17}$$

$$B^{(j,\delta_j)} = g_J\left(\alpha_{N+t} : B^{(J,\delta_J)}\right), \tag{5.18}$$

where

$$t = \left\lfloor \delta_j \frac{R_{j-1} - R_j}{\gamma_j} \right\rfloor + R_j - R_a + 1, \tag{5.19}$$

and J is the integer satisfying

$$\sum_{x=1}^{J-1} \gamma_x < \delta_j \textit{ mod } \frac{\gamma_j}{(R_{j-1} - R_j)} \leq \sum_{x=1}^{J} \gamma_x \tag{5.20}$$

and

$$\delta_J = \delta_j \textit{ mod } \frac{\gamma_j}{(R_{j-1} - R_j)} - \sum_{x=1}^{J-1} \gamma_x. \tag{5.21}$$

Intuitively, the t-th extra parities in all previous layers are encoded in Layer j, for all t satisfying (5.16). *Equations* (5.19)*,* (5.20)*, and* (5.21) *simply mean that these extra parities are ordered from left to right and from top to bottom (see Fig.* 5.3 *for an example).*

Denote $\Gamma_{j,\delta_j}(\alpha_i)$ as the δ_j-th task in Layer j calculated in Server i, for $i \in [N], j \in [a], \delta_j \in [\gamma_j]$, where

$$\Gamma_{j,\delta_j}(\alpha_i) = f_j\left(\alpha_i; A^{(j,\delta_j)}\right) \cdot g_j\left(\alpha_i; B^{(j,\delta_j)}\right), \tag{5.22}$$

The calculation tasks of the construction are shown in Table 5.2*.*

Note that there are in total $\tilde{\gamma} = \sum_{j=1}^{a} \gamma_j$ tasks. The shares and the tasks are

$$\tilde{\mathcal{A}}_i = \{A^{(j,\delta_j)} \mid j \in [a], \delta_j \in [\gamma_j]\}, \tag{5.23}$$

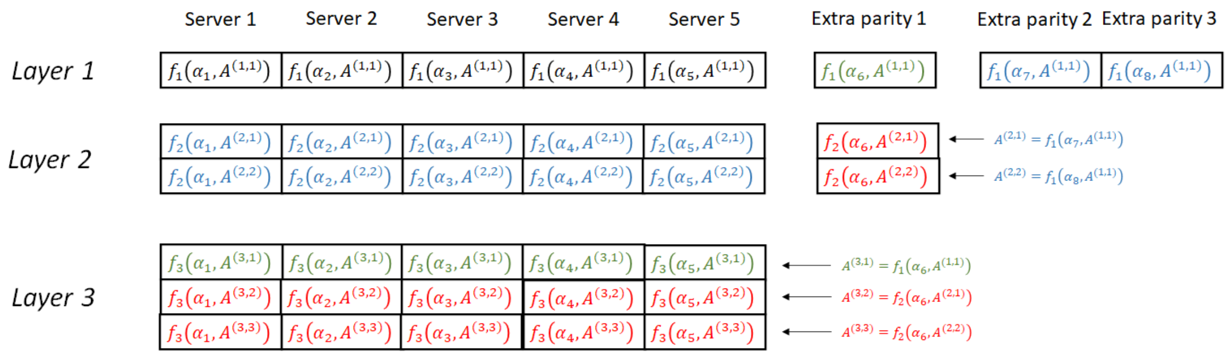
$$\tilde{\mathcal{B}}_i = \{B^{(j,\delta_j)} \mid j \in [a], \delta_j \in [\gamma_j]\}, \tag{5.24}$$

$$\tilde{\Gamma}_{i,\sum_{x=1}^{j-1}\gamma_x+\delta_j} = \Gamma_{j,\delta_j}(\alpha_i). \tag{5.25}$$

Table 5.2: Calculation tasks in each server for the multiple-layer construction, where δ_j ranges between 1 and γ_j as defined in (5.12), $j \in [a]$.

	Server 1	$\dots$	Server N	Extra parity 1	$\dots$	$\dots$	Extra parity $R_1 - R_a$
Layer 1	$\Gamma_{1,1}(\alpha_1)$	$\dots$	$\Gamma_{1,1}(\alpha_N)$	$\Gamma_{1,1}(\alpha_{N+1})$	$\dots$	$\dots$	$\Gamma_{1,1}(\alpha_{N+R_1-R_a})$
Layer 2	$\Gamma_{2,\delta_2}(\alpha_1)$	$\dots$	$\Gamma_{2,\delta_2}(\alpha_N)$	$\Gamma_{2,\delta_2}(\alpha_{N+1})$	$\dots$	$\Gamma_{2,\delta_2}(\alpha_{N+R_2-R_a})$	
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$		
Layer a	$\Gamma_{a,\delta_a}(\alpha_1)$	$\dots$	$\Gamma_{a,\delta_a}(\alpha_N)$				

Example 5.2. *An example of a 3-layer construction is shown in Fig. 5.3. We set $N = 5, R = 2, (R_1, R_2, R_3) = (5, 3, 2)$. In Fig. 5.3, we show that the coded matrices transmitted from Source 1 and Source 2 are similar. In Layer 1 ($A^{(1,1)} = A$), the coded matrices $f_1(\alpha_i; A^{(1,1)})$ are transmitted to Server i, $i \in [5]$, and $f_1(\alpha_{5+t}; A^{(1,1)}), t \in [3]$ are the extra parities. These parities are used in Layers 2 and 3. Specifically, $A^{(2,1)} = f_1(\alpha_7; A^{(1,1)})$ and $A^{(2,2)} = f_1(\alpha_8; A^{(1,1)})$ are used in Layer 2 and $A^{(3,1)} = f_1(\alpha_6; A^{(1,1)})$ is used in Layer 3. In Layer 2, $f_2(\alpha_i; A^{(2,\delta_2)}), \delta_2 \in [2], i \in [5]$ are encoded using the above extra parities from Layer 1. The generated extra parities $A^{(3,2)} = f_2(\alpha_6; A^{(2,1)})$ and $A^{(3,3)} = f_2(\alpha_6; A^{(2,2)})$ are used in Layer 3.*



Server 1
Server 2
Server 3
Server 4
Server 5
Extra parity 1
Extra parity 2
Extra parity 3
Layer 1
Layer 2
Layer 3
$f_1(\alpha_1, A^{(1,1)})$
$f_1(\alpha_2, A^{(1,1)})$
$f_1(\alpha_3, A^{(1,1)})$
$f_1(\alpha_4, A^{(1,1)})$
$f_1(\alpha_5, A^{(1,1)})$
$f_1(\alpha_6, A^{(1,1)})$
$f_1(\alpha_7, A^{(1,1)})$
$f_1(\alpha_8, A^{(1,1)})$
$f_2(\alpha_1, A^{(2,1)})$
$f_2(\alpha_2, A^{(2,1)})$
$f_2(\alpha_3, A^{(2,1)})$
$f_2(\alpha_4, A^{(2,1)})$
$f_2(\alpha_5, A^{(2,1)})$
$f_2(\alpha_6, A^{(2,1)})$
$A^{(2,1)} = f_1(\alpha_7, A^{(1,1)})$
$f_2(\alpha_1, A^{(2,2)})$
$f_2(\alpha_2, A^{(2,2)})$
$f_2(\alpha_3, A^{(2,2)})$
$f_2(\alpha_4, A^{(2,2)})$
$f_2(\alpha_5, A^{(2,2)})$
$f_2(\alpha_6, A^{(2,2)})$
$A^{(2,2)} = f_1(\alpha_8, A^{(1,1)})$
$f_3(\alpha_1, A^{(3,1)})$
$f_3(\alpha_2, A^{(3,1)})$
$f_3(\alpha_3, A^{(3,1)})$
$f_3(\alpha_4, A^{(3,1)})$
$f_3(\alpha_5, A^{(3,1)})$
$A^{(3,1)} = f_1(\alpha_6, A^{(1,1)})$
$f_3(\alpha_1, A^{(3,2)})$
$f_3(\alpha_2, A^{(3,2)})$
$f_3(\alpha_3, A^{(3,2)})$
$f_3(\alpha_4, A^{(3,2)})$
$f_3(\alpha_5, A^{(3,2)})$
$A^{(3,2)} = f_2(\alpha_6, A^{(2,1)})$
$f_3(\alpha_1, A^{(3,3)})$
$f_3(\alpha_2, A^{(3,3)})$
$f_3(\alpha_3, A^{(3,3)})$
$f_3(\alpha_4, A^{(3,3)})$
$f_3(\alpha_5, A^{(3,3)})$
$A^{(3,3)} = f_2(\alpha_6, A^{(2,2)})$

Figure 5.3: Example of coded matrices for 3-layer construction, $N = R_1 = 5, R_2 = 3, R_3 = R = 2$.

Theorem 5.1, below, states the performance of the flexible construction in terms of storage and computation. This result is based on the following decoding strategy: in the presence of $\hat{R}$ available servers, $R_j \leq \hat{R} < R_{j-1}$, all tasks in Layers $1, 2, \ldots, j-1$ and some tasks in Layer j are executed. It should be noted that the sum of storage sizes in all layers corresponds to the one-round communication model. However, under the multi-round communication model, the server storage size is only the maximum over the pairs of coded matrices. Since the coded matrix in a layer is encoded from sub-matrices in the previous layer, the higher the layer is, the smaller the size becomes. Hence, the storage size is just that of the first pair of coded matrices.

THEOREM 5.1. *In Construction* 5.1*, assume we have $\hat{R}$ available servers and $R \leq \hat{R} \leq N$, we need*

$$L_{\textit{flex}} = \begin{cases} L_1, & \hat{R} \geq R_1, \\ \left(1 + \frac{R_{j-1} - \hat{R}}{p_j m_j n_j}\right) \sum\limits_{J=1}^{j-1} L_J, & R_j \leq \hat{R} < R_{j-1}, j \geq 2, \end{cases} \tag{5.26}$$

computation load at each server to obtain the final result, where

$$L_j = \begin{cases} \frac{\lambda \kappa \mu}{m_1 p_1 n_1}, & j = 1, \\ \frac{R_{j-1} - R_j}{p_j m_j n_j} \sum\limits_{J=1}^{j-1} L_J, & j \geq 2, \end{cases} \tag{5.27}$$

is the total computation load at each server in Layer j. The server storage size required in Layer j is

$$C_j = C_{j,A} + C_{j,B}, \tag{5.28}$$

where

$$C_{j,A} = \begin{cases} \frac{\lambda \kappa}{p_1 m_1}, & j = 1, \\ \frac{R_{j-1} - R_j}{p_j m_j} \sum\limits_{J=1}^{j-1} C_{J,A}, & j \geq 2. \end{cases} \tag{5.29}$$

$$C_{j,B} = \begin{cases} \frac{\kappa\mu}{p_1 n_1}, & j = 1, \\ \frac{R_{j-1}-R_j}{p_j n_j} \sum\limits_{J=1}^{j-1} C_{J,B}, & j \geq 2. \end{cases} \tag{5.30}$$

Proof. In the following, we first prove (5.27). Then, we show that with the computation load in (5.26), the central node is able to obtain the matrix product. At last, we prove the storage size required in each layer.

In Layer $j = 1$, from (5.15), we know that $f_1(\alpha_i; A^{(1,1)})$ and $f_1(\alpha_i; B^{(1,1)})$ have sizes $\frac{\lambda}{m_1} \times \frac{\kappa}{p_1}$ and $\frac{\kappa}{p_1} \times \frac{\mu}{n_1}$, respectively. Thus, the computation load in Layer 1 is

$$L_1 = \frac{\lambda\kappa\mu}{m_1 p_1 n_1}. \tag{5.31}$$

In Layer j, according to (5.15), (5.13), (5.14), and (5.22), the computation load of $\{\Gamma_{j,\delta_j}(\alpha_i) = f_j(\alpha_i; A^{(j,\delta_j)}) \cdot g_j(\alpha_i; B^{(j,\delta_j)}) : \delta_j \in [\gamma_j]\}$ is $1/(p_j m_j n_j)$ fraction of that of $\mathcal{X} \triangleq \{A^{(j,\delta_j)} \cdot B^{(j,\delta_j)} : \delta_j \in [\gamma_j]\}$. Moreover, the computation load of $\mathcal{Y}(J,t) \triangleq \{f_J(\alpha_{N+t} : A^{(J,\delta_J)}) \cdot f_J(\alpha_{N+t} : B^{(J,\delta_J)}) : \delta_J \in [\gamma_J]\}$ is equal to the load (per server) at the J-th layer, which is L_J. Based on (5.16), (5.17) and (5.18), the computation load of $\mathcal{X}$ is equal to the load of $\mathcal{Y}(J,t)$ for all $J \in [j-1]$, $R_j - R_a + 1 \leq t \leq R_{j-1} - R_a$, which is $(R_{j-1} - R_j) \sum\limits_{J=1}^{j-1} L_J$. Therefore, (5.27) is satisfied.

In the case that the number of available servers $\hat{R} \geq R_1$, according to the correctness of EP codes [128], the required results can be obtained by collecting R_1 evaluation points of $\Gamma_{1,1}(\alpha_i)$. Thus, we only need the computation in Layer 1.

In the case that $R_j \leq \hat{R} < R_{j-1}$, we first calculate all the tasks in Layers 1 to $j-1$, whose computation load is $\sum\limits_{J=1}^{j-1} L_J$. Then, in Layer j, Server i calculates $\frac{R_{j-1}-\hat{R}}{R_{j-1}-R_j}\gamma_j$ tasks, i.e., $\Gamma_{j,\delta_j}(\alpha_i), \delta_j = 1, 2, ..., \frac{R_{j-1}-\hat{R}}{R_{j-1}-R_j}\gamma_j, i \in [N]$. Thus, the total computation is $\left(1 + \frac{R_{j-1}-\hat{R}}{p_j m_j n_j}\right) \sum\limits_{J=1}^{j-1} L_J$.

Claim: R_J evaluations from $\{\Gamma_{J,\delta_J}(\alpha_i), i \in [N]\}$ can be obtained by the above calculations for $J = j, j-1, \ldots, 2, 1$.

We prove it by induction on J. As a consequence, the polynomial $\Gamma_{J,\delta_J}(\cdot)$ is decoded due to the correctness of EP codes [128]. Hence, the final result can be decoded with $J = 1$ and (5.26) is proved.

Base case: In Layer j, since $\hat{R} \geq R_j$, the claim holds trivially.

Induction step: Suppose the claim holds for Layers $j, j-1, \ldots, J+1$. We show that it will hold for Layer J. Note that $J < j$. The associated polynomials are decoded in Layers $j, j-1, \ldots, J+1$. Then, from Eqs. (5.16), (5.17) and (5.18), one can calculate $\Gamma_{J,\delta_J}(\alpha_{N+t})$, for $R_j - R_a + 1 \leq t \leq R_{j-1} - R_a - (\hat{R} - R_j)$ from Layer j and $R_{J'} - R_a + 1 \leq t \leq R_{J'-1} - R_a$ from Layers $J' = j-1, \ldots, J+1$. In total, $R_J - \hat{R}$ extra parities are obtained for the polynomial $\Gamma_{J,\delta_J}(\cdot)$. Thus, together with $\hat{R}$ available nodes, R_J evaluation points of $\Gamma_{J,\delta_J}(\cdot)$ are obtained, for all $\delta_J \in [\gamma_J]$.

The proof of the storage size is similar to the proof of (5.27). The proof sketch is as follows.

In Layer 1, the server needs to store $f_1(\alpha_i; A^{(1,1)}), f_1(\alpha_i; B^{(1,1)}), i \in [N]$, then

$$C_1 = \frac{1}{p_1}\left(\frac{\lambda\kappa}{m_1} + \frac{\kappa\mu}{n_1}\right). \tag{5.32}$$

In Layer $j \geq 2$, from (5.12), (5.17) and (5.18), the γ_j tasks in Layer j are encoded from the extra parities in Layers 1 to $j-1$. Based on (5.15), (5.13), (5.14), and (5.22), the size of $f_j(\alpha_i; A^{(j,\delta_j)}), i \in [N]$ is $p_j m_j$ fractions of $A^{(j,\delta_j)}$, and the size of $f_j(\alpha_i; B^{(j,\delta_j)}), i \in [N]$ is $p_j n_j$ fractions of $B^{(j,\delta_j)}$. Thus, (5.29) and (5.30) are obtained. □

REMARK 5.1. *In Fig.* 5.3*, partial computation results can be also utilized to accelerate the computation in several cases such that the nodes contribute different number of results depending on their speed. For example, when Servers* 1 *and* 2 *complete their first* 4 *tasks and Server* 3 *completes its first* 2 *tasks, we are able to obtain* $f_1(\alpha_i, A^{(1,1)})$ *for* $i = 1, 2, 3, 6, 7$*, thus obtain the final results. Similar partial results utilization can be found in our general constructions, but in this chapter we assume a server is either available or not able to provide any results.*

REMARK 5.2. *CSA codes and GCSA codes [55] are designed to handle batch processing of matrix multiplication, namely, the multiplication of two sequences of matrices. They also provide solutions for secure distributed computation. Combined with these codes, our construction can be easily modified to handle batch processing and secure distributed computation.*

The following corollary states a special case of the computation load that will be useful in the optimization discussed in Section 5.4 under the multi-round communication model.

COROLLARY 5.1. *In the case of $p_j = 1, j \geq 2$, we have $m_j n_j = R_j$ in Construction* 5.1. *The j-th layer's computation load of each server is*

$$L_j = \begin{cases} \frac{\lambda\kappa\mu}{m_1 p_1 n_1}, & j = 1, \\ \frac{R_1(R_{j-1}-R_j)}{R_{j-1}R_j} L_1, & j \geq 2, \end{cases} \tag{5.33}$$

and the total computation of each server is

$$L_{\mathit{flex}} = \begin{cases} L_1, & R_1 \leq \hat{R}, \\ \frac{R_1(R_j+R_{j-1}-\hat{R})}{R_{j-1}R_j} L_1, & R_j \leq \hat{R} < R_{j-1}, j \geq 2, \end{cases} \tag{5.34}$$

where $\hat{R}$ is the number of non-straggler servers. Specifically, when $\hat{R} = R_j$,

$$L_{\mathit{flex}} = \frac{R_1}{R_j} L_1. \tag{5.35}$$

Proof. We first prove (5.33) by induction.

Base case: When $j = 2$, we get $L_2 = \frac{R_1 - R_2}{R_2} L_1$ from (5.27) and it satisfies (5.33).

Induction step: Suppose $L_2, ..., L_j$ satisfy (5.33). From (5.27) and $p_j m_j n_j = R_j$ we know that

$$L_j = \frac{R_{j-1} - R_j}{R_j} \sum_{J=1}^{j-1} L_J. \tag{5.36}$$

Then, we have

$$\begin{aligned} L_{j+1} &= \frac{R_j - R_{j+1}}{R_{j+1}} \sum_{J=1}^{j-1} L_J + \frac{R_j - R_{j+1}}{R_{j+1}} L_j \\ &= \frac{R_j - R_{j+1}}{R_{j+1}} \frac{R_j}{R_{j-1} - R_j} L_j + \frac{R_j - R_{j+1}}{R_{j+1}} L_j \\ &= \frac{R_{j-1}(R_j - R_{j+1})}{R_{j+1}(R_{j-1} - R_j)} L_j \\ &= \frac{R_1(R_j - R_{j+1})}{R_j R_{j+1}} L_1, \end{aligned} \tag{5.37}$$

which satisfies (5.33).

Then, for the total computation, from (5.26) we can easily check that for $R_j \leq \hat{R} < R_{j-1}, j \in [a]$, we have

$$\begin{aligned} L_{\text{flex}} &= \left(1 + \frac{R_{j-1} - \hat{R}}{R_j}\right) \sum_{J=1}^{j-1} L_J \\ &= \left(1 + \frac{R_{j-1} - \hat{R}}{R_j}\right) \frac{R_j}{R_{j-1} - R_j} L_j \\ &= \frac{R_1(R_j + R_{j-1} - \hat{R})}{R_{j-1} R_j} L_1. \end{aligned} \tag{5.38}$$

The proof is completed. □

5.4 Computation Load Optimization

In this section, we discuss how to pick the matrix partition parameters and the recovery profile to optimize the computation load given the storage capacity. Under the one-round communication model, we find the optimal parameters for the 2-layer flexible construction and show that when the storage capacity is above a threshold, the flexible construction outperforms the fixed EP code. For the multi-round communication model, we show that all layers except the first layer reduce to block-wise matrix-vector multiplication, and when the straggler probability is small, the most number of layers is optimal.

Recall $\hat{R}$ is the number of non-straggler servers, $R \leq \hat{R} \leq N$. We consider the expected computation load over the realizations of $\hat{R}$. Assume for each instance of computing, $\hat{R}$ is independent and identically distributed. Denote q_j as the probability of j stragglers in the system. Formally,

$$q_j = P(\hat{R} = N - j), \ \ \forall j \in \{0, , 1, \cdots, N - R\}. \tag{5.39}$$

Here, R is chosen such that the probability of having more than $N - R$ stragglers is negligible. Therefore, j is assumed to be in the range between 0 and $N - R$, and

$$\sum_{j=0}^{N-R} q_j = 1. \tag{5.40}$$

The expectation of the computation load is

$$E[L_{\text{flex}}] = \sum_{j=0}^{N-R} q_j L_{\text{flex}}(\hat{R} = N - j), \tag{5.41}$$

where $L_{\text{flex}}(\hat{R})$ is the computation load for $\hat{R}$ non-straggler servers. The goal is to minimize $E[L_{\text{flex}}]$ over the partitioning parameters $p_j, m_j, n_j,\ j \in [a]$ and the recovery profile $\{R_1, \cdots, R_a\}$, given the recovery threshold $R_a = R$ and the storage constraint C in each server. Although in practical systems $p_j, m_j, n_j, R_j, j \in [a]$ are required to be integers, in this section, we only assume them as real numbers to simplify the optimization analysis. To find an integer solution (not necessarily optimal), we pick the parameters close to the optimal real values that satisfy the recovery threshold and the storage constraint.

5.4.1 Optimization on Entangled Polynomial codes

As a warm-up, let us start with an EP code with a fixed recovery threshold R, which satisfies $R = m_0p_0n_0 + p_0 - 1$ according to [128], for some undetermined partition parameters p_0, m_0, n_0. The computation load of EP codes remains the same if the number of stragglers is no greater than $N - R$. According to [128], the computation load and the required storage size are

$$L_{\text{EP}}(\hat{R}) = \frac{\lambda\kappa\mu}{m_0p_0n_0}, \hat{R} \geq R, \tag{5.42}$$

$$C_{\text{EP}} = \frac{1}{p_0}\left(\frac{\lambda\kappa}{m_0} + \frac{\kappa\mu}{n_0}\right). \tag{5.43}$$

Thus, the optimization problem can be formulated as

$$\begin{aligned} \min_{p_0,m_0,n_0} \quad & L_{\text{EP}} = \frac{\lambda\kappa\mu}{m_0p_0n_0}, \\ \text{s.t.} \quad & R = p_0m_0n_0 + p_0 - 1, \\ & \frac{\lambda\kappa}{p_0m_0} + \frac{\kappa\mu}{p_0n_0} \leq C. \end{aligned} \tag{5.44}$$

THEOREM 5.2. *The solution of the EP code optimization problem in* (5.44) *is*

$$L_{EP}^* = \frac{2C\lambda\kappa\mu}{C(R+1) + \sqrt{C^2(R+1)^2 - 16\lambda\kappa^2\mu}} \tag{5.45}$$

with

$$p_0^* = \frac{1}{2}(R+1) - \frac{1}{2}\sqrt{(R+1)^2 - 16\frac{\lambda\kappa^2\mu}{C^2}}, \tag{5.46}$$

and m_0^*, n_0^* *are given by* $m_0^*n_0^* = \frac{R+1}{p_0^*} - 1$ *and* $\lambda\kappa n_0^* = \kappa\mu m_0^*$.

Proof. Using the threshold constraint

$$m_0 n_0 = \frac{R+1}{p_0} - 1, \tag{5.47}$$

we have $L_{\text{EP}} = \frac{\lambda\kappa\mu}{R+1-p_0}$, which is an increasing function of p_0. So, we minimize p_0 under the constraint that

$$\frac{\lambda\kappa n_0 + \kappa\mu m_0}{R+1-p_0} \leq C. \tag{5.48}$$

Note that

$$\lambda\kappa n_0 + \kappa\mu m_0 \geq 2\sqrt{\lambda\kappa^2\mu m_0 n_0} = 2\sqrt{\lambda\kappa^2\mu\frac{R+1-p_0}{p_0}} \tag{5.49}$$

and it holds with equality if and only if $\lambda\kappa n_0 = \kappa\mu m_0$. Combining (5.48) with (5.49) results in

$$2\sqrt{\frac{\lambda\kappa^2\mu}{(R+1-p_0)p_0}} \leq C. \tag{5.50}$$

Note that $2\sqrt{\frac{\lambda\kappa^2\mu}{(R+1-p_0)p_0}}$ decreases with p_0 because the derivative

$$\frac{\mathrm{d}\,(R+1-p_0)p_0}{\mathrm{d}\,p_0} = R+1-2p_0 = p_0 m_0 n_0 - p_0 \geq 0. \tag{5.51}$$

Therefore, L_{EP} reaches its optimal value when $\lambda\kappa n_0 = \kappa\mu m_0$ and (5.50) holds with equality, i.e., $p_0^* = \frac{1}{2}(R+1) - \frac{1}{2}\sqrt{(R+1)^2 - 16\frac{\lambda\kappa^2\mu}{C^2}}$. As a result, m_0^*, n_0^* can be obtained by $m_0^* n_0^* = \frac{R+1}{p_0^*} - 1$ and $\lambda\kappa n_0^* = \kappa\mu m_0^*$. The optimal computation load is

$$L_{EP}^* = \frac{2C\lambda\kappa\mu}{C(R+1) + \sqrt{C^2(R+1)^2 - 16\lambda\kappa^2\mu}}. \tag{5.52}$$

The theorem is proved. □

REMARK 5.3. *In Theorem* 5.2, *the storage capacity is required to satisfy*

$$C \geq \frac{4\kappa\sqrt{\lambda\mu}}{1+R}. \tag{5.53}$$

to have a valid L^*_{EP} *in* (5.45). *In addition, by combining* (5.48) *and* (5.49), *we obtain the minimum storage required as* $2\sqrt{\frac{\lambda\kappa^2\mu}{(R+1-p_0)p_0}}$ *in* (5.50). *Since* $p_0 = \frac{R+1}{m_0n_0+1}$ *and* m_0, n_0 *are at least 1, we conclude that* (5.53) *is the minimum storage constraint requirement to use EP codes for distributed matrix multiplication.*

5.4.2 Optimization for the one-round communication model

Next, we consider the flexible constructions with the one-round communication model. In this model, all the tasks are sent to the server in one communication round. Thus, the sum of the task sizes should not exceed the storage constraint. Since the more layers, the larger the total size of the tasks is, only the 2-layer construction is considered. We first optimize the partition parameters with predetermined $R_1, R_2 = R$. After that, R_1 is optimized.

By the expression of the computation load in Theorem 5.1, the expectation of the computation load in (5.41) becomes

$$E[L_{\text{flex}}] = \sum_{j=0}^{N-R_2} q_j \frac{\lambda\kappa\mu}{p_1m_1n_1} + \sum_{j=N-R_1+1}^{N-R_2} q_j \frac{\lambda\kappa\mu(R_1+j-N)}{m_1m_2p_1p_2n_1n_2}. \tag{5.54}$$

In practical systems, the probability of having many stragglers is usually small. For instance, less than 110 failures occur over a 3000-node production cluster of Facebook per day [94]. So, we ignore the second term in (5.54) and use the approximation $L_{\text{flex}} = \frac{\lambda\kappa\mu}{p_1m_1n_1}$ in our optimization problem. Combined with (5.28), the optimization problem can be formulated as

$$\begin{aligned} \min_{p_1,m_1,n_1,p_2,m_2,n_2} & L_{\text{flex}} = \frac{\lambda\kappa\mu}{p_1 m_1 n_1}, \\ \text{s.t.} \quad & R_1 = p_1 m_1 n_1 + p_1 - 1, \\ & R_2 = p_2 m_2 n_2 + p_2 - 1, \\ & \frac{1}{p_1}\left(\frac{\lambda\kappa}{m_1} + \frac{\kappa\mu}{n_1}\right) + \frac{(R_1 - R_2)}{p_1 p_2}\left(\frac{\lambda\kappa}{m_1 m_2} + \frac{\kappa\mu}{n_1 n_2}\right) \le C. \end{aligned} \tag{5.55}$$

It should be noted that when $R_1 = R_2 = R$, the 2-layer flexible construction reduces to the fixed EP code and the optimal partition parameters remain the same.

THEOREM 5.3. *Fix $R_1, R_2 = R$. The solution of the 2-layer flexible construction optimization* (5.55) *is*

$$L^*_{\mathit{flex}} = \frac{2C(R_2+1)\lambda\kappa\mu}{C(R_1+1)(R_2+1) + \sqrt{C^2(R_1+1)^2(R_2+1)^2 - 16\lambda\kappa^2\mu(2R_1 - R_2 + 1)^2}}, \tag{5.56}$$

with

$$p_1^* = \frac{1}{2}(R_1+1) - \frac{1}{2}\sqrt{(R_1+1)^2 - \frac{16\lambda\kappa^2\mu(2R_1 - R_2 + 1)^2}{C^2(R_2+1)^2}}, \tag{5.57}$$

and m_1^*, n_1^* are given by $m_1^* n_1^* = \frac{R_1+1}{p_1^*} - 1$ and $\lambda\kappa n_1^* = \kappa\mu m_1^*$, and $p_2^* = \frac{R_2+1}{2}, m_2^* = 1, n_2^* = 1$.

Proof. Using $m_1 n_1 = \frac{R_1+1}{p_1} - 1$, we have $L_{\text{flex}} = \frac{\lambda\kappa\mu}{R_1+1-p_1}$, which is an increasing function of p_1. Therefore, to maximize L_{flex} we need to minimize p_1.

Using $m_1 n_1 = \frac{R_1+1}{p_1} - 1$ and $m_2 n_2 = \frac{R_2+1}{p_2} - 1$, similar to (5.48) and (5.49), we have:

$$\begin{aligned} & C \\ \ge & \frac{1}{p_1}\left(\frac{\lambda\kappa}{m_1} + \frac{\kappa\mu}{n_1}\right) + \frac{(R_1 - R_2)}{p_1 p_2}\left(\frac{\lambda\kappa}{m_1 m_2} + \frac{\kappa\mu}{n_1 n_2}\right) \end{aligned} \tag{5.58}$$

$$\geq 2\sqrt{\frac{\lambda\kappa^2\mu}{(R_1+1-p_1)p_1}} + 2(R_1-R_2)\sqrt{\frac{\lambda\kappa^2\mu}{(R_1+1-p_1)(R_2+1-p_2)p_1p_2}}. \tag{5.59}$$

Here (5.58) holds with equality when $\lambda\kappa n_1 = \kappa\mu m_1$ and $\lambda\kappa n_1 n_2 = \kappa\mu m_1 m_2$ or $n_2 = m_2$. Similar to (5.51), it is easy to show that (5.59) is a decreasing function of p_1 and p_2. For any fixed p_2, to obtain the minimum p_1, we should set (5.59) equal to C. When (5.59) is fixed, p_1 is minimized when p_2 reaches its maximum because a bigger p_2 results in a smaller p_1 when (5.59) is equal to C. Noticing that $p_2 = \frac{R_2+1}{m_2n_2+1}$ and m_2, n_2 are at least 1, we set $p_2^* = \frac{R_2+1}{2}, m_2^* = 1, n_2^* = 1$. The optimal p_1^* and L^*_{flex} are obtained accordingly. □

If R_2 is odd, then the choices of p_2, m_2, n_2 in the above theorem are the exact optimal integer parameters.

REMARK 5.4. *In Theorem* 5.3*, the storage capacity is required to satisfy*

$$C \geq \frac{4\kappa\sqrt{\lambda\mu}(2R_1-R_2+1)}{(1+R_2)(1+R_1)} \tag{5.60}$$

to have a valid $L^*_{\textit{flex}}$ in (5.56)*. In addition, we obtain the minimum storage required in* (5.59)*. Since $p_1 = \frac{R_1+1}{m_1n_1+1}, p_2 = \frac{R_2+1}{m_2n_2+1}$, and m_1, n_1, m_2, n_2 are at least 1, we conclude that* (5.60) *is the minimum storage constraint requirement to use our 2-layer flexible codes for the distributed matrix multiplication. When $R_1 = R_2$,* (5.60) *is the same as* (5.53).

Next, we provide an example for the optimal integer solutions of the partition parameters.

EXAMPLE 5.3. *Assume there are $N = 8$ servers and we need to tolerate $N - R = 1$ straggler. $\lambda = \kappa = \mu$ and the storage size of each server is limited by $C = \frac{8}{7}\lambda\kappa$. Using the EP code, the optimal choice of $\{p_0, m_0, n_0\}$ is $\{1, 1, 7\}$, which results in a storage size of $\frac{8}{7}\lambda\kappa$ and a computation load per server of $\frac{1}{7}\lambda\kappa\mu = 0.143\lambda\kappa\mu$. Using the 2-layer flexible construction with $R_1 = 8$ and $R_2 = 7$, the optimal parameters are chosen as $p_1 = 1, m_1 = 2, n_1 = 4, p_2 = 4, m_2 = 1, n_2 = 1$, which cost a storage size of $\frac{15}{16}\lambda\kappa$ and a computation load of $\frac{1}{8}\lambda\kappa\mu$ when there is no straggler, with* *an additional computation load of $\frac{1}{32}\lambda\kappa\mu$ when there is one straggler. Assuming the probability of one straggler to be 10%, the average computation load is $0.128\lambda\kappa\mu$. In this example, we save both storage size and average computation load while maintaining one straggler tolerance.*

Having found the best computation load for a fixed recovery profile as in Theorem 5.2, next, we discuss the optimization of the recovery profile. Given the straggler tolerance level $N - R_2$, we just need to optimize R_1, such that $R_2 \le R_1 \le N$.

THEOREM 5.4. *To minimize the 2-layer computation load L^*_{flex} in (5.56), the optimal R_1^* is*

$$R_1^* = \begin{cases} N, & C \ge \frac{8\kappa\sqrt{\lambda\mu}}{R_2+1}, \\ \min\left(N, \frac{C^2(R_2+1)^2(R_2+3)+64\lambda\kappa^2\mu(R_2-1)}{2(64\lambda\kappa^2\mu - C^2(R_2+1)^2)}\right), & \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}} < C < \frac{8\kappa\sqrt{\lambda\mu}}{R_2+1}, \\ R_2, & C \le \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}}. \end{cases} \tag{5.61}$$

Proof. The optimal computation given R_1 is shown in (5.56). Since the numerator is a constant not related to R_1, we set Y as the denominator and L^*_{flex} has the minimum value when Y reaches its maximum.

$$\frac{dY}{dR_1} = C(R_2+1) + \frac{C^2(R_2+1)^2(R_1+1) - 32\lambda\kappa^2\mu(2R_1 - R_2 + 1)}{\sqrt{C^2(R_1+1)^2(R_2+1)^2 - 16\lambda\kappa^2\mu(2R_1 - R_2 + 1)^2}}. \tag{5.62}$$

Setting $\frac{dY}{dR_1} = 0$, we have

$$\left(\frac{32\lambda\kappa^2\mu(2R_1 - R_2 + 1)}{C(R_2+1)}\right)^2 = 64\lambda\kappa^2\mu(2R_1 - R_2 + 1)(R_1+1) - 16\lambda\kappa^2\mu(2R_1 - R_2 + 1)^2. \tag{5.63}$$

Since $R_1 \ge R_2$, the term $\lambda\kappa^2\mu(2R_1 - R_2 + 1) \ne 0$ can be cancelled and the solution to the above equation is

$$\hat{R}_1 \triangleq \frac{C^2(R_2+1)^2(R_2+3) + 64\lambda\kappa^2\mu(R_2-1)}{2(64\lambda\kappa^2\mu - C^2(R_2+1)^2)}. \tag{5.64}$$

Let $X = C^2(R_1+1)^2(R_2+1)^2 - 16\lambda\kappa^2\mu(2R_1 - R_2 + 1)^2$. We have $X \geq 0$ due to the minimum storage constraint in Remark 5.4. We simplify (5.62) as

$$\frac{dY}{dR_1} = C(R_2+1) + \frac{\frac{dX}{dR_1}}{2\sqrt{X}}, \tag{5.65}$$

where

$$\frac{dX}{dR_1} = 2(C^2(R_2+1)^2 - 64\lambda\kappa^2\mu)R_1 + 2C^2(R_2+1)^2 + 64\lambda\kappa^2\mu(R_2-1) \tag{5.66}$$

is a linear function of R_1 and the constant term $2C^2(R_2+1)^2 + 64\lambda\kappa^2\mu(R_2-1) > 0$ since R_2 is at least 1.

In the case of $C \geq \frac{8\kappa\sqrt{\lambda\mu}}{R_2+1}$, we have

$$C^2(R_2+1)^2 - 64\lambda\kappa^2\mu \geq 0 \Rightarrow \frac{dX}{dR_1} > 0 \Rightarrow \frac{dY}{dR_1} > 0. \tag{5.67}$$

Thus, we should pick $R_1^* = N$.

In the case of $C < \frac{8\kappa\sqrt{\lambda\mu}}{R_2+1}$, we get

$$C^2(R_2+1)^2 - 64\lambda\kappa^2\mu < 0 \Rightarrow \frac{dX}{dR_1} \text{ is a decreasing linear function.} \tag{5.68}$$

We discuss $\frac{dY}{dR_1}$ when R_1 varies between $(0, \frac{C^2(R_2+1)^2+32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu - C^2(R_2+1)^2}]$ and $(\frac{C^2(R_2+1)^2+32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu - C^2(R_2+1)^2}, +\infty)$ separately. In the first region, we have

$$R_1 \leq \frac{C^2(R_2+1)^2 + 32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu - C^2(R_2+1)^2} \Rightarrow \frac{dX}{dR_1} \geq 0 \Rightarrow \frac{dY}{dR_1} > 0, \tag{5.69}$$

Y reach its maximum when $R_1 = \frac{C^2(R_2+1)^2+32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu - C^2(R_2+1)^2}$. In the second region, we have

$$R_1 > \frac{C^2(R_2+1)^2 + 32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu - C^2(R_2+1)^2} \Rightarrow \frac{dX}{dR_1} < 0. \tag{5.70}$$

Clearly, $\frac{\frac{dX}{dR_1}}{2\sqrt{X}}$ is a decreasing function of R_1, because $\frac{dX}{dR_1}$ is a negative decreasing function of R_1 by (5.68) and (5.70), and $\sqrt{X}$ is a positive decreasing function of R_1 by (5.70). Then, with (5.65) we can conclude that

$$\frac{d^2Y}{dR_1^2} = \frac{d\frac{\frac{dX}{dR_1}}{2\sqrt{X}}}{dR_1} < 0. \tag{5.71}$$

In addition, we know from (5.64) that $\hat{R}_1$ is located in $(\frac{C^2(R_2+1)^2+32\lambda\kappa^2\mu}{64\lambda\kappa^2\mu-C^2(R_2+1)^2}, +\infty)$ when $R_2 \geq 2$, and hence is a local maximum. Therefore, combining the 2 ranges of R_1, we conclude that Y reaches its maximum in (5.64). Finally, the proof is completed considering the requirement that $R_2 \leq R_1 \leq N$, and the fact that $\hat{R}_1 \geq R_2$ is satisfied when $C \geq \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}}$. □

COROLLARY 5.2. *The flexible construction with 2 layers is better than a fixed EP code in terms of the computation load when the storage constraint C satisfy:*

$$C > \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}}. \tag{5.72}$$

Proof. From Theorems 5.2 and 5.3 we have

$$L^*_{\text{flex}}|_{R_1=R_2} = L^*_{\text{EP}}. \tag{5.73}$$

Also, it is easy to check that $R_1^* > R_2$ in (5.61) when (5.72) is satisfied. Then, combining Theorem 5.4 we conclude that

$$L^*_{\text{flex}}|_{R_1=R_1^*} < L^*_{\text{flex}}|_{R_1=R_2} = L^*_{\text{EP}}. \tag{5.74}$$

The proof is completed. □

We summarize how to choose the optimal constructions in different situations in Table 5.3.

Table 5.3: Optimal choices of the flexible constructions given the number of servers N, the failure tolerance $N-R$ and the storage constraint C.

Storage constraint C	Optimal constructions	Optimal matrix partition
$C < \frac{4\kappa\sqrt{\lambda\mu}}{1+R_2}$	Not available	Not available
$\frac{4\kappa\sqrt{\lambda\mu}}{1+R_2} \le C \le \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}}$	Fixed EP codes	p_0, m_0, n_0 chosen in Theorem 5.2
$C > \frac{8\kappa}{R_2+1}\sqrt{\frac{\lambda\mu}{3}}$	Flexible codes with R_1 chosen in Theorem 5.4	$p_j, m_j, n_j, j \in [2]$ chosen in Theorem 5.3

Fig. 5.4 shows a comparison of our 2-layer flexible codes and the fixed EP codes. For the approximate computation load, only the computation load in the first layer is considered as in (5.55). The expected computation load is computed based on (5.54) with a truncated binomial distribution

$$q_j = \frac{1}{\theta}\binom{N}{j}(1-\epsilon)^{N-j}\epsilon^j, 0 \le j \le N-R, \tag{5.75}$$

where $\epsilon = 0.05$ is the probability that each server is a straggler. To limit the number of stragglers below $N-R$, we truncate the binomial distribution below $N-R$ and $\theta = \sum_{i=0}^{N-R} \binom{N}{i}\epsilon^i(1-\epsilon)^{N-i}$ is the probability that there are at least $N-R$ available nodes.[2] In Fig. 5.4, we have $1-\theta < 10^{-4}$. The minimum required storage constraint is $C = 0.33$. When $C < 0.45$, our 2-layer construction reduces to the EP code. When the storage constraint $C \ge 0.45$, our 2-layer constructions have better performance. For example, when $C = 0.9$, the optimal EP code has $p_0 = 2, m_0 = 1, n_0 = 5$ and its expected computation is $L_{\text{EP}} = 0.1$. However, our 2-layer optimal flexible code has $p_1 = 1, m_1 = 3, n_1 = 5, p_2 = 6, m_2 = 1, n_2 = 1, R_1 = 15$, and its expected computation load is $L_{\text{flex}} = 0.069$, i.e., we save more than 30% in terms of computation load. In addition, the approximate computation load of the 2-layer flexible code in this case is 0.067, which is very close to the expected computation load when the computation in both layers are considered.

5.4.3 Optimization for the multi-round communication model

Now let us consider the multi-round communication model where coded matrices are sent sequentially from the source to the server. In this case, it is only required that the maximum size of the

[2] In practice, R is chosen such that the probability of having more than $N-R$ stragglers is negligible.

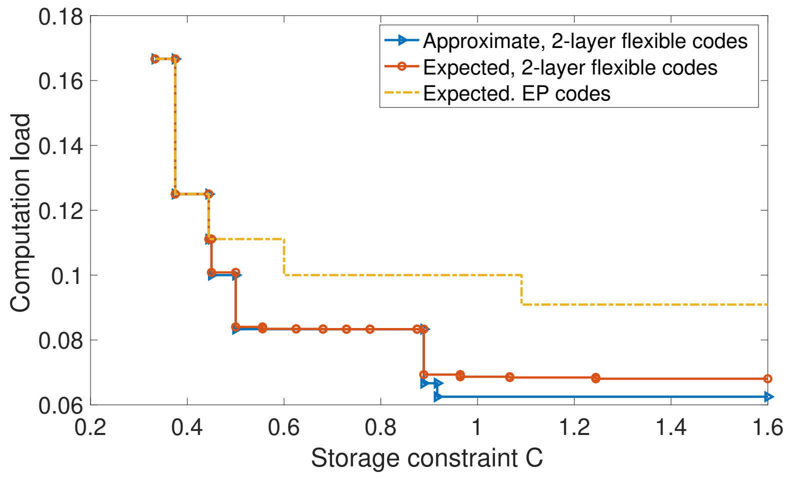


Figure 5.4: Computation load comparison of our 2-layer flexible codes and fixed EP codes under different storage constraint. $N = 16, R = 11$, and $\lambda = \kappa = \mu = 1$ unit. Each server has a failure probability of 0.05. The approximate computation load in (5.56) and the expected computation load in (5.54) are both shown in the figure. For all cases, the optimal matrix partitioning and the recovery profile are found by exhaustive search.

coded matrices does not exceed the storage size. As mentioned before Theorem 5.1, the storage capacity just needs to exceed the size of the first pair of coded matrices. We first optimize the partitioning parameters for a fixed recovery profile and then optimize the number of layers and the recovery profile.

Let us consider the construction with a layers and predetermined $R_j, j \in [a]$ such that $N \geq R_1 > R_2 > ... > R_a = R$. Assuming $R_j, j \in [a]$, and the storage constraint C are given, we first minimize the computation load in each layer:

$$\begin{aligned} \min_{p_j, m_j, n_j} \quad & L_j \\ \text{s.t.} \quad & R_j = p_j m_j n_j + p_j - 1 \\ & \frac{\lambda\kappa}{p_1 m_1} + \frac{\kappa\mu}{p_1 n_1} \leq C, \end{aligned} \tag{5.76}$$

where L_j is shown in (5.27). Note that once $L_j, j \in [a]$, are minimized, by Theorem 5.1 the computation load $L_{\text{flex}}(\hat{R})$ for any number of non-stragglers $\hat{R}$ is also minimized. Hence the optimization in (5.76) is stronger than optimizing the expected computation load defined in (5.41).

THEOREM 5.5. *The optimal solution of* (5.76) *for the flexible construction under the multi-round communication model is*

$$L_j^* = \begin{cases} \frac{2C\lambda\kappa\mu}{C(R+1)+\sqrt{C^2(R+1)^2-16\lambda\kappa^2\mu}}, & j=1, \\ \frac{R_1(R_{j-1}-R_j)}{R_{j-1}R_j}L_1, & j\geq 2, \end{cases} \tag{5.77}$$

with

$$p_1^* = \frac{1}{2}(R+1) - \frac{1}{2}\sqrt{(R+1)^2 - \frac{16\lambda\kappa^2\mu}{C^2}}, \tag{5.78}$$

m_1^*, n_1^* *are given by* $m_1^*n_1^* = \frac{R+1}{p_1^*} - 1$ *and* $\lambda\kappa n_1^* = \kappa\mu m_1^*$ *and* $p_j^* = 1, m_j^*n_j^* = R_j$ *for* $j \geq 2$.

Proof. When $j=1$, it is the same optimization problem in Theorem 5.2.

For $j \geq 2$, We prove by induction.

Base case: For $j=2$, since there is no constraint on storage size of Layer 2, by Theorem 5.1 we have

$$L_2 = \frac{R_1 - R_2}{p_2 m_2 n_2} L_1 = \frac{R_1 - R_2}{R_2 - p_2 + 1} L_1, \tag{5.79}$$

which is an increasing function of p_2. Thus, we have $p_2^* = 1, m_2^*n_2^* = R_2$.

Induction step: Assume the minimum L_J^* is achieved when $p_J^* = 1, m_J^*n_J^* = R_J$ for $J = 2, 3, ..., j-1$. For $J = j$, we have

$$L_j = \frac{R_{j-1} - R_j}{p_j m_j n_j} \sum_{J=1}^{j-1} L_J = \frac{R_{j-1} - R_j}{R_j - p_j + 1} \sum_{J=1}^{j-1} L_J, \tag{5.80}$$

which is an increasing function of p_j and $L_J, J \in [j-1]$, respectively. Hence, we should pick the minimum $p_j^* = 1$ and the minimum $L_J^*, 1 \leq J \leq j-1$ to optimize L_j. Therefore, $p_J^* = 1, m_J^*n_J^* =$

R_J for all $2 \leq J \leq j$. $\square$

Notice that in the case of $j \geq 2$, we have $R_j = m_j n_j$, there is at least one integer solution with $m_j = R_j, n_j = 1$, which simplifies the problem to be matrix-vector multiplication.

Next, we discuss how to set the number of layers and the recovery profile to minimize the computation load. First, we state a lemma to show that adding more layers does not increase the computation load. Then, a theorem is proposed to show how to set R_1.

LEMMA 5.1. *Given R, R_1, and $p_j = 1, j \geq 2$, adding another layer does not increase the computation load of each server.*

Proof. Let us add a layer between Layers $j-1$ and j. When $R_{j-1} = R_j + 1$, no layers can be added. When $R_{j-1} > R_j + 1$, consider adding one extra layer with $R_{\text{add}} = R_j + 1$ between Layer $j-1$ and j so that $R_j < R_{\text{add}} < R_{j-1}$. Denote the computation load of the new construction by L_{add}, which is a function of the number of non-stragglers, $\hat{R}$.

When $\hat{R} \leq R_j$ or $\hat{R} \geq R_{j-1}$, based on (5.34), the computation load of each server does not change, i.e., $L_{\text{add}} = L_{\text{flex}}$.

When $R_{\text{add}} \leq \hat{R} < R_{j-1}$, by Corollary 5.1, the new computation load is

$$\begin{aligned}
L_{\text{add}} &= \frac{R_1(R_{\text{add}} + R_{j-1} - \hat{R})}{R_{j-1}R_{\text{add}}} L_1 \\
&= \frac{R_1}{R_{j-1}} L_1 + \frac{R_1(R_{j-1} - \hat{R})}{R_{j-1}R_{\text{add}}} L_1, \\
&< \frac{R_1}{R_{j-1}} L_1 + \frac{R_1(R_{j-1} - \hat{R})}{R_{j-1}R_j} L_1 \\
&= L_{\text{flex}}, \qquad (5.81)
\end{aligned}$$

where the inequality results from the fact that $R_{\text{add}} > R_j$. Thus, by adding one layer with $R_{\text{add}} = R_j + 1$, computation load does not increase. Similarly, more layers can be added between Layer

$j-1$ and j. Therefore, adding more layers between R_1 and R does not increase the computation load of each server. $\square$

Based on Lemma 5.1, given R_1 and R, the optimal scheme is to add one layer for each value between R_1 and R. Thus, the recovery profile should be chosen to be $(R_1, R_1-1, R_1-2, \ldots, R)$. The only problem left is how to set R_1. According to Corollary 5.1 and Theorem 5.5,

$$L_{\text{flex}} = \begin{cases} L_1 = \frac{2\lambda\kappa\mu}{(R+1)+\sqrt{(R+1)^2-\frac{16\lambda\kappa^2\mu}{C^2}}}, & \text{if } \hat{R} > R_1, \\ \frac{R_1}{N-j}L_1, & \text{if } \hat{R} = N-j, N-R_1 \le j \le N-R. \end{cases} \tag{5.82}$$

Based on (5.82), we see $\frac{\lambda\kappa\mu}{R_1+1} < L_1 < \frac{2\lambda\kappa\mu}{R_1+1}$. Denote $L_1 = \eta\frac{\lambda\kappa\mu}{R_1+1}$, where

$$\eta = \frac{2}{1+\sqrt{1-\frac{16\lambda\kappa^2\mu}{C^2(1+R_1)^2}}}. \tag{5.83}$$

Note that given λ, κ, μ, C, the value of η decreases as R_1 increases. Then, for fixed R_1, the expectation of the computation load is

$$E\left[L_{\text{flex}}^{(R_1)}\right] = L_1 \sum_{j=0}^{N-R_1-1} q_j + \sum_{j=N-R_1}^{N-R} q_j \frac{R_1}{N-j} L_1 \tag{5.84}$$

$$= \lambda\kappa\mu\eta \left(\sum_{j=0}^{N-R_1-1} \frac{q_j}{1+R_1} + \sum_{j=N-R_1}^{N-R} \frac{q_j R_1}{(N-j)(1+R_1)} \right). \tag{5.85}$$

Here, the superscript R_1 indicates that the computation load depends on R_1. The goal is to minimize $E\left[L_{\text{flex}}^{(R_1)}\right]$ over R_1 where $R \le R_1 \le N$.

The theorem below states a sufficient condition for which we should set $R_1 = N$ and use the maximum number of layers. In particular, the recovery profile should be $(N, N-1, N-2, \ldots, R)$.

Theorem 5.6. *When $q_0 > \sum_{j=1}^{N-R} \frac{q_j}{N-j}$, the optimal R_1 to minimize* (5.85) *is achieved when $R_1^* = N$.*

Proof. Denote the term in the parentheses of (5.85) as

$$h(R_1) = \frac{1}{1+R_1}\sum_{j=0}^{N-R_1-1} q_j + \frac{R_1}{1+R_1}\sum_{j=N-R_1}^{N-R} \frac{q_j}{N-j}. \tag{5.86}$$

When $R_1 = N$,

$$h(N) = \frac{N}{1+N}\sum_{j=0}^{N-R} \frac{q_j}{N-j}. \tag{5.87}$$

When $R_1 = N-k, k \in [1, N-R]$,

$$h(N-k) = \frac{1}{N-k+1}\sum_{j=0}^{k-1} q_j + \frac{N-k}{N-k+1}\sum_{j=k}^{N-R} \frac{q_j}{N-j}. \tag{5.88}$$

Then, their difference is

$$h(N-k) - h(N) \tag{5.89}$$

$$= \frac{1}{N-k+1}\sum_{j=0}^{k-1} q_j + \frac{N-k}{N-k+1}\sum_{j=k}^{N-R} \frac{q_j}{N-j} - \frac{N}{1+N}\sum_{j=0}^{N-R} \frac{q_j}{N-j} \tag{5.90}$$

$$= \frac{1}{(N+1)(N-k+1)}\left(\sum_{j=0}^{k-1} \frac{(k-j)N-j}{N-j} q_j - k\sum_{j=k}^{N-R} \frac{q_j}{N-j}\right). \tag{5.91}$$

When $q_0 > \sum_{j=1}^{N-R} \frac{q_j}{N-j}$, the first term in the parentheses of (5.91) is

$$\sum_{j=0}^{k-1} \frac{(k-j)N-j}{N-j} q_j \geq k q_0 > k\sum_{j=1}^{N-R} \frac{q_j}{N-j} \geq k\sum_{j=k}^{N-R} \frac{q_j}{N-j}. \tag{5.92}$$

Thus, $h(N-k) > h(N)$. Since η increases as R_1 decreases, by (5.85) we conclude that $E\left[L_{\text{flex}}^{(N)}\right] < E\left[L_{\text{flex}}^{(N-k)}\right]$. Therefore, R_1^* should be set as N. $\square$

EXAMPLE 5.4. *Suppose* $N = 50, R = 40$ *and assume the number of stragglers follows a truncated binomial distribution similar to* (5.75)*, i.e.,* $q_j = \theta\binom{N}{j}\epsilon^j(1-\epsilon)^{N-j}$*, for the constant factor* $\theta =$

$\frac{1}{\sum_{i=0}^{N-R}\binom{N}{i}\epsilon^i(1-\epsilon)^{N-i}}$. *According to Theorem* 5.6, R_1 *can be set as* N *as long as* $\epsilon < 7.4\%$.

5.5 Conclusion

In this chapter, we consider coded distributed matrix multiplication. A flexible construction for distributed matrix multiplication is proposed and the optimal parameters are discussed. The construction can also be generalized to batch processing of matrix multiplication and secure distributed computation. Flexible constructions are also found in other problems such as communication-efficient secret sharing [51], adaptive gradient codes [78], coded elastic computing [120] and flexible storage [52, 76]. It is worthwhile to explore more applications of flexible constructions, such as distributed machine learning and secure multi-party computation.

Chapter 6

Conclusion

In this dissertation, we focus on the privacy, security, and flexibility of distributed computing. The fundamental trade-offs are investigated. We characterized the capacities of T-private information retrieval with private side information, symmetric T-private information retrieval with private side information, and private search. These works lead us to the fundamental understanding of the roles of side information and dependence in distributed computing. We proposed GCSA codes with noise alignment for secure coded multi-party batch matrix multiplication and a flexible construction for coded distributed matrix multiplication. These schemes can be generalized to other distributed computation problems, such as multi-source distributed computing. To conclude the dissertation, we present several promising directions for future work.

There are many constructions for different distributed computing scenarios. However, the converse arguments for distributed computing are not known in most cases. Exploring converse arguments for distributed computing is an interesting direction for future work. For example, private search is the first step to explore the PIR with dependent messages. We propose a loose bound for DPIR, but the tight bound of general DPIR is still unknown. Another example is distributed matrix multiplication. Recently, there are a large number of works on the schemes of coded distributed matrix multiplication. However, there are rarely works on the converse bounds. For a distributed matrix multiplication problem, Yu et al. [128] proposed a loose bound for the optimum linear recovery threshold with the bilinear complexity. It would be interesting to find converse arguments for more coded matrix multiplication problems.

For flexible constructions, they are highly related to the problems such as communication efficient secret sharing [51], adaptive gradient codes [78], coded elastic computing [120] and flexible storage [76]. It offers a fertile research landscape for discovering new coding structures and converse arguments. It is also worthwhile to explore more applications of flexible constructions, like distributed machine learning, DNA storage, and consistent data storage.

Bibliography

[1] M. Aliasgari, O. Simeone, and J. Kliewer. Distributed and private coded matrix computation with flexible communication load. *arXiv preprint arXiv:1901.07705*, 2019.

[2] M. M. Amiri and D. Gündüz. Computation scheduling for distributed machine learning with straggling workers. *IEEE Transactions on Signal Processing*, 67(24):6270–6284, 2019.

[3] G. Ananthanarayanan, S. Kandula, A. Greenberg, I. Stoica, Y. Lu, B. Saha, and E. Harris. Reining in the outliers in map-reduce clusters using mantri. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI'10, page 265–278, USA, 2010. USENIX Association.

[4] T. Baharav, K. Lee, O. Ocal, and K. Ramchandran. Straggler-proofing massive-scale distributed matrix multiplication with d-dimensional product codes. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1993–1997. IEEE, 2018.

[5] K. Banawan and S. Ulukus. Multi-message private information retrieval: Capacity results and near-optimal schemes. *arXiv preprint arXiv:1702.01739*, 2017.

[6] K. Banawan and S. Ulukus. Multi-message private information retrieval: Capacity results and near-optimal schemes. *IEEE Transactions on Information Theory*, 2018.

[7] K. Banawan and S. Ulukus. The Capacity of Private Information Retrieval from Coded Databases. *IEEE Transactions on Information Theory*, 64(3):1945–1956, 2018.

[8] K. Banawan and S. Ulukus. The capacity of private information retrieval from byzantine and colluding databases. *IEEE Transactions on Information Theory*, 65(2):1206–1219, Feb 2019.

[9] A. Beimel, Y. Ishai, and E. Kushilevitz. General constructions for information-theoretic private information retrieval. *Journal of Computer and System Sciences*, 71(2):213–247, 2005.

[10] J. Bethencourt, D. Song, and B. Waters. New constructions and practical applications for private stream searching. *IEEE Symposium on Security and Privacy*, 2006.

[11] R. Bitar, P. Parag, and S. E. Rouayheb. Minimizing latency for secure coded computing using secret sharing via staircase codes. *IEEE Transactions on Communications*, 68(8):4609–4619, 2020.

[12] R. Bitar and S. E. Rouayheb. Staircase-pir: Universally robust private information retrieval. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2018.

[13] R. Bitar, M. Xhemrishi, and A. Wachter-Zeh. Adaptive private distributed matrix multiplication. *arXiv preprint arXiv:2101.05681*, 2021.

[14] R. Bitar, Y. Xing, Y. Keshtkarjahromi, V. Dasari, S. E. Rouayheb, and H. Seferoglu. Private and rateless adaptive coded matrix-vector multiplication. *arXiv preprint arXiv:1909.12611*, 2019.

[15] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. *Proc. Adv. Cryptol.-Euro-crypt*, pages 506–522, 2004.

[16] T. H. Chan, S.-W. Ho, and H. Yamamoto. Private Information Retrieval for Coded Storage. *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pages 2842–2846, 2015.

[17] W. Chang and R. Tandon. On the capacity of secure distributed matrix multiplication. *IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2018.

[18] Z. Chen, Z. Jia, Z. Wang, and S. A. Jafar. GCSA Codes with Noise Alignment for Secure Coded Multi-Party Batch Matrix Multiplication. *ArXiv:2002.07750*, 2020.

[19] Z. Chen, Z. Wang, and S. Jafar. The capacity of private information retrieval with private side information. *arXiv preprint arXiv:1709.03022v1*, Sep. 2017.

[20] Z. Chen, Z. Wang, and S. A. Jafar. The capacity of t-private information retrieval with private side information. *IEEE Transactions on Information Theory*, 66(8):4761–4773, 2020.

[21] B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. *Technical Report TR CS0917*, 1997.

[22] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 41–50, 1995.

[23] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private Information Retrieval. *Journal of the ACM (JACM)*, 45(6):965–981, 1998.

[24] A. B. Das and A. Ramamoorthy. Coded sparse matrix computation schemes that leverage partial stragglers. *arXiv:2012.06065*, 2020.

[25] A. B. Das, L. Tang, and A. Ramamoorthy. C3les: Codes for coded computation that leverage stragglers. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2018.

[26] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56:74–80, 2013.

[27] S. Dutta, Z. Bai, H. Jeong, T. Low, and P. Grover. A Unified Coded Deep Neural Network Training Strategy Based on Generalized PolyDot Codes for Matrix Multiplication. *ArXiv:1811.10751*, Nov. 2018.

[28] S. Dutta, V. Cadambe, and P. Grover. Short-dot: Computing large linear transforms distributedly using coded short dot products. In *Advances In Neural Information Processing Systems*, pages 2100–2108, 2016.

[29] S. Dutta, V. Cadambe, and P. Grover. Coded convolution for parallel and distributed computing within a deadline. *arXiv preprint arXiv:1705.03875*, 2017.

[30] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover. On the Optimal Recovery Threshold of Coded Matrix Multiplication. *IEEE Transactions on Information Theory*, 66(1):278–301, 2020.

[31] R. G. D'Oliveira, S. E. Rouayheb, and D. Karpuk. Gasp codes for secure distributed matrix multiplication. *IEEE Transactions on Information Theory*, 2020. early access, DOI: 10.1109/TIT.2020.2975021.

[32] R. G. L. D'Oliveira, S. E. Rouayheb, D. Heinlein, and D. Karpuk. Notes on Communication and Computation in Secure Distributed Matrix Multiplication. *arXiv preprint arXiv:2001.05568*, 2020.

[33] X. Fan, P. Soto, X. Zhong, D. Xi, Y. Wang, and J. Li. Leveraging stragglers in coded computing with heterogeneous servers. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pages 1–10, 2020.

[34] G. Fanti. Private media search on public Databases. *Master thesis, University of California, Berkeley*, 2012.

[35] G. Fanti. Privacy-preserving Messaging and Search: A Collaborative Approach. *PhD thesis, University of California, Berkeley*, 2015.

[36] N. Ferdinand and S. C. Draper. Hierarchical coded computations. *IEEE International Symposium on Information Theory*, 2018.

[37] M. Finiasz and K. Ramchandran. Private stream search at the same communication cost as a regular search: Role of LDPC codes. *IEEE International Symposium on Information Theory*, pages 2556–2560, 2012.

[38] W. Gasarch. A Survey on Private Information Retrieval. In *Bulletin of the EATCS*, 2004.

[39] M. Gasca, J. Martinez, and G. Mühlbach. Computation of Rational Interpolants with Prescribed Poles. *Journal of Computational and Applied Mathematics*, 26(3):297–309, 1989.

[40] S. B. Gashkov and I. S. Sergeev. Complexity of computation in finite fields. *Journal of Mathematical Sciences*, 191(5):661–685, 2013.

[41] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 151–160. ACM, 1998.

[42] E.-J. Goh. Secure indexes. *IACR Cryptol. ePrint Archive*, page 216, 2003.

[43] P. Golle, J. Staddon, and B. Waters. Secure conjunctive keyword search over encrypted data. *Proc. Appl. Cryptography Netw. Secur.*, pages 31–45, 2004.

[44] F. Haddadpour and V. R. Cadambe. Codes for distributed finite alphabet matrix-vector multiplication. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1625–1629. IEEE, 2018.

[45] B. Hasırcıoglu, J. Gómez-Vilardebó, and D. Gündüz. Bivariate polynomial coding for exploiting stragglers in heterogeneous coded computing systems. *ArXiv:2001.07227*, 2020.

[46] B. Hasırcıoğlu, J. Gómez-Vilardebó, and D. Gündüz. Bivariate hermitian polynomial coding for efficient distributed matrix multiplicationn. *2020 IEEE Global Communications Conference*, pages 1–6, 2020.

[47] A. Heidarzadeh, B. Garcia, S. Kadhe, S. E. Rouayheb, and A. Sprintson. On the capacity of single-server multi-message private information retrieval with side information. *arXiv preprint arXiv:1807.09908*, 2018.

[48] A. Heidarzadeh, S. Kadhe, S. E. Rouayheb, and A. Sprintson. Single-server multi-message individually-private information retrieval with side information. *arXiv preprint arXiv:1901.07509*, 2019.

[49] A. Heidarzadeh, F. Kazemi, and A. Sprintson. Capacity of single-server single-message private information retrieval with coded side information. *arXiv preprint arXiv:1806.00661*, 2018.

[50] A. Heidarzadeh, F. Kazemi, and A. Sprintson. The role of coded side information in single-server private information retrieval. *arXiv preprint arXiv:1910.07612*, 2019.

[51] W. Huang, M. Langberg, J. Kliewer, and J. Bruck. Communication efficient secret sharing. *IEEE Transactions on Information Theory*, 62(12):7195–7206, 2016.

[52] H. Jafarkhani and M. Hajiaghayi. Cost-efficient repair for storage systems using progressive engagement. US Patent 10,187,088., Jan. 2019.

[53] T. Jahani-Nezhad and M. A. Maddah-Ali. Codedsketch: A coding scheme for distributed computation of approximated matrix multiplications. *arXiv preprint arXiv:1812.10460*, 2018.

[54] H. Jeong, F. Ye, and P. Grover. Locally recoverable coded matrix multiplication. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 715–722. IEEE, 2018.

[55] Z. Jia and S. Jafar. Cross-subspace alignment codes for coded distributed batch computation. *ArXiv:1909.13873*, 2019.

[56] Z. Jia and S. A. Jafar. On the Asymptotic Capacity of X-Secure T-Private Information Retrieval with Graph Based Replicated Storage. *ArXiv:1904.05906*, 2019.

[57] Z. Jia and S. A. Jafar. X-secure T-private Information Retrieval from MDS Coded Storage with Byzantine and Unresponsive Servers. *ArXiv:1908.10854*, 2019.

[58] Z. Jia and S. A. Jafar. X-secure t-private information retrieval from mds coded storage with byzantine and unresponsive servers. *IEEE Transactions on Information Theory*, 66(12):7427–7438, 2020.

[59] Z. Jia, H. Sun, and S. A. Jafar. Cross Subspace Alignment and the Asymptotic Capacity of X -Secure T -Private Information Retrieval. *IEEE Trans. on Info. Theory*, 65(9):5783–5798, Sep. 2019.

[60] S. Kadhe, B. Garcia, A. Heidarzadeh, S. E. Rouayheb, and A. Sprintson. Private information retrieval with side information. *arXiv preprint arXiv: 1709.00112*, 2017.

[61] S. Kadhe, B. Garcia, A. Heidarzadeh, S. E. Rouayheb, and A. Sprintson. Private information retrieval with side information. *IEEE Transactions on Information Theory*, 66(4):2032–2043, 2020.

[62] J. Kakar, S. Ebadifar, and A. Sezgin. On the capacity and straggler-robustness of distributed secure matrix multiplication. *IEEE Access*, 7:45783–45799, 2019.

[63] J. Katz, A. Sahai, and B. Waters. Secure conjunctive keyword search over encrypted data. *Proc. Adv. Cryptol.*, pages 146–162, 2008.

[64] F. Kazemi, E. Karimi, A. Heidarzadeh, and A. Sprintson. Single-server single-message online private information retrieval with side information. *arXiv preprint arXiv:1901.07748*, 2019.

[65] S. Kiani, N. Ferdinand, and S. C. Draper. Exploitation of stragglers in coded computation. *IEEE International Symposium on Information Theory*, 2018.

[66] S. Kianidehkordi, N. Ferdinand, and S. C. Draper. Hierarchical coded matrix multiplication. *IEEE Transactions on Information Theory*, 67(2):726–754, 2021.

[67] M. Kim and J. Lee. Private secure coded computation. *IEEE Communications Letters*, 23(11):1918–1921, 2019.

[68] M. Kim, J.-y. Sohn, and J. Moon. Coded matrix multiplication on a group-based model. *arXiv preprint arXiv:1901.05162*, 2019.

[69] M. Kuzu, M. S. Islam, and M. Kantarcioglu. Efficient similarity search over encrypted data. *Proc. IEEE 28th Int. Conf. Data Eng.*, pages 1156–1167, 2012.

[70] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory*, 64(3):1514–1529, 2017.

[71] K. Lee, C. Suh, and K. Ramchandran. High-dimensional coded matrix multiplication. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2418–2422. IEEE, 2017.

[72] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy keyword search over encrypted data in cloud computing. *Proc. IEEE INFOCOM*, pages 1–5, 2010.

[73] S. Li and M. Gastpar. Converse for multi-server single-message pir with side information. *arXiv preprint arXiv:1809.09861*, 2018.

[74] S. Li and M. Gastpar. Single-server multi-user private information retrieval with side information. *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2018.

[75] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr. Coding for distributed fog computing. *IEEE Communications Magazine*, 55(4):34–40, 2017.

[76] W. Li, Z. Wang, T. Lu, and H. Jafarkhani. Storage codes with flexible number of nodes. *arXiv preprint arXiv:2106.11336*, 2021.

[77] S. Lin and D. J. Costello. *Error control coding*, volume 2. Prentice hall, 2001.

[78] Y. Malitsky and K. Mishchenko. Adaptive gradient descent without descent. *arXiv preprint arXiv:1910.09529*, 2019.

[79] A. Mallick, M. Chaudhari, U. Sheth, G. Palanikumar, and G. Joshi. Rateless codes for near-perfect load balancing in distributed matrix-vector multiplication. *Proc. ACM Meas. Anal. Comput. Syst.*, 3(3), 2019.

[80] M. Mirmohseni and M. A. Maddah-Ali. Private function retrieval. *arXiv preprint arXiv:1711.04677*, 2017.

[81] M. H. Mousavi, M. A. Maddah-Ali, and M. Mirmohseni. Private inner product retrieval for distributed machine learning. *arXiv preprint arXiv: 1902.06319*, 2019.

[82] H. A. Nodehi and M. A. Maddah-Ali. Limited-sharing multi-party computation for massive matrix operations. *IEEE International Symposium on Information Theory*, 2018.

[83] H. A. Nodehi and M. A. Maddah-Ali. Secure coded multi-party computation for massive matrix operations. *ArXiv:1908.04255*, 2019.

[84] H. A. Nodehi, S. R. H. Najarkolaei, and M. A. Maddah-Ali. Entangled polynomial coding in limited-sharing multi-party computation. *IEEE Information Theory Workshop*, 2018.

[85] S. A. Obead and J. Kliewer. Achievable rate of private function retrieval from mds coded databases. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 2117–2121, 2018.

[86] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer. Capacity of private linear computation for coded databases. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 813–820, 2018.

[87] V. Olshevsky and A. Shokrollahi. A Superfast Algorithm for Confluent Rational Tangential Interpolation Problem via Matrix-Vector Multiplication for Confluent Cauchy-like Matrices. *Structured Matrices in Mathematics, Computer Science, and Engineering I, Contemporary Mathematics series*, 280:32–46, 2001.

[88] R. Ostrovsky and W. E. S. III. Private searching on streaming data. *Advances in Cryptology - CRYPTO*, pages 223–240, 2005.

[89] R. Ostrovsky and W. E. Skeith III. A Survey of Single-database Private Information Retrieval: Techniques and Applications. In *Public Key Cryptography–PKC 2007*, pages 393–411. Springer, 2007.

[90] H. Park, K. Lee, J.-y. Sohn, C. Suh, and J. Moon. Hierarchical coding for distributed computing. *arXiv preprint arXiv:1801.04686*, 2018.

[91] A. Ramamoorthy, L. Tang, and P. O. Vontobel. Universally decodable matrices for distributed matrix-vector multiplication. *arXiv:1901.10674*, 2019.

[92] N. Raviv and D. A. Karpuk. Private polynomial computation from lagrange encoding. *IEEE Transactions on Information Forensics and Security*, 15:553–563, 2020.

[93] A. Reisizadeh, S. Prakash, R. Pedarsani, and A. S. Avestimehr. Coded Computation over Heterogeneous Clusters. *IEEE Transactions on Information Theory*, 65(7):4227–4242, 2019.

[94] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur. Xoring elephants: Novel erasure codes for big data. In *39th Int. Conf. Very Large Data Bases*, volume 6, pages 325–336, 2013.

[95] A. Severinson, A. G. i Amat, and E. Rosnes. Block-diagonal and lt codes for distributed computing with straggling servers. *IEEE Transactions on Communications*, 67(3):1739–1753, 2018.

[96] N. Shah, K. Rashmi, and K. Ramchandran. One Extra Bit of Download Ensures Perfectly Private Information Retrieval. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pages 856–860, 2014.

[97] S. P. Shariatpanahi, M. J. Siavoshani, and M. A. Maddah-Ali. Multi-message private information retrieval with private side information. *arXiv preprint arXiv:1805.11892*, 2018.

[98] U. Sheth, S. Dutta, M. Chaudhari, H. Jeong, Y. Yang, J. Kohonen, T. Roos, and P. Grover. An application of storage-optimal matdot codes for coded matrix multiplication: Fast k-nearest neighbors estimation. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1113–1120. IEEE, 2018.

[99] V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

[100] G. Suh, K. Lee, and C. Suh. Matrix sparsification for coded matrix multiplication. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1271–1278. IEEE, 2017.

[101] H. Sun and S. A. Jafar. Blind interference alignment for private information retrieval. *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 560–564, 2016.

[102] H. Sun and S. A. Jafar. The Capacity of Private Information Retrieval. *IEEE Transactions on Information Theory*, 63(7):4075–4088, July 2017.

[103] H. Sun and S. A. Jafar. The capacity of symmetric private information retrieval. *IEEE Transactions on Information Theory*, 2018.

[104] H. Sun and S. A. Jafar. Multiround Private Information Retrieval: Capacity and Storage Overhead. *IEEE Transactions on Information Theory*, 64(8):5743–5754, August 2018.

[105] H. Sun and S. A. Jafar. The Capacity of Robust Private Information Retrieval with Colluding Databases. *IEEE Transactions on Information Theory*, 64(4):2361–2370, April 2018.

[106] H. Sun and S. A. Jafar. The capacity of private computation. *IEEE Transactions on Information Theory*, 65(6):3880–3897, June 2019.

[107] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb. Private Information Retrieval from MDS Coded Data in Distributed Storage Systems. *IEEE Transactions on Information Theory*, 2018.

[108] R. Tandon. The capacity of cache aided private information retrieval. *arXiv preprint arXiv:1706.07035*, 2017.

[109] R. Tandon. The capacity of cache aided private information retrieval. *arXiv preprint arXiv:1706.07035*, 2017.

[110] R. Tandon, M. Abdul-Wahid, F. Almoualem, and D. Kumar. Pir from storage constrained databases - coded caching meets pir. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–7, 2018.

[111] C. Wang, N. Cao, K. Ren, and W. Lou. Enabling secure and efficient ranked keyword search over outsourced cloud data. *IEEE Trans. Parallel Distrib. Syst.*, 23(8):1467–1479, 2012.

[112] Q. Wang and M. Skoglund. Linear symmetric private information retrieval for MDS coded distributed storage with colluding servers. *arXiv preprint arXiv:1708.05673*, 2017.

[113] Q. Wang and M. Skoglund. Secure symmetric private information retrieval from colluding databases with adversaries. *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1083–1090, 2017.

[114] Q. Wang and M. Skoglund. Symmetric private information retrieval for mds coded distributed storage. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, 2017.

[115] S. Wang, J. Liu, and N. Shroff. Coded sparse matrix multiplication. *arXiv preprint arXiv:1802.03430*, 2018.

[116] S. Wang, J. Liu, N. Shroff, and P. Yang. Fundamental limits of coded linear transform. *arXiv preprint arXiv:1804.09791*, 2018.

[117] Y. Wei, K. Banawan, and S. Ulukus. Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching. *IEEE Transactions on Information Theory*, 65(5):3215–3232, May 2019.

[118] Y. Wei, K. Banawan, and S. Ulukus. Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching. *IEEE Transactions on Information Theory*, 65(5):3215–3232, 2019.

[119] Y.-P. Wei, K. Banawan, and S. Ulukus. Cache-aided private information retrieval with partially known uncoded prefetching: fundamental limits. *IEEE Jour. on Selected Areas in Communications*, 36(6):1126–1139, 2018.

[120] Y. Yang, M. Interlandi, P. Grover, S. Kar, S. Amizadeh, and M. Weimer. Coded elastic computing. *arXiv preprint arXiv:1812.06411*, 2018.

[121] A. C. Yao. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 160–164. IEEE, 1982.

[122] A. C. Yao. Protocols for secure computations (extended abstract). *23rd Annual Symposium on Foundations of Computer Science*, pages 160–164, 1982.

[123] S. Yekhanin. Private Information Retrieval. *Communications of the ACM*, 53(4):68–73, 2010.

[124] Q. Yu and A. S. Avestimehr. Entangled Polynomial Codes for Secure, Private, and Batch Distributed Matrix Multiplication: Breaking the "Cubic Barrier". *ArXiv:2001.05101*, 2020.

[125] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr. Lagrange coded computing: Optimal design for resiliency, security, and privacy. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1215–1225. PMLR, 2019.

[126] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr. Coded fourier transform. *arXiv preprint arXiv:1710.06471*, 2017.

[127] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr. Polynomial Codes: an Optimal Design for High-Dimensional Coded Matrix Multiplication. *arXiv preprint arXiv:1705.10464*, 2017.

[128] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr. Straggler Mitigation in Distributed Matrix Multiplication: Fundamental Limits and Optimal Coding. *IEEE Transactions on Information Theory*, 66(3):1920–1933, 2020.

[129] W. Zhao, X. Ming, S. Mikael, and P. H. Vincent. Secure degrees of freedom of wireless x networks using artificial noise alignment. *IEEE Transactions on communications*, 63(7):2632–2646, 2015.