ABSTRACT

Title of dissertation: TOWARDS SECURE COMPUTATION AND LEARNING FROM SYMMETRIC PRIVATE INFORMATION RETRIEVAL

Zhusheng Wang, Doctor of Philosophy, 2023

Dissertation directed by: Professor Sennur Ulukus
Department of Electrical and Computer Engineering

This dissertation focuses on secure computation and learning from the perspective of information-theoretic symmetric private information retrieval (SPIR). Private information retrieval (PIR) is an elementary and classical problem in computer science, aiming to protect the privacy of users. In a typical PIR setting, a user wishes to correctly download the desired message out of a set of messages from multiple non-colluding and replicated databases. This constraint is known as the reliability constraint. Meanwhile, no individual database should learn anything about which message the user has downloaded. This constraint is known as the user privacy constraint. In SPIR, further, the user should learn nothing beyond the particular message it has downloaded. This constraint is known as the database privacy constraint.

As a fundamental problem in secure multi-party computation, two-party private set intersection (PSI) refers to the problem where two parties wish to collaboratively determine the common elements without leaking any further information

about the remaining elements to each other. There are three requirements for two-party PSI: At least one of the two parties should be able to decode the intersection correctly (reliability), queried party should not learn the information being queried (user privacy) and the querying party should not learn anything further than what it has queried (database privacy). The last two constraints protect the privacy of the remaining elements in two parties. Thus, the constraints in SPIR and two-party PSI are equivalent and have a one-to-one correspondence. Further, SPIR is a distributed (multi-database) version of 1-out-of-K oblivious transfer (OT). It is well-known that reliability, user privacy and database privacy contradict each other, and thus, basic single-database SPIR, and basic two-party PSI are infeasible. In order to achieve a valid two-party PSI protocol, our primary solutions are to relax the privacy constraints via use of multiple databases and utilize auxiliary randomness data. Further, we study the secure computation problem via SPIR and apply it to the secure learning problem.

First, we investigate the problem of two-party PSI by using SPIR. It is well-known that single-database SPIR is not feasible. Hence, in our two-party PSI setting, we assume that each party stores its data set across multiple non-colluding and replicated databases instead of a single database. As a relaxation of the previous privacy requirements, the remaining elements of each party only needs to be kept private from each individual database in the other party. As a consequence, we propose a new valid two-party PSI achievable scheme under this new privacy requirement. More concretely, we consider the multi-message SPIR (MM-SPIR) problem, which is an extension of the conventional single-message SPIR (SM-SPIR). In MM-

SPIR, the user retrieves multiple messages at a time from the databases. We obtain the capacity of MM-SPIR as a stand-alone result and show that the MM-SPIR capacity equals the SM-SPIR capacity. We also unify the schemes of multi-message PIR (MM-PIR) and MM-SPIR by proposing a new capacity-achieving MM-SPIR scheme. Finally, we establish the equivalence between two-party PSI and MM-SPIR with i.i.d. messages of length 1 through an incidence vector mapping, which implies the minimum download cost for two-party PSI.

Second, we investigate the problem of SPIR with user-side common randomness where the user is provided with a random subset of the shared database common randomness, which is unknown to the databases. We determine the exact capacity region of the triple (d, ρ_S, ρ_U), where d is the download cost, ρ_S is the amount of shared database common randomness, and ρ_U is the amount of available user-side common randomness. With an appropriate amount of ρ_U, this new SPIR can achieve the conventional PIR capacity. As a corollary, single-database SPIR becomes feasible. Consequently, the user-side common randomness in SPIR can be deemed as auxiliary randomness data and then applied to two-party PSI problem where each party now possesses only a single database. Likewise, the minimum download cost for two-party PSI in this case is derived.

Third, we consider the problem of multi-party PSI (MP-PSI). In MP-SPI, several parties wish to jointly determine the intersection of their respective elements while protecting the information about the remaining elements. MP-PSI is a non-trivial extension of the two-party PSI as it cannot be implemented via multiple use of two-party PSI. We propose a new information-theoretic MP-PSI scheme that

builds on the connection between PSI and MM-SPIR. Our scheme is a non-trivial generalization of the two-party PSI scheme since it needs an intricate design of the shared common randomness among the parties. With the aid of this auxiliary randomness data, we show that our scheme does not incur any penalty, in terms of the download cost, due to the more stringent privacy constraints in MP-PSI compared to two-party PSI.

Fourth, as the communication cost in the PSI problem consists of upload cost and download cost, we consider the total (upload plus download) communication cost of SPIR with a focus on two-database setting through its relationships to conditional disclosure of secrets (CDS) and conditional disclosure of multiple secrets (CDMS). In CDS, two parties each holding an individual input and sharing a common secret wish to disclose this secret to an external party efficiently when their inputs satisfy some condition. As a natural extension of CDS, in CDMS, two parties share multiple i.i.d. common secrets. Inspired by the equivalence between a special configuration of CDMS and two-database SPIR, we design download cost efficient SPIR schemes for given upload cost using bipartite graph representations of CDS and CDMS.

Fifth, as a novel and interesting extension of SPIR, we consider the problem of random SPIR (RSPIR) where the user does not pick a specific message to download, instead is content with any one of the messages stored in the databases. This is digital blind box, also known as gachapon, which implements this specified setting with physical objects for entertainment. This is also the blind version of 1-out-of-K OT, an important cryptographic primitive. We explore the capacity of two-database

RSPIR and provide its corresponding achievable scheme.

Sixth, we consider the private set union (PSU) problem and then apply it to the federated submodel learning (FSL) problem. As a dual problem of two-party PSI, two-party PSU refers to the problem where two parties wish to collaboratively compute the union of their elements without revealing anything beyond this union to each other. Hence, we also establish the equivalence between two-party PSU and MM-SPIR. In FSL, the full learning model in the server is divided into multiple submodels such that a large number of clients collectively update the model by downloading only the needed submodels and uploading the corresponding increments while keeping clients' local training data private from the server. As a conventional approach in FSL, secure aggregation ensures that the server can only learn the aggregate model from the clients and nothing beyond that. By contrast, if one of the parties is selected as a leader party to derive the union, multi-party PSU (MP-PSU) requires that this leader party obtains only the union from the remaining parties and nothing beyond that. By unifying secure aggregation and MP-PSU in the same framework, we propose a new FSL scheme that achieves low communication cost, and is also robust against client drop-outs, client late-arrivals and database drop-outs.

Finally, seventh, we consider the FSL problem in a distributed storage system. The server now comprises multiple independent databases and the full model is stored through ramp secure regenerating coding (RSRC) across these databases. This novel RSRC mechanism is proposed to resolve passive eavesdropper and database failure issues together in an efficient manner by allowing the eavesdropper to learn

a controllable amount of submodel information. Our new RSRC-based distributed FSL approach is constructed on the basis of our previous two-database FSL scheme which uses PSU. In addition to the previous robustness against client drop-outs, client late-arrivals and database drop-outs, and newly-added robustness against database failures and a passive eavesdropper, this advanced FSL approach also guarantees robustness against an active adversary.

Towards Secure Computation and Learning from Symmetric Private Information Retrieval

by

Zhusheng Wang

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2023

Advisory Committee:
Professor Sennur Ulukus, Chair/Advisor
Professor Sanghamitra Dutta
Professor Prakash Narayan
Professor Adrian Papamarcou
Professor Nirupam Roy
Professor Lawrence C. Washington

Dedication

To my family and friends.

Acknowledgments

First and foremost, I would like to thank my advisor Professor Sennur Ulukus for her support and guidance throughout my Ph.D. career. I am very lucky to learn the truth no pains no gains from her at the beginning of my Ph.D. career. This truth motivates me to produce academic achievements all the way. She also taught me how to write a good paper and give an excellent presentation from scratch. After our first project, she gave me enough freedom to pursue new projects that intrigued me. Her door was always open whenever I encountered any difficulty and needed to talk with her. Her broad knowledge and rich experience enlightened me a lot. It is really my honor to collaborate with such a nice mentor in the past five years.

Second, I would like to thank Professors Sanghamitra Dutta, Prakash Narayan, Adrian Papamarcou, Nirupam Roy and Lawrence C. Washington for being in my dissertation committee. I appreciate their time, energy and valuable feedback. I am also thankful to all the professors that I once interacted with throughout my Ph.D. career at the University of Maryland. In particular, I would like to thank Professor Alexander Barg for being my academic advisor, research proposal committee member and teaching three meaningful classes I took.

Third, I would like to thank Professor Huifang Chen at Zhejiang University who introduced me to the joy of research, and Professor Sandeep Pradhan at the University of Michigan who led me into the fascinating realm of information theory.

Next, I would like to thank my research group mates Sajani Vithana, Priyanka Kaswan, Matin Mortaheb, Cemil Vahapoglu, Purbesh Mitra, Mustafa Doger, Sub-

hankar Banerjee, Shreya Meel, Alptug Aytekin, Sahan Liyanaarachchi, Mohamed Nomeir, Arunabh Srivastava, Batuhan Arasli, Brian Kim, Baturalp Buyukates, Melih Bastopcu. I am very fortunate to work with these brilliant and earnest people. In particular, I would like to thank Sajani Vithana. We had many enlightening discussions about privacy and security issues. I am also thankful to Karim Banawan. Even though we did not meet with each other, he provided much help of great value to me during my first two projects.

Besides, I would like to thank my friends at or outside the Univeristy of Maryland. I really cherish our friendship. Without their company, my life cannot be such colorful and fulfilling. In particular, I would like to thank Yuan Liu. Ever since we met, she has been inspiring me throughout my growth process. Until now, she is still believing in me and encouraging me to pursue what I truly love. I am also thankful to Haoran Li and Xingyu Ren. I knew Haoran through house renting, and then we participated in many interesting activities together. With Xingyu, we took many common classes and talk about many gossipy topics beyond classes as well.

Finally, I would like to thank my family for their unconditional love and support. Sometimes, words cannot express my sincere gratitude to my father, Xiguang Wang, and my mother Guiju Xu. No matter what happens, they are always my last resort.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

In the era of big data, it is common to complete a task by utilizing the data with a large size distributed in the form of isolated islands, e.g., perform a target computation over the personal data possessed by a few parties or train a global learning model by collecting local data from many clients. Therefore, during the data exchange process, preserving the privacy of data is an essential but challenging problem. In this dissertation, we investigate privacy and security issues arising in such settings from an information-theoretic perspective.

As an elementary two-party secure computation problem, two-party private set intersection (PSI) refers to the problem of determining the common elements in two data sets without leaking any further information about the remaining elements in the sets. This problem has been a major research topic in the field of cryptography starting with the work [1]. The PSI problem can be motivated by many practical examples, for instance: The national security agency (NSA) and the customs and border protection (CBP) need to check whether a specific group of suspected criminals has entered the country. The NSA has a list of suspected

criminals, while the CBP has a complete list of individuals who entered the country. Both agencies want to find the intersection between these lists. However, the NSA does not want to share its complete list of suspects, and the CBP cannot reveal the entire catalog of records either. As another example, consider a major service provider (e.g., Whatsapp) and a new customer who wishes to join this service. The user wishes to find out which members of his/her contact list are already using this service without revealing his/her entire contact list to the service provider. Similarly, the service provider wishes to determine the intersection without revealing its entire list of customers.

To formulate the two-party PSI problem information-theoretically, consider a setting where party P_1 stores a data set $\mathcal{P}_1$ in a single database and party P_2 stores another data set $\mathcal{P}_2$ in another single database. They want to jointly calculate the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ in a private way. Without loss of generality, we assume here that the party P_1 initiates the PSI process by sending some query information Q to the other party P_2. Subsequently, P_2 sends some answer information A as a response back to P_1. Hence, we have the following three constraints. First, P_1 should be able to derive the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ reliably,

$$\text{[PSI reliability]} \qquad H(\mathcal{P}_1 \cap \mathcal{P}_2 | Q, A, \mathcal{P}_1) = 0 \tag{1.1}$$

Second, P_1 wants to protect its personal elements in $\mathcal{P}_1 \setminus (\mathcal{P}_1 \cap \mathcal{P}_2) = \mathcal{P}_1 \setminus \mathcal{P}_2$ from P_2. However, since P_1 does not know $\mathcal{P}_2$, the query Q cannot depend on $\mathcal{P}_2$, and P_1 should protect all of $\mathcal{P}_1$ in query. Therefore, the query sent by P_1 should not leak

any information about the data set $\mathcal{P}_1$,

$$[P_1 \text{ privacy}] \qquad I(\mathcal{P}_1; Q, A, \mathcal{P}_2) = 0 \tag{1.2}$$

Third, by symmetry, P_2 wants to protect its personal elements in $\mathcal{P}_2 \setminus (\mathcal{P}_1 \cap \mathcal{P}_2) = \mathcal{P}_2 \setminus \mathcal{P}_1$ from P_1. Moreover, P_1 should not learn the absence of the remaining elements in the set $\overline{(\mathcal{P}_1 \cup \mathcal{P}_2)}$. Therefore, P_1 should not learn any information about P_2's inclusion status of the elements in $(\mathcal{P}_2 \setminus \mathcal{P}_1) \cup \overline{(\mathcal{P}_1 \cup \mathcal{P}_2)} = \bar{\mathcal{P}}_1$ (we denote this information by $E_{\bar{\mathcal{P}}_1}$),

$$[P_2 \text{ privacy}] \qquad I(E_{\bar{\mathcal{P}}_1}; Q, A, \mathcal{P}_1) = 0 \tag{1.3}$$

Here, we prove that these three constraints conflict with each other such that the basic two-party PSI is indeed not feasible, which matches our intuition. Note that given the knowledge about P_2's inclusion status of the elements in $\mathcal{P}_1$ (we denote this information by $E_{\mathcal{P}_1}$), P_1 is able to obtain the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$, and vice versa. Hence, the reliability constraint becomes

$$[\text{PSI reliability}] \qquad H(E_{\mathcal{P}_1} | Q, A, \mathcal{P}_1) = 0 \tag{1.4}$$

Due to the P_1 privacy constraint (1.2), the query Q and answer A are identically distributed for any $\mathcal{P}_1$. Then, due to the reliability constraint (1.4), from $Q, A, \mathcal{P}_1$, the information $E_{\mathcal{P}_1}$ is always decodable to P_1 for any $\mathcal{P}_1$. Putting these two facts together, from Q, A and variable $\mathcal{P}_1$, we must have that P_1 can obtain the informa-

tion about P_2's inclusion status of all the elements, which obviously contradicts the P_2 privacy constraint (1.3).

All the existing PSI works in the computer science community uses computational guarantees to ensure the privacy of the elements beyond the intersection; see [1–4]. Computational security relies on the fact that there is no computer system powerful enough to crack the cipher in a reasonable amount of time, and thus can be broken by an attack with unlimited computation power. By contrast, information-theoretic security is secure against an adversary with unlimited computing resources and time. According to the indistinguishability of limited computation power, it is apparent that previous valid two-party PSI protocols relax the stringent requirement of information-theoretic security to computational security. In this dissertation, we aim to achieve valid two-party PSI protocols in different ways, but still under the umbrella of information-theoretic security. In brief, our first approach is to relax the privacy constraint through including multiple databases in each party, while the second approach is to distribute and utilize additional auxiliary randomness data at both parties. By convention, the performance of our new valid two-party PSI protocols is still evaluated in terms of the total communication cost that is generated within the whole process.

In the cryptography field, private information retrieval (PIR) refers to a fundamental problem where a user wishes to retrieve a specific message out of a set of messages that is stored across multiple non-colluding and replicated databases without leaking any information about this desired message index to any individual database [5]. The requirement that the user can obtain the desired message without

any error is referred to as the reliability constraint, and the requirement that each individual database learns no knowledge about the index of this desired message is referred to as the user privacy constraint. The capacity of PIR, which is defined as the maximum number of desired message bits that can be privately retrieved per downloaded bit from the databases, is proved to be $C_{\text{PIR}} = (1 - \frac{1}{N})/(1 - \left(\frac{1}{N}\right)^K)$ in [6] for an N-database K-message system. As an extended version of PIR, symmetric PIR (SPIR) additionally requires that the user learns no knowledge about the remaining messages in the databases after downloading its desired message [7]. This requirement is referred to as the database privacy constraint. Following the same definition as in the capacity of PIR, the capacity of SPIR is proved to be $C_{\text{SPIR}} = 1 - \frac{1}{N}$ in [8]. Although PIR does not need any shared common randomness among the databases, it is well-known that information-theoretic SPIR is possible only when the databases share a certain minimum amount of common randomness that is unknown to the user. Following the seminal paper on information-theoretic PIR capacity [6], PIR and SPIR problems have attracted tremendous attention in the information theory community recently, e.g., [9–63].

In this dissertation, we concentrate on the SPIR problem taking into consideration the fact that two-party PSI and SPIR share three essential constraints with similar characteristics. In addition, we note that SPIR is basically a multi-database version of 1-out-of-K oblivious transfer (OT). OT, first introduced in [64] and then developed in [65], is an essential building block in modern cryptography because of a variety of applications that can be built based on it [64]. A 1-out-of-K OT protocol consists of two parties, a sender with K input messages and a receiver with a choice

$k \in [K]$. The objective of the protocol is that the receiver will receive the kth message without the sender learning the index k, while the sender can guarantee that the receiver only received one of the K messages. We explore secure computation and learning by utilizing SPIR as an essential building block from the perspective of information theory. In particular, we use a probabilistic idea as in [66, 67] to design new SPIR achievable schemes with better performance and then apply them to secure computation and learning problems with the main objective of reducing the overall communication cost.

In Chapter 2, we study the two-party PSI problem on the basis of SPIR. Following the conventional configuration in SPIR [8], each party now stores its own data set across multiple non-colluding and replicated databases. As a consequence, the previous privacy requirements can be relaxed such that only the privacy of the remaining elements in each party needs to be guaranteed against each individual database in the other party. In this way, the information-theoretic privacy can be still satisfied in our new two-party PSI scheme. In our PSI problem formulation, there are two parties P_i, for $i = 1, 2$, each storing a data set $\mathcal{P}_i$, whose elements are picked from a finite set $\mathbb{S}_K$ (a.k.a. global alphabet), on N_i replicated and non-colluding databases. Now, to use SPIR to implement PSI, one party needs to privately check the presence of each element in the other party. This implies that the ith party needs to retrieve *multiple messages* from the other party, where the messages here correspond to the *incidences* of each element of the set $\mathcal{P}_i$. This establishes the connection between two-party PSI and multi-message SPIR (MM-SPIR). The papers that are most closely related to our work are the ones that focus on *sym-*

metry and *multi-message* aspects of PIR. Reference [8] derives the SPIR capacity. Reference [23] considers multi-message PIR (MM-PIR) and determines the exact capacity when the number of desired messages P is at least half of the total number of messages K or when K/P is an integer; for all other cases [23] provides a novel PIR scheme which is near-optimal. Reference [35] studies multi-database MM-PIR with private side information. References [36, 37] study single-database MM-PIR with side information. Reference [13] studies SPIR from MDS-coded databases. None of these works consider the interplay between the data privacy constraint and the joint retrieval of multiple messages, as needed in MM-SPIR.

In this chapter, we first focus on MM-SPIR as a stand-alone problem, we derive its capacity. Our results show that the sum capacity of MM-SPIR is exactly equal to the capacity of single-message SPIR (SM-SPIR), i.e., $C_{SM-SPIR} = C_{MM-SPIR} = 1 - \frac{1}{N}$. We show that the databases need to share a random variable S such that $H(S) \geq \frac{P}{N-1}$ per desired symbol, which is P multiple of the common randomness required for SM-SPIR. This implies that, unlike MM-PIR, there is no gain from jointly retrieving the P messages, and it suffices to download the P messages successively using the SM-SPIR scheme in [8], provided that statistically independent common randomness symbols are used at each time. Further, for MM-SPIR, we propose a novel capacity-achieving scheme for $1 \leq P \leq K-1$. Compared with the one in [8], the form of this achievable scheme is much closer to the achievable scheme in [6]. The query structure of the scheme resembles its counterpart in [23], in particular, we construct the greedy algorithm in [6] backwards as in [23]. The major difference between our proposed scheme and the MM-PIR scheme in [23] is the fact that databases

add the common randomness to the returned answer strings to satisfy the database privacy constraint. Our scheme is surprisingly optimal for all P and K in contrast to the scheme in [23] which is proved to be optimal only if P is at least half of K or K/P is an integer. By plugging $P = 1$, our scheme serves as an alternative capacity-achieving scheme for the SM-SPIR scheme in [8]. Finally, we consider two-party PSI. In transforming two-party PSI into MM-SPIR, each party constructs the incidence vector of its own elements with respect to the global alphabet. The incidence vector is a binary vector of length K that stores a 1 in jth position if jth element is in the data set. Then, P_i performs MM-SPIR of the messages corresponding to its incidence vector of $\mathcal{P}_i$ within the databases of the other party. Thus, the equivalence between two-party PSI and MM-SPIR with i.i.d. messages of length 1 can be established. We show that the optimum download cost of two-party PSI is $\min\left\{\left\lceil\frac{|\mathcal{P}_1|N_2}{N_2-1}\right\rceil, \left\lceil\frac{|\mathcal{P}_2|N_1}{N_1-1}\right\rceil\right\}$, which is linear in the size of the smaller set, i.e., $\min\{|\mathcal{P}_1|, |\mathcal{P}_2|\}$. The linear scaling appears in the problem of determining the set intersection even without any privacy constraints.

In Chapter 3, we study the problem of SPIR with user-side common randomness. As we discussed above, the privacy constraint of two-party PSI is loosened by considering the setting of multiple non-colluding and replicated databases in each party. However, in practical applications, enforcing non-collusion among databases could be difficult, as in some cases, all databases may naturally belong to the same entity. If all databases collude or belong to the same entity, the system essentially becomes a single-database system. In addition, we are interested in exploring new ways to further increase the SPIR capacity. The information-theoretic capacity of

PIR and SPIR have been found in [6] and [8] as

$$C_{\text{PIR}} = \frac{1 - \frac{1}{N}}{1 - \left(\frac{1}{N}\right)^K} \qquad \text{and} \qquad C_{\text{SPIR}} = 1 - \frac{1}{N} \tag{1.5}$$

First, C_{SPIR} is smaller than C_{PIR}, since SPIR is a more constrained problem than PIR. Second, single-database SPIR is infeasible as $C_{\text{SPIR}} = 0$ for $N = 1$, while single-database PIR is feasible as $C_{\text{PIR}} = \frac{1}{K}$ for $N = 1$. Our goal in this chapter is two-fold: To explore ways to increase SPIR capacity to the level of PIR capacity, and as importantly, to make single-database SPIR feasible. In SPIR, the databases share two sets of information systems: a message information system and a common randomness information system. As stated in [33, Theorem 2], the capacity of SPIR with private side information is exactly equal to the capacity of SPIR without any private side information. This means that the message information system cannot be utilized as private side information to improve the capacity of SPIR. As an alternative, we turn to the common randomness information system.

In this chapter, we introduce SPIR with user-side common randomness to solve two issues mentioned above together. It is an extension of the classical SPIR problem where the user is provided with a random subset of the shared database common randomness, which is unknown to the databases. One way to implement this is for the user to fetch a part of the common randomness from the databases uniformly randomly, i.e., without the user knowing what it will get and without the databases knowing what it got, except for its cardinality. That is, all subsets of a certain size are equally likely to be obtained by the user. Another practical

implementation could be for an external helper to distribute common randomness to the user and the databases randomly. For database-side (server-side) common randomness of amount ρ_S and user-side common randomness of amount ρ_U, we determine the exact capacity region of the triple (d, ρ_S, ρ_U), where d is the download cost which is the inverse of the capacity. We show that with a suitable ρ_U, SPIR capacity becomes equal to the conventional PIR capacity. For the single-database case, since the conventional PIR capacity is $\frac{1}{K}$, this implies that single-database SPIR with user-side common randomness is feasible. In addition, the presence of user-side ρ_U reduces the amount of required server-side ρ_S. As a consequence, by utilizing user-side common randomness as auxiliary randomness data, two-party PSI with the setting of single database in each party becomes feasible, and the minimum download cost in this case is obtained.

In Chapter 4, following our work on two-party PSI problem, we investigate the multi-party PSI (MP-PSI) problem, which is a multi-party secure computation problem; see [68–71]. Unlike PIR/SPIR, the PSI problem may involve more than two parties in the practical setting. Returning to the example involving the NSA and CBP above, suppose now that the NSA needs to narrow down the search to check whether the suspects have entered the country via a specific airline. The airline company has a list of all passengers that took its flights all over the world. The company needs to protect the privacy of its passengers as well. The problem of finding the set of suspects who entered the country via this specific airline becomes a three-party PSI. Unfortunately, the NSA cannot just apply a two-party PSI scheme with the airline company and the CBP, as the NSA will learn extra information than

the intersection of the three lists, for example, the NSA will learn about some of its suspects who boarded a flight with this airline company but never landed in this country. This illustrates that the MP-PSI is a non-trivial extension of the two-party PSI because it cannot be realized through multiple implementations of two-party PSI.

In this chapter, we investigate MP-PSI from an information-theoretic perspective. There are M independent parties. For $i = 1, \cdots, M$, the ith party denoted by P_i possesses a data set $\mathcal{P}_i$. The elements of all data sets are picked from a finite set $\mathbb{S}_K$. The data set $\mathcal{P}_i$ is stored in N_i replicated and non-colluding databases. We aim at privately determining the intersection of all the M data sets, i.e., $\mathcal{P} = \cap_{i=1}^{M} \mathcal{P}_i$ in such a way that no party can learn any information beyond the intersection $\mathcal{P}$. Inspired by the classical achievable scheme in [1, 68], we focus on a specific communication strategy between the parties in this work; see Fig. 1.1. In particular, we assume that the parties agree on choosing one of them as a *leader* party, while the remaining parties act as *client* parties. Without loss of generality, we pick P_M as a leader party, and then the remaining parties $P_1, \cdots, P_{M-1}$ are all client parties. The leader party P_M initiates the MP-PSI determination protocol by generating and submitting queries to the client parties. Before MP-PSI, the client parties are allowed to generate and share well-designed intricate common randomness (common randomness residing in the jth database of party P_i is shown by $\mathcal{R}_{i,j}$ in Fig. 1.1). This is motivated by the results of [7, 8], which assert that using common randomness is strictly necessary to enable symmetrically private communication. The client parties then respond truthfully to the leader's queries. The download cost

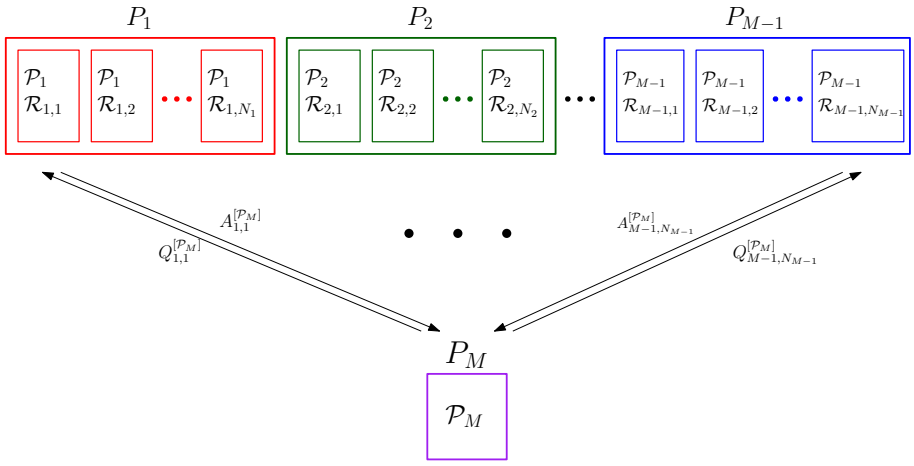


Figure 1.1: Multi-party private set intersection (MP-PSI) system model.

of our scheme is $\min_{t\in\{1,\cdots,M\}} \sum_{i\in\{1,\cdots M\}\setminus t} \left\lceil \frac{|\mathcal{P}_t|N_i}{N_i-1} \right\rceil$. Note that the optimal download cost with one-round communication is $D^* = \min\left\{ \left\lceil \frac{|\mathcal{P}_1|N_2}{N_2-1} \right\rceil, \left\lceil \frac{|\mathcal{P}_2|N_1}{N_1-1} \right\rceil \right\}$ in two-party PSI [72]. This means that although in MP-PSI, the privacy constraints are more stringent than that in two-party PSI, we incur no penalty for the download cost. This indeed comes from the common randomness shared among the client parties, which can be deemed as auxiliary randomness data. In addition, our achievable download cost scales linearly with the cardinality of the leader set, which outperforms the best-known MP-PSI scheme, which scales with the sum of the cardinalities of all the data sets [70].

In Chapter 5, we consider the total communication cost of SPIR. The total communication cost of SPIR consists of two parts: the total number of bits sent from the user to the databases (*upload cost*) denoted by U, and the total number of bits downloaded by the user from the databases (*download cost*) denoted by D. For a message length of L bits, the total communication cost $(U + D)$ of SPIR depends on three basic parameters (N, K, L). In [8], without any constraints on U and L,

the optimal download cost for SPIR is found to be $\frac{NL}{N-1}$, which does not depend on K. In [73], without any constraints on D and L, the optimal upload cost for SPIR is found to be $\log_2(\lceil K^{\frac{1}{N-1}} \rceil)$ which does not depend on L. In addition, [13, 15, 52, 72, 74–76] explore the optimal download cost of SPIR under various extended conditions without a consideration on the upload cost. As a classical cryptographic primitive, conditional disclosure of secrets (CDS) is first introduced in [7] as well to help devise an achievable SPIR scheme. Since CDS itself functions as an essential building block in applications such as secret sharing and attribute based encryption [77–79], CDS has also attracted significant attention as a stand-alone computer science problem. Recently, information-theoretic CDS is formulated in [80, 81] to characterize the maximum number of secret bits that can be securely disclosed per communication bit whenever a pre-defined condition is satisfied. Through extending CDS, we introduce a new concept called conditional disclosure of multiple secrets (CDMS) where two parties now share multiple i.i.d. common secrets instead of a single common secret as in CDS.

In this chapter, we investigate the overall communication cost of two-database SPIR with a particular focus on $L = 1$ in an information-theoretic setting. Our focus on $L = 1$ is motivated by two observations: 1) as pointed out in [6], when L is allowed to approach infinity, download cost dominates the upload cost, and the consideration of total cost becomes trivial. 2) in some cryptographic applications, e.g., two-party PSI and MP-PSI, only $L = 1$ may make practical sense. We first show the equivalence between a special CDMS configuration and the two-database SPIR. Following this equivalence, we explore the total communication cost of two-database

SPIR through the characteristics of CDS and CDMS. We utilize CDS/CDMS to determine an upload cost, and proceed to minimize the download cost for the given fixed upload cost. We then consider the feasible upload and download cost achievable region. In the example of $K = 3$ and $L = 1$, we find two optimal corner points for the upload and download cost pair. These two corner points outperform the best-known results in the literature [8, 73] and lead to the optimal total communication cost. Finally, we can apply total communication cost conclusion in SPIR to PSI as the PSI problem itself consists of upload cost and download cost.

In Chapter 6, we consider the random SPIR (RSPIR) problem. Gachapon is a vending machine-dispensed capsule toy by means of a roulette mechanism, which makes it random and unpredictable for customers [82]. In addition, gachapon is being adapted as a random-type item in online games and 3D printing, and its digital form is catching on quickly in the worldwide market [83,84]. Due to packaging requirements prior to official distribution, gachapon is also referred to as a *blind box* [82]. Following the concepts of gachapon as well as blind box, we introduce a *digital blind box* between a user and a server in a communication network with the following characteristics: 1) A user will ultimately receive a random box (content) from the server. However, the user does not know anything about what is in the box (what the content is) until it receives a box (content) from the server. 2) For the sake of unpredictability, a user should also know nothing about the current box (content) based on what it has received in the previous transactions. A user should not know anything about what other users have received before communicating with them. In other words, a user should not know anything beyond what it receives from

the current box (content). This requirement also protects the content privacy of the server. 3) In order to protect the privacy of the users, the server should learn nothing about what a specific user has received. Therefore, a new concept called RSPIR is put forward. In reference to the conventional SPIR, the only difference is that, in RSPIR there is no input at the user side. That is, the user does not send any queries to the databases, and ultimately receives a random message from the databases. This requirement is referred to as *random reliability*. Interestingly, the three requirements of RSPIR, namely, random reliability, database privacy and user privacy, strictly correspond to the three characteristics of the digital blind box described above. Thus, the digital blind box is equivalent to the RSPIR. An instance of RSPIR is that users can share symmetric keys if the databases operate in a broadcasting manner to transmit the information. In addition, an important variant of 1-out-of-K OT is that the receiver has no input. For example, this variant can be used as a subroutine in contract signing and certified mail protocols [65]. Thus, RSPIR can be viewed as a distributed version of this variant of 1-out-of-K OT.

In this chapter, we formulate two-database RSPIR and investigate its capacity. We determine its capacity as well as the minimal amount of required common randomness in the cases of $K = 2, 3, 4$ messages. This determines the capacity of digital blind box. While we give a general achievable scheme for any number of messages, the exact capacity of RSPIR for $K \geq 5$ remains an open problem.

In Chapter 7, we consider the federated submodel learning (FSL) problem, which is a particular federated learning (FL) model involving the PIR problem. In

FL, multiple isolated clients collaboratively perform a learning task while protecting the privacy of their stored local data against the global server [85, 86]. Recently, a new framework called FSL has been proposed to further reduce the communication and computation overhead at both server and client sides [87]. In the FSL framework, the full learning model stored in the server is divided into multiple submodels based on their data characteristics. Instead of accessing and updating the full model as in conventional FL, each selected client downloads only the needed submodel(s) from the server and then uploads the corresponding submodel updates based on the local data type in FSL. As pointed out by [87], there are two fundamental problems that can be abstracted out of the FSL framework: One is how can each client download its desired submodels without disclosing these submodel indices to the server. This is basically a *private read* problem, which is equivalent to PIR. The other is how can each client update/write-back these desired submodels still without disclosing the indices or the content of the updated submodels to the curious server. This is basically a *private write* problem, which is tightly related to oblivious random-access machine (ORAM) and secure aggregation.

At present, there are a few different FSL approaches relying on different ideas. One class of approaches is based on ORAM. Assume that the storage in the server encompasses multiple data blocks with the same size. ORAM is introduced to hide the data access pattern from the server, namely, which blocks are read/written from/to the server [88], by making sure that any two data access patterns are completely indistinguishable from the perspective of each individual database. Most ORAM schemes are based on computational security [89, 90]. Using the idea in ORAM as

reference and the X-secure T-PIR scheme in [55, 91] as a building block, [92] puts forward an FSL approach to achieve information-theoretic security. Specifically, as the databases in the server are distributed [89], the complete model is divided into submodels which are viewed as data blocks and encrypted by the server-side common randomness, and then stored across multiple distributed databases in the server. For each round of the FSL process, one client who is interested in accessing and updating a specific submodel participates in the training by sending two carefully-designed queries (a read query and a write query) to each database. Along this research line, a new technique called private read update write (PRUW) is proposed in order to further improve the total communication cost efficiency of FSL [93]. In PRUW, a user downloads (reads), updates and uploads (writes) the increments back to the chosen data blocks while taking the privacy of the content downloaded, uploaded and their positions into account simultaneously. Through over-designing the PRUW with additional server-side common randomness in storage, the communication cost is decreased notably. Moreover, PRUW is extended by incorporating gradient sparsification where only a subset of the overall parameters in the full learning model is downloaded and updated [94, 95]. Another class of approaches is based on secure aggregation. As a critical building block of FL, secure aggregation aims to aggregate the locally trained model updates from a large number of clients at the server side in a secure manner, namely, no information about each client's local training data is leaked to the others except that the aggregated result can be learned by the server [96]. Most previous secure aggregation works concentrate on computational security, see e.g., [97–99]. Similar to PSI, private set union (PSU) refers to the

problem of determining the union of elements in all the available data sets without revealing any more information than this union result [68, 100]. Using secure aggregation and PSU separately, an FSL approach is put forward in [87] with computational security guarantee. Basically, in this approach, the server first calculates the union of the clients' desired submodels through a Bloom filter-based PSU protocol. Then, through secure aggregation, the training model is updated by the clients within this submodel union at the sever side. The drawback of this approach is that the submodel union result is not accurate, and thus, the potential update efficiency of clients is not fully utilized. Recently, several secure aggregation protocols towards achieving information-theoretic security are proposed, see e.g., [101–104]. Thus, the second fundamental problem in FSL has close connections with PRUW and PSU; see detailed discussions in [105].

In this chapter, through unifying secure aggregation and PSU in the same framework, we propose a new FSL scheme that retains the main advantages of the above-mentioned two classes of approaches still with an information-theoretic security. First, the server securely calculates the clients' desired submodel index union. This is well-known as the PSU problem and referred to as FSL-PSU phase. Then, the server securely aggregates clients' generated updates in the calculated set union. This is well-known as the secure aggregation problem and referred to as FSL-write phase. In both phases, the server can only learn the ultimate result, without knowing which client has made which contribution to the ultimate result. Note that the constraints in PSI and PSU are analogous, we first establish the equivalence between PSU and MM-SPIR, and then extend PSU to multi-party PSU (MP-PSU).

Similar to typical PIR/SPIR formulations, we consider the simplest setting where the FSL server has two databases. In Fig. 1.2, we show the techniques used, their relationships, and the roadmap of the development in this chapter. The classical information-theoretic SPIR serves as a starting point to formulate our new FSL scheme. Due to the long duration of FSL process, it is possible for some clients to drop-out. Thus, we design our scheme in such a way that even if some clients lose their connection to the server, our scheme continues to work normally. It is also possible that some clients' generated answers arrive at their associated databases late and the corresponding databases make the wrong judgement that the clients have dropped-out. Our scheme is designed such that these late answers do not leak any additional information about these late clients to the databases. Moreover, our scheme continues to work normally even when some of the databases become inactive, especially when the total number of databases is large enough. Finally, our FSL scheme can be run iteratively in multiple rounds until a pre-defined termination criterion is met.

In Chapter 8, we consider the FSL problem in a distributed storage system where our above-mentioned two-database FSL approach is extended by considering more databases at the server side. As the number of databases is large now, several issues may arise, such as: a set of databases may be captured by an eavesdropper who can passively read database content and listen to database communications to capture database and client information [53]; a set of databases may fail [106]. To solve the eavesdropping and database failure challenges in a distributed storage setting, [107] proposes secure regenerating codes, where the eavesdropper learns no

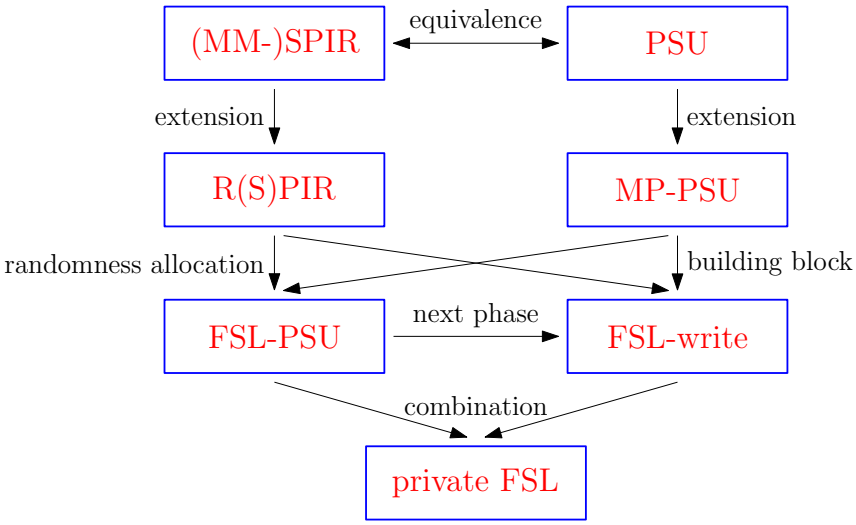


Figure 1.2: Techniques used, their relationships, and the roadmap of the development of the private FSL in Chapter 7.

useful information, and a replacement database can be constructed to replace the failed database by communicating with the remaining databases. Further, classical secret sharing refers to a setting where a secret is shared among multiple parties in such a way that any t parties can recover the secret, but any fewer than t parties learns nothing about the secret [108]. In ramp secret sharing [109], a ramp zone is established such that, any t parties can recover the secret, any $\tilde{t}$ parties learns nothing about the secret, and a set of parties whose cardinality is between $\tilde{t}$ and t learns partial knowledge about the secret. This partial knowledge can be quantified by the mutual information and goes up as the cardinality of this set of parties in the ramp zone increases. By combining the idea in ramp secret sharing [109] together with the idea in secure regenerating codes [107], we put forward a new customized coding scheme that we coin ramp secure regenerating code (RSRC) to develop an FSL scheme that is resilient to passive adversaries and database equipment failures. Hence, in our RSRC scheme, the useful information learned by the eavesdropper is

quantifiable and controllable, and the performance of our RSRC-based distributed FSL scheme is adjustable. Further, a set of databases may be captured by an adversary who may actively overwrite the responses generated by its controlled databases [25]. In this case, the clients will receive erroneous information. The performance of an FSL scheme is evaluated by three critical metrics: computation cost, communication cost, and storage cost. Towards achieving information-theoretic security, the schemes generally rely on operations in a finite field. As these operations are simple, the computation cost can be neglected. Moreover, since we are concentrating on distributed storage across the databases, we consider only the server-side storage cost and neglect the client-side storage cost.

In this chapter, we propose a new RSRC-based FSL scheme that is efficient in terms of communication cost and server-side storage cost under the circumstance of distributed storage. We also prove that our proposed scheme is fully robust against permanent database failures, eavesdroppers, active adversaries, database drop-outs, client droup-outs, client late-arrivals, for one-round distributed FSL. Again, this one-round distributed FSL scheme can be performed in an iterative manner until a termination criterion is satified.

In Chapter 9, we present the conclusions of this dissertation.

CHAPTER 2

Private Set Intersection: A Multi-Message Symmetric Private Information Retrieval Perspective

2.1 Introduction

In this chapter, we study the two-party PSI problem where each party stores its personal data set across multiple non-colluding and replicated databases. In particular, there are two entities E_i, for $i = 1, 2$, each storing a data set $\mathcal{P}_i$, whose elements are picked from a finite set $\mathbb{S}_K$, on N_i replicated and non-colluding databases. It is required to determine the set intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ without leaking any information about the remaining elements to the other entity, and to do this with the least amount of downloaded bits. We first show that the two-party PSI problem can be recast as an MM-SPIR problem with certain added restrictions. Next, as a stand-alone result, we derive the information-theoretic sum capacity of MM-SPIR, $C_{MM-SPIR}$. We show that with K messages, N databases, and a given size of the desired message set P, the exact capacity of MM-SPIR is $C_{MM-SPIR} = 1 - \frac{1}{N}$ when $P \leq K - 1$, provided that the entropy of the server-side common randomness S satisfies $H(S) \geq \frac{P}{N-1}$ per desired symbol. When $P = K$, the MM-SPIR capacity is

trivially 1 without the need for any server-side common randomness S. This result implies that there is no gain for MM-SPIR over successive SM-SPIR. For the MM-SPIR problem, we present a novel capacity-achieving scheme which builds seamlessly over the near-optimal MM-PIR scheme in [23] without any database privacy constraints. Surprisingly, our scheme here is exactly optimal for MM-SPIR for any P, in contrast to the scheme for MM-PIR, which was proved only to be near-optimal. Our scheme is an alternative to the successive usage of the SM-SPIR scheme in [8]. Based on this capacity result for the MM-SPIR problem, and after addressing the added requirements in its conversion to the two-party PSI problem, we show that the optimal download cost for two-party PSI is given by $\min\left\{\left\lceil\frac{P_1N_2}{N_2-1}\right\rceil, \left\lceil\frac{P_2N_1}{N_1-1}\right\rceil\right\}$, where P_i is the cardinality of set $\mathcal{P}_i$.

2.2 PSI: Problem Formulation

Consider the problem of privately determining the intersection of two sets (or lists) picked from a finite set[1] $\mathbb{S}_K$. For convenience, we denote a random variable and its realization by using the same general uppercase letter when distinction is clear from the context. We address this issue additionally whenever clarification is needed. Consider a setting where there are two entities E_1 and E_2. For $i = 1, 2$, the entity E_i

[1]The restriction of generating the set from a finite set is without loss of generality as the set elements of any kind can be mapped into corresponding finite set elements for sufficiently large size. For example, the elements of the set that contains the names of suspected terrorists in the United States can be mapped into elements from the finite set $\mathbb{S}_K$, where K is the population size on this planet. As we will show next, the download cost is independent of K. Hence, the optimization of the alphabet size is irrelevant to our formulation. Nevertheless, it is advisable to choose K to be the lowest integer such that $\mathcal{P}_1, \mathcal{P}_2 \subseteq \mathbb{S}_K$ to minimize the upload cost. It suffices to have $K > P_1 + P_2$.

stores a set $\mathcal{P}_i$. For each element of the finite set $\mathbb{S}_K$, the entity E_i adds[2] this element to its set $\mathcal{P}_i$ independently from the remaining field elements with probability q_i. In this work, we focus on the case of $q_i = \frac{1}{2}$ for $i = 1, 2$. After generation of the set $\mathcal{P}_i$, the cardinality of $\mathcal{P}_i \subseteq \mathbb{S}_K^{P_i}$ is denoted by $|\mathcal{P}_i| = P_i$, and is public knowledge.[3] The entity E_i stores $\mathcal{P}_i$ in a replicated fashion on N_i replicated and non-colluding databases.

The entities E_1 and E_2 want to compute the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ privately (see Fig. 2.1). To that end, the entity[4] E_1 sends N_2 queries to the databases associated with E_2. Specifically, E_1 sends the query $Q_{n_2}^{[\mathcal{P}_1]}$ to the n_2th database for all $n_2 \in [N_2]$, where $[N_2]$ (and also $[1 : N_2]$) denotes integers from 1 to N_2. Since E_1 does not know $\mathcal{P}_2$ in advance, it generates the queries $Q_{1:N_2}^{[\mathcal{P}_1]} = \left\{ Q_{n_2}^{[\mathcal{P}_1]} : n_2 \in [N_2] \right\}$ independently from $\mathcal{P}_2$, hence,

$$I(Q_{1:N_2}^{[\mathcal{P}_1]}; \mathcal{P}_2) = 0 \tag{2.1}$$

The databases associated with E_2 respond truthfully with answers $A_{1:N_2}^{[\mathcal{P}_1]} = \left\{ A_{n_2}^{[\mathcal{P}_1]} : n_2 \in [N_2] \right\}$. The n_2th answer $A_{n_2}^{[\mathcal{P}_1]}$ is a deterministic function of the set $\mathcal{P}_2$,

[2]We note that our achievability scheme works for any statistical distribution imposed on the sets, i.e., the i.i.d. generation assumption presented here is not needed for the achievability proof.

[3]We note that choosing to have P_i to be a global knowledge is for the consistency with MM-SPIR problem and convenient execution. This knowledge enables the entities to determine which entity should initiate the PSI process to have the least download cost (or if any is needed at all, as in the case of $P_i = K$, for an i; see Remark 2.1). If the cardinalities are not public knowledge, our achievability works by choosing one of the entities arbitrarily to initiate the PSI process assuming that the other entity has sufficient common randomness. We note, however, that keeping the cardinalities private is indeed a challenging problem and it is outside the scope of this work.

[4]The entities E_1, E_2 should agree on a specific order of retrieval operations such that this order results in the minimal download cost. Without loss of generality, we assume here that the optimal order of operation starts with entity E_1 sending queries to the databases associated with entity E_2.

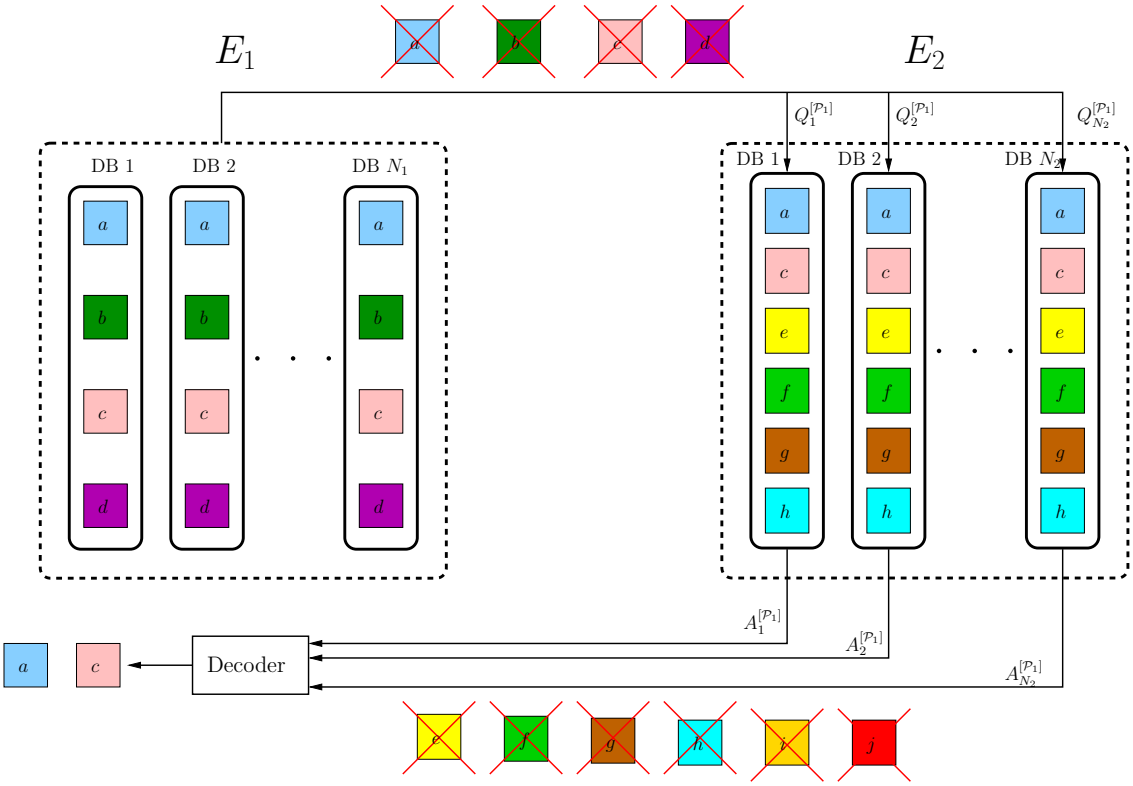


Figure 2.1: Example for the private set intersection (PSI) problem. E_1 has the set $\mathcal{P}_1 = \{a, b, c, d\}$ and E_2 has the set $\mathcal{P}_2 = \{a, c, e, f, g, h\}$. E_1 submits queries to E_2 that do not leak information about $\mathcal{P}_1$, while E_2 responds with answers that do not leak information about e, f, g, h (or non-existence of i, j). By decoding the answers, E_1 learns that $\mathcal{P}_1 \cap \mathcal{P}_2 = \{a, c\}$.

the query $Q_{n_2}^{[\mathcal{P}_1]}$ and the existing common randomness S, thus,

$$H(A_{n_2}^{[\mathcal{P}_1]}|Q_{n_2}^{[\mathcal{P}_1]}, \mathcal{P}_2, S) = 0, \quad n_2 \in [N_2] \tag{2.2}$$

Denote the cardinality of the intersection $|\mathcal{P}_1 \cap \mathcal{P}_2|$ by M. The entity[5] E_1 should be able to reliably compute the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ based on the sent queries $Q_{1:N_2}^{[\mathcal{P}_1]}$, the collected answers $A_{1:N_2}^{[\mathcal{P}_1]}$ and the knowledge of $\mathcal{P}_1$ without knowing M in advance.

[5]After calculating $\mathcal{P}_1 \cap \mathcal{P}_2$ at E_1, the entity E_1 sends the result of $\mathcal{P}_1 \cap \mathcal{P}_2$ directly to E_2 if needed.

This is captured by the following PSI reliability constraint,

$$[\text{PSI reliability}] \qquad H(\mathcal{P}_1 \cap \mathcal{P}_2 | Q_{1:N_2}^{[\mathcal{P}_1]}, A_{1:N_2}^{[\mathcal{P}_1]}, \mathcal{P}_1) = 0 \tag{2.3}$$

The privacy requirements can be expressed as the following two privacy constraints: E_1 privacy and E_2 privacy. First, the queries sent by E_1 should not leak any information about[6] $\mathcal{P}_1$, i.e., any individual database associated with E_2 learns nothing about $\mathcal{P}_1$ from the query $Q_{n_2}^{[\mathcal{P}_1]}$, the answer $A_{n_2}^{[\mathcal{P}_1]}$, the knowledge of $\mathcal{P}_2$ and the existing common randomness S,

$$[E_1 \text{ privacy}] \qquad I(\mathcal{P}_1; Q_{n_2}^{[\mathcal{P}_1]}, A_{n_2}^{[\mathcal{P}_1]}, \mathcal{P}_2, S) = 0, \quad n_2 \in [N_2] \tag{2.4}$$

Second, E_1 should not be able to learn anything further than $\mathcal{P}_1 \cap \mathcal{P}_2$, i.e., E_1 should not learn the elements in $\mathcal{P}_2$ other than the intersection, $\mathcal{P}_2 \setminus (\mathcal{P}_1 \cap \mathcal{P}_2) = \mathcal{P}_2 \setminus \mathcal{P}_1$. Moreover[7], E_1 should not learn the absence of the remaining field elements in E_2, i.e., the set $\overline{(\mathcal{P}_1 \cup \mathcal{P}_2)}$. Thus, E_1 should learn nothing about whether E_2 contains $(\mathcal{P}_2 \setminus \mathcal{P}_1) \cup \overline{(\mathcal{P}_1 \cup \mathcal{P}_2)} = \bar{\mathcal{P}}_1$ or not (we denote this information by $E_{2,\bar{\mathcal{P}}_1}$) from the

[6] While checking the presence of elements of $\mathcal{P}_1$ in $\mathcal{P}_2$, E_1 wants to protect $\mathcal{P}_1 \setminus \mathcal{P}_2$. However, since E_1 does not know $\mathcal{P}_2$, the queries cannot depend on $\mathcal{P}_2$ (see also (2.1)), and E_1 should protect all of $\mathcal{P}_1$ in queries.

[7] Although it is tempting to formulate the E_2 privacy constraint as $I(\mathcal{P}_2 \setminus \mathcal{P}_1; A_{1:N_2}^{[\mathcal{P}_1]}) = 0$, this constraint permits leaking information about the remaining field elements that do not exist in $\mathcal{P}_2$. More specifically, if we adopted this constraint in the example in Fig. 2.1, the answers should not leak information about e, f, g, h, however, E_1 may learn that the elements i, j do not exist in $\mathcal{P}_2$. To properly formalize the constraint that E_1 learns nothing other than the intersection, we need to protect $\overline{(\mathcal{P}_1 \cup \mathcal{P}_2)}$ as well.

collected answers $A_{1:N_2}^{[\mathcal{P}_1]}$ given the generated queries $Q_{1:N_2}^{[\mathcal{P}_1]}$ and the knowledge of $\mathcal{P}_1$,

$$[E_2 \text{ privacy}] \qquad I(E_{2,\bar{\mathcal{P}}_1}; Q_{1:N_2}^{[\mathcal{P}_1]}, A_{1:N_2}^{[\mathcal{P}_1]}, \mathcal{P}_1) = 0 \tag{2.5}$$

For given finite set size K, set sizes P_1 and P_2, and number of databases N_1 and N_2, an achievable PSI scheme is a scheme that satisfies the PSI reliability constraint (2.3), the E_1 privacy constraint (2.4), and the E_2 privacy constraint (2.5). In this chapter, we measure the efficiency of a scheme by the maximal number of downloaded bits by one of the entities E_1 or E_2 in order to compute $\mathcal{P}_1 \cap \mathcal{P}_2$. We denote the maximal number of downloaded bits by D. Then, the optimal download cost is $D^* = \inf D$ over all achievable PSI schemes.[8]

2.3 From PSI to MM-SPIR

In this section, we show that the PSI problem can be reduced to an MM-SPIR problem, if the entities allow storing their sets in a specific searchable format. This transformation has the same flavor as [110] and [42], where the original contents of the databases are mapped into searchable lists to enable PIR, which assumes that

[8] A more natural efficiency metric is to consider the sum of the maximal number of uploaded bits (denoted by U) and the maximal number of downloaded bits (denoted by D) by one of the entities E_1 or E_2 to compute $\mathcal{P}_1 \cap \mathcal{P}_2$. In this case, the most efficient scheme is the scheme with the lowest communication cost, i.e., that achieves the optimal communication cost $C^* = \inf(U + D)$ over all achievable PSI schemes. The SPIR problem [8] under combined upload and download costs is still an open problem. As we will see, our framework builds on the SPIR problem. Therefore, in this work, we consider only the download cost. The PSI under combined upload and download costs is an interesting future direction, which is outside the scope of this chapter. In Section 2.7.2, we provide an illustrative example to show that the upload cost can be reduced without affecting the download cost. Nevertheless, we argue that if the PSI determination is repeated (for example, if one list is kept the same and the other list is regularly updated, we always use the fixed list to initiate the PSI process), the queries could be used repeatedly without compromising the user privacy as long as the databases do not collude. In this case, the upload cost would not scale with the number of PSI determination rounds, unlike the download cost.

the user knows the position of the desired file in the databases. To that end, define the incidence vector $X_i \in \mathbb{F}_2^K$ as a binary vector of size K associated with the set $\mathcal{P}_i$. Denote the jth element of the incidence vector X_i by $X_i(j)$ where

$$X_i(j) = \begin{cases} 1, & j \in \mathcal{P}_i \\ 0, & j \notin \mathcal{P}_i \end{cases} \tag{2.6}$$

for all $j \in \mathbb{S}_K$. Hence, $X_i(j)$ is an i.i.d. random variable for all $j \in [K]$ such that $X_i(j) \sim \text{Ber}(q_i)$. The entity E_i constructs the incidence vector X_i corresponding to the set $\mathcal{P}_i$ (see Fig. 2.2). The entity E_i replicates the vector X_i at all of its N_i associated databases (see Fig. 2.3). Note that X_i is a sufficient statistic for $\mathcal{P}_i$ for a given K. The PSI determination process is performed over X_1 or X_2, and not over the original $\mathcal{P}_1$ or $\mathcal{P}_2$.

To solidify ideas, we state the variables defined so far explicitly over a specific example. Consider the example in Fig. 2.1. Here, the entity E_1 has the set $\mathcal{P}_1 = \{a, b, c, d\}$ and the entity E_2 has the set $\mathcal{P}_2 = \{a, c, e, f, g, h\}$. Therefore, the intersection is $\mathcal{P}_1 \cap \mathcal{P}_2 = \{a, c\}$. Let us assume that the alphabet, $\mathcal{P}_{alph}$, for this example is $\mathcal{P}_{alph} = \{a, b, c, d, e, f, g, h, i, j\}$ as shown in Fig. 2.2. Then, the incidence vectors at the entities are $X_1 = [1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0]$ and $X_2 = [1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0]$, which are also shown in Fig. 2.2. For this example, $P_1 = 4$, $P_2 = 6$, $K = 10$, and $M = 2$. Finally, the MM-SPIR is conducted over the replicated incidence vectors at the two entities as shown in Fig. 2.3.

Without loss of generality, assume that E_1 initiates the PSI process. E_1 does

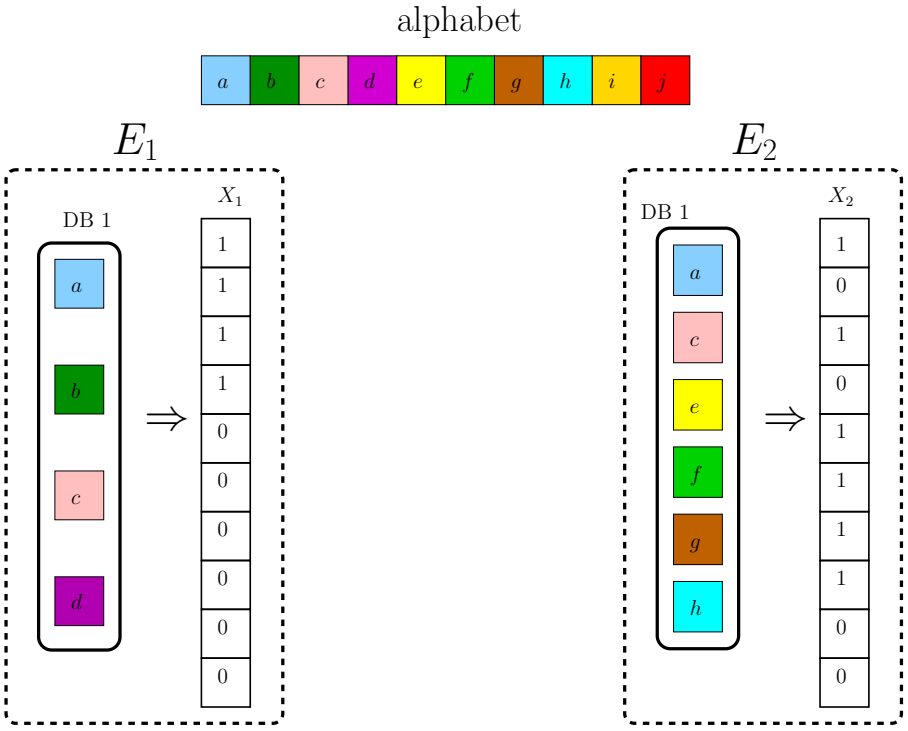


Figure 2.2: Example for the transformation from sets to incidence vectors. E_1 has the set $\mathcal{P}_1 = \{a, b, c, d\}$ and E_2 has the set $\mathcal{P}_2 = \{a, c, e, f, g, h\}$. The alphabet is $\mathcal{P}_{alph} = \{a, b, c, d, e, f, g, h, i, j\}$. Entity E_i constructs an incidence vector X_i to facilitate MM-SPIR.

not know M in advance. The only information E_1 has is $\mathcal{P}_1$. Consequently, E_1 wants to verify the existence of each element of $\mathcal{P}_1$ in $\mathcal{P}_2$ to deduce $\mathcal{P}_1 \cap \mathcal{P}_2$. Thus, E_1 needs to jointly and reliably download the bits $W_{\mathcal{P}_1} = \{X_2(j) : j \in \mathcal{P}_1\}$ by sending N_2 queries to the databases associated with E_2 and collecting the corresponding answers with the knowledge of X_1, i.e., $H(\mathcal{P}_1 \cap \mathcal{P}_2 | W_{\mathcal{P}_1}, X_1) = 0$. Hence, we can write the PSI reliability constraint as,

$$H(W_{\mathcal{P}_1} | Q_{1:N_2}^{[\mathcal{P}_1]}, A_{1:N_2}^{[\mathcal{P}_1]}, X_1) = 0 \tag{2.7}$$

This is exactly the reliability constraint in MM-SPIR noting that $\mathcal{P}$ is known to the

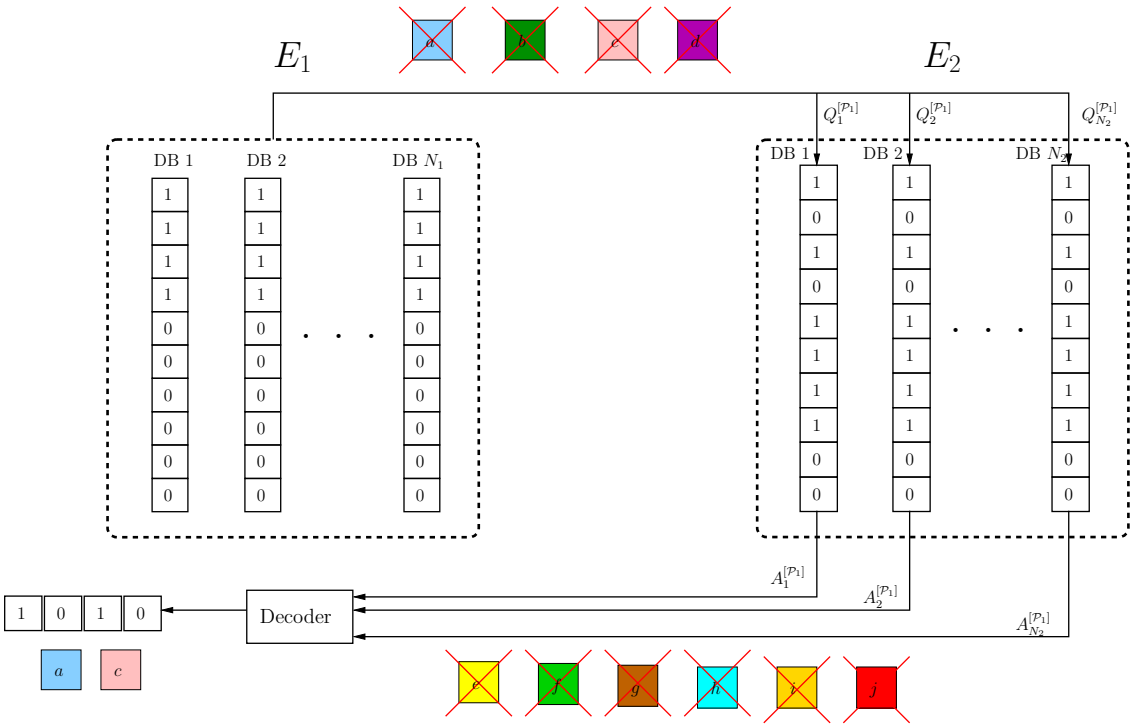


Figure 2.3: Example for the transformation from the PSI problem to an MM-SPIR problem. E_1 needs to retrieve the elements corresponding to $\mathcal{P}_1$ from the incidence vector X_2 without revealing $\mathcal{P}_1$, while E_2 responds with answer strings that do not leak $\bar{\mathcal{P}}_1$.

user; see Section 2.5.1. Meanwhile, given the knowledge of X_1 and the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$, $W_{\mathcal{P}_1}$ can also be deduced by E_1, i.e., $H(W_{\mathcal{P}_1}|\mathcal{P}_1 \cap \mathcal{P}_2, X_1) = 0$. Hence, the MM-SPIR reliability constraint can revert back to the PSI reliability constraint.

Since E_1 is searching for the existence of all elements of $\mathcal{P}_1$ in $\mathcal{P}_2$ without leaking any information about $\mathcal{P}_1$ to any individual database associated with E_2, the E_1 privacy constraint in (2.4) dictates,

$$I(\mathcal{P}_1; Q_{n_2}^{[\mathcal{P}_1]}, A_{n_2}^{[\mathcal{P}_1]}, X_2, S) = 0, \quad n_2 \in [N_2] \tag{2.8}$$

This is exactly the privacy constraint in MM-SPIR if we treat X_2 as the messages

in the databases with length 1; see Section 2.5.1.

As the databases associated with E_2 store X_2 now, to ensure the E_2 privacy constraint in (2.5), the answers from E_2 databases should not leak anything about $E_{2,\bar{\mathcal{P}}_1}$, which can be further mapped to not leaking any information about $W_{\bar{\mathcal{P}}_1} = \{X_2(j) : j \notin \mathcal{P}_1\}$ as,

$$I(W_{\bar{\mathcal{P}}_1}; Q_{1:N_2}^{[\mathcal{P}_1]}, A_{1:N_2}^{[\mathcal{P}_1]}, X_1) = 0 \tag{2.9}$$

This is exactly the database privacy constraint in MM-SPIR as $\mathcal{P}$ is known to the user; see Section 2.5.1.

Consequently, with the cardinality of sets in each entity being global knowledge, the PSI problem is formally equivalent to the MM-SPIR problem with i.i.d. messages of length 1 bit each (also see Fig. 2.3), when the entities E_1 and E_2 are allowed to construct the corresponding incidence vectors for the original sets $\mathcal{P}_1$ and $\mathcal{P}_2$. The message length constraint of 1 bit per message, i.e., $H(W_k) = 1$ for all $k \in [K]$, comes due to messages representing incidences in the SPIR problem. The i.i.d. property of the messages that we have here in this chapter is a consequence of the i.i.d. generation of the sets with probability q_i, and it is not true in general. In Section 2.5, we derive in detail the capacity of the MM-SPIR problem (see also Section 2.6), which in turn gives the most efficient information-theoretic PSI scheme in terms of the download cost.

2.4 Main Result

In this section, we present our main result concerning the PSI problem. The result provides the optimal (minimum) download cost for the PSI problem under the assumptions in Sections 2.2 and 2.3. The result is based on the optimal download cost of the MM-SPIR problem, which is presented in detail in Section 2.5; see also Section 2.6.

Theorem 2.1 *In the PSI problem, the elements of the sets are added independently with probability $q_i = \frac{1}{2}$ from a finite set of size K. Once the set generation is finished, the fixed set $\mathcal{P}_1$ where $|\mathcal{P}_1| = P_1 < K$ is stored among N_1 databases and the fixed set $\mathcal{P}_2$ where $|\mathcal{P}_2| = P_2 < K$ is stored among N_2 databases. The set cardinalities P_1 and P_2 are made public. The amount of common randomness satiesfies $H(S) \geq \min\{\left\lceil \frac{P_1}{N_2-1} \right\rceil, \left\lceil \frac{P_2}{N_1-1} \right\rceil\}$. Then, the optimal download cost with one-round communication (one entity sends the queries to the other entity and then receives feedback) is,*

$$D^* = \min\left\{ \left\lceil \frac{P_1 N_2}{N_2 - 1} \right\rceil, \left\lceil \frac{P_2 N_1}{N_1 - 1} \right\rceil \right\} \tag{2.10}$$

The proof of Theorem 2.1 is a direct consequence of the capacity result for MM-SPIR presented in Section 2.5; see also Section 2.6. We have the following remarks.

Remark 2.1 *In the special case of having $P_i = K$ for $i = 1$ or $i = 2$, the download*

cost is trivially zero. This is due to the fact that if $P_1 = K$ for example, the entity E_2 directly concludes that the intersection $\mathcal{P}_1 \cap \mathcal{P}_2 = \mathcal{P}_2$ without sending any queries to E_1 or requiring any common randomness.

Remark 2.2 *The relationship $M = |\mathcal{P}_1| + |\mathcal{P}_2| - |\mathcal{P}_1 \cup \mathcal{P}_2| \geq |\mathcal{P}_1| + |\mathcal{P}_2| - K$ is always satisfied automatically. However, in the special case of $M = |\mathcal{P}_1| + |\mathcal{P}_2| - K$, the entire list of the entity that starts the PSI determination will be inevitably leaked to the second entity, as the list sizes $|\mathcal{P}_1|, |\mathcal{P}_2|$ are globally known. Consequently, our results hold for the strict inequality case $M > |\mathcal{P}_1| + |\mathcal{P}_2| - K$. It is worth noting that this restriction is due to the nature of the PSI problem itself and not an artifact of our proposed scheme. Furthermore, entity 1 cannot simply announce its list directly as the cardinality of the intersection set M is unknown in advance.*

Remark 2.3 *The* min *term in Theorem 2.1 comes from the fact that either entity can initiate the PSI determination process so that the overall download cost is minimized.*

Remark 2.4 *We note that although our result is exact, i.e., the download cost capacity (in the sense of matching achievability and converse proofs) under the assumptions of independent generation model for the lists with $q_i = \frac{1}{2}$, our scheme is achievable for any list generation model with arbitrary q_i (see Footnote 2).*

Remark 2.5 *Our result is private in information-theoretic (absolute) sense and does not need any assumptions about the computational powers of the entities. Furthermore, the achievable scheme is fairly simple and easy to implement compared to*

the fully homomorphic encryption needed in [3]. A drawback of our approach is that it needs multiple non-colluding databases (N_1 or N_2 needs to be strictly larger than 1), otherwise, our scheme is infeasible.

Remark 2.6 *The linear scalability of our scheme matches the linear scalability of the best-known set intersection algorithms without any privacy constraints.*

2.5 MM-SPIR as a Stand-Alone Problem

In this section, we consider the MM-SPIR problem. We present the problem in a stand-alone format, i.e., we present a formal problem description in Section 2.5.1, followed by the main result in Section 2.5.2, the converse in Section 2.5.3, and a novel achievability in Section 2.5.4.

2.5.1 MM-SPIR: Formal Problem Description

There are N non-colluding databases each storing K i.i.d. messages. Each message is composed of L [9] i.i.d. and uniformly chosen symbols from a sufficiently large finite field $\mathbb{F}_q$. Then,

$$H(W_k) = L, \quad k \in [K] \tag{2.11}$$

$$H(W_{1:K}) = KL \tag{2.12}$$

In the MM-SPIR problem, our goal is to retrieve a set of messages $W_{\mathcal{P}}$ out

[9] As in most PIR problems, the message length L can approach infinity.

of the K available messages without leaking any information regarding the index set $\mathcal{P}$ to any individual database where $\mathcal{P} = \{i_1, i_2, \cdots, i_P\} \subseteq [K]$ such that its cardinality is $|\mathcal{P}| = P$.[10] We assume that the cardinality of the potential message set, P, is known to all databases in the server. This is the user privacy constraint. In addition, our goal is to not retrieve any messages beyond the desired set of messages $W_{\mathcal{P}}$. This is the database privacy constraint.

Following the SPIR formulation in [8], let $\mathcal{F}$ denote the randomness in the retrieving strategy adopted by the user. Because of the user privacy constraint, $\mathcal{F}$ is a random variable whose realization is only known to the user, but is unknown to the databases. A necessary common randomness S must be shared among the N databases to satisfy the database privacy constraint. The random variable S is generated independent of the message set $W_{1:K}$. Similarly, $\mathcal{F}$ is independent of $W_{1:K}$ as the user does not know message realizations in advance. Moreover, $\mathcal{F}$ and S are generated independently without knowing the desired index set $\mathcal{P}$. Then,

$$H(\mathcal{F}, S, \mathcal{P}, W_{1:K}) = H(\mathcal{F}) + H(S) + H(\mathcal{P}) + H(W_{1:K}) \tag{2.13}$$

To perform MM-SPIR, a user generates one query $Q_n^{[\mathcal{P}]}$ for each database according to the randomness $\mathcal{F}$ and then sends it to the nth database. Hence, the queries $Q_{1:N}^{[\mathcal{P}]}$ are deterministic functions of $\mathcal{F}$, i.e.,

$$H(Q_1^{[\mathcal{P}]}, Q_2^{[\mathcal{P}]}, \cdots, Q_N^{[\mathcal{P}]} | \mathcal{F}) = 0, \quad \forall \mathcal{P} \tag{2.14}$$

[10] We use the symbol $\mathcal{P}$ to denote the random variable corresponding to the desired set and its realization with little abuse of notation.

Combining (2.13) and (2.14), the queries are independent of the messages, i.e.,

$$I(Q_{1:N}^{[\mathcal{P}]}; W_{1:K}) = 0 \tag{2.15}$$

After receiving a query from the user, each database truthfully generates an answer string based on the messages and the common randomness, hence,

$$H(A_n^{[\mathcal{P}]} | Q_n^{[\mathcal{P}]}, W_{1:K}, S) = 0, \quad \forall n, \forall \mathcal{P} \tag{2.16}$$

After collecting all the answer strings from the N databases, the user should be able to decode the desired messages $W_{\mathcal{P}}$ reliably, therefore,

$$\text{[reliability]} \quad H(W_{\mathcal{P}} | A_{1:N}^{[\mathcal{P}]}, Q_{1:N}^{[\mathcal{P}]}, \mathcal{F}) \overset{(2.14)}{=} H(W_{\mathcal{P}} | A_{1:N}^{[\mathcal{P}]}, \mathcal{F}) = 0, \quad \forall \mathcal{P} \tag{2.17}$$

In order to protect the user's privacy, the query generated to retrieve the set of messages $W_{\mathcal{P}_1}$ should be statistically indistinguishable from the one generated to retrieve the set of messages $W_{\mathcal{P}_2}$ where $|\mathcal{P}_1| = |\mathcal{P}_2| = P$, i.e.,

$$\begin{aligned} \text{[user privacy]} \quad & (Q_n^{[\mathcal{P}_1]}, A_n^{[\mathcal{P}_1]}, W_{1:K}, S) \\ & \sim (Q_n^{[\mathcal{P}_2]}, A_n^{[\mathcal{P}_2]}, W_{1:K}, S), \ \forall n, \ \forall \mathcal{P}_1, \mathcal{P}_2 \text{ s.t. } |\mathcal{P}_i| = P \end{aligned} \tag{2.18}$$

The user privacy constraint in (2.18) is equivalent to,

$$\text{[user privacy]} \quad I(\mathcal{P}; Q_n^{[\mathcal{P}]}, A_n^{[\mathcal{P}]}, W_{1:K}, S) = 0, \quad \forall \mathcal{P} \tag{2.19}$$

In order to protect the databases' privacy, the user should learn nothing about $W_{\bar{\mathcal{P}}}$ which is the complement of $W_{\mathcal{P}}$, i.e., $W_{\bar{\mathcal{P}}} = W_{1:K} \backslash W_{\mathcal{P}}$,

$$\text{[database privacy]} \quad I(W_{\bar{\mathcal{P}}}; Q_{1:N}^{[\mathcal{P}]}, A_{1:N}^{[\mathcal{P}]}, \mathcal{F}) = 0, \quad \forall \mathcal{P} \tag{2.20}$$

An achievable MM-SPIR scheme is a scheme that satisfies the MM-SPIR reliability constraint (2.17), the user privacy constraint (2.18)-(2.19), and the database privacy constraint (2.20). The efficiency of the scheme is measured in terms of the maximal number of downloaded bits by the user from all the databases, denoted by $D_{MM-SPIR}$. Thus, the sum retrieval rate of MM-SPIR is given by

$$R_{MM-SPIR} = \frac{PL}{D_{MM-SPIR}} \tag{2.21}$$

The sum capacity of MM-SPIR, $C_{MM-SPIR}$, is the supremum of the sum retrieval rates $R_{MM-SPIR}$ over all achievable schemes.

2.5.2 MM-SPIR: Main Results

Our stand-alone result for MM-SPIR is stated in the following theorem. We only consider $N \geq 2$ as SPIR is infeasible for $N = 1$.

Theorem 2.2 *The MM-SPIR capacity for $N \geq 2$, $K \geq 2$, and a fixed $P \leq K$, is*

given by,

$$C_{MM-SPIR} = \begin{cases} 1, & P = K \\ 1 - \frac{1}{N}, & 1 \leq P \leq K-1,\ H(S) \geq \frac{PL}{N-1} \\ 0, & \textit{otherwise} \end{cases} \tag{2.22}$$

The converse proof is given is Section 2.5.3, and the achievability proof is given in Section 2.5.4. We have the following remarks concerning Theorem 2.2.

Remark 2.7 *The result implies that the capacity of MM-SPIR is exactly the same as the capacity of SM-SPIR [8]. Hence, there is no gain from joint retrieval in comparison to successive single-message SPIR [8]. This in contrast to the gain in MM-PIR [23] in comparison to successive single-message PIR [6]. MM-SPIR capacity expression in Theorem 2.2 inherits all of the structural remarks from [8].*

Remark 2.8 *Similar to the SM-SPIR problem, we observe a threshold effect on the size of the required common randomness. Specifically, we note that there is a minimal required size for the common randomness above which the problem is feasible. This threshold is P times the threshold in SM-SPIR. Using a common randomness in the amount of the threshold achieves the full capacity, and there is no need to use any more randomness than the threshold.*

Remark 2.9 *For the extreme case of $P = K$, the SPIR capacity is* 1 *without using any common randomness. This is due to the fact that the user privacy and the database privacy constraints are trivially satisfied, and hence the user can simply*

download all of the messages from one of the databases without using any common randomness.

2.5.3 MM-SPIR: Converse Proof

In this section, we derive the converse for Theorem 2.2. In the converse proof, we focus on the case $P \leq K-1$. Because when $P = K$, the trivial upper bound for the retrieval rate $R \leq 1$ and the trivial lower bound for the common randomness $H(S) \geq 0$ suffice. Further, we exclusively focus on the case $K \geq 3$. When $K = 1$, we have $P = 1$, and the converse trivially follows since $P = K$. When $K = 2$: If $P = 2$, the converse trivially follows from the converse of $P = K$, and when $P = 1$, the converse follows from the converse of SM-SPIR [8].

Now, focusing on the case $K \geq 3$, and $P \leq K-1$, the total number of possible choices for the index set $\mathcal{P}$ is $\beta = \binom{K}{P} \geq 3$. Thus, there always exist at least three non-identical index sets $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ such that $|\mathcal{P}_i| = P$, $i = 1, 2, 3$.

To prove the converse of Theorem 2.2, we first need the following lemmas. Lemmas 2.1 are 2.2 are direct extensions to [8, Lemmas 1 and 2] to the setting of MM-SPIR. Lemma 2.1 simply states that an answer string $A_n^{[\mathcal{P}_1]}$ which is received at the user to retrieve $W_{\mathcal{P}_1}$ has the same size as $A_n^{[\mathcal{P}_2]}$, i.e., all answer strings are symmetric in length, even if we condition over the desired message set $W_{\mathcal{P}_1}$. This lemma is a direct consequence of the user privacy constraint.

Lemma 2.1 (Symmetry)

$$H(A_n^{[\mathcal{P}_1]}|W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_1]}) = H(A_n^{[\mathcal{P}_2]}|W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_2]}), \quad \forall n,\ \forall \mathcal{P}_1, \mathcal{P}_2\ s.t.\ \mathcal{P}_1 \neq \mathcal{P}_2, |\mathcal{P}_1| = |\mathcal{P}_2| \tag{2.23}$$

$$H(A_n^{[\mathcal{P}_1]}|Q_n^{[\mathcal{P}_1]}) = H(A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}), \qquad \forall n,\ \forall \mathcal{P}_1, \mathcal{P}_2\ s.t.\ \mathcal{P}_1 \neq \mathcal{P}_2, |\mathcal{P}_1| = |\mathcal{P}_2| \tag{2.24}$$

Proof: From the user privacy constraint (2.18), we have

$$H(A_n^{[\mathcal{P}_1]}, W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_1]}) = H(A_n^{[\mathcal{P}_2]}, W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_2]}) \tag{2.25}$$

$$H(W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_1]}) = H(W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_2]}) \tag{2.26}$$

Using the definition of conditional entropy $H(X|Y) = H(X,Y) - H(Y)$, we obtain (2.23). The proof of (2.24) follows from the user privacy constraint as well with noting that $H(A_n^{[\mathcal{P}_1]}, Q_n^{[\mathcal{P}_1]}) = H(A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$ and $H(A_n^{[\mathcal{P}_1]}) = H(A_n^{[\mathcal{P}_2]})$. ■

Next, Lemma 2.2 states that knowing the user's private randomness $\mathcal{F}$ does not help in decreasing the uncertainty of the answer string $A_n^{[\mathcal{P}]}$.

Lemma 2.2 (Effect of conditioning on user's randomness)

$$H(A_n^{[\mathcal{P}]}|W_{\mathcal{P}}, \mathcal{F}, Q_n^{[\mathcal{P}]}) = H(A_n^{[\mathcal{P}]}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}), \quad \forall n, \forall \mathcal{P} \tag{2.27}$$

Proof: We start with the following mutual information,

$$I(A_n^{[\mathcal{P}]}; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) \leq I(A_n^{[\mathcal{P}]}, W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) \tag{2.28}$$

$$= I(W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) + I(A_n^{[\mathcal{P}]}; \mathcal{F}|W_{1:K}, S, W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) \tag{2.29}$$

$$= I(W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) + I(A_n^{[\mathcal{P}]}; \mathcal{F}|W_{1:K}, S, Q_n^{[\mathcal{P}]}) \tag{2.30}$$

$$= I(W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) + H(A_n^{[\mathcal{P}]}|W_{1:K}, S, Q_n^{[\mathcal{P}]}) - H(A_n^{[\mathcal{P}]}|\mathcal{F}, W_{1:K}, S, Q_n^{[\mathcal{P}]}) \tag{2.31}$$

$$= I(W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) \tag{2.32}$$

$$\leq I(W_{1:K}, S; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) + I(W_{\mathcal{P}}; \mathcal{F}|Q_n^{[\mathcal{P}]}) \tag{2.33}$$

$$= I(W_{1:K}, W_{\mathcal{P}}, S; \mathcal{F}|Q_n^{[\mathcal{P}]}) \tag{2.34}$$

$$= I(W_{1:K}, S; \mathcal{F}|Q_n^{[\mathcal{P}]}) \tag{2.35}$$

$$\leq I(W_{1:K}, S; \mathcal{F}|Q_n^{[\mathcal{P}]}) + I(W_{1:K}, S; Q_n^{[\mathcal{P}]}) \tag{2.36}$$

$$= I(W_{1:K}, S; \mathcal{F}, Q_n^{[\mathcal{P}]}) \tag{2.37}$$

$$= 0 \tag{2.38}$$

where (2.32) follows from the fact that the answer strings are deterministic functions of the queries and the messages, and (2.38) follows from the independence of $(W_{1:K}, S, \mathcal{F})$ and (2.14). Since mutual information cannot be negative, it must be equal to zero, and

$$H(A_n^{[\mathcal{P}]}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) - H(A_n^{[\mathcal{P}]}|W_{\mathcal{P}}, \mathcal{F}, Q_n^{[\mathcal{P}]}) = I(A_n^{[\mathcal{P}]}; \mathcal{F}|W_{\mathcal{P}}, Q_n^{[\mathcal{P}]}) = 0 \tag{2.39}$$

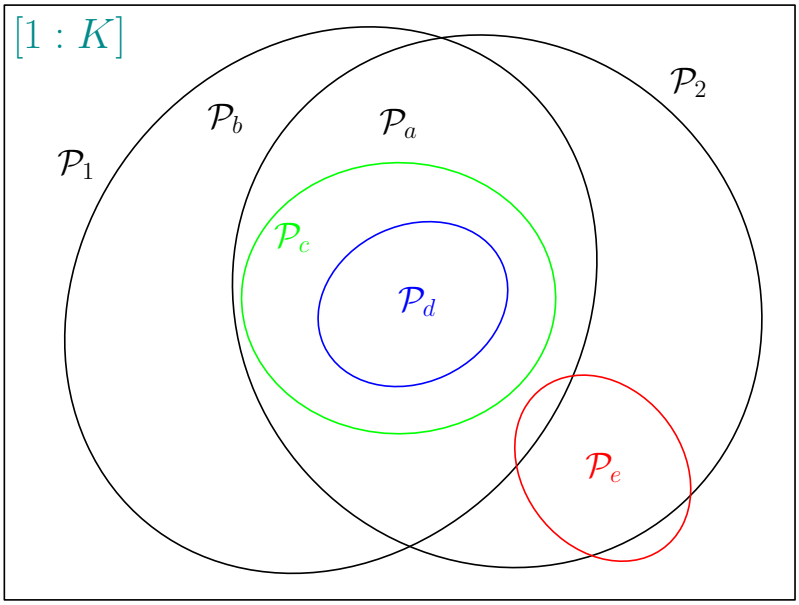


Figure 2.4: The relation of the index sets presented in Lemma 2.3 and used in Lemmas 2.4 and 2.5.

completing the proof. ■

Next, we need Lemma 2.3, which is an existence proof for index sets with specific properties. This technical lemma is needed in the proofs of upcoming two lemmas, Lemma 2.4 and Lemma 2.5. First, we give the definitions of relevant index sets $\mathcal{P}_a$, $\mathcal{P}_b$, $\mathcal{P}_c$, $\mathcal{P}_d$, and an element i_m. Given $\mathcal{P}_1$ and $\mathcal{P}_2$, we divide $\mathcal{P}_1$ into two disjoint partitions $\mathcal{P}_a$ and $\mathcal{P}_b$ (i.e., $\mathcal{P}_a \cup \mathcal{P}_b = \mathcal{P}_1$ and $\mathcal{P}_a \cap \mathcal{P}_b = \emptyset$), where $\mathcal{P}_a \subseteq \mathcal{P}_2$ (i.e., $\mathcal{P}_1 \cap \mathcal{P}_2 = \mathcal{P}_a$), $\mathcal{P}_b \subseteq \bar{\mathcal{P}}_2$. Suppose $|\mathcal{P}_a| = M \in [1 : P-1]$. Note that since $\mathcal{P}_1 \neq \mathcal{P}_2$, we cannot have $M = P$. We assume that $\mathcal{P}_a = \{i_1, \cdots, i_M\}$ for clarity of presentation. Given an arbitrary number $m \in [1 : M]$, we define a new index set $\mathcal{P}_c = \{i_1, \cdots, i_m\}$ which consists of exactly the first m elements in the index set $\mathcal{P}_a$. Let i_m be the last element from the index set $\mathcal{P}_c$. We obtain a new index set $\mathcal{P}_d = \{i_1, \cdots, i_{m-1}\}$ after removing this element. That means $\mathcal{P}_c = \mathcal{P}_d \cup \{i_m\}$. The relation of all these mentioned index sets is shown in Fig. 2.4.

Lemma 2.3 *For $K \geq 3$, $1 \leq P \leq K-1$, given index sets $\mathcal{P}_1$, $\mathcal{P}_2$ such that $|\mathcal{P}_i| = P$ for $i = 1, 2$ and $\mathcal{P}_1 \neq \mathcal{P}_2$, we can construct an index set $\mathcal{P}_3$ such that,*

i) $\mathcal{P}_3 \neq \mathcal{P}_1$ and $\mathcal{P}_3 \neq \mathcal{P}_2$,

ii) $|\mathcal{P}_3| = P$, and

iii) $\mathcal{P}_3$ includes $\mathcal{P}_b \cup \mathcal{P}_d$ but does not include the common element i_m in $\mathcal{P}_1 \cap \mathcal{P}_2$.

Proof: The key is to construct an index set $\mathcal{P}_e$ which satisfies the following two constraints: $\mathcal{P}_e \subseteq [1 : K] \backslash \{\mathcal{P}_b, \mathcal{P}_c\}$ and $|\mathcal{P}_e| = M - (m-1)$. As we can see, $|\mathcal{P}_a \backslash \mathcal{P}_c| = M - m$ and $|\mathcal{P}_2 \backslash \mathcal{P}_a| \geq 1$. One way to construct the index set $\mathcal{P}_e$ is to include all the $(M-m)$ elements from the index set $\mathcal{P}_a \backslash \mathcal{P}_c$ and one more element from the index set $\mathcal{P}_2 \backslash \mathcal{P}_a$, i.e.,

$$\mathcal{P}_e = (\mathcal{P}_a \backslash \mathcal{P}_c) \cup \{i_*\} \tag{2.40}$$

where $i_* \in \mathcal{P}_2 \backslash \mathcal{P}_a$. The index set $\mathcal{P}_e$ is generally not unique (for some examples, see Examples 1 and 2 below). Now, we are ready to construct the index set $\mathcal{P}_3$ as,

$$\mathcal{P}_3 = \mathcal{P}_b \cup \mathcal{P}_d \cup \mathcal{P}_e \tag{2.41}$$

Since $\mathcal{P}_b$, $\mathcal{P}_d$, $\mathcal{P}_e$ are disjoint sets, $|\mathcal{P}_3| = |\mathcal{P}_b| + |\mathcal{P}_d| + |\mathcal{P}_e| = (P - M) + (m-1) + (M - m + 1) = P$. Thus, we are able to construct $\mathcal{P}_3$ such that $|\mathcal{P}_3| = P$. Based on the formulation of $\mathcal{P}_b$, $\mathcal{P}_d$ and $\mathcal{P}_e$, these three index sets do not include the element i_m. Hence, $i_m \notin \mathcal{P}_3$. Since both $\mathcal{P}_1$ and $\mathcal{P}_2$ have the element i_m as i_m belongs to

their intersection $\mathcal{P}_a$, $\mathcal{P}_3$ is not the same as $\mathcal{P}_1$ or $\mathcal{P}_2$, i.e., $\mathcal{P}_3 \neq \mathcal{P}_1$, $\mathcal{P}_3 \neq \mathcal{P}_2$ and $|\mathcal{P}_3| = P$. ■

The following two examples illustrate the relations between the aforementioned sets, which will be important for the converse proof through the proofs of Lemmas 2.4 and 2.5.

Example 1: Suppose $K = 3$, $P = 2$ and $N \geq 2$ is an arbitrary positive integer. The total possible number of index sets is $\binom{K}{P} = 3$. Assume $\mathcal{P}_1 = \{1, 2\}$, $\mathcal{P}_2 = \{1, 3\}$ without loss of generality. Then, $\mathcal{P}_a = \{1\}$, $\mathcal{P}_b = \{2\}$ and the corresponding M is 1. Thus, m can only take the value 1. That means $\mathcal{P}_c = \{1\}$ and $\mathcal{P}_d$ has to be an empty set. For $\mathcal{P}_e$, we cannot take any element from the set $\mathcal{P}_a \backslash \mathcal{P}_c$ as it is empty, instead we can take the element 3 from the set $\mathcal{P}_2 \backslash \mathcal{P}_a$. Thus, $\mathcal{P}_e$ is formed as $\{3\}$, and we construct $\mathcal{P}_3 = \{2, 3\}$.

Example 2: Suppose $K = 6$, $P = 4$ and $N \geq 2$ is an arbitrary positive integer. The total possible number of index sets is $\binom{K}{P} = 15$. Assume $\mathcal{P}_1 = \{1, 3, 5, 6\}$, $\mathcal{P}_2 = \{2, 3, 5, 6\}$ without loss of generality. Then, $\mathcal{P}_a = \{3, 5, 6\}$, $\mathcal{P}_b = \{1\}$ and the corresponding M is 3. Thus, m can take the values 1, 2 or 3. To avoid being repetitive, we only consider the cases of $m = 2$ or $m = 3$, which are different from Example 1.

When $m = 2$, $\mathcal{P}_c = \{3, 5\}$ and $\mathcal{P}_d = \{3\}$. For $\mathcal{P}_e$, we can take the element 6 from the set $\mathcal{P}_a \backslash \mathcal{P}_c$ and then take the element 2 from the set $\mathcal{P}_2 \backslash \mathcal{P}_a$. Alternatively, we can pick the element 4 outside the union $\mathcal{P}_1 \cup \mathcal{P}_2$ instead of the element 6 from the set $\mathcal{P}_a \backslash \mathcal{P}_c$. Thus, $\mathcal{P}_e$ is formed as $\{2, 6\}$ (or $\{4, 6\}$). Therefore, we finally obtain

$\mathcal{P}_3 = \{1, 2, 3, 6\}$ (or $\{1, 3, 4, 6\}$).

When $m = 3$, $\mathcal{P}_c = \{3, 5, 6\}$ and $\mathcal{P}_d = \{3, 5\}$. For $\mathcal{P}_e$, we cannot take any element from the set $\mathcal{P}_a \backslash \mathcal{P}_c$ since it is empty. We take the element 2 from the set $\mathcal{P}_2 \backslash \mathcal{P}_a$ or take the element 4 outside the union $\mathcal{P}_1 \cup \mathcal{P}_2$. Thus, $\mathcal{P}_e$ is formed as $\{2\}$ (or $\{4\}$), and we construct $\mathcal{P}_3 = \{1, 2, 3, 5\}$ (or $\{1, 3, 4, 5\}$).

Next, we need the following lemma. Lemma 2.4 states that revealing any individual answer given the messages $(W_{\mathcal{P}_b}, W_{\mathcal{P}_d})$ does not leak any information about the message W_{i_m}.

Lemma 2.4 (Message leakage within any individual answer string) *When $1 \leq P \leq K - 1$ and $M \geq 1$, for arbitrary $m \in [1 : M]$, the following equality is always true,*

$$H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_2]}) \tag{2.42}$$

Remark 2.10 *The goal of Lemma 2.4 is to prove a key step, equation* (2.63)*, in the proof of Lemma 2.5. We remark that Lemma 2.4 is true for any $m \in [2 : M]$ when $M \geq 1$ as proved below. In the case when $m = 1$, the messages set $W_{i_1:i_{m-1}}$ (i.e., $\mathcal{P}_d$) is an empty set and thus Lemma 2.4 is still true in this case.*

Proof: From the user privacy constraint (2.18), we have,

$$H(W_{\mathcal{P}_b}, W_{\mathcal{P}_c}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{\mathcal{P}_b}, W_{\mathcal{P}_c}, A_n^{[\mathcal{P}_3]}, Q_n^{[\mathcal{P}_3]}) \tag{2.43}$$

$$H(W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, A_n^{[\mathcal{P}_3]}, Q_n^{[\mathcal{P}_3]}) \tag{2.44}$$

Since $\mathcal{P}_c = \mathcal{P}_d \cup i_m$, we have

$$H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, A_n^{[\mathcal{P}_3]}, Q_n^{[\mathcal{P}_3]}) \tag{2.45}$$

Similarly,

$$H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_2]}) = H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) \tag{2.46}$$

From the database privacy constraint (2.20), we have,

$$\begin{aligned}
0 &= I(W_{\bar{\mathcal{P}}_3}; A_{1:N}^{[\mathcal{P}_3]}, Q_{1:N}^{[\mathcal{P}_3]}, \mathcal{F}) && (2.47)\\
&= I(W_{\bar{\mathcal{P}}_3}; A_{1:N}^{[\mathcal{P}_3]}, W_{\mathcal{P}_3}, Q_{1:N}^{[\mathcal{P}_3]}, \mathcal{F}) && (2.48)\\
&\geq I(W_{\bar{\mathcal{P}}_3}; A_{1:N}^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_{1:N}^{[\mathcal{P}_3]}) && (2.49)\\
&\geq I(W_{i_m}; A_{1:N}^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_{1:N}^{[\mathcal{P}_3]}) && (2.50)\\
&\geq I(W_{i_m}; A_n^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) && (2.51)\\
&= I(W_{i_m}; A_n^{[\mathcal{P}_3]}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) && (2.52)\\
&= H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) - H(W_{i_m}|A_n^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) && (2.53)
\end{aligned}$$

where (2.48) comes from the MM-SPIR reliability constraint (2.17), (2.49) comes from the relationship $\mathcal{P}_3 = \mathcal{P}_b \cup \mathcal{P}_d \cup \mathcal{P}_e$ (i.e, $\mathcal{P}_b \cup \mathcal{P}_d \subseteq \mathcal{P}_3$), and (2.50) comes from the relationship $i_m \in \bar{\mathcal{P}}_3$. Thus, $H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) \leq H(W_{i_m}|A_n^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]})$. This concludes the proof by observing that $H(W_{i_m}|W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]}) \geq H(W_{i_m}|A_n^{[\mathcal{P}_3]}, W_{\mathcal{P}_b}, W_{\mathcal{P}_d}, Q_n^{[\mathcal{P}_3]})$ trivially as conditioning

cannot increase entropy. ■

Finally, the following lemma states that conditioning on an undesired message set does not decrease the uncertainty on any individual answer string. This is a consequence of the database privacy constraint.

Lemma 2.5 (Effect of conditioning on an undesired message set)

$$H(A_n^{[\mathcal{P}_2]}|W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_2]}) = H(A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}),\ \forall n,\ \forall \mathcal{P}_1, \mathcal{P}_2 \text{ s.t. } \mathcal{P}_1 \neq \mathcal{P}_2, |\mathcal{P}_1| = |\mathcal{P}_2| \quad (2.54)$$

Remark 2.11 *We note that although Lemma 2.5 has the same flavor as [8, eqn. (39)], the proof is much more involved. The main reason for this difficulty is the inter-relations between subsets of messages of size P. Specifically, in SM-SPIR, all message subsets are of size $P = 1$, and therefore, they are disjoint. However, in MM-SPIR, the message subsets are of size P, and they intersect in general, i.e., for a given $\mathcal{P}_1$, $\mathcal{P}_2$ such that $|\mathcal{P}_1| = |\mathcal{P}_2| = P$, the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ is not an empty set in general in contrast to SM-SPIR. Dealing with message subset intersections is the essence of introducing and proving Lemmas 2.3, 2.4 and 2.5.*

Proof: From the database privacy constraint (2.20), we have,

$$0 = I(W_{\bar{\mathcal{P}}_2}; A_{1:N}^{[\mathcal{P}_2]}, Q_{1:N}^{[\mathcal{P}_2]}, \mathcal{F}) \quad (2.55)$$

$$\geq I(W_{\bar{\mathcal{P}}_2}; A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \quad (2.56)$$

$$\geq I(W_{\mathcal{P}_b}; A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \quad (2.57)$$

$$= I(W_{\mathcal{P}_b}; A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) \quad (2.58)$$

$$= H(W_{\mathcal{P}_b}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_b}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.59}$$

where (2.57) comes from the relationship $\mathcal{P}_b \subseteq \bar{\mathcal{P}}_2$, (2.58) follows from the independence of messages and queries. Hence, $H(W_{\mathcal{P}_b}|Q_n^{[\mathcal{P}_2]}) = H(W_{\mathcal{P}_b}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$ as the reverse implication follows form the fact that conditioning cannot increase entropy.

Case 1: $M = 0$: In this case, there is no intersection between $\mathcal{P}_1$ and $\mathcal{P}_2$. $W_{\mathcal{P}_a}$ is an empty set of messages and then $W_{\mathcal{P}_1} = W_{\mathcal{P}_b}$. Hence,

$$I(W_{\mathcal{P}_1}; A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) = I(W_{\mathcal{P}_b}; A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) = 0 \tag{2.60}$$

where (2.60) follows from (2.58). This proves (2.54), the claim of lemma, when $M = 0$.

Case 2: $M \geq 1$: In this case, $W_{\mathcal{P}_1} = W_{\mathcal{P}_a} \cup W_{\mathcal{P}_b}$ and $W_{\mathcal{P}_a} = \{W_{i_1}, \cdots, W_{i_M}\}$.

$$H(W_{\mathcal{P}_a}|W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{i_1:i_M}|W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.61}$$

$$= H(W_{i_1}|W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) + H(W_{i_2}|W_{i_1}, W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$$

$$+ \cdots + H(W_{i_M}|W_{i_1:i_M-1}, W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.62}$$

$$= H(W_{i_1}|W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]}) + H(W_{i_2}|W_{i_1}, W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]})$$

$$+ \cdots + H(W_{i_M}|W_{i_1:i_M-1}, W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]}) \tag{2.63}$$

$$= H(W_{i_1:i_M}|W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]}) \tag{2.64}$$

$$= H(W_{\mathcal{P}_a}|W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]}) \tag{2.65}$$

where (2.63) comes from the direct application of Lemma 2.4.

Thus, we have,

$$I(W_{\mathcal{P}_1}; A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) = H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_1}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.66}$$

$$= H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_a}, W_{\mathcal{P}_b}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.67}$$

$$= H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_b}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$$

$$- H(W_{\mathcal{P}_a}|W_{\mathcal{P}_b}, A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) \tag{2.68}$$

$$= H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_b}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_a}|W_{\mathcal{P}_b}, Q_n^{[\mathcal{P}_2]}) \tag{2.69}$$

$$= H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_a}, W_{\mathcal{P}_b}|Q_n^{[\mathcal{P}_2]}) \tag{2.70}$$

$$= H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) - H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]}) \tag{2.71}$$

$$= 0 \tag{2.72}$$

where (2.69) follows from (2.59) and (2.65). This proves (2.54), the claim of lemma, when $M \geq 1$.

Combining (2.60) and (2.72) proves (2.54) in all cases completing the proof.

■

Remark 2.12 *The intuition behind Lemma 2.5 is as follows: If the pair* $(A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$ *provide any information about* $W_{\mathcal{P}_1}$*, they have to provide some information about* $W_{\bar{\mathcal{P}}_1}$ *under the user privacy constraint. However, database privacy constraint is thus obviously violated if the user receives any information about* $W_{\bar{\mathcal{P}}_1}$*. Consequently, the pair* $(A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]})$ *can never provide any information about* $W_{\bar{\mathcal{P}}_1}$*. Therefore, we are able to derive* $H(W_{\mathcal{P}_1}|A_n^{[\mathcal{P}_2]}, Q_n^{[\mathcal{P}_2]}) = H(W_{\mathcal{P}_1}) \overset{(2.15)}{=} H(W_{\mathcal{P}_1}|Q_n^{[\mathcal{P}_2]})$,

and hence $I(W_{\mathcal{P}_1}; A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) = 0$.

Now, we are ready to construct the main body of the converse proof for MM-SPIR, as well as the minimal entropy of common randomness required to achieve perfect MM-SPIR. Since we dealt with the inter-relations between message subsets in the previous lemmas and reached similar conclusions to those in SM-SPIR [8], the main body of the converse proof will be similar in structure to its counterpart in SM-SPIR.

The proof for $R \leq C_{MM-SPIR}$**:**

$$PL = H(W_{\mathcal{P}_1}) \tag{2.73}$$

$$= H(W_{\mathcal{P}_1}|\mathcal{F}) \tag{2.74}$$

$$= H(W_{\mathcal{P}_1}|\mathcal{F}) - H(W_{\mathcal{P}_1}|A_{1:N}^{[\mathcal{P}_1]}, \mathcal{F}) \tag{2.75}$$

$$= I(W_{\mathcal{P}_1}; A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) \tag{2.76}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_{1:N}^{[\mathcal{P}_1]}|W_{\mathcal{P}_1}, \mathcal{F}) \tag{2.77}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_{1:N}^{[\mathcal{P}_1]}|W_{\mathcal{P}_1}, \mathcal{F}, Q_n^{[\mathcal{P}_1]}) \tag{2.78}$$

$$\leq H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_1]}|W_{\mathcal{P}_1}, \mathcal{F}, Q_n^{[\mathcal{P}_1]}) \tag{2.79}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_1]}|W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_1]}) \tag{2.80}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_2]}|W_{\mathcal{P}_1}, Q_n^{[\mathcal{P}_2]}) \tag{2.81}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_2]}|Q_n^{[\mathcal{P}_2]}) \tag{2.82}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_1]}|Q_n^{[\mathcal{P}_1]}) \tag{2.83}$$

$$\leq H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_1]}|Q_n^{[\mathcal{P}_1]}, \mathcal{F}) \tag{2.84}$$

$$= H(A_{1:N}^{[\mathcal{P}_1]}|\mathcal{F}) - H(A_n^{[\mathcal{P}_1]}|\mathcal{F}) \tag{2.85}$$

where (2.74) follows from the independence of the user's private randomness and the messages, (2.75) follows from the MM-SPIR reliability constraint (2.17), (2.78) follows from the fact that the queries are deterministic functions of the user's private randomness $\mathcal{F}$ (2.14), (2.80) follows from Lemma 2.2, (2.81) follows from the first part of Lemma 2.1, (2.82) follows from Lemma 2.5, (2.83) follows from the second part Lemma 2.1, and (2.85) again follows from the fact that the queries are deterministic functions of the user's private randomness $\mathcal{F}$ (2.14).

By summing (2.85) up for all $n \in [1 : N]$ and letting $\mathcal{P}$ denote the general desired index set, we obtain,

$$NPL \leq NH(A_{1:N}^{[\mathcal{P}]}|\mathcal{F}) - \sum_{n=1}^{N} H(A_n^{[\mathcal{P}]}|\mathcal{F}) \tag{2.86}$$

$$\leq NH(A_{1:N}^{[\mathcal{P}]}|\mathcal{F}) - H(A_{1:N}^{[\mathcal{P}]}|\mathcal{F}) \tag{2.87}$$

$$= (N-1)H(A_{1:N}^{[\mathcal{P}]}|\mathcal{F}) \tag{2.88}$$

$$\leq (N-1)\sum_{n=1}^{N} H(A_n^{[\mathcal{P}]}|\mathcal{F}) \tag{2.89}$$

$$\leq (N-1)\sum_{n=1}^{N} H(A_n^{[\mathcal{P}]}) \tag{2.90}$$

which leads to the desired converse result on the retrieval rate,

$$R_{MM-SPIR} = \frac{PL}{D_{MM-SPIR}} \leq \frac{PL}{\sum_{n=1}^{N} H(A_n^{[\mathcal{P}]})} \leq \frac{N-1}{N} = 1 - \frac{1}{N} \tag{2.91}$$

The proof for $H(S) \geq \frac{PL}{N-1}$:

$$0 = I(W_{\bar{\mathcal{P}}_1}; A_{1:N}^{[\mathcal{P}_1]}, Q_{1:N}^{[\mathcal{P}_1]}, \mathcal{F}) \tag{2.92}$$

$$\geq I(W_{\bar{\mathcal{P}}_1}; A_{1:N}^{[\mathcal{P}_1]}, \mathcal{F}) \tag{2.93}$$

$$= I(W_{\bar{\mathcal{P}}_1}; A_{1:N}^{[\mathcal{P}_1]}, W_{\mathcal{P}_1}, \mathcal{F}) \tag{2.94}$$

$$= I(W_{\bar{\mathcal{P}}_1}; A_{1:N}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) \tag{2.95}$$

$$\geq I(W_{\bar{\mathcal{P}}_1}; A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) \tag{2.96}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - H(A_{n}^{[\mathcal{P}_1]} | W_{1:K}, \mathcal{F}) \tag{2.97}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - H(A_{n}^{[\mathcal{P}_1]} | W_{1:K}, \mathcal{F}) + H(A_{n}^{[\mathcal{P}_1]} | W_{1:K}, \mathcal{F}, S) \tag{2.98}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - I(S; A_{n}^{[\mathcal{P}_1]} | W_{1:K}, \mathcal{F}) \tag{2.99}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - H(S | W_{1:K}, \mathcal{F}) + H(S | A_{n}^{[\mathcal{P}_1]}, W_{1:K}, \mathcal{F}) \tag{2.100}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - H(S) + H(S | A_{n}^{[\mathcal{P}_1]}, W_{1:K}, \mathcal{F}) \tag{2.101}$$

$$\geq H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}) - H(S) \tag{2.102}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | W_{\mathcal{P}_1}, \mathcal{F}, Q_{n}^{[\mathcal{P}_1]}) - H(S) \tag{2.103}$$

$$= H(A_{n}^{[\mathcal{P}_1]} | Q_{n}^{[\mathcal{P}_1]}) - H(S) \tag{2.104}$$

where (2.92) follows from the database privacy constraint (2.20), (2.94) follows from the MM-SPIR reliability constraint (2.17), (2.98) follows from the fact that the answer strings are deterministic functions of messages and queries which are also functions of the randomness $\mathcal{F}$ as in (2.14) and (2.16), (2.101) follows from the independence of the common randomness, messages, and user's private randomness as in (2.13), (2.103) follows from (2.14), and (2.104) follows from the steps between

(2.80)-(2.83) by applying Lemma 2.1, 2.2 and 2.5 again.

By summing (2.104) up for all $n \in [1:N]$ and letting $\mathcal{P}$ denote the general desired index set again, we obtain,

$$0 \geq \sum_{n=1}^{N} H(A_n^{[\mathcal{P}]}|Q_n^{[\mathcal{P}]}) - NH(S) \tag{2.105}$$

$$\geq H(A_{1:N}^{[\mathcal{P}]}|Q_n^{[\mathcal{P}]}) - NH(S) \tag{2.106}$$

$$\geq H(A_{1:N}^{[\mathcal{P}]}|Q_n^{[\mathcal{P}]}, \mathcal{F}) - NH(S) \tag{2.107}$$

$$= H(A_{1:N}^{[\mathcal{P}]}|\mathcal{F}) - NH(S) \tag{2.108}$$

$$\geq \frac{N}{N-1} PL - NH(S) \tag{2.109}$$

where (2.108) follows from (2.14) and (2.109) follows from (2.88), which leads to a lower bound for the minimal required entropy of common randomness S,

$$H(S) \geq \frac{PL}{N-1} \tag{2.110}$$

2.5.4 MM-SPIR: Achievability Proof

Since the MM-SPIR capacity is the same as the SM-SPIR capacity, and the required common randomness is P times the required common randomness for SM-SPIR, we can use the achievable scheme in [8] successively P times in a row (by utilizing independent common randomness each time) to achieve the MM-SPIR capacity. Although the query structure for the capacity-achieving scheme for SPIR in [8] is quite simple, it is fundamentally different than the query structure for the capacity-

achieving scheme for PIR in [6]. This means that user/databases should execute different query structures for different database privacy levels. In this chapter, by combining ideas for achievability from [23] and [15], we propose an alternative capacity-achieving scheme for MM-SPIR for any[11] P. Our achievability scheme enables us to switch between MM-PIR and MM-SPIR seamlessly, and therefore support different database privacy levels, as the basic query structures are similar[12]. We start with two motivating examples in Section 2.5.4.1, give the general achievable scheme in Section 2.5.4.2, and calculate its rate and required common randomness amount in Section 2.5.4.3.

For convenience, we use the *k-sum* notation in [6, 23]. A k-sum is a sum of k symbols from k different messages. Thus, a k-sum symbol appears only in round k. We denote the number of stages in round k by α_k, which was originally introduced in [23]. In addition, we use ν to denote the number of repetitions of the scheme[13] in [23] we need before we start assigning common randomness symbols.

2.5.4.1 Motivating Examples

Example 3: Consider the case $K = 3$, $P = 1$, $N = 3$. Our achievable scheme is as follows: First, we generate an initial query table, which strictly follows the query

[11] We note that the capacity-achieving scheme for $K = P$ is simply to download all messages from one of the databases, hence, without loss of generality, we focus on the case $1 \leq P \leq K - 1$ in this section.

[12] We note that the presented scheme in this section can be thought of as a stand-alone capacity-achieving scheme for the MM-SPIR problem when the message lengths are unconstrained. Consequently, our proposed scheme in Section 2.5.4 cannot be applied to the PSI problem, as it requires the message size to be constrained to $L = 1$.

[13] When we refer to the scheme in [23], we refer to the near-optimal scheme in [23] which was introduced for $K/P \geq 2$. Reference [23] has a different, optimal, scheme for $K/P \leq 2$. However, in this chapter, even when $K/P \leq 2$, we still refer to (and use) the near-optimal scheme in [23].

table generation in [23]. For this case, from [23], we obtain the number of stages needed in each round as $\alpha_1 = 1, \alpha_2 = 2, \alpha_3 = 4$. From the perspective of a database, before the assignment of common randomness symbols begins, the total number of downloaded desired symbols in round 1 is $\alpha_1 P = 1 \times 1 = 1$. Thus, we need 1 previously downloaded common randomness symbol for this desired symbol. Since this common randomness symbol needs to come from the other $N-1 = 2$ databases, the required common randomness to be downloaded from each database is $\frac{1}{2}$ symbols (assuming a symmetric scheme that distributes downloads equally over the other 2 databases). Thus, in order to obtain an integer number of common randomness symbols to be downloaded from each database, we repeat the scheme in [23] two times (i.e., $\nu = 2$) before we begin assigning the common randomness symbols. Hence, the number of stages in each round become $\nu\alpha_k = 2\alpha_k$, for $k = 1, 2, 3$. That is we have 2 stages of 1-sums, 4 stages of 2-sums and 8 stages of 3-sums; see Table 2.1.

We are now ready to start assigning the common randomness symbols. We first download 1 common randomness symbol from each database; for instance, we download s_1 from database 1. In round 1, we mix (i.e., add) a common randomness symbol to each 1-sum. All the common randomness symbols at each database should be distinct; for instance, observe that, we add $s_2, s_3, s_4, s_5, s_6, s_7$ at database 1. Second, the common randomness symbols added to the desired symbols (a symbols in this example) must be downloaded from other databases; for instance, note that s_2 and s_3 added to symbols a_1 and a_2 are downloaded from databases 2 and 3. Note that the indices of the common randomness symbols added to the undesired symbols

Database 1	Database 2	Database 3
s_1	s_2	s_3
$a_1 + s_2$	$a_3 + s_1$	$a_5 + s_1$
$a_2 + s_3$	$a_4 + s_3$	$a_6 + s_2$
$b_1 + s_4$	$b_3 + s_8$	$b_5 + s_{12}$
$b_2 + s_5$	$b_4 + s_9$	$b_6 + s_{13}$
$c_1 + s_6$	$c_3 + s_{10}$	$c_5 + s_{14}$
$c_2 + s_7$	$c_4 + s_{11}$	$c_6 + s_{15}$
$a_7 + b_3 + s_8$	$a_{15} + b_1 + s_4$	$a_{23} + b_1 + s_4$
$a_8 + b_4 + s_9$	$a_{16} + b_2 + s_5$	$a_{24} + b_2 + s_5$
$a_9 + b_5 + s_{12}$	$a_{17} + b_5 + s_{12}$	$a_{25} + b_3 + s_8$
$a_{10} + b_6 + s_{13}$	$a_{18} + b_6 + s_{13}$	$a_{26} + b_4 + s_9$
$a_{11} + c_3 + s_{10}$	$a_{19} + c_1 + s_6$	$a_{27} + c_1 + s_6$
$a_{12} + c_4 + s_{11}$	$a_{20} + c_2 + s_7$	$a_{28} + c_2 + s_7$
$a_{13} + c_5 + s_{14}$	$a_{21} + c_5 + s_{14}$	$a_{29} + c_3 + s_{10}$
$a_{14} + c_6 + s_{15}$	$a_{22} + c_6 + s_{15}$	$a_{30} + c_4 + s_{11}$
$b_7 + c_7 + s_{16}$	$b_{11} + c_{11} + s_{20}$	$b_{15} + c_{15} + s_{24}$
$b_8 + c_8 + s_{17}$	$b_{12} + c_{12} + s_{21}$	$b_{16} + c_{16} + s_{25}$
$b_9 + c_9 + s_{18}$	$b_{13} + c_{13} + s_{22}$	$b_{17} + c_{17} + s_{26}$
$b_{10} + c_{10} + s_{19}$	$b_{14} + c_{14} + s_{23}$	$b_{18} + c_{18} + s_{27}$
$a_{31} + b_{11} + c_{11} + s_{20}$	$a_{39} + b_7 + c_7 + s_{16}$	$a_{47} + b_7 + c_7 + s_{16}$
$a_{32} + b_{12} + c_{12} + s_{21}$	$a_{40} + b_8 + c_8 + s_{17}$	$a_{48} + b_8 + c_8 + s_{17}$
$a_{33} + b_{13} + c_{13} + s_{22}$	$a_{41} + b_9 + c_9 + s_{18}$	$a_{49} + b_9 + c_9 + s_{18}$
$a_{34} + b_{13} + c_{14} + s_{23}$	$a_{42} + b_{10} + c_{10} + s_{19}$	$a_{50} + b_{10} + c_{10} + s_{19}$
$a_{35} + b_{15} + c_{15} + s_{24}$	$a_{43} + b_{15} + c_{15} + s_{24}$	$a_{51} + b_{11} + c_{11} + s_{20}$
$a_{36} + b_{16} + c_{16} + s_{25}$	$a_{44} + b_{16} + c_{16} + s_{25}$	$a_{52} + b_{12} + c_{12} + s_{21}$
$a_{37} + b_{17} + c_{17} + s_{26}$	$a_{45} + b_{17} + c_{17} + s_{26}$	$a_{53} + b_{13} + c_{13} + s_{22}$
$a_{38} + b_{18} + c_{18} + s_{27}$	$a_{46} + b_{18} + c_{18} + s_{27}$	$a_{54} + b_{14} + c_{14} + s_{23}$

Table 2.1: The query table for the case $K = 3, P = 1, N = 3$.

(symbols b and c) increase cumulatively, e.g., s_4, s_5, s_6, s_7 at database 1 in round 1, and these symbols are not separately downloaded by the user.

In round 2, for every 2-sum containing a desired message symbol, we add a side information symbol downloaded from another database which already contains a common randomness symbol; for instance, we add b_3+s_8 that is already downloaded from database 2, to the desired symbol a_7 at database 1, i.e., we download $a_7+b_3+s_8$. On the other hand, for every 2-sum not containing any desired message symbols,

we add a new distinct common randomness symbol with a cumulatively increasing index; for instance, for the download $b_7 + c_7$ from database 1, we add s_{16} which is a new non-downloaded common randomness symbol, and download $b_7 + c_7 + s_{16}$. Finally, in round 3, where we download 3-sums, and hence every download contains a desired symbol, we add the side information generated at other databases; for instance, we add $b_{11} + c_{11} + s_{20}$ downloaded from database 2, to a_{31} and download $a_{31}+b_{11}+c_{11}+s_{20}$. This completes the achievable scheme for this case. The complete query table is shown in Table 2.1.

Now, we calculate the rate of this scheme. The length of each message is $L = 54$, and the total number of downloads is $D = 81$. Thus, the rate R of this scheme is $\frac{54}{81} = \frac{2}{3} = 1 - \frac{1}{3}$, which matches the capacity expression. In addition, we used 27 common randomness symbols, hence the required common randomness $H(S)$ is $27 = \frac{54}{2}$, which matches the required minimum.

Example 4: Consider the case $K = 5, P = 3, N = 2$. Our achievable scheme is as follows: Again, first, we generate an initial query table, which strictly follows the query table generation in [23]. Note that, we still use the near-optimal scheme in [23], even though for this case $K/P \leq 2$ (see Footnote 13). For this case, from [23], we obtain the number of stages needed in each round as $\alpha_1 = 3$, $\alpha_2 = 1$, $\alpha_3 = \alpha_4 = 0$ and $\alpha_5 = 1$. In this case, from the perspective of a database, before the assignment of common randomness symbols begins, the total number of downloaded desired symbols in round 1 is $\alpha_1 P = 3 \times 3 = 9$. Thus, we need 9 previously downloaded common randomness symbols for these desired symbols. These common randomness

symbols need to come from the other $N-1=1$ database. In this case, since $9/1=9$ is an integer already, we do not need to repeat the scheme unlike the case in Example 3. Thus, $\nu=1$ here, there is no need for repetition, and the underlying query structure before adding common randomness symbols is exactly the same as [23]; see Table 2.2.

We are now ready to start assigning the common randomness symbols. We first download 9 common randomness symbols from each database; for instance, we download $s_1,\cdots,s_9$ from database 1. In round 1, we add a common randomness symbol to each 1-sum. All the common randomness symbols at each database should be distinct; for instance, observe that, we add $s_{10},\cdots,s_{24}$ at database 1. Second, the common randomness symbols added to the desired symbols (a, b, c symbols in this example) must be downloaded from the other databases; for instance, note that $s_{10},\cdots,s_{18}$ added to symbols $a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3$ are downloaded from database 2. Note that the indices of the common randomness symbols added to the undesired symbols (symbols d and e) increase cumulatively, e.g., $s_{19}\cdots,s_{24}$ at database 1 in round 1, and these symbols are not separately downloaded by the user.

In round 2, for every 2-sum containing only one desired message symbol, we add a side information symbol downloaded from the other database which already contains a common randomness symbol; for instance, we add d_4+s_{25} that is already downloaded from database 2, to the desired bit a_8 at database 1, i.e., we download $a_8+d_4+s_{25}$. On the other hand, for every 2-sum containing two of the desired message symbols, we add a new distinct common randomness symbol and download

Database 1	Database 2
s_1, s_2, s_3	s_{10}, s_{11}, s_{12}
s_4, s_5, s_6	s_{13}, s_{14}, s_{15}
s_7, s_8, s_9	s_{16}, s_{17}, s_{18}
s_{31}, s_{32}, s_{33}	s_{34}, s_{35}, s_{36}
$a_1 + s_{10}$	$a_4 + s_1$
$b_1 + s_{11}$	$b_4 + s_2$
$c_1 + s_{12}$	$c_4 + s_3$
$d_1 + s_{19}$	$d_4 + s_{25}$
$e_1 + s_{20}$	$e_4 + s_{26}$
$a_2 + s_{13}$	$a_5 + s_4$
$b_2 + s_{14}$	$b_5 + s_5$
$c_2 + s_{15}$	$c_5 + s_6$
$d_2 + s_{21}$	$d_5 + s_{27}$
$e_2 + s_{22}$	$e_5 + s_{28}$
$a_3 + s_{16}$	$a_6 + s_7$
$b_3 + s_{17}$	$b_6 + s_8$
$c_3 + s_{18}$	$c_6 + s_9$
$d_3 + s_{23}$	$d_6 + s_{29}$
$e_3 + s_{24}$	$e_6 + s_{30}$
$a_7 + b_4 + s_{34}$	$a_{10} + b_1 + s_{31}$
$a_4 + c_7 + s_{35}$	$a_1 + c_{10} + s_{32}$
$a_8 + d_4 + s_{25}$	$a_{11} + d_1 + s_{19}$
$a_9 + e_4 + s_{26}$	$a_{12} + e_1 + s_{20}$
$b_7 + c_4 + s_{36}$	$b_{10} + c_1 + s_{33}$
$b_8 + d_5 + s_{27}$	$b_{11} + d_2 + s_{21}$
$b_9 + e_5 + s_{28}$	$b_{12} + e_2 + s_{22}$
$c_8 + d_6 + s_{29}$	$c_{11} + d_3 + s_{23}$
$c_9 + e_6 + s_{30}$	$c_{12} + e_3 + s_{24}$
$d_7 + e_7 + s_{37}$	$d_8 + e_8 + s_{38}$
$a_{13} + b_5 + c_5 + d_8 + e_8 + s_{38}$	$a_2 + b_{13} + c_2 + d_7 + e_7 + s_{37}$

Table 2.2: The query table for the case $K = 5, P = 3, N = 2$.

it separately from the other database; for instance, for the download $a_7 + b_4$ from database 1, we add s_{34} and download s_{34} separately from database 2, and download $a_7 + b_4 + s_{34}$. Therefore, for this, we need to download 3 common randomness symbols (s_{34}, s_{35}, s_{36}) from database 2. Further, for every 2-sum not containing any desired message symbols, we add a new distinct common randomness symbol with a

cumulatively increasing index; for instance, for the download $d_7 + e_7$ from database 1, we add s_{37} which is a new non-downloaded common randomness symbol, and download $b_7 + c_7 + s_{37}$. We skip rounds 3 and 4 because $\alpha_3 = \alpha_4 = 0$. Finally, in round 5, where we download 5-sums, we add the side information generated at the other databases; for instance, we add $d_8 + e_8 + s_{38}$ downloaded from database 2, to $a_{13} + b_5 + c_5$ and download $a_{13} + b_5 + c_5 + d_8 + e_8 + s_{38}$. This completes the achievable scheme for this case. The complete query table is shown in Table 2.2.

Now, we calculate the rate of this scheme. We downloaded 13 a symbols, 13 b symbols and 12 c symbols, hence a total of $L = 38$ desired symbols. The total number of downloads is $D = 76$. Thus, the rate R of this scheme is $\frac{38}{76} = \frac{1}{2} = 1 - \frac{1}{2}$, which matches the capacity expression. In addition, we used 38 common randomness symbols, hence the required common randomness $H(S)$ is $38 = \frac{38}{1}$, which matches the required minimum.

We finally note that, since we downloaded asymmetric number of symbols from desired messages, i.e., 13 a symbols, 13 b symbols and 12 c symbols, we can repeat this scheme 3 times changing the roles of a, b and c, and have a symmetric scheme where we download 38 a symbols, 38 b symbols and 38 c symbols. This will not change the normalized download cost and normalized downloaded common randomness symbol numbers, hence, all the calculations (rate and common randomness calculations) will remain the same.

2.5.4.2 General Achievable Scheme

Our achievability scheme is primarily based on the one in [23], with the addition of downloading and/or mixing common randomness variables into symbol downloads appropriately. We note that, here we extend the *near-optimal* algorithm in [23], which was originally proposed for $P \leq \frac{K}{2}$, to the case of $P \geq \frac{K}{2}$, and therefore, use it for all $1 \leq P \leq K-1$ (see Footnote 13). Our achievability scheme comprises the following steps:

1. *Initial MM-PIR Query Generation:* Generate an initial query table strictly following the near-optimal procedure in [23] for arbitrary K, P and N.

2. *Repetition:* Repeat Step 1 for a total of ν times. The purpose of the repetition is to *i)* get an integer number of common randomness generated at each database by a symmetric algorithm (as exemplified in Example 3), and *ii)* get equal number of symbols downloaded from each desired message (as exemplified in Example 4). Let ν_0 be the smallest integer such that $\frac{(N-1)^{K-P}N\nu_0}{P}$ (i.e., $\frac{\alpha_K N\nu_0}{P}$) is an integer. Similarly, for $1 \leq k \leq \min\{P, K-P\}$, let ν_k be the smallest integer such that $\frac{\binom{P}{k}\alpha_k\nu_k}{N-1}$ is an integer ($k \leq K-P$ comes from $\alpha_{K-P+1} = \cdots = \alpha_{P-1} = 0$ in [23, eqn. (51)]). Then, choose ν as the lowest common multiple of these ν_k, where $k \in [0 : \min\{P, K-P\}]$.

3. *Common Randomness Assignment:* Assign the common randomness as follows:

   (a) In round 1, assign $\frac{\nu P\alpha_1}{N-1}$ independent common randomness symbols to

each database, and download them. At each database, mix every 1-sum symbol containing a desired message symbol with an arbitrary common randomness symbol already downloaded from another database, making sure that every 1-sum symbol at each database is mixed with a different common randomness symbol. Mix all other 1-sum symbols not containing a desired symbol with a new common randomness symbol which is not downloaded by the user.

(b) In round k ($k \geq 2$), assign $\frac{\nu\binom{P}{k}\alpha_k}{N-1}$ independent common randomness symbols to each database, and download them. At each database: Mix every k-sum symbol containing only desired message symbols with an arbitrary common randomness symbol already downloaded from another database. Mix every k-sum symbol containing p desired message symbols ($1 \leq p \leq k-1$) with the common randomness symbol from the $(k-p)$-sum symbol having the same $k-p$ undesired message symbols downloaded at any other database. Mix every k-sum symbol not containing any desired message symbols with a new common randomness symbol which is not downloaded by the user.

(c) Repeat Step 3b until k reaches K. Note that if $\alpha_k = 0$, nothing is done.

This scheme inherits the user privacy property from the underlying scheme in [23], as the new common randomness symbols, which are separately downloaded and subtracted out, make no difference. Due to the procedure in Step 3, where non-downloaded common randomness symbols are added to the downloads, no undesired

symbol is decodable because of the added unknown common randomness, ensuring the database privacy constraint.

2.5.4.3 Rate and Common Randomness Amount Calculation

We calculate the achievable rate and the minimal required common randomness for only one repetition of the scheme. The reason for this is that, in every repetition, every involved term would be multiplied by T, and thus T can be cancelled in the numerator and the denominator of the normalized rate and normalized required common randomness expressions.

For each database, before the assignment of common randomness, let D_1 be the total number of downloaded symbols, U_1 be the total number of downloaded undesired symbols, U_2 be the total number of downloaded symbols including only desired message symbols, and D_2 be the total number of downloaded common randomness symbols. The achievable rate is then given by,

$$R = \frac{D_1 - U_1}{D_1 + D_2} \tag{2.111}$$

Using the respective results in [23, eqns. (66)-(69) and (70)-(72)], we have

$$D_1 = \sum_{k=1}^{K} \binom{K}{k} \alpha_k = \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[\left(1 + \frac{1}{r_i} \right)^K - 1 \right] \tag{2.112}$$

$$U_1 = \sum_{k=1}^{K-P} \binom{K-P}{k} \alpha_k = \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[\left(1 + \frac{1}{r_i} \right)^{K-P} - 1 \right] \tag{2.113}$$

In the proposed new achievable scheme, every k-sum symbol ($1 \leq k \leq \min\{P, K -$

$P\}$) containing only desired message symbols is mixed with an arbitrary common randomness symbol which is downloaded from another database. In addition, these downloaded common randomness symbols are uniformly requested from the other $(N-1)$ databases. Thus,

$$U_2 = \sum_{k=1}^{\min\{K-P,P\}} \binom{P}{k} \alpha_k \tag{2.114}$$

$$D_2 = \frac{1}{N-1} U_2 = \frac{1}{N-1} \sum_{k=1}^{\min\{K-P,P\}} \binom{P}{k} \alpha_k \tag{2.115}$$

With these observations we have the following two lemmas where we compute the MM-SPIR rate and the required common randomness amount.

Lemma 2.6 *The rate of the proposed achievable scheme is,*

$$R = 1 - \frac{1}{N} \tag{2.116}$$

Proof: We first calculate D_2 in two possible settings. When $P \leq \frac{K}{2}$, i.e., $P \leq K-P$,

$$D_2 = \frac{1}{N-1} \sum_{k=1}^{P} \binom{P}{k} \alpha_k \tag{2.117}$$

$$= \frac{1}{N-1} \sum_{k=1}^{P} \binom{P}{k} \sum_{i=1}^{P} \gamma_i r_i^{K-P-k} \tag{2.118}$$

$$= \frac{1}{N-1} \sum_{k=1}^{P} \sum_{i=1}^{P} \binom{P}{k} \gamma_i r_i^{K-P-k} \tag{2.119}$$

$$= \frac{1}{N-1} \sum_{i=1}^{P} \sum_{k=1}^{P} \binom{P}{k} \gamma_i r_i^{K-P-k} \tag{2.120}$$

$$= \frac{1}{N-1} \sum_{i=1}^{P} \gamma_i r_i^{K-2P} \sum_{k=1}^{P} \binom{P}{k} r_i^{P-k} \tag{2.121}$$

$$= \frac{1}{N-1} \sum_{i=1}^{P} \gamma_i r_i^{K-2P} (N-1) r_i^{P} \tag{2.122}$$

$$= \frac{1}{N-1} \sum_{i=1}^{P} \gamma_i r_i^{K-P} (N-1) \tag{2.123}$$

where (2.122) follows because r_i is a root of the characteristic equation [23, eqn. (59)].

When $\frac{K}{2} \leq P \leq K-1$, i.e., $K-P \leq P$,

$$D_2 = \frac{1}{N-1} \sum_{k=1}^{K-P} \binom{P}{k} \alpha_k \tag{2.124}$$

$$= \frac{1}{N-1} \sum_{k=1}^{P} \binom{P}{k} \alpha_k - \sum_{k=K-P+1}^{P} \binom{P}{k} \alpha_k \tag{2.125}$$

$$= \frac{1}{N-1} \sum_{k=1}^{P} \binom{P}{k} \alpha_k \tag{2.126}$$

$$= \frac{1}{N-1} \sum_{i=1}^{P} \gamma_i r_i^{K-P} (N-1) \tag{2.127}$$

where (2.126) follows because $\alpha_{K-P+1} = \cdots = \alpha_{P-1} = 0$ due to [23, eqn. (51)], and (2.127) follows from (2.123).

Therefore, from (2.123) and (2.127), for all P, where $1 \leq P \leq K-1$, we always have

$$D_2 = \frac{1}{N-1} \sum_{k=1}^{P} \binom{P}{k} \alpha_k = \frac{1}{N-1} \sum_{i=1}^{P} \gamma_i r_i^{K-P} (N-1) \tag{2.128}$$

Now, in order to show that $R = \frac{D_1 - U_1}{D_1 + D_2} = 1 - \frac{1}{N}$, we need to equivalently show

that $D_1 = NU_1 + (N-1)D_2$. Thus, we proceed as,

$$NU_1 + (N-1)D_2 = N\sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[\left(1 + \frac{1}{r_i}\right)^{K-P} - 1 \right] + \sum_{i=1}^{P} \gamma_i r_i^{K-P}(N-1) \tag{2.129}$$

$$= \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[N\left(1 + \frac{1}{r_i}\right)^{K-P} - N + N - 1 \right] \tag{2.130}$$

$$= \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[N\left(1 + \frac{1}{r_i}\right)^{K-P} - 1 \right] \tag{2.131}$$

$$= \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[N\left(1 + \frac{1}{r_i}\right)^{-P} \left(1 + \frac{1}{r_i}\right)^{K} - 1 \right] \tag{2.132}$$

$$= \sum_{i=1}^{P} \gamma_i r_i^{K-P} \left[\left(1 + \frac{1}{r_i}\right)^{K} - 1 \right] \tag{2.133}$$

$$= D_1 \tag{2.134}$$

where (2.133) follows because $N(1 + \frac{1}{r_i})^{-P} = 1$, which comes from [23, eqn. (62)].

■

Lemma 2.7 *The minimal required common randomness in the proposed achievable scheme is,*

$$H(S) = \frac{PL}{N-1} \tag{2.135}$$

Proof: In our proposed scheme, at each database, a new common randomness symbol is employed only in two cases. The first case is when a new common randomness symbol is added to a k-sum symbol that contains only desired message symbols. In this case, the common randomness symbols are equally distributed over the $(N-1)$

databases and downloaded from them. The second case is when a new common randomness symbol is assigned to a k-sum symbol that does not contain any desired message symbol. In this case, the common randomness symbols are not downloaded. Therefore, we count the total number of distinct common randomness symbols as $H(S) = U_1 + D_2$. We note that L can be written as $\frac{1}{P}(D_1 - U_1)$. Thus,

$$\frac{PL}{N-1} = \frac{\frac{P}{P}(D_1 - U_1)}{N-1} \tag{2.136}$$

$$= \frac{D_1 - U_1}{N-1} \tag{2.137}$$

$$= \frac{NU_1 + (N-1)D_2 - U_1}{N-1} \tag{2.138}$$

$$= \frac{(N-1)U_1 + (N-1)D_2}{N-1} \tag{2.139}$$

$$= U_1 + D_2 \tag{2.140}$$

$$= H(S) \tag{2.141}$$

where (2.138) comes from (2.134), i.e., $D_1 = NU_1 + (N-1)D_2$. ■

2.6 MM-LSPIR: Arbitrary Message Lengths

Since the message sizes in the PSI problem are given and fixed, in particular, they are fixed to be 1 (as the incidence vectors are composed 0s and 1s), we need to determine the capacity of MM-SPIR with a given and fixed message size L. We call this setting MM-LSPIR. The capacity of MM-LSPIR is given in the next theorem.

Theorem 2.3 *The MM-LSPIR capacity for $N \geq 2$, $K \geq 2$, and $P \leq K$, for an*

arbitrary message length L is given by,

$$C_{MM-LSPIR} = \begin{cases} 1, & P = K \\ \frac{PL}{\lceil \frac{NPL}{N-1} \rceil}, & 1 \leq P \leq K-1,\ H(S) \geq \lceil \frac{PL}{N-1} \rceil \\ 0, & otherwise \end{cases} \tag{2.142}$$

We give the converse of Theorem 2.3 in Section 2.6.1, the achievability in Section 2.6.2, and map MM-LSPIR back to PSI in Section 2.6.3.

2.6.1 MM-LSPIR: Converse Proof

From the converse proof of Theorem 2.2, using (2.21) and (2.91), we have

$$R_{MM-LSPIR} = \frac{PL}{D_{MM-LSPIR}} \leq \frac{PL}{\sum_{n=1}^{N} H(A_n^{[\mathcal{P}]})} \leq \frac{N-1}{N} = 1 - \frac{1}{N} \tag{2.143}$$

Note that, for an arbitrary finite fixed message length L, the download cost $D_{MM-LSPIR}$ must be a positive integer. Thus, we have,

$$D_{MM-LSPIR} \geq \left\lceil \frac{NPL}{N-1} \right\rceil \tag{2.144}$$

and therefore, the converse result for a finite and fixed L, is

$$R_{MM-LSPIR} = \frac{PL}{D_{MM-LSPIR}} \leq \frac{PL}{\lceil \frac{NPL}{N-1} \rceil} \tag{2.145}$$

Similarly, the entropy of common randomness must also be a positive integer,

as the common randomness symbols are picked uniformly and independently from the same field as the message symbols. Thus, with a careful look at going from (2.109) to (2.110), we have,

$$H(S) \geq \left\lceil \frac{PL}{N-1} \right\rceil \tag{2.146}$$

Therefore, (2.145) and (2.146) constitute the converse for Theorem 2.3.

2.6.2 MM-LSPIR: Achievability Proof

Following the converse results in (2.144) and (2.146), we provide an achievable scheme for MM-SPIR with any arbitrary parameters K, N, P, L in this section. Starting with the achievable scheme presented in [8, Section IV.B.1], we set the value of l_K to be 1 and build a corresponding SPIR achievable scheme for the case of $\lceil \frac{K}{P} \rceil, N, 1, PL$. The value of $\frac{K}{P}$ is taken to ensure that the total number of message symbols in the databases are the same for SPIR and MM-SPIR. If $\frac{K}{P}$ is not an integer, we choose $\lceil \frac{K}{P} \rceil$, in which case there exist some redundant message symbols in SPIR. The remedy is to make all these redundant message symbols dummy. In other words, all these redundant message symbols are set to be 0, and thus will not make any difference in the following process. Thus far, the only remaining step is to change the message symbol index such that the converted scheme is consistent with the original MM-SPIR problem with message length L.

For clarity, we consider a simple MM-SPIR problem with $K = 4, N = 3, P = 2, L = 1$. The first step is to build an achievable scheme with $K = 2, N = 3, P =$

$1, L = 2$ according to [8, Section IV.B.1]. Assume that we are only interested in the first message $W_1 = [W_{1,1}, W_{1,2}]$ but not the second message $W_2 = [W_{2,1}, W_{2,2}]$ without loss of generality. The concrete scheme is given next: The queries sent to the databases are,

$$Q_1^{[1]} = [h_1 \quad h_2 \quad h_3 \quad h_4] \tag{2.147}$$

$$Q_2^{[1]} = [h_1 + 1 \quad h_2 \quad h_3 \quad h_4] \tag{2.148}$$

$$Q_3^{[1]} = [h_1 \quad h_2 + 1 \quad h_3 \quad h_4] \tag{2.149}$$

where h_1, h_2, h_3, h_4 are all uniform bits in $\mathbb{F}_2$. The corresponding answers received from all the database are,

$$A_1^{[1]} = h_1 W_{1,1} + h_2 W_{1,2} + h_3 W_{2,1} + h_4 W_{2,2} + S \tag{2.150}$$

$$A_2^{[1]} = h_1 W_{1,1} + h_2 W_{1,2} + h_3 W_{2,1} + h_4 W_{2,2} + W_{1,1} + S \tag{2.151}$$

$$A_3^{[1]} = h_1 W_{1,1} + h_2 W_{1,2} + h_3 W_{2,1} + h_4 W_{2,2} + W_{1,2} + S \tag{2.152}$$

After tuning the message symbol index to coincide with the original MM-SPIR problem, again assuming that the desired message indices are 1 and 2, the ultimate scheme for original MM-SPIR problem with $K = 4, N = 3, P = 2, L = 1$ is as follows: The queries sent to the databases are,

$$Q_1^{[1,2]} = [h_1 \quad h_2 \quad h_3 \quad h_4] \tag{2.153}$$

$$Q_2^{[1,2]} = [h_1 + 1 \quad h_2 \quad h_3 \quad h_4] \tag{2.154}$$

$$Q_3^{[1,2]} = [h_1 \quad h_2 + 1 \quad h_3 \quad h_4] \tag{2.155}$$

where h_1, h_2, h_3, h_4 are all uniform bits in $\mathbb{F}_2$. The corresponding answers received from all the databases are,

$$A_1^{[1,2]} = h_1 W_1 + h_2 W_2 + h_3 W_3 + h_4 W_4 + S \tag{2.156}$$

$$A_2^{[1,2]} = h_1 W_1 + h_2 W_2 + h_3 W_3 + h_4 W_4 + W_1 + S \tag{2.157}$$

$$A_3^{[1,2]} = h_1 W_1 + h_2 W_2 + h_3 W_3 + h_4 W_4 + W_2 + S \tag{2.158}$$

In summary, we can always readily construct an MM-SPIR scheme with parameters P, L on the basis of a single-message SPIR scheme with parameters $1, PL$ such that the induced download cost and the amount of common randomness for an arbitrary fixed message length are both optimal. The optimal values are exactly the ones given in (2.144) and (2.146).

2.6.3 Mapping MM-LSPIR Back to PSI

Finally, we map our MM-SPIR results back to the PSI problem to obtain Theorem 2.1. Recall that, in the PSI problem, by generating the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ by i.i.d. drawing the elements from the alphabet $\mathcal{P}_{alph}$, we obtain i.i.d. messages in the corresponding MM-SPIR problem. Further, by choosing the probability q_i of choosing each element to be included in the set $\mathcal{P}_i$ to be $q_i = \frac{1}{2}$, for $i = 1, 2$, we obtain uniformly distributed messages, with message size $L = 1$. Therefore, the PSI prob-

lem is equivalent to an MM-LSPIR problem with $L = 1$. Now, using Theorem 2.3 with $L = 1$, we obtain the ultimate result of this chapter in Theorem 2.1.

2.7 Conclusion

In this chapter, we investigated the two-party PSI problem over a finite set $\mathbb{S}_K$ from an information-theoretic point of view. We showed that the problem can be recast as an MM-SPIR problem with a message size 1. This is under the assumption that the sets (or their corresponding incidence vectors) can be stored in replicated and non-colluding databases. Further, the set elements are generated in an i.i.d. fashion with a probability $\frac{1}{2}$ of adding any element to any of the sets.

To that end, we explored the information-theoretic capacity of MM-SPIR as a stand-alone problem. We showed that joint multi-message retrieval does not outperform the successive application of single-message SPIR. This is unlike the case of MM-PIR, where significant performance gains can be obtained due to joint retrieval. We remark that SM-SPIR is a special case of the problem studied in this chapter by plugging $P = 1$. For the converse proof, we extended the proof techniques of [8] to the setting of multi-messages. In particular, the proof of Lemma 2.5 is significantly more involved than the proof in [8]. This is due to the fact that the desired message subsets in the case of MM-SPIR may not be disjoint. To unify the query structures of MM-PIR and MM-SPIR, we proposed a new capacity-achieving scheme for any P as an alternative to the successive usage of the scheme in [8]. Our scheme primarily consists of three steps: Exploiting the achievable scheme in [23], making necessary

repetitions to symmetrize the scheme, and adding the needed common randomness properly. The last step is inspired by [15]. Based on these results, we showed that the optimal download cost for PSI is $\min\left\{\left\lceil\frac{P_1N_2}{N_2-1}\right\rceil, \left\lceil\frac{P_2N_1}{N_1-1}\right\rceil\right\}$.

In the following subsections, we make a few remarks about assumptions made in this chapter, and directions for further research.

2.7.1 Data Generation Model

In this work, we add elements to each set in an i.i.d. manner and with probability $\frac{1}{2}$. This assumption is made for two reasons, first, to have i.i.d. incidence vectors, therefore, i.i.d. messages in the MM-SPIR problem, and second, to have uniform messages to avoid the need for compressing the messages $W_{1:K}$ before/within retrieval. However, this assumption may be restrictive, as with this assumption, the expected sizes of both sets are $\frac{K}{2}$. Even with keeping the i.i.d. generation assumption, the probability of adding each element to set i could be generalized to be an arbitrary q_i. In this more general case, the expected sizes of the sets, Kq_1 and Kq_2, could be arbitrary. This may be done by using appropriate compression before/during retrieval, but needs to be studied further. Regarding the i.i.d. selection of elements, while this assumption is not needed from the achievability side, it is needed for the converse proof. To overcome these restrictions, as future work, it may be worthwhile to investigate the MM-SPIR problem with correlated messages.

2.7.2 Upload Cost Reduction

In this chapter, we have focused on the download cost as the sole performance metric. A more natural performance metric is to consider the combined upload and download cost. In this section, we provide an illustrative example, which shows that the upload cost may be reduced without sacrificing the download cost. Nevertheless, the characterization of the optimal combined upload and download cost is an interesting future direction that is outside the scope of this chapter.

Example 5: Consider the SPIR problem with $K = 3$, $N = 2$, $P = 1$, $L = 1$. The original SPIR scheme in [8] achieves the optimal download cost of $D = 2$ bits, while the upload cost is $U = 6$ bits. Inspired by [73], we show that the upload cost can be reduced to just 4 bits without increasing the download cost. Our new achievable scheme is as follows:

For any one of the two databases, there are four possible answers $A_n^{(q)}$, where $n \in [2], q \in [4]$ and common randomness S is a uniformly distributed bit:

$$A_1^{(1)} = W_1 + W_2 + W_3 + S, \qquad A_2^{(1)} = W_2 + W_3 + S \tag{2.159}$$

$$A_1^{(2)} = W_1 + S, \qquad A_2^{(2)} = S \tag{2.160}$$

$$A_1^{(3)} = W_2 + S, \qquad A_2^{(3)} = W_1 + W_2 + S \tag{2.161}$$

$$A_1^{(4)} = W_3 + S, \qquad A_2^{(4)} = W_1 + W_3 + S \tag{2.162}$$

The corresponding queries for different desired messages are generated accord-

ing to the following distributions:

$$\begin{aligned} W_1 : (Q_1^{[1]}, Q_2^{[1]}) \quad &\text{is uniform over} \quad \{(1,1),(2,2),(3,3),(4,4)\}, \\ W_2 : (Q_1^{[2]}, Q_2^{[2]}) \quad &\text{is uniform over} \quad \{(1,4),(2,3),(3,2),(4,1)\}, \\ W_3 : (Q_1^{[3]}, Q_2^{[3]}) \quad &\text{is uniform over} \quad \{(1,3),(2,4),(3,1),(4,2)\}. \end{aligned}$$

The reliability constraint follows from the fact for every query pair, the user can cancel the interfering messages and the common randomness S from the other database. For the database-privacy constraint, we note that the undesired messages are always mixed with S. Hence, the information leakage from undesired messages is zero. For the user-privacy constraint, we have

$$P(Q_n^{[k]} = q) = P(Q_n^{[k']} = q), \quad \forall k, k' \in [3],\ \forall n \in [2],\ \forall q \in [4] \tag{2.163}$$

i.e., from the point of view of any database, the same set of queries is used for any desired message W_i, where $i = 1, 2, 3$ with the same probability distribution.

For the proposed scheme, the required download cost is $D = 2$ bits and the required upload cost is $U = 4$ bits, which outperforms the one in [8] in terms of upload cost.

2.7.3 Communication Model

We note that our optimality result is restricted to the presented communication scenario, where a sender submits queries to a receiver in one round. An interesting

future direction is to investigate whether there is a more efficient communication scheme or whether there can be an impossibility result that can assert that no other communication scheme can outperform our presented scheme.

2.7.4 Single Database Assumption

Our scheme is infeasible for $N_1 = N_2 = 1$ due to the capacity result for MM-SPIR. It would be interesting to see if MM-SPIR can be made feasible with certain modifications to the problem, e.g., side information, or alternatively, if PSI can be transformed into other problems, in the case of a single-server. We can refer to the work in Chapter 3.

CHAPTER 3

Symmetric Private Information Retrieval at the Private Information Retrieval Rate

3.1 Introduction

In this chapter, we consider the problem of SPIR with user-side common randomness. Note that the privacy constraint in SPIR is *symmetric* between the user and the databases, SPIR has the following three properties: its capacity is smaller than the capacity of PIR which requires only user privacy; it is infeasible in the case of a single database; and it requires presence of shared common randomness among the databases. We introduce a new variant of SPIR where the user is provided with a random subset of the shared database common randomness, which is unknown to the databases. We determine the exact capacity region of the triple (d, ρ_S, ρ_U), where d is the download cost, ρ_S is the amount of shared database (server) common randomness, and ρ_U is the amount of available user-side common randomness. The user-side common randomness utilized here can be deemed as auxiliary randomness data. We show that with a suitable amount of ρ_U, this new variant of SPIR achieves the capacity of the conventional PIR. As a corollary, single-database

SPIR becomes feasible. Further, the presence of user-side ρ_U reduces the amount of required server-side ρ_S.

3.2 Problem Formulation

We consider a system of $N \geq 1$ non-colluding databases each storing the same set of $K \geq 2$ i.i.d. messages each of which consisting of L i.i.d. symbols uniformly selected from a sufficiently large finite field $\mathbb{F}_q$, i.e.,

$$H(W_k) = L, \quad k \in [K] \tag{3.1}$$

$$H(W_{1:K}) = H(W_1) + \cdots + H(W_K) = KL \tag{3.2}$$

For convenience, we denote a random variable and its realization by using the same general uppercase letter when distinction is clear from the context. We address this issue additionally whenever clarification is needed. As in [8], we use a random variable $\mathcal{F}$ to denote the randomness in the retrieval strategy implemented by the user. Due to the user privacy constraint, the realization of $\mathcal{F}$ is only known to the user, and is unknown to any of the databases. Due to the database privacy constraint, databases need to share some amount of common randomness $\mathcal{R}_S$; we will call this *server-side* common randomness. The server-side common randomness $\mathcal{R}_S$ with size M is a set of i.i.d. symbols $\{S_1, S_2, \cdots, S_M\}$ uniformly selected from $\mathbb{F}_q$, i.e.,

$$H(S_m) = 1, \quad m \in [M] \tag{3.3}$$

$$H(S_{1:M}) = H(S_1) + \cdots + H(S_M) = M \tag{3.4}$$

Moreover, the set of indices $\{1, 2, \cdots, M\}$ forms an alphabet $\mathcal{A}$, i.e., $\mathcal{A} = \{1, 2, \cdots, M\}$. Before the retrieval process starts, the user obtains a partial knowledge of $\mathcal{R}_S$. We denote it by $\mathcal{R}_U$, and call it *user-side* common randomness. Therefore, we introduce a new random variable $\mathcal{A}_U$ corresponding to the uniform selection of elements without replacement from $\mathcal{A}$ (the sample space of $\mathcal{A}_U$ is the power set of $\mathcal{A}$). User-side common randomness $\mathcal{R}_U$ is a set of i.i.d. symbols from $\mathbb{F}_q$ where the indices of the symbols are constituted by $\mathcal{A}_U$. Further, we assume that $\mathcal{A}_U$ is not known to any individual database and also is kept private throughout the retrieval process[1], although the cardinality of $\mathcal{A}_U$ can be public information to the databases. In addition, we introduce another new random variable $\bar{\mathcal{A}}_U$, which is the complement of $\mathcal{A}_U$ with respect to the universe $\mathcal{A}$, i.e., $\bar{\mathcal{A}}_U = \mathcal{A} \backslash \mathcal{A}_U$. Likewise, $\mathcal{R}_S \backslash \mathcal{R}_U$ is also a set of i.i.d. symbols from $\mathbb{F}_q$ where the indices of symbols are constituted by $\bar{\mathcal{A}}_U$. Thus, after determining the selection $\mathcal{A}_U$, $\bar{\mathcal{A}}_U$ is also deterministic; see Fig. 3.1 for the specific system model.

The server-side common randomness $\mathcal{R}_S$ is generated independently of the stored message set in the databases. The desired message index k, the random selection $\mathcal{A}_U$ and the retrieval strategy randomness $\mathcal{F}$, are all determined at the user-side before the retrieval process starts. Moreover, all these random variables

[1]We note that this assumption is with some loss of generality. There could be a version of the problem where we do not care about the privacy of $\mathcal{A}_U$ against the databases during the retrieval process. This version of the problem could potentially have a higher retieval rate. This choice is akin to enforcing "W-privacy" versus "W-S privacy" (see [27, 29, 30, 32–34, 36, 37, 39, 111, 112], especially [32, 33]), where W-privacy stands for message privacy only and W-S privacy stands for message and side-information privacy in a PIR setting with side information.

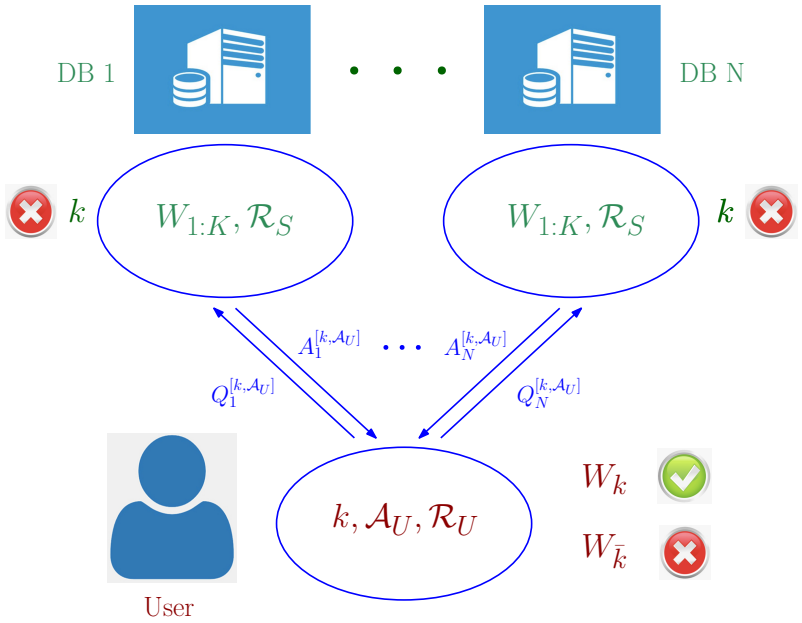


Figure 3.1: System model for SPIR with user-side common randomness.

are mutually independent, thus,

$$H(W_{1:K}, \mathcal{R}_S, k, \mathcal{A}_U, \mathcal{F}) = H(W_{1:K}) + H(\mathcal{R}_S) + H(k) + H(\mathcal{A}_U) + H(\mathcal{F}) \tag{3.5}$$

Using the desired message index and the user-side common randomness indices, the user generates a query for each database according to the retrieval strategy randomness $\mathcal{F}$. Hence, the queries $Q_n^{[k,\mathcal{A}_U]}, n \in [N]$ are deterministic functions of $\mathcal{F}$,

$$H(Q_1^{[k,\mathcal{A}_U]}, Q_2^{[k,\mathcal{A}_U]}, \ldots, Q_N^{[k,\mathcal{A}_U]} | \mathcal{F}) = 0, \quad \forall k, \ \forall \mathcal{A}_U \tag{3.6}$$

During the independent query generation stage, (3.5) and (3.6) lead to the following relationship,

$$I(Q_{1:N}^{[k,\mathcal{A}_U]}; W_{1:K}, \mathcal{R}_S) = 0, \quad \forall k, \ \forall \mathcal{A}_U \tag{3.7}$$

After receiving a query from the user, each database generates a truthful answer based on the stored message set $W_{1:K}$ and the server-side common randomness $\mathcal{R}_S$,

$$H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S) = 0, \quad \forall n,\ \forall k,\ \forall \mathcal{A}_U \tag{3.8}$$

After collecting all N answers from the databases, the user should be able to decode the desired messages W_k reliably,

$$\text{[reliability]} \quad H(W_k|Q_{1:N}^{[k,\mathcal{A}_U]}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \overset{(3.6)}{=} H(W_k|\mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = 0, \quad \forall k,\ \forall \mathcal{A}_U \tag{3.9}$$

Due to the user privacy constraint, the query generated to retrieve the desired message should be statistically indistinguishable from other queries. Specifically, for all k, k', all n, and all $\mathcal{A}_U$, there exists some $\mathcal{A}'_U$ with $|\mathcal{A}'_U| = |\mathcal{A}_U|$, i.e., $H(\mathcal{R}'_U) = H(\mathcal{R}_U)$[2], such that,

$$\text{[user privacy]} \quad (Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S) \sim (Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, W_{1:K}, \mathcal{R}_S) \tag{3.10}$$

As in [46, 66], the joint probability distribution of all random variables at the

[2]In the single-database case, $\mathcal{A}_U$ and $\mathcal{A}'_U$ can not be exactly the same although some overlap is allowed, nor can $\mathcal{R}_U$ and $\mathcal{R}'_U$. Otherwise, user-privacy, database-privacy and reliability constraints jointly form a contradiction, and as a consequence, the problem degenerates to the infeasible conventional single-database SPIR problem, which is trivial. However, this constraint on the strict difference between $\mathcal{A}_U$ and $\mathcal{A}'_U$ (also $\mathcal{R}_U$ and $\mathcal{R}'_U$) does not apply to the multi-database case. This is because its accompanying reliability constraint requires the user to collect the answers from all the databases, not only an individual one. Moreover, in the remaining content of this chapter, we always assume that $\mathcal{A}'_U$ has the same cardinality as $\mathcal{A}_U$ and $\mathcal{R}'_U$ has the same entropy as $\mathcal{R}_U$.

databases can be factorized in the following way,

$$
\begin{aligned}
&P\big((Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S) = (q, a, w_{1:K}, r_S)\big) \\
&\quad = P(Q_n^{[k,\mathcal{A}_U]} = q) \cdot P\big((W_{1:K}, \mathcal{R}_S) = (w_{1:K}, r_S)\big|Q_n^{[k,\mathcal{A}_U]} = q\big) \\
&\qquad \cdot P\big(A_n^{[k,\mathcal{A}_U]} = a\big|(Q_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S) = (q, w_{1:K}, r_S)\big) \qquad (3.11) \\
&\quad = P\big(Q_n^{[k,\mathcal{A}_U]} = q\big) \cdot P\big((W_{1:K}, \mathcal{R}_S) = (w_{1:K}, r_S)\big) \cdot c \qquad (3.12)
\end{aligned}
$$

where the second term in (3.12) comes from the independent query generation of message set as well as server-side common randomness (3.7) and becomes a constant depending on the realizations of the pair $(W_{1:K}, \mathcal{R}_S)$, and the third term c in (3.12) is also a constant either taking the value of 0 or 1 depending on the choice of a because of the fact that the generated answer in a database is a deterministic function of the information that database possesses (3.8). As a consequence, we obtain the following equivalent expression for user privacy for all potential query realizations q,

$$
\text{[user privacy]} \quad P(Q_n^{[k,\mathcal{A}_U]} = q) = P(Q_n^{[k',\mathcal{A}'_U]} = q) \qquad (3.13)
$$

Due to the database privacy constraint, the user should learn nothing about $W_{\bar{k}}$ which is the complement of W_k, i.e., $W_{\bar{k}} = \{W_1, \cdots, W_{k-1}, W_{k+1}, \cdots, W_K\}$,

$$
\text{[database privacy]} \quad I(W_{\bar{k}}; \mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = 0, \quad \forall k,\ \forall \mathcal{A}_U \qquad (3.14)
$$

Again due to the database privacy, the user should not learn all the information about the remaining common randomness among the databases when the retrieval is complete. However, in order to formulate the problem in an easier and clearer way, we add an additional requirement that the user should not gain any knowledge about the remaining common randomness among the databases even after retrieving the desired message,

$$I(\mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) = 0, \quad \forall k, \ \forall \mathcal{A}_U \tag{3.15}$$

An achievable SPIR scheme is a scheme that satisfies the reliability constraint (3.9), the user privacy constraint (3.13) and the database privacy constraint (3.14). The efficiency of a scheme is measured in terms of the number of downloaded bits by the user from all databases denoted by D. We define the normalized download cost d, the normalized server-side common randomness ρ_S, and the normalized user-side common randomness ρ_U, as

$$d = \frac{D}{L}, \qquad \rho_S = \frac{H(\mathcal{R}_S)}{L}, \qquad \rho_U = \frac{H(\mathcal{R}_U)}{L} \tag{3.16}$$

where L is the message length. Thus, the triple (d, ρ_S, ρ_U) is said to be achievable if all three values can be realized simultaneously by a valid achievable scheme. Our goal in this chapter is to determine the capacity region over all achievable triples (d, ρ_S, ρ_U).

3.3 Main Result

We state the main result of this chapter in the following theorem which is the *capacity region* for the triple (d, ρ_S, ρ_U).

Theorem 3.1 *With user-side common randomness, the multi-database SPIR capacity region for $N \geq 2$ and $K \geq 2$ is*

$$d \geq 1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \tag{3.17}$$

$$\rho_S - \rho_U \geq \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \tag{3.18}$$

$$\frac{N-1}{N} d + \rho_U \geq 1 \tag{3.19}$$

$$\frac{N}{N-1} \rho_U + N \rho_S \geq \frac{N}{N-1} \tag{3.20}$$

Remark 3.1 *The capacity region is defined in the form of a triple (d, ρ_S, ρ_U), where d is the reciprocal of the capacity defined in [6, 8], ρ_S is the required amount of common randomness shared among the databases relative to the message size, ρ_U is the total amount of common randomness obtained by the user before the retrieval starts relative to the message size. Theorem 3.1 gives the optimal tradeoff among these three variables and determines the exact capacity region. There are two corner points in this capacity region. The first corner point is $(\frac{N}{N-1}, \frac{1}{N-1}, 0)$, which is an intersection point among (19), (20) and the implicit constraint $\rho_U \geq 0$. The second corner point is $(1 + \frac{1}{N} + \cdots + \frac{1}{N^{K-1}}, \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^K}, \frac{1}{N^K})$, which is an intersection point among (17), (18) and (20). The achievability of the first corner*

point is provided in the existing paper [8]. The new achievability of the second corner point is introduced in Section 3.6. Furthermore, any point on the line segment joining these two corner points can be achieved by time-sharing between these two different schemes. Any other remaining point in the capacity region can be achieved by adding extra common randomness at the user- and server-side simultaneously, or by increasing the server-side common randomness and the download cost.

Remark 3.2 *The right hand side of (3.17) is the optimum normalized download cost of classical PIR,* $d_{PIR} = 1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}$ *[6]. Thus, (3.17) states that* $d \geq d_{PIR}$.

When $\rho_U = 0$, *i.e., when there is no user-side common randomness, (3.19) becomes* $d \geq d_{SPIR}$, *where* $d_{SPIR} = \frac{N}{N-1}$ *is the optimum normalized download cost of classical SPIR [8], (3.20) gives* $\rho_S \geq \frac{1}{N-1}$, *and (3.17) and (3.18) are non-binding. Note that* $d_{SPIR} > d_{PIR}$ *for all* N. *Therefore, when* $\rho_U = 0$, *Theorem 3.1 reduces to the capacity of classical SPIR [8], and it corresponds to the first corner point.*

When $\rho_U = \frac{1}{N^K}$, *both (3.17) and (3.19) become equivalent to* $d \geq d_{PIR}$, *and the new SPIR download cost achieves* $d = d_{PIR}$. *In addition, from (3.18) and (3.20), we deduce that* $\rho_S \geq \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} + \frac{1}{N^K} = \frac{1}{N} d_{PIR}$, *which implies that the minimum amount of required server-side common randomness must be no smaller than the download cost in each database with symmetry across databases. Therefore, when* $\rho_U = \frac{1}{N^K}$, *the new SPIR download cost equals the download cost of the traditional PIR, and it corresponds to the second corner point.*

Corollary 3.1 *When* $\rho_U = 0$, *Theorem 3.1 reduces to the capacity of classical SPIR.*

That is, $d \geq \frac{N}{N-1} = d_{SPIR}$ *and* $\rho_S \geq \frac{1}{N-1} = \frac{1}{N} d_{SPIR}$.

Corollary 3.2 *When* $\rho_U = \frac{1}{N^K}$*, new SPIR download cost equals the download cost of the traditional PIR,* $d = d_{PIR}$*. The required amount of server-side common randomness becomes* $\rho_S \geq \frac{1}{N} d_{PIR}$.

Remark 3.3 *The gap between* ρ_S *and* ρ_U *must be no smaller than a specific value as a function of* N *and* K *as given on the right hand side of (3.18). This comes from the database privacy constraint, where part of the common randomness, i.e.,* $\mathcal{R}_S \backslash \mathcal{R}_U$*, is utilized to hide the undesired messages.*

Remark 3.4 *From (3.20), we observe that the existence of user-side common randomness can help reduce the required amount of server-side common randomness. In fact, from Corollary 3.1, when* $\rho_U = 0$*, we need* $\rho_S \geq \frac{1}{N} d_{SPIR}$*, whereas from Corollary 3.2, when* $\rho_U = \frac{1}{N^K}$*, we need* $\rho_S \geq \frac{1}{N} d_{PIR}$*. Noting that* $d_{PIR} \leq d_{SPIR}$*, the required server-side common randomness in Corollary 3.2 is smaller compared to Corollary 3.1. For instance, for* $N = 2$ *databases and* $K = 2$ *messages, classical SPIR optimum download cost* $d = d_{SPIR} = 2$ *is achieved by* $\rho_S = 1$ *[8]. In Theorem 3.1,* $d = 2$ *can be achieved by* $\rho_S = \frac{3}{4}$ *with* $\rho_U = \frac{1}{4}$.

Remark 3.5 *It is well-known that, for* $N = 1$*, classical SPIR is not feasible [8]. With user-side common randomness, single-database SPIR becomes feasible. The following corollary states the capacity region of this case as a reduction from Theorem 3.1.*

Corollary 3.3 *With user-side common randomness, the single-database SPIR capacity region for $N = 1$ and $K \geq 2$ is*

$$d \geq K \tag{3.21}$$

$$\rho_S - \rho_U \geq K - 1 \tag{3.22}$$

$$\rho_U \geq 1 \tag{3.23}$$

Remark 3.6 *The optimal normalized download cost for single-database PIR is $d = K$ [6, 32], which is achieved by downloading all messages from the database. One of the difficulties of single-database SPIR is that downloading all messages is not a valid SPIR scheme. Corollary 3.3 shows that single-database PIR capacity can be achieved for single-database SPIR by means of user-side common randomness.*

Remark 3.7 *The first two terms in Corollary 3.3 follow from the first two terms in Theorem 3.1. The third term in Corollary 3.3 follows from the last two terms in Theorem 3.1 by multiplying both sides of the fourth term in Theorem 3.1 by $N - 1$.*

Remark 3.8 *Like multi-database SPIR, in the single-database SPIR as well, the gap between ρ_S and ρ_U must be no smaller than a specific value as a function of K as given in (3.22) to avoid information leakage on undesired messages.*

Remark 3.9 *From Corollary 3.3, the minimum download cost for single-database SPIR with user-side common randomness is $d = K$, the minimum required server-side common randomness is $\rho_S = K$, of which $\rho_U = 1$ must be acquired randomly by the user.*

3.4 Motivating Examples

Example 6: We consider a single-database case $N = 1$, $K = 3$ and $L = 1$. We use W_1, W_2 and W_3 to denote the three message symbols uniformly selected from a finite field $\mathbb{F}_q$. The common randomness S_1, S_2 and S_3 stored in the database are also three uniformly selected symbols from $\mathbb{F}_q$. Our new achievable scheme is given in Table 3.1. In Table 3.1, we go from the table on the left hand side to the table on the right hand side by compact denotation of the queries as $q_1 = [W_1 + S_1, W_2 + S_2, W_3 + S_3]$, $q_2 = [W_1 + S_2, W_2 + S_3, W_3 + S_1]$ and $q_3 = [W_1 + S_3, W_2 + S_1, W_3 + S_2]$. This compact notation on the right hand side table makes it more apparent that queries q_1, q_2, q_3 are used for all user-side common randomness settings, e.g., S_1, S_2, S_3 and for all desired messages, e.g., W_1, W_2, W_3, with equal probability.

$\mathcal{R}_U$	desired message		
	W_1	W_2	W_3
S_1	$W_1 + S_1$	$W_2 + S_1$	$W_3 + S_1$
	$W_2 + S_2$	$W_3 + S_2$	$W_1 + S_2$
	$W_3 + S_3$	$W_1 + S_3$	$W_2 + S_3$
S_2	$W_1 + S_2$	$W_2 + S_2$	$W_3 + S_2$
	$W_2 + S_3$	$W_3 + S_3$	$W_1 + S_3$
	$W_3 + S_1$	$W_1 + S_1$	$W_2 + S_1$
S_3	$W_1 + S_3$	$W_2 + S_3$	$W_3 + S_3$
	$W_2 + S_1$	$W_3 + S_1$	$W_1 + S_1$
	$W_3 + S_2$	$W_1 + S_2$	$W_2 + S_2$

$\mathcal{R}_U$	desired message		
	W_1	W_2	W_3
S_1	q_1	q_3	q_2
S_2	q_2	q_1	q_3
S_3	q_3	q_2	q_1

Table 3.1: The query table for the case $N = 1$, $K = 3$, $L = 1$. The table on the right denotes the query sets compactly as q_1, q_2 and q_3.

The reliability constraint follows from the fact that the user can always decode the desired message by using its own common randomness. The database privacy constraint follows from the fact that the undesired messages are always mixed with

unknown common randomness. For the user-privacy constraint, we have for all $k, k' \in [3], k' \neq k$ and a random selection $\mathcal{A}_U \in \big\{\{1\},\{2\},\{3\}\big\}$ under a uniform distribution, there exists another different $\mathcal{A}'_U \in \big\{\{1\},\{2\},\{3\}\big\}$, such that,

$$P(Q^{[k,\mathcal{A}_U]} = q) = P(Q^{[k',\mathcal{A}'_U]} = q) = \frac{1}{3} \tag{3.24}$$

where $q \in \{q_1, q_2, q_3\}$. Specifically from the point of view of the database, the same set of queries can be invoked for any desired message $W_i, i \in [3]$ with the same probability distribution. This scheme achieves $d = 3$, $\rho_U = 1$ and $\rho_S = 3$, which exactly matches the boundary of the SPIR capacity region for $N = 1$ and $K = 3$ in Corollary 3.3.

Example 7: We consider a multi-database case $N = 2$, $K = 2$ and $L = 4$. We use W_1 and W_2 to denote the two messages each consisting of 4 symbols that are uniformly selected from a finite field $\mathbb{F}_q$. The common randomness S_1, S_2 and S_3 shared between the two databases are also uniformly selected symbols from $\mathbb{F}_q$. Then, we use $[a_1, a_2, a_3, a_4]$ as a random uniform permutation of the symbols in the first message W_1, and independently, $[b_1, b_2, b_3, b_4]$ as another random uniform permutation of the symbols in the second message W_2. Our new achievable scheme is given in Table 3.2. Each set of queries shown in Table 3.2 (e.g., $a_1 + S_1, b_1 + S_2, a_3 + b_2 + S_3, a_2 + S_1, b_2 + S_3, a_4 + b_1 + S_2$) is one possible choice after performing message symbol index permutation and unknown server-side common randomness index permutation. Due to space limitations, we use one particular permutation to represent all possible permutation outcomes. During an actual implementation, the

user should uniformly randomly select one random permutation out of all possible permutations.

$\mathcal{R}_U$	desired message: W_1		desired message: W_2	
	DB1	DB2	DB1	DB2
S_1	$a_1 + S_1$ $b_1 + S_2$ $a_3 + b_2 + S_3$	$a_2 + S_1$ $b_2 + S_3$ $a_4 + b_1 + S_2$	$b_1 + S_1$ $a_1 + S_2$ $b_3 + a_2 + S_3$	$b_2 + S_1$ $a_2 + S_3$ $b_4 + a_1 + S_2$
S_2	$a_1 + S_2$ $b_1 + S_3$ $a_3 + b_2 + S_1$	$a_2 + S_2$ $b_2 + S_1$ $a_4 + b_1 + S_3$	$b_1 + S_2$ $a_1 + S_3$ $b_3 + a_2 + S_1$	$b_2 + S_2$ $a_2 + S_1$ $b_4 + a_1 + S_3$
S_3	$a_1 + S_3$ $b_1 + S_1$ $a_3 + b_2 + S_2$	$a_2 + S_3$ $b_2 + S_2$ $a_4 + b_1 + S_1$	$b_1 + S_3$ $a_1 + S_1$ $b_3 + a_2 + S_2$	$b_2 + S_3$ $a_2 + S_2$ $b_4 + a_1 + S_1$

Table 3.2: The query table for the case $N = 2$, $K = 2$, $L = 4$.

Verification that this proposed scheme achieves reliability, user privacy and database privacy constraints is similar to the one analyzed in Example 6. Specifically with regard to the user privacy, for any $n \in [2]$, given a random selection $\mathcal{A}_U \in \big\{\{1\},\{2\},\{3\}\big\}$ under a uniform distribution, a user wishes to retrieve W_k, that database can always find $\mathcal{A}'_U \in \big\{\{1\},\{2\},\{3\}\big\}$ such that $Q_n^{[k',\mathcal{A}'_U]} = Q_n^{[k,\mathcal{A}_U]}$ for all $k' \neq k$. In other words, that database is not able to recognize the desired message index from the query taking into consideration message symbol index permutation and unknown server-side common randomness index permutation. This scheme achieves $d = \frac{3}{2}$, $\rho_U = \frac{1}{4}$ and $\rho_S = \frac{3}{4}$. This is the second corner point of the capacity region in Theorem 3.1 where all inequalities are satisfied with equality, i.e., here $\rho_U = \frac{1}{N^K}$, $d = d_{\text{PIR}} = 1 + \frac{1}{N}$, and $\rho_S = \frac{1}{N} d_{\text{PIR}}$.

Example 8: We consider a multi-database case $N = 3$, $K = 2$, $L = 36$ and $\rho_U = \frac{1}{18}$. We use W_1 and W_2 to denote the two messages each consisting of 36

symbols that are uniformly selected from a finite field $\mathbb{F}_q$. The common randomness $S_1, \cdots, S_{17}$ shared among the three database are also uniformly selected symbols from $\mathbb{F}_q$. Then, We use $[a_1, \ldots, a_{36}]$ as a random uniform permutation of the symbols in the first message W_1, and independently, $[b_1, \ldots, b_{36}]$ as another random uniform permutation of the symbols in the second message W_2. For the first 9 bits of the desired message after message symbol index permutation, i.e., $[a_1, \ldots, a_9]$, we utilize server-side common randomness $\{S_1, S_2, S_3, S_4\}$ and then our new achievable scheme for one random selection of unknown server-side common randomness index permutation is given in Table 3. For the next 9 bits, i.e., $[a_{10}, \ldots, a_{18}]$, we select another set of server-side common randomness, e.g., $\{S_5, S_6, S_7, S_8\}$, and then use our scheme in Table 3 once more. For the last 18 bits, we use the classical SPIR scheme in [8].

$\mathcal{R}_U$	desired message: W_1			desired message: W_2		
	DB1	DB2	DB3	DB1	DB2	DB3
S_1	$a_1 + S_1$	$a_2 + S_1$	$a_3 + S_1$	$b_1 + S_1$	$b_2 + S_1$	$b_3 + S_1$
	$b_1 + S_2$	$b_2 + S_3$	$b_3 + S_4$	$a_1 + S_2$	$a_2 + S_3$	$a_3 + S_4$
	$a_4 + b_2 + S_3$	$a_6 + b_1 + S_2$	$a_8 + b_1 + S_2$	$b_4 + a_2 + S_3$	$b_6 + a_1 + S_2$	$b_8 + a_1 + S_2$
	$a_5 + b_3 + S_4$	$a_7 + b_3 + S_4$	$a_9 + b_2 + S_3$	$b_5 + a_3 + S_4$	$b_7 + a_3 + S_4$	$b_9 + a_2 + S_3$
S_2	$a_1 + S_2$	$a_2 + S_2$	$a_3 + S_2$	$b_1 + S_2$	$b_2 + S_2$	$b_3 + S_2$
	$b_1 + S_3$	$b_2 + S_4$	$b_3 + S_1$	$a_1 + S_3$	$a_2 + S_4$	$a_3 + S_1$
	$a_4 + b_2 + S_4$	$a_6 + b_1 + S_3$	$a_8 + b_1 + S_3$	$b_4 + a_2 + S_4$	$b_6 + a_1 + S_3$	$b_8 + a_1 + S_3$
	$a_5 + b_3 + S_1$	$a_7 + b_3 + S_1$	$a_9 + b_2 + S_4$	$b_5 + a_3 + S_1$	$b_7 + a_3 + S_1$	$b_9 + a_2 + S_4$
S_3	$a_1 + S_3$	$a_2 + S_3$	$a_3 + S_3$	$b_1 + S_3$	$b_2 + S_3$	$b_3 + S_3$
	$b_1 + S_4$	$b_2 + S_1$	$b_3 + S_2$	$a_1 + S_4$	$a_2 + S_1$	$a_3 + S_2$
	$a_4 + b_2 + S_1$	$a_6 + b_1 + S_4$	$a_8 + b_1 + S_4$	$b_4 + a_2 + S_1$	$b_6 + a_1 + S_4$	$b_8 + a_1 + S_4$
	$a_5 + b_3 + S_2$	$a_7 + b_3 + S_2$	$a_9 + b_2 + S_1$	$b_5 + a_3 + S_2$	$b_7 + a_3 + S_2$	$b_9 + a_2 + S_1$
S_4	$a_1 + S_4$	$a_2 + S_4$	$a_3 + S_4$	$b_1 + S_4$	$b_2 + S_4$	$b_3 + S_4$
	$b_1 + S_1$	$b_2 + S_2$	$b_3 + S_3$	$a_1 + S_1$	$a_2 + S_2$	$a_3 + S_3$
	$a_4 + b_2 + S_2$	$a_6 + b_1 + S_1$	$a_8 + b_1 + S_1$	$b_4 + a_2 + S_2$	$b_6 + a_1 + S_1$	$b_8 + a_1 + S_1$
	$a_5 + b_3 + S_3$	$a_7 + b_3 + S_3$	$a_9 + b_2 + S_2$	$b_5 + a_3 + S_3$	$b_7 + a_3 + S_3$	$b_9 + a_2 + S_2$

Table 3.3: The query table for the first 9 bits in the case $N = 3$, $K = 2$, $L = 36$.

We observe from Table 3.3 that, for the first 9 bits, this scheme achieves

$D = 12$, $H(\mathcal{R}_U) = 1$ and $H(\mathcal{R}_S) = 4$. Doubling these, for the first 18 bits, this scheme achieves $D = 24$, $H(\mathcal{R}_U) = 2$ and $H(\mathcal{R}_S) = 8$. For the last 18 bits, we use the classical SPIR scheme in [8], which achieves $D = 27$, $H(\mathcal{R}_U) = 0$ and $H(\mathcal{R}_S) = 9$. Thus, by combining these two different schemes in a time-sharing manner, we ultimately have $d = \frac{24+27}{36} = \frac{17}{12}$, $\rho_U = \frac{2+0}{36} = \frac{1}{18}$, and $\rho_S = \frac{8+9}{36} = \frac{17}{36}$, which corresponds to a point on the line segment joining the first corner point where $\rho_U = 0$ and the second corner point where $\rho_U = \frac{1}{9}$ of the capacity region in Theorem 3.1.

3.5 Converse Proof

In this section, we provide the converse proof of Theorem 3.1. The four inequalities in Theorem 3.1 are proved in Lemmas 3.3, 3.4, 3.9 and 3.10 below. Towards proving these four lemmas, we need Lemmas 3.1-3.2 and Lemmas 3.5-3.8 below. We note that Lemmas 3.1-3.2 extend [6, Lemmas 5-6], and Lemmas 3.5-3.8 extend [8, Lemmas 1-2, Eqn. (39)]. These extensions are needed because we have two additional sets of random variables in our system model: $\mathcal{R}_S$ and $\mathcal{R}_U$ with respect to techniques in [6], and $\mathcal{R}_U$ with respect to techniques in [8].

Lemma 3.1 (Messages Dependence Upper Bound)

$$I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S | W_1) \leq D - L \tag{3.25}$$

Proof:

$$
\begin{aligned}
&I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S | W_1) \\
&\quad = I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S | W_1) + I(W_{2:K}; W_1) && (3.26)\\
&\quad = I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) && (3.27)\\
&\quad = I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) + I(W_{2:K}; W_1 | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) && (3.28)\\
&\quad = I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) && (3.29)\\
&\quad = I(W_{2:K}; A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) + I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) && (3.30)\\
&\quad = I(W_{2:K}; A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) && (3.31)\\
&\quad = H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) - H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.32)\\
&\quad = H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) - H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) \\
&\qquad - H(W_1 | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.33)\\
&\quad = H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S) - H(W_1, A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.34)\\
&\quad \leq H(A_{1:N}^{[1,\mathcal{A}_U]}) - H(W_1, A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.35)\\
&\quad \leq D - H(W_1, A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.36)\\
&\quad = D - H(W_1 | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) - H(A_{1:N}^{[1,\mathcal{A}_U]} | Q_{1:N}^{[1,\mathcal{A}_U]}, W_1, W_{2:K}, \mathcal{R}_S) && (3.37)\\
&\quad = D - H(W_1 | Q_{1:N}^{[1,\mathcal{A}_U]}, W_{2:K}, \mathcal{R}_S) && (3.38)\\
&\quad = D - L && (3.39)
\end{aligned}
$$

where (3.26) follows from the i.i.d. message setting in the databases (3.2), (3.29) follows from the reliable decoding of the first message (3.9), (3.31) follows from the

independence of the message set (3.5) and the independent query generation (3.7), (3.33) follows from the reliable decoding of the first message (3.9) again, (3.38) follows from the truthful deterministic answer generation at each database (3.8), (3.39) follows from the joint application of (3.1), (3.2), (3.5) and (3.7). ■

Lemma 3.2 (Messages Dependence Lower Bound)

$$I(W_{k:K}; Q_{1:N}^{[k-1,\mathcal{A}_U]}, A_{1:N}^{[k-1,\mathcal{A}_U]}, \mathcal{R}_S | W_{1:k-1}) \geq \frac{1}{N} I(W_{k+1:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}, \mathcal{R}_S | W_{1:k}) + \frac{L}{N}, \quad \forall k \in [2:K] \tag{3.40}$$

Proof:

$$\begin{aligned}
&NI(W_{k:K}; Q_{1:N}^{[k-1,\mathcal{A}_U]}, A_{1:N}^{[k-1,\mathcal{A}_U]}, \mathcal{R}_S | W_{1:k-1}) \\
&\geq \sum_{n=1}^{N} I(W_{k:K}; Q_n^{[k-1,\mathcal{A}_U]}, A_n^{[k-1,\mathcal{A}_U]}, \mathcal{R}_S | W_{1:k-1}) && (3.41)\\
&= \sum_{n=1}^{N} I(W_{k:K}; Q_n^{[k,\mathcal{A}'_U]}, A_n^{[k,\mathcal{A}'_U]}, \mathcal{R}_S | W_{1:k-1}) && (3.42)\\
&\geq \sum_{n=1}^{N} I(W_{k:K}; A_n^{[k,\mathcal{A}'_U]} | Q_n^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) && (3.43)\\
&= \sum_{n=1}^{N} \left(H(A_n^{[k,\mathcal{A}'_U]} | Q_n^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) - H(A_n^{[k,\mathcal{A}'_U]} | Q_n^{[k,\mathcal{A}'_U]}, W_{1:K}, \mathcal{R}_S) \right) && (3.44)\\
&= \sum_{n=1}^{N} H(A_n^{[k,\mathcal{A}'_U]} | Q_n^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) && (3.45)\\
&\geq \sum_{n=1}^{N} H(A_n^{[k,\mathcal{A}'_U]} | Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:n-1}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) && (3.46)\\
&= \sum_{n=1}^{N} \big(H(A_n^{[k,\mathcal{A}'_U]} | Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:n-1}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S)
\end{aligned}$$

$$- H(A_n^{[k,\mathcal{A}'_U]}|Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:n-1}^{[k,\mathcal{A}'_U]}, W_{1:K}, \mathcal{R}_S)\big) \tag{3.47}$$

$$= \sum_{n=1}^{N} I(W_{k:K}; A_n^{[k,\mathcal{A}'_U]}|Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:n-1}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) \tag{3.48}$$

$$= I(W_{k:K}; A_{1:N}^{[k,\mathcal{A}'_U]}|Q_{1:N}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) \tag{3.49}$$

$$= I(W_{k:K}; A_{1:N}^{[k,\mathcal{A}'_U]}|Q_{1:N}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) + I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k-1}, \mathcal{R}_S) \tag{3.50}$$

$$= I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k-1}, \mathcal{R}_S) \tag{3.51}$$

$$= I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k-1}, \mathcal{R}_S) + I(W_{k:K}; W_k|Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}, W_{1:k-1}, \mathcal{R}_S) \tag{3.52}$$

$$= I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}, W_k|W_{1:k-1}, \mathcal{R}_S) \tag{3.53}$$

$$= I(W_{k:K}; W_k|W_{1:k-1}, \mathcal{R}_S) + I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, \mathcal{R}_S) \tag{3.54}$$

$$= L + I(W_{k:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, \mathcal{R}_S) \tag{3.55}$$

$$= I(W_{k+1:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, \mathcal{R}_S)$$
$$+ I(W_k; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, W_{k+1:K}, \mathcal{R}_S) + L \tag{3.56}$$

$$= I(W_{k+1:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, \mathcal{R}_S) + L \tag{3.57}$$

$$= I(W_{k+1:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}|W_{1:k}, \mathcal{R}_S) + I(W_{k+1:K}; \mathcal{R}_S|W_{1:k}) + L \tag{3.58}$$

$$= I(W_{k+1:K}; Q_{1:N}^{[k,\mathcal{A}'_U]}, A_{1:N}^{[k,\mathcal{A}'_U]}, \mathcal{R}_S|W_{1:k}) + L \tag{3.59}$$

where (3.42) follows from the application of user-privacy (3.10), (3.45) and (3.47) both follow from truthful deterministic answer generation by each database (3.8), (3.50) follows from (3.2), (3.5) and (3.7), (3.52) follows from reliability constraint (3.9), (3.55) follows from (3.1), (3.2) and (3.5), (3.58) follows from (3.2) and (3.5) again. Dividing both sides by N completes the proof. ■

Lemma 3.3 (Minimal download cost d)

$$d \geq 1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \tag{3.60}$$

Proof: Following steps similar to [6, Eqns. (62)-(67)] for Lemma 3.2, we obtain

$$I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S | W_1) \geq \left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \right) L \tag{3.61}$$

Combining the upper bound in Lemma 3.1 and the lower bound in (3.61) completes the proof. ■

Lemma 3.4 (Minimal difference between ρ_S and ρ_U)

$$\rho_S - \rho_U \geq \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \tag{3.62}$$

Proof: From (3.61), we have,

$$\begin{aligned}
&I(W_{2:K}; Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, \mathcal{R}_S | W_1) \\
&\quad = H(W_{2:K}|W_1) - H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) && (3.63) \\
&\quad = (K-1)L - H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) && (3.64) \\
&\quad \geq \left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \right) L && (3.65)
\end{aligned}$$

Thus, we obtain,

$$H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) \leq (K-1)L - \left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}\right) L \tag{3.66}$$

Next, we have the following upper bound,

$$\begin{aligned}
&I(W_{2:K}; \mathcal{R}_S \backslash \mathcal{R}_U | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_U) \\
&\quad = H(\mathcal{R}_S \backslash \mathcal{R}_U | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_U) - H(\mathcal{R}_S \backslash \mathcal{R}_U | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_U) && (3.67) \\
&\quad \leq H(\mathcal{R}_S \backslash \mathcal{R}_U | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_U) && (3.68) \\
&\quad = H(\mathcal{R}_S \backslash \mathcal{R}_U) && (3.69) \\
&\quad = H(\mathcal{R}_S) - H(\mathcal{R}_U) && (3.70)
\end{aligned}$$

where (3.69) follows from the presumed independence of the remaining common randomness among the databases when the retrieval is complete (3.15).

In addition, we have the following lower bound,

$$\begin{aligned}
&I(W_{2:K}; \mathcal{R}_S \backslash \mathcal{R}_U | Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_U) \\
&\quad = H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_U) - H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) && (3.71) \\
&\quad = (K-1)L - H(W_{2:K}|Q_{1:N}^{[1,\mathcal{A}_U]}, A_{1:N}^{[1,\mathcal{A}_U]}, W_1, \mathcal{R}_S) && (3.72) \\
&\quad \geq (K-1)L - (K-1)L + \left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}\right) L && (3.73)
\end{aligned}$$

$$= \left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} \right) L \tag{3.74}$$

where (3.72) follows from the database privacy constraint (3.14) in the realization of $k = 1$ and reliability constraint (3.9), and (3.73) follows from (3.66).

Combining (3.70) and (3.74) yields the desired result. ■

Lemma 3.5 (Different Indices Effect on the Same Message)

$$H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{R}'_U) - H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \leq H(\mathcal{R}_U), \quad \forall k' \neq k \tag{3.75}$$

Proof: From the user privacy constraint (3.10), we have,

$$H(W_k|Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}_U) \tag{3.76}$$

From the deterministic queries relying on the retrieval strategy (3.6), database privacy constraint (3.14), and the fact $W_k \in W_{\bar{k}'}$, we have,

$$0 = I(W_{\bar{k}'}; Q_{1:N}^{[k',\mathcal{A}'_U]}, A_{1:N}^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.77}$$

$$= I(W_k; Q_{1:N}^{[k',\mathcal{A}'_U]}, A_{1:N}^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.78}$$

$$= I(W_k; Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}|\mathcal{R}'_U) \tag{3.79}$$

$$= H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}) - H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.80}$$

Using the equations derived above, we derive an upper bound for the following term,

$$
\begin{aligned}
&H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) - H(W_k|Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \\
&\quad = H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}) - H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}_U) && (3.81)\\
&\quad = I(W_k; \mathcal{R}_U|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}) && (3.82)\\
&\quad = H(\mathcal{R}_U|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}) - H(\mathcal{R}_U|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, W_k) && (3.83)\\
&\quad \leq H(\mathcal{R}_U|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}) && (3.84)\\
&\quad \leq H(\mathcal{R}_U) && (3.85)
\end{aligned}
$$

where (3.81) follows from (3.76) and (3.80). This is different from the equivalence $H(W_k|Q_n^{[k']}, A_n^{[k']}) = H(W_k|Q_n^{[k]}, A_n^{[k]})$ in the SPIR problem without user-side common randomness and it leads to the difference between our Lemma 3.5 and [8, Lemma 1].

Once again from the user privacy constraint (3.10), we have,

$$
H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_S) = H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}_S) \tag{3.86}
$$

which is gives the following equality,

$$
\begin{aligned}
&H(\mathcal{R}_S \backslash \mathcal{R}_U|Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \\
&\quad = H(\mathcal{R}_S \backslash \mathcal{R}'_U|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) + H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) && (3.87)
\end{aligned}
$$

Noting the independence of the remaining common randomness among the databases

after the retrieval process (3.15), thus, we have,

$$H(\mathcal{R}_S \backslash \mathcal{R}_U) + H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = H(\mathcal{R}_S \backslash \mathcal{R}'_U) + H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.88}$$

which leads to

$$H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.89}$$

Likewise, without taking into consideration the answers, we also have,

$$H(Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = H(Q_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) \tag{3.90}$$

As a consequence, we derive the following relation by utilizing the independent message set (3.5) and also deterministic queries (3.6),

$$H(Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) = H(Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(W_k) \tag{3.91}$$

$$= H(Q_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) + H(W_k) \tag{3.92}$$

$$= H(Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{R}'_U) \tag{3.93}$$

Now, we are ready to prove the Lemma 3.5,

$$H(A_n^{[k,\mathcal{A}_U]} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U)$$

$$= H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) - H(Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.94}$$

$$= H(W_k|Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(Q^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) - H(Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.95}$$

$$\geq H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) - H(\mathcal{R}_U) + H(Q_n^{[k,\mathcal{A}_U]}, A_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U)$$
$$- H(Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.96}$$

$$= H(W_k|Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) + H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U)$$
$$- H(Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{A}'_U) - H(\mathcal{R}_U) \tag{3.97}$$

$$= H(Q_n^{[k',\mathcal{A}'_U]}, A_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{A}'_U) - H(Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{A}'_U) - H(\mathcal{R}_U) \tag{3.98}$$

$$= H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{A}'_U) - H(\mathcal{R}_U) \tag{3.99}$$

where (3.96) follows from (3.85), and (3.97) follows from (3.89) and (3.93). ■

Lemma 3.6 (Symmetry)

$$H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) = H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U), \quad \forall k' \neq k \tag{3.100}$$

Proof: The proof of Lemma 3.6 follows from (3.89) and (3.90). ■

Lemma 3.7 (Effect of conditioning on retrieval strategy randomness)

$$H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) = H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.101}$$

Proof: We prove Lemma 3.7 by showing that the following conditional mutual

information is non-positive and thus is zero,

$$
\begin{aligned}
&I(A_n^{[k,\mathcal{A}_U]}; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \\
&\quad \leq I(A_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) && (3.102) \\
&\quad = I(W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) + I(A_n^{[k,\mathcal{A}_U]}; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_S) && (3.103) \\
&\quad = I(W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) && (3.104) \\
&\quad \leq I(W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) + I(W_k; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) && (3.105) \\
&\quad = I(W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F} | Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) && (3.106) \\
&\quad \leq I(W_{1:K}, \mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) && (3.107) \\
&\quad = I(\mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + I(W_{1:K}; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U | \mathcal{R}_S \backslash \mathcal{R}_U) && (3.108) \\
&\quad = I(\mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(W_{1:K} | \mathcal{R}_S \backslash \mathcal{R}_U) - H(W_{1:K} | \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_S) && (3.109) \\
&\quad = I(\mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(W_{1:K}) - H(W_{1:K}) && (3.110) \\
&\quad = I(\mathcal{R}_S \backslash \mathcal{R}_U; \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) && (3.111) \\
&\quad = H(\mathcal{R}_S \backslash \mathcal{R}_U) - H(\mathcal{R}_S \backslash \mathcal{R}_U | \mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) && (3.112) \\
&\quad = H(\mathcal{R}_S \backslash \mathcal{R}_U) - H(\mathcal{R}_S \backslash \mathcal{R}_U) && (3.113) \\
&\quad = 0 && (3.114)
\end{aligned}
$$

where (3.104) follows from the fact that the answer is a deterministic function of the corresponding query, message set and server-side common randomness (3.8), (3.110)

follows from the independence of message set (3.5) and the query is a deterministic function of the realization of retrieval strategy randomness (3.6), and (3.113) follows from the independence of the remaining common randomness among the databases (3.15) and (3.6). ■

Lemma 3.8 (Effect of conditioning on an undesired message)

$$H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]},\mathcal{R}'_U) = H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]},W_k,\mathcal{R}'_U), \quad \forall k' \neq k \tag{3.115}$$

Proof: From the database privacy constraint (3.14) and noting that $W_k \in W_{\bar{k}'}$, we have,

$$\begin{aligned} 0 &= I(W_{\bar{k}'};Q_{1:N}^{[k',\mathcal{A}'_U]},A_{1:N}^{[k',\mathcal{A}'_U]},\mathcal{R}'_U) && (3.116)\\ &= I(W_k;Q_{1:N}^{[k',\mathcal{A}'_U]},A_{1:N}^{[k',\mathcal{A}'_U]},\mathcal{R}'_U) && (3.117)\\ &= I(A_n^{[k',\mathcal{A}'_U]};W_k|Q_n^{[k',\mathcal{A}'_U]},\mathcal{R}'_U) && (3.118)\\ &= H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]},\mathcal{R}'_U) - H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]},W_k,\mathcal{A}'_U) && (3.119) \end{aligned}$$

which is the desired result. ■

Lemma 3.9 (Minimal bound for d and ρ_U)

$$\frac{N-1}{N}d + \rho_U \geq 1 \tag{3.120}$$

Proof: Starting from the message length assumption (3.1),

$$L = H(W_k) \tag{3.121}$$

$$= H(W_k|\mathcal{F}, \mathcal{R}_U) \tag{3.122}$$

$$= H(W_k|\mathcal{F}, \mathcal{R}_U) - H(W_k|\mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \tag{3.123}$$

$$= I(W_k; A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) \tag{3.124}$$

$$= H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.125}$$

$$= H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.126}$$

$$= H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) \tag{3.127}$$

$$\leq H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]}, W_k, \mathcal{R}'_U) + H(\mathcal{R}_U) \tag{3.128}$$

$$= H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k',\mathcal{A}'_U]}|Q_n^{[k',\mathcal{A}'_U]}, \mathcal{R}'_U) + H(\mathcal{R}_U) \tag{3.129}$$

$$= H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) + H(\mathcal{R}_U) \tag{3.130}$$

$$\leq H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) + H(\mathcal{R}_U) \tag{3.131}$$

where (3.122) follows from independence of the message set (3.5), (3.123) follows from the reliable decoding of message W_k, (3.126) and (3.131) both follow from the fact that each query is determined by the retrieval strategy (3.6), (3.127) follows from Lemma 3.7, (3.128) follows from Lemma 3.5, (3.129) follows from Lemma 3.8, (3.130) follows from Lemma 3.6.

By summing (3.131) over all $n \in [1:N]$, we obtain the following relationship,

$$NL \leq NH(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - \sum_{n=1}^{N} H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) + NH(\mathcal{R}_U) \tag{3.132}$$

$$\leq (N-1)H(A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F},\mathcal{R}_U) + NH(\mathcal{R}_U) \tag{3.133}$$

$$\leq (N-1)\sum_{n=1}^{N} H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F},\mathcal{R}_U) + NH(\mathcal{R}_U) \tag{3.134}$$

$$\leq (N-1)D + NH(\mathcal{R}_U) \tag{3.135}$$

which completes the proof. ■

Lemma 3.10 (Minimal bound for ρ_U and ρ_S)

$$\frac{N}{N-1}\rho_U + N\rho_S \geq \frac{N}{N-1} \tag{3.136}$$

Proof: Starting with the database privacy constraint (3.14),

$$0 = I(W_{\bar{k}};\mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \tag{3.137}$$

$$= I(W_{\bar{k}}; A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U|\mathcal{F}) \tag{3.138}$$

$$= I(W_{\bar{k}}; A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U|\mathcal{F}) + I(W_{\bar{k}}; W_k|\mathcal{F}, A_{1:N}^{[k,\mathcal{A}_U]}, \mathcal{R}_U) \tag{3.139}$$

$$= I(W_{\bar{k}}; A_{1:N}^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U|\mathcal{F}) \tag{3.140}$$

$$= I(W_{\bar{k}}; A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) + I(W_{\bar{k}}; W_k, \mathcal{R}_U|\mathcal{F}) \tag{3.141}$$

$$= I(W_{\bar{k}}; A_{1:N}^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.142}$$

$$\geq I(W_{\bar{k}}; A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.143}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_{1:K}, \mathcal{R}_U) + H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_{1:K}, \mathcal{R}_S) \tag{3.144}$$

$$\geq H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_{1:K}, \mathcal{R}_U) + H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_{1:K}, \mathcal{R}_S, \mathcal{R}_U) \tag{3.145}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - I(A_n^{[k,\mathcal{A}_U]}; \mathcal{R}_S|\mathcal{F}, W_{1:K}, \mathcal{R}_U) \tag{3.146}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(\mathcal{R}_S|\mathcal{F}, W_{1:K}, \mathcal{R}_U) + H(\mathcal{R}_S|\mathcal{F}, A_n^{[k,\mathcal{A}_U]}, W_{1:K}, \mathcal{R}_U) \tag{3.147}$$

$$\geq H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(\mathcal{R}_S|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.148}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(\mathcal{R}_U, \mathcal{R}_S \backslash \mathcal{R}_U|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.149}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(\mathcal{R}_S \backslash \mathcal{R}_U|\mathcal{F}, W_k, \mathcal{R}_U) \tag{3.150}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, W_k, \mathcal{R}_U) - H(\mathcal{R}_S \backslash \mathcal{R}_U) \tag{3.151}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|\mathcal{F}, Q_n^{[k,\mathcal{A}_U]}, W_k, \mathcal{R}_U) - H(\mathcal{R}_S) + H(\mathcal{R}_U) \tag{3.152}$$

$$= H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) - H(\mathcal{R}_S) \tag{3.153}$$

where (3.139) follows from the reliability constraint (3.9), (3.142) follows from (3.2) and (3.5), (3.144) follows from the deterministic answer generation by each database (3.8) and (3.6), (3.151) follows from the independent remaining common randomness among the databases (3.15), (3.152) follows from the deterministic queries relying on the retrieval strategy (3.6), and (3.153) follows from the steps between (3.127)-(3.130) by applying Lemma 3.5 through Lemma 3.8 again.

By summing (3.153) over all $n \in [1 : N]$, we obtain the following relationship,

$$0 \geq \sum_{n=1}^{N} H(A_n^{[k,\mathcal{A}_U]}|Q_n^{[k,\mathcal{A}_U]}, \mathcal{R}_U) - NH(\mathcal{R}_S) \tag{3.154}$$

$$\geq H(A_{1:N}^{[1,\mathcal{A}_U]}|\mathcal{F}, Q_n^{[1,\mathcal{A}_U]}, \mathcal{R}_U) - NH(\mathcal{R}_S) \tag{3.155}$$

$$= H(A_{1:N}^{[1,\mathcal{A}_U]}|\mathcal{F}, \mathcal{R}_U) - NH(\mathcal{R}_S) \tag{3.156}$$

$$\geq \frac{N}{N-1}L - \frac{N}{N-1}H(\mathcal{R}_U) - NH(\mathcal{R}_S) \tag{3.157}$$

where (3.157) follows from (3.133), completing the proof. ■

3.6 Achievability Proof

Following the critical idea in [113], our new achievable scheme corresponding to the second corner point in Theorem 3.1 is based on the principle of converting a given PIR scheme into a valid SPIR scheme using the server-side and user-side common randomness in a manner that does not compromise the download cost. To that end, given any existing information-theoretic PIR achievable scheme, we add a new distinct common randomness to each message symbol. The common randomness added to the desired symbols are substracted out as they are available at the user side, and the remaining common randomness unknown to the user are used to protect the undesired messages. There are two main challenges to constructing such an achievable scheme: first is to simultaneously reduce the amount of required server-side and user-side common randomness to the extent possible, and second is to implement this achievable scheme for all possible user-side common randomness realizations which are unknown ahead of time. By means of converting the PIR scheme in [6] to a corresponding valid SPIR scheme, our proposed new achievable scheme consists of the following steps:

1. *Initial PIR query generation:* For given N and K, generate an initial PIR query table for each desired message using the scheme in [6], e.g., Tables 3.1-3.3 without common randomness S_i's.

2. *Server-side common randomness assignment:* Mix all 1-sum symbols from the desired message across all the databases with the same new common randomness. We call it seed common randomness (e.g., S_1 in first three rows of Table 3.2). Assign a new distinct common randomness to every 1-sum symbol from the undesired messages. For every k-sum symbol containing a desired message symbol, mix it with the common randomness from the $(k-1)$-sum symbol having the same $k-1$ undesired message symbols queried at another database. For every k-sum symbol not containing any desired message symbol, assign a new distinct common randomness. Repeat this until k reaches K. We call this whole modified query table a *query cell*.[3]

3. *Server-side common randomness cycling:* While keeping each query cell, create a new one by adding 1 (mod $|\mathcal{A}|$) to each common randomness index (e.g., S_1 becomes S_2 in Table 3.2). Repeat it $|\mathcal{A}|$ times such that each query cell has a different seed common randomness index.

4. *Query cell determination:* The user has $|\mathcal{A}_U|$ server-side common randomness. The user determines the query cell to be invoked, and selects a random permutation within that cell, by matching its user-side common randomness to

[3]As we did in Example 7, in this step and next step, we use one particular permutation to represent all possible permutation outcomes coming from message symbol index permutation and unknown server-side common randomness index permutation. We do not show all possible permutations for simplicity.

the seed common randomness of the cell.

Reliability: The reliability follows from the reliability of the PIR achievable scheme in [6]. One of the desired message symbols is coupled with a common randomness that is known to the user in advance. The other desired message symbols are coupled with interference that are downloaded from other databases.

User Privacy: From the perspective of each database, the same query can be adopted for any desired message with equal probability. Specifically, as in (3.13), for any $n \in [N]$, any $k \in [K]$, any provided $\mathcal{A}_U$, any selected query q, we always have $P(Q_n^{[k,\mathcal{A}_U]} = q)$ being a constant, which does not depend on the realizations of n, k, $\mathcal{A}_U$ and q.

Database Privacy: From the perspective of the user, every undesired message symbol is always mixed with some unknown common randomness. As a result, no information about undesired message is leaked to the user.

Performance: We compute the performance of the proposed achievable scheme with regard to ρ_S, ρ_U and d. As in [6], the message length L is N^K,[4] and d is $1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}$ because the total number of downloaded symbols across all the databases does not change. Combining the first statement of [6, Lemma 1] and our assignment of server-side common randomness in step 2, we calculate the

[4]In Examples 6-7, the message length is strictly $L = N^K$. However, in Example 8, we note that the classical SPIR scheme achieving d_{SPIR} in [8] requires the message length to be a multiple of $N-1 = 2$, and our new SPIR scheme achieving d_{PIR} requires the message length to be a multiple of $N^K = 9$. In order to execute an appropriate half-to-half time-sharing between these two different schemes, we set the overall message length to be 36.

value of $|\mathcal{A}|$,

$$|\mathcal{A}| = 1 + N \cdot \sum_{k=1}^{K-1} (N-1)^{k-1} \binom{K-1}{k} \tag{3.158}$$

$$= 1 + \frac{N}{N-1} \cdot \sum_{k=1}^{K-1} (N-1)^{k} \binom{K-1}{k} \tag{3.159}$$

$$= 1 + \frac{N}{N-1} \cdot \left(\sum_{k=0}^{K-1} \binom{K-1}{k} (N-1)^{k} 1^{K-1-k} - 1 \right) \tag{3.160}$$

$$= 1 + \frac{N}{N-1} \cdot \left((N-1+1)^{K-1} - 1 \right) \tag{3.161}$$

$$= 1 + \frac{N}{N-1} \cdot (N^{K-1} - 1) \tag{3.162}$$

$$= \frac{N^K - 1}{N-1} \tag{3.163}$$

$$= 1 + \cdots + N^{K-1} \tag{3.164}$$

which implies that $\rho_S = \frac{H(\mathcal{R}_S)}{L} = \frac{|\mathcal{A}|}{L}$ is $\frac{1}{N} + \cdots + \frac{1}{N^K}$ since $L = N^K$. The total amount of required user-side common randomness $|\mathcal{A}_U|$ is 1 since the user only has one seed common randomness before the retrieval takes place. Thus, $\rho_U = \frac{H(\mathcal{R}_U)}{L} = \frac{|\mathcal{A}_U|}{L}$ is $\frac{1}{N^K}$ since $L = N^K$.

3.7 Conclusion

In this chapter, we considered SPIR which is a fundamental primitive in cryptography, as an essential building block in many cryptographic applications, such as OT, secure multi-party computation and zero knowledge proofs. Single-database SPIR could be critical in applications where colluding of all databases cannot be ruled out. Further, side-information and/or cached information is a useful dimension to explore

to improve private download rates. Hence, we introduced an extended version of the SPIR problem, where the user randomly fetches a portion of the available shared common randomness at the databases. This fetched database common randomness can be viewed as a form of side-information at the user. We showed that this side-information increases the SPIR rate, and it can increase it to the level of PIR rate. Since single-database SPIR is infeasible while single-database PIR is feasible, the proposed non-trivial use of user-side common randomness makes single-database SPIR feasible. Finally, we determined the exact capacity region of the download cost, database-side common randomness, and user-side common randomness. Open problems include considering upload cost together with download cost in this system and encoding user-side and server-side common randomness as in the coded PIR problem.

CHAPTER 4

Multi-Party Private Set Intersection: An Information-Theoretic Approach

4.1 Introduction

In this chapter, we investigate the problem of MP-PSI. In particular, there are M parties, each storing a data set $\mathcal{P}_i$ over N_i replicated and non-colluding databases, and we want to calculate the intersection of the data sets $\cap_{i=1}^{M}\mathcal{P}_i$ without leaking any information beyond the set intersection to any of the parties. We consider a specific communication protocol where one of the parties, called the leader party, initiates the MP-PSI protocol by sending queries to the remaining parties which are called client parties. The client parties are not allowed to communicate with each other. We propose an information-theoretic scheme that privately calculates the intersection $\cap_{i=1}^{M}\mathcal{P}_i$ with a download cost of $D = \min_{t\in\{1,\cdots,M\}} \sum_{i\in\{1,\cdots M\}\setminus t} \left\lceil \frac{|\mathcal{P}_t|N_i}{N_i-1} \right\rceil$. Similar to the two-party PSI problem, our scheme builds on the connection between the PSI problem and the MM-SPIR problem. Our scheme is a non-trivial generalization of the two-party PSI scheme as it needs an intricate design of the shared common randomness among the client parties before the MP-PSI process starts. Interestingly,

by means of this auxiliary randomness data, in terms of the download cost, our scheme does not incur any penalty due to the more stringent privacy constraints in the MP-PSI problem compared to the two-party PSI problem.

4.2 Problem Formulation

Consider a setting where there are M independent parties[1], denoted by P_i, $i = 1, 2, \cdots, M$. The ith party possesses a data set $\mathcal{P}_i$ for $i \in [1 : M]$. The data set $\mathcal{P}_i$ is stored within N_i replicated and non-colluding databases[2]. Given that K is large enough, the elements in each data set $\mathcal{P}_i$ are picked independently from a finite set $\mathbb{S}_K$ of cardinality K with an arbitrary statistical distribution[3]. More specifically, before the data sets generation, the data sets $\mathcal{P}_i, i \in [1 : M]$ are all random variables and they are mutually independent. We assume that the cardinality of data set $|\mathcal{P}_i|$ is public knowledge.

Motivated by the relation between 2-party PSI and MM-SPIR in [72], the ith party maps its data set $\mathcal{P}_i$ into a searchable list to facilitate PIR. To that end, the party P_i constructs an incidence vector X_i, which is a binary vector of size K

[1]In this work, we only consider semi-honest (honest but curious) parties in the sense that parties exactly follow the prescribed scheme but curious to learn more about the others. MP-PSI under malicious/adversarial attacks and in the presence of dishonest parties is an interesting future direction that is outside the scope of this work.

[2]We note that the multi-server assumption exists in almost all information-theoretic PIR literature. In practice, the data content may be distributed to the databases by a central content generator who does not communicate directly with other parties, i.e., does not have access to the exchanged queries. The databases do not have any direct communication links among each other and they update their content by downloading the data from the content generator. Hence, in this setting, the databases are replicated but not colluding.

[3]The presented achievability scheme works for any data set generation model and even for distribution-free data sets. The specific data set generation model in the 2-party PSI problem in [72] was introduced only for settling the converse.

associated with the data set $\mathcal{P}_i$ for all $i \in [1:M]$, such that

$$X_{i,j} = \begin{cases} 1, & j \in \mathcal{P}_i \\ 0, & j \notin \mathcal{P}_i \end{cases} \tag{4.1}$$

where $X_{i,j}$ is the jth element of X_i for all $j \in \mathbb{S}_K$. Note that X_i is a sufficient statistic for $\mathcal{P}_i$ for a given K. Hence, the MP-PSI determination is performed over X_i instead of $\mathcal{P}_i$.

We consider a specific communication protocol in this work. The parties agree on a *leader* party, which sends queries to the remaining parties and eventually calculates the desired intersection $\cap_{i=1}^{M} \mathcal{P}_i$. The remaining parties are called *client* parties. Without loss of generality, assume that the leader party is P_M. The leader party P_M sends the query $Q_{i,j}^{[\mathcal{P}_M]}$ to the jth database in the client party P_i for all $i \in [1:M-1]$ and $j \in [1:N_i]$. Since P_M has no information about data set $\mathcal{P}_i$ before the communication, the generated queries $Q_{i,j}^{[\mathcal{P}_M]}$ are independent from $\mathcal{P}_i$. Hence,

$$I(Q_{i,j}^{[\mathcal{P}_M]}; \mathcal{P}_i) = 0, \quad \forall i \in [1:M-1],\ \forall j \in [1:N_i] \tag{4.2}$$

The jth database associated with the client party P_i responds truthfully with an answer $A_{i,j}^{[\mathcal{P}_M]}$ for all $i \in [1:M-1]$, and $j \in [1:N_i]$. The answer is a deterministic function of the query $Q_{i,j}^{[\mathcal{P}_M]}$, the data set $\mathcal{P}_i$, and some common randomness[4] $\mathcal{R}_{i,j}$

[4]We note that the common randomness (key) exchange is an interesting stand-alone problem that is outside the scope of this chapter. One practical solution to this problem in our setting is to have an external *helper*, who generates and shares the common randomness prior to the

that is available to the jth database of P_i. Thus,

$$H(A_{i,j}^{[\mathcal{P}_M]}|Q_{i,j}^{[\mathcal{P}_M]}, \mathcal{P}_i, \mathcal{R}_{i,j}) = 0, \quad \forall i \in [1 : M-1],\ \forall j \in [1 : N_i] \tag{4.3}$$

Let us denote all the queries generated by P_M as $Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}$ and all the answers collected by P_M as $A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}$, i.e.,

$$Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]} = \left\{ Q_{i,j}^{[\mathcal{P}_M]} : i \in [1 : M-1], j \in [1 : N_i] \right\} \tag{4.4}$$

$$A_{1:M-1,1:N_i}^{[\mathcal{P}_M]} = \left\{ A_{i,j}^{[\mathcal{P}_M]} : i \in [1 : M-1], j \in [1 : N_i] \right\} \tag{4.5}$$

Three formal requirements are needed to be satisfied for the MP-PSI problem:

First, the leader party P_M should be able to reliably determine the intersection $\mathcal{P} = \cap_{i=1}^{M} \mathcal{P}_i$ based on $Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}$, $A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}$ and the knowledge of $\mathcal{P}_M$ without knowing $|\mathcal{P}|$ in advance. This is captured by the following MP-PSI reliability constraint,

$$\text{[MP-PSI reliability]} \qquad H(\mathcal{P}|Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) = 0 \tag{4.6}$$

Second, the queries sent by P_M should not leak any information about $\mathcal{P}_M$ except the cardinality of $\mathcal{P}_M$ to any individual database. Thus, $\mathcal{P}_M$ should be independent of all the information available in the jth database of P_i for all $i \in$

MP-PSI determination process. The external helper is not involved in the MP-PSI process itself, i.e., it does not observe the queries or the answers. In this case, the client parties do not need to communicate with each other to exchange the common randomness and there is no leakage from their queries/answers to the external helper. We note that the SPIR problem [8] (and by extension our scheme) is infeasible if no common randomness exists.

$[1 : M-1]$ and $j \in [1 : N_i]$. This is described by the following leader's privacy constraint,

$$\text{[Leader's privacy]} \;\; I(\mathcal{P}_M; Q_{i,j}^{[\mathcal{P}_M]}, A_{i,j}^{[\mathcal{P}_M]}, \mathcal{P}_i, \mathcal{R}_{i,j}) = 0, \;\; \forall i \in [1 : M-1], \; \forall j \in [1 : N_i] \tag{4.7}$$

Note that the communication between any two client parties is not allowed in our protocol. This implies that the party P_i is not able to get any information about the remaining $M-2$ client parties. Thus, the mutual independence required by the problem formulation is thereby satisfied from the perspective of the party P_i.

Third, client's privacy requires that the leader party does not learn any information other than the intersection $\mathcal{P}$ from the collected answer strings. Let $X_{i,\bar{\mathcal{P}}}$ be the set of elements in X_i that do not belong to $\mathcal{P}$, i.e., $X_{i,\bar{\mathcal{P}}} = \{X_{i,k} : k \in \bar{\mathcal{P}}\}$. Hence, the set $\left\{X_{1,\bar{\mathcal{P}}}, \cdots, X_{M-1,\bar{\mathcal{P}}}\right\} = \left\{X_{1,k}, \cdots, X_{M-1,k}, k \in \bar{\mathcal{P}}\right\}$ should be independent of all the information available in P_M. Note that if an element in $\mathcal{P}_M$ is not in the intersection $\mathcal{P}$, the leader party is supposed to conclude that not all the client parties contain this element simultaneously. On the basis of this fact, we define a new set $X_{\bar{\mathcal{P}}} = \left\{\{X_{1,\bar{\mathcal{P}}}, \cdots, X_{M-1,\bar{\mathcal{P}}}\} : X_{1,k} + \cdots + X_{M-1,k} < M-1, \forall k \in \mathcal{P}_M \cap \bar{\mathcal{P}}\}\right\}$, we have the following client's privacy constraint,

$$\text{[Client's privacy]} \qquad I(X_{\bar{\mathcal{P}}}; Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) = 0 \tag{4.8}$$

For a given field size K and individual parties with associated databases, an MP-PSI achievability scheme is a scheme that satisfies the MP-PSI reliability con-

straint (4.6), the leader's privacy constraint (4.7) and the client's privacy constraint (4.8). The efficiency of an achievable MP-PSI scheme is measured by its download cost[5] which is the number of downloaded bits (denoted by D) by one of the parties in order to compute the intersection $\mathcal{P}$. The optimal download cost is $D^* = \inf D$ over all MP-PSI achievability schemes.

4.3 Main Result

In this section, we state our main result concerning the performance of our MP-PSI scheme in terms of the download cost. This is summarized in the following theorem, whose proof is given in Section 4.5.

Theorem 4.1 *In the MP-PSI problem with M independent parties with data sets $\mathcal{P}_i$, assuming that the parties follow a leader-to-clients communication policy, if the data sets are stored within N_i replicated and non-colluding databases for $i = 1, \cdots, M$, then the optimal download cost, D^*, is upper bounded by*

$$D^* \leq \min_{t \in \{1,\cdots,M\}} \sum_{i \in \{1,\cdots M\} \setminus t} \left\lceil \frac{|\mathcal{P}_t| N_i}{N_i - 1} \right\rceil \tag{4.9}$$

Remark 4.1 *In the special case of having an arbitrary party P_i where $|\mathcal{P}_i| = K$,*

[5]We note that although a more natural performance metric is to consider the combined upload and download cost, we argue that the upload cost may not scale with the number of MP-PSI determination rounds if the MP-PSI is regularly repeated [72, footnote 8]. Since the core of [72] (and this chapter also) relies on SPIR, we give a detailed discussion of how to reduce the upload cost of the SPIR scheme without sacrificing the download cost in [72, Section 7.2]. The optimal download cost of the SPIR problem is characterized in [8] with keeping the upload cost unconstrained. In addition, the optimal upload cost of the SPIR problem is characterized in [73] with keeping the download cost unconstrained. The optimal combined download and upload cost for the canonical SPIR problem is still an open problem.

we discard this party P_i before we perform the MP-PSI determination process, and thereby, the M-party MP-PSI problem reduces to an $M-1$-party MP-PSI problem. In the extreme case, where all parties have $|\mathcal{P}_i| = |\mathbb{S}_K| = K$, the download cost becomes zero, i.e., no party needs to exchange any information with any other, as the intersection is immediate.

Remark 4.2 *The minimization problem in (4.9) in Theorem 4.1 corresponds to the fact that the parties can agree on the party with the minimum $\sum_{i\in\{1,\cdots M\}\setminus t}\left\lceil\frac{|\mathcal{P}_t|N_i}{N_i-1}\right\rceil$ to be the leader party. We note that the leader party may not be the party with the least $|\mathcal{P}_i|$, as the download cost also depends on the number of the databases at all parties.*

Remark 4.3 *The download cost of our achievability scheme is equal to the sum of the download costs of $M-1$ pair-wise PSI schemes. This implies that there is no penalty incurred due to adopting a stringent clients' privacy constraint over the E_2 privacy constraint. Note that the E_2 privacy constraint is a relaxed version of client's privacy (4.8) when $M=2$ [72]. More specifically, the E_2 privacy constraint asserts that the leakage from elements outside the set $\mathcal{P}_1$ in the answers returned by E_2 is zero, i.e., $I(\bar{\mathcal{P}}_1; A_{1:N_2}^{[\mathcal{P}_1]}) = 0$.*

Remark 4.4 *Our achievability scheme is private in the information-theoretic (absolute) sense and is fairly simple to implement. A drawback of our approach is that it needs multiple replicated non-colluding databases as in the 2-party PSI problem in [72]; otherwise, our scheme is infeasible if $N_i = 1$ for all i.*

Remark 4.5 *Comparing our result with the most closely related information-theoretic MP-PSI schemes [69], we argue that our scheme outperforms theirs in terms of the communication cost as our download cost is linear in both the number of parties M and the size of the sets p, assuming that $|\mathcal{P}_i| = p$ for all $i = 1, \cdots, M$ in contrast of $O(M^4p^2)$ in [69]. We note, however, that the work [69] allows for potential distrust between the parties in the sense that an active adversary may corrupt up to $M/3$ parties. The issue of parties' misbehavior is an interesting future direction for our work, which is outside the scope of this chapter.*

4.4 Motivating Example: 3 Parties with 3 Databases Each ($M = 3$ with $N_1 = N_2 = N_3 = 3$)

In this section, we motivate our scheme by presenting the following example. In this example, we have $M = 3$ parties, each possessing $N_i = 3$ replicated and non-colluding databases. Assume that each party stores an independently generated set $\mathcal{P}_i \subseteq \mathbb{S}_K$, where $\mathbb{S}_K = \{1, 2, 3, 4\}$. Specifically, we assume that $\mathcal{P}_1 = \{1, 2\}$, $\mathcal{P}_2 = \{1, 3\}$, and $\mathcal{P}_3 = \{1, 4\}$. We aim at reliably calculating the intersection $\mathcal{P}_1 \cap \mathcal{P}_2 \cap \mathcal{P}_3 = \{1\}$ without leaking any further information to any of the parties according to the defined communication policy. Without loss of generality, we pick P_3 to be the leader party. The remaining parties P_1, P_2 are referred to as clients.

We map the sets into the corresponding incidence vectors as in [72], i.e., we

construct a vector X_i, such that $X_{i,k} = 1$ if $k \in \mathcal{P}_i$, hence,

$$\text{Party } P_1: \quad \mathcal{P}_1 = \{1,2\} \quad \Rightarrow \quad X_1 = [X_{1,1}\ X_{1,2}\ X_{1,3}\ X_{1,4}]^T = [1\ 1\ 0\ 0]^T \quad (4.10)$$

$$\text{Party } P_2: \quad \mathcal{P}_2 = \{1,3\} \quad \Rightarrow \quad X_2 = [X_{2,1}\ X_{2,2}\ X_{2,3}\ X_{2,4}]^T = [1\ 0\ 1\ 0]^T \quad (4.11)$$

$$\text{Party } P_3: \quad \mathcal{P}_3 = \{1,4\} \quad \Rightarrow \quad X_3 = [X_{3,1}\ X_{3,2}\ X_{3,3}\ X_{3,4}]^T = [1\ 0\ 0\ 1]^T \quad (4.12)$$

To carry out the MP-PSI calculations, the parties agree on a finite field $\mathbb{F}_L$, where L is a prime number such that $L \geq M$. Therefore, we pick $L = 3$ in our case, i.e., all summations are performed as modulo-3 arithmetic.

The leader party P_3 initiates the MP-PSI determination protocol by sending queries $Q_{i,j}^{[\mathcal{P}_3]}$ for $i \in \{1,2\}$ and $j \in \{1,2,3\}$. The queries aim at *privately retrieving* the messages $X_{1,1}$, $X_{1,4}$ and $X_{2,1}$, $X_{2,4}$ using the SPIR retrieval scheme in [8] (the same query structure was introduced in the original work of [5]). Note that in this example we have $N_i = |\mathcal{P}_3| + 1$, thus, the leader party sends exactly 1 query to each client database. More specifically, let h_k, where $k = 1, \cdots, 4$, be a random variable picked uniformly and independently from $\mathbb{F}_3$, then, for client party P_1, the queries sent from the leader party P_3 are generated as follows,

$$Q_{1,1}^{[\mathcal{P}_3]} = [h_1\ h_2\ h_3\ h_4]^T \quad (4.13)$$

$$Q_{1,2}^{[\mathcal{P}_3]} = [h_1 + 1\ h_2\ h_3\ h_4]^T \quad (4.14)$$

$$Q_{1,3}^{[\mathcal{P}_3]} = [h_1\ h_2\ h_3\ h_4 + 1]^T \quad (4.15)$$

i.e., the leader party sends a random vector $\mathbf{h} = [h_1\ h_2\ h_3\ h_4] \in \mathbb{F}_3^4$ to the first

database as a query. The queries for the remaining databases add a 1 to the positions corresponding to $\mathcal{P}_3$. For client party P_2, the leader party submits the same set of queries,

$$Q_{2,1}^{[\mathcal{P}_3]} = [h_1 \;\; h_2 \;\; h_3 \;\; h_4]^T \tag{4.16}$$

$$Q_{2,2}^{[\mathcal{P}_3]} = [h_1 + 1 \;\; h_2 \;\; h_3 \;\; h_4]^T \tag{4.17}$$

$$Q_{2,3}^{[\mathcal{P}_3]} = [h_1 \;\; h_2 \;\; h_3 \;\; h_4 + 1]^T \tag{4.18}$$

Originally in 2-party PSI, the client databases obtain the inner product of X_i and $Q_{i,j}^{[\mathcal{P}_3]}$ and add a common randomness. In MP-PSI, however, we note that applying the answering strategy of [8,72] compromises the clients' privacy constraint (4.8). This is due to the fact that the leader, in this case, can decode that $X_{1,4} = 0$ and $X_{2,4} = 0$ and not only the intersection $\cap_{i=1,2,3}\, \mathcal{P}_i$. Consequently, the clients' databases need to share intricate common randomness prior to the retrieval phase to prevent that. To that end, the client parties generate and/or share the following randomness (see Fig. 4.1):

1. *Local randomness:* This is denoted by the random variable s_i, for $i = 1, 2$. The random variable s_i is picked uniformly from $\mathbb{F}_3$ independent of all data sets and other randomness sources. The local randomness s_i is shared among all the databases belonging to the ith client party and not shared with other parties. This local randomness acts as the common randomness needed for SPIR [8], and is added to the inner product of the incidence vector and the

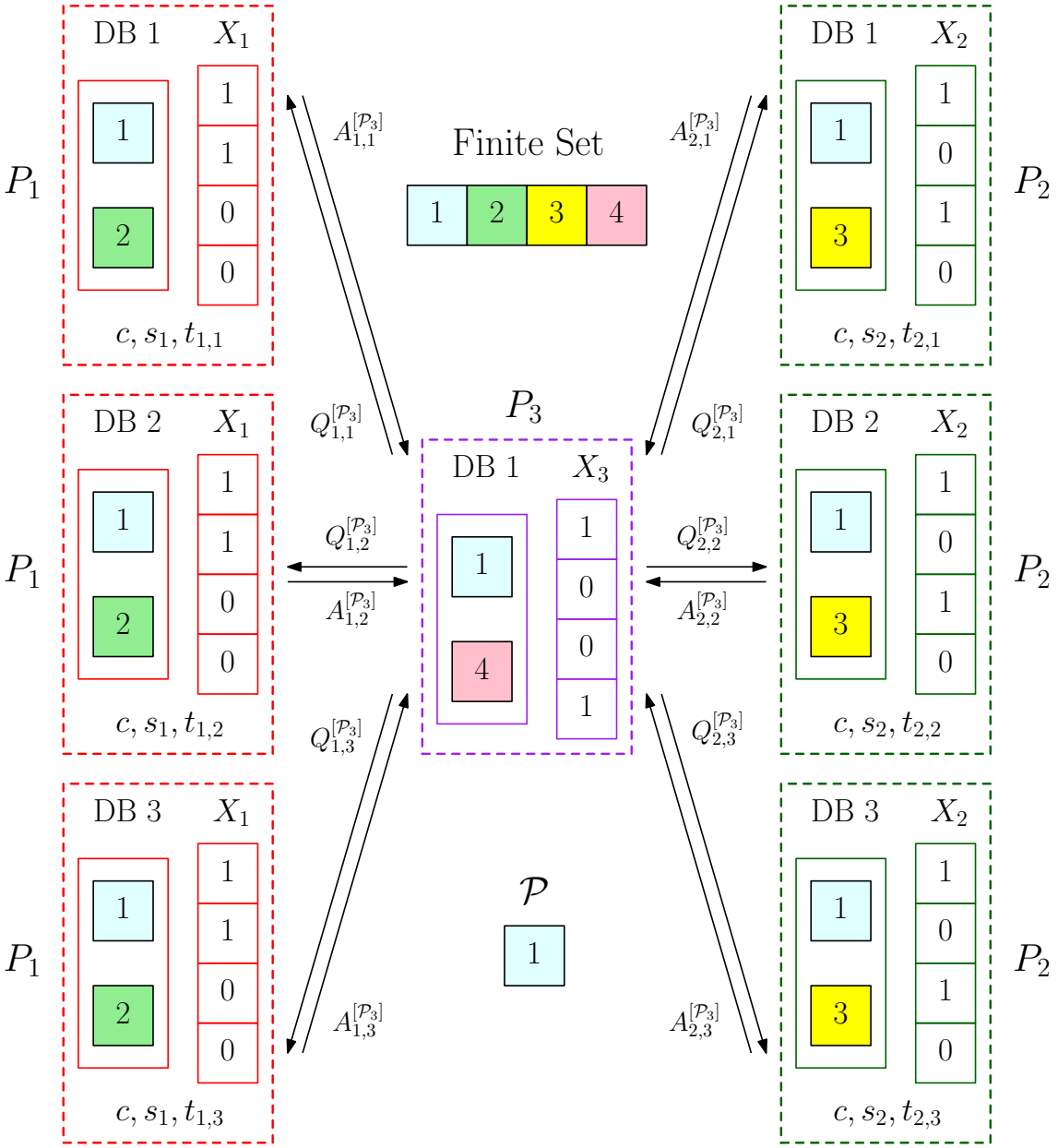


Figure 4.1: MP-PSI for the motivating example.

query.

2. *Individual correlated randomness:* This is possessed by each client's database, and is denoted by the random variables $t_{i,j}$ for $i = 1, 2$, and $j = 1, 2, 3$. This is needed to prevent the leader party from decoding $X_{1,4}$, and $X_{2,4}$. However, since we also need the leader party to decode the intersection, the random

variables $t_{i,j}$ need to be correlated such that their effect can be removed if $X_{i,j}$ belongs to the intersection. To that end, we choose $t_{1,1} = t_{2,1} = 0$. Database 2 of the party P_1 generates uniformly and independently $t_{1,2}$ from $\mathbb{F}_3$ and sends it to database 2 of party P_2. Database 2 of the party P_2 calculates $t_{2,2} = 1 - t_{1,2}$. Similarly, database 3 of the party P_1 generates $t_{1,3}$ uniformly and independently from $\mathbb{F}_3$ and shares it with database 3 of P_2. Hence,

$$t_{1,j} \sim \text{uniform}\{0, 1, 2\}, \quad j = 2, 3 \tag{4.19}$$

$$t_{1,j} + t_{2,j} = 1, \quad j = 2, 3 \tag{4.20}$$

This randomness is added to each response as well. Note that client parties do not know each other's data sets while generating/sharing this randomness.

3. *Global randomness:* This is denoted by the random variable c. The random variable c is generated randomly and independently of all data sets and other randomness variables. The global randomness c is picked uniformly from $\mathbb{F}_3 \setminus \{0\} = \{1, 2\}$. The global randomness is shared among all databases of all client parties P_1 and P_2. The global randomness is used as a multiplier to the responses.

After sharing the common randomness needed to construct the answer strings as shown above, the jth database of the ith client party responds to the query $Q_{i,j}^{[\mathcal{P}_3]}$

as follows,

$$A_{i,j}^{[\mathcal{P}_3]} = c(X_i^T Q_{i,j}^{[\mathcal{P}_3]} + s_i + t_{i,j}), \quad i = 1, 2, \ j = 1, 2, 3 \tag{4.21}$$

Hence, noting that $t_{1,1} = 0$, the answer strings from P_1 can be explicitly written as,

$$A_{1,1}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{1,k} + s_1\right) \tag{4.22}$$

$$A_{1,2}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{1,k} + X_{1,1} + s_1 + t_{1,2}\right) \tag{4.23}$$

$$A_{1,3}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{1,k} + X_{1,4} + s_1 + t_{1,3}\right) \tag{4.24}$$

Similarly, the answer strings from P_2 are,

$$A_{2,1}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{2,k} + s_2\right) \tag{4.25}$$

$$A_{2,2}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{2,k} + X_{2,1} + s_2 + t_{2,2}\right) \tag{4.26}$$

$$A_{2,3}^{[\mathcal{P}_3]} = c\left(\sum_{k=1}^{4} h_k X_{2,k} + X_{2,4} + s_2 + t_{2,3}\right) \tag{4.27}$$

Note that, by this construction, the local randomness s_i is used to protect the random sum $\sum_{k=1}^{4} h_k X_{i,k}$ as in SPIR, and the individual randomness $t_{i,j}$ is needed to prevent the leader party from directly decoding $X_{i,j+1}$. Note that s_1 and s_2 need to be independent to avoid the information leakage about the relationship between $\sum_{k=1}^{4} h_k X_{1,k}$ and $\sum_{k=1}^{4} h_k X_{2,k}$.

Reliability: To calculate $\cap_{i=1,2,3} \mathcal{P}_i$ based on the answer strings the leader

party has received, the leader party subtracts $A_{1,1}^{[\mathcal{P}_3]}$ and $A_{2,1}^{[\mathcal{P}_3]}$ from the remaining answer strings. Denote the result of subtraction related to the jth element in $\mathbb{S}_K$ at P_i by $Z_{i,j}$. This leads to,

$$Z_{1,1} = c(X_{1,1} + t_{1,2}) = A_{1,2}^{[\mathcal{P}_3]} - A_{1,1}^{[\mathcal{P}_3]} \tag{4.28}$$

$$Z_{1,4} = c(X_{1,4} + t_{1,3}) = A_{1,3}^{[\mathcal{P}_3]} - A_{1,1}^{[\mathcal{P}_3]} \tag{4.29}$$

$$Z_{2,1} = c(X_{2,1} + t_{2,2}) = A_{2,2}^{[\mathcal{P}_3]} - A_{2,1}^{[\mathcal{P}_3]} \tag{4.30}$$

$$Z_{2,4} = c(X_{2,4} + t_{2,3}) = A_{2,3}^{[\mathcal{P}_3]} - A_{2,1}^{[\mathcal{P}_3]} \tag{4.31}$$

Now, let E_j be an indicator of having the jth element in $\mathbb{S}_K$ in the intersection $\cap_{i=1,2,3} \mathcal{P}_i$, such that $E_j = 0$ if and only if $j \in \cap_{i=1,2,3} \mathcal{P}_i$. To that end, define E_j as the modulo-L sum of $Z_{i,j}$ along all clients, i.e.,

$$E_j = \sum_{i=1}^{M-1} Z_{i,j} \tag{4.32}$$

Looking deeper at E_1, we note that,

$$E_1 = Z_{1,1} + Z_{2,1} \tag{4.33}$$

$$= c(X_{1,1} + X_{2,1} + t_{1,2} + t_{2,2}) \tag{4.34}$$

$$= c(X_{1,1} + X_{2,1} + 1) \tag{4.35}$$

where $t_{1,2} + t_{2,2} = 1$ by the construction of the individual correlated randomness. Therefore, $E_1 = 0$ if and only if $X_{1,1} = 1$ and $X_{2,1} = 1$ simultaneously. In this

case, $E_1 = 0$ irrespective of the value of c and the leader party verifies that $\{1\} \subseteq \cap_{i=1,2,3} \mathcal{P}_i$.

On the other hand, when P_3 calculates E_4,

$$E_4 = Z_{1,4} + Z_{2,4} = c(X_{1,4} + X_{2,4} + 1) \neq 0 \tag{4.36}$$

Consequently, the leader party confirms that $\cap_{i=1,2,3} \mathcal{P}_i = \{1\}$ and does not include 4.

Leader's Privacy: The leader's privacy constraint follows from the user's privacy constraint of the inherent SPIR scheme [8]. The queries of the leader to any party have the same structure as the queries of the user in the SPIR problem. More specifically, the privacy of leader party is preserved as each element in the queries is uniformly distributed over the finite field $\mathbb{F}_3$. Hence, no information about $\mathcal{P}_3$ is leaked from the queries.

Client's Privacy: To see the client's privacy, we note that no information is leaked about $\overline{\mathcal{P}_1 \cap \mathcal{P}_3}$ or $\overline{\mathcal{P}_2 \cap \mathcal{P}_3}$ due to s_1 and s_2, respectively. Nevertheless, in MP-PSI, we need to verify that the leader does not know which of the two parties possesses the element $\{4\}$, i.e., knowing the fact that $E_4 \neq 0$, we need to show that $\mathbb{P}(X_{1,4} + X_{2,4} = 0) = \mathbb{P}(X_{1,4} + X_{2,4} = 1) = \frac{1}{2}$. Specifically, if E_4 is 1, $\mathbb{P}(X_{1,4} + X_{2,4} = 0) = \mathbb{P}(X_{1,4} + X_{2,4} = 1) = \frac{1}{2}$ because c is uniformly distributed over 1 and 2 and the sum $t_{1,3} + t_{2,3} = 1$ by construction. The conclusion is exactly the same when E_4 equals 2. Thus, the only information that P_3 can obtain for the element 4 is that client parties P_1 and P_2 do not contain it at the same time (this is

no further leak, as if they did contain it at the same time, it would have been in the intersection). Hence, c is used such that the leader party P_3 does not know whether the sum $X_{1,4} + X_{2,4}$ is 0 or 1.

Download Cost: In our example, the leader party P_3 downloads $N_i = |\mathcal{P}_M| + 1$ symbols from each client party. Hence, the total download cost is $D = (M-1)(|\mathcal{P}_M| + 1) = 6$.

4.5 Achievability Proof

In this section, we describe our general achievable scheme for MP-PSI for arbitrary number of parties M, arbitrary set sizes $|\mathcal{P}_i|$, and arbitrary number of databases per party N_i, for $i \in \{1, \cdots, M\}$. The leader's querying policy is based on the SPIR scheme presented in [8] (originally introduced in [5]). Our novel ideas in this scheme are concerned with the construction of the answering strings. More specifically, the scheme hinges on the intricate design of generating and sharing common randomness among the clients' databases in such a way that the leader party cannot learn anything but the intersection $\cap_{i=1}^{M} \mathcal{P}_i$.

4.5.1 General Achievability Scheme

In the following, assume that $\mathcal{P}_i \subseteq \mathbb{S}_K$, where $|\mathbb{S}_K| = K$.

1. *Initialization:* The parties agree on a retrieval finite field $\mathbb{F}_L$ to carry out the

calculations needed for MP-PSI determination protocol. L is chosen such that,

$$L = \min \{L \geq M : L \text{ is a prime}\} \tag{4.37}$$

The parties agree on a leader P_{t^*} such that:

$$t^* = \arg \min_{t \in \{1,\cdots,M\}} \sum_{i \neq t} \left\lceil \frac{|\mathcal{P}_t| N_i}{N_i - 1} \right\rceil \tag{4.38}$$

Without loss of generality, we assume that $t^* = M$ in the sequel. Furthermore, assume that $\mathcal{P}_{t^*} = \mathcal{P}_M = \{Y_1, Y_2, \cdots, Y_R\}$ with cardinality $|\mathcal{P}_M| = R$.

2. *Query generation:* The leader party P_M independently and uniformly generates κ random vectors $\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_\kappa\}$, where κ is given by,

$$\kappa = \max_{i \in \{1,\cdots,M-1\}} \left\lceil \frac{|\mathcal{P}_M|}{N_i - 1} \right\rceil \tag{4.39}$$

The vector $\mathbf{h}_\ell$, for $\ell = 1, 2, \cdots, \kappa$ is picked uniformly from $\mathbf{F}_L^K$ such that,

$$\mathbf{h}_\ell = [h_\ell(1) \quad h_\ell(2) \quad \cdots \quad h_\ell(K)] \tag{4.40}$$

Denote $\eta_i = \left\lceil \frac{|\mathcal{P}_M|}{N_i - 1} \right\rceil$, and let $\mathcal{P}_M^{\ell_i} = \{Y_1^{\ell_i}, Y_2^{\ell_i}, \cdots Y_{N_i-1}^{\ell_i}\}$, for $i = 1, \cdots, M-1$. The leader party P_M submits η_i random vectors from $\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_\kappa\}$ to the first database of the ith client party as queries. Each submitted random vector can be reused in the remaining $N_i - 1$ databases to retrieve $N_i - 1$ symbols. This can be done by adding 1 to the positions corresponding to the desired

symbols. More specifically, take ℓ_i to be a running index, i.e., $\ell_i = 1, 2, \cdots, \eta_i$, and assume that $\mathcal{P}_M = \cup_{\ell_i=1}^{\eta_i} \mathcal{P}_M^{\ell_i}$, where $\mathcal{P}_M^{\ell_i} \subseteq \mathcal{P}_M$ are disjoint partitions of $\mathcal{P}_M$ such that $|\mathcal{P}_M^{\ell_i}| = N_i - 1$ (except potentially for the last subset $\mathcal{P}_M^{\eta_i}$), then for $i = 1, 2, \cdots, M-1$, the query structure is given by:

$$Q_{i,1}^{[\mathcal{P}_M^{\ell_1}]} = [h_1(1)\ h_1(2)\ \cdots\ h_1(K)] \tag{4.41}$$

$$Q_{i,2}^{[\mathcal{P}_M^{\ell_1}]} = [h_1(1)\ \cdots\ h_1(Y_1^{\ell_1} - 1)\ h_1(Y_1^{\ell_1}) + 1\ \ h_1(Y_1^{\ell_1} + 1)\ \cdots\ h_1(K)] \tag{4.42}$$

$$\vdots$$

$$Q_{i,N_i}^{[\mathcal{P}_M^{\ell_1}]} = [h_1(1)\ \cdots\ h_1(Y_{N_i-1}^{\ell_1} - 1)\ h_1(Y_{N_i-1}^{\ell_1}) + 1\ \ h_1(Y_{N_i-1}^{\ell_1} + 1)\ \cdots\ h_1(K)] \tag{4.43}$$

$$\vdots$$

$$Q_{i,1}^{[\mathcal{P}_M^{\eta_i}]} = [h_{\eta_i}(1)\ h_{\eta_i}(2)\ \cdots\ h_{\eta_i}(K)] \tag{4.44}$$

$$Q_{i,2}^{[\mathcal{P}_M^{\eta_i}]} = [h_{\eta_i}(1)\ \cdots\ h_{\eta_i}(Y_1^{\eta_i} - 1)\ h_{\eta_i}(Y_1^{\eta_i}) + 1\ \ h_{\eta_i}(Y_1^{\eta_i} + 1)\ \cdots\ h_{\eta_i}(K)] \tag{4.45}$$

$$\vdots$$

$$Q_{i,N_i}^{[\mathcal{P}_M^{\eta_i}]} = [h_{\eta_i}(1)\ \cdots\ h_{\eta_i}(Y_{N_i-1}^{\eta_i} - 1)\ h_{\eta_i}(Y_{N_i-1}^{\eta_i}) + 1\ \ h_{\eta_i}(Y_{N_i-1}^{\eta_i} + 1)\ \cdots\ h_{\eta_i}(K)] \tag{4.46}$$

i.e., P_M simply partitions the set $\mathcal{P}_M$ into subsets of size $N_i - 1$. For each set, P_M uses different $\mathbf{h}_\ell$. P_M submits $\mathbf{h}_\ell$ into the first database. For the remaining databases, it adds 1 for the positions that corresponds to the partition.

3. *Common randomness generation:* In order to respond to the leader party, the

clients need to generate and share common randomness. Specifically, there are three types of randomness:

- *Local randomness:* This is denoted by $\mathbf{s}_i = [s_i(1)\ \ s_i(2)\ \ s_i(\eta_i)]$. Each element of $\mathbf{s}_i$ is generated independently and uniformly from $\mathbb{F}_L$. The local randomness $\mathbf{s}_i$ is shared between the databases associated with P_i. The local randomness is added to the responses as in SPIR [8]. Note that each database uses a different element from $\mathbf{s}_i$ for each submitted query.

- *Individual correlated randomness:* The jth database associated with the ith client possesses an individual randomness $\mathbf{t}_{i,j} = [t_{i,j}(1)\ t_{i,j}(2)\ t_{i,j}(\eta_i)]$ for $i = 1, \cdots, M-1$, and $j = 1, \cdots, N_i$. The elements $t_{i,1} = 0$ for all i. For $i = 1, \cdots, M-2$, the vector $\mathbf{t}_{i,j}$ is independently and uniformly picked from $\mathbf{F}_L^{\eta_i}$. All these random vectors are sent to the party P_{M-1}. The client P_{M-1} generates its individual randomness $\mathbf{t}_{M-1,j}$ according to the received individual randomness from the remaining parties. For simplicity, let us (re)denote the individual randomness components by $\tilde{t}_{i,k}$, where i is the index of the client party and $k = 1, 2, \cdots, R$ is just a monotonically increasing index of the randomness component used within the databases 2 to N_i of the ith client. Thus,

$$\tilde{t}_{i,1} = t_{i,2}(1), \quad \tilde{t}_{i,1} = t_{i,2}(2), \cdots, \tilde{t}_{i,R} = t_{i,N_i}(\eta_i) \tag{4.47}$$

  With this re-definition, the client P_{M-1} calculates its individual random-

ness as,

$$\tilde{t}_{M-1,j} = L - (M-1) - \sum_{i=1}^{M-2} \tilde{t}_{i,j}, \quad j = 1, 2, \cdots, R \tag{4.48}$$

This ensures that the individual randomness are correlated such that $\sum_{i=1}^{M-1} \tilde{t}_{i,j} = L - (M-1)$. The individual randomness is added to the responses.

- *Global randomness:* This is denoted by c. c is picked uniformly and independently from $\mathbb{F}_L \setminus \{0\}$. c is shared among all the databases at all clients. c is used as a multiplier for the answering string.

4. *Response generation:* The clients respond to the submitted queries by using the queries as a combining vector to their contents, i.e., each database calculates the inner product of the query and its contents. Next, it adds the local and individual randomness. Finally, it multiplies the result by the global randomness. More specifically, the answer string of the jth database, which is associated with the ith client to retrieve one of the elements of the partition $\mathcal{P}_M^{\ell_i}$, $A_{i,j}^{[\mathcal{P}_M^{\ell_i}]}$, is given by,

$$A_{i,j}^{[\mathcal{P}_M^{\ell_i}]} = c\left(X_i^T Q_{i,j}^{[\mathcal{P}_M^{\ell_i}]} + s_i(\ell_i) + t_{i,j}(\ell_i)\right) \tag{4.49}$$

From the collected answers the leader party can determine the intersection $\cap_{i=1}^{M} \mathcal{P}_i$ reliably and privately.

4.5.2 Download Cost, Reliability, Leader's Privacy, Clients' Privacy

Download cost: By observing the queries associated with the MP-PSI scheme in the previous section, one can note that the desired symbols are divided into $\eta_i = \left\lceil \frac{|\mathcal{P}_M|}{N_i - 1} \right\rceil$ subsets. Each subset consists of $N_i - 1$ desired symbols. The leader needs to download 1 bit from all N_i databases to query the entire subset, as the leader downloads useless random linear combination of the contents from the first database. Hence, the download cost is given by,

$$D = \sum_{i=1}^{M-1} N_i \eta_i \tag{4.50}$$

$$= \sum_{i=1}^{M-1} \left\lceil \frac{|\mathcal{P}_M| N_i}{N_i - 1} \right\rceil \tag{4.51}$$

Reliability: To verify reliability, we follow the leader's processing of the responses. First, we note that the answer string that is returned from database 1 is a random linear combination of the contents of the database besides the common randomness, and is given by,

$$A_{i,1}^{[\mathcal{P}_M^{\ell_i}]} = c \left(\sum_{k=1}^{K} h_{\ell_i}(k) X_{i,k} + s_i(\ell_i) \right), \quad i = 1, \cdots, M-1 \tag{4.52}$$

Note that $t_{i,1} = 0$ by construction. The leader subtracts this response from each response that belongs to the same partition. Denote the subtraction result at the

ith client that contains the element $X_{i,k}$ by $Z_{i,k}$, hence,

$$Z_{i,k} = c(X_{i,k} + \tilde{t}_{i,k}) = A_{i,j^*}^{[\mathcal{P}_M^{\ell_i}]} - A_{i,1}^{[\mathcal{P}_M^{\ell_i}]}, \quad k \in \mathcal{P}_M^{\ell_i} \tag{4.53}$$

for some unique j^* that $A_{i,j^*}^{[\mathcal{P}_M^{\ell_i}]}$ is a response of the query that adds 1 to the kth position of the query vector. In particular, for the special case of $N_i = |\mathcal{P}_i| + 1$ for all $i = 1, \cdots, M-1$, we have $j^* = k+1$ and $\mathcal{P}_M^{\ell_i} = \mathcal{P}_M$ (one partition). Note that we used the alternative notation $\tilde{t}_{i,k}$ as it is counted in sequence.

Next, the leader constructs the intersection indicator variable E_k, where E_k is given by,

$$E_k = \sum_{i=1}^{M-1} Z_{i,k} \tag{4.54}$$

$$= c\left(\sum_{i=1}^{M-1} X_{i,k} + \sum_{i=1}^{M-1} \tilde{t}_{i,k}\right) \tag{4.55}$$

$$= c\left(\sum_{i=1}^{M-1} X_{i,k} + L - (M-1)\right) \tag{4.56}$$

where (4.56) follows from the construction of the individual randomness. Now, the element $E_k = 0$ if and only if $\sum_{i=1}^{M-1} X_{i,k} = M-1$, which implies that $X_{i,k} = 1$ for all $i = 1, 2, \cdots, M-1$. Consequently, $Y_k \in \cap_{i=1}^{M} \mathcal{P}_i$ if and only if $E_k = 0$. This proves the reliability of the scheme.

Leader's privacy: The leader's privacy follows from the fact that the random vectors $\{\mathbf{h}_1, \cdots, \mathbf{h}_\kappa\}$ are uniformly generated over $\mathbb{F}_L^K$. Adding 1 to these vectors does not change the statistical distribution of the vector. Since the leader submits

independent vectors each time it queries a database, all queries are equally likely and the leader's privacy is preserved.

Clients' privacy: Without loss of generality, we derive the proof of the client's privacy for the homogeneous number of databases, i.e., $N_i = R+1, \forall i \in [1 : M-1]$. The general proof in the heterogeneous case follows the same steps and after removing the response of the first databases, we will be left with $Z_{i,k}$ that has the same structure of homogeneous case. Consequently, we present the homogeneous case here for convenience only. In the following proof, we adopt the notation that for a random variable $\zeta_{i,j}$ indexed by two indices (i, j),

$$\zeta_{i_1:i_M,j_1:j_R} = \{\zeta_{i,j} : i \in \{i_1, \cdots, i_M\},\ j \in \{j_1, \cdots, j_R\}\} \tag{4.57}$$

For the proof, we need the following lemmas. Lemma 4.1 shows that the effect of the local randomness is to make the response of the first database at all parties independent of $X_{\bar{\mathcal{P}}}$.

Lemma 4.1 *For the presented achievable scheme, we have,*

$$I(X_{\bar{\mathcal{P}}}; A_{1:M-1,1}^{[\mathcal{P}_M]} | Z_{1:M-1,Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) = 0 \tag{4.58}$$

Proof: Intuitively, the proof follows from the fact that $A_{i,1}^{[\mathcal{P}_M]}, i \in [1 : M-1]$ is a random variable uniformly distributed over $[0 : L-1]$ because of the local randomness s_i, and thus, is independent of the data sets, queries and the subtraction

results. More specifically,

$$
\begin{aligned}
&I(X_{\bar{\mathcal{P}}}; A^{[\mathcal{P}_M]}_{1:M-1,1}|Z_{1:M-1,Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
&\quad = H(A^{[\mathcal{P}_M]}_{1:M-1,1}|Z_{1:M-1,Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
&\qquad - H(A^{[\mathcal{P}_M]}_{1:M-1,1}|X_{\bar{\mathcal{P}}}, Z_{1:M-1,Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \qquad (4.59) \\
&\quad \leq H(A^{[\mathcal{P}_M]}_{1:M-1,1}) - H(A^{[\mathcal{P}_M]}_{1:M-1,1}|X_{1:M-1}, c, X_{\bar{\mathcal{P}}}, Z_{1:M-1,Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
& \qquad (4.60) \\
&\quad \leq (M-1) - H(s_1, \cdots, s_{M-1}) \qquad (4.61) \\
&\quad = (M-1) - (M-1) = 0 \qquad (4.62)
\end{aligned}
$$

This concludes the proof, since $I(X_{\bar{\mathcal{P}}}; A^{[\mathcal{P}_M]}_{1:M-1,1}|Z_{1:M-1,Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \geq 0$. ■

Lemma 4.2 asserts that for $i \in [1 : M-2]$, $j \in [1 : R]$ the effect of individual randomness $t_{i,j+1}$ is to force the random variables Z_{i,Y_j} to be independent of $X_{\bar{\mathcal{P}}}$. Note that we do not claim anything about Z_{M-1,Y_j} as the individual randomness are correlated at party $M-1$.

Lemma 4.2 *For the presented scheme, we have,*

$$I(X_{\bar{\mathcal{P}}}; Z_{1:M-2,Y_1:Y_R}|E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) = 0 \qquad (4.63)$$

Proof: Intuitively, similar to the proof of Lemma 4.1, the proof follows from the fact that $Z_{i,Y_j}, i \in [1 : M-2], j \in [1 : R]$ is a random variable uniformly distributed over $[0 : L-1]$ because of the individual randomness $t_{i,j+1}$, and thus, is independent

of the data sets, queries, and the data sets in the client parties E_{Y_j},

$$
\begin{aligned}
&I(X_{\bar{\mathcal{P}}}; Z_{1:M-2,Y_1:Y_R} | E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
&\quad = H(Z_{1:M-2,Y_1:Y_R} | E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
&\qquad - H(Z_{1:M-2,Y_1:Y_R} | X_{\bar{\mathcal{P}}}, E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \qquad (4.64) \\
&\quad \leq H(Z_{1:M-2,Y_1:Y_R}) - H(Z_{1:M-2,Y_1:Y_R} | X_{1:M-1}, c, X_{\bar{\mathcal{P}}}, E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
& \qquad (4.65) \\
&\quad \leq ((M-2)R) - H(t_{1:M-2,Y_1:Y_R}) \qquad (4.66) \\
&\quad = ((M-2)R) - ((M-2)R) = 0 \qquad (4.67)
\end{aligned}
$$

This concludes the proof as the reverse implication is true by the non-negativity of mutual information. ■

The following lemma asserts that indicator functions E_{Y_j} for all j do not leak any information about $X_{\bar{\mathcal{P}}}$.

Lemma 4.3 *For the presented scheme, we have,*

$$
I(X_{\bar{\mathcal{P}}}; E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) = 0 \qquad (4.68)
$$

Proof: Note that if $Y_j \in \mathcal{P}_M$ is in the intersection, $E_{Y_j} = 0$ has nothing to do with $X_{\bar{\mathcal{P}}}$ since $X_{\bar{\mathcal{P}}}$ is defined on the elements not in the intersection. However, if Y_j is not in the intersection, $E_{Y_j} = c(X_{1,Y_j} + \cdots + X_{M-1,Y_j} + L - (M-1)), Y_j \in \mathcal{P}_M \cap \bar{\mathcal{P}}$ received by the leader party would be a realization within the range of $\mathbb{F}_L \setminus \{0\}$ because

of the global randomness c. However, the leader party only knows that the global randomness c is uniformly distributed over $\mathbb{F}_L \backslash \{0\}$ and has no information about the specific value of c in the client parties. As a result, from the perspective of the leader part P_M, $X_{1,Y_j} + \cdots + X_{M-1,Y_j} + L - (M-1)$ is uniformly distributed over $[1 : L-1]$ according to the information contained in E_{Y_j}. This comes from the fact that the set $\mathbb{F}_L \setminus \{0\}$ of all $L-1$ non-zero elements must form a finite cyclic group under multiplication given a finite field $\mathbb{F}_L$. That means that, in the additive table under multiplication operation, each element in $\mathbb{F}_L \setminus \{0\}$ appears precisely once in each row and column of the table. The probability $\mathbb{P}(X_{1,Y_j} + \cdots + X_{M-1,Y_j} + L - (M-1) = l)$ would always be $\frac{1}{L-1}$ for any $l \in [1 : L-1]$. Then, $X_{1,Y_j} + \cdots + X_{M-1,Y_j}$ is uniformly distributed over $[M-L : M-2]$ (i.e., $[0 : M-2] \cup [M : L-1]$) and we can further conclude that $X_{1,Y_j} + \cdots + X_{M-1,Y_j}$ is uniformly distributed over $[0 : M-2]$ because its largest possible value is $M-2$ if Y_j is not in the intersection. Thus, the only information we can learn from $E_{Y_1}, \cdots, E_{Y_R}$ and the accompanying queries about $X_{\bar{\mathcal{P}}}$ is $X_{1,k} + \cdots + X_{M-1,k} < M-1, \forall k \in \mathcal{P}_M \cap \bar{\mathcal{P}}$ without knowing the specific value of $X_{1,k} + \cdots + X_{M-1,k}$, which already exists in the definition of $X_{\bar{\mathcal{P}}}$. Thus, we obtain,

$$
\begin{aligned}
&I(X_{\bar{\mathcal{P}}}; E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) \\
&\quad = I(X_{\bar{\mathcal{P}}}; E_{Y_1:Y_R} | Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) && (4.69) \\
&\quad = H(X_{\bar{\mathcal{P}}} | Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) - H(X_{\bar{\mathcal{P}}} | E_{Y_1:Y_R}, Q^{[\mathcal{P}_M]}_{1:M-1,1:N_i}, \mathcal{P}_M) && (4.70) \\
&\quad = H(X_{\bar{\mathcal{P}}}) - H(X_{\bar{\mathcal{P}}}) && (4.71)
\end{aligned}
$$

$$= 0 \tag{4.72}$$

where (4.69) follows from the fact that queries and $\mathcal{P}_M$ are independent of the data sets in the client parties E_{Y_j} in (4.2). ■

Now, we are ready to show that our achievability satisfies the client's privacy constraint,

$$\begin{aligned}
&I(X_{\bar{\mathcal{P}}}; Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) \\
&\quad = I(X_{\bar{\mathcal{P}}}; A_{1:M-1,1}^{[\mathcal{P}_M]}, Z_{1:M-1,Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) && (4.73) \\
&\quad = I(X_{\bar{\mathcal{P}}}; Z_{1:M-1,Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) \\
&\qquad + I(X_{\bar{\mathcal{P}}}; A_{1:M-1,1}^{[\mathcal{P}_M]} | Z_{1:M-1,Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) && (4.74) \\
&\quad = I(X_{\bar{\mathcal{P}}}; Z_{1:M-1,Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) && (4.75) \\
&\quad = I(X_{\bar{\mathcal{P}}}; Z_{1:M-2,Y_1:Y_R}, E_{Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) && (4.76) \\
&\quad = I(X_{\bar{\mathcal{P}}}; E_{Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) + I(X_{\bar{\mathcal{P}}}; Z_{1:M-2,Y_1:Y_R} | E_{Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) \\
& && (4.77) \\
&\quad = I(X_{\bar{\mathcal{P}}}; E_{Y_1:Y_R}, Q_{1:M-1,1:N_i}^{[\mathcal{P}_M]}, \mathcal{P}_M) && (4.78) \\
&\quad = 0 && (4.79)
\end{aligned}$$

where (4.73) follows from the fact that there is a bijective transformation between $A_{1:M-1,1:N_i}^{[\mathcal{P}_M]}$ and $(A_{1:M-1,1}^{[\mathcal{P}_M]}, Z_{1:M-1,Y_1:Y_R})$, (4.75) follows from Lemma 4.1, (4.76) follows from the fact that there is a bijective transformation between $Z_{1:M-1,Y_1:Y_R}$ and $(Z_{1:M-2,Y_1:Y_R}, E_{Y_1:Y_R})$, (4.78) follows from Lemma 4.2, and (4.79) follows from

Lemma 4.3.

4.6 Further Examples

In this section, we present two examples of our achievable scheme. Unlike the motivating example in Section 4.4, in these examples, the number of databases per party does not need to be $N_i = |\mathcal{P}_M| + 1$ or even be homogeneous in general[6].

4.6.1 An Example for $N_i < |\mathcal{P}_M| + 1$

In this example, we use the same setting of Section 4.4 with $\mathcal{P}_1 = \{1, 2\}$, $\mathcal{P}_2 = \{1, 3\}$, and $\mathcal{P}_3 = \{1, 4\}$ with P_3 being the leader party and the retrieval field being $\mathbb{F}_3$. The incidence vectors X_i, for $i = 1, 2$ remain the same. However, to illustrate that our scheme works for $N_i < |\mathcal{P}_M| + 1$, we assume that $N_1 = N_2 = 2$. As we will show next, when $N_i < |\mathcal{P}_M| + 1$, we need to send $\kappa = \eta_i = \left\lceil \frac{|\mathcal{P}_M|}{N_i - 1} \right\rceil = 2$ queries to the first database of the ith party (in contrast to 1 query only when $N_i \geq |\mathcal{P}_M| + 1$). Moreover, the common randomness components $\mathbf{s}_i$, and $\mathbf{t}_{i,j}$ need to be vectors of size $\left\lceil \frac{|\mathcal{P}_M|}{N_i - 1} \right\rceil = 2$. Note that, in this case, the leader's set is divided into 2 subsets $\mathcal{P}_M^{\ell_1} = \{1\}$ and $\mathcal{P}_M^{\ell_1} = \{4\}$ as $|\mathcal{P}_M^{\ell_i}| = N_i - 1 = 1$.

For the queries, since both client parties have the same number of databases, the leader P_3 submits the same query vectors to the databases of both clients. The first databases of each client receives 2 uniformly generated vectors $\mathbf{h}, \bar{\mathbf{h}} \in \mathbb{F}_3^4$, where $\mathbf{h} = [h_1\ h_2\ h_3\ h_4]^T$ and $\bar{\mathbf{h}} = [\bar{h}_1\ \bar{h}_2\ \bar{h}_3\ \bar{h}_4]^T$. P_3 submits the same two vectors to

[6]For $N_i > |\mathcal{P}_M| + 1$, we just use any arbitrary $|\mathcal{P}_M| + 1$ databases to execute the MP-PSI determination protocol.

the second databases of P_1 and P_2 with adding 1 to the desired positions. More specifically, let $Q_{i,j}^{[k]}$ be the query to the jth database of P_i to retrieve the element k, then P_3 submits the following queries:

$$Q_{1,1}^{[1]} = Q_{2,1}^{[1]} = [h_1 \;\; h_2 \;\; h_3 \;\; h_4]^T \tag{4.80}$$

$$Q_{1,2}^{[1]} = Q_{2,2}^{[1]} = [h_1 + 1 \;\; h_2 \;\; h_3 \;\; h_4]^T \tag{4.81}$$

$$Q_{1,1}^{[4]} = Q_{2,1}^{[4]} = [\bar{h}_1 \;\; \bar{h}_2 \;\; \bar{h}_3 \;\; \bar{h}_4]^T \tag{4.82}$$

$$Q_{1,2}^{[4]} = Q_{2,2}^{[4]} = [\bar{h}_1 \;\; \bar{h}_2 \;\; \bar{h}_3 \;\; \bar{h}_4 + 1]^T \tag{4.83}$$

At the clients' side, the clients share a global randomness $c \sim \text{uniform}\{1, 2\}$ among all the databases of both clients. For $i = 1, 2$, the ith client generates and shares a local randomness $\mathbf{s}_i = [s_i(1) \;\; s_i(2)]^T$, such that $s_i(\ell) \sim \text{uniform}\{0, 1, 2\}$ among the databases that belong to the ith client. Finally, for $i = 1, 2$, the second database of the ith client has an individual correlated randomness $\mathbf{t}_{i,2} = [t_{i,2}(1) \;\; t_{i,2}(2)]^T$, such that $t_{1,2}(1) \sim t_{1,2}(2) \sim \text{uniform}\{0, 1, 2\}$, $t_{1,2}(1) + t_{2,2}(1) = 1$, and $t_{1,2}(2) + t_{2,2}(2) = 1$. Assume that $t_{1,1} = t_{2,1} = 0$. All randomness components are independently generated of each other and of the data sets.

The answer string $A_{i,j}^{[k]}$, for $i = 1, 2$, $j = 1, 2$, $k = 1, 4$, is given by,

$$A_{i,j}^{[k]} = c \left(X_i^T Q_{i,j}^{[k]} + s_i(\ell(k)) + t_{i,j}(\ell(k)) \right) \tag{4.84}$$

where $\ell(1) = 1$ and $\ell(4) = 2$.

Thus, the leader party receives the following answer strings from P_1,

$$A_{1,1}^{[1]} = c\left(\sum_{k=1}^{4} h_k X_{1,k} + s_1(1)\right) \tag{4.85}$$

$$A_{1,2}^{[1]} = c\left(\sum_{k=1}^{4} h_k X_{1,k} + X_{1,1} + s_1(1) + t_{1,2}(1)\right) \tag{4.86}$$

$$A_{1,1}^{[4]} = c\left(\sum_{k=1}^{4} \bar{h}_k X_{1,k} + s_1(2)\right) \tag{4.87}$$

$$A_{1,2}^{[4]} = c\left(\sum_{k=1}^{4} \bar{h}_k X_{1,k} + X_{1,4} + s_1(2) + t_{1,2}(2)\right) \tag{4.88}$$

and the following answer strings from P_2,

$$A_{2,1}^{[1]} = c\left(\sum_{k=1}^{4} h_k X_{2,k} + s_2(1)\right) \tag{4.89}$$

$$A_{2,2}^{[1]} = c\left(\sum_{k=1}^{4} h_k X_{2,k} + X_{1,1} + s_2(1) + t_{2,2}(1)\right) \tag{4.90}$$

$$A_{2,1}^{[4]} = c\left(\sum_{k=1}^{4} \bar{h}_k X_{2,k} + s_2(2)\right) \tag{4.91}$$

$$A_{2,2}^{[4]} = c\left(\sum_{k=1}^{4} \bar{h}_k X_{2,k} + X_{1,4} + s_2(2) + t_{2,2}(2)\right) \tag{4.92}$$

The leader party constructs the subtractions $Z_{i,j}$ as follows,

$$Z_{1,1} = c(X_{1,1} + t_{1,2}(1)) = A_{1,2}^{[1]} - A_{1,1}^{[1]} \tag{4.93}$$

$$Z_{1,4} = c(X_{1,4} + t_{1,2}(2)) = A_{1,2}^{[4]} - A_{1,1}^{[4]} \tag{4.94}$$

$$Z_{2,1} = c(X_{1,1} + t_{2,2}(1)) = A_{2,2}^{[1]} - A_{2,1}^{[1]} \tag{4.95}$$

$$Z_{2,4} = c(X_{1,4} + t_{2,2}(2)) = A_{2,2}^{[4]} - A_{2,1}^{[4]} \tag{4.96}$$

These are exactly the statistics in (4.28)-(4.31). Hence, the reliability and privacy constraints follow exactly as in Section 4.4. The total download cost in this case is $D = 4 + 4 = 8$, which is consistent with the download cost for the general case in (4.9), $D = \left\lceil \frac{2*2}{2-1} \right\rceil + \left\lceil \frac{2*2}{2-1} \right\rceil = 8$.

4.6.2 An Example for Heterogeneous Number of Databases

In this example, we consider a general case, where there are no constraints on the number of databases associated with each party or on the cardinality of the sets. In this example, we have $M = 4$ parties with $N_1 = 2$, $N_2 = 3$, $N_3 = 5$, and $N_4 = 4$ associated databases. The four parties have the following data sets and the corresponding incidence vectors,

$$\text{Party } P_1 : \ \mathcal{P}_1 = \{1, 2, 3, 4\}, X_1 = [X_{1,1}\ X_{1,2}\ X_{1,3}\ X_{1,4}\ X_{1,5}]^T = [1\ 1\ 1\ 1\ 0]^T \quad (4.97)$$

$$\text{Party } P_2 : \ \mathcal{P}_2 = \{1, 2, 4\}, \quad X_2 = [X_{2,1}\ X_{2,2}\ X_{2,3}\ X_{2,4}\ X_{2,5}]^T = [1\ 1\ 0\ 1\ 0]^T \quad (4.98)$$

$$\text{Party } P_3 : \ \mathcal{P}_3 = \{1, 3, 4\}, \quad X_3 = [X_{3,1}\ X_{3,2}\ X_{3,3}\ X_{3,4}\ X_{3,5}]^T = [1\ 0\ 1\ 1\ 0]^T \quad (4.99)$$

$$\text{Party } P_4 : \ \mathcal{P}_4 = \{1, 4, 5\}, \quad X_4 = [X_{4,1}\ X_{4,2}\ X_{4,3}\ X_{4,4}\ X_{4,5}]^T = [1\ 0\ 0\ 1\ 1]^T \quad (4.100)$$

First, we choose party P_4 for the role of the leader party, as it results in the minimum download cost $D_t = \sum_{i \neq t} \left\lceil \frac{|\mathcal{P}_t| N_i}{N_i - 1} \right\rceil$. Since $M = 4$, we choose a retrieval field $\mathbb{F}_L$, such that $L = 5$, as L is the smallest prime number that satisfies $L \geq M$.

Now, $\kappa = \max_i \left\lceil \frac{|\mathcal{P}_4|}{N_i - 1} \right\rceil = 3$. Hence, for the queries, the leader P_4 generates $\kappa = 3$ random vectors. From which, it submits $\eta_i = \left\lceil \frac{|\mathcal{P}_4|}{N_i - 1} \right\rceil$ to the first database

associated with the ith party, $i = 1, 2, 3$. Each random vector can be reused for retrieving $N_i - 1$ elements from the remaining databases by adding 1 to the query vector in the positions of the desired symbols.

Specifically, party P_1 has only two databases and P_4 is supposed to submits $\eta_1 = \lceil \frac{3}{2-1} \rceil = 3$ random vectors to database 1, denoted by $\mathbf{h}_\ell = [h_\ell(1)\ h_\ell(2)\ \cdots\ h_\ell(5)]^T$, where $\ell = 1, 2, 3$. The leader's set is divided as $\mathcal{P}_4^{11} = \{1\}$, $\mathcal{P}_4^{12} = \{4\}$, and $\mathcal{P}_4^{13} = \{5\}$ with $|\mathcal{P}_M^{\ell_1}| = N_1 - 1 = 1$. These random vectors are generated uniformly from $\mathbb{F}_5^5$ Thus, the queries sent from P_4 to P_1 are generated as follows,

$$Q_{1,1}^{[1]} = [h_1(1) \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.101}$$

$$Q_{1,2}^{[1]} = [h_1(1) + 1 \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.102}$$

$$Q_{1,1}^{[4]} = [h_2(1) \quad h_2(2) \quad h_2(3) \quad h_2(4) \quad h_2(5)]^T \tag{4.103}$$

$$Q_{1,2}^{[4]} = [h_2(1) \quad h_2(2) \quad h_2(3) \quad h_2(4) + 1 \quad h_2(5)]^T \tag{4.104}$$

$$Q_{1,1}^{[5]} = [h_3(1) \quad h_3(2) \quad h_3(3) \quad h_3(4) \quad h_3(5)]^T \tag{4.105}$$

$$Q_{1,2}^{[5]} = [h_3(1) \quad h_3(2) \quad h_3(3) \quad h_3(4) \quad h_3(5) + 1]^T \tag{4.106}$$

Party P_2 has three databases and P_4 only needs to send $\eta_2 = \lceil \frac{4}{3-1} \rceil = 2$ random vectors to database 1 of client P_2. Each random vector can be reused at databases 2, 3 to retrieve 2 desired symbols. The leader's set is divided as $\mathcal{P}_4^{21} = \{1, 4\}$, and $\mathcal{P}_4^{22} = \{5\}$. Without loss of generality, P_4 uses $\mathbf{h}_1$ to obtain the information of $X_{2,1}, X_{2,4}$ and $\mathbf{h}_2$ is used to obtain the information of $X_{2,5}$. Note that, in this case

no query is needed to be sent to the third database to retrieve $X_{2,5}$. Thus, the queries sent from P_4 to P_2 are generated as follows,

$$Q_{2,1}^{[1,4]} = [h_1(1) \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.107}$$

$$Q_{2,2}^{[1,4]} = [h_1(1)+1 \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.108}$$

$$Q_{2,3}^{[1,4]} = [h_1(1) \quad h_1(2) \quad h_1(3) \quad h_1(4)+1 \quad h_1(5)]^T \tag{4.109}$$

$$Q_{2,1}^{[5]} = [h_2(1) \quad h_2(2) \quad h_2(3) \quad h_2(4) \quad h_2(5)]^T \tag{4.110}$$

$$Q_{2,2}^{[5]} = [h_2(1) \quad h_2(2) \quad h_2(3) \quad h_2(4) \quad h_2(5)+1]^T \tag{4.111}$$

Party P_3 has five databases and P_4 needs to send $\eta_3 = \left\lceil \frac{4}{5-1} \right\rceil = 1$ random vector to database 1 and reuse this vector to retrieve all the desired symbols from databases 2 through 4. Thus, the queries sent from P_4 to P_3 are generated as follows,

$$Q_{3,1}^{[1,4,5]} = [h_1(1) \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.112}$$

$$Q_{3,2}^{[1,4,5]} = [h_1(1)+1 \quad h_1(2) \quad h_1(3) \quad h_1(4) \quad h_1(5)]^T \tag{4.113}$$

$$Q_{3,3}^{[1,4,5]} = [h_1(1) \quad h_1(2) \quad h_1(3) \quad h_1(4)+1 \quad h_1(5)]^T \tag{4.114}$$

$$Q_{3,4}^{[1,4,5]} = [h_2(1) \quad h_2(2) \quad h_2(3) \quad h_2(4) \quad h_2(5)+1]^T \tag{4.115}$$

The clients share the following common randomness. A global randomness $c \sim$ uniform$\{1,2,3,4\}$ is shared among all databases at all clients. A local randomness $\mathbf{s}_1 = [s_1(1) \;\; s_1(2) \;\; s_1(3)]$ is shared among the databases of P_1, and similarly $\mathbf{s}_2 = [s_2(1) \; s_1(2)]$, $\mathbf{s}_3 = [s_3(1)]$ are shared among the databases of P_2 and P_3, respectively.

The random variable $s_i(\ell) \sim \text{uniform}\{0,1,2,3,4\}$. Finally, database 2 which is associated with P_1, generates the individual randomness $t_{1,2} = [t_{1,2}(1)\, t_{1,2}(2)\, t_{1,2}(3)]$. Similarly, at P_2, database 2 generates $t_{2,2} = [t_{2,2}(1)\, t_{2,2}(2)]$, and database 3 generates $t_{2,3}$. Each element of the common randomness $\mathbf{t}_{i,j}$ for $i = 1,2$ and $j = 2,3$ is generated uniformly and independently from $\mathbb{F}_5$. The variables ($\mathbf{t}_{i,j}$, $i = 1,2$, $j = 2,3$) are sent to P_3. The individual correlated randomness $t_{3,j}$ at P_3 is calculated as,

$$t_{3,2} = 2 - t_{1,2}(1) - t_{2,2}(1) \quad \Longleftrightarrow \quad t_{1,2}(1) + t_{2,2}(1) + t_{3,2} = 2 \tag{4.116}$$

$$t_{3,3} = 2 - t_{1,2}(2) - t_{2,3} \quad \Longleftrightarrow \quad t_{1,2}(2) + t_{2,3} + t_{3,3} = 2 \tag{4.117}$$

$$t_{3,4} = 2 - t_{1,2}(3) - t_{2,2}(2) \quad \Longleftrightarrow \quad t_{1,2}(3) + t_{2,2}(2) + t_{3,4} = 2 \tag{4.118}$$

According to this construction, the leader receives the following answer strings from P_1,

$$A_{1,1}^{[1]} = c\left(\sum_{k=1}^{5} h_1(k) X_{1,k} + s_1(1)\right) \tag{4.119}$$

$$A_{1,2}^{[1]} = c\left(\sum_{k=1}^{5} h_1(k) X_{1,k} + s_1(1) + X_{1,1} + t_{1,2}(1)\right) \tag{4.120}$$

$$A_{1,1}^{[4]} = c\left(\sum_{k=1}^{5} h_2(k) X_{1,k} + s_1(2)\right) \tag{4.121}$$

$$A_{1,2}^{[4]} = c\left(\sum_{k=1}^{5} h_2(k) X_{1,k} + s_1(2) + X_{1,4} + t_{1,2}(2)\right) \tag{4.122}$$

$$A_{1,1}^{[5]} = c\left(\sum_{k=1}^{5} h_3(k) X_{1,k} + s_1(3)\right) \tag{4.123}$$

$$A_{1,2}^{[5]} = c\left(\sum_{k=1}^{5} h_3(k)X_{1,k} + s_1(3) + X_{1,5} + t_{1,2}(3)\right) \tag{4.124}$$

Similarly, P_4 receives the following responses from P_2,

$$A_{2,1}^{[1,4]} = c\left(\sum_{k=1}^{5} h_1(k)X_{2,k} + s_2(1)\right) \tag{4.125}$$

$$A_{2,2}^{[1,4]} = c\left(\sum_{k=1}^{5} h_1(k)X_{2,k} + s_2(1) + X_{2,1} + t_{2,2}(1)\right) \tag{4.126}$$

$$A_{2,3}^{[1,4]} = c\left(\sum_{k=1}^{5} h_1(k)X_{2,k} + s_2(1) + X_{2,4} + t_{2,3}\right) \tag{4.127}$$

$$A_{2,1}^{[5]} = c\left(\sum_{k=1}^{5} h_2(k)X_{2,k} + s_2(2)\right) \tag{4.128}$$

$$A_{2,2}^{[5]} = c\left(\sum_{k=1}^{5} h_2(k)X_{2,k} + s_2(2) + X_{2,5} + t_{2,2}(2)\right) \tag{4.129}$$

Finally, P_4 receives the following responses from P_3,

$$A_{3,1}^{[1,4,5]} = c\left(\sum_{k=1}^{5} h_1(k)X_{3,k} + s_3(1)\right) \tag{4.130}$$

$$A_{3,2}^{[1,4,5]} = c\left(\sum_{k=1}^{5} h_1(k)X_{3,k} + s_3(1) + X_{3,1} + t_{3,2}\right) \tag{4.131}$$

$$A_{3,3}^{[1,4,5]} = c\left(\sum_{k=1}^{5} h_1(k)X_{3,k} + s_3(1) + X_{3,4} + t_{3,3}\right) \tag{4.132}$$

$$A_{3,4}^{[1,4,5]} = c\left(\sum_{k=1}^{5} h_1(k)X_{3,k} + s_3(1) + X_{3,5} + t_{3,4}\right) \tag{4.133}$$

The leader party P_4 proceeds with decoding by removing the random responses created at database 1 of all clients P_1, P_2 and P_3, i.e., it constructs $Z_{i,j}$ for $i = 1, 2, 3$

and $j = 1, 2, 3, 4, 5$ by subtracting the responses $A_{i,1}$,

$$Z_{1,1} = c(X_{1,1} + t_{1,2}(1)) = A_{1,2}^{[1]} - A_{1,1}^{[1]} \tag{4.134}$$

$$Z_{1,4} = c(X_{1,4} + t_{1,2}(2)) = A_{1,2}^{[4]} - A_{1,1}^{[4]} \tag{4.135}$$

$$Z_{1,5} = c(X_{1,5} + t_{1,2}(3)) = A_{1,2}^{[5]} - A_{1,1}^{[5]} \tag{4.136}$$

$$Z_{2,1} = c(X_{2,1} + t_{2,2}(1)) = A_{2,2}^{[1,4]} - A_{2,1}^{[1,4]} \tag{4.137}$$

$$Z_{2,4} = c(X_{2,4} + t_{2,3}) = A_{2,3}^{[1,4]} - A_{2,1}^{[1,4]} \tag{4.138}$$

$$Z_{2,5} = c(X_{2,5} + t_{2,2}(2)) = A_{2,2}^{[5]} - A_{2,1}^{[5]} \tag{4.139}$$

$$Z_{3,1} = c(X_{3,1} + t_{3,2}) = A_{3,2}^{[1,4,5]} - A_{3,1}^{[1,4,5]} \tag{4.140}$$

$$Z_{3,4} = c(X_{3,4} + t_{3,3}) = A_{3,3}^{[1,4,5]} - A_{3,1}^{[1,4,5]} \tag{4.141}$$

$$Z_{3,5} = c(X_{3,5} + t_{3,4}) = A_{3,4}^{[1,4,5]} - A_{3,1}^{[1,4,5]} \tag{4.142}$$

The MP-PSI determination at P_4 concludes by evaluating the following indicators, E_j, for $j = 1, 4, 5$ as,

$$E_1 = \sum_{i=1}^{3} Z_{i,1} = c(X_{1,1} + X_{2,1} + X_{3,1} + t_{1,2}(1) + t_{2,2}(1) + t_{3,2}) \tag{4.143}$$

$$E_4 = \sum_{i=1}^{3} Z_{i,4} = c(X_{1,4} + X_{2,4} + X_{3,4} + t_{1,2}(2) + t_{2,3} + t_{3,3}) \tag{4.144}$$

$$E_5 = \sum_{i=1}^{3} Z_{i,5} = c(X_{1,5} + X_{2,5} + X_{3,5} + t_{1,2}(3) + t_{2,2}(2) + t_{3,4}) \tag{4.145}$$

By observing that the sum of the correlated randomness in E_j according to (4.116)-(4.118) is equal 2, we note that $E_j = 0$ if and only if $\sum_{i=1}^{3} X_{i,j} = 3$, i.e., if $X_{1,j} = X_{2,j} = X_{3,j} = 1$ simultaneously. Consequently, $E_1 = E_4 = 0$ irrespective to

c, while $E_5 \neq 0$ and P_4 can reliably calculate $\cap_{i=1,2,3,4} \mathcal{P}_i = \{1, 4\}$. On the other hand, for E_5, $X_{1,5} + X_{2,5} + X_{3,5} + t_{1,4} + t_{2,4} + t_{3,4}$ is equal to 2 and then E_5 must be one value in the set $\{1, 2, 3, 4\}$ depending on the value of c. Now, we calculate the value of the expression $X_{1,5} + X_{2,5} + X_{3,5}$ from the perspective of the leader party P_4. If E_5 is 1, $\mathbb{P}(X_{1,5} + X_{2,5} + X_{3,5} = l) = \frac{1}{4}, \forall l = \{0, 1, 2, 3\}$ because c is uniformly distributed over $\{1, 2, 3, 4\}$. The conclusion is exactly the same when E_5 is equal to 2, 3 or 4. Thus, the only information that P_4 can obtain for the element 5 is that client parties P_1 , P_2 and P_3 cannot contain it at the same time. The privacy of leader party is preserved because each element in the queries is uniformly distributed over the finite field $\mathbb{F}_5$. Hence, no information about P_4 is leaked from the queries. The total download cost in this case is $D = 6 + 5 + 4 = 15$, which is consistent with the download cost for the general case in (4.9), $D = \lceil \frac{3*2}{2-1} \rceil + \lceil \frac{3*3}{3-1} \rceil + \lceil \frac{3*5}{5-1} \rceil = 15$.

4.7 Conclusion

In this chapter, we formulated the problem of MP-PSI from an information-theoretic point of view. We investigated a specific mode of communication, namely, single round communication between the leader and clients. We proposed a novel achievable scheme for the MP-PSI problem. Our scheme hinges on a careful design and sharing of randomness between client parties prior to commencing the MP-PSI operation. Our scheme is not a straightforward extension to the two-party PSI scheme, as applying the two-party PSI scheme $M - 1$ times leaks information beyond the intersection $\cap_{i=1}^{M} \mathcal{P}_i$. The download cost of our scheme matches the sum of download

cost of pair-wise PSI despite the stringent privacy constraint in the case of MP-PSI. We note that this work provides only an achievable scheme with no claim of optimality. A converse proof is needed to assess the efficiency of our scheme. Furthermore, several interesting directions can be pursued based on this work. First, one can investigate the MP-PSI in more general communication settings (not necessarily leader-to-clients). Second, one can study the case where the communication between the parties is done over multiple rounds (in contrast to the single round of communication in this work). Third, one can investigate the case of calculating more general set functions (not necessarily the intersection).

CHAPTER 5

Communication Cost of Two-Database Symmetric Private Information Retrieval: A Conditional Disclosure of Multiple Secrets Perspective

5.1 Introduction

In this chapter, since the communication cost in the PSI problem is composed of upload cost and download cost, we consider the total (upload plus download) communication cost of two-database SPIR through its relationship to CDS. In CDS, two parties each holding an individual input and sharing a common secret expect to disclose this secret to an external party in an efficient manner if and only if their inputs satisfy a public deterministic function. As a natural extension of CDS, we introduce conditional disclosure of multiple secrets (CDMS) where two parties share multiple i.i.d. common secrets rather than a single common secret as in CDS. We show that a special configuration of CDMS is equivalent to two-database SPIR. Inspired by this equivalence, given specific upload cost, we design download cost efficient SPIR schemes using bipartite graph representation of CDS and CDMS. Following this

idea, we determine the exact minimum total communication cost of two-database SPIR for $K = 3$ messages when the message length is 1.

5.2 Problem Formulation

5.2.1 Symmetric Private Information Retrieval

Following the classical SPIR problem statement in [8], we consider $N \geq 2$ non-colluding databases with each individual database storing the replicated set of $K \geq 2$ i.i.d. messages $W_{1:K}$. Moreover, L i.i.d. symbols within each message are uniformly selected from a sufficiently large finite field $\mathbb{F}_q$,

$$H(W_k) = L, \quad \forall k \tag{5.1}$$

$$H(W_{1:K}) = H(W_1) + \cdots + H(W_K) = KL \tag{5.2}$$

A random variable $\mathcal{F}$ is used to denote the randomness of the retrieval strategy selection implemented by the user. Due to the user privacy constraint, the realization of $\mathcal{F}$ is only known to the user, and unknown to any of the databases. Due to the database privacy constraint, databases need to share some amount of common randomness $\mathcal{R}$.

The message set $W_{1:K}$ stored in the databases is independent of retrieval strategy randomness $\mathcal{F}$, common randomness $\mathcal{R}$ and user's desired message index θ,

which is a random variable uniformly distributed over the set $[K]$,

$$I(W_{1:K}; \theta, \mathcal{F}, \mathcal{R}) = 0 \tag{5.3}$$

Using the desired message index, the user generates a query for each database according to $\mathcal{F}$. Hence, the queries $Q_n^{[k]}, n \in [N]$ are deterministic functions of $\mathcal{F}$,

$$H(Q_1^{[k]}, \ldots, Q_N^{[k]} | \mathcal{F}) = 0, \quad \forall k \tag{5.4}$$

After receiving a query from the user, each database should respond with a truthful answer based on the stored message set and common randomness,

$$\text{[deterministic answer]} \quad H(A_n^{[k]} | Q_n^{[k]}, W_{1:K}, \mathcal{R}) = 0, \quad \forall n, \ \forall k \tag{5.5}$$

After collecting all N answers from the databases, the user should be able to decode the desired message reliably,

$$\text{[reliability]} \quad H(W_k | \mathcal{F}, A_{1:N}^{[k]}) = 0, \quad \forall k \tag{5.6}$$

Due to the user privacy constraint, the query generated to retrieve the desired message should be statistically indistinguishable from other queries, thus, for all $k, k' \in [K], k' \neq k$,

$$\text{[user privacy]} \quad (Q_n^{[k]}, A_n^{[k]}, W_{1:K}, \mathcal{R}) \sim (Q_n^{[k']}, A_n^{[k']}, W_{1:K}, \mathcal{R}) \tag{5.7}$$

Due to the database privacy constraint, the user should learn nothing about $W_{\bar{k}}$ which is the complement of W_k, i.e., $W_{\bar{k}} = \{W_1, \cdots, W_{k-1}, W_{k+1}, \cdots, W_K\}$,

$$\text{[database privacy]} \quad I(W_{\bar{k}}; \mathcal{F}, A_{1:N}^{[k]}) = 0, \quad \forall k \tag{5.8}$$

An achievable SPIR scheme is a scheme that satisfies the reliability constraint (5.6), the user privacy constraint (5.7) and the database privacy constraint (5.8). In this chapter, we focus on the overall communication cost, which is a sum of the number of uploaded bits (named *upload cost* and denoted by U) and the number of downloaded bits (named *download cost* and denoted by D), within the retrieval scheme. As a consequence, the most efficient achievable scheme is the scheme with the lowest total communication cost, i.e., the one that achieves $C^* = \inf(U + D)$ over all achievable SPIR schemes.

5.2.2 Conditional Disclosure of a Secret

Two parties Alice and Bob possess their respective inputs X, Y and share a common secret S. Alice and Bob also share an independent randomness $\mathcal{R}$ to assist the secret disclosure of S. With the knowledge of the inputs X, Y but without knowing the common randomness $\mathcal{R}$, another party Carol wishes to learn the secret S under a specific condition by communicating with Alice and Bob simultaneously. Generally, this condition is described as a deterministic public function. Specifically, given a globally public function f, the secret S is disclosed to Carol if and only if $f(X, Y) = 1$ is true. By contrast, if $f(X, Y)$ is not equal to 1, no information about the secret S

should be revealed to Carol. To that end, Alice sends a signal A_X and Bob sends another signal B_Y to Carol.

The signals are determined by all the information contained in Alice or Bob before being sent to Carol,

$$\begin{aligned} \text{[deterministic signal]} \quad H(A_X|X,S,\mathcal{R}) &= 0 \\ H(B_Y|Y,S,\mathcal{R}) &= 0 \end{aligned} \tag{5.9}$$

If the condition is satisfied, Carol is able to decode the secret by using all the information she possesses,

$$\text{[validity]} \quad H(S|X,Y,A_X,B_Y) = 0, \quad \text{if } f(X,Y) = 1 \tag{5.10}$$

Otherwise, if the condition is not satisfied, Carol cannot learn anything about the secret based on all the information she has,

$$\text{[security]} \quad I(S;X,Y,A_X,B_Y) = 0, \quad \text{if } f(X,Y) \neq 1 \tag{5.11}$$

The information-theoretic objective of CDS is to minimize the number of bits contained in A_X and B_Y.

5.2.3 Conditional Disclosure of Multiple Secrets

Here, we introduce the concept of CDMS as an extension of CDS. Given the same setting except sharing K i.i.d. common secrets $S_1, \ldots, S_K$ in Alice and Bob, Carol expects to learn partial secrets under some specific conditions (one for each secret) by communicating with Alice and Bob simultaneously. Now, a sequence of functions $f_k, k \in [K]$ are globally public, the constraints in CDMS generalize to the following ones.

The integrated signals are determined by all the information contained in Alice or Bob before being sent to Carol,

$$\begin{aligned} \text{[deterministic signal]} \quad & H(A_X|X, S_{1:K}, \mathcal{R}) = 0 \\ & H(B_Y|Y, S_{1:K}, \mathcal{R}) = 0 \end{aligned} \tag{5.12}$$

For all $k \in [K]$, if the condition f_k is satisfied, Carol is able to decode the secret S_k,

$$\text{[validity]} \quad H(S_k|X, Y, A_X, B_Y) = 0, \quad \text{if } f_k(X, Y) = 1 \tag{5.13}$$

For all $k \in [K]$, if the condition f_k is not satisfied, Carol learns nothing about the secret S_k,

$$\text{[security]} \quad I(S_k; X, Y, A_X, B_Y) = 0, \quad \text{if } f_k(X, Y) \neq 1 \tag{5.14}$$

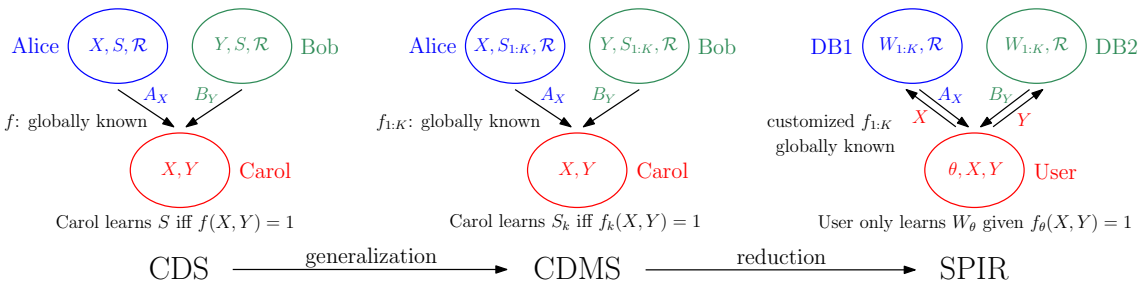


Figure 5.1: Relationship among CDS, CDMS and SPIR.

Likewise, the information-theoretic objective of CDMS is to minimize the number of bits contained in A_X and B_Y.

5.3 Main Results

We design the particular CDMS configuration given below:

1. First, Carol selects a random desired index θ, which is uniformly distributed over $[K]$; θ is independent of the secrets as well as common randomness in Alice and Bob.

2. Second, Carol selects two random vectors X and Y such that no information about θ is leaked in the individual vectors X or Y, i.e., $I(\theta; X) = 0$ and $I(\theta; Y) = 0$.

3. Third, Carol sends X to Alice and Y to Bob.

4. Globally known condition functions are set in accordance with the selection of random vectors X and Y, such that, at all times only one condition function f_θ can be 1.

Theorem 5.1 *CDMS configured as above is equivalent to SPIR with two replicated and non-colluding databases.*

Proof: Within the given configuration, Alice and Bob can be treated as database 1 and database 2, and Carol as the user; the secrets $S_{1:K}$ can be treated as the message set $W_{1:K}$; the random variable θ as the desired message index at the user; the inputs X, Y as the queries $Q_1^{[\theta]}, Q_2^{[\theta]}$; and the signals A_X, B_Y as the answers $A_1^{[\theta]}, A_2^{[\theta]}$. Thus, we have the following conversions, which complete the proof:

1. *Deterministic signal* becomes *deterministic answer*,

$$H(A_1^{[\theta]}|Q_1^{[\theta]}, W_{1:K}, \mathcal{R}) = 0 \tag{5.15}$$

$$H(A_2^{[\theta]}|Q_2^{[\theta]}, W_{1:K}, \mathcal{R}) = 0 \tag{5.16}$$

2. From the first two steps in the CDMS configuration, we obtain the *user privacy* for each database,

$$I(\theta; Q_1^{[\theta]}, A_1^{[\theta]}, W_{1:K}, \mathcal{R}) = 0 \tag{5.17}$$

$$I(\theta; Q_2^{[\theta]}, A_2^{[\theta]}, W_{1:K}, \mathcal{R}) = 0 \tag{5.18}$$

3. *Validity* becomes *reliability* due to the unique decodable secret S_θ,

$$H(W_\theta|Q_1^{[\theta]}, Q_2^{[\theta]}, A_1^{[\theta]}, A_2^{[\theta]}) = 0 \tag{5.19}$$

4. *Security* becomes *database privacy* due to the remaining undecodable secrets,

$$I(W_{\bar{\theta}}; Q_1^{[\theta]}, Q_2^{[\theta]}, A_1^{[\theta]}, A_2^{[\theta]}) = 0 \tag{5.20}$$

■

We are ready to investigate the total communication cost of two-database SPIR by means of the characteristics of CDS and CDMS. We use the terminologies in [114] for the bipartite graph in CDS/CDMS.

Remark 5.1 *We can construct an upload cost starting from* $2\log_2 K$ *in two-database SPIR while satisfying the constraints in the second step of the particular CDMS configuration above. Intuitively, the upload cost of* $2\log_2 K$ *comes from the needed* $\log_2 K$ *bits to be sent to each database to represent any one of the* K *messages. The upload cost* $2\log_2 K$ *can be achieved by the following setting:* X *and* Y *are two uniformly selected symbols from a finite set* $\mathbb{S}_K = \{0, 1, \ldots, K-1\}$ *such that* $X + Y = \theta - 1$ *under an assumption that the sum is always calculated over module* K*. In order to construct a larger upload cost, we can select a larger finite set by utilizing additional dummy messages. As an aside, we note that a larger finite set can be denoted by using multiple symbols from a smaller finite set. This further increases the diversity of upload cost constructions. For example, we can use two symbols from* $\mathbb{S}_3 = \{0, 1, 2\}$ *to include every option in* $\mathbb{S}_8 = \{0, 1, \ldots, 7\}$*. Thus, when* $K = 8$*,* X *and* Y *can either be two one-symbol vectors from* S_8 *or two two-symbol vectors from* S_3*.*

Remark 5.2 *As in CDS and CDMS, we can use a bipartite graph to specify two-database SPIR constraints. As introduced in [80, 81], CDS can be viewed as a data storage system over a bipartite graph where the nodes in each side of the graph are used to denote the input values in each party, and the connectivity of the links is used to indicate the satisfaction of the condition after selecting two nodes (input values) from two parties. In the extension to CDMS, we assign a distinct color c_k to each independent secret $S_k, \forall k \in [K]$. Hence, in CDMS, the color of links is used to indicate which secret should be revealed while keeping all the other secrets completely private. Following CDMS, in two-database SPIR, the nodes are used to denote the queries received by the databases, and the links with different colors are used to indicate which message should be retrieved while keeping all the other messages completely private, which implies reliability and database privacy.*

Remark 5.3 *In the bipartite graph, the links that are incident to any node should include all possible colors with equal number, due to user privacy.*

Example 9: In this example, we will show the use of bipartite graphs for SPIR for $N = 2$, $K = 3$ and two example upload costs of $U = 2\log_2 3$ and $U = 4$. We use colors red, yellow and green to denote messages W_1, W_2 and W_3, respectively.

For upload cost of $U = 2\log_2 3$, we use one-symbol vectors X and Y where X and Y are both uniformly selected from $\mathbb{S}_3$ s.t. $X + Y = \theta - 1$ for message W_θ. In this case, globally known condition functions are set accordingly as: $f_i(X, Y) = X + Y + 2 - i$, for $i \in [3]$. Then, we use A_0, A_1, A_2 to denote the three choices for the queries in database 1, and B_0, B_1, B_2 to denote the three choices for the queries

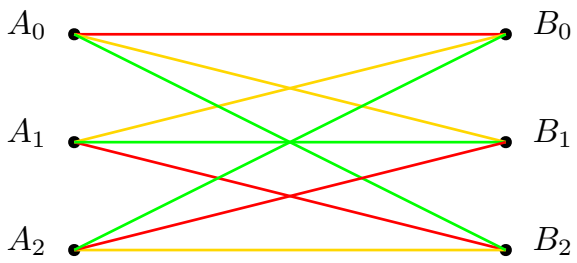


Figure 5.2: Bipartite graph for $K = 3$ messages and $U = 2\log_2 3$ upload cost.

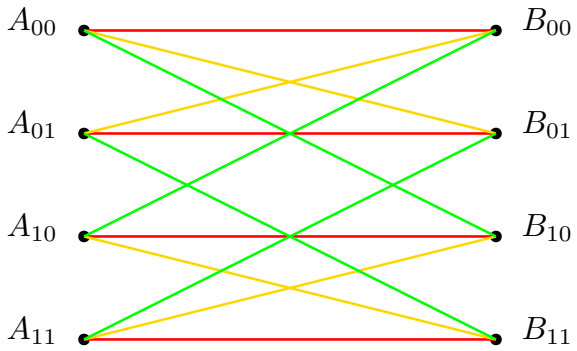


Figure 5.3: Bipartite graph for $K = 3$ messages and $U = 4$ upload cost.

in database 2. The corresponding bipartite graph is shown in Fig. 5.2.

For upload cost of $U = 4$, we use two-symbol vectors $X = \{X_2, X_1\}$ and $Y = \{Y_2, Y_1\}$ where X_1, X_2, Y_1, Y_2 are all uniformly selected from $\mathbb{S}_2$ s.t. $2(X_2 + Y_2) + (X_1 + Y_1) = \theta - 1$ for message W_θ. The setting of globally known condition functions is similar: $f_i(X, Y) = 2(X_2 + Y_2) + (X_1 + Y_1) + 2 - i$, for $i \in [3]$. Then, $A_{00}, A_{01}, A_{10}, A_{11}$ and $B_{00}, B_{01}, B_{10}, B_{11}$ are used to denote the choices for the queries in two databases. The corresponding bipartite graph is shown in Fig. 5.3.

Remark 5.4 *Given an achievable scheme for two-database SPIR with $K = P$ messages with known upload cost U and download cost D, we can construct a new achievable scheme for $K = 2P$ messages with upload cost $U + 2$ and download cost $2D$. We use the following simple example to illustrate the idea of the general construction.*

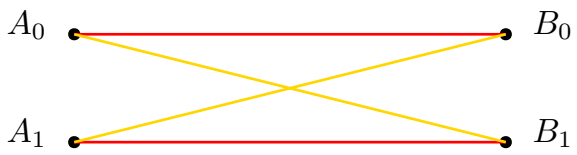


Figure 5.4: Bipartite graph for $K = 2$ messages and $U = 2$ upload cost.

Example 10: Consider two-database SPIR with $K = 4$ messages, where colors red, yellow, green, blue are assigned to messages W_1, W_2, W_3, W_4, respectively. Now, first consider a two-database SPIR with $K = 2$ messages with a special bipartite graph provided in Fig. 5.4. Following this bipartite graph, we generate an SPIR achievable scheme for $K = 2$ and $L = 1$, with $U = 2$ and $D = 2$ as follows:

$$A_0 = R_1, \qquad B_0 = W_1 + R_1 \tag{5.21}$$

$$A_1 = W_1 + W_2 + R_1, \qquad B_1 = W_2 + R_1 \tag{5.22}$$

Now, we use the bipartite graph in Fig. 5.4 as a building block to construct an SPIR scheme for $K = 4$ messages as stated in Remark 5.4. First, we replicate this bipartite graph, thus, we need to use one extra bit to describe the query choices in each database, see the left part of Fig. 5.5. Then, we replicate the whole left part, change the color of links to green and blue, and then also exchange the order of query choices in the second column, see the right part of Fig. 5.5. Combining the left part and the right part in Fig. 5.5, we can verify that this new bipartite graph is a valid one by checking Remark 5.2 and Remark 5.3. Moreover, following this bipartite graph for $K = 4$, the corresponding upload cost increases by 2 and the corresponding download cost doubles; see the following achievable scheme with

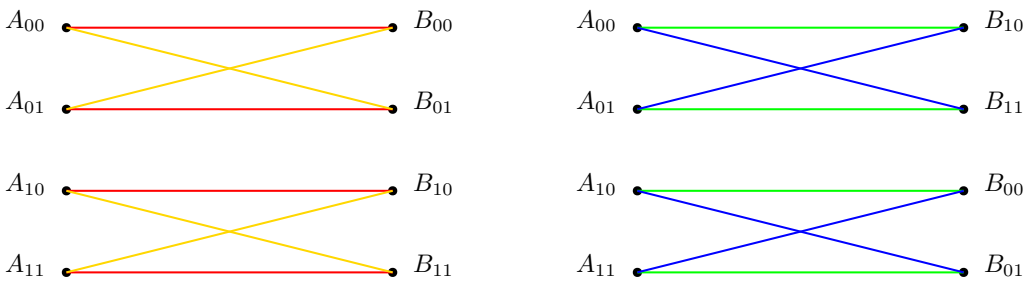


Figure 5.5: Bipartite graph for $K = 4$ messages and $U = 4$ upload cost.

$L = 1$:

$$A_{00} = \{R_1, R_3\}, \qquad B_{00} = \{W_1 + R_1, W_3 + R_4\} \tag{5.23}$$

$$A_{01} = \{W_1 + W_2 + R_1, W_3 + W_4 + R_3\}, \qquad B_{01} = \{W_2 + R_1, W_4 + R_4\} \tag{5.24}$$

$$A_{10} = \{R_2, R_4\}, \qquad B_{10} = \{W_1 + R_2, W_3 + R_3\} \tag{5.25}$$

$$A_{11} = \{W_1 + W_2 + R_2, W_3 + W_4 + R_4\}, \qquad B_{11} = \{W_2 + R_2, W_4 + R_3\} \tag{5.26}$$

5.4 Exact Upload-Download Region $N = 2, K = 3$

In this section, we give the exact achievable (U, D) cost region of two-database SPIR for $K = 3$ messages with $L = 1$ using the results of the previous section. In particular, for the upload cost of $U = 2\log_2 3$, we achieve a download cost of $D = 3$. This outperforms the best-known result of $D = 4$ in [73]. We show $(2\log_2 3, 3)$ corner point to be optimum with a converse. Further, by increasing the query selection for each database by one, we achieve a download cost of $D = 2$. This means that $U = 4$ is sufficient to achieve $D = 2$, and having $U = 6$ is not necessary as in [8]. We show

$(4, 2)$ corner point to be optimum as well with a converse.

Theorem 5.2 *In the two-database SPIR with $K = 3$ messages with message length L, when the upload cost is $U = 2\log_2 3$, the optimal download cost is $D = 3L$ and the minimal amount of required common randomness is $2L$.*

Corollary 5.1 *In the two-database SPIR with $K = 3$ messages, if the message length is confined to be $L = 1$, the optimal total communication cost is $2\log_2 3 + 3$ with minimal amount of required common randomness being 2.*

Proof: We present the converse proof first. First, we select two random nodes from the two columns. Without loss of generality, let them be A_1 and B_1, respectively. From A_1, B_1, we can recover one random message W_p without learning anything about the remaining messages $W_{\bar{p}}$. Next, we select another two nodes A_i, $i \neq 1$ and B_j, $j \neq 1$ such that W_p can be recovered from A_i, B_j once again with no knowledge about $W_{\bar{p}}$. Thus, from A_i, B_1, we can only recover another random message $W_q, q \neq p$. Then, we have,

$$
\begin{aligned}
&H(A_1|\mathcal{F}) + H(B_1|\mathcal{F}) \\
&\quad \geq H(A_1|A_i, B_1, \mathcal{F}) + H(B_1|A_i, B_j, \mathcal{F}) && (5.27)\\
&\quad = H(A_1, A_i, B_1, \mathcal{F}) + H(A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_1, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) && (5.28)\\
&\quad = H(W_p, A_1, A_i, B_1, \mathcal{F}) + H(W_p, A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_1, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \\
& && (5.29)
\end{aligned}
$$

$$\geq H(W_p, A_i, B_1, \mathcal{F}) + H(W_p, A_1, A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_1, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \tag{5.30}$$

$$= H(W_p) + H(W_p, A_1, A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \tag{5.31}$$

$$= H(W_p) + H(W_{\bar{p}}, A_1, A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \tag{5.32}$$

$$\geq H(W_p) + H(W_{\bar{p}}, A_i, B_j, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \tag{5.33}$$

$$= H(W_p) + H(W_{\bar{p}}) \tag{5.34}$$

$$= 3L \tag{5.35}$$

where (5.29) follows from the decodability of message W_p from A_1, B_1 and from A_i, B_j, (5.30) follows form the fact that conditioning cannot increase entropy, i.e., $H(A_1|W_p, A_i, B_1, \mathcal{F}) \geq H(A_1|W_p, A_i, B_1, B_j, \mathcal{F})$, (5.31) and (5.34) both come from the database privacy (5.8), (5.32) follows from the fact that we can decode $W_{1:3}$ from $A_1, B_1, A_i, B_j, \mathcal{F}$, which can be readily proved by contradiction in the bipartite graph. As a result, we reach the desired converse result regarding the download cost,

$$D \geq H(A_1) + H(B_1) \geq H(A_1|\mathcal{F}) + H(B_1|\mathcal{F}) \geq 3L \tag{5.36}$$

Next, we prove $H(\mathcal{R}) \geq 2L$:

$$0 = I(W_{\bar{p}}; A_1, B_1, \mathcal{F}) \tag{5.37}$$

$$= I(W_{\bar{p}}; A_1, B_1|W_p, \mathcal{F}) \tag{5.38}$$

$$= H(A_1, B_1|W_p, \mathcal{F}) - H(A_1, B_1|W_{1:K}, \mathcal{F}) + H(A_1, B_1|W_{1:K}, \mathcal{F}, \mathcal{R}) \tag{5.39}$$

$$= H(A_1, B_1|W_p, \mathcal{F}) - I(A_1, B_1; \mathcal{R}|W_{1:K}, \mathcal{F}) \tag{5.40}$$

$$\geq H(A_1, B_1|W_p, \mathcal{F}) - H(\mathcal{R}) \tag{5.41}$$

where (5.38) follows from the combination of (5.2) and (5.3), (5.39) follows from the deterministic answers (5.5), and (5.41) follows from (5.3) again. Therefore, we turn to find a lower bound for the expression $H(A_1, B_1|W_p, \mathcal{F})$,

$$H(A_1, B_1|W_p, \mathcal{F})$$

$$\geq H(A_1|W_p, A_i, B_1, \mathcal{F}) + H(B_1|W_p, A_i, B_j, \mathcal{F}) \tag{5.42}$$

$$= H(A_1, A_i, B_1, \mathcal{F}) + H(A_i, B_1, B_j, \mathcal{F}) - H(W_p, A_i, B_1, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) \tag{5.43}$$

$$= H(A_1, A_i, B_1, \mathcal{F}) + H(A_i, B_1, B_j, \mathcal{F}) - H(A_i, B_1, \mathcal{F}) - H(A_i, B_j, \mathcal{F}) - H(W_p) \tag{5.44}$$

$$\geq H(W_p) + H(W_{\bar{p}}) - H(W_p) \tag{5.45}$$

$$= 2L \tag{5.46}$$

where (5.45) exactly follows from the steps between (5.28)-(5.34). As a consequence, we reach the desired converse result regarding the minimal amount of required common randomness,

$$H(\mathcal{R}) \geq 2L \tag{5.47}$$

Next, we move to the achievability. We use the structure in Fig. 5.2 and the corresponding answers for $L = 1$ are as follows (we use this achievable scheme multiple times for larger L),

$$A_0 = (R_1, R_2), \qquad B_0 = W_1 + R_1 \tag{5.48}$$

$$A_1 = (W_1 + W_2 + R_1, W_2 + W_3 + R_2), \qquad B_1 = W_2 + R_2 \tag{5.49}$$

$$A_2 = (W_1 + W_3 + R_1, W_1 + W_2 + R_2), \qquad B_2 = W_3 + R_1 + R_2 \tag{5.50}$$

The achievability in (5.48)-(5.50) together with converses in (5.36) and (5.47) complete the proof. ■

Theorem 5.3 *In the two-database SPIR with $K = 3$ messages with message length L, when the upload cost is $U = 2\log_2 4 = 4$, the optimal download cost is $D = 2L$ and the minimal amount of required common randomness is L.*

Corollary 5.2 *In the two-database SPIR with $K = 3$ messages, if the message length is confined to be $L = 1$, the optimal total communication cost is $4 + 2 = 6$ with minimal amount of required common randomness being 1.*

Proof: The converse proof comes from [8, Thm. 1]. The achievability comes from the following answers corresponding to the structure in Fig. 5.3,

$$A_{00} = R_1, \qquad B_{00} = W_1 + R_1 \tag{5.51}$$

$$A_{01} = W_1 + W_2 + R_1, \qquad B_{01} = W_2 + R_1 \tag{5.52}$$

$$A_{10} = W_1 + W_3 + R_1, \qquad B_{10} = W_3 + R_1 \tag{5.53}$$

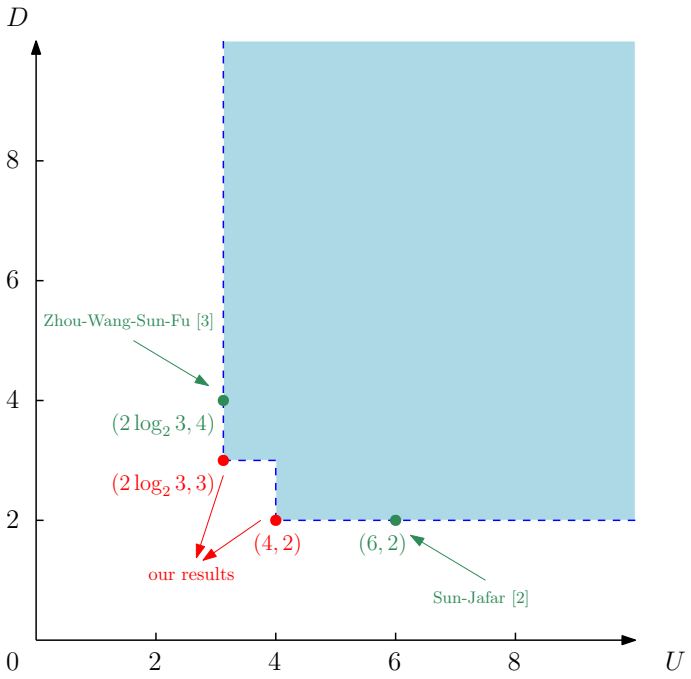


Figure 5.6: Achievable (U, D) region for two-database SPIR with $K = 3$, $L = 1$.

$$A_{11} = W_2 + W_3 + R_1, \qquad B_{11} = W_1 + W_2 + W_3 + R_1 \tag{5.54}$$

which complete the proof. ■

Combining Theorem 5.2 and Theorem 5.3, we obtain the achievable (U, D) region for two-database SPIR for $K = 3$ and $L = 1$ in Fig. 5.6. Any point within the light blue area is achievable, while all the remaining points are not achievable. Thus, the optimal communication cost is $4 + 2 = 6$.

5.5 Conclusion

In this chapter, we investigated the overall communication cost of two-database SPIR from the perspective of information theory. This is a first and leading attempt on this problem. Although the download cost of SPIR is fully understood

now, the upload cost or joint upload-download cost of SPIR is still open. To that end, we utilized CDS/CDMS to provide a potential connection in this work. Following this direction, we developed an understanding of how to construct the general upload cost for two-database SPIR, and determined a few principles of building two-database SPIR schemes from small K to large K. Finally, by providing a complete upload-download cost achievable region, we determined the exact minimum total communication cost of two-database SPIR for $K = 3$ messages when the message length is 1. In the future, we aim to extend these ideas to more general parameters. In other words, we want to find the optimal communication cost in two-database SPIR when the total number of messages K is arbitrarily large, and devise a corresponding general scheme to achieve this communication cost. We also want to consider the case with more than two databases.

CHAPTER 6

Digital Blind Box: Random Symmetric Private Information Retrieval

6.1 Introduction

In this chapter, we introduce the problem of RSPIR, which a novel and interesting extension of SPIR. Different from SPIR, the user does not have an input to the databases, i.e., the user does not select a specific message to download. As an alternative, this user is satisfied with any arbitrary message in the message set that is available to the databases. Therefore, the databases need to send symbols to the user in such a way that the user is guaranteed to download a message correctly (random reliability), the databases do not know which message the user has received (user privacy), and the user does not learn anything further than the one message it has received (database privacy). This is the digital version of a blind box, also known as gachapon, which implements the above specified setting with physical objects for entertainment. This is also the blind version of 1-out-of-K OT, an important cryptographic primitive in the field of cryptography. We study the information-theoretic capacity of two-database RSPIR and determine its exact capacity for the

cases of $K = 2, 3, 4$ messages. While we provide a general achievable scheme that is applicable to any number of messages, the capacity for $K \geq 5$ remains open.

6.2 RSPIR: Problem Formulation

In this chapter, we consider $N = 2$ non-colluding databases each storing the same set of $K \geq 2$ i.i.d. messages. Each message consists of L i.i.d. uniformly chosen symbols from a sufficiently large finite field $\mathbb{F}_q$, i.e.,

$$H(W_k) = L, \quad k \in [K] \tag{6.1}$$

$$H(W_{1:K}) = H(W_1) + \cdots + H(W_K) = KL \tag{6.2}$$

The two databases jointly share a necessary common randomness random variable $\mathcal{S}$, which is generated independent of the message set $W_{1:K}$. Thus,

$$H(W_{1:K}, \mathcal{S}) = H(W_{1:K}) + H(\mathcal{S}) \tag{6.3}$$

Before the RSPIR process starts, an answer set $\mathcal{A}$ with cardinality M_1 is assigned to database 1 while another answer set $\mathcal{B}$ with cardinality M_2 is assigned to database 2. Since there is no input at the user side in the RSPIR process, the databases will never receive a query from the user. Therefore, as a simple approach, each database individually selects a random answer under a uniform distribution from its corresponding answer set and then transmits it to the user. The indices of the answers for two databases are denoted by X and Y, respectively, i.e., database

1 will select $A_X \in \mathcal{A}$ and database 2 will select $B_Y \in \mathcal{B}$. Moreover, we use x and y to denote the realizations of the random variables X and Y, respectively. We note that every answer from any answer set is generated based on the message set and the common randomness, hence, for all $X \in [M_1]$ and $Y \in [M_2]$, we have,

$$\text{[deterministic answer]} \quad H(A_X, B_Y | X, Y, W_{1:K}, \mathcal{S}) = 0 \tag{6.4}$$

After collecting two arbitrary answers from the databases, the user should always be able to decode a random message reliably. Thus, for all $X \in [M_1]$ and $Y \in [M_2]$, we can always find an index $\theta_{X,Y} \in [K]$ (the mapping here is not deterministic) such that

$$\text{[random reliability]} \quad H(W_{\theta_{X,Y}} | X, Y, A_X, B_Y) = 0 \tag{6.5}$$

Because of the database privacy constraint, the user is supposed to learn nothing about $W_{\bar{\theta}_{X,Y}}$ which is the complement of the randomly retrieved message $W_{\theta_{X,Y}}$, i.e., $W_{\bar{\theta}_{X,Y}} = \{W_1, \ldots, W_{\theta_{X,Y}-1}, W_{\theta_{X,Y}+1}, \ldots, W_K\}$,

$$\text{[database privacy]} \quad I(W_{\bar{\theta}_{X,Y}}; X, Y, A_X, B_Y) = 0 \tag{6.6}$$

Because of the user privacy constraint, i.e., the protection of this randomly retrieved message's index in the user, from the perspective of each individual database, this index must be indistinguishable for each randomly selected answer under a uniform distribution. In other words, even though an answer from one database is

deterministic, the user can still decode every potential message in the message set with equal probability through the variation of the answer from the other database. Thus, for the first database, given any realization $x \in [M_1]$, we always have the following probability distribution of the random variable $\theta_{x,Y}$ with respect to the random variable Y,

$$P(\theta_{x,Y} = k) = \frac{1}{K}, \quad \forall k \in [K] \tag{6.7}$$

which is equivalent to

$$\text{[user privacy]} \quad I(x, A_x, W_{1:K}, \mathcal{S}; \theta_{x,Y}) = 0 \tag{6.8}$$

By symmetry, for database 2, given any realization $y \in [M_2]$, we also have the following probability distribution of the random variable $\theta_{X,y}$ with respect to the random variable X,

$$P(\theta_{X,y} = k) = \frac{1}{K}, \quad \forall k \in [K] \tag{6.9}$$

which is equivalent to

$$\text{[user privacy]} \quad I(y, B_y, W_{1:K}, \mathcal{S}; \theta_{X,y}) = 0 \tag{6.10}$$

As a consequence, we obtain the following theorem regarding the cardinality of the answer sets, which can be proved by contradiction using the user privacy

constraint.

Theorem 6.1 *The total possible number of answers in the answer set for each database must be a multiple of K, i.e.,*

$$M_1 = t_1 K, \ M_2 = t_2 K, \quad t_1, t_2 \in N^+ \tag{6.11}$$

Moreover, we also have the following theorem concerning the common randomness distribution in the databases.

Theorem 6.2 *As in multi-database SPIR [7, 8], in RSPIR, the databases must share some necessary common randomness that is unknown to the user before the retrieval process starts. Otherwise, RSPIR is not feasible.*

Proof: Without any common randomness in the databases, for any $X \in [M_1]$ and $Y \in [M_2]$, the random reliability constraint and the database privacy constraint collectively lead to,

$$0 = I(W_{\bar{\theta}_{X,Y}}; X, Y, A_X, B_Y) \tag{6.12}$$

$$= I(W_{\bar{\theta}_{X,Y}}; W_{\theta_{X,Y}}, X, Y, A_X, B_Y) \tag{6.13}$$

$$= H(W_{\bar{\theta}_{X,Y}}) - H(W_{\bar{\theta}_{X,Y}} | W_{\theta_{X,Y}}, X, Y, A_X, B_Y) \tag{6.14}$$

Then, we consider the following expression

$$I(X, Y, A_X, B_Y; W_{\bar{\theta}_{X,Y}} | W_{\theta_{X,Y}})$$

$$= H(W_{\bar{\theta}_{X,Y}}|W_{\theta_{X,Y}}) - H(W_{\bar{\theta}_{X,Y}}|W_{\theta_{X,Y}}, X, Y, A_X, B_Y) \tag{6.15}$$

$$= H(W_{\bar{\theta}_{X,Y}}) - H(W_{\bar{\theta}_{X,Y}}) \tag{6.16}$$

$$= 0 \tag{6.17}$$

where (6.16) follows from (6.14). For any realization x,

$$0 = I(A_x; W_{\bar{\theta}_{x,Y}}|x, W_{\theta_{x,Y}}) \tag{6.18}$$

$$= H(A_x|x, W_{\theta_{x,Y}}) - H(A_x|x, W_{1:K}) \tag{6.19}$$

$$= H(A_x|W_{\theta_{x,Y}}) \tag{6.20}$$

where (6.18) follows from (6.17), and (6.20) follows from the deterministic answer constraint $H(A_x|x, W_{1:K}) = 0$ without common randomness. Taking into consideration the fact that (6.20) is true for any realization $y \in [M_2]$ as well as the user privacy constraint (6.7), we have $H(A_x|W_1) = \cdots = H(A_x|W_K) = 0$. Since messages are all mutually independent, it is easy to derive that $H(A_x) = 0$, which forms a contradiction. ■

A valid two-database RSPIR achievable scheme is a scheme that satisfies the user privacy constraint (6.8), (6.10), the database privacy constraint (6.6) and the random reliability constraint (6.5).

The efficiency of a scheme is measured in terms of the total number of downloaded bits by the user from the two databases, named as the download cost. According to the formulation above, the download cost consists of the answer indices

X, Y and the answers themself A_X, B_Y. Compared with the answer cost, the answer index cost can be neglected as it does not scale with the message length if we reuse them to decode each symbol in the randomly retrieved message. Thus, we use D_{RSPIR} to denote the expected number of bits contained in the answers A_X, B_Y over the indices X, Y. Then the retrieval rate of RSPIR is given by,

$$R_{RSPIR} = \frac{L}{D_{RSPIR}} \tag{6.21}$$

The capacity of RSPIR, C_{RSPIR}, is the supremum of the retrieval rates R_{RSPIR} over all valid achievable schemes.

6.3 Main Results

Theorem 6.3 *In the two-database RSPIR problem, in the case of $K = 2$, the capacity is $\frac{1}{2}$ with minimal amount of required common randomness being L. In the case of $K = 3, 4$, the capacity is $\frac{1}{3}$ with minimal amount of required common randomness being $2L$.*

The converse proof of Theorem 6.3 is given in Section 6.4, and the achievability proof of Theorem 6.3 is presented in Section 6.5. The capacity and its minimal amount of required common randomness in the case of $K \geq 5$ is an open problem.

Remark 6.1 *It is well known [8] that the capacity of multi-database SPIR is $1 - \frac{1}{N}$, where N is the number of replicated and non-colluding databases. As a corollary, the capacity of two-database SPIR is $\frac{1}{2}$, which does not depend on the number of*

messages K stored in the databases. By contrast, the capacity of RSPIR does depend on the value of K. Even though the capacity of RSPIR achieves the same limit as SPIR in the case of $K = 2$, the capacity of RSPIR decreases to $\frac{1}{3}$ when the value of K increases to 3.

Remark 6.2 *Because of the equivalence between RSPIR and the digital blind box, in a digital blind box setting where two non-colluding databases share K messages and some necessary common randomness, perfect digital blind box delivery can be achieved with a linear download cost KL. The proof is a direct consequence of the second general achievable scheme given in Section 6.5.*

Remark 6.3 *In the problem formulation part of our previous work [75], we assume that the user is able to obtain a random subset of the shared common randomness that is unknown to any individual database before the SPIR retrieval process starts. Although we mention the idea of fetching common randomness like side-information in advance, we do not specify in [75] a corresponding practical implementation. Now, it is clear that the achievability provided here for RSPIR can be used as a practical approach for this problem if common randomness is treated as another independent message system.*

6.4 Converse Proof

Theorem 6.4 *In the two-database RSPIR problem, the capacity is realized in the case where M_1 and M_2 are both exactly K.*

Proof: We provide a sketch of proof here. The idea of the proof is that once we multiply the value of M_1 by an integer $t \geq 2$, it is straightforward to see that additional constraints will be added to each pair A_X, B_Y for all $X \in [tM_1]$ and $Y \in [M_2]$ after considering the index permutation, which will either increase or maintain the minimal value of $H(A_X) + H(B_Y)$. This analysis also applies to the increase of M_2. ■

In the case of $K = 2$, motivated by Theorem 6.4, we consider the simplest case where $M_1 = 2$ and $M_2 = 2$. Then, we only need to investigate the following constraints since all the other potential system of constraints have the same structure as this one and will lead to the same conclusions,

$$H(W_1|A_1, B_1) = 0, \quad H(W_1|A_2, B_2) = 0 \tag{6.22}$$

$$H(W_2|A_1, B_2) = 0, \quad H(W_2|A_2, B_1) = 0 \tag{6.23}$$

These constraints exactly reflect the random reliability constraint (6.5) and user privacy constraint (6.7), (6.9) involved in this problem. First, we prove a lower bound for $H(A_1) + H(B_1)$,

$$\begin{aligned} & H(A_1) + H(B_1) \\ & \geq H(A_1|A_2, B_1) + H(B_1|A_2, B_2) && (6.24) \\ & = H(A_1, A_2, B_1) + H(A_2, B_1, B_2) - H(A_2, B_1) - H(A_2, B_2) && (6.25) \\ & = H(W_1, A_1, A_2, B_1) + H(W_1, A_2, B_1, B_2) - H(A_2, B_1) - H(A_2, B_2) && (6.26) \end{aligned}$$

$$\geq H(W_1, A_2, B_1) + H(W_1, A_1, A_2, B_1, B_2) - H(A_2, B_1) - H(A_2, B_2) \tag{6.27}$$

$$= H(A_1, A_2, B_1, B_2) - H(A_2, B_2) + H(W_1) \tag{6.28}$$

$$\geq H(W_2, A_2, B_2) - H(A_2, B_2) + H(W_1) \tag{6.29}$$

$$= H(W_2) + H(W_1) \tag{6.30}$$

$$= 2L \tag{6.31}$$

where (6.28) and (6.30) follow from the database privacy constraint. Likewise, we can always obtain $H(A_X) + H(B_Y) \geq 2L$ for any other answer pair $A_X, B_Y, X, Y \in [2]$. As a result, we reach a converse result for the capacity when $K = 2$,

$$R = \frac{L}{D} \leq \frac{L}{H(A_X) + H(B_Y)} \leq \frac{L}{2L} = \frac{1}{2} \tag{6.32}$$

Next, we prove the minimal required amount of common randomness shared in the two databases.

$$0 = I(W_2; A_1, B_1) \tag{6.33}$$

$$= I(W_2; A_1, B_1 | W_1) \tag{6.34}$$

$$= H(A_1, B_1 | W_1) - H(A_1, B_1 | W_1, W_2) + H(A_1, B_1 | W_1, W_2, \mathcal{S}) \tag{6.35}$$

$$= H(A_1, B_1 | W_1) - I(A_1, B_1; \mathcal{S} | W_1, W_2) \tag{6.36}$$

$$= H(A_1, B_1 | W_1) - H(\mathcal{S} | W_1, W_2) + H(\mathcal{S} | W_1, W_2, A_1, B_1) \tag{6.37}$$

$$\geq H(A_1, B_1 | W_1) - H(\mathcal{S}) \tag{6.38}$$

where (6.35) follows from the deterministic answer constraint (6.4) and (6.38) follows from the independence between message set and the common randomness (6.3). Therefore, we turn to find a lower bound for the expression $H(A_1, B_1|W_1)$,

$$
\begin{aligned}
& H(A_1, B_1|W_1) \\
& \quad = H(A_1|W_1, B_1) + H(B_1|W_1) && (6.39) \\
& \quad \geq H(A_1|W_1, A_2, B_1) + H(B_1|W_1, A_2, B_2) && (6.40) \\
& \quad = H(A_1, A_2, B_1) + H(A_2, B_1, B_2) - H(W_1, A_2, B_1) - H(A_2, B_2) && (6.41) \\
& \quad = H(A_1, A_2, B_1) + H(A_2, B_1, B_2) - H(A_2, B_1) - H(A_2, B_2) - H(W_1) && (6.42) \\
& \quad \geq H(W_2) && (6.43) \\
& \quad = L && (6.44)
\end{aligned}
$$

where (6.42) follows from the database privacy constraint and (6.43) exactly follows from the steps between (6.25)-(6.30). As a consequence, we reach a converse result for the minimal amount of required common randomness,

$$H(\mathcal{S}) \geq L \tag{6.45}$$

In the case of $K = 3$, M_1 and M_2 both take the value 3, after converting the random reliability constraint and user privacy constraint into pairwise constraints as in (6.22)-(6.23), we can proceed with the converse steps. As in the converse proof in the case of $K = 2$ above, the concrete process is to utilize the converse proof of [115, Theorem 2] once more after eliminating the influence of retrieval strategy

randomness and its generated queries. Thus, we have the same conclusions as the one in [115, Theorem 2] in the case of $K = 3$,

$$R \leq \frac{1}{3}, \quad H(\mathcal{S}) \geq 2L \tag{6.46}$$

In the case of $K = 4$, it is easy to verify that each answer pair $A_X, B_Y, X, Y \in [4]$ has more constraints than the one when $K = 3$. Thus, a converse proof for the capacity and the minimal amount of required common randomness in the case of $K = 4$ can be inherited from the case of $K = 3$, i.e.,

$$R \leq \frac{1}{3}, \quad H(\mathcal{S}) \geq 2L \tag{6.47}$$

A tight converse proof for the capacity and the minimal amount of required common randomness remains to be found in the case of $K \geq 5$.

6.5 Achievability

The work in [73] provides a scheme that can be readily converted into an achievable scheme (albeit suboptimal) for the two-database RSPIR problem. For clarity, we restate the result from the new perspective of RSPIR here. Assuming that $L = 1$ for the time being, two databases share K common randomness symbols $S_1, \ldots, S_K$, which are all uniformly selected from $\mathbb{F}_q$. For database 1, the answer set $\mathcal{A}$ is

composed of K elements in the following form,

$$A_1 = (W_1 + S_1, W_2 + S_2, \ldots, W_K + S_K) \tag{6.48}$$

$$A_2 = (W_1 + S_2, W_2 + S_3, \ldots, W_K + S_1) \tag{6.49}$$

$$\vdots$$

$$A_K = (W_1 + S_K, W_2 + S_1, \ldots, W_K + S_{K-1}) \tag{6.50}$$

Basically, we only rotate common randomness symbols by one place in the sequence of answers. A homomorphic variation of $\mathcal{A}$ is to rotate message symbols by one place without imposing any influence on the answer set $\mathcal{B}$ and it is shown as follows,

$$A_1 = (W_1 + S_1, W_2 + S_2, \ldots, W_K + S_K) \tag{6.51}$$

$$A_2 = (W_2 + S_1, W_3 + S_2, \ldots, W_1 + S_K) \tag{6.52}$$

$$\vdots$$

$$A_K = (W_K + S_1, W_1 + S_2, \ldots, W_{K-1} + S_K) \tag{6.53}$$

For database 2, the answer set $\mathcal{B}$ also including K elements is shown as follows,

$$B_1 = S_1, \tag{6.54}$$

$$B_2 = S_2, \tag{6.55}$$

$$\vdots$$

$$B_K = S_K \tag{6.56}$$

The answer set construction in these two databases is public knowledge to the user. Afterwards, database 1 selects a random answer under a uniform distribution from $\mathcal{A}$, and then sends the values of symbols as well as the index belonging to this answer to the user. Likewise, database 2 performs the same selection and transmission. The reason for sending the answer indices is that the user does not know how to use the values of symbols in the answers to decode a random message without the help of the answer indices. After receiving two answers, the user is always able to decode one random message reliably. Moreover, since each database is doing the uniform selection, this random message is equally likely to be any message in the message set. Therefore, it is impossible for each individual database to learn the index of this randomly retrieved message at the user side. Meanwhile, the user cannot learn any information about the remaining messages because of the existence of unknown common randomness symbols. When each message includes multiple symbols, we can simply perform this scheme repeatedly for each symbol while there is no need to do the new selection nor send the answer index for each database after first execution. Thus, the download cost of answer index can be ignored as illustrated in the problem formulation when L is large enough. Obviously, the download cost is $D = (K + 1)L$ in this scheme but it is not optimal.

Here, we provide a new general scheme that achieves the download cost of $D = KL$ when L goes to infinity. Assuming that $L = 1$ temporarily, let two databases share $K - 1$ common randomness symbols $S_1, \ldots, S_{K-1}$. For database 1,

the answer set $\mathcal{A}$ contains K elements in the following form,

$$A_1 = (S_1, S_2, \ldots, S_{K-1}) \tag{6.57}$$

$$A_2 = (W_1 + W_2 + S_1, W_2 + W_3 + S_2, \ldots, W_{K-1} + W_K + S_{K-1}) \tag{6.58}$$

$$A_3 = (W_1 + W_3 + S_1, W_2 + W_4 + S_2, \ldots, W_{K-1} + W_1 + S_{K-1}) \tag{6.59}$$

$$\vdots$$

$$A_K = (W_1 + W_K + S_1, W_2 + W_1 + S_2, \ldots, W_{K-1} + W_{K-2} + S_{K-1}) \tag{6.60}$$

Basically, except for the first answer, we only rotate the second message symbol by one place in the sequence of answers while keeping the first message symbol. For database 2, the answer set $\mathcal{B}$ consists of K elements in the following form,

$$B_1 = W_1 + S_1 \tag{6.61}$$

$$B_2 = W_2 + S_2 \tag{6.62}$$

$$\vdots$$

$$B_{K-1} = W_{K-1} + S_{K-1} \tag{6.63}$$

$$B_K = W_K + S_1 + S_2 + \ldots S_{K-1} \tag{6.64}$$

Now, note that, since this scheme achieves a download cost of $D = KL$, it achieves a rate of $R = \frac{L}{D} = \frac{L}{KL} = \frac{1}{K}$. For $K = 2$ and $K = 3$, this scheme achieves rates $\frac{1}{2}$ and $\frac{1}{3}$ meeting the converse bounds in (6.32) and (6.46), respectively.

Specifically, when $K = 2$, we have the following answer sets,

$$A_1 = S_1, \qquad B_1 = W_1 + S_1 \tag{6.65}$$

$$A_2 = W_1 + W_2 + S_2, \qquad B_2 = W_2 + S_2 \tag{6.66}$$

When $K = 3$, we have the following answer sets,

$$A_1 = (S_1, S_2), \qquad B_1 = W_1 + S_1 \tag{6.67}$$

$$A_2 = (W_1 + W_2 + S_1, W_2 + W_3 + S_2), \qquad B_2 = W_2 + S_2 \tag{6.68}$$

$$A_3 = (W_1 + W_3 + S_1, W_2 + W_1 + S_2), \qquad B_3 = W_3 + S_1 + S_2 \tag{6.69}$$

When $K = 4$, this achievable scheme achieves a rate $R = \frac{1}{K} = \frac{1}{4}$ whereas the converse in (6.47) gives a bound of $\frac{1}{3}$. Now, we provide a better scheme that achieves the converse in the case of $K = 4$. The message length L is assumed to be 2 such that $W_1 = \{a_1, a_2\}, W_2 = \{b_1, b_2\}, W_3 = \{c_1, c_2\}$ and $W_4 = \{d_1, d_2\}$. Moreover, two databases share 4 common randomness symbols S_1, S_2, S_3, S_4. For database 1, the answer set $\mathcal{A}$ containing 4 elements is in the following form,

$$A_1 = (S_1, S_2, S_3) \tag{6.70}$$

$$A_2 = (a_1 + c_1 + c_2 + S_1, b_2 + d_1 + S_1 + S_3, c_2 + S_4) \tag{6.71}$$

$$A_3 = (a_1 + d_2 + S_1 + S_4, a_2 + d_1 + d_2 + S_2, b_1 + c_2 + S_2 + S_3) \tag{6.72}$$

$$A_4 = (b_1 + S_4, a_1 + a_2 + b_1 + b_2 + S_1 + S_2, c_1 + d_2 + S_1 + S_2 + S_3) \tag{6.73}$$

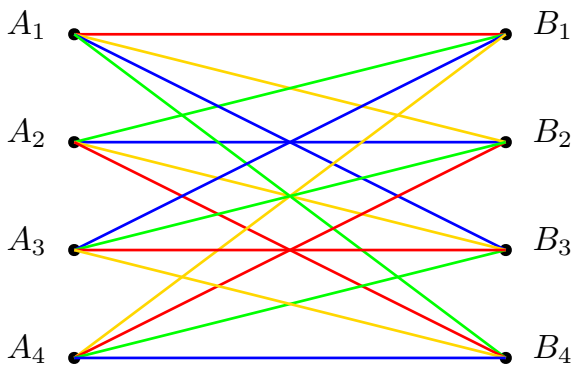


Figure 6.1: A two-database RSPIR bipartite graph for $K = 4$ messages.

For database 2, the answer set $\mathcal{B}$ with 4 elements is as follows,

$$B_1 = (a_1 + S_1, a_2 + S_2, S_4) \tag{6.74}$$

$$B_2 = (b_1 + b_2 + S_1 + S_2, b_1 + S_2 + S_3, a_1 + c_1 + d_2 + S_1 + S_4) \tag{6.75}$$

$$B_3 = (d_1 + d_2 + S_2, b_1 + c_2 + S_4, d_1 + S_1 + S_3) \tag{6.76}$$

$$B_4 = (c_2 + S_2 + S_3, c_1 + c_2 + S_1, a_1 + a_2 + b_2 + c_1 + d_1 + S_3 + S_4) \tag{6.77}$$

Here, the download cost is $D = 6$ and the rate is $R = \frac{L}{D} = \frac{1}{3}$. The remaining steps and verification of this specific achievable scheme are the same as the last two general ones. Specifically, regarding verification, we can use the bipartite graph in Fig. 6.1. In this bipartite graph, by using colors red, yellow, green and blue for messages W_1, W_2, W_3 and W_4, respectively, the color of links indicates which message should be decoded while keeping all the other messages private. Moreover, each node is always connected to 4 links with different colors.

6.6 Conclusion

In this chapter, we investigated the capacity of two-party RSPIR from the perspective of information theory. This work provides an initial Shannon-theoretic study of RSPIR. It is well known that the capacity of two-database SPIR is always one half, which is independent of the number of messages K stored in the databases. By contrast, the capacity of two-party RSPIR does depend on the value of K. In addition, because of the equivalence between RSPIR and digital blind box, we can achieve a linear download cost KL for perfect digital blind box delivery. Moreover, according to our observation in Chapter 3, an important application of RSPIR is to enable the user to fetch a random subset of the common randomness available at the databases for the sake of user-side common randomness formation that is unknown to the databases (also unknown to the user before it is received by the user). Finally, we determined the exact capacity of two-party RSPIR for the cases of $K = 2, 3, 4$. In particular, during the design of the achievable scheme in the case of $K = 4$, the message length needs to be set as 2 rather than the usual 1. In our future work, we want to find the capacity of two-database RSPIR in the case of an arbitrary number of messages K. Similarly, we also want to explore the general capacity of RSPIR when there are more databases at the server side.

CHAPTER 7

Private Federated Submodel Learning via Private Set Union

7.1 Introduction

In this chapter, we consider the FSL problem and propose an approach where clients are able to update the central model information-theoretically privately. Our approach is based on PSU, which is further based on MM-SPIR. The server has two non-colluding databases which keep the model in a replicated manner. With our scheme, the server does not get to learn anything further than the subset of submodels updated by the clients: the server does not get to know which client updated which submodel(s), or anything about the local client data. In comparison to the state-of-the-art private FSL schemes in [92, 94], our scheme does not require noisy storage of the model at the databases; and in comparison to the recent secure aggregation scheme in [101], our scheme does not require pre-distribution of client-side common randomness, instead, our scheme creates the required client-side common randomness via RSPIR and one-time pads. Our system is initialized with a replicated storage of submodels and a sufficient amount of common randomness in two databases at the server side. The protocol starts with a common randomness gen-

eration (CRG) phase where the two databases establish common randomness at the client side using RSPIR and one-time pads (this phase is called FSL-CRG). Next, the clients utilize the established client-side common randomness to have the server determine privately the union of submodel indices to be updated collectively by the clients (this phase is called FSL-PSU). Then, the two databases broadcast the current versions of the submodels in the index set union to clients. The clients update the submodels based on their local training data. Finally, the clients use a variation of FSL-PSU to write the updates back to the databases privately (this phase is called FSL-write). Since the databases at the server do not need to communicate, as a novel approach, we utilize randomly chosen alive clients to route the required information between the two databases. Our proposed private FSL scheme achieves low communication cost, and is also robust against client drop-outs, client late-arrivals, and database drop-outs.

7.2 Problem Formulation

7.2.1 MM-SPIR

As in [72], we consider $N \geq 1$ non-colluding databases with each individual database storing the replicated set of $K \geq 2$ i.i.d. messages $W_{[K]} = \{W_1, \ldots, W_K\}$. The L i.i.d. symbols within each message are uniformly selected from a sufficiently large finite field $\mathbb{F}_q$, hence,

$$H(W_k) = L, \quad \forall k \tag{7.1}$$

$$H(W_{[K]}) = H(W_1) + \cdots + H(W_K) = KL \tag{7.2}$$

The goal of the MM-SPIR problem is to retrieve a set of messages W_Ω out of the message set $W_{[K]}$ without leaking any information regarding the retrieved index set $\Omega = \{i_1, i_2, \cdots, i_P\} \subseteq [K]$ with cardinality $|\Omega| = P$ to any individual database (user privacy constraint), and while not obtaining any further information beyond the desired message set W_Ω (database privacy constraint). The cardinality of the retrieved message set P is public knowledge and known by all the databases. Due to the database privacy constraint, databases need to share some amount of server-side common randomness $\mathcal{R}_S$ that is unknown to the user. The server-side common randomness $\mathcal{R}_S$ is independent of the message set $W_{[K]}$ in the server.

The desired message index set Ω is a random variable corresponding to a uniform selection of elements without replacement from the set $[K]$ and the sample space of Ω is the power set of $[K]$. We use $\mathcal{P}$ to denote the realization of the random variable Ω. Based on the desired message set Ω, the user generates a set of queries $Q_{[N]}^{[\Omega]}$ without knowing the message set $W_{[K]}$ stored in the databases, hence,

$$I(W_{[K]}; Q_{[N]}^{[\Omega]}, \Omega) = 0 \tag{7.3}$$

For any desired message index set $\mathcal{P}$, after receiving a query from the user, each database responds with a truthful answer based on the stored message set and

the server-side common randomness,

$$\text{[MM-SPIR deterministic answer]} \quad H(A_n^{[\mathcal{P}]}|Q_n^{[\mathcal{P}]}, W_{[K]}, \mathcal{R}_S) = 0, \quad \forall n, \ \forall \mathcal{P} \tag{7.4}$$

Subsequently, the user should be able to decode the desired set of messages reliably after collecting N answers from all the databases,

$$\text{[MM-SPIR reliability]} \quad H(W_{\mathcal{P}}|Q_{[N]}^{[\mathcal{P}]}, A_{[N]}^{[\mathcal{P}]}, \mathcal{P}) = 0, \quad \forall \mathcal{P} \tag{7.5}$$

Due to the user privacy constraint, the query generated to retrieve the desired set of messages should be statistically indistinguishable from other queries. Thus, for all realizations $\mathcal{P}$ and $\mathcal{P}'$, such that $\mathcal{P} \neq \mathcal{P}'$ and $|\mathcal{P}| = |\mathcal{P}'| = P$,

$$\text{[MM-SPIR user privacy]} \quad (Q_n^{[\mathcal{P}]}, A_n^{[\mathcal{P}]}, W_{[K]}, \mathcal{R}_S) \sim (Q_n^{[\mathcal{P}']}, A_n^{[\mathcal{P}']}, W_{[K]}, \mathcal{R}_S) \tag{7.6}$$

which is equivalent to the following one,

$$\text{[MM-SPIR user privacy]} \quad I(\Omega; Q_n^{[\Omega]}, A_n^{[\Omega]}, W_{[K]}, \mathcal{R}_S) = 0, \quad \forall n \tag{7.7}$$

Due to the database privacy constraint, the user should learn nothing about $W_{\bar{\mathcal{P}}}$ which is the complement of $W_{\mathcal{P}}$, i.e., $W_{\bar{\mathcal{P}}} = W_{[K]\setminus\mathcal{P}}$,

$$\text{[MM-SPIR database privacy]} \quad I(W_{\bar{\mathcal{P}}}; Q_{[N]}^{[\mathcal{P}]}, A_{[N]}^{[\mathcal{P}]}, \mathcal{P}) = 0, \quad \forall \mathcal{P} \tag{7.8}$$

An achievable MM-SPIR scheme is a scheme that satisfies the reliability constraint (7.5), the user privacy constraint (7.6) and the database privacy constraint (7.8). Similar to single-database SPIR [8], single-database MM-SPIR is infeasible as well. In order to make single-database MM-SPIR feasible, we use the multi-message version of the extended SPIR formulation in [75], where the user is able to fetch a random subset of the shared server-side common randomness before the retrieval process starts, in the MM-SPIR setting.

7.2.2 PSU

In the PSU problem, two parties each holding a dataset, wish to jointly compute the union of their sets without revealing anything else to either party. Let $\mathcal{A}$ denote the global alphabet. The first party P_1 stores a dataset Ω_1 across its own $N_1 \geq 1$ replicated and non-colluding databases, and the second party P_2 stores a dataset Ω_2 across its own $N_2 \geq 1$ replicated and non-colluding databases. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ denote the realizations of the random variables Ω_1 and Ω_2, respectively. All elements in $\mathcal{P}_1$ and $\mathcal{P}_2$ are selected from $\mathcal{A}$ under an arbitrary statistical distribution, i.e., $\mathcal{P}_1, \mathcal{P}_2 \subseteq \mathcal{A}$. We denote one of the parties as the leader/server and the other as the client/user. Without loss of generality, let party P_1 be the server. Then, as in [72], P_1 privacy, P_2 privacy and PSU reliability constraints jointly form a contradiction, and as in all SPIR formulations [7, 8], the server databases need to share an amount of common randomness $\mathcal{R}_S$ besides their own datasets. Then, the party P_2 generates N_1 queries $Q_{[N_1]}^{[\mathcal{P}_2]}$ and sends them to the databases associated with the party P_1.

After receiving the query $Q_{n_1}^{[\mathcal{P}_2]}$, the n_1th database of the party P_1 responds with an answer $A_{n_1}^{[\mathcal{P}_2]}$.

For each database in the party P_1, the answer $A_{n_1}^{[\mathcal{P}_2]}$ should be generated truthfully according to the received query, its own dataset and its own common randomness,

$$\text{[PSU deterministic answer]} \quad H(A_{n_1}^{[\mathcal{P}_2]}|Q_{n_1}^{[\mathcal{P}_2]}, \Omega_1, \mathcal{R}_S) = 0, \quad \forall n_1, \ \forall \mathcal{P}_2 \tag{7.9}$$

When the PSU process is complete, the party P_2 should be able to reliably compute the union $\Omega_1 \cup \Omega_2$ based on the sent queries, the collected answers and the knowledge of Ω_2 without knowing $|\Omega_1 \cup \Omega_2|$ in advance. This is captured by the PSU reliability constraint,

$$\text{[PSU reliability]} \quad H(\Omega_1 \cup \Omega_2|Q_{[N_1]}^{[\mathcal{P}_2]}, A_{[N_1]}^{[\mathcal{P}_2]}, \Omega_2) = 0, \quad \forall \mathcal{P}_2 \tag{7.10}$$

The privacy requirements in PSU can be divided into two parts to protect each participating party: P_1 privacy and P_2 privacy. First, the party P_2 wants to protect $\Omega_1 \cup \Omega_2$, however, since the party P_2 does not know Ω_1 when generating its queries, the queries cannot depend on Ω_1, and thus, P_2 should only protect Ω_2 in queries. Thus, the queries sent by P_2 should not leak any information about its own dataset, i.e., any individual database associated with P_1 learns nothing about Ω_2

from all the information it has,

$$\text{[PSU } P_2 \text{ privacy]} \qquad I(\Omega_2; Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) = 0, \quad \forall n_1 \tag{7.11}$$

Because of the known and fixed global alphabet $\mathcal{A}$, it is obvious that we have the following two constraints $H(\Omega_2|\bar{\Omega}_2) = 0$ and $H(\bar{\Omega}_2|\Omega_2) = 0$, which lead to the following relationship,

$$H(\Omega_2) = H(\Omega_2) - H(\Omega_2|\bar{\Omega}_2) \tag{7.12}$$

$$= I(\Omega_2; \bar{\Omega}_2) \tag{7.13}$$

$$= H(\bar{\Omega}_2) - H(\bar{\Omega}_2|\Omega_2) \tag{7.14}$$

$$= H(\bar{\Omega}_2) \tag{7.15}$$

Thus, we obtain the following identity,

$$I(\bar{\Omega}_2; Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S)$$

$$= H(\bar{\Omega}_2) - H(\bar{\Omega}_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) \tag{7.16}$$

$$= H(\bar{\Omega}_2) - H(\bar{\Omega}_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) + H(\bar{\Omega}_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \Omega_2, \mathcal{R}_S) \tag{7.17}$$

$$= H(\bar{\Omega}_2) - I(\bar{\Omega}_2; \Omega_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) \tag{7.18}$$

$$= H(\Omega_2) - H(\Omega_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) + H(\Omega_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \bar{\Omega}_2, \mathcal{R}_S) \tag{7.19}$$

$$= H(\Omega_2) - H(\Omega_2|Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) \tag{7.20}$$

$$= I(\Omega_2; Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) \tag{7.21}$$

As a consequence, we obtain the following equivalent expression for P_2 privacy,

$$[\text{PSU } P_2 \text{ privacy}] \qquad I(\bar{\Omega}_2; Q_{n_1}^{[\Omega_2]}, A_{n_1}^{[\Omega_2]}, \Omega_1, \mathcal{R}_S) = 0, \quad \forall n_1 \tag{7.22}$$

From the union result $\Omega_1 \cup \Omega_2$, the party P_2 always knows that the party P_1 contains the elements in $(\Omega_1 \cup \Omega_2) \backslash \Omega_2$ and does not contain the elements in $\overline{(\Omega_1 \cup \Omega_2)}$. Noting that $(\Omega_1 \cup \Omega_2) \backslash \Omega_2 \cup \overline{(\Omega_1 \cup \Omega_2)} = \bar{\Omega}_2$, thus, P_2 should learn nothing about whether P_1 contains any element in Ω_2 (we denote this information by E_{1,Ω_2}) from the generated queries, the collected answers and its own dataset,

$$[\text{PSU } P_1 \text{ privacy}] \quad I(E_{1,\Omega_2}; Q_{[N_1]}^{[\mathcal{P}_2]}, A_{[N_1]}^{[\mathcal{P}_2]}, \Omega_2) = 0, \quad \forall \mathcal{P}_2 \tag{7.23}$$

Theorem 7.1 *PSU is equivalent to MM-SPIR with $L = 1$ and $P = |\mathcal{A}| - |\Omega_2|$.*

Proof: We prove the equivalence between PSU and MM-SPIR similar to the proof of equivalence between PSI and MM-SPIR in [72] after mapping the dataset in each party to a corresponding incidence vector. Specifically, the P_1 privacy, P_2 privacy and PSU reliability constraints in the PSU problem are consistent with the database privacy, user privacy and reliability constraints in the MM-SPIR problem if $\bar{\Omega}_2$ in PSU is treated as Ω in MM-SPIR. By contrast, the consistency of the three constraints of PSI and MM-SPIR is true if Ω_2 in PSI is treated as Ω in MM-SPIR.

■

Remark 7.1 *From [72], we note that PSI is equivalent to MM-SPIR with $L = 1$ and $P = |\Omega_2|$. From Theorem 7.1 above, we note that PSU is equivalent to MM-*

SPIR with $L = 1$ and $P = |\mathcal{A}| - |\Omega_2|$.[1] *From the de Morgan's law, which says $\overline{A \cup B} = \overline{A} \cap \overline{B}$, we have that $A \cup B = \overline{\overline{A} \cap \overline{B}}$, thus, the set union can be obtained by a composition of set intersection and set complement. This shows the duality between PSU and PSI problems. We note, however, that the parties should agree on whether they will perform PSU or PSI, as the specific protocol will depend on it. In this chapter, we focus on designing specific PSU protocols.*

Remark 7.2 *In certain applications of PSU, one or both of the parties may have only a single database. Since PSU is equivalent to MM-SPIR from Theorem 7.1, and since single-database MM-SPIR is infeasible [8], in such cases, one of the two parties may obtain (fetch) a random subset of the shared server-side common randomness from the other party prior to the start of the PSU process, as in [75]. This makes MM-SPIR feasible, and thus, PSU feasible.*

Remark 7.3 *As PSI was generalized to multi-party PSI (MP-PSI) in [74], PSU can be generalized to MP-PSU. As in MP-PSI, MP-PSU will require additional common randomness allocation among the clients. To avoid repetition, we skip the detailed development of MP-PSU, however, in the next subsection, we present a particular MP-PSU in detail, where one party has no input. As a critical difference, in the reliability verification stage, we need to have the sum in [74, Eqn. (56)] equal to 0 if all the clients contain the same element in the MP-PSI problem while this sum should be 0 if none of the clients contain this element in the MP-PSU problem;*

[1] These two conclusion are built upon the assumption that the party P_2 is the user. As an alternative, if the party P_1 is treated as the user, we just need to replace Ω_1 with Ω_2 in these two statements.

see Example 13 for details. In MP-PSU, if all the parties have a single database, we can construct an achievable scheme by using pre-fetched server-side common randomness from the leader party as in [75]. In addition, for common randomness allocation among the clients, we make use of the distributed property of non-colluding databases as well as the RSPIR approach introduced in [116].

7.2.3 Private Distributed FSL

We consider a distributed FSL problem with one server that contains $N = 2$ non-colluding and replicated databases[2], and C clients that are selected by the server to participate in one round of the FSL process; see Fig. 7.1. By convention, every client establishes a direct secure and authenticated communication channel with both databases. The full model for learning stored at the server side comprises K submodels, each one of which consisting of L i.i.d. symbols that are uniformly selected from a finite field $\mathbb{F}_q$. Thus, each database in the server contains the full model $M_{[K]}$, and we have,

$$H(M_k) = L, \quad \forall k \tag{7.24}$$

$$H(M_{[K]}) = H(M_1) + \cdots + H(M_K) = KL \tag{7.25}$$

The two databases also share an amount of server-side common randomness $\mathcal{R}_S$ that is unknown to the clients. Each selected client is interested in updating one

[2]We start this investigation with the simplest case of two databases. Our achievable scheme works for any number of databases after minor modifications. However, how to improve the performance by increasing the number of databases needs further study.

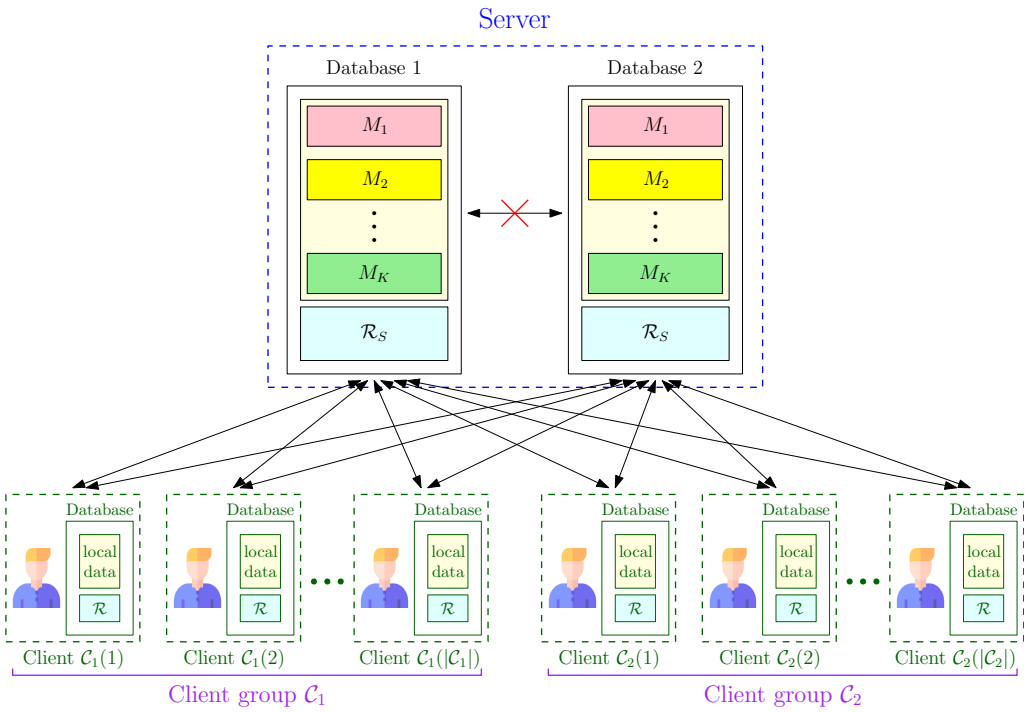


Figure 7.1: Distributed federated submodel learning (FSL) system model.

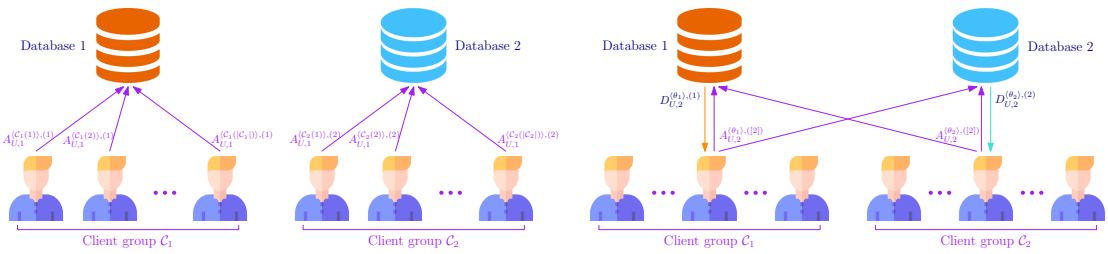
or more submodels according to its local training data. Specifically, for $i \in [C]$, the ith client wishes to update the submodels whose index set is denoted by the random variable $\Gamma^{\langle i \rangle}$, whose realization is denoted by $\gamma^{\langle i \rangle}$. For $i \in [C]$, the random variable $Y^{\langle i \rangle} = \{Y_1^{\langle i \rangle}, Y_2^{\langle i \rangle}, \ldots, Y_K^{\langle i \rangle}\}$ is used to denote the corresponding incidence vector of $\Gamma^{\langle i \rangle}$ after mapping to the alphabet as in [72, 74].

We formulate our FSL process following the seminal FSL work in [87]. At the beginning, each individual database in the server needs to calculate the union of the selected clients' desired submodel index sets $\Gamma^{\langle 1 \rangle} \cup \Gamma^{\langle 2 \rangle} \cup \cdots \cup \Gamma^{\langle C \rangle}$ denoted by Γ. This phase is referred to as the FSL-PSU phase. Due to the constraint that the two databases in the server cannot communicate with each other directly, our solution is to use randomly selected alive clients as intermediators to route the information received by the two databases rather than to enforce each client to

send the same answer to both databases. The main objective of this new approach is to reduce the total communication cost and the needed communication time. Thus, we separate C clients into two groups: a group of clients whose index set denoted by $\mathcal{C}_1 = \{\mathcal{C}_1(1), \mathcal{C}_1(2), \ldots, \mathcal{C}_1(|\mathcal{C}_1|)\}$ are associated with database 1, and the other group of clients whose index set denoted by $\mathcal{C}_2 = \{\mathcal{C}_2(1), \mathcal{C}_2(2), \ldots, \mathcal{C}_2(|\mathcal{C}_2|)\}$ are associated with database 2. A potential separation method is to rely on each client's communication channel bandwidth (or quality) with the two databases. For instance, a client is classified as belonging to $\mathcal{C}_1$ if its channel with database 1 has a higher bandwidth (quality) than the channel with database 2. Otherwise, this client is considered as belonging to $\mathcal{C}_2$. Note that $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$ and $\mathcal{C}_1 \cup \mathcal{C}_2 = [C]$. Please see Figs. 7.1 and 7.2 for depictions.

The FSL-PSU phase is further divided into two steps considering the fact that two random clients (one from each client group) are utilized to relay the information between the databases; see Fig. 7.2. This information is produced from the answers that are collected by the two databases individually from their associated client groups. In the first step, there is no need for each client to download any information from the databases since the server itself is not involved in the PSU computation, namely the downloads $D_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}$ are null[3] for all $j \in [2]$. As a consequence, the only operation in this step is to make clients send their well-designed

[3]In this work, we use the value in $\langle\rangle$ to denote the index of client and the value in () to denote the index of database for clarity. The superscript of the download D or the answer A in the following text implies the information flow during the client-database communication. The first subscript of D or A is used to show it is within the FSL-PSU phase or FSL-write phase (the letter U stands for union and the letter W stands for write), whereas the second subscript is used to denote the step number within this phase. In particular, "$D_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}$ are null" here means that the communication between any client in $\mathcal{C}_j$ and database j is always empty in the first step of FSL-PSU phase.



(a) FSL-PSU phase step 1. (b) FSL-PSU phase step 2.

Figure 7.2: Data flow in the FSL-PSU phase of our FSL system model.

answers $A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}$ to the associated database. In the second step, for all $j \in [2]$, database j processes the answers received from its associated clients with the aid of its own server-side common randomness, and then the produced $D_{U,2}^{\langle \theta_j \rangle,(j)}$ is merely downloaded by a randomly chosen client whose index is θ_j within its associated client group $\mathcal{C}_j$. Finally, client θ_j forwards the same processed answer $A_{U,2}^{\langle \theta_j \rangle,([2])}$ based on the received download to both databases; see Fig. 7.2.

Similar to the conventional multi-user PIR/SPIR problem formulated in [117, 118], the constraints accompanying FSL-PSU phase comprises three parts. First, each database j should be able to reliably determine the union Γ using all the collected answers $\{A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{U,2}^{\langle \theta_{[2]} \rangle,(j)}\}$ within two FSL-PSU steps and its own server-side common randomness $\mathcal{R}_S$, which is captured by,

$$\text{[FSL-PSU reliability]} \quad H(\Gamma | A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{U,2}^{\langle \theta_{[2]} \rangle,(j)}, \mathcal{R}_S) = 0, \quad \forall j \tag{7.26}$$

Second, the databases should not learn anything further about the set $\Gamma^{\langle [C] \rangle}$ or $Y^{\langle [C] \rangle}$ other than the union Γ. Note that if an element is not in the union Γ, each database concludes that no client contains this element. Otherwise, each database

learns that at least one client contains this element. Let $Y_\Gamma^{\langle i \rangle} = \{Y_k^{\langle i \rangle} : k \in \Gamma\}$, we define a new set $Y_\Gamma = Y_\Gamma^{\langle [C] \rangle}$, then,

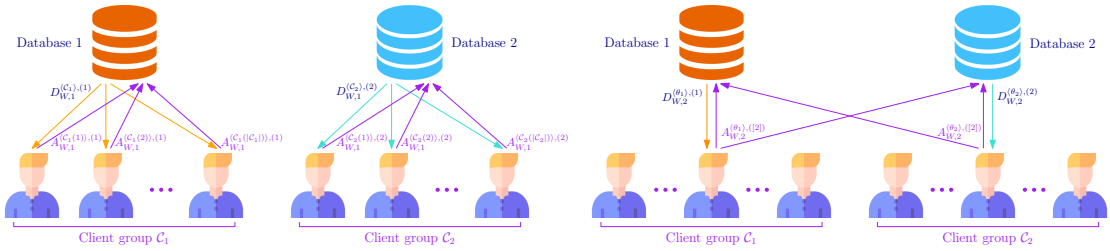
$$\text{[FSL-PSU privacy]} \quad I(Y_\Gamma; A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{U,2}^{\langle \theta_{[2]} \rangle,(j)}, \mathcal{R}_S | \sum_{i \in [C]} Y_k^{\langle i \rangle} > 0, \forall k \in \Gamma) = 0, \quad \forall j \tag{7.27}$$

Third, client θ_j that obtains the download $D_{U,2}^{\langle \theta_j \rangle,(j)}$ from database j should learn nothing about the other clients' desired submodel indices. Hence, we have the following constraint,

$$\text{[FSL-PSU inter-client privacy]} \quad I(Y_\Gamma^{\langle [C] \setminus \theta_j \rangle}; D_{U,2}^{\langle \theta_j \rangle,(j)}, Y^{\langle \theta_j \rangle}) = 0, \quad \forall j \tag{7.28}$$

A valid FSL-PSU phase is one that satisfies the FSL-PSU reliability (7.26), the FSL-PSU privacy (7.27) and the FSL-PSU inter-client privacy (7.28). The efficiency of an FSL-PSU phase is measured in terms of the number of bits in the involved communication strings. Therefore, for the FSL-PSU phase itself, we wish to reduce the total number of bits in the answers $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}, A_{U,2}^{\langle \theta_1 \rangle,([2])}, A_{U,2}^{\langle \theta_2 \rangle,([2])}\}$ and downloads $\{D_{U,2}^{\langle \theta_1 \rangle,(1)}, D_{U,2}^{\langle \theta_2 \rangle,(2)}\}$ to the extent possible.

When the FSL-PSU phase is completed, each database will learn Γ, the union of the submodel indices to be updated. Next, we proceed to the FSL-write phase where each database will update the full learning model synchronously. The FSL-write phase is analogous to the FSL-PSU phase, and therefore, is also divided into two steps as the FSL-PSU phase. The difference is that in the first step, both



(a) FSL-write phase step 1. (b) FSL-write phase step 2.

Figure 7.3: Data flow in the FSL-write phase of our FSL system model.

databases broadcast the same set of submodels $M_\Gamma = \{M_k : k \in \Gamma\}$ to their associated clients before each client trains its desired submodel set $M_{\Gamma^{\langle i \rangle}}$ by employing its local data. Hence, for all $j \in [2]$, the downloads $D_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}$ are always in the form of M_Γ. Subsequently, clients send their well-processed answer $A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}$ corresponding to the submodel updates back to the associated database. In the second step, for all $j \in [2]$, database j processes its associated clients' answers through different server-side common randomness and then the produced $D_{W,2}^{\langle \theta_j \rangle,(j)}$ is downloaded by the θ_jth client again. Finally, the θ_jth client forwards the same resulting answer $A_{W,2}^{\langle \theta_j \rangle,([2])}$ to both databases after processing the newly received download; see Fig. 7.3.

Likewise, the constraints accompanying FSL-write phase comprises three parts. First, each database j should be able to reliably obtain the aggregation of all the submodel updates according to all the collected answers $\{A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{W,2}^{\langle \theta_{[2]} \rangle,(j)}\}$ within two FSL-write steps, its own current full model $M_{[K]}$ and its own server-side common randomness $\mathcal{R}_S$. When the submodel training by means of the local data in the ith client is complete, for all $k \in \Gamma^{\langle i \rangle}$, this client will generate the increment $\Delta_k = \{\Delta_{k,1}^{\langle i \rangle}, \Delta_{k,2}^{\langle i \rangle}, \ldots, \Delta_{k,L}^{\langle i \rangle}\}$ for each symbol in the submodel $M_k = \{M_{k,1}, M_{k,2}, \ldots, M_{k,L}\}$. Thus, for the kth submodel, let Φ_k be the set of clients

whose desired submodel index set $\Gamma^{\langle i \rangle}$ contains k, its correct updated version should be $M'_k = \{M'_{k,1}, M'_{k,2}, \ldots, M'_{k,L}\} = \{M_{k,1} + \sum_{i \in \Phi_k} \Delta_{k,1}^{\langle i \rangle}, M_{k,2} + \sum_{i \in \Phi_k} \Delta_{k,2}^{\langle i \rangle}, \ldots, M_{k,L} + \sum_{i \in \Phi_k} \Delta_{k,L}^{\langle i \rangle}\}$. For each database in the server, the correct updated submodel aggregation should be $M'_\Gamma = \{M'_k \colon k \in \Gamma\}$ and thus the first constraint can be expressed as,

$$[\text{FSL-write reliability}] \quad H(M'_\Gamma | A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{W,2}^{\langle \theta_{[2]} \rangle,(j)}, M_{[K]}, \mathcal{R}_S) = 0, \quad \forall j \tag{7.29}$$

Second, no database should learn any knowledge about each client's desired submodel index set or any further information beyond the updated submodel aggregation about each client's submodel increment. For the ith client's submodel increment, let $\Delta_\Gamma^{\langle i \rangle} = \{\Delta_{k,l}^{\langle i \rangle} \colon k \in \Gamma, l \in [L]\}$, we define a new set $\Delta_\Gamma = \Delta_\Gamma^{\langle [C] \rangle}$, then,[4]

$$[\text{FSL-write privacy}] I(\Delta_\Gamma; A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{W,2}^{\langle \theta_{[2]} \rangle,(j)}, M_{[K]}, \mathcal{R}_S | \sum_{i \in \Phi_k} \Delta_{k,l}^{\langle i \rangle}, \forall k \in \Gamma, \forall l \in [L]) = 0, \forall j \tag{7.30}$$

Third, the θ_jth client should learn nothing about the other clients' desired submodel indices or submodel increments according to its obtained download $D_{W,2}^{\langle \theta_j \rangle,(j)}$

[4]In general, the first term in the following conditional mutual information should be Y_Γ, Δ_Γ rather than Δ_Γ. In the FSL-write phase, we note that the information transmission only involves the submodel increment regarding M_Γ and it has nothing to do with the incidence vectors $Y^{\langle [C] \rangle}$. That means that if a database learns nothing beyond the aggregation increment from all the selected clients, this database definitely learns nothing about the incidence vector Y_Γ. Therefore, the expression Δ_Γ takes the place of Y_Γ, Δ_Γ. This observation also applies to the FSL-write inter-client privacy constraint (7.31) in which the expression $\Delta_\Gamma^{\langle [C] \setminus \theta_j \rangle}$ is used in place of $Y_\Gamma^{\langle [C] \setminus \theta_j \rangle}, \Delta_\Gamma^{\langle [C] \setminus \theta_j \rangle}$.

from database j. Hence, we have the following constraint,

$$\text{[FSL-write inter-client privacy]} \quad I(\Delta_{\Gamma}^{\langle [C]\backslash\theta_j\rangle}; D_{W,2}^{\langle\theta_j\rangle,(j)}, \Gamma^{\langle\theta_j\rangle}, \Delta_{\Gamma^{\langle\theta_j\rangle}}) = 0 \tag{7.31}$$

A valid FSL-write phase is a one that satisfies the FSL-write reliability (7.29), the FSL-write privacy (7.30) and the FSL-write inter-client privacy (7.31). Given any specific FSL problem with fixed initial parameters, the communication cost of sending the set of submodels M_{Γ} to each client from the two databases is a constant. Hence, the efficiency of an FSL-write phase is also measured in terms of the total number of bits in the answers $\{A_{W,1}^{\langle\mathcal{C}_1\rangle,(1)}, A_{W,1}^{\langle\mathcal{C}_2\rangle,(2)}, A_{W,2}^{\langle\theta_1\rangle,([2])}, A_{W,2}^{\langle\theta_2\rangle,([2])}\}$ and downloads $\{D_{W,2}^{\langle\theta_1\rangle,(1)}, D_{W,2}^{\langle\theta_2\rangle,(2)}\}$, and we wish to minimize it as much as possible. If we do not consider the generation of client-side common randomness that is necessary to perform the FSL, one complete FSL round consists of two phases: FSL-PSU phase and FSL-write phase. Our objective is to make the total number of communication bits exchanged in these two phases as small as possible. Further, this FSL round can be executed in an iterative manner until a predefined termination criterion is satisfied, e.g., the accuracy of the updated global model exceeds the preset threshold or a preset maximal number of iterations is reached.

7.3 Main Result

Our main result is a new private FSL algorithm as described above. The following theorem gives its performance in terms of the total communication cost in the entire process including the cost of FSL-PSU, FSL-write, and the generation of the

necessary common randomness at the clients. The proof of the theorem is given in Section 7.5.2 and Section 7.5.3.

Theorem 7.2 *The total communication cost of the proposed private FSL scheme in one round is $\mathcal{O}(CK+C|\Gamma|L)$ in q-ary bits, where C is the number of selected clients, K is the total number of submodels, and $|\Gamma|$ is the number of updated submodels in the given round. Here, $\mathcal{O}(CK)$ is due to the FSL-PSU phase, while $\mathcal{O}(C|\Gamma|L)$ is due to the FSL-write phase.*

Remark 7.4 *The achievability of the theorem starts with an MM-SPIR with multiple replicated and non-colluding databases. The storage in the databases is uncoded and without noise. We unify PSU and secure aggregation in a common information theoretic framework, and propose a novel private FSL scheme. We take advantage of the non-colluding aspect of the databases to implement simple common randomness generation/distribution across selected clients.*

Remark 7.5 *Our proposed FSL achieves unconditional information theoretic privacy. This is different from most prior secure aggregation works that focus on the computational security, e.g., [87, 96, 99, 119, 120]. It is also different from prior private read update write (PRUW) works [92, 93, 121–124] in which only a single client at a time updates the full model in an FSL round, although information theoretic security is satisfied. Our proposed private FSL scheme is robust against client drop-outs, client late-arrivals, and database drop-outs. Moreover, there is no constraint on the number of clients that may drop-out during the FSL process.*

Remark 7.6 *The communication cost of our proposed private FSL,* $\mathcal{O}(CK + C|\Gamma|L)$*, outperforms the best-known communication cost in the existing literature [96–99], which is at least* $\mathcal{O}(CKL)$*. In the seminal FSL work [87], the communication cost is* $\mathcal{O}(C|\Gamma|)$ *for the PSU phase and* $\mathcal{O}(C|\Gamma|L)$ *for the whole FSL process with much weaker privacy guarantee. Although this communication cost is a little better than our communication cost in terms of the PSU phase, the PSU [87] yields erroneous results while our PSU yields completely accurate results. Furthermore, the PSU problem and the subsequent secure aggregation problem are considered separately in [87]. We note that the total number of submodels* K *is very large when each product is represented by an individual submodel in the e-commerce recommendation system in [87]. Thus, given the scale of the full learning model and the general average size of clients' desired products in practice, we can further optimize the communication cost by adjusting the size of* K*, e.g., combining relevant products into the same goods category. Specifically, as we decrease* K*, the product of* $|\Gamma|$ *and* L *will likely increase such that* K *and* $|\Gamma|L$ *will have the same order. Thus, the communication cost of our scheme is superior to existing schemes, and can be further improved by optimizing the system model parameters. However, it is difficult to find a fair metric to compare our communication cost with the ones in [92, 94]. The main reason for this is that the schemes in [92, 94] require that only one client updates one submodel at a time, and also heavily rely on the sufficiently large number of databases* N*. That is, the schemes in [92, 94] require at least* $N \geq 4$ *databases, and cannot be compared to the scheme in this chapter where the number of databases is* $N = 2$*. If we follow the asymptotic assumption* $L \gg K$ *and let* C *take value* 1*, the*

only conclusion we can draw is that, the communication cost in these two different schemes are both a linear function of the submodel size L.

Remark 7.7 *Generally, the existing private FL schemes in the computer science literature rely on heavy cryptographic computations, while our proposed FSL scheme relies only on simple addition and multiplication computations in the finite field $\mathbb{F}_q$, at both client and server sides. In addition, due to unstable inter-client communications in practice, and the impermissible inter-database communication in our assumption, our FSL scheme relies only on client-database communications. In order to alleviate the challenges arising from the lack of inter-database communications, as a novel approach in our FSL scheme, we utilize random clients to route the required information between the databases in the server. The routed information comes from the answers collected by each database from its associated clients, and we further protect this information between the clients (inter-client privacy).*

Remark 7.8 *In practical implementations, for each client, the upload speeds are typically much slower than download speeds during the client-database communications. Unlike the classical secure aggregation scheme in [96], the total communication time in our FSL process is further improved, since almost all of the alive clients send only one answer to one database in each phase. In addition, while determining the two client groups to be connected to the two databases, we can further improve the total communication time based on the actual bandwidth/quality of each client-database communication channel.*

Remark 7.9 *The proposed private FSL scheme can be used iteratively in multiple*

rounds of an FSL process by refreshing server-side and client-side common randomness.

7.4 Examples for Blocks of Private Distributed FSL

In this section, we give examples to explain the functionalities of the modules (boxes) in Fig. 1.2. The examples get progressively more complex: Example 11 considers a two-party PSU setting where the client party has multiple databases and the leader party has a single database. Example 12 considers the slightly more difficult version of Example 11, in that the client party also has a single database. In this case, single-database SPIR is infeasible, and the leader party needs to fetch client-side common randomness to use as leader-side side information as in [75]. Example 13 considers generalized version of Example 12 to a multi-party (MP) case; in particular, there are 5 parties and each party has a single database. Example 14 is an extended version of Example 13, where the leader party has two databases. This example reflects how the FSL-PSU phase of the proposed private FSL scheme works. Finally, Example 15 shows how the private write works. The FSL-PSU in Example 14 and the FSL-write in Example 15 together constitute our proposed private FSL scheme.

Example 11: Two-party PSU; two-database client; one-database leader: Consider a two-party PSU problem with a global alphabet $\mathcal{A} = \{1, 2, 3, 4\}$. The first party P_1 contains element 1 and element 2, i.e., $\mathcal{P}_1 = \{1, 2\}$. The second party P_2 contains element 1 and element 3, i.e., $\mathcal{P}_2 = \{1, 3\}$. For convenience, the total number of elements in each party is public knowledge. The parties want to jointly

compute the union of their element sets without revealing anything else to each other. By mapping their element sets into the corresponding incidence vectors, two parties construct the vectors as follows,

$$\text{Party } P_1: \quad \mathcal{P}_1 = \{1,2\} \quad \Rightarrow \quad X^{\langle 1\rangle} = [X_1^{\langle 1\rangle} \ X_2^{\langle 1\rangle} \ X_3^{\langle 1\rangle} \ X_4^{\langle 1\rangle}]^T = [1 \ 1 \ 0 \ 0]^T \quad (7.32)$$

$$\text{Party } P_2: \quad \mathcal{P}_2 = \{1,3\} \quad \Rightarrow \quad X^{\langle 2\rangle} = [X_1^{\langle 2\rangle} \ X_2^{\langle 2\rangle} \ X_3^{\langle 2\rangle} \ X_4^{\langle 2\rangle}]^T = [1 \ 0 \ 1 \ 0]^T \quad (7.33)$$

First, P_2 (leader party) asks for the value of $X_2^{\langle 1\rangle}$ from P_1 using an SPIR approach. P_1 (client party) has two replicated and non-colluding databases. These two databases share a server-side common randomness symbol S_1 that is uniformly selected from the finite field $\mathbb{F}_2$ and unknown to P_2. As a consequence, P_1 generates the answer table for two individual databases in the following form,

$$A_U^{(1)}(1) = S_1, \qquad A_U^{(2)}(1) = X_1^{\langle 1\rangle} + S_1 \quad (7.34)$$

$$A_U^{(1)}(2) = X_1^{\langle 1\rangle} + X_2^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(2) = X_2^{\langle 1\rangle} + S_1 \quad (7.35)$$

$$A_U^{(1)}(3) = X_1^{\langle 1\rangle} + X_3^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(3) = X_3^{\langle 1\rangle} + S_1 \quad (7.36)$$

$$A_U^{(1)}(4) = X_1^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(4) = X_4^{\langle 1\rangle} + S_1 \quad (7.37)$$

$$A_U^{(1)}(5) = X_2^{\langle 1\rangle} + X_3^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(5) = X_1^{\langle 1\rangle} + X_2^{\langle 1\rangle} + X_3^{\langle 1\rangle} + S_1 \quad (7.38)$$

$$A_U^{(1)}(6) = X_2^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(6) = X_1^{\langle 1\rangle} + X_2^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1 \quad (7.39)$$

$$A_U^{(1)}(7) = X_3^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1, \qquad A_U^{(2)}(7) = X_1^{\langle 1\rangle} + X_3^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1 \quad (7.40)$$

$$A_U^{(1)}(8) = X_1^{\langle 1\rangle} + X_2^{\langle 1\rangle} + X_3^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1, \quad A_U^{(2)}(8) = X_2^{\langle 1\rangle} + X_3^{\langle 1\rangle} + X_4^{\langle 1\rangle} + S_1 \quad (7.41)$$

Following the notation in Section 7.2, the superscript of A denotes the database index of P_1 while the index on the right-hand side of A denotes the potential query choice that can be chosen by P_2.

In order to retrieve $X_2^{\langle 1\rangle}$, P_2 selects a random query choice for the first database of P_1 and its coupled query for the second database of P_1. For instance, P_2 chooses 1 for the first database of P_1 and 2 for the second database of P_1. After receiving the query symbol 1, the first database of P_1 responds with the answer $A_U^{(1)}(1) = S_1$. Meanwhile, the second database of P_1 responds with the answer $A_U^{(2)}(2) = X_2^{\langle 1\rangle} + S_1$ when the query symbol 2 is received. Next, P_2 asks for the value of $X_4^{\langle 1\rangle}$ from P_1 in the same way.

Since there are 8 possible queries to each database of P_1, the communication cost from P_2 to P_1 is $3+3 = 6$ bits; and since each database of P_1 sends back a single bit of answer, the communication cost from P_1 to P_2 is $1 + 1 = 2$ bits. Thus, the total communication cost for learning $X_2^{\langle 1\rangle}$ and $X_4^{\langle 1\rangle}$ is $2 \cdot (6 + 2) = 16$ bits through this MM-SPIR approach. After learning the values of $X_2^{\langle 1\rangle}$ and $X_4^{\langle 1\rangle}$, P_2 knows that P_1 has element 2 but does not have element 4. Combining its own elements, P_2 is able to calculate the union, which is $\{1, 2, 3\}$. Thus, the PSU reliability constraint is satisfied. Regarding the two privacy constraints, due to the user privacy constraint in the SPIR problem, each individual database in P_1 can only learn that P_2 has two elements without learning any knowledge about what these two specific elements are. Due to the database privacy constraint in the SPIR problem, P_2 can only learn that P_1 possesses element 2 and does not possess element 4 without any additional knowledge about whether P_1 has elements $1, 3$. In particular, whether P_1 has element

4 or not can be deduced by P_2 from the ultimate union result and its own elements. Thus, both of P_1 and P_2 privacy constraints are guaranteed. Thus, this is a valid two-party PSU scheme.

Example 12: Two-party PSU; one-database client; one-database leader: Compared to Example 11, the only modification in the setting is that party P_1 now has a single database. Party P_1 also holds four server-side common randomness symbols S_1, S_2, S_3 and S_4 that are all uniformly selected from the finite field $\mathbb{F}_2$. In order to have a feasible single-database SPIR approach as illustrated in [75], the party P_2 obtains one random server-side common randomness symbol ahead of time. As a consequence, P_1 generates the following answer table for the only database,

$$A_U^{(1)}(1) = \{X_1^{\langle 1\rangle}+S_1, X_2^{\langle 1\rangle}+S_2, X_3^{\langle 1\rangle}+S_3, X_4^{\langle 1\rangle}+S_4\} \tag{7.42}$$

$$A_U^{(1)}(2) = \{X_1^{\langle 1\rangle}+S_2, X_2^{\langle 1\rangle}+S_3, X_3^{\langle 1\rangle}+S_4, X_4^{\langle 1\rangle}+S_1\} \tag{7.43}$$

$$A_U^{(1)}(3) = \{X_1^{\langle 1\rangle}+S_3, X_2^{\langle 1\rangle}+S_4, X_3^{\langle 1\rangle}+S_1, X_4^{\langle 1\rangle}+S_2\} \tag{7.44}$$

$$A_U^{(1)}(4) = \{X_1^{\langle 1\rangle}+S_4, X_2^{\langle 1\rangle}+S_1, X_3^{\langle 1\rangle}+S_2, X_4^{\langle 1\rangle}+S_3\} \tag{7.45}$$

Subsequently, P_2 selects a query choice that matches its pre-fetched server-side common randomness symbol. For instance, if its pre-fetched symbol is S_1, in order to retrieve $X_2^{\langle 1\rangle}$, P_2 chooses 4 and then sends this query symbol to P_1. The database belonging to P_1 responds with the answer $A_U^{(1)}(4) = \{X_1^{\langle 1\rangle} + S_4, X_2^{\langle 1\rangle} + S_1, X_3^{\langle 1\rangle} + S_2, X_4^{\langle 1\rangle} + S_3\}$. Likewise, P_2 also asks for the value of $X_4^{\langle 1\rangle}$ from P_1 in the same way. Since there are 4 possible queries to the database of P_1, the communication cost

from P_2 to P_1 is 2 bits; and since P_1 sends back an answer with 4 components, the communication cost from P_1 to P_2 is 4 bits. Thus, the total communication cost for learning $X_2^{\langle 1\rangle}$ and $X_4^{\langle 1\rangle}$ is $2 \cdot (2+4) = 12$ bits through this MM-SPIR approach, without considering the communication cost generated by the pre-fetched server-side common randomness symbol. Verification that this achievable scheme satisfies the PSU reliability, P_1 privacy and P_2 privacy constraints follows similarly as in Example 11.

Example 13: Five-party PSU; one-database per client; one-database leader: As a generalization of Examples 11 and 12, in this example, we consider a multi-party setting, again with the global alphabet $\mathcal{A} = \{1, 2, 3, 4\}$. Here, the first party P_1 contains element 1, i.e., $\mathcal{P}_1 = \{1\}$. The second party P_2 contains element 1 and element 3, i.e., $\mathcal{P}_2 = \{1, 3\}$. The third party P_3 contains element 1 and element 4, i.e., $\mathcal{P}_3 = \{1, 4\}$. The fourth party P_4 contains element 1, element 3 and element 4, i.e., $\mathcal{P}_4 = \{1, 3, 4\}$. The fifth party P_5 contains nothing, i.e., $\mathcal{P}_5 = \emptyset$. As before, we assume that the total number of elements in each party is public knowledge. The parties construct the corresponding incidence vectors $X^{\langle [5]\rangle}$ as follows,

$$\text{Party } P_1: \ \mathcal{P}_1 = \{1\} \quad \Rightarrow \ X^{\langle 1\rangle} = [X_1^{\langle 1\rangle} \ X_2^{\langle 1\rangle} \ X_3^{\langle 1\rangle} \ X_4^{\langle 1\rangle}]^T = [1 \ 0 \ 0 \ 0]^T \quad (7.46)$$

$$\text{Party } P_2: \ \mathcal{P}_2 = \{1, 3\} \quad \Rightarrow \ X^{\langle 2\rangle} = [X_1^{\langle 2\rangle} \ X_2^{\langle 2\rangle} \ X_3^{\langle 2\rangle} \ X_4^{\langle 2\rangle}]^T = [1 \ 0 \ 1 \ 0]^T \quad (7.47)$$

$$\text{Party } P_3: \ \mathcal{P}_3 = \{1, 4\} \quad \Rightarrow \ X^{\langle 3\rangle} = [X_1^{\langle 3\rangle} \ X_2^{\langle 3\rangle} \ X_3^{\langle 3\rangle} \ X_4^{\langle 3\rangle}]^T = [1 \ 0 \ 0 \ 1]^T \quad (7.48)$$

$$\text{Party } P_4: \ \mathcal{P}_4 = \{1, 3, 4\} \quad \Rightarrow \ X^{\langle 4\rangle} = [X_1^{\langle 4\rangle} \ X_2^{\langle 4\rangle} \ X_3^{\langle 4\rangle} \ X_4^{\langle 4\rangle}]^T = [1 \ 0 \ 1 \ 1]^T \quad (7.49)$$

$$\text{Party } P_5: \ \mathcal{P}_5 = \emptyset \quad \Rightarrow \ X^{\langle 5\rangle} = [X_1^{\langle 5\rangle} \ X_2^{\langle 5\rangle} \ X_3^{\langle 5\rangle} \ X_4^{\langle 5\rangle}]^T = [0 \ 0 \ 0 \ 0]^T \quad (7.50)$$

Using the MP-PSI achievable scheme in Chapter 4 as a reference, we select party P_5 as the leader party, and the remaining parties as client parties, as party P_5 is globally known as an empty party. Thus, there is no need for P_5 to send any queries to the remaining parties and the server-side common randomness employed in the previous two examples is not necessary any more. In this example, all parties have a single database. Besides their own incidence vectors, each client party holds the same set of common randomness symbols[5] $\{u_\alpha^{\langle[4]\rangle} : \alpha \in [4]\}$ from the finite field $\mathbb{F}_5$ as well as the same global common randomness symbol c that is uniformly distributed over $\{1, 2, 3, 4\}$. Moreover, $\{u_\alpha^{\langle[4]\rangle} : \alpha \in [4]\}$ are such that the sum $\sum_{i\in[4]} u_\alpha^{\langle i\rangle}$ is always equal to 0 for all $\alpha \in [4]$. The answers from the client parties are,

$$A_U^{\langle 1\rangle} = \{c(X_1^{\langle 1\rangle}+u_1^{\langle 1\rangle}), c(X_2^{\langle 1\rangle}+u_2^{\langle 1\rangle}), c(X_3^{\langle 1\rangle}+u_3^{\langle 1\rangle}), c(X_4^{\langle 1\rangle}+u_4^{\langle 1\rangle})\} \tag{7.51}$$

$$A_U^{\langle 2\rangle} = \{c(X_1^{\langle 2\rangle}+u_1^{\langle 2\rangle}), c(X_2^{\langle 2\rangle}+u_2^{\langle 2\rangle}), c(X_3^{\langle 2\rangle}+u_3^{\langle 2\rangle}), c(X_4^{\langle 2\rangle}+u_4^{\langle 2\rangle})\} \tag{7.52}$$

$$A_U^{\langle 3\rangle} = \{c(X_1^{\langle 3\rangle}+u_1^{\langle 3\rangle}), c(X_2^{\langle 3\rangle}+u_2^{\langle 3\rangle}), c(X_3^{\langle 3\rangle}+u_3^{\langle 3\rangle}), c(X_4^{\langle 3\rangle}+u_4^{\langle 3\rangle})\} \tag{7.53}$$

$$A_U^{\langle 4\rangle} = \{c(X_1^{\langle 4\rangle}+u_1^{\langle 4\rangle}), c(X_2^{\langle 4\rangle}+u_2^{\langle 4\rangle}), c(X_3^{\langle 4\rangle}+u_3^{\langle 4\rangle}), c(X_4^{\langle 4\rangle}+u_4^{\langle 4\rangle})\} \tag{7.54}$$

Regarding reliability: The leader party P_5 calculates the following expressions,

$$\begin{aligned}\text{Element 1}: \quad & c(X_1^{\langle 1\rangle}+u_1^{\langle 1\rangle})+c(X_1^{\langle 2\rangle}+u_1^{\langle 2\rangle})+c(X_1^{\langle 3\rangle}+u_1^{\langle 3\rangle})+c(X_1^{\langle 4\rangle}+u_1^{\langle 4\rangle}) \\ & = c(\sum_{i\in[4]} X_1^{\langle i\rangle}+\sum_{i\in[4]} u_1^{\langle i\rangle}) = c(\sum_{i\in[4]} X_1^{\langle i\rangle}) = c\cdot 4 \neq 0 \end{aligned} \tag{7.55}$$

[5]The common randomness symbol $u_\alpha^{\langle i\rangle}$ in this MP-PSU implementation functions exactly in the same way as the common randomness symbols $t_{i,j}$ functioned in the MP-PSI in Chapter 4. The same is true for subsequent common randomness symbols $w_\alpha^{\langle i\rangle}$ in the write-back implementation.

$$\text{Element 2}: \quad c(X_2^{\langle 1\rangle}+u_2^{\langle 1\rangle})+c(X_2^{\langle 2\rangle}+u_2^{\langle 2\rangle})+c(X_2^{\langle 3\rangle}+u_2^{\langle 3\rangle})+c(X_2^{\langle 4\rangle}+u_2^{\langle 4\rangle})$$

$$= c(\sum_{i\in[4]} X_2^{\langle i\rangle}+\sum_{i\in[4]} u_2^{\langle i\rangle}) = c(\sum_{i\in[4]} X_2^{\langle i\rangle}) = c\cdot 0 = 0 \tag{7.56}$$

$$\text{Element 3}: \quad c(X_3^{\langle 1\rangle}+u_3^{\langle 1\rangle})+c(X_3^{\langle 2\rangle}+u_3^{\langle 2\rangle})+c(X_3^{\langle 3\rangle}+u_3^{\langle 3\rangle})+c(X_3^{\langle 4\rangle}+u_3^{\langle 4\rangle})$$

$$= c(\sum_{i\in[4]} X_3^{\langle i\rangle}+\sum_{i\in[4]} u_3^{\langle i\rangle}) = c(\sum_{i\in[4]} X_3^{\langle i\rangle}) = c\cdot 2 \neq 0 \tag{7.57}$$

$$\text{Element 4}: \quad c(X_4^{\langle 1\rangle}+u_4^{\langle 1\rangle})+c(X_4^{\langle 2\rangle}+u_4^{\langle 2\rangle})+c(X_4^{\langle 3\rangle}+u_4^{\langle 3\rangle})+c(X_4^{\langle 4\rangle}+u_4^{\langle 4\rangle})$$

$$= c(\sum_{i\in[4]} X_4^{\langle i\rangle}+\sum_{i\in[4]} u_4^{\langle i\rangle}) = c(\sum_{i\in[4]} X_4^{\langle i\rangle}) = c\cdot 2 \neq 0 \tag{7.58}$$

Thus, P_5 concludes that the union $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3 \cup \mathcal{P}_4$ is $\{1, 3, 4\}$ because the first, third and fourth expressions are not equal to 0.

Regarding privacy: The leader party's privacy constraint is trivially satisfied since the leader party is empty and sent no queries to the clients. The clients' privacies are protected by the common randomness symbols $\{u_\alpha^{\langle [4]\rangle} : \alpha \in [4]\}$ and c. First, the values of the individual components of the incidence vector $\{X_\alpha^{\langle [4]\rangle} : \alpha \in [4]\}$ are kept private from P_5 by the added randomness symbols $\{u_\alpha^{\langle [4]\rangle} : \alpha \in [4]\}$. These *coupled* (i.e., correlated) random variables disappear when the components coming from clients are added up as $\sum_{i\in[4]} u_\alpha^{\langle i\rangle}$ is always 0 for all $\alpha \in [4]$. Finally, the global common randomness symbol c protects the actual value of the sum $\sum_i X_\alpha^{\langle i\rangle}$ for all α. That is, leader P_5 can only learn whether these sums are zero or not and nothing beyond that. Thus, this is a valid scheme satisfying reliability and privacy.

Example 14: Five-party PSU; one-database per client; two-database leader: With respect to the MP-PSU configuration in Example 13, we only change

the number of databases in the leader party P_5, which now contains two replicated and non-colluding databases. As these two databases do not communicate with each other directly, a straightforward approach could be to have each client send its answer shown in (7.51)-(7.54) to both databases in the leader party. This way, each database could individually learn the union while the privacy constraints are still satisfied. Here, we put forth an alternative approach, where two random client parties are utilized as intermediaries to route the information between the two non-colluding databases in the leader party such that there is no need for a client to send the replicated answer to both databases in P_5. To that end, each client party also holds another duplicate set of common randomness symbols $\{u_\alpha : \alpha \in [4]\}$ that are all uniformly selected from $\mathbb{F}_5$ on the basis of the existing common randomness symbols $\{u_\alpha^{\langle [4] \rangle} : \alpha \in [4]\}$. Since P_5 does not have any element, there is no need for the other parties to download any information from P_5 in the beginning, i.e., $D_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}$ and $D_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}$ are both null. At this point, let the client parties P_1 and P_2 form the first group. They send their respective answers $A_{U,1}^{\langle 1 \rangle,(1)}$ and $A_{U,1}^{\langle 2 \rangle,(1)}$ as shown in (7.51)-(7.52) to the first database of P_5 since they are associated with database 1. This database produces a response $D_{U,2}^{\langle 2 \rangle,(1)}$ to be downloaded by client 2 through element-wisely adding its received answers and appending leader party common randomness symbols that are all uniformly selected from the finite field $\mathbb{F}_5$,

$$\begin{aligned} D_{U,2}^{\langle 2 \rangle,(1)} = \{ & c(X_1^{\langle 1 \rangle} + X_1^{\langle 2 \rangle} + u_1^{\langle 1 \rangle} + u_1^{\langle 2 \rangle}) + S_1, c(X_2^{\langle 1 \rangle} + X_2^{\langle 2 \rangle} + u_2^{\langle 1 \rangle} + u_2^{\langle 2 \rangle}) + S_2, \\ & c(X_3^{\langle 1 \rangle} + X_3^{\langle 2 \rangle} + u_3^{\langle 1 \rangle} + u_3^{\langle 2 \rangle}) + S_3, c(X_4^{\langle 1 \rangle} + X_4^{\langle 2 \rangle} + u_4^{\langle 1 \rangle} + u_4^{\langle 2 \rangle}) + S_4 \} \end{aligned} \tag{7.59}$$

This database then sends this response back to P_2. After adding extra common randomness to the received response, P_2 forwards the following answer to both databases in P_5,

$$A_{U,2}^{\langle 2\rangle,([2])} = \{c(X_1^{\langle 1\rangle}+X_1^{\langle 2\rangle}+u_1^{\langle 1\rangle}+u_1^{\langle 2\rangle})+u_1+S_1, c(X_2^{\langle 1\rangle}+X_2^{\langle 2\rangle}+u_2^{\langle 1\rangle}+u_2^{\langle 2\rangle})+u_2+S_2,$$
$$c(X_3^{\langle 1\rangle}+X_3^{\langle 2\rangle}+u_3^{\langle 1\rangle}+u_3^{\langle 2\rangle})+u_3+S_3, c(X_4^{\langle 1\rangle}+X_4^{\langle 2\rangle}+u_4^{\langle 1\rangle}+u_4^{\langle 2\rangle})+u_4+S_4\} \tag{7.60}$$

Meanwhile, the client parties P_3 and P_4, which form the second group, send their respective answers $A_{U,1}^{\langle 3\rangle,(2)}$ and $A_{U,1}^{\langle 4\rangle,(2)}$ as shown in (7.53)-(7.54) to the second database of P_5. Similarly, this database produces a response $D_{U,2}^{\langle 3\rangle,(2)}$ to be downloaded by client 3 as follows, and sends it back to P_3,

$$D_{U,2}^{\langle 3\rangle,(2)} = \{c(X_1^{\langle 3\rangle}+X_1^{\langle 4\rangle}+u_1^{\langle 3\rangle}+u_1^{\langle 4\rangle})-S_1, c(X_2^{\langle 3\rangle}+X_2^{\langle 4\rangle}+u_2^{\langle 3\rangle}+u_2^{\langle 4\rangle})-S_2,$$
$$c(X_3^{\langle 3\rangle}+X_3^{\langle 4\rangle}+u_3^{\langle 3\rangle}+u_3^{\langle 4\rangle})-S_3, c(X_4^{\langle 3\rangle}+X_4^{\langle 4\rangle}+u_4^{\langle 3\rangle}+u_4^{\langle 4\rangle})-S_4\} \tag{7.61}$$

Then, P_3 forwards the following further processed answer to both databases in P_5,

$$A_{U,2}^{\langle 3\rangle,([2])} = \{c(X_1^{\langle 3\rangle}+X_1^{\langle 4\rangle}+u_1^{\langle 3\rangle}+u_1^{\langle 4\rangle})-u_1-S_1, c(X_2^{\langle 3\rangle}+X_2^{\langle 4\rangle}+u_2^{\langle 3\rangle}+u_2^{\langle 4\rangle})-u_2-S_2,$$
$$c(X_3^{\langle 3\rangle}+X_3^{\langle 4\rangle}+u_3^{\langle 3\rangle}+u_3^{\langle 4\rangle})-u_3-S_3, c(X_4^{\langle 3\rangle}+X_4^{\langle 4\rangle}+u_4^{\langle 3\rangle}+u_4^{\langle 4\rangle})-u_4-S_4\} \tag{7.62}$$

After collecting the answers in the second communication step, each individual database j in P_5 finds the desired submodel union by element-wisely adding $A_{U,2}^{\langle 2\rangle,(j)}$

and $A_{U,2}^{\langle 3\rangle,(j)}$,

$$A_{U,2}^{\langle 2\rangle,(j)} + A_{U,2}^{\langle 3\rangle,(j)} = \left\{ c(\sum_{i\in[4]} X_1^{\langle i\rangle}), c(\sum_{i\in[4]} X_2^{\langle i\rangle}), c(\sum_{i\in[4]} X_3^{\langle i\rangle}), c(\sum_{i\in[4]} X_4^{\langle i\rangle}) \right\}, \quad \forall j \quad (7.63)$$

Regarding reliability: The MP-PSU reliability in the leader party P_5 is inherited from the MP-PSU reliability analysis in Example 13. Specifically, each individual database in P_5 can make the same analysis as shown in (7.55)-(7.58) for each element in the alphabet to derive the union $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3 \cup \mathcal{P}_4$. Also, P_5 can send this union result to any client party if needed.

Regarding privacy: The privacy analysis of the client parties P_1 and P_4 is trivial, since neither of them has received any information from the remaining parties. Regarding the client party P_2, due to the appended leader party common randomness $\{S_\alpha, \alpha \in [4]\}$, this party cannot learn anything about the incidence vector symbols in the remaining parties from its only received information $D_{U,2}^{\langle 2\rangle,(1)}$. This analysis also applies to the client party P_3. Regarding the leader party P_5, it is obvious that the received information $\{A_{U,1}^{\langle 1\rangle,(1)}, A_{U,1}^{\langle 2\rangle,(1)}, A_{U,2}^{\langle 2\rangle,(1)}, A_{U,2}^{\langle 3\rangle,(1)}\}$ in the first database and the received information $\{A_{U,1}^{\langle 3\rangle,(2)}, A_{U,1}^{\langle 4\rangle,(2)}, A_{U,2}^{\langle 2\rangle,(2)}, A_{U,2}^{\langle 3\rangle,(2)}\}$ in the second database, individually, contain less information about the incidence vector symbols in the client parties than the answer set $A_U^{\langle [4]\rangle}$ received by P_5 in Example 13. Therefore, the leader party P_5 can only learn the union and nothing beyond that. Thus, this is a valid MP-PSU scheme.

Next, we consider situations that are commonly encountered in practical implementations.

First, one or more client parties may drop-out during the MP-PSU process. For instance, P_1 may lose connection to P_5, in which case, the first database of P_5 will only receive the answer from P_2. The download produced in the original method now becomes,

$$D'^{\langle 2\rangle,(1)}_{U,2} = \{c(X_1^{\langle 2\rangle}+u_1^{\langle 2\rangle})+S_1, c(X_2^{\langle 2\rangle}+u_2^{\langle 2\rangle})+S_2, \\ c(X_3^{\langle 2\rangle}+u_3^{\langle 2\rangle})+S_3, c(X_4^{\langle 2\rangle}+u_4^{\langle 2\rangle})+S_4\} \tag{7.64}$$

It is easy to observe that the common randomness symbols $u_\alpha^{\langle [4]\rangle}$ in these two downloads cannot be cancelled completely as before. However, note that P_2 possesses the missing common randomness symbols incurred by P_1 drop-out. Hence, P_2 can add the required common randomness itself as long as it learns from database 1 that P_1 has dropped-out. Thus, the adjusted answer in the second step $A'^{\langle 2\rangle,([2])}_{U,2}$ is as follows and will be sent back to both databases in P_5,

$$A'^{\langle 2\rangle,([2])}_{U,2} = \{c(X_1^{\langle 2\rangle}+u_1^{\langle 1\rangle}+u_1^{\langle 2\rangle})+u_1+S_1, c(X_2^{\langle 2\rangle}+u_2^{\langle 1\rangle}+u_2^{\langle 2\rangle})+u_2+S_2, \\ c(X_3^{\langle 2\rangle}+u_3^{\langle 1\rangle}+u_3^{\langle 2\rangle})+u_3+S_3, c(X_4^{\langle 2\rangle}+u_4^{\langle 1\rangle}+u_4^{\langle 2\rangle})+u_4+S_4\} \tag{7.65}$$

Further, if P_2 loses its connection to P_5, the remaining active client party P_1 in the first client party group functions as a router. Since no one in the second client group drops-out, the download $D^{\langle 3\rangle,(2)}_{U,2}$ remains the same,

$$D^{\langle 3\rangle,(2)}_{U,2} = \{c(X_1^{\langle 3\rangle}+X_1^{\langle 4\rangle}+u_1^{\langle 3\rangle}+u_1^{\langle 4\rangle})-S_1, c(X_2^{\langle 3\rangle}+X_2^{\langle 4\rangle}+u_2^{\langle 3\rangle}+u_2^{\langle 4\rangle})-S_2,$$

$$c(X_3^{\langle 3\rangle}+X_3^{\langle 4\rangle}+u_3^{\langle 3\rangle}+u_3^{\langle 4\rangle})-S_3, c(X_4^{\langle 3\rangle}+X_4^{\langle 4\rangle}+u_4^{\langle 3\rangle}+u_4^{\langle 4\rangle})-S_4\} \quad (7.66)$$

The result forwarded by P_3 and received by the databases in P_5 remains the same,

$$\begin{aligned} A_{U,2}^{\langle 3\rangle,([2])} = \{&c(X_1^{\langle 3\rangle}+X_1^{\langle 4\rangle}+u_1^{\langle 3\rangle}+u_1^{\langle 4\rangle})-u_1-S_1, c(X_2^{\langle 3\rangle}+X_2^{\langle 4\rangle}+u_2^{\langle 3\rangle}+u_2^{\langle 4\rangle})-u_2-S_2, \\ &c(X_3^{\langle 3\rangle}+X_3^{\langle 4\rangle}+u_3^{\langle 3\rangle}+u_3^{\langle 4\rangle})-u_3-S_3, c(X_4^{\langle 3\rangle}+X_4^{\langle 4\rangle}+u_4^{\langle 3\rangle}+u_4^{\langle 4\rangle})-u_4-S_4\} \end{aligned} \quad (7.67)$$

We can now verify that both databases in P_5 can determine the union $\mathcal{P}_2 \cup \mathcal{P}_3 \cup \mathcal{P}_4$ without the participation of P_1.

Second, the answer $A_{U,1}^{\langle 1\rangle,(1)}$ generated by P_1 may arrive at database 1 in P_5 so late that database 1 may believe that P_1 has dropped-out. In such a case, the privacy in our MP-PSU is still preserved. If we look at the received information $\{A_{U,1}^{\langle 1\rangle,(1)}, A_{U,1}^{\langle 2\rangle,(1)}, A_{U,2}^{\prime\langle 2\rangle,(1)}, A_{U,2}^{\langle 3\rangle,(1)}\}$ in database 1 of P_5, no information about the incidence vector $X^{\langle 1\rangle}$ is leaked due to the existence of extra common randomness symbols $\{u_\alpha : \alpha \in [4]\}$. Moreover, this late answer $A_{U,1}^{\langle 1\rangle,(1)}$ will never be transmitted to any other client parties by P_5 in order to avoid the further leakage of $X^{\langle 1\rangle}$. The usage of extra common randomness u_α here is similar to the double-masking idea in [96] so as to resolve this late arrival problem, but in a very simple manner.

Third, one of the two databases in P_5 may also drop-out during the implementation. For instance, if database 2 drops-out, the same answers $\{A_{U,1}^{\langle 1\rangle,(1)}, A_{U,1}^{\langle 2\rangle,(1)}, A_{U,2}^{\langle 2\rangle,(1)}\}$ can still be received by database 1 in P_5 from P_1 and P_2, but $\{A_{U,2}^{\langle 3\rangle,(1)}\}$ cannot be received from the other client party group as usual.

The corresponding remedy is that the surviving database asks for the values of $\{c(u_\alpha^{\langle 3\rangle} + u_\alpha^{\langle 4\rangle}) : \alpha \in [4]\}$ from P_2 through one more communication step. In this way, it is easy to see that the first database in P_5 can derive the union $\mathcal{P}_1 \cup \mathcal{P}_2$ associated with the first client party group.

Example 15: Five-party PSU; one-database per client; two-database leader; together with FSL-write: Consider a distributed FSL problem involving a server consisting of two replicated and non-colluding databases and four selected clients in this round of FSL process. Each individual database stores 4 independent submodels each containing 2 i.i.d. symbols uniformly selected from a sufficiently large finite field $\mathbb{F}_q$, $q \geq 5$, i.e., $M_1 = [M_{1,1}, M_{1,2}], M_2 = [M_{2,1}, M_{2,2}], M_3 = [M_{3,1}, M_{3,2}], M_4 = [M_{4,1}, M_{4,2}]$ and some required server-side common randomness symbols. According to the clients' respective local training data, client 1 can be used to update the submodel 1, client 2 can be used to update the submodels $1, 3$, client 3 can be used to update the submodels $1, 4$ and client 4 can be used to update the submodels $1, 3, 4$, i.e., $\Gamma^{\langle 1\rangle} = \{1\}, \Gamma^{\langle 2\rangle} = \{1, 3\}, \Gamma^{\langle 3\rangle} = \{1, 4\}, \Gamma^{\langle 4\rangle} = \{1, 3, 4\}$. Both databases in the server can communicate with each client through a secure and authenticated channel. We further assume that the channels connected to database 1 have higher bandwidth than the ones connected to database 2 for clients $1, 2$ and it is the opposite for clients $3, 4$. Thus, the FSL-PSU phase is executed exactly as in the MP-PSU in Example 14, and each database in the server learns the desired submodel union $\Gamma = \{1, 3, 4\}$ when this phase is complete.

Due to the similarities between the formulations of FSL-PSU phase and the

FSL-write phase, we use the idea in Example 14 one more time to execute the FSL-write phase. Database 1 sends the submodels $1, 3, 4$ to client 1 and client 2, while database 2 sends the submodels $1, 3, 4$ to client 3 and client 4, i.e., the downloads $D_{W,1}^{\langle 1,2\rangle,(1)}$ and $D_{W,1}^{\langle 3,4\rangle,(2)}$ are both $\{M_1, M_3, M_4\}$. After receiving the desired submodels from the server, client 1 generates the increment $\left\{\Delta_{1,1}^{\langle 1\rangle}, \Delta_{1,2}^{\langle 1\rangle}\right\}$ for submodel 1, client 2 generates the increment $\left\{\Delta_{1,1}^{\langle 2\rangle}, \Delta_{1,2}^{\langle 2\rangle}, \Delta_{3,1}^{\langle 2\rangle}, \Delta_{3,2}^{\langle 2\rangle}\right\}$ for submodels $1, 3$, client 3 generates the increment $\left\{\Delta_{1,1}^{\langle 3\rangle}, \Delta_{1,2}^{\langle 3\rangle}, \Delta_{4,1}^{\langle 3\rangle}, \Delta_{4,2}^{\langle 3\rangle}\right\}$ for submodels $1, 4$, client 4 generates the increment $\left\{\Delta_{1,1}^{\langle 4\rangle}, \Delta_{1,2}^{\langle 4\rangle}, \Delta_{3,1}^{\langle 4\rangle}, \Delta_{3,2}^{\langle 4\rangle}, \Delta_{4,1}^{\langle 4\rangle}, \Delta_{4,2}^{\langle 4\rangle}\right\}$ for submodels $1, 3, 4$ after performing their respective local training. In addition, we assume that each client has already obtained two sets of common randomness symbols $\{w_{k,l}^{\langle \Gamma\rangle} : k \in [4], l \in [2]\}$ and $\{w_{k,l} : k \in [4], l \in [2]\}$ from the finite field $\mathbb{F}_q$ as in the previous examples. For all $k \in \Gamma$ and all $l \in [2]$, the sum $\sum_{i\in[4]} w_{k,l}^{\langle i\rangle}$ is always equal to 0. Thus, the answers sent to database 1 in the server from clients 1 and 2 are as follows,

$$A_{W,1}^{\langle 1\rangle,(1)} = \{\Delta_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 1\rangle}, \Delta_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 1\rangle}, w_{3,1}^{\langle 1\rangle}, w_{3,2}^{\langle 1\rangle}, w_{4,1}^{\langle 1\rangle}, w_{4,2}^{\langle 1\rangle}\} \tag{7.68}$$

$$A_{W,1}^{\langle 2\rangle,(1)} = \{\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 2\rangle}, \Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 2\rangle}, \Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 2\rangle}, \Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 2\rangle}, w_{4,1}^{\langle 2\rangle}, w_{4,2}^{\langle 2\rangle}\} \tag{7.69}$$

After collecting the answers from clients $1, 2$, database 1 performs the element-wise summation with the aid of its own server-side common randomness symbols and transmits the following response to client 2 in its associated client group,

$$D_{W,2}^{\langle 2\rangle,(1)} = \{\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+S_{1,1}, \Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+S_{1,2},$$
$$\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+S_{3,1}, \Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+S_{3,2},$$

$$w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+S_{4,1}, w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+S_{4,2}\} \tag{7.70}$$

Afterwards, client 2 processes the received response by adding extra common randomness and then forwards the following answer to both databases in the server,

$$\begin{aligned} A_{W,2}^{\langle 2\rangle,([2])} = \{&\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+w_{1,1}+S_{1,1}, \Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+w_{1,2}+S_{1,2},\\ &\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+w_{3,1}+S_{3,1}, \Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+w_{3,2}+S_{3,2},\\ &w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+w_{4,1}+S_{4,1}, w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+w_{4,2}+S_{4,2}\} \end{aligned} \tag{7.71}$$

At the same time, the answers sent to database 2 in the server from clients 3 and 4 are,

$$A_{W,1}^{\langle 3\rangle,(2)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}, \Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}, w_{3,1}^{\langle 3\rangle}, w_{3,2}^{\langle 3\rangle}, \Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}, \Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}\} \tag{7.72}$$

$$A_{W,1}^{\langle 4\rangle,(2)} = \{\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}, \Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}, \Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}, \Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}, \Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}, \Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}\} \tag{7.73}$$

When the collection and computation is finished, database 2 sends the following response to client 3 who belongs to its associated client group,

$$\begin{aligned} D_{W,2}^{\langle 3\rangle,(2)} = \{&\Delta_{1,1}^{\langle 3\rangle}+\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 4\rangle}-S_{1,1}, \Delta_{1,2}^{\langle 3\rangle}+\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 4\rangle}-S_{1,2},\\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 3\rangle}+w_{3,1}^{\langle 4\rangle}-S_{3,1}, \Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 3\rangle}+w_{3,2}^{\langle 4\rangle}-S_{3,2},\\ &\Delta_{4,1}^{\langle 3\rangle}+\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 4\rangle}-S_{4,1}, \Delta_{4,2}^{\langle 3\rangle}+\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 4\rangle}-S_{4,2}\} \end{aligned} \tag{7.74}$$

Similarly, client 3 processes the received response by adding extra common randomness again and then forwards the following answer to both databases in the server,

$$\begin{aligned} A_{W,2}^{\langle 3\rangle,([2])} = \{&\Delta_{1,1}^{\langle 3\rangle}+\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 4\rangle}-w_{1,1}-S_{1,1}, \Delta_{1,2}^{\langle 3\rangle}+\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 4\rangle}-w_{1,2}-S_{1,2},\\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 3\rangle}+w_{3,1}^{\langle 4\rangle}-w_{3,1}-S_{3,1}, \Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 3\rangle}+w_{3,2}^{\langle 4\rangle}-w_{3,2}-S_{3,2},\\ &\Delta_{4,1}^{\langle 3\rangle}+\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 4\rangle}-w_{4,1}-S_{4,1}, \Delta_{4,2}^{\langle 3\rangle}+\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 4\rangle}-w_{4,2}-S_{4,2}\} \end{aligned} \tag{7.75}$$

At this point, both databases can update the corresponding submodels after receiving the answers in the second step and removing all the involved common randomness symbols through element-wise summation,

$$M_1' = \{M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+\Delta_{1,1}^{\langle 3\rangle}+\Delta_{1,1}^{\langle 4\rangle}, M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+\Delta_{1,2}^{\langle 3\rangle}+\Delta_{1,2}^{\langle 4\rangle}\} \tag{7.76}$$

$$M_3' = \{M_{3,1}+\Delta_{3,1}^{\langle 2\rangle}+\Delta_{3,1}^{\langle 4\rangle}, M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+\Delta_{3,2}^{\langle 4\rangle}\} \tag{7.77}$$

$$M_4' = \{M_{4,1}+\Delta_{4,1}^{\langle 3\rangle}+\Delta_{4,1}^{\langle 4\rangle}, M_{4,2}+\Delta_{4,2}^{\langle 3\rangle}+\Delta_{4,2}^{\langle 4\rangle}\} \tag{7.78}$$

In this example, we note that the scheme used in the FSL-write phase is a simplified version of the MP-PSU scheme used in Example 14 without considering the global common randomness symbol c. Therefore, regarding this FSL-write scheme, we can readily verify the FSL-write reliability constraint as well as the FSL-write privacy constraint for each individual database at the server side in reference to the leader party P_5 in Example 14, and the FSL-write inter-client privacy constraint

for clients $2, 3$ in reference to the client parties P_2, P_3 in Example 14. Likewise, the robustness against client drop-outs, client late arrivals and database drop-outs possessed by this FSL-write scheme is also inherited from the one in Example 14. Finally, as required by the FSL process itself, the FSL-PSU phase and FSL-write phase introduced in this example can be executed repeatedly if a new set of clients are selected to perform another round of this FSL process. New sets of common randomness symbols and server-side common randomness symbols are needed to ensure privacy in each round.

7.5 General FSL Achievable Scheme

In this section, we describe our general achievable scheme for a distributed FSL model with any arbitrary initial parameters; see the model formulation in Section 7.2.3. Our general achievable scheme has three phases: common randomness generation phase (FSL-CRG), private determination of the union of indices of submodels to be updated (FSL-PSU), and private writing of the updated submodels in the union back to the databases (FSL-write). In Section 7.4, we have given examples of FSL-PSU, and combined FSL-PSU and FSL-write. The FSL-PSU and FSL-write phases make use of pre-established common randomness at the client side. In Section 7.4, we have presumed that the common randomness needed for FSL-PSU and FSL-write have already been established. In this section, we first describe the establishment of the necessary common randomness across the clients. Our FSL-CRG scheme exploits the distributed nature of the server databases, and

uses one-time pads and the RSPIR scheme introduced in [116] to generate common randomness. Then, we describe FSL-PSU and FSL-write for the most general case.

7.5.1 Common Randomness Generation (FSL-CRG) Phase

The two databases in the server aim to establish two types of common randomness across the clients: The first type is a global common randomness symbol c that is uniformly selected from the set $\mathbb{F}_q \backslash \{0\}$. The second type is a set of general common randomness symbols $\{R_0, R_1, \ldots, R_C\}$ with a flexible set length $C+1$, where each symbol is uniformly selected from $\mathbb{F}_q$ and the sum of the last C symbols is equal to 0, i.e., $\sum_{i\in[C]} R_i = 0$. As a result, R_0 can be used as u_k or $w_{k,l}$ while $R_{[C]}$ can be used as $u_k^{\langle [C] \rangle}$ or $w_{k,l}^{\langle [C] \rangle}$ in the next two phases. The FSL-CRG phase is independent of the FSL-PSU and FSL-write phases in a practical implementation, and therefore, can be potentially executed during the off-peak hours.

We start with a scheme for the second type of common randomness allocation. First, each database individually selects a random client from its client group as routing clients. Their indices are denoted by θ_1 and θ_2, respectively. Second, both databases randomly select a set of symbols with size C from $\mathbb{F}_q$ under a uniform distribution, and then broadcast this set to the routing clients and the last client. Thus, these clients obtain a new set of symbols with size C through element-wise summation, and then append one more symbol R_C to the existing set such that the sum of the last C symbols equals zero. Moreover, each database also sends its $i+1$th random symbol to client i for all $i \in [C-1]$. Thus, client i can obtain the

symbol R_i through summation. Each individual database has no knowledge about client-side common randomness because of the one-time pad encryption.

We next consider the first type of common randomness allocation, i.e., the allocation of c. Note that c needs to be uniform in $\mathbb{F}_q \backslash \{0\}$. We could use a modified version of the above method as follows: Each database individually selects a random symbol from $\mathbb{F}_q$ under a uniform distribution and then broadcasts its selected symbol to each client. The global common randomness symbol c is calculated as the sum of the two random values that are transmitted from the two databases. However, c now can take the value 0 with probability $\frac{1}{|q|}$, thus, the constraint that c is uniformly distributed over $\mathbb{F}_q \backslash \{0\}$ is not immediately satisfied. The two databases in the server can repeat this procedure until c falls into the allowed region, which would require feedback from the clients as explained above.

In order to overcome this shortcoming, we propose a novel common randomness allocation method via a broadcast variation of the RSPIR scheme introduced in [116]. We consider a RSPIR problem with $N = 2, K = |q| - 1, L = 1$ and make use of the potentially suboptimal[6] RSPIR achievable scheme provided in [116, Section V]. The corresponding message set stored in the two databases is set as $W_1 = 1, W_2 = 2, \ldots, W_{|q|-1} = |q| - 1$. We note that there is no need to protect the privacy of the remaining messages at this point since all these messages can be globally known to the clients. As a consequence, the required server-side common randomness in the original RSPIR problem [116] can be discarded.[7] Database 1 has

[6]This scheme was proved to be optimal for $K = 2, 3$, but is a valid scheme for any K.

[7]The new idea proposed here seems to be closer to the definition of random private information retrieval (RPIR) as opposed to random *symmetric* private information retrieval (RSPIR) studied in [116]. RPIR and RSPIR can be further studied to design more efficient and powerful com-

the following set of messages and broadcasts one of them randomly to active clients,

$$c_1^{(1)} = \emptyset \tag{7.79}$$

$$c_2^{(1)} = (W_1 + W_2, W_2 + W_3, \ldots, W_{|q|-2} + W_{|q|-1}) \tag{7.80}$$

$$c_3^{(1)} = (W_1 + W_3, W_2 + W_4, \ldots, W_{|q|-2} + W_1) \tag{7.81}$$

$$\vdots$$

$$c_{|q|-1}^{(1)} = (W_1 + W_{|q|-1}, W_2 + W_1, \ldots, W_{|q|-2} + W_{|q|-3}) \tag{7.82}$$

Similarly, database 2 has the following set of messages and broadcasts one of them randomly to active clients,

$$c_1^{(2)} = W_1 \tag{7.83}$$

$$c_2^{(2)} = W_2 \tag{7.84}$$

$$\vdots$$

$$c_{|q|-2}^{(2)} = W_{|q|-2} \tag{7.85}$$

$$c_{|q|-1}^{(2)} = W_{|q|-1} \tag{7.86}$$

By applying the decoding procedure in the RSPIR approach in [116], all clients will be able to obtain the same global common randomness symbol c that is uniformly distributed over the set $\mathbb{F}_q \backslash \{0\}$. Moreover, the obtained random symbol c will be unknown to each individual database in the server due to the user privacy constraint

mon randomness construction among the clients. We leave this as an interesting future research direction.

in RSPIR [116].

7.5.2 Private Set Union (FSL-PSU) Phase

After the FSL-CRG phase is completed, each selected client will obtain a global common randomness symbol c that is uniformly distributed over $\mathbb{F}_q \backslash \{0\}$, and a set of common randomness symbols $\{u_k^{\langle [C] \rangle} : k \in [K]\}$ in which the identity $\sum_{i \in [C]} u_k^{\langle i \rangle} = 0$ is true for all $k \in [K]$, and another set of common randomness symbols $\{u_k : k \in [K]\}$ that are all uniformly distributed over $\mathbb{F}_q$. Following our distributed FSL model in Section 7.2.3, C selected clients in this round of FSL process are divided into two groups $\mathcal{C}_1$ and $\mathcal{C}_2$. The clients in $\mathcal{C}_1$ send their answers to database 1 in the server, while the clients in $\mathcal{C}_2$ send their answers to database 2 in the server. Then, the ith client in $\mathcal{C}_1$, after mapping its desired submodel index set $\Gamma^{\langle i \rangle}$ into a corresponding incidence vector $Y^{\langle i \rangle}$, constructs its answer as,

$$A_{U,1}^{\langle i \rangle,(1)} = \{c(Y_1^{\langle i \rangle} + u_1^{\langle i \rangle}), c(Y_2^{\langle i \rangle} + u_2^{\langle i \rangle}), \ldots, c(Y_K^{\langle i \rangle} + u_K^{\langle i \rangle})\} \tag{7.87}$$

Similarly, the ith client in $\mathcal{C}_2$ constructs its answer as,

$$A_{U,1}^{\langle i \rangle,(2)} = \{c(Y_1^{\langle i \rangle} + u_1^{\langle i \rangle}), c(Y_2^{\langle i \rangle} + u_2^{\langle i \rangle}), \ldots, c(Y_K^{\langle i \rangle} + u_K^{\langle i \rangle})\} \tag{7.88}$$

Once the first database in the server receives all the answers from its associated clients in $\mathcal{C}_1$, it produces a corresponding response to be downloaded as,

$$D_{U,2}^{\langle\theta_1\rangle,(1)} = \left\{ c \sum_{i\in\mathcal{C}_1} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) + S_k \colon k \in [K] \right\} \tag{7.89}$$

where $\{S_k \colon k \in [K]\}$ are shared server-side common randomness symbols that are uniformly selected from $\mathbb{F}_q$ as well. This produced response $D_{U,2}^{\langle\theta_1\rangle,(1)}$ will then be downloaded by a random client whose index θ_1 belongs to $\mathcal{C}_1$. Afterwards, the θ_1th client processes the received response by adding extra common randomness to it, and then, forwards the following answer to both databases in the server,

$$A_{U,2}^{\langle\theta_1\rangle,([2])} = \left\{ c \sum_{i\in\mathcal{C}_1} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) + u_k + S_k \colon k \in [K] \right\} \tag{7.90}$$

Likewise, the second database produces a response to be downloaded as follows, after receiving all the answers in the first step from the client group $\mathcal{C}_2$,

$$D_{U,2}^{\langle\theta_2\rangle,(2)} = \left\{ c \sum_{i\in\mathcal{C}_2} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) - S_k \colon k \in [K] \right\} \tag{7.91}$$

This produced response will then be downloaded by a random client in $\mathcal{C}_2$ whose index is denoted by θ_2. Afterwards, like the θ_1th client, this client also forwards the following further processed answer to both databases in the server,

$$A_{U,2}^{\langle\theta_2\rangle,([2])} = \left\{ c \sum_{i\in\mathcal{C}_2} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) - u_k - S_k \colon k \in [K] \right\} \tag{7.92}$$

After collecting these two answer sets in the second communication round, each individual database j in the server is ready to derive the desired submodel union Γ by performing the following element-wise summation,

$$\begin{aligned} A_{U,2}^{\langle\theta_1\rangle,(j)} + A_{U,2}^{\langle\theta_2\rangle,(j)} = \Bigg\{ & c\sum_{i\in\mathcal{C}_1}(Y_1^{\langle i\rangle}+u_1^{\langle i\rangle})+u_1+S_1+c\sum_{i\in\mathcal{C}_2}(Y_1^{\langle i\rangle}+u_1^{\langle i\rangle})-u_1-S_1, \\ & c\sum_{i\in\mathcal{C}_1}(Y_2^{\langle i\rangle}+u_2^{\langle i\rangle})+u_2+S_2+c\sum_{i\in\mathcal{C}_2}(Y_2^{\langle i\rangle}+u_2^{\langle i\rangle})-u_2-S_2, \\ & \quad\vdots \\ & c\sum_{i\in\mathcal{C}_1}(Y_K^{\langle i\rangle}+u_K^{\langle i\rangle})+u_K+S_K+c\sum_{i\in\mathcal{C}_2}(Y_K^{\langle i\rangle}+u_K^{\langle i\rangle})-u_K-S_K \Bigg\} \end{aligned} \tag{7.93}$$

$$= \left\{ c\sum_{i\in[C]} Y_1^{\langle i\rangle}, c\sum_{i\in[C]} Y_2^{\langle i\rangle}, \ldots, c\sum_{i\in[C]} Y_K^{\langle i\rangle} \right\} \tag{7.94}$$

FSL-PSU reliability: Each individual database j in the server makes use of the K elements in $A_{U,2}^{\langle\theta_1\rangle,(j)} + A_{U,2}^{\langle\theta_2\rangle,(j)}$ to decide whether an arbitrary element in the set $[K]$ is in the ultimate submodel index union, and thereby, to determine Γ. Let us use an arbitrary index k as an example to analyze the statement above. On the one hand, if any client's desired submodel index set includes k, the sum $\sum_{i\in[C]} Y_k^{\langle i\rangle}$ must be a value that is not zero and the expression $c\sum_{i\in[C]} Y_k^{\langle i\rangle}$ must be in $\mathbb{F}_q\backslash\{0\}$. On the other hand, if none of these clients' desired submodel index set includes k, the sum $\sum_{i\in[C]} Y_k^{\langle i\rangle}$ and its associated expression $c\sum_{i\in[C]} Y_k^{\langle i\rangle}$ are both equal to zero. Therefore, each database utilizes the value of its calculated expression $c\sum_{i\in[C]} Y_k^{\langle i\rangle}$ (whether it is zero or not) to judge whether the index k is in the union Γ or not. Following the same analysis for each $k \in [K]$, both databases can ultimately obtain

the correct submodel union Γ. Thus, the FSL-PSU reliability is satisfied.

FSL-PSU privacy: We analyze the FSL-PSU privacy based on the availability of the answer sets $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}\}$. For all $i \in [C]$ and all $k \in \Gamma$, the common randomness $u_k^{\langle i \rangle}$ is used to protect the privacy of $Y_k^{\langle i \rangle}$ such that each database knows nothing about the value of $Y_k^{\langle i \rangle}$ because of the one-time pad encryption. Further, for all $k \in \Gamma$, the common randomness c is used to protect the privacy of $\sum_{i\in[C]} Y_k^{\langle i \rangle}$ such that each database knows nothing about the value of $\sum_{i\in[C]} Y_k^{\langle i \rangle}$ beyond that this sum is zero or not because of the finite cyclic group under multiplication in $\mathbb{F}_q \backslash \{0\}$. Hence, the FSL-PSU privacy is preserved when the answer sets $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}\}$ are received by each database. The concrete proof follows from the proof of client's privacy in [74, Subsection V.B]. In reality, the received answer set in database 1 is $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,2}^{\langle \theta_1 \rangle,(1)}, A_{U,2}^{\langle \theta_2 \rangle,(1)}\}$, which is equivalent to $A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}$ and $\{c \sum_{i\in[C]} Y_k^{\langle i \rangle} : k \in [K]\}$ because of the unknown extra common randomness $\{u_k \colon k \in [K]\}$. Meanwhile, the received answer in database 2 is $\{A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}, A_{U,2}^{\langle \theta_1 \rangle,(2)}, A_{U,2}^{\langle \theta_2 \rangle,(2)}\}$, which is equivalent to $A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}$ and $\{c \sum_{i\in[C]} Y_k^{\langle i \rangle} : k \in [K]\}$ for the same reason. That means that each database receives less information with respect to the incidence vectors $Y^{\langle [C] \rangle}$ than the answer set $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}\}$. Thus, the FSL-PSU privacy constraint is satisfied.

FSL-PSU inter-client privacy: Only the clients θ_1 and θ_2 receive information from outside. Due to the existence of the unknown server-side common randomness in the downloads $D_{U,2}^{\langle \theta_1 \rangle,(1)}$ and $D_{U,2}^{\langle \theta_2 \rangle,(2)}$, neither the θ_1th client nor the θ_2th client can

learn any knowledge about the incidence vector within the other clients. Therefore, the FSL-PSU inter-client privacy constraint is satisfied as well.

FSL-PSU communication cost: Without considering the communication cost generated in the FSL-CRG phase, the communication cost in this phase is $(C+6)K$ in q-ary bits. Moreover, following the common randomness generation approach provided in Section 7.5.1, the extra communication cost is about $8CK$ for the required client-side common randomness. Further, for the global common randomness symbol c, the required communication cost is approximately $2(|q|-1)C$ in q-ary bits, which is negligible since the value of K is generally very large. Therefore, the total communication cost in this phase is $(9C+6)K$ in q-ary bits.

FSL-PSU client drop-out robustness: In the first step of FSL-PSU phase, for all $i \in [C]$, client i sends its generated answer to its associated database in the server. Without loss of generality, we assume that a set of clients $\mathcal{C}_{1,\mathcal{D}}$ belonging to the first client group $\mathcal{C}_1$ and another set of clients $\mathcal{C}_{2,\mathcal{D}}$ belonging to the second client group $\mathcal{C}_2$ drop-out in this step. Hence, the response to be downloaded produced by database 1 is as follows,

$$D_{U,2}^{\prime\langle\theta_1\rangle,(1)} = \left\{ c \sum_{i\in\mathcal{C}_1\setminus\mathcal{C}_{1,\mathcal{D}}} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) + S_k \colon k \in [K] \right\} \tag{7.95}$$

After receiving the response $D_{U,2}^{\prime\langle\theta_1\rangle,(1)}$ as well as the index set of out-of-operation clients $\mathcal{C}_{1,\mathcal{D}}$, client θ_1 can adjust the answer by additionally appending the sum of missing common randomness symbols $c\sum_{i\in\mathcal{C}_{1,\mathcal{D}}} u_k^{\langle i\rangle}$ for all $k \in [K]$. Hence, the

answer generated by client θ_1 in the second step is as follows,

$$A_{U,2}^{\prime\langle\theta_1\rangle,([2])} = \left\{ c \sum_{i\in\mathcal{C}_1\setminus\mathcal{C}_{1,\mathcal{D}}} Y_k^{\langle i\rangle} + c \sum_{i\in\mathcal{C}_1} u_k^{\langle i\rangle} + u_k + S_k \colon k \in [K] \right\} \tag{7.96}$$

Likewise, the response to be downloaded produced by database 2 is as follows,

$$D_{U,2}^{\prime\langle\theta_2\rangle,(2)} = \left\{ c \sum_{i\in\mathcal{C}_2\setminus\mathcal{C}_{2,\mathcal{D}}} (Y_k^{\langle i\rangle} + u_k^{\langle i\rangle}) + S_k \colon k \in [K] \right\} \tag{7.97}$$

The answer generated by client θ_2 in the second step is as follows,

$$A_{U,2}^{\prime\langle\theta_2\rangle,([2])} = \left\{ c \sum_{i\in\mathcal{C}_2\setminus\mathcal{C}_{2,\mathcal{D}}} Y_k^{\langle i\rangle} + c \sum_{i\in\mathcal{C}_2} u_k^{\langle i\rangle} - u_k - S_k \colon k \in [K] \right\} \tag{7.98}$$

After collecting the answers $A_{U,2}^{\prime\langle\theta_1\rangle,(j)}$ and $A_{U,2}^{\prime\langle\theta_2\rangle,(j)}$, by adding them up element-wisely, each individual database j in the server will obtain the union result as $\Gamma_{\mathcal{C}_1\setminus\mathcal{C}_{1,\mathcal{D}}} \cup \Gamma_{\mathcal{C}_2\setminus\mathcal{C}_{2,\mathcal{D}}}$ containing all the active selected clients following the steps in the FSL-PSU reliability constraints. Another non-trivial point is that the randomly selected clients θ_1 and θ_2 may also drop-out during the implementation of FSL-PSU phase step 2. In a practical application, a potential solution is that each database individually randomly selects a small set of clients to route the information in parallel like the clients θ_1 and θ_2. Further, we may use the observed client drop-out rate to determine the cardinality of this small relaying set.

FSL-PSU client late-arrival robustness: Without loss of generality, we assume that an answer generated by an arbitrary client with index $i \in \mathcal{C}_j$ in the first

step arrives at database j late. Even though database j receives the information $A_{U,1}^{\langle i\rangle,(j)}$ separately, it is still not able to extract any information about the incidence vector $Y^{\langle i\rangle}$ from the received answers in the two steps of FSL-PSU phase because of the unknown extra common randomness $\{u_k : k \in [K]\}$. This conclusion can be extended to a set of arbitrary clients who arrive at the same database late. Moreover, it is easy to guarantee that this late answer will never be transmitted to any other client in order to avoid information leakage.

FSL-PSU database drop-out robustness: If database 1 drops-out and cannot function normally, database 2 can still receive the answers $A_{U,1}^{\langle \mathcal{C}_2\rangle,(2)}$ and $A_{U,2}^{\langle \theta_2\rangle,(2)}$ as normal but cannot receive any answer from the relaying client in $\mathcal{C}_1$. In order to derive the union $\Gamma_{\mathcal{C}_2}$ through decoding the set $\{c\sum_{i\in\mathcal{C}_2} Y_k^{\langle i\rangle} : k \in [K]\}$ from its existing information, database 2 needs to communicate with client θ_2 one more time for the sake of the values of $\{c\sum_{i\in\mathcal{C}_1} u_k^{\langle i\rangle} : k \in [K]\}$. Likewise, if database 2 cannot function normally, this time, database 1 can still derive the union $\Gamma_{\mathcal{C}_1}$ following the same way. Further, if we encounter client drop-out or client late-arrival in addition to the occurrence of database drop-out, the last two robustness analyses can be utilized accordingly to make this scheme function well.

7.5.3 Private Write (FSL-write) Phase

When the FSL-PSU phase is complete, the server learns the desired submodel union Γ from all the selected clients in this FSL round. Then, each database in the server individually sends the set of submodels M_Γ to its associated clients. From

the FSL-CRG phase preceding the FSL-PSU phase, each selected client has also obtained two sets of common randomness symbols $\{w_{k,l}^{\langle [C] \rangle} : k \in \Gamma, l \in [L]\}$ and $\{w_{k,l} \colon k \in \Gamma, l \in [L]\}$ from $\mathbb{F}_q$. Likewise, we always have $\sum_{i \in [C]} w_{k,l}^{\langle i \rangle} = 0$ for all $k \in \Gamma$ and $l \in [L]$. Therefore, the ith client in $\mathcal{C}_1$ will generate the increments for each desired submodel whose index belongs to $\Gamma^{\langle i \rangle}$ according to its local training data and then construct a well-processed answer accordingly. Specifically, for all $k \in \Gamma^{\langle i \rangle}$, the answer symbols are generated in the following form,

$$A_{W,1}^{\langle i \rangle,(1)}(k) = \{\Delta_{k,1}^{\langle i \rangle} + w_{k,1}^{\langle i \rangle}, \Delta_{k,2}^{\langle i \rangle} + w_{k,2}^{\langle i \rangle}, \ldots, \Delta_{k,L}^{\langle i \rangle} + w_{k,L}^{\langle i \rangle}\} \tag{7.99}$$

In addition, for all $k \in \Gamma \backslash \Gamma^{\langle i \rangle}$, the answer symbols are generated as follows without any updates concerning the current submodel,

$$A_{W,1}^{\langle i \rangle,(1)}(k) = \{w_{k,1}^{\langle i \rangle}, w_{k,2}^{\langle i \rangle}, \ldots, w_{k,L}^{\langle i \rangle}\} \tag{7.100}$$

Thus, the ultimate answer generated by this client in the first step is $A_{W,1}^{\langle i \rangle,(1)} = \{A_{W,1}^{\langle i \rangle,(1)}(k) \colon k \in \Gamma\}$. The ith client in $\mathcal{C}_2$ will generate an ultimate answer $A_{W,1}^{\langle i \rangle,(2)} = \{A_{W,1}^{\langle i \rangle,(2)}(k) \colon k \in \Gamma\}$ in the same way, where

$$A_{W,1}^{\langle i \rangle,(2)}(k) = \{\Delta_{k,1}^{\langle i \rangle} + w_{k,1}^{\langle i \rangle}, \Delta_{k,2}^{\langle i \rangle} + w_{k,2}^{\langle i \rangle}, \ldots, \Delta_{k,L}^{\langle i \rangle} + w_{k,L}^{\langle i \rangle}\}, \quad k \in \Gamma^{\langle i \rangle} \tag{7.101}$$

$$A_{W,1}^{\langle i \rangle,(2)}(k) = \{w_{k,1}^{\langle i \rangle}, w_{k,2}^{\langle i \rangle}, \ldots, w_{k,L}^{\langle i \rangle}\}, \quad k \in \Gamma \backslash \Gamma^{\langle i \rangle} \tag{7.102}$$

Subsequently, each client sends its answer to its associated database in the server. These two databases also share another set of server-side common randomness symbols $\{S_{k,l} : k \in [K], l \in [L]\}$ from $\mathbb{F}_q$. Let $C_k^{(1)}$ be the index set of clients in the first client group $\mathcal{C}_1$ whose desired submodel index set includes the index k, i.e., $C_k^{(1)} = \{i \in \mathcal{C}_1 | k \in \Gamma^{\langle i \rangle}\}$. Similarly, $C_k^{(2)}$ and C_k are defined as $\{i \in \mathcal{C}_2 | k \in \Gamma^{\langle i \rangle}\}$ and $\{i \in [C] | k \in \Gamma^{\langle i \rangle}\}$, respectively. After collecting all the answers $A_{W,1}^{\langle \mathcal{C}_1 \rangle,(1)}$ from $\mathcal{C}_1$, database 1 calculates the following aggregation increment for the lth symbol of the kth submodel where k belongs to the union set Γ,

$$\sum_{i \in C_k^{(1)}} \left(\Delta_{k,l}^{\langle i \rangle} + w_{k,l}^{\langle i \rangle} \right) + \sum_{i \in \mathcal{C}_1 \setminus C_k^{(1)}} w_{k,l}^{\langle i \rangle} = \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} \tag{7.103}$$

As in the last FSL-PSU phase, after adding server-side common randomness, the corresponding response is produced as follows and will be downloaded by the client θ_1,

$$D_{W,2}^{\langle \theta_1 \rangle,(1)} = \left\{ \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} + S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{7.104}$$

Once this response is received by client θ_1, this client only adds extra common randomness and then forwards the following answer to both databases,

$$A_{W,2}^{\langle \theta_1 \rangle,([2])} = \left\{ \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} + w_{k,l} + S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{7.105}$$

Meanwhile, after collecting all the answers $A_{W,1}^{\langle \mathcal{C}_2 \rangle,(2)}$ from $\mathcal{C}_2$, database 2 produces the following response and this response will be downloaded by the client θ_2,

$$D_{W,2}^{\langle \theta_2 \rangle,(2)} = \left\{ \sum_{i \in C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_2} w_{k,l}^{\langle i \rangle} - S_{k,l} \colon k \in \Gamma, l \in [L] \right\} \tag{7.106}$$

The answer that is forwarded by client θ_2 to both databases is as follows,

$$A_{W,2}^{\langle \theta_2 \rangle,([2])} = \left\{ \sum_{i \in C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_2} w_{k,l}^{\langle i \rangle} - w_{k,l} - S_{k,l} \colon k \in \Gamma, l \in [L] \right\} \tag{7.107}$$

At this point, each individual database in the server is ready to aggregate the updates as desired from all the selected clients in this round of FSL. For the lth symbol of the kth submodel in M_Γ, the ultimate aggregation increment is calculated as follows,

$$\begin{aligned} &\sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} + w_{k,l} + S_{k,l} + \sum_{i \in C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_2} w_{k,l}^{\langle i \rangle} - w_{k,l} - S_{k,l} \\ &= \sum_{i \in C_k^{(1)} \cup C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1 \cup \mathcal{C}_2} w_{k,l}^{\langle i \rangle} \qquad (7.108) \\ &= \sum_{i \in C_k} \Delta_{k,l}^{\langle i \rangle} \qquad (7.109) \end{aligned}$$

The updated lth symbol of the kth submodel $M'_{k,l}$ stored in the server after performing this round of FSL-write should finally be

$$M'_{k,l} = M_{k,l} + \sum_{i \in C_k} \Delta_{k,l}^{\langle i \rangle} \tag{7.110}$$

It is clear that this scheme satisfies the FSL-write reliability constraint. It is important to note that the scheme in FSL-write phase is essentially a repetitive application of a simplified version of the scheme in FSL-PSU phase without involving the global common randomness symbol c. Thus, the FSL-write scheme satisfies the FSL-write privacy constraint as well as FSL-write inter-client privacy constraints, and also is robust against client drop-out, client late-arrival and database drop-out events.

FSL-write communication cost: If we also do not consider the communication cost generated in the accompanying common randomness generation, the communication cost is $(2C+6)|\Gamma|L$ in q-ary bits in which $C|\Gamma|L$ is for the clients to download the submodels from the server. The communication cost of obtaining the required client-side common randomness sets is $8C|\Gamma|L$ in q-ary bits. Therefore, the total communication cost in this phase is $(10C+6)|\Gamma|L$ in q-ary bits.

The complete procedure involving FSL-PSU phase and FSL-write phase in this round of FSL process can be executed repeatedly to update the full learning model iteratively until a pre-specified termination criterion is met. All the characteristics introduced above are preserved in all FSL rounds.

7.6 Conclusion

In this chapter, we proposed a new private distributed FSL achievable scheme with a communication cost that is order-wise similar to the communication cost of existing

schemes which provide much weaker privacy guarantees. Compared to the existing schemes with similar privacy guarantees, our proposed scheme does not require noisy storage of the submodels in the databases. Our proposed scheme is resilient against client drop-outs, client late-arrivals, and database drop-outs. The main ideas of this scheme are based on PSU and its variation for private write, together with random PIR and one-time pads for needed common randomness generation at the client side.

Our scheme starts with replicated storage of the submodels in two non-colluding databases at the server together with some amount of server-side common randomness. Our scheme privately generates needed common randomness at the client side, privately determines the union of the indices of the submodels to be updated, and privately writes the updated submodels back to the databases. Neither the indices of the submodels updated within the union, nor their updated values are leaked to the databases.

In this work, we considered the simplest version of this new formulation. The issues that need to be studied further include: 1) The case when the server has more than two databases. 2) Privacy of stored data against databases. 3) Colluding databases. 4) Byzantine databases that send erroneous information. 5) Collusion among clients. 6) Byzantine clients that send erroneous updates to poison the learning process. 7) Schemes to reduce the communication and storage cost, and potential communication-storage trade-off. 8) MDS coded storage and/or MDS coded user-side common randomness. 9) Optimum partitioning of the clients among databases, especially, with colluding databases under a known colluding structure.

CHAPTER 8

Fully Robust Federated Submodel Learning in Distributed Storage System

8.1 Introduction

In this chapter, we consider the FSL problem in a distributed storage system. At this point, the server comprises multiple independent databases and the full model is stored across these databases. An eavesdropper passively observes all the storage and listens to all the communicated data, of its controlled databases, to gain knowledge about the remote client data and the submodel information. In addition, a subset of databases may fail, negatively affecting the FSL process, as FSL process may take a non-negligible amount of time for large models. To resolve these two issues together (i.e., security and database repair), we propose a novel coding mechanism coined RSRC, to store the full model in a distributed manner. Using our new RSRC method, the eavesdropper is permitted to learn a controllable amount of submodel information for the sake of reducing the communication and storage costs. Further, during the database repair process, in the construction of the replacement database, the submodels to be updated are stored in the form of their latest ver-

sion from updating clients, while the remaining submodels are obtained from the previous version in other databases through routing clients. Our new RSRC-based distributed FSL approach is constructed on top of our earlier two-database PSU-based FSL scheme in Chapter 7. A complete one-round FSL process consists of: 1) an FSL-PSU phase where the union of the submodel indices to be updated by the selected clients in the current round is determined, 2) an FSL-write phase where the updated submodels are written back to the databases, and 3) additional auxiliary phases where sufficient amounts of necessary common randomness are generated at both server and client sides. Our proposed distributed FSL scheme is also robust against database drop-outs, client drop-outs, client late-arrivals, as well as any manipulations of database information by an active adversary who corrupts the communicated data.

8.2 Problem Formulation

In this work, we consider a distributed FSL problem with one server that comprises N independent databases and C clients that are selected by the server to participate in one-round of the FSL process; see Fig. 8.1. By convention, each client at the user side establishes a direct secure and authenticated communication channel with each database at the server side.[1] In addition, the mutual communication among databases in the server is not required in this work. The full learning model[2]

[1] Our distributed FSL scheme relies only on the client-database communication for the sake of simplicity and stability.

[2] For any arbitrary positive integer Z, we use the notation $[Z] = \{1, 2, \ldots, Z\}$ in this work for simplicity.

$M_{[K]} = \{M_1, M_2, \ldots, M_K\}$ encompasses K submodels, with each one consisting of L i.i.d. symbols that are uniformly selected from a finite field $\mathbb{F}_q$, and is stored across the databases. Thus, we have

$$H(M_k) = L, \quad \forall k \tag{8.1}$$

$$H(M_{[K]}) = H(M_1) + H(M_2) + \cdots + H(M_K) = KL \tag{8.2}$$

For each $j \in [N]$, database j takes as inputs the full model $M_{[K]}$ and the server-side common randomness $\mathcal{R}_S$, and stores the coded submodel information $G_j(M_{[K]}, \mathcal{R}_S)$ by using its own encoder G_j. Some additional server-side common randomness $\hat{\mathcal{R}}_S$ in plain form is also stored across the databases to assist the execution of the FSL process. For each $i \in [C]$, client i has its own data $\mathcal{D}_i$, which is used to train some submodels. Some necessary client-side common randomness $\mathcal{R}_C$ will be distributed to the clients before the FSL process starts. Following the client partition idea in [125], the large amount of C clients are separated into N groups according to their best communication channel bandwidth (or quality) with one specific database, i.e., if client i has the optimum communication with database j compared with the other databases, we assume that client i belongs to the client group $\mathcal{C}_j$. As a consequence, we have $\mathcal{C}_{j_1} \cap \mathcal{C}_{j_2} = \emptyset$ for any $j_1, j_2 \in [N], j_1 \neq j_2$ and $\cup_{j\in[N]}\mathcal{C}_j = [C]$. Each client intends to update one or more submodels according to its local training data. Specifically, for $i \in [C]$, client i wishes to update a set of submodels with its index set denoted by the random variable $\Gamma^{\langle i \rangle}$ (we use $\gamma^{\langle i \rangle}$ to denote the corresponding realization of $\Gamma^{\langle i \rangle}$). Moreover, for $i \in [C]$, we use the

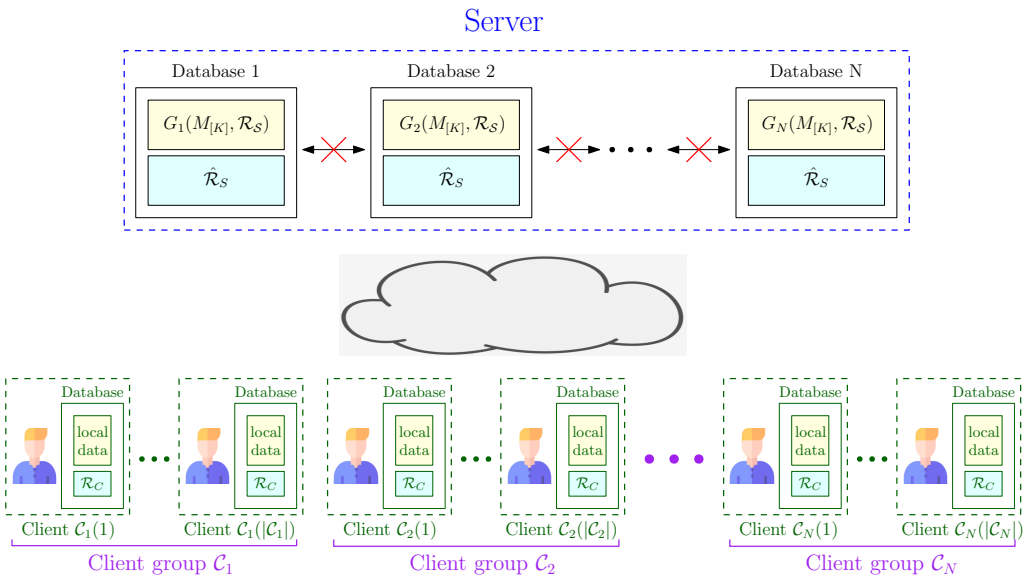


Figure 8.1: Federated submodel learning (FSL) problem in distributed storage system.

random variable $Y^{\langle i \rangle} = \{Y_1^{\langle i \rangle}, Y_2^{\langle i \rangle}, \ldots, Y_K^{\langle i \rangle}\}$ to denote the corresponding incidence vector of $\Gamma^{\langle i \rangle}$ after being mapped to the alphabet as in [72, 74].

Once a full round of the FSL process is complete, the submodels whose indices belong to the union $\Gamma = \Gamma^{\langle 1 \rangle} \cup \Gamma^{\langle 2 \rangle} \cup \cdots \cup \Gamma^{\langle C \rangle}$ are updated by the selected clients collaboratively while the remaining submodels remain the same. For $i \in [C]$, $k \in \Gamma$ and $l \in [L]$, the update $\Delta_{k,l}^{\langle i \rangle}$ is used to denote the corresponding increment generated in client i for the submodel symbol $M_{k,l}$. If $k \notin \Gamma^{\langle i \rangle}$, the update $\Delta_{k,l}^{\langle i \rangle}$ is simply set as 0. Thus, for $k \in \Gamma$, the overall increment applied on the submodel symbol $M_{k,l}$ is $\sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}$. The full increment sum vector over all submodels and over all symbols is defined as $\Delta_\Gamma = \{\sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}, k \in \Gamma, l \in [L]\}$. Thus, the updated full learning

model $M'_{[K]}$ for any $l \in [L]$ is follows,

$$M'_{k,l} = \begin{cases} M_{k,l} + \sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}, & \text{if } k \in \Gamma \\ M_{k,l}, & \text{otherwise} \end{cases} \tag{8.3}$$

Likewise, within the refreshed server-side common randomness $\mathcal{R}'_S$ and $\hat{\mathcal{R}}'_S$, the part coupled with the submodels in $M_\Gamma = \{M_k\}_{k \in \Gamma}$ should be updated. Thus, the storage of each database j should be updated according to the up-to-date full model $M'_{[K]}$ and refreshed server-side common randomness $\mathcal{R}'_S, \hat{\mathcal{R}}'_S$, while preserving its initial coding form.

Let $\mathcal{M}_j$ denote all the information that can be attained by database j including transmission information and storage information. Then, the FSL reliability in one-round of the FSL process is given by

$$\text{[reliability]} \quad H(G_j(M'_{[K]}, \mathcal{R}'_S), \hat{\mathcal{R}}'_S | \mathcal{M}_j) = 0, \quad \forall j \in [N] \tag{8.4}$$

As introduced in [85], the privacy constraint in FL typically requires that the aggregator learns nothing about clients' individual inputs except for their sum. In FSL, since the full model is divided into multiple submodels, the privacy constraint needs to be tuned, namely, the aggregator learns nothing about clients' local data except for their desired submodel union and submodel increment sum. Let $\mathcal{J} \subseteq [N]$ be an index set, then $\mathcal{M}_\mathcal{J} = \{\mathcal{M}_j\}_{j \in \mathcal{J}}$ is used to denote all the information involved in a set of databases whose indices belong to $\mathcal{J}$. Within this framework, we enforce

that any set of databases with cardinality at most J cannot infer any additional information about clients' local data $\mathcal{D}_{[C]} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_C\}$ beyond the union Γ and the full increment sum vector Δ_Γ, which is expressed by

$$[\text{privacy}] \quad I(\mathcal{M}_\mathcal{J}; \mathcal{D}_{[C]} | \Gamma, \Delta_\Gamma) = 0, \quad \forall \mathcal{J} \subseteq [N], \ |\mathcal{J}| \leq J \tag{8.5}$$

Following the multi-user PIR/SPIR problem formulated in [117,118], it is also required that each participating client should not gain any knowledge about the other clients' local data. Let $\mathcal{W}_i$ denote all the information that can be attained by client i including transmission information and storage information, and let $\mathcal{D}_{\bar{i}}$ denote the set $\{\mathcal{D}_1, \ldots, \mathcal{D}_{i-1}, \mathcal{D}_{i+1}, \ldots, \mathcal{D}_C\}$. Then, we have the following inter-client privacy constraint,

$$[\text{inter-client privacy}] \quad I(\mathcal{W}_i; \mathcal{D}_{\bar{i}}) = 0, \quad \forall i \in [C] \tag{8.6}$$

Two different types of security threats to the FSL model are investigated in this chapter. On the one hand, a passive eavesdropper can take control of any arbitrary E databases at the server side. Let $\mathcal{E}$ be the set of indices corresponding to these E databases where the cardinality of $\mathcal{E}$ is E. The eavesdropper can learn all the transmission information and storage information denoted by $\mathcal{M}_\mathcal{E}$. The eavesdropper is honest but curious in the sense that the goal of the eavesdropper is to obtain some additional information about the full learning model and clients' local data, but does not corrupt any transmissions. For simplicity, we assume that

E is smaller than or equal to J in this work, the privacy constraint (8.5) implies that the eavesdropper cannot learn any knowledge about the clients' local data further than Γ and Δ_Γ. Hence, the information leakage to the eavesdropper can be measured only in the amount of up-to-date full learning model $M'_{[K]}$. Not like the conventional configuration in which the eavesdropper should learn nothing about $M'_{[K]}$ [53, 55], we use the idea in [17, 67] for reference and introduce a new parameter δ, which is defined as the maximal fraction of latest full model information that can be learned by the eavesdropper. Thus, δ can be any rational number between 0 and 1.[3] In the presence of a passive eavesdropper, we have the following security constraint,

$$\text{[eavesdropper security]} \quad \frac{1}{KL} I(M'_{[K]}; \mathcal{M}_\mathcal{E}) \leq \delta, \quad \forall \mathcal{E} \subseteq [N],\ |\mathcal{E}| = E \tag{8.7}$$

On the other hand, an active adversary can get command of any arbitrary A databases at the server side to overwrite their transmissions to the clients. The adversary is Byzantine in the sense that the goal of the adversary is to intervene the normal running of the FSL process by generating arbitrarily erroneous information, without following the agreed upon protocol. The eavesdropper/adversary has an unlimited computational power and full knowledge of the FSL system. Neither the server nor the clients have any knowledge about the identities of the databases tapped in by the eavesdropper/adversary.

A basic one-round FSL achievable scheme under distributed coded storage is a one that satisfies the reliability constraint (8.4), the privacy constraint (8.5), the

[3] If δ is equal to 1, no eavesdropper security constraint is imposed in the problem formulation.

inter-client privacy constraint (8.6) and the eavesdropper security constraint (8.7). A fully robust one-round FSL achievable scheme in distributed coded storage should satisfy these four basic constraints, especially the reliability constraint (8.4) at all times, even in the presence of active adversaries, database failures, database drop-outs, client drop-outs and client late-arrivals. Moreover, we also need to guarantee that this one-round distributed FSL scheme can be executed iteratively with no errors until a predefined termination criterion is satisfied. As discussed in the introduction, the performance of a fully robust FSL scheme is evaluated by two metrics: communication cost and storage cost, which are both measured in the number of q-ary bits. Our goal is to develop a fully robust FSL scheme for a given set of FSL system parameters such that the total communication cost and the total storage are as small as possible.

8.3 Main Result

The main contribution of this chapter is a novel fully robust distributed FSL protocol. The performance of our protocol is evaluated in terms of the total communication cost and the total storage cost in each FSL round. The communication cost includes the cost incurred within the client-side and server-side common randomness generation. The main result of this chapter is presented in the following theorem, which is proved in Section 8.6.7.

Theorem 8.1 *The total communication cost and the total storage cost of the proposed distributed FSL achievable scheme in one round are* $\mathcal{O}(CK + C|\Gamma|L))$ *and*

$\mathcal{O}(KL)$, respectively, where C is the total number of participating clients, K is the total number of submodels, and $|\Gamma|$ is the number of updated submodels in the given round.

Remark 8.1 *Our new distributed FSL protocol is constructed as a generalization of our previous two-database FSL scheme via PSU [125]. Different than [125], here we have many databases at the server side. As in [125], we rely only on simple operations in a finite field at both client and server sides. Here, we achieve information-theoretic privacy for the clients as in [125], and additionally information-theoretic security against eavesdroppers. Unlike [125], here, the submodel information and server-side common randomness are stored across the databases in a coded form through our new RSRC technique. With RSRC, we achieve robustness against passive eavesdroppers, active adversaries and database failures, in addition to the existing resilience in [125] against database drop-outs, client drop-outs and client late-arrivals.*

Remark 8.2 *The communication cost in our new FSL protocol is order-wise the same as the one in [125]. Hence, all the conclusions in [125, Remark 6] are applicable here. In particular, once a database fails, the order-wise communication cost incurred for the repair process is $\mathcal{O}(KL)$, which is not negligible compared with the communication cost in the normal FSL process. Note that when the RSRC mechanism is utilized, although the order-wise cost makes no difference, the communication cost of obtaining the desired submodel information at the beginning of the FSL-write phase, the communication cost of repairing the submodel information in*

the replacement database under a database failure, and the storage cost, can be decreased simultaneously, while permitting the eavesdropper to learn more information about the full learning model.

Remark 8.3 *The storage cost in our new FSL protocol is $\mathcal{O}(KL)$, which is order-wise the same as storing the plain full learning model in the server. If we calculate the storage cost of the scheme in [125], it is also $\mathcal{O}(KL)$ including the server-side common randomness. Compared with previous FL or FSL approaches in [87, 96], our storage cost $\mathcal{O}(KL)$ does not include the term C^2, which is incurred by the secret sharing scheme across the clients.*

Remark 8.4 *We analyze the required number of databases here. Note that each client needs to contact D working databases out of N databases in total to recover the desired submodels M_Γ, in the presence of an eavesdropper who controls E databases, we must have $N > D > E$ in general. In order to preform the FSL process reliably, the total number of databases that drop-out or fail must be smaller than or equal to $N-D$. Furthermore, in the presence of an adversary who controls A databases, we must have $N \geq \max(2A+D, (J+1)(2A+1))$ according to the analysis of active adversary robustness in Section 8.6.6.*

8.4 RSRC Technique

In a distributed storage system, a set of messages are stored accross multiple databases either in a plain form or in a coded form. The secure regenerating code is developed such that in the presence of a passive eavesdropper or an active adversary,

a client can recover the message (reconstruction process) and a replacement database can be built to replace the failed database (repair process) by communicating with some working databases in an efficient way [107]. To evaluate the performance of a secure regenerating code, three main metrics are considered: reconstruction communication cost that counts the total number of symbols transferred from the working databases to the client in the reconstruction process, repair communication cost that counts the total number of symbols transferred from the working databases to the replacement database in the repair process, storage cost that counts the total number of symbols needed for the customized storage across all the databases. Following our FSL system model in Section 8.2, we assume that each client is permitted to contact D working databases to download submodel recovery information (reconstruction process) and database repair information (repair process) if necessary. Note that the coded storage in database j is $G_j(M_{[K]}, \mathcal{R}_S)$ where $M_{[K]}$ and $\mathcal{R}_S$ denote all the message symbols and randomness symbols, respectively. The following three constraints must be guaranteed while applying a secure regenerating code to our FSL system. First, a client can recover $M_{[K]}$ by communicating with any D working databases. This is referred to as the reconstruction constraint. Second, for all $j \in [N]$, a client can derive $G_j(M_{[K]}, \mathcal{R}_S)$ by communicating with any D remaining databases if database j fails, and then forward this database repair information to the replacement database.[4] This is referred to as the repair constraint. Third, at most a fraction δ of $M_{[K]}$ can be learned by any arbitrary E databases. This is

[4]In a practical implementation, if this constraint is always satisfied, multiple clients can work together with each one routing part of $G_j(M_{[K]}, \mathcal{R}_S)$ to the replacement database. Finally, the replacement database can still receive $G_j(M_{[K]}, \mathcal{R}_S)$.

referred to as the information leakage constraint.

In this chapter, we propose a novel secure regenerating coding mechanism called RSRC, which is devised specifically for our FSL system. This secure regenerating coding scheme is inspired by the ramp secret sharing idea in [109, 126]. Compared with previous secure regenerating code introduced in [107], there are a number of innovative points in our coding scheme. First, the information leakage fraction δ is not limited to the choice of 0. Second, in addition to the repair communication cost, reconstruction communication cost and storage cost are also considered for optimization. Third, in our coding scheme, we find that all of repair communication cost, reconstruction communication cost, and storage cost are certain simple functions of the parameter δ. In our ultimate construction of our general FSL achievable scheme, we use our RSRC technique as an elementary building block.

8.4.1 Construction and Performance of General RSRC

Following the product-matrix code [127], we first define the encoding matrix Ψ. Encoding matrix Ψ is basically an $N \times D$ Vandermonde matrix in the following form where all the elements $\{\psi_1, \psi_2, \psi_3, \ldots, \psi_N\}$ included in this matrix are selected

from a sufficiently large finite field $\mathbb{F}_q$ and are all distinct,

$$\Psi = \begin{bmatrix} 1 & \psi_1 & \psi_1^2 & \cdots & \psi_1^{D-1} \\ 1 & \psi_2 & \psi_2^2 & \cdots & \psi_2^{D-1} \\ 1 & \psi_3 & \psi_3^2 & \cdots & \psi_3^{D-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \psi_N & \psi_N^2 & \cdots & \psi_N^{D-1} \end{bmatrix}_{N \times D} \tag{8.8}$$

Next, we define a message matrix Ω which is simply a $D \times D$ symmetric matrix. The $D(D+1)/2$ distinct symbols in Ω consist of two parts: message symbols $M_{[K]}$ and randomness symbols $\mathcal{R}_S$. Assume that the number of message symbols is B, then the number of randomness symbols is $D(D+1)/2-B$. The code matrix ζ is obtained from the product of encoding matrix Ψ and message matrix Ω, i.e., $\zeta = \Psi\Omega$. For any $j \in [N]$, the jth row of ζ containing D symbols denoted by ζ_j^T is stored in database j. As the size of $M_{[K]}$ is generally large, we apply this coding method multiple times in a duplicate way. The coded storage $G_j(M_{[K]}, \mathcal{R}_S)$ is formed by concatenating all the generated ζ_j^T.

We use C_1 to denote the reconstruction communication cost, C_2 to denote the repair communication cost and S to denote the storage cost in each database as the storage cost is uniform over all the available databases. Moreover, if we use ℓ_λ to measure the extent of the message information leakage in any λ databases and $\mathcal{M}_\lambda$ to denote all the available information included in these λ databases, we have $\ell_\lambda = \frac{I(M_{[K]};\mathcal{M}_\lambda)}{H(M_{[K]})}$. After normalizing C_1, C_2, S by the total number of message symbols

B, we derive the following theorem regarding the performance of our general RSRC scheme.

Theorem 8.2 *Given the values of N and D that satisfy $N > D \geq 2$, for any arbitrary integer λ that satifies $0 < \lambda < D$ and any arbitrary rational number ℓ_λ that satisfies $0 \leq \ell_\lambda \leq 1$, an RSRC scheme with the following normalized performance can always be realized,*

$$\frac{C_1}{B} \geq \begin{cases} \frac{D+\lambda+1}{D-\lambda+1}(1-\ell_\lambda), & \text{if } \ell_\lambda \leq \frac{2\lambda}{D+\lambda+1} \\ 1, & \text{otherwise} \end{cases} \tag{8.9}$$

$$\frac{C_2}{B} \geq \begin{cases} \frac{2D}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda), & \text{if } \ell_\lambda \leq \frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)} \\ \frac{2}{D+1}, & \text{otherwise} \end{cases} \tag{8.10}$$

$$\frac{S}{B} \geq \begin{cases} \frac{2D^2}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda), & \text{if } \ell_\lambda \leq \frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)} \\ \frac{2D}{D+1}, & \text{otherwise} \end{cases} \tag{8.11}$$

Remark 8.5 *When $\ell_\lambda = 0$, the first inequality in Theorem 8.2 reduces to the result in [107].*

Remark 8.6 *According to the results in Theorem 8.2, when $0 \leq \ell_\lambda \leq \frac{2\lambda}{D+\lambda+1}$, the achieved normalized performance metrics of our RSRC scheme are all linear functions of the parameter ℓ_λ. That means that, by allowing larger message information leakage, we can further reduce all of the costs simultaneously. When $\frac{2\lambda}{D+\lambda+1} \leq \ell_\lambda \leq \frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)}$, the normalized C_2 and normalized S can be further de-*

creased if ℓ_λ is further increased, whereas the normalized C_1 reaches its minimum 1. When $\frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)} \leq \ell_\lambda \leq 1$, only the fraction $\frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)}$ of message information is leaked, which is smaller than the given upper bound ℓ_λ.

Proof: We first determine the starting message matrix Ω. As shown in Fig. 8.2, we fill the upper left corner of size $(D-\lambda) \times (D-\lambda)$ with message symbols, whereas the remaining zone with randomness symbols. Thus, the total number of message symbols B is $\frac{(D-\lambda)(D-\lambda+1)}{2}$ and the total number of randomness symbols is $\frac{D(D+1)}{2} - \frac{(D-\lambda)(D-\lambda+1)}{2} = \lambda D - \frac{\lambda(\lambda-1)}{2}$. We denote the current message matrix by Ω_1. Following [107, Thm. 11], as a reduced version, any λ databases gain no information about the message, i.e., $I(M_{[K]}; \mathcal{M}_\lambda) = 0$ and $\ell_\lambda = 0$. From another perspective of information-theoretic security analysis, by just applying linear computation, λD coded symbols included in these databases can be transformed into λD equivalent symbols where $\lambda D - \frac{\lambda(\lambda-1)}{2}$ symbols are associated with one distinct randomness symbol and $\frac{\lambda(\lambda-1)}{2}$ symbols are redundant.[5] Afterwards, for the lower left corner with size $\lambda \times (D-\lambda)$, the existing randomness symbol is substituted one-by-one by new message symbols. Once a randomness symbol vanishes, a new symbol only involving message symbol appears. This is also the basic idea in the construction of ramp secret sharing on the basis of classical secret sharing [109]. As a consequence, when the the lower left corner is filled with message symbols,[6] the total number of message symbols B now becomes $\frac{(D-\lambda)(D+\lambda+1)}{2}$ and the total number of randomness symbols is $\frac{\lambda(\lambda+1)}{2}$. We

[5] This result can also be proved by simply using linear algebraic calculations where the conversion matrix is invertible. The number of redundant symbols comes from the symmetric property of the message matrix.

[6] The upper right corner with size $(D-\lambda) \times \lambda$ is also filled with message symbols since the message matrix is always symmetric.

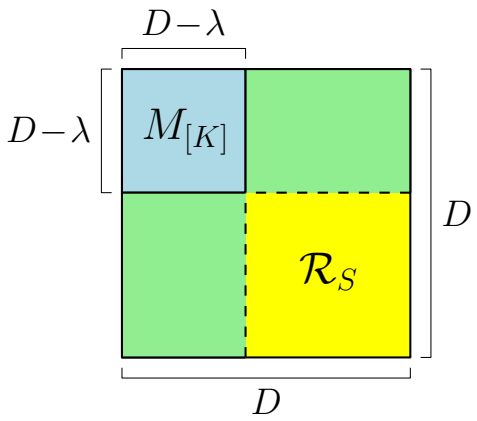


Figure 8.2: Structure of the $D \times D$ message matrix Ω.

denote this message matrix by Ω_2. The green zone in Fig. 8.2 can be considered as a ramp zone in which the message information leakage increases gradually. At this point, λD coded symbols included in these databases are equivalent to $\lambda(D-\lambda)$ symbols only consisting of message symbols, $\frac{\lambda(\lambda+1)}{2}$ symbols associated with one distinct randomness symbol and $\frac{\lambda(\lambda-1)}{2}$ redundant symbols. The message information leakage ℓ_λ in this scheme is $\frac{2\lambda(D-\lambda)}{D(D+1)-\lambda(\lambda+1)}$. Furthermore, if the whole message matrix is filled with message symbols without any randomness symbols, the total number of message symbols B is thus $\frac{D(D+1)}{2}$. We denote this message matrix by Ω_3. Out of λD coded symbols, $\frac{\lambda(\lambda-1)}{2}$ symbols are redundant. The message information leakage ℓ_λ in this scheme is $\frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)}$.

Regarding the message recovery in the reconstruction process, for the scheme with Ω_1 or Ω_2 or any message matrix in between, if all the coded symbols from these D connected databases are put together to form a $D \times D$ matrix, we can only concentrate on the $D \times (D-\lambda)$ left submatrix as all the message symbols have already been included. Thus, as the encoding matrix Ψ is a Vandermonde matrix, the client can download the full $D \times (D-\lambda)$ left submatrix except for the redundant upper

right isosceles triangle (or lower right isosceles triangle) with side length $D-\lambda-1$ to recover the message. Hence, the reconstruction communication cost C_1 is always $\frac{(D-\lambda)(D+\lambda+1)}{2}$. For the scheme with Ω_3, the client is supposed to download the full $D \times D$ matrix except for the redundant upper right isosceles triangle (or lower right isosceles triangle) with side length $D-1$. Hence, the corresponding C_1 is $\frac{D(D+1)}{2}$. Regarding the database repair, for message matrix Ω in any form, the client only needs to download 1 symbol from each database according to [107, Thms. 6, 11]. Hence, the repair communication cost C_2 is always D.

For completeness, we also state the basic idea of the repair process. For all $j \in [N]$, let Ψ_j^T denote the jth row of the encoding matrix Ψ. Then, the storage in database j is $\zeta_j^T = \Psi_j^T \Omega$. Without loss of generality, let database f fail. The corresponding storage in database f is $\zeta_f^T = \Psi_f^T \Omega$, which is the exact form required in the replacement database. Note that a client can contact any D working databases whose index set is denoted by $\mathcal{J}$. Then, for each $j \in \mathcal{J}$, database j only needs to pass one symbol $\zeta_j^T \Psi_f$ to the client. After collecting D symbols from the working databases, the client forwards the database f repair information $\zeta_{\mathcal{J}}^T \Psi_f = \Psi_{\mathcal{J}}^T \Omega \Psi_f$ to the replacement database. Since the $D \times D$ submatrix $\Psi_{\mathcal{J}}^T$ of Ψ is always invertible, the replacement database can rebuild $\Omega \Psi_f$, and then the required $\Psi_f^T \Omega$, because the message matrix Ω is symmetric. Note that the imported information to repair the failed database is equivalent to the original information stored in that database in terms of the message. As a consequence, even though a database repair operation is performed, the security analysis of information leakage constraint is not influenced. Regarding the storage cost, for any $D \times D$ message matrix Ω, each database always

needs to store D^2 symbols, i.e., $S = D^2$.

Now, we employ the time-sharing idea to obtain the normalized performance result given in Theorem 8.2. For any rational number ℓ_λ that satisfies $0 \leq \ell_\lambda \leq \frac{2\lambda}{D+\lambda+1}$, we can utilize the scheme with Ω_1 and the scheme with Ω_2 in a time-sharing manner to achieve it. Let ℓ_1 take the quotient form $\frac{p_1}{p_2}$ where p_1 and p_2 are both positive integers. We now use the first scheme q_1 times and the second scheme q_2 times such that the overall message leakage fraction ℓ_1 is still $\frac{p_1}{p_2}$. Hence, we must have

$$\frac{0q_1 + \lambda(D-\lambda)q_2}{B} = \frac{0q_1 + \lambda(D-\lambda)q_2}{\frac{(D-\lambda)(D-\lambda+1)}{2} \cdot q_1 + \frac{(D-\lambda)(D+\lambda+1)}{2} \cdot q_2} = \frac{p_1}{p_2} \tag{8.12}$$

If q_1 takes the value $2\lambda(D-\lambda)p_2-(D-\lambda)(D+\lambda+1)p_1$, q_2 is equal to $(D-\lambda)(D-\lambda+1)p_1$. As a consequence, we can just use the first scheme $2\lambda(D-\lambda)p_2-(D-\lambda)(D+\lambda+1)p_1$ times and the second scheme $(D-\lambda)(D-\lambda+1)p_1$ times. The total number of message symbols B is

$$B = \lambda(D-\lambda)q_2 \cdot \frac{p_2}{p_1} = \lambda(D-\lambda)^2(D-\lambda+1)p_2 \tag{8.13}$$

The total reconstruction communication cost C_1 is

$$C_1 = \frac{(D-\lambda)(D+\lambda+1)}{2} \cdot (q_1 + q_2) = \lambda(D-\lambda)^2(D+\lambda+1)(p_2 - p_1) \tag{8.14}$$

After normalizing it by B, we have

$$\frac{C_1}{B} = \frac{D+\lambda+1}{D-\lambda+1}(1-\ell_\lambda) \tag{8.15}$$

The total repair communication cost C_2 is

$$C_2 = D \cdot (q_1 + q_2) = 2D\lambda(D-\lambda)(p_2 - p_1) \tag{8.16}$$

After normalizing it by B, we have

$$\frac{C_2}{B} = \frac{2D}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda) \tag{8.17}$$

The storage cost S is always D multiple of C_2, which implies

$$\frac{S}{B} = \frac{2D^2}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda) \tag{8.18}$$

For any rational number ℓ_λ that satisfies $\frac{2\lambda}{D+\lambda+1} \leq \ell_\lambda \leq \frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)}$, we can apply time-sharing idea once more by combining the scheme with Ω_2 and the scheme with Ω_3 now. Hence, we must have

$$\frac{\lambda(D-\lambda)q_1 + (\lambda D - \frac{\lambda(\lambda-1)}{2})q_2}{B} = \frac{\lambda(D-\lambda)q_1 + (\lambda D - \frac{\lambda(\lambda-1)}{2})q_2}{\frac{(D-\lambda)(D+\lambda+1)}{2} \cdot q_1 + \frac{D(D+1)}{2} \cdot q_2} = \frac{p_1}{p_2} \tag{8.19}$$

If q_1 takes the value $(2\lambda D-\lambda^2+\lambda)p_2 - D(D+1)p_1$, q_2 is now equal to $(D-\lambda)(D+\lambda+1)p_1 - 2\lambda(D-\lambda)p_2$. As a consequence, we can just use the first scheme $(2\lambda D-$

$\lambda^2+\lambda)p_2-D(D+1)p_1$ times and the second scheme $(D-\lambda)(D+\lambda+1)p_1-2\lambda(D-\lambda)p_2$ times. The total number of message symbols B is

$$B = \frac{(D-\lambda)(D+\lambda+1)}{2} \cdot q_1 + \frac{D(D+1)}{2} \cdot q_2 = \frac{\lambda(\lambda+1)(D-\lambda)(D-\lambda+1)}{2} p_2 \tag{8.20}$$

The total reconstruction communication cost C_1 is

$$C_1 = \frac{(D-\lambda)(D+\lambda+1)}{2} \cdot q_1 + \frac{D(D+1)}{2} \cdot q_2 = \frac{\lambda(\lambda+1)(D-\lambda)(D-\lambda+1)}{2} p_2 \tag{8.21}$$

After normalizing it by B, we have

$$\frac{C_1}{B} = 1 \tag{8.22}$$

The total repair communication cost C_2 is

$$C_2 = D \cdot (q_1 + q_2) = \lambda(\lambda+1)D(p_2 - p_1) \tag{8.23}$$

After normalizing it by B, we have

$$\frac{C_2}{B} = \frac{2D}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda) \tag{8.24}$$

The storage cost S is still D multiples of C_2, which implies

$$\frac{S}{B} = \frac{2D^2}{(D-\lambda)(D-\lambda+1)}(1-\ell_\lambda) \tag{8.25}$$

For any rational number ℓ_λ that satisfies $\frac{2\lambda D-\lambda(\lambda-1)}{D(D+1)} \leq \ell_\lambda \leq 1$, we can use the scheme with Ω_3 all the time. Hence, we must have

$$\frac{C_1}{B} = \frac{\frac{D(D+1)}{2}}{\frac{D(D+1)}{2}} = 1 \tag{8.26}$$

$$\frac{C_2}{B} = \frac{D}{\frac{D(D+1)}{2}} = \frac{2}{D+1} \tag{8.27}$$

$$\frac{S}{B} = \frac{D^2}{\frac{D(D+1)}{2}} = \frac{2D}{D+1} \tag{8.28}$$

concluding the proof. ■

8.4.2 Examples to Illustrate the Basic Idea of RSRC

Consider the special case of $N = 4, D = 3$, all the symbols are operated in the finite field $\mathbb{F}_{13}$, and the encoding matrix Ψ is

$$\Psi = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 9 \\ 1 & 4 & 3 \end{bmatrix}_{4\times 3} \tag{8.29}$$

The message matrix Ω thus contains 9 symbols and 6 of them are distinct. If $\lambda = D = 3$, the study on ℓ_3 is trivial. Due to the reconstruction constraint that a client can always recover the message $M_{[K]}$ by connecting to any 3 working databases, these 3 databases must include all the information about $M_{[K]}$, namely, $H(M_{[K]}|\mathcal{M}_3) = 0$ and $\ell_3 = 1$. Without loss of generality, we now assume that database 4 fails.

Example 16: We first consider the situation where $\lambda = 1$. If the message matrix Ω takes the following form where M_1, M_2, M_3 are i.i.d. and uniformly selected message symbols from $\mathbb{F}_{13}$, R_1, R_2, R_3 are i.i.d. and uniformly selected randomness symbols from $\mathbb{F}_{13}$,

$$\Omega = \begin{bmatrix} M_1 & M_2 & R_1 \\ M_2 & M_3 & R_2 \\ R_1 & R_2 & R_3 \end{bmatrix} \tag{8.30}$$

The message length B is 3 and the coded storage across the databases is as follows,

$$\text{DB 1}: \quad M_1+M_2+R_1, \quad M_2+M_3+R_2, \quad R_1+R_2+R_3 \tag{8.31}$$

$$\text{DB 2}: \quad M_1+2M_2+3R_1, \ M_2+2M_3+3R_2, \ R_1+2R_2+3R_3 \tag{8.32}$$

$$\text{DB 3}: \quad M_1+3M_2+9R_1, \ M_2+3M_3+9R_2, \ R_1+3R_2+9R_3 \tag{8.33}$$

$$\text{DB 4}: \quad M_1+4M_2+3R_1, \ M_2+4M_3+3R_2, \ R_1+4R_2+3R_3 \tag{8.34}$$

For any one database, as the value of the randomness symbol R_3 is unknown, the database cannot learn any knowledge about the randomness symbols R_1, R_2 from the

third coded symbol. Furthermore, as none of R_1, R_2 are known, the database cannot learn any knowledge about the message symbols M_1, M_2, M_3 from the first two coded symbols. Therefore, each individual database learns nothing about the message set $M_{[K]}$, i.e., $I(M_{[K]}; \mathcal{M}_1) = 0$ and $\ell_1 = 0$. Following the performance analysis of RSRC in the last subsection, a client can download $\{M_1+M_2+R_1, M_2+M_3+R_2\}$, $\{M_1+2M_2+3R_1, M_2+2M_3+3R_2\}$, $M_1+3M_2+9R_1$ from database 1, 2, 3, respectively, to recover the message, and download $M_1+M_2+R_1+4(M_2+M_3+R_2)+3(R_1+R_2+R_3)$, $M_1+2M_2+3R_1+4(M_2+2M_3+3R_2)+3(R_1+2R_2+3R_3)$, $M_1+3M_2+9R_1+4(M_2+3M_3+9R_2)+3(R_1+3R_2+9R_3)$ from database 1, 2, 3, respectively, to repair the failed database because of the following equality,

$$\begin{bmatrix} M_1+M_2+R_1+4(M_2+M_3+R_2)+3(R_1+R_2+R_3) \\ M_1+2M_2+3R_1+4(M_2+2M_3+3R_2)+3(R_1+2R_2+3R_3) \\ M_1+3M_2+9R_1+4(M_2+3M_3+9R_2)+3(R_1+3R_2+9R_3) \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 9 \end{bmatrix} \begin{bmatrix} M_1 & M_2 & R_1 \\ M_2 & M_3 & R_2 \\ R_1 & R_2 & R_3 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 9 \end{bmatrix} \begin{bmatrix} M_1+4M_2+3R_1 \\ M_2+4M_3+3R_2 \\ R_1+4R_2+3R_3 \end{bmatrix} \quad (8.35)$$

This scheme achieves $C_1 = 5$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = \frac{5}{3}$, $\frac{C_2}{B} = 1$, $\frac{S}{B} = 3$.

If one more message symbol is added to the message matrix Ω in place of the

existing randomness symbol, the new Ω becomes

$$\Omega = \begin{bmatrix} M_1 & M_2 & M_3 \\ M_2 & M_4 & R_1 \\ M_3 & R_1 & R_2 \end{bmatrix} \tag{8.36}$$

The message length B is 4 and the coded storage across the databases is follows,

$$\text{DB 1}: \quad M_1+M_2+M_3, \quad M_2+M_4+R_1, \quad M_3+R_1+R_2 \tag{8.37}$$

$$\text{DB 2}: \quad M_1+2M_2+3M_3, \; M_2+2M_4+3R_1, \; M_3+2R_1+3R_2 \tag{8.38}$$

$$\text{DB 3}: \quad M_1+3M_2+9M_3, \; M_2+3M_4+9R_1, \; M_3+3R_1+9R_2 \tag{8.39}$$

$$\text{DB 4}: \quad M_1+4M_2+3M_3, \; M_2+4M_4+3R_1, \; M_3+4R_1+3R_2 \tag{8.40}$$

For any one database, due to the existence of the randomness symbols R_1, R_2, the only information concerning the message that can be learned by the database is the first coded symbol, which contains the ambiguity of $\frac{1}{4}H(M_{[K]})$. Therefore, we have $\ell_1 = \frac{1}{4}$ for each individual database. This scheme achieves the same performance, i.e., $C_1 = 5$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = \frac{5}{4}$, $\frac{C_2}{B} = \frac{3}{4}$, $\frac{S}{B} = \frac{9}{4}$. Therefore, by allowing the leak of partial information about the messages to the database, the normalized values of C_1, C_2 and S can be reduced. This partial information leakage idea can be considered as setting up a ramp zone where information leakage is not strictly prohibited.

If another message symbol is added to the current message matrix Ω in place

of one of the two remaining randomness symbols, the new Ω becomes

$$\Omega = \begin{bmatrix} M_1 & M_2 & M_3 \\ M_2 & M_4 & M_5 \\ M_3 & M_5 & R_1 \end{bmatrix} \tag{8.41}$$

The message length B is 5 and the coded storage across the databases is as follows,

$$\text{DB 1}: \quad M_1+M_2+M_3, \quad M_2+M_4+M_5, \quad M_3+M_5+R_1 \tag{8.42}$$

$$\text{DB 2}: \quad M_1+2M_2+3M_3, \ M_2+2M_4+3M_5, \ M_3+2M_5+3R_1 \tag{8.43}$$

$$\text{DB 3}: \quad M_1+3M_2+9M_3, \ M_2+3M_4+9M_5, \ M_3+3M_5+9R_1 \tag{8.44}$$

$$\text{DB 4}: \quad M_1+4M_2+3M_3, \ M_2+4M_4+3M_5, \ M_3+4M_5+3R_1 \tag{8.45}$$

Because of the existence of the randomness symbol R_1, each database can learn some information concerning the message from the first two coded symbols, which contains the ambiguity of $\frac{2}{5}H(M_{[K]})$. Therefore, we have $\ell_1 = \frac{2}{5}$ for each individual database. The performance of this scheme is still exactly the same as before, i.e., $C_1 = 5$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = 1$, $\frac{C_2}{B} = \frac{3}{5}$, $\frac{S}{B} = \frac{9}{5}$.

For any rational number ℓ_1 that satisfies $0 \leq \ell_1 \leq \frac{2}{5}$, we can utilize the scheme with Ω in (8.30) q_1 times and the scheme with Ω in (8.41) q_2 times in a time-sharing manner to achieve ℓ_1, which can also be expressed in the form of $\frac{p_1}{p_2}$. Hence, we must

have

$$\frac{0q_1 + 2q_2}{3q_1 + 5q_2} = \frac{p_1}{p_2} \tag{8.46}$$

If q_1 takes the value $2p_2 - 5p_1$, q_2 is now equal to $3p_1$. As a consequence, we can just use the first scheme $2p_2 - 5p_1$ times and the second scheme $3p_1$ times. After simple calculation, the overall message length B is $6p_2$, the overall reconstruction communication cost C_1 is $10(p_2 - p_1)$, the overall repair communication cost C_2 is $6(p_2 - p_1)$, and the overall storage cost S is $18(p_2 - p_1)$. By normalizing these values by B, for $0 \leq \ell_1 \leq \frac{2}{5}$, we obtain

$$\frac{C_1}{B} = \frac{5}{3} \cdot \frac{p_2 - p_1}{p_2} = \frac{5}{3}(1 - \ell_1),\ \frac{C_2}{B} = \frac{p_2 - p_1}{p_2} = 1 - \ell_1,\ \frac{S}{B} = 3 \cdot \frac{p_2 - p_1}{p_2} = 3(1 - \ell_1) \tag{8.47}$$

Note that by using these two schemes jointly to achieve $\ell_1 = \frac{1}{4}$, it has exactly the same normalized performance as the one obtained by using the scheme with Ω in (8.36) directly.

If all the symbols in the message matrix Ω are message symbols without any randomness symbols, Ω is in the following form,

$$\Omega = \begin{bmatrix} M_1 & M_2 & M_3 \\ M_2 & M_4 & M_5 \\ M_3 & M_5 & M_6 \end{bmatrix} \tag{8.48}$$

Now, the message length B becomes the maximal possible value 6 and the coded storage across the databases is as follows,

$$\text{DB 1}: \quad M_1+M_2+M_3, \quad M_2+M_4+M_5, \quad M_3+M_5+M_6 \tag{8.49}$$

$$\text{DB 2}: \quad M_1+2M_2+3M_3, \ M_2+2M_4+3M_5, \ M_3+2M_5+3M_6 \tag{8.50}$$

$$\text{DB 3}: \quad M_1+3M_2+9M_3, \ M_2+3M_4+9M_5, \ M_3+3M_5+9M_6 \tag{8.51}$$

$$\text{DB 4}: \quad M_1+4M_2+3M_3, \ M_2+4M_4+3M_5, \ M_3+4M_5+3M_6 \tag{8.52}$$

Each individual database learns three coded symbols with each one only containing message symbols. Therefore, we have $\ell_1 = \frac{3}{6} = \frac{1}{2}$ for each database. For the message recovery, a client can download $\{M_1+M_2+M_3, M_2+M_4+M_5, M_3+M_5+M_6\}$, $\{M_1+2M_2+3M_3, M_2+2M_4+3M_5\}$, $M_1+3M_2+9M_3$ from database 1, 2, 3, respectively. The corresponding performance of this scheme now becomes, $C_1 = 6$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = 1$, $\frac{C_2}{B} = \frac{1}{2}$, $\frac{S}{B} = \frac{3}{2}$.

For any rational number ℓ_1 that satisfies $\frac{2}{5} \leq \ell_1 \leq \frac{1}{2}$, we can utilize the time-sharing idea again by combining the scheme with Ω in (8.41) and the scheme with Ω in (8.48). At this point, the first scheme is used q_1 times and the second scheme is used q_2 times such that the overall ℓ_1 is equivalent to $\frac{p_1}{p_2}$, and we must have

$$\frac{2q_1 + 3q_2}{5q_1 + 6q_2} = \frac{p_1}{p_2} \tag{8.53}$$

If q_1 takes the value $3p_2 - 6p_1$, q_2 equals $5p_1 - 2p_2$. As a result, we can just apply the first scheme $3p_2 - 6p_1$ times and the second scheme $5p_1 - 2p_2$ times. After simple

calculation, B is $3p_2$, C_1 is $3p_2$, C_2 is $3(p_2 - p_1)$ and S is $9(p_2 - p_1)$. By normalizing these values by B, for $\frac{2}{5} \leq \ell_1 \leq \frac{1}{2}$, we obtain

$$\frac{C_1}{B} = \frac{3p_2}{3p_2} = 1, \quad \frac{C_2}{B} = \frac{p_2 - p_1}{p_2} = 1 - \ell_1, \quad \frac{S}{B} = 3 \cdot \frac{p_2 - p_1}{p_2} = 3(1 - \ell_1) \tag{8.54}$$

By combining the results in (8.47), (8.54) and the performance of the scheme using Ω in (8.48) for $\frac{1}{2} \leq \ell_1 \leq 1$, we can get the full normalized performance in the case of $N = 4$, $D = 3$, $\lambda = 1$, which matches the result in Theorem 8.2.

Example 17: We then consider the situation where $\lambda = 2$. The starting message matrix now takes the following form,

$$\Omega = \begin{bmatrix} M_1 & R_1 & R_2 \\ R_1 & R_3 & R_4 \\ R_2 & R_4 & R_5 \end{bmatrix} \tag{8.55}$$

The message length B is 1 and the coded storage across the databases is as follows,

$$\text{DB 1}: \quad M_1+R_1+R_2, \quad R_1+R_3+R_4, \quad R_2+R_4+R_5 \tag{8.56}$$

$$\text{DB 2}: \quad M_1+2R_1+3R_2, \; R_1+2R_3+3R_4, \; R_2+2R_4+3R_5 \tag{8.57}$$

$$\text{DB 3}: \quad M_1+3R_1+9R_2, \; R_1+3R_3+9R_4, \; R_2+3R_4+9R_5 \tag{8.58}$$

$$\text{DB 4}: \quad M_1+4R_1+3R_2, \; R_1+4R_3+3R_4, \; R_2+4R_4+3R_5 \tag{8.59}$$

For any two databases, according to the last two coded symbols from these two databases, neither of the randomness symbols R_1 and R_2 are decodable. Thus,

for the first coded symbols from these two database, the message symbol M_1 is completely unknown to two databases. Therefore, we have $I(M_{[K]}; \mathcal{M}_2) = 0$ and $\ell_2 = 0$. To recover the message, a client can simply download the first coded symbol from each database. The corresponding C_1 is 3 while $C_2 = 3$ and $S = 9$ are not changed, that means $\frac{C_1}{B} = 3$, $\frac{C_2}{B} = 3$, $\frac{S}{B} = 9$.

As in Example 16, we import message symbols into the message matrix Ω gradually to replace the randomness symbols, and the new Ω first becomes

$$\Omega = \begin{bmatrix} M_1 & M_2 & R_1 \\ M_2 & R_2 & R_3 \\ R_1 & R_3 & R_4 \end{bmatrix} \tag{8.60}$$

The message length B is 2 and the coded storage across the databases is follows,

$$\text{DB 1}: \quad M_1+M_2+R_1, \quad M_2+R_2+R_3, \quad R_1+R_3+R_4 \tag{8.61}$$

$$\text{DB 2}: \quad M_1+2M_2+3R_1, \; M_2+2R_2+3R_3, \; R_1+2R_3+3R_4 \tag{8.62}$$

$$\text{DB 3}: \quad M_1+3M_2+9R_1, \; M_2+3R_2+9R_3, \; R_1+3R_3+9R_4 \tag{8.63}$$

$$\text{DB 4}: \quad M_1+4M_2+3R_1, \; M_2+4R_2+3R_3, \; R_1+4R_3+3R_4 \tag{8.64}$$

For any two databases, one useful value that only involves a linear combination of message symbols M_1 and M_2 can be attained, whereas the other information is useless in terms of the message. We use database 1 and database 2 as an example here, by subtracting the coded symbol in database 2 from the triple coded symbol

in database 1 and the double coded symbol in database 1 element-wisely, the six coded symbols from these two databases are equal to $2M_1+M_2$, M_1-R_1, $2M_2+R_2$, M_2-R_3, R_1-R_4 and redundant $2R_1+R_3$. Therefore, we have $\ell_2 = \frac{1}{2}$ for any two databases. Regarding the performance of this scheme, all metrics remain the same, i.e., $C_1 = 3$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = \frac{3}{2}$, $\frac{C_2}{B} = \frac{3}{2}$, $\frac{S}{B} = \frac{9}{2}$.

When we move forward, the new Ω next becomes

$$\Omega_3 = \begin{bmatrix} M_1 & M_2 & M_3 \\ M_2 & R_1 & R_2 \\ M_3 & R_2 & R_3 \end{bmatrix} \tag{8.65}$$

The message length B is 3 and the coded storage across the databases is follows,

$$\text{DB 1}: \quad M_1+M_2+M_3, \quad M_2+R_1+R_2, \quad M_3+R_2+R_3 \tag{8.66}$$

$$\text{DB 2}: \quad M_1+2M_2+3M_3, \ M_2+2R_1+3R_2, \ M_3+2R_2+3R_3 \tag{8.67}$$

$$\text{DB 3}: \quad M_1+3M_2+9M_3, \ M_2+3R_1+9R_2, \ M_3+3R_2+9R_3 \tag{8.68}$$

$$\text{DB 4}: \quad M_1+4M_2+3M_3, \ M_2+4R_1+3R_2, \ M_3+4R_2+3R_3 \tag{8.69}$$

Note that this construction is different from the one in (8.30), although the message length is the same. By downloading the first coded symbol from each database, the reconstruction communication cost C_1 in this scheme is now 3 rather than 5. For any two databases, two useful values only involving a linear combination of message symbols can be attained, whereas the other information is still useless in terms of

the message. If we look at database 1 and database 2, the six coded symbols from these two databases are equivalent to $2M_1+M_2$, M_1-M_3, $2M_2+R_1$, M_2-R_2, M_3-R_3 and redundant $2M_3+R_2$. Therefore, we have $\ell_2 = \frac{2}{3}$ for any two databases. The performance of this scheme is maintained, i.e., $C_1 = 3$, $C_2 = 3$, $S = 9$, which implies $\frac{C_1}{B} = 1$, $\frac{C_2}{B} = 1$, $\frac{S}{B} = 3$.

Following the time-sharing idea in Example 16, by unifying the scheme with Ω in (8.55) and the scheme with Ω in (8.65), we obtain the following normalized performance for $0 \leq \ell_2 \leq \frac{2}{3}$,

$$\frac{C_1}{B} = 3(1-\ell_2), \quad \frac{C_2}{B} = 3(1-\ell_2), \quad \frac{S}{B} = 9(1-\ell_2) \tag{8.70}$$

When ℓ_2 takes the value $\frac{1}{2}$, this is exactly the performance of the scheme that uses Ω in (8.60). If we keep importing message symbols and using the time-sharing idea, we can obtain the remaining normalized performance in this situation when $\frac{2}{3} \leq \ell_2 \leq \frac{5}{6}$, namely,

$$\frac{C_1}{B} = 1, \quad \frac{C_2}{B} = 3(1-\ell_2), \quad \frac{S}{B} = 9(1-\ell_2) \tag{8.71}$$

The normalized reconstruction communication cost $\frac{C_1}{B}$ reaches the limit 1 yet the other two values remain the same. By putting the results in (8.70), (8.71) and the performance of the scheme using Ω in (8.48) for $\frac{5}{6} \leq \ell_1 \leq 1$ together, we can derive the full performance for $N = 4$, $D = 3$, $\lambda = 2$, which also matches the result in Theorem 8.2.

8.5 Distributed FSL Motivating Example

In this section, we consider a simple FSL setting as a toy example and provide a secure and robust achievable scheme in the presence of a passive eavesdropper and a database failure. The full learning model is divided into $K = 4$ submodels, $M_{[4]} = \{M_1, M_2, M_3, M_4\}$ with each submodel consisting of $L = 2$ symbols from the finite field F_{13} is stored in a coded form across $N = 4$ individual databases in the server. Any arbitrary $J = 2$ databases can collude with each other to learn the remote client data. In addition, $C = 4$ random clients are selected by the server to update the submodels in this round of the FSL process. Each client should be able to obtain the required submodel information by communicating with any arbitrary $D = 3$ working databases. The desired submodel index set for each client is

$$\text{Client } 1 \in \mathcal{C}_1 \colon \Gamma^{\langle 1 \rangle} = \{1\} \quad \Rightarrow Y^{\langle 1 \rangle} = [Y_1^{\langle 1 \rangle} \ Y_2^{\langle 1 \rangle} \ Y_3^{\langle 1 \rangle} \ Y_4^{\langle 1 \rangle}]^T = [1 \ 0 \ 0 \ 0]^T \quad (8.72)$$

$$\text{Client } 2 \in \mathcal{C}_1 \colon \Gamma^{\langle 2 \rangle} = \{1, 3\} \quad \Rightarrow Y^{\langle 2 \rangle} = [Y_1^{\langle 2 \rangle} \ Y_2^{\langle 2 \rangle} \ Y_3^{\langle 2 \rangle} \ Y_4^{\langle 2 \rangle}]^T = [1 \ 0 \ 1 \ 0]^T \quad (8.73)$$

$$\text{Client } 3 \in \mathcal{C}_2 \colon \Gamma^{\langle 3 \rangle} = \{1, 4\} \quad \Rightarrow Y^{\langle 3 \rangle} = [Y_1^{\langle 3 \rangle} \ Y_2^{\langle 3 \rangle} \ Y_3^{\langle 3 \rangle} \ Y_4^{\langle 3 \rangle}]^T = [1 \ 0 \ 0 \ 1]^T \quad (8.74)$$

$$\text{Client } 4 \in \mathcal{C}_3 \colon \Gamma^{\langle 4 \rangle} = \{1, 3, 4\} \Rightarrow Y^{\langle 4 \rangle} = [Y_1^{\langle 4 \rangle} \ Y_2^{\langle 4 \rangle} \ Y_3^{\langle 4 \rangle} \ Y_4^{\langle 4 \rangle}]^T = [1 \ 0 \ 1 \ 1]^T \quad (8.75)$$

Example 18: A passive eavesdropper can tap in on any arbitrary $E = 2$ databases to learn the storage data as well as all the communication data that comes in and goes out. Assuming that the submodel leakage parameter δ is 0.5, the submodel information is then coded through the RSRC scheme with encoding matrix Ψ in (8.29) and message matrix Ω in (8.60) for better FSL performance. Thus, the stor-

age across the databases including coded submodel information and extra uncoded server-side common randomness is initialized as in Table 8.1. At this point, database 4 has a failure and cannot provide any reliable responses. In this example, the generation of client-side common randomness and server-side common randomness are skipped, but it will be introduced in detail in the general FSL achievable scheme.

Database	Storage	
DB 1	$M_{1,1}+M_{1,2}+R_{1,1},\ M_{1,2}+R_{1,2}+R_{1,3},\ R_{1,1}+R_{1,3}+R_{1,4}$	$\hat{R}_1, \hat{R}_{1,1}, \hat{R}_{1,2}$
	$M_{2,1}+M_{2,2}+R_{2,1},\ M_{2,2}+R_{2,2}+R_{2,3},\ R_{2,1}+R_{2,3}+R_{2,4}$	$\hat{R}_2, \hat{R}_{2,1}, \hat{R}_{2,2}$
	$M_{3,1}+M_{3,2}+R_{3,1},\ M_{3,2}+R_{3,2}+R_{3,3},\ R_{3,1}+R_{3,3}+R_{3,4}$	$\hat{R}_3, \hat{R}_{3,1}, \hat{R}_{3,2}$
	$M_{4,1}+M_{4,2}+R_{4,1},\ M_{4,2}+R_{4,2}+R_{4,3},\ R_{4,1}+R_{4,3}+R_{4,4}$	$\hat{R}_4, \hat{R}_{4,1}, \hat{R}_{4,2}$
DB 2	$M_{1,1}+2M_{1,2}+3R_{1,1},\ M_{1,2}+2R_{1,2}+3R_{1,3},\ R_{1,1}+2R_{1,3}+3R_{1,4}$	$\hat{R}_1, \hat{R}_{1,1}, \hat{R}_{1,2}$
	$M_{2,1}+2M_{2,2}+3R_{2,1},\ M_{2,2}+2R_{2,2}+3R_{2,3},\ R_{2,1}+2R_{2,3}+3R_{2,4}$	$\hat{R}_2, \hat{R}_{2,1}, \hat{R}_{2,2}$
	$M_{3,1}+2M_{3,2}+3R_{3,1},\ M_{3,2}+2R_{3,2}+3R_{3,3},\ R_{3,1}+2R_{3,3}+3R_{3,4}$	$\hat{R}_3, \hat{R}_{3,1}, \hat{R}_{3,2}$
	$M_{4,1}+2M_{4,2}+3R_{4,1},\ M_{4,2}+2R_{4,2}+3R_{4,3},\ R_{4,1}+2R_{4,3}+3R_{4,4}$	$\hat{R}_4, \hat{R}_{4,1}, \hat{R}_{4,2}$
DB 3	$M_{1,1}+3M_{1,2}+9R_{1,1},\ M_{1,2}+3R_{1,2}+9R_{1,3},\ R_{1,1}+3R_{1,3}+9R_{1,4}$	$\hat{R}_1, \hat{R}_{1,1}, \hat{R}_{1,2}$
	$M_{2,1}+3M_{2,2}+9R_{2,1},\ M_{2,2}+3R_{2,2}+9R_{2,3},\ R_{2,1}+3R_{2,3}+9R_{2,4}$	$\hat{R}_2, \hat{R}_{2,1}, \hat{R}_{2,2}$
	$M_{3,1}+3M_{3,2}+9R_{3,1},\ M_{3,2}+3R_{3,2}+9R_{3,3},\ R_{3,1}+3R_{3,3}+9R_{3,4}$	$\hat{R}_3, \hat{R}_{3,1}, \hat{R}_{3,2}$
	$M_{4,1}+3M_{4,2}+9R_{4,1},\ M_{4,2}+3R_{4,2}+9R_{4,3},\ R_{4,1}+3R_{4,3}+9R_{4,4}$	$\hat{R}_4, \hat{R}_{4,1}, \hat{R}_{4,2}$
DB 4	$M_{1,1}+4M_{1,2}+3R_{1,1},\ M_{1,2}+4R_{1,2}+3R_{1,3},\ R_{1,1}+4R_{1,3}+3R_{1,4}$	$\hat{R}_1, \hat{R}_{1,1}, \hat{R}_{1,2}$
	$M_{2,1}+4M_{2,2}+3R_{2,1},\ M_{2,2}+4R_{2,2}+3R_{2,3},\ R_{2,1}+4R_{2,3}+3R_{2,4}$	$\hat{R}_2, \hat{R}_{2,1}, \hat{R}_{2,2}$
	$M_{3,1}+4M_{3,2}+3R_{3,1},\ M_{3,2}+4R_{3,2}+3R_{3,3},\ R_{3,1}+4R_{3,3}+3R_{3,4}$	$\hat{R}_3, \hat{R}_{3,1}, \hat{R}_{3,2}$
	$M_{4,1}+4M_{4,2}+3R_{4,1},\ M_{4,2}+4R_{4,2}+3R_{4,3},\ R_{4,1}+4R_{4,3}+3R_{4,4}$	$\hat{R}_4, \hat{R}_{4,1}, \hat{R}_{4,2}$

Table 8.1: Storage across the databases in the server when $D = 3$, $J = E = 2$ and $\delta = 0.5$.

FSL-PSU phase: In the first step of FSL-PSU phase, according to the client partition, the client answers are generated as follows[7] where the symbols c and $\{w_k^{\langle i \rangle} : i \in [4], k \in [4]\}$ are both client-side common randomness that satisfies $\sum_{i\in[4]} w_k^{\langle i \rangle} = 0$

[7] As in our previous work [125], the value in $\langle\rangle$ is used to denote the index of client and the value in () is used to denote the index of database. In addition, the first subscript of the download D or the answer A is used to show it is within the FSL-PSU phase or FSL-write phase as the letter U stands for union and the letter W stands for write, whereas the second subscript is used to denote the step number within this phase.

for all $k \in [4]$ and is unknown to any 2 databases at the server side,

$$A_{U,1}^{\langle 1\rangle,(1)} = \{c(Y_1^{\langle 1\rangle}+w_1^{\langle 1\rangle}), c(Y_2^{\langle 1\rangle}+w_2^{\langle 1\rangle}), c(Y_3^{\langle 1\rangle}+w_3^{\langle 1\rangle}), c(Y_4^{\langle 1\rangle}+w_4^{\langle 1\rangle})\} \tag{8.76}$$

$$A_{U,1}^{\langle 2\rangle,(1)} = \{c(Y_1^{\langle 2\rangle}+w_1^{\langle 2\rangle}), c(Y_2^{\langle 2\rangle}+w_2^{\langle 2\rangle}), c(Y_3^{\langle 2\rangle}+w_3^{\langle 2\rangle}), c(Y_4^{\langle 2\rangle}+w_4^{\langle 2\rangle})\} \tag{8.77}$$

$$A_{U,1}^{\langle 3\rangle,(2)} = \{c(Y_1^{\langle 3\rangle}+w_1^{\langle 3\rangle}), c(Y_2^{\langle 3\rangle}+w_2^{\langle 3\rangle}), c(Y_3^{\langle 3\rangle}+w_3^{\langle 3\rangle}), c(Y_4^{\langle 3\rangle}+w_4^{\langle 3\rangle})\} \tag{8.78}$$

$$A_{U,1}^{\langle 4\rangle,(3)} = \{c(Y_1^{\langle 4\rangle}+w_1^{\langle 4\rangle}), c(Y_2^{\langle 4\rangle}+w_2^{\langle 4\rangle}), c(Y_3^{\langle 4\rangle}+w_3^{\langle 4\rangle}), c(Y_4^{\langle 4\rangle}+w_4^{\langle 4\rangle})\} \tag{8.79}$$

In the second step of FSL-PSU phase, after collecting the answers from its associated clients 1 and 2, database 1 does the element-wise summation with the aid of its own extra uncoded server-side common randomness $\{\hat{R}_k : k \in [4]\}$ that is unknown to each individual client. This information is subsequently downloaded by a randomly selected client from the client group $\mathcal{C}_1$, say client 2,

$$\begin{aligned} D_{U,2}^{\langle 2\rangle,(1)} = \{&c(Y_1^{\langle 1\rangle}+Y_1^{\langle 2\rangle}+w_1^{\langle 1\rangle}+w_1^{\langle 2\rangle})+\hat{R}_1, c(Y_2^{\langle 1\rangle}+Y_2^{\langle 2\rangle}+w_2^{\langle 1\rangle}+w_2^{\langle 2\rangle})+\hat{R}_2, \\ &c(Y_3^{\langle 1\rangle}+Y_3^{\langle 2\rangle}+w_3^{\langle 1\rangle}+w_3^{\langle 2\rangle})+\hat{R}_3, c(Y_4^{\langle 1\rangle}+Y_4^{\langle 2\rangle}+w_4^{\langle 1\rangle}+w_4^{\langle 2\rangle})+\hat{R}_4\} \end{aligned} \tag{8.80}$$

After further processing the received information $D_{U,2}^{\langle 2\rangle,(1)}$ through additional client-side common randomness $\{w_k^{(j)} : j \in [3], k \in [4]\}$ that satisfies $\sum_{j\in[3]} w_k^{(j)} = 0$ for all $k \in [4]$ and is unknown to any 2 databases, client 2 forwards the following answer to all the functioning databases, i.e., database $1, 2, 3$,

$$A_{U,2}^{\langle 2\rangle,([3])}=\{c(Y_1^{\langle 1\rangle}+Y_1^{\langle 2\rangle}+w_1^{\langle 1\rangle}+w_1^{\langle 2\rangle})+\hat{R}_1+w_1^{(1)}, c(Y_2^{\langle 1\rangle}+Y_2^{\langle 2\rangle}+w_2^{\langle 1\rangle}+w_2^{\langle 2\rangle})+\hat{R}_2+w_2^{(1)},$$

$$c(Y_3^{\langle 1\rangle}+Y_3^{\langle 2\rangle}+w_3^{\langle 1\rangle}+w_3^{\langle 2\rangle})+\hat{R}_3+w_3^{(1)},c(Y_4^{\langle 1\rangle}+Y_4^{\langle 2\rangle}+w_4^{\langle 1\rangle}+w_4^{\langle 2\rangle})+\hat{R}_4+w_4^{(1)}\}$$

(8.81)

Likewise, after downloading the following information $D_{U,2}^{\langle 3\rangle,(2)}$ and $D_{U,2}^{\langle 4\rangle,(3)}$ from database 2 and database 3, respectively, client 3 and client 4 forward the following generated answers $A_{U,2}^{\langle 3\rangle,([3])}$ and $A_{U,2}^{\langle 4\rangle,([3])}$ to all the functioning databases as well,

$$\begin{aligned} D_{U,2}^{\langle 3\rangle,(2)} = \{&c(Y_1^{\langle 3\rangle}+w_1^{\langle 3\rangle})+\hat{R}_1, c(Y_2^{\langle 3\rangle}+w_2^{\langle 3\rangle})+\hat{R}_2, \\ &c(Y_3^{\langle 3\rangle}+w_3^{\langle 3\rangle})+\hat{R}_3, c(Y_4^{\langle 3\rangle}+w_4^{\langle 3\rangle})+\hat{R}_4\} \end{aligned} \tag{8.82}$$

$$\begin{aligned} D_{U,2}^{\langle 4\rangle,(3)} = \{&c(Y_1^{\langle 4\rangle}+w_1^{\langle 4\rangle})+\hat{R}_1, c(Y_2^{\langle 4\rangle}+w_2^{\langle 4\rangle})+\hat{R}_2, \\ &c(Y_3^{\langle 4\rangle}+w_3^{\langle 4\rangle})+\hat{R}_3, c(Y_4^{\langle 4\rangle}+w_4^{\langle 4\rangle})+\hat{R}_4\} \end{aligned} \tag{8.83}$$

$$\begin{aligned} A_{U,2}^{\langle 3\rangle,([3])} = \{&c(Y_1^{\langle 3\rangle}+w_1^{\langle 3\rangle})+\hat{R}_1+w_1^{(2)}, c(Y_2^{\langle 3\rangle}+w_2^{\langle 3\rangle})+\hat{R}_2+w_2^{(2)}, \\ &c(Y_3^{\langle 3\rangle}+w_3^{\langle 3\rangle})+\hat{R}_3+w_3^{(2)}, c(Y_4^{\langle 3\rangle}+w_4^{\langle 3\rangle})+\hat{R}_4+w_4^{(2)}\} \end{aligned} \tag{8.84}$$

$$\begin{aligned} A_{U,2}^{\langle 4\rangle,([3])} = \{&c(Y_1^{\langle 4\rangle}+w_1^{\langle 4\rangle})+\hat{R}_1+w_1^{(3)}, c(Y_2^{\langle 4\rangle}+w_2^{\langle 4\rangle})+\hat{R}_2+w_2^{(3)}, \\ &c(Y_3^{\langle 4\rangle}+w_3^{\langle 4\rangle})+\hat{R}_3+w_3^{(3)}, c(Y_4^{\langle 4\rangle}+w_4^{\langle 4\rangle})+\hat{R}_4+w_4^{(3)}\} \end{aligned} \tag{8.85}$$

For all $j \in [3]$, each an alive database j is able to find the desired submodel union through element-wise summation after collecting the available answers $A_{U,2}^{\langle 2\rangle,(j)}$, $A_{U,2}^{\langle 3\rangle,(j)}$ and $A_{U,2}^{\langle 4\rangle,(j)}$ from the routing clients $2,3,4$. For all $k \in [4]$, database j can get the result $c(\sum_{i\in[4]} Y_k^{\langle i\rangle})+3\hat{R}_k$ since $\sum_{i\in[4]} w_k^{\langle i\rangle} = 0$ and $\sum_{j\in[3]} w_k^{(j)} = 0$, which implies the value of $c(\sum_{i\in[4]} Y_k^{\langle i\rangle})$ since $\hat{R}_k$ is a known constant. It is straightforward to see that submodel k is in the union if $c(\sum_{i\in[4]} Y_k^{\langle i\rangle}) \neq 0$, otherwise, k is not in the union.

Therefore, the union result $\Gamma = \{1, 3, 4\}$ is obtained by each alive database. Due to the limited information uploaded by the selected clients, it is easy to verify that the server can only learn this union and nothing beyond the union, even though all these databases can collude with each other.

FSL-write phase: First, by downloading information from 3 functioning databases, each selected client needs to recover all the submodels in the submodel union $M_\Gamma = \{M_1, M_2, M_4\}$ to be updated in this FSL round. For all $i \in [3]$, we have

$$D_{W,1}^{\langle i\rangle,(1)} = \{M_{1,1}+M_{1,2}+R_{1,1}, M_{3,1}+M_{3,2}+R_{3,1}, M_{4,1}+M_{4,2}+R_{4,1}\} \tag{8.86}$$

$$D_{W,1}^{\langle i\rangle,(2)} = \{M_{1,1}+2M_{1,2}+3R_{1,1}, M_{3,1}+2M_{3,2}+3R_{3,1}, M_{4,1}+2M_{4,2}+3R_{4,1}\} \tag{8.87}$$

$$D_{W,1}^{\langle i\rangle,(3)} = \{M_{1,1}+4M_{1,2}+3R_{1,1}, M_{3,1}+4M_{3,2}+3R_{3,1}, M_{4,1}+3M_{4,2}+9R_{4,1}\} \tag{8.88}$$

Meanwhile, without loss of generality, client 4 is utilized to route the required information of submodel M_2 to the replacement database as a substitution for the failed database 4. Thus,

$$D_{W,1}^{\langle 4\rangle,(1)}=\{D_{W,1}^{\langle i\rangle,(1)},M_{2,1}+M_{2,2}+R_{2,1}+4(M_{2,2}+R_{2,2}+R_{2,3})+3(R_{2,1}+R_{2,3}+R_{2,4})\} \tag{8.89}$$

$$D_{W,1}^{\langle 4\rangle,(2)}=\{D_{W,1}^{\langle i\rangle,(2)},M_{2,1}+2M_{2,2}+3R_{2,1}+4(M_{2,2}+2R_{2,2}+3R_{2,3})+3(R_{2,1}+2R_{2,3}+3R_{2,4})\} \tag{8.90}$$

$$D_{W,1}^{\langle 4\rangle,(3)}=\{D_{W,1}^{\langle i\rangle,(3)},M_{2,1}+3M_{2,2}+9R_{2,1}+4(M_{2,2}+3R_{2,2}+9R_{2,3})+3(R_{2,1}+3R_{2,3}+9R_{2,4})\} \tag{8.91}$$

Due to the reconstruction constraint of the RSRC scheme, each client can reliably decode the desired submodels M_Γ as well as the server-side common randomness symbols $R_{1,1}, R_{3,1}, R_{4,1}$. When the local training is done, the answers sent by the clients in the first step of FSL-write phase are as follows where the symbols $\{w_{k,l}^{\langle i \rangle} : i \in [4], k \in \Gamma, l \in [2]\}$ are client-side common randomness that satisfy $\sum_{i\in[4]} w_{k,l}^{\langle i \rangle} = 0$ for all $k \in \Gamma, l \in [2]$ and is unknown to any 2 databases,

$$A_{W,1}^{\langle 1 \rangle,(1)} = \{\Delta_{1,1}^{\langle 1 \rangle}+w_{1,1}^{\langle 1 \rangle}, \Delta_{1,2}^{\langle 1 \rangle}+w_{1,2}^{\langle 1 \rangle}, w_{3,1}^{\langle 1 \rangle}, w_{3,2}^{\langle 1 \rangle}, w_{4,1}^{\langle 1 \rangle}, w_{4,2}^{\langle 1 \rangle}\} \tag{8.92}$$

$$A_{W,1}^{\langle 2 \rangle,(1)} = \{\Delta_{1,1}^{\langle 2 \rangle}+w_{1,1}^{\langle 2 \rangle}, \Delta_{1,2}^{\langle 2 \rangle}+w_{1,2}^{\langle 2 \rangle}, \Delta_{3,1}^{\langle 2 \rangle}+w_{3,1}^{\langle 2 \rangle}, \Delta_{3,2}^{\langle 2 \rangle}+w_{3,2}^{\langle 2 \rangle}, w_{4,1}^{\langle 2 \rangle}, w_{4,2}^{\langle 2 \rangle}\} \tag{8.93}$$

$$A_{W,1}^{\langle 3 \rangle,(2)} = \{\Delta_{1,1}^{\langle 3 \rangle}+w_{1,1}^{\langle 3 \rangle}, \Delta_{1,2}^{\langle 3 \rangle}+w_{1,2}^{\langle 3 \rangle}, w_{3,1}^{\langle 3 \rangle}, w_{3,2}^{\langle 3 \rangle}, \Delta_{4,1}^{\langle 3 \rangle}+w_{4,1}^{\langle 3 \rangle}, \Delta_{4,2}^{\langle 3 \rangle}+w_{4,2}^{\langle 3 \rangle}\} \tag{8.94}$$

$$A_{W,1}^{\langle 4 \rangle,(3)} = \{\Delta_{1,1}^{\langle 4 \rangle}+w_{1,1}^{\langle 4 \rangle}, \Delta_{1,2}^{\langle 4 \rangle}+w_{1,2}^{\langle 4 \rangle}, \Delta_{3,1}^{\langle 4 \rangle}+w_{3,1}^{\langle 4 \rangle}, \Delta_{3,2}^{\langle 4 \rangle}+w_{3,2}^{\langle 4 \rangle}, \Delta_{4,1}^{\langle 4 \rangle}+w_{4,1}^{\langle 4 \rangle}, \Delta_{4,2}^{\langle 4 \rangle}+w_{4,2}^{\langle 4 \rangle}\} \tag{8.95}$$

Following the similar execution in the previous FSL-PSU phase, client 2 downloads the following information from database 1 in the second step of FSL-write phase where the symbols $\{\hat{R}_{k,l} : k \in \Gamma, l \in [2]\}$ are extra uncoded server-side common randomness that is unknown to each individual client,

$$\begin{aligned} D_{W,2}^{\langle 2 \rangle,(1)} = \{&\Delta_{1,1}^{\langle 1 \rangle}+\Delta_{1,1}^{\langle 2 \rangle}+w_{1,1}^{\langle 1 \rangle}+w_{1,1}^{\langle 2 \rangle}+\hat{R}_{1,1}, \Delta_{1,2}^{\langle 1 \rangle}+\Delta_{1,2}^{\langle 2 \rangle}+w_{1,2}^{\langle 1 \rangle}+w_{1,2}^{\langle 2 \rangle}+\hat{R}_{1,2}, \Delta_{3,1}^{\langle 2 \rangle}+w_{3,1}^{\langle 1 \rangle} \\ &+w_{3,1}^{\langle 2 \rangle}+\hat{R}_{3,1}, \Delta_{3,2}^{\langle 2 \rangle}+w_{3,2}^{\langle 1 \rangle}+w_{3,2}^{\langle 2 \rangle}+\hat{R}_{3,2}, w_{4,1}^{\langle 1 \rangle}+w_{4,1}^{\langle 2 \rangle}+\hat{R}_{4,1}, w_{4,2}^{\langle 1 \rangle}+w_{4,2}^{\langle 2 \rangle}+\hat{R}_{4,2}\} \end{aligned} \tag{8.96}$$

Afterwards, client 2 transmits the different coded answers to the different working databases after appending its own randomness $\{w_{k,l_0}^{(1)} : k \in \Gamma, l_0 \in [4]\}$ that is

randomly selected only by client 2 under a uniform distribution from F_{13}, and thus, completely unknown to all the databases in the server,

$$
\begin{aligned}
A_{W,2}^{\langle 2\rangle,(1)} = \{&M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+\hat{R}_{1,1}+M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}\\
&+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2}+w_{1,1}^{(1)},\\
&M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2}+w_{1,2}^{(1)}+w_{1,3}^{(1)}, w_{1,1}^{(1)}+w_{1,3}^{(1)}+w_{1,4}^{(1)},\\
&M_{3,1}+\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+\hat{R}_{3,1}+M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2}+w_{3,1}^{(1)},\\
&M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2}+w_{3,2}^{(1)}+w_{3,3}^{(1)}, w_{3,1}^{(1)}+w_{3,3}^{(1)}+w_{3,4}^{(1)},\\
&M_{4,1}+w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+\hat{R}_{4,1}+M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2}+w_{4,1}^{(1)},\\
&M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2}+w_{4,2}^{(1)}+w_{4,3}^{(1)}, w_{4,1}^{(1)}+w_{4,3}^{(1)}+w_{4,4}^{(1)}\}
\end{aligned}
\tag{8.97}
$$

$$
\begin{aligned}
A_{W,2}^{\langle 2\rangle,(2)} = \{&M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+\hat{R}_{1,1}+2(M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}\\
&+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(1)},\\
&M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2}+2w_{1,2}^{(1)}+3w_{1,3}^{(1)}, w_{1,1}^{(1)}+2w_{1,3}^{(1)}+3w_{1,4}^{(1)},\\
&M_{3,1}+\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+\hat{R}_{3,1}+2(M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2})+3w_{3,1}^{(1)},\\
&M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2}+2w_{3,2}^{(1)}+3w_{3,3}^{(1)}, w_{3,1}^{(1)}+2w_{3,3}^{(1)}+3w_{3,4}^{(1)},\\
&M_{4,1}+w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+\hat{R}_{4,1}+2(M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(1)},\\
&M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2}+2w_{4,2}^{(1)}+3w_{4,3}^{(1)}, w_{4,1}^{(1)}+2w_{4,3}^{(1)}+3w_{4,4}^{(1)}\}
\end{aligned}
\tag{8.98}
$$

$$
\begin{aligned}
A_{W,2}^{\langle 2\rangle,(3)} = \{&M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+\hat{R}_{1,1}+3(M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}\\
&+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2})+9w_{1,1}^{(1)},\\
&M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2}+3w_{1,2}^{(1)}+9w_{1,3}^{(1)}, w_{1,1}^{(1)}+3w_{1,3}^{(1)}+9w_{1,4}^{(1)},\\
&M_{3,1}+\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+\hat{R}_{3,1}+3(M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2})+9w_{3,1}^{(1)},
\end{aligned}
$$

$$M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2}+3w_{3,2}^{(1)}+9w_{3,3}^{(1)}, w_{3,1}^{(1)}+3w_{3,3}^{(1)}+9w_{3,4}^{(1)},$$

$$M_{4,1}+w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+\hat{R}_{4,1}+3(M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2})+9w_{4,1}^{(1)},$$

$$M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2}+3w_{4,2}^{(1)}+9w_{4,3}^{(1)}, w_{4,1}^{(1)}+3w_{4,3}^{(1)}+9w_{4,4}^{(1)}\} \tag{8.99}$$

$$A_{W,2}^{\langle 2\rangle,(4)} = \{M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+\hat{R}_{1,1}+4(M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}$$

$$+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(1)},$$

$$M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2}+4w_{1,2}^{(1)}+3w_{1,3}^{(1)}, w_{1,1}^{(1)}+4w_{1,3}^{(1)}+3w_{1,4}^{(1)},$$

$$M_{3,1}+\Delta_{3,1}^{\langle 2\rangle}+w_{3,1}^{\langle 1\rangle}+w_{3,1}^{\langle 2\rangle}+\hat{R}_{3,1}+4(M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2})+3w_{3,1}^{(1)},$$

$$M_{3,2}+\Delta_{3,2}^{\langle 2\rangle}+w_{3,2}^{\langle 1\rangle}+w_{3,2}^{\langle 2\rangle}+\hat{R}_{3,2}+4w_{3,2}^{(1)}+3w_{3,3}^{(1)}, w_{3,1}^{(1)}+4w_{3,3}^{(1)}+3w_{3,4}^{(1)},$$

$$M_{4,1}+w_{4,1}^{\langle 1\rangle}+w_{4,1}^{\langle 2\rangle}+\hat{R}_{4,1}+4(M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(1)},$$

$$M_{4,2}+w_{4,2}^{\langle 1\rangle}+w_{4,2}^{\langle 2\rangle}+\hat{R}_{4,2}+4w_{4,2}^{(1)}+3w_{4,3}^{(1)}, w_{4,1}^{(1)}+4w_{4,3}^{(1)}+3w_{4,4}^{(1)}\} \tag{8.100}$$

Likewise, the following information is downloaded by client 3,

$$D_{W,2}^{\langle 3\rangle,(2)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}, \Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}, w_{3,1}^{\langle 3\rangle}+\hat{R}_{3,1},$$

$$w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}, \Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}+\hat{R}_{4,1}, \Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}\} \tag{8.101}$$

Exactly like client 2, client 3 transmits the coded answers to the corresponding databases as follows where $\{w_{k,l_0}^{(2)} : k \in \Gamma, l_0 \in [4]\}$ is its own randomness,

$$A_{W,2}^{\langle 3\rangle,(1)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}+\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}+w_{1,1}^{(2)},$$

$$\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}+w_{1,2}^{(2)}+w_{1,3}^{(2)}, w_{1,1}^{(2)}+w_{1,3}^{(2)}+w_{1,4}^{(2)},$$

$$w_{3,1}^{\langle 3\rangle}+\hat{R}_{3,1}+w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}+w_{3,1}^{(2)},$$

$$w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}+w_{3,2}^{(2)}+w_{3,3}^{(2)},\, w_{3,1}^{(2)}+w_{3,3}^{(2)}+w_{3,4}^{(2)},$$

$$\Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}+\hat{R}_{4,1}+\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}+w_{4,1}^{(2)},$$

$$\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}+w_{4,2}^{(2)}+w_{4,3}^{(2)},\, w_{4,1}^{(2)}+w_{4,3}^{(2)}+w_{4,4}^{(2)}\} \tag{8.102}$$

$$A_{W,2}^{\langle 3\rangle,(2)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}+2(\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(2)},$$

$$\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}+2w_{1,2}^{(2)}+3w_{1,3}^{(2)},\, w_{1,1}^{(2)}+2w_{1,3}^{(2)}+3w_{1,4}^{(2)},$$

$$w_{3,1}^{\langle 3\rangle}+\hat{R}_{3,1}+2(w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2})+3w_{3,1}^{(2)},$$

$$w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}+2w_{3,2}^{(2)}+3w_{3,3}^{(2)},\, w_{3,1}^{(2)}+2w_{3,3}^{(2)}+3w_{3,4}^{(2)},$$

$$\Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}+\hat{R}_{4,1}+2(\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(2)},$$

$$\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}+2w_{4,2}^{(2)}+3w_{4,3}^{(2)},\, w_{4,1}^{(2)}+2w_{4,3}^{(2)}+3w_{4,4}^{(2)}\} \tag{8.103}$$

$$A_{W,2}^{\langle 3\rangle,(3)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}+3(\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2})+9w_{1,1}^{(2)},$$

$$\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}+3w_{1,2}^{(2)}+9w_{1,3}^{(2)},\, w_{1,1}^{(2)}+3w_{1,3}^{(2)}+9w_{1,4}^{(2)},$$

$$w_{3,1}^{\langle 3\rangle}+\hat{R}_{3,1}+3(w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2})+9w_{3,1}^{(2)},$$

$$w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}+3w_{3,2}^{(2)}+9w_{3,3}^{(2)},\, w_{3,1}^{(2)}+3w_{3,3}^{(2)}+9w_{3,4}^{(2)},$$

$$\Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}+\hat{R}_{4,1}+3(\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2})+9w_{4,1}^{(2)},$$

$$\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}+3w_{4,2}^{(2)}+9w_{4,3}^{(2)},\, w_{4,1}^{(2)}+3w_{4,3}^{(2)}+9w_{4,4}^{(2)}\} \tag{8.104}$$

$$A_{W,2}^{\langle 3\rangle,(4)} = \{\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}+4(\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(2)},$$

$$\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2}+4w_{1,2}^{(2)}+3w_{1,3}^{(2)},\, w_{1,1}^{(2)}+4w_{1,3}^{(2)}+3w_{1,4}^{(2)},$$

$$w_{3,1}^{\langle 3\rangle}+\hat{R}_{3,1}+4(w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2})+3w_{3,1}^{(2)},$$

$$w_{3,2}^{\langle 3\rangle}+\hat{R}_{3,2}+4w_{3,2}^{(2)}+3w_{3,3}^{(2)},\, w_{3,1}^{(2)}+4w_{3,3}^{(2)}+3w_{3,4}^{(2)},$$

$$\Delta_{4,1}^{\langle 3\rangle}+w_{4,1}^{\langle 3\rangle}+\hat{R}_{4,1}+4(\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(2)},$$

$$\Delta_{4,2}^{\langle 3\rangle}+w_{4,2}^{\langle 3\rangle}+\hat{R}_{4,2}+4w_{4,2}^{(2)}+3w_{4,3}^{(2)}, w_{4,1}^{(2)}+4w_{4,3}^{(2)}+3w_{4,4}^{(2)}\} \tag{8.105}$$

At the same time, the following information is downloaded by client 4,

$$\begin{aligned} D_{W,2}^{\langle 4\rangle,(3)} = \{&\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}, \Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}, \Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}+\hat{R}_{3,1}, \\ &\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,2}, \Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}+\hat{R}_{4,1}, \Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}\} \end{aligned} \tag{8.106}$$

As client 4 is also employed to route the information of submodel 2 to the replacement database, it transmits the coded answers to the corresponding databases in the following way where $\{w_{k,l_0}^{(3)}: k \in \Gamma, l_0 \in [4]\}$ is its own randomness and $A_{W,2}^{\langle 4\rangle,(4)}$ is particularly different,

$$\begin{aligned} A_{W,2}^{\langle 4\rangle,(1)} = \{&\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}+\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}+w_{1,1}^{(3)}, \\ &\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}+w_{1,2}^{(3)}+w_{1,3}^{(3)}, w_{1,1}^{(3)}+w_{1,3}^{(3)}+w_{1,4}^{(3)} \\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}+\hat{R}_{3,1}+\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,1}+w_{3,1}^{(3)}, \\ &\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,2}+w_{3,2}^{(3)}+w_{3,3}^{(3)}, w_{3,1}^{(3)}+w_{3,3}^{(3)}+w_{3,4}^{(3)}, \\ &\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}+\hat{R}_{4,1}+\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}+w_{4,1}^{(3)}, \\ &\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}+w_{4,2}^{(3)}+w_{4,3}^{(3)}, w_{4,1}^{(3)}+w_{4,3}^{(3)}+w_{4,4}^{(3)}\} \end{aligned} \tag{8.107}$$

$$\begin{aligned} A_{W,2}^{\langle 4\rangle,(2)} = \{&\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}+2(\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(3)}, \\ &\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}+2w_{1,2}^{(3)}+3w_{1,3}^{(3)}, w_{1,1}^{(3)}+2w_{1,3}^{(3)}+3w_{1,4}^{(3)} \\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}+\hat{R}_{3,1}+2(\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,1})+3w_{3,1}^{(3)}, \\ &\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,2}+2w_{3,2}^{(3)}+3w_{3,3}^{(3)}, w_{3,1}^{(3)}+2w_{3,3}^{(3)}+3w_{3,4}^{(3)}, \end{aligned}$$

$$\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}+\hat{R}_{4,1}+2(\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(3)},$$

$$\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}+2w_{4,2}^{(3)}+3w_{4,3}^{(3)}, w_{4,1}^{(3)}+2w_{4,3}^{(3)}+3w_{4,4}^{(3)}\} \tag{8.108}$$

$$\begin{aligned}A_{W,2}^{\langle 4\rangle,(3)} = \{&\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}+3(\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2})+9w_{1,1}^{(3)},\\ &\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}+3w_{1,2}^{(3)}+9w_{1,3}^{(3)}, w_{1,1}^{(3)}+3w_{1,3}^{(3)}+9w_{1,4}^{(3)}\\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}+\hat{R}_{3,1}+3(\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,1})+9w_{3,1}^{(3)},\\ &\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,2}+3w_{3,2}^{(3)}+9w_{3,3}^{(3)}, w_{3,1}^{(3)}+3w_{3,3}^{(3)}+9w_{3,4}^{(3)},\\ &\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}+\hat{R}_{4,1}+3(\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2})+9w_{4,1}^{(3)},\\ &\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}+3w_{4,2}^{(3)}+9w_{4,3}^{(3)}, w_{4,1}^{(3)}+3w_{4,3}^{(3)}+9w_{4,4}^{(3)}\}\end{aligned} \tag{8.109}$$

$$\begin{aligned}A_{W,2}^{\langle 4\rangle,(4)} = \{&\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}+4(\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(3)},\\ &\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2}+4w_{1,2}^{(3)}+3w_{1,3}^{(3)}, w_{1,1}^{(3)}+4w_{1,3}^{(3)}+3w_{1,4}^{(3)}\\ &\Delta_{3,1}^{\langle 4\rangle}+w_{3,1}^{\langle 4\rangle}+\hat{R}_{3,1}+4(\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,1})+3w_{3,1}^{(3)},\\ &\Delta_{3,2}^{\langle 4\rangle}+w_{3,2}^{\langle 4\rangle}+\hat{R}_{3,2}+4w_{3,2}^{(3)}+3w_{3,3}^{(3)}, w_{3,1}^{(3)}+4w_{3,3}^{(3)}+3w_{3,4}^{(3)},\\ &\Delta_{4,1}^{\langle 4\rangle}+w_{4,1}^{\langle 4\rangle}+\hat{R}_{4,1}+4(\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2})+3w_{4,1}^{(3)},\\ &\Delta_{4,2}^{\langle 4\rangle}+w_{4,2}^{\langle 4\rangle}+\hat{R}_{4,2}+4w_{4,2}^{(3)}+3w_{4,3}^{(3)}, w_{4,1}^{(3)}+4w_{4,3}^{(3)}+3w_{4,4}^{(3)},\\ &M_{2,1}+M_{2,2}+R_{2,1}+4(M_{2,2}+R_{2,2}+R_{2,3})+3(R_{2,1}+R_{2,3}+R_{2,4}),\\ &M_{2,1}+2M_{2,2}+3R_{2,1}+4(M_{2,2}+2R_{2,2}+3R_{2,3})+3(R_{2,1}+2R_{2,3}+3R_{2,4}),\\ &M_{2,1}+3M_{2,2}+9R_{2,1}+4(M_{2,2}+3R_{2,2}+9R_{2,3})+3(R_{2,1}+3R_{2,3}+9R_{2,4})\}\end{aligned} \tag{8.110}$$

As it is easy to check that the privacy constraint (8.5) and the inter-client privacy constraint (8.6) are both inherited directly from our previous FSL-PSU

scheme in [125], our emphasis here will be on the analysis of the reliability constraint (8.4), the eavesdropper security constraint (8.7) and the database failure robustness.

Regarding the reliability constraint: After collecting the answers from the routing clients $2, 3, 4$ in the second step of the FSL-write phase, each database just does the element-wise summation. Without loss of generality, let us focus on the first coded submodel symbol $M_{1,1}+2M_{1,2}+3R_{1,1}$ in database 2, then we have the following calculation where $\sum_{i\in[4]} w_{1,1}^{\langle i\rangle} = \sum_{i\in[4]} w_{1,2}^{\langle i\rangle} = 0$,

$$\begin{aligned}
&M_{1,1}+\Delta_{1,1}^{\langle 1\rangle}+\Delta_{1,1}^{\langle 2\rangle}+w_{1,1}^{\langle 1\rangle}+w_{1,1}^{\langle 2\rangle}+\hat{R}_{1,1}+2(M_{1,2}+\Delta_{1,2}^{\langle 1\rangle}+\Delta_{1,2}^{\langle 2\rangle}+w_{1,2}^{\langle 1\rangle}+w_{1,2}^{\langle 2\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(1)}\\
&\quad+\Delta_{1,1}^{\langle 3\rangle}+w_{1,1}^{\langle 3\rangle}+\hat{R}_{1,1}+2(\Delta_{1,2}^{\langle 3\rangle}+w_{1,2}^{\langle 3\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(2)}\\
&\quad+\Delta_{1,1}^{\langle 4\rangle}+w_{1,1}^{\langle 4\rangle}+\hat{R}_{1,1}+2(\Delta_{1,2}^{\langle 4\rangle}+w_{1,2}^{\langle 4\rangle}+\hat{R}_{1,2})+3w_{1,1}^{(3)} \qquad (8.111)\\
&= M_{1,1}+\sum_{i\in[4]}\Delta_{1,1}^{\langle i\rangle}+2(M_{1,2}+\sum_{i\in[4]}\Delta_{1,2}^{\langle i\rangle})+3(w_{1,1}^{(1)}+w_{1,1}^{(2)}+w_{1,1}^{(3)})+3\hat{R}_{1,1}+6\hat{R}_{1,2} \qquad (8.112)\\
&= M'_{1,1}+2M'_{1,2}+3R'_{1,1}+3\hat{R}_{1,1}+6\hat{R}_{1,2} \qquad (8.113)
\end{aligned}$$

As $3\hat{R}_{1,1}+6\hat{R}_{1,2}$ is a known constant to database 2 and $w_{1,1}^{(1)}+w_{1,1}^{(2)}+w_{1,1}^{(3)}$ can be treated as $R'_{1,1}$, this database is able to decode the value of $M'_{1,1}+2M'_{1,2}+3R'_{1,1}$, which will be stored as a new coded submodel symbol for the next round of the FSL process. For the other needed symbols across the databases, the same calculation can be performed. In addition, in order to make the storage consistent across the databases and achieve perfect privacy in the next round, all the extra uncoded server-side common randomness also needs to be refreshed. When this round of the FSL process is complete, the updated storage in the server is shown in Table 8.2.

Database	Storage	
DB 1	$M'_{1,1}+M'_{1,2}+R'_{1,1},\ M'_{1,2}+R'_{1,2}+R'_{1,3},\ R'_{1,1}+R'_{1,3}+R'_{1,4}$ $M_{2,1}+M_{2,2}+R_{2,1},\ M_{2,2}+R_{2,2}+R_{2,3},\ R_{2,1}+R_{2,3}+R_{2,4}$ $M'_{3,1}+M'_{3,2}+R'_{3,1},\ M'_{3,2}+R'_{3,2}+R'_{3,3},\ R'_{3,1}+R'_{3,3}+R'_{3,4}$ $M'_{4,1}+M'_{4,2}+R'_{4,1},\ M'_{4,2}+R'_{4,2}+R'_{4,3},\ R'_{4,1}+R'_{4,3}+R'_{4,4}$	$\hat{R}'_1, \hat{R}'_{1,1}, \hat{R}'_{1,2}$ $\hat{R}'_2, \hat{R}'_{2,1}, \hat{R}'_{2,2}$ $\hat{R}'_3, \hat{R}'_{3,1}, \hat{R}'_{3,2}$ $\hat{R}'_4, \hat{R}'_{4,1}, \hat{R}'_{4,2}$
DB 2	$M'_{1,1}+2M'_{1,2}+3R'_{1,1},\ M'_{1,2}+2R'_{1,2}+3R'_{1,3},\ R'_{1,1}+2R'_{1,3}+3R'_{1,4}$ $M_{2,1}+2M_{2,2}+3R_{2,1},\ M_{2,2}+2R_{2,2}+3R_{2,3},\ R_{2,1}+2R_{2,3}+3R_{2,4}$ $M'_{3,1}+2M'_{3,2}+3R'_{3,1},\ M'_{3,2}+2R'_{3,2}+3R'_{3,3},\ R'_{3,1}+2R'_{3,3}+3R'_{3,4}$ $M'_{4,1}+2M'_{4,2}+3R'_{4,1},\ M'_{4,2}+2R'_{4,2}+3R'_{4,3},\ R'_{4,1}+2R'_{4,3}+3R'_{4,4}$	$\hat{R}'_1, \hat{R}'_{1,1}, \hat{R}'_{1,2}$ $\hat{R}'_2, \hat{R}'_{2,1}, \hat{R}'_{2,2}$ $\hat{R}'_3, \hat{R}'_{3,1}, \hat{R}'_{3,2}$ $\hat{R}'_4, \hat{R}'_{4,1}, \hat{R}'_{4,2}$
DB 3	$M'_{1,1}+3M'_{1,2}+9R'_{1,1},\ M'_{1,2}+3R'_{1,2}+9R'_{1,3},\ R'_{1,1}+3R'_{1,3}+9R'_{1,4}$ $M_{2,1}+3M_{2,2}+9R_{2,1},\ M_{2,2}+3R_{2,2}+9R_{2,3},\ R_{2,1}+3R_{2,3}+9R_{2,4}$ $M'_{3,1}+3M'_{3,2}+9R'_{3,1},\ M'_{3,2}+3R'_{3,2}+9R'_{3,3},\ R'_{3,1}+3R'_{3,3}+9R'_{3,4}$ $M'_{4,1}+3M'_{4,2}+9R'_{4,1},\ M'_{4,2}+3R'_{4,2}+9R'_{4,3},\ R'_{4,1}+3R'_{4,3}+9R'_{4,4}$	$\hat{R}'_1, \hat{R}'_{1,1}, \hat{R}'_{1,2}$ $\hat{R}'_2, \hat{R}'_{2,1}, \hat{R}'_{2,2}$ $\hat{R}'_3, \hat{R}'_{3,1}, \hat{R}'_{3,2}$ $\hat{R}'_4, \hat{R}'_{4,1}, \hat{R}'_{4,2}$
DB 4	$M'_{1,1}+4M'_{1,2}+3R'_{1,1},\ M'_{1,2}+4R'_{1,2}+3R'_{1,3},\ R'_{1,1}+4R'_{1,3}+3R'_{1,4}$ $M_{2,1}+4M_{2,2}+3R_{2,1},\ M_{2,2}+4R_{2,2}+3R_{2,3},\ R_{2,1}+4R_{2,3}+3R_{2,4}$ $M'_{3,1}+4M'_{3,2}+3R'_{3,1},\ M'_{3,2}+4R'_{3,2}+3R'_{3,3},\ R'_{3,1}+4R'_{3,3}+3R'_{3,4}$ $M'_{4,1}+4M'_{4,2}+3R'_{4,1},\ M'_{4,2}+4R'_{4,2}+3R'_{4,3},\ R'_{4,1}+4R'_{4,3}+3R'_{4,4}$	$\hat{R}'_1, \hat{R}'_{1,1}, \hat{R}'_{1,2}$ $\hat{R}'_2, \hat{R}'_{2,1}, \hat{R}'_{2,2}$ $\hat{R}'_3, \hat{R}'_{3,1}, \hat{R}'_{3,2}$ $\hat{R}'_4, \hat{R}'_{4,1}, \hat{R}'_{4,2}$

Table 8.2: Updated storage across the databases in the server after one FSL training round when $D = 3$, $J = E = 2$ and $\delta = 0.5$.

Regarding the eavesdropper security constraint: Because the FSL-PSU phase has nothing to do with the updated full learning model $M'_{[4]}$, we only need to consider the FSL-write phase. In terms of $M'_{[4]}$, for all $j \in [4]$, it is easy to prove that all the storage data and communication data that can be obtained by each database j is equivalent to the middle box in database j in Table 8.2, i.e., $G_j(M'_{\{1,3,4\}}, \mathcal{R}'_S)$ and $G_j(M_{\{2\}}, \mathcal{R}_S)$ after cancelling the carefully-designed client-side common randomness. Therefore, the guarantee of eavesdropper security is directly inherited from the information leakage constraint of the RSRC scheme.

Regarding the database failure robustness: From the last 3 symbols in $A_{W,2}^{\langle 4\rangle,(4)}$, the replacement database can correctly decode the original storage for submodel 2 in failed database 4 due to the repair constraint of the RSRC scheme. For the updated submodels $1, 3, 4$, the desired storage can also be attained from the routing

clients due to the satisfaction of the reliability constraint. The current replacement database is reflected in the database 4 part in Table 8.2.

8.6 General Distributed FSL Achievable Scheme

Following the distributed FSL problem formulated in Section 8.2, we provide our fully robust FSL achievable scheme for the general case in this section. Based on our previous work [125], the complete one-round FSL training is composed of four phases: client-side common randomness generation (FSL-CRG) phase that aims to distribute necessary client-side common randomness across all the selected clients, FSL-PSU phase that aims to privately determine the union of the submodel indices to be updated, FSL-write phase that aims to securely write the updated submodels in the union back to the databases, and server-side common randomness refresh (FSL-CRR) phase that aims to refresh the necessary server-side common randomness in preparation for the next round of the FSL process. In a practical implementation, the auxiliary FSL-CRG and FSL-CRR phases can be executed during the off-peak times because they are independent of the FSL-PSU and FSL-write phases.

8.6.1 FSL-CRG Phase

All the databases in the server aim to collectively establish the desired client-side common randomness across the clients such that every client-side common randomness symbol is completely unknown to any set of J colluding databases. The first

type of client-side common randomness is a set of symbols $\{w_1, w_2, \ldots, w_{\mathcal{L}}\}$ with a flexible set length $\mathcal{L}$. Within this set, each symbol is randomly and uniformly selected from $\mathbb{F}_q$ and the sum of these symbols is exactly 0, i.e., $\sum_{i\in[\mathcal{L}]} w_i = 0$. The second type of client-side common randomness is a random symbol c that is uniform over the set $\mathbb{F}_q\backslash\{0\}$.

For the first type, every $J+1$ databases can collaborate with each other to allocate the same set of client-side common randomness $\{w_1, w_2, \ldots, w_{\mathcal{L}}\}$ across a small set of clients. To that end, each database in a database set of size $J+1$ first individually selects $\mathcal{L}-1$ random symbols from $\mathbb{F}_q$ under a uniform distribution, and then simply broadcasts them to N distinct routing clients with indices $\theta_{[N]} = \{\theta_1, \theta_2, \ldots, \theta_N\}$ randomly chosen from N distinct client groups. After collecting all the random symbols from $J+1$ databases, these routing clients can just perform the element-wise summation over these $J+1$ random symbol sets of size $\mathcal{L}-1$ to obtain a new set of size $\mathcal{L}-1$. Subsequently, one more symbol is appended to the existing new set such that the set sum equals zero. At this point, this newly formed set can be used as $\{w_1, w_2, \ldots, w_{\mathcal{L}}\}$ because the one-time pad encryption guarantees the privacy of this client-side common randomness set against any J colluding databases.

If the value of $\mathcal{L}$ is N, the symbols in this set can be used as $w_k^{(j)}$ or $w_{k,d_2}^{(j)}$ for the next two phases. However, if the value of $\mathcal{L}$ is C, for all $i \in [C-1]\backslash\theta_{[N]}$, each database in this database set also needs to send its ith random symbol to client i. For client C, if C does not belong to $\theta_{[N]}$, these $J+1$ databases send all the generated random symbols to client C like routing clients. Thus, client C is able

to calculate w_C. Now, the symbols in this set are ready to be used as $w_k^{\langle i \rangle}$ or $w_{k,l}^{\langle i \rangle}$ for the next two phases. As the required number of client-side common randomness sets is very large, the communication time in this phase can be further optimized by wisely constituting some database subset of size $J+1$ from a set of N databases according to the actual situation. For example, if client i has a high-bandwidth communication channel with a particular database and the bandwidth utilization ratio of this database with all the clients is currently low, client i may select this database to participate in the client-side common randomness distribution.

For the second type, we can also select a set of $J+1$ databases to participate. Each database first selects a random symbol from $\mathbb{F}_q \backslash \{0\}$ under a uniform distribution, and then simply broadcasts it to each selected client. Once the client receives all the random symbols from $J+1$ databases, it just calculates the product of these $J+1$ symbols within $\mathbb{F}_q$. This new product can be used as c because the finite cyclic group $\mathbb{F}_q \backslash \{0\}$ under multiplication ensures the privacy of this client-side common randomness symbol against any J colluding databases. This symbol c can be used in the next FSL-PSU phase.

8.6.2 FSL-PSU Phase

The FSL-PSU phase in this work is similar to the private set union (FSL-PSU) phase in [125, Sect. 5.2] where nothing needs to be downloaded from the server to the clients at the beginning of FSL-PSU phase. After receiving required client-side common randomness in the last phase, each client uploads the index information of

its desired submodels to the server in a private way. For any client i that belongs to the client group $\mathcal{C}_j$, this client generates the following answer and sends it to database j in the first step of FSL-PSU phase,

$$A_{U,1}^{\langle i\rangle,(j)} = \{c(Y_k^{\langle i\rangle} + w_k^{\langle i\rangle}) \colon k \in [K]\}, \quad \forall i \in [C] \tag{8.114}$$

In the second step of FSL-PSU phase, for all $j \in [N]$, once database j completes the collection of all the answers from its associated clients in $\mathcal{C}_j$, it produces a response via element-wise summation as follows where θ_j is the index of the randomly selected routing client in $\mathcal{C}_j$ and $\hat{R}_k$ is shared server-side common randomness,

$$D_{U,2}^{\langle \theta_j\rangle,(j)} = \left\{ c \sum_{i\in\mathcal{C}_j} (Y_k^{\langle i\rangle} + w_k^{\langle i\rangle}) + \hat{R}_k \colon k \in [K] \right\}, \quad \forall j \in [N] \tag{8.115}$$

Then, this response is merely downloaded by its associated client θ_j. After further processing this response, each client θ_j forwards the following answer to all the databases in the server,

$$A_{U,2}^{\langle \theta_j\rangle,([N])} = \left\{ c \sum_{i\in\mathcal{C}_j} (Y_k^{\langle i\rangle} + w_k^{\langle i\rangle}) + w_k^{(j)} + \hat{R}_k \colon k \in [K] \right\}, \quad \forall j \in [N] \tag{8.116}$$

Each database j receives the same N answer sets. By summing these N answer sets up element-wise, each database derives the value of the expression $c \sum_{i\in[C]} Y_k^{\langle i\rangle}$ for all $k \in [K]$ because server-side common randomness is known to the database

and client-side common randomness is eliminated,

$$\sum_{j_0\in[N]} A_{U,2}^{\langle\theta_{j_0}\rangle,(j)} = \sum_{j_0\in[N]} (c\sum_{i\in\mathcal{C}_{j_0}} (Y_k^{\langle i\rangle}+w_k^{\langle i\rangle})+w_k^{(j_0)}+\hat{R}_k) \tag{8.117}$$

$$= c\sum_{j_0\in[N]}\sum_{i\in\mathcal{C}_{j_0}} Y_k^{\langle i\rangle}+c\sum_{j_0\in[N]}\sum_{i\in\mathcal{C}_{j_0}} w_k^{\langle i\rangle}+\sum_{j_0\in[N]} w_k^{(j_0)}+\sum_{j_0\in[N]}\hat{R}_k \tag{8.118}$$

$$= c\sum_{i\in[C]} Y_k^{\langle i\rangle}+c\sum_{i\in[C]} w_k^{\langle i\rangle}+\sum_{j_0\in[N]} w_k^{(j_0)}+\sum_{j_0\in[N]}\hat{R}_k \tag{8.119}$$

$$= c\sum_{i\in[C]} Y_k^{\langle i\rangle}+N\hat{R}_k \tag{8.120}$$

For any arbitrary submodel index k, if at least one client wishes to update the submodel k, the value of the expression $c\sum_{i\in[C]} Y_k^{\langle i\rangle}$ cannot be zero. Otherwise, this value is equal to zero. Therefore, each individual database is able to determine the desired submodel union Γ without any error by analyzing each submodel index one-by-one.

8.6.3 FSL-Write Phase

In the presence of an eavesdropper who can control any arbitrary E databases and get at most the fraction δ of the up-to-date full learning model $M'_{[K]}$, we can always find a RSRC-based FSL approach to satisfy the eavesdropper security constraint (8.7). Here, the variables λ and ℓ_λ in Section 8.4.1 take the values E and δ, respectively. By permitting the dummy message symbols to fill the message matrix Ω, if $0 \leq \delta \leq \frac{2E}{D+E+1}$, we can use the Ω_1-based RSRC scheme and the Ω_2-based RSRC scheme in a time-sharing manner to store the coded submodel information

across the databases. If $\frac{2E}{D+E+1} \leq \delta \leq \frac{2DE-E(E-1)}{D(D+1)}$, we can use the Ω_2-based RSRC scheme and the Ω_3-based RSRC scheme in a time-sharing manner. Otherwise, if $\frac{2DE-E(E-1)}{D(D+1)} \leq \delta \leq 1$, we can just use the Ω_3-based RSRC scheme.

Given the concrete form of $N \times D$ encoding matrix Ψ in (8.8), the concrete form of $D \times D$ message matrix Ω for the submodel k is as follows without loss of generality,

$$\Omega = \begin{bmatrix} X_{1,1}^k & X_{1,2}^k & X_{1,3}^k & \cdots & X_{1,D}^k \\ X_{2,1}^k & X_{2,2}^k & X_{2,3}^k & \cdots & X_{2,D}^k \\ X_{3,1}^k & X_{3,2}^k & X_{3,3}^k & \cdots & X_{3,D}^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X_{D,1}^k & X_{D,2}^k & X_{D,3}^k & \cdots & X_{D,D}^k \end{bmatrix} \tag{8.121}$$

For the first B message symbols in the submodel k, its corresponding storage in database j consisting of D symbols is in the following form,

$$\zeta_{j,k}^T = \left\{ \sum_{d_1 \in [D]} \psi_j^{d_1-1} X_{d_1,d_2}^k : d_2 \in [D] \right\}, \quad \forall j \in [N],\ \forall k \in [K] \tag{8.122}$$

where the symbol X_{d_1,d_2}^k in Ω can be a message symbol in submodel k or a randomness symbol depending on the realization of message matrix Ω.

When the FSL-PSU phase is finished, each database learns the desired submodel union Γ privately. In this subsection, we focus on the update of the first B symbols in each submodel whose index belongs to Γ. According to the statements in Section 8.4.1, all the remaining symbols can be fully updated through the repetition

and time-sharing ideas. In the first step of the FSL-write phase, each client needs to download C_1 coded symbols from D working databases in order to recover the first B message symbols in each submodel in M_Γ. For client i, let $\mathcal{N}_i$ denote the index set of D databases it communicates with, then,

$$D_{W,1}^{\langle i\rangle,(\mathcal{N}_i)} = \left\{Z_{k,\alpha} \colon k \in \Gamma,\ \alpha \in [C_1]\right\}, \quad \forall i \in [C] \tag{8.123}$$

where $Z_{k,\alpha}$ is picked from the set $\zeta_{j,k}^T$. Afterwards, this client generates the increments for its desired submodels whose index belongs to $\Gamma^{\langle i\rangle}$ when its local training is complete. Therefore, for any client i in the client group $\mathcal{C}_j$, the answer transmitted from client i to database j is in the following form,

$$A_{W,1}^{\langle i\rangle,(j)} = \left\{\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle} \colon k \in \Gamma,\ l \in [B]\right\}, \quad \forall i \in [C] \tag{8.124}$$

Like the second step of the FSL-PSU phase, each database j now generates an answer to be downloaded by the randomly selected routing client θ_j from the client group $\mathcal{C}_j$ as follows,

$$D_{W,2}^{\langle\theta_j\rangle,(j)} = \left\{\sum_{i\in\mathcal{C}_j}(\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle}) + \hat{R}_{k,l} \colon k \in \Gamma,\ l \in [B]\right\}, \quad \forall j \in [N] \tag{8.125}$$

Finally, each routing client θ_j needs to transfer different answers in different coded forms to all the available databases. Specifically, if X_{d_1,d_2}^k is a submodel symbol, say $M_{k,l}$, then $\mathcal{X}_{d_1,d_2}^k$ is equal to $M_{k,l} + \sum_{i\in\mathcal{C}_1}(\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle}) + \hat{R}_{k,l}$ if $j = 1$ and is equal to $\sum_{i\in\mathcal{C}_j}(\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle}) + \hat{R}_{k,l}$ if $j = 2, \ldots, N$. Otherwise, if X_{d_1,d_2}^k is a randomness symbol,

then $\mathcal{X}^k_{d_1,d_2}$ is a random value selected only by client θ_j under a uniform distribution from $\mathbb{F}_q$. Therefore, we have

$$A^{\langle\theta_j\rangle,(j_0)}_{W,2} = \left\{ \sum_{d_1\in[D]} \psi^{d_1-1}_{j_0} \mathcal{X}^k_{d_1,d_2} : k \in \Gamma,\ d_2 \in [D] \right\}, \quad \forall j \in [N],\ \forall j_0 \in [N] \quad (8.126)$$

Moreover, for some d_2, if all the symbols in the set $\{X^k_{d_1,d_2} : d_1 \in [D]\}$ are submodel symbols, a client-side common randomness needs to be appended, i.e.,

$$A^{\langle\theta_j\rangle,(j_0)}_{W,2} = \left\{ \sum_{d_1\in[D]} \psi^{d_1-1}_{j_0} \mathcal{X}^k_{d_1,d_2} + w^{(j)}_{k,d_2} : k \in \Gamma,\ d_2 \in [D] \right\}, \quad \forall j \in [N],\ \forall j_0 \in [N] \quad (8.127)$$

8.6.4 FSL-CRR Phase

In this phase, all the selected clients aim to jointly refresh the uncoded server-side common randomness $\hat{\mathcal{R}}_S$ coupled with the submodels that are updated in this round of the FSL process. In other words, the server-side common randomness symbols $\{\hat{R}_k : k \in [K]\}$ and $\{\hat{R}_{k,l} : k \in \Gamma, l \in [L]\}$ need to be refreshed. As each server-side common randomness symbol should be unknown to any individual client, every pair of clients can collaborate with each other to complete this task. More specifically, both clients select a random symbol from $\mathbb{F}_q$ and then forward it to each available database. By simply adding these two received random symbols together, the databases can now share a new server-side common randomness symbol that can be used as refreshed $\hat{R}'_k$ or $\hat{R}'_{k,l}$ for the next round of the FSL process. Like the

FSL-CRG phase, the required number of refreshed server-side common randomness symbols is also large. Hence, in a practical implementation, the communication time in this phase can also be optimized through partitioning the clients in a smart way.

8.6.5 Basic Characteristics Verification

In this section, we verify the basic characteristics of a complete round of the FSL process including the four phases mentioned above.

Reliability: According to the FSL-write phase in Section 8.6.3, for all $j \in [N]$, database j selects a random client θ_j from its associated client group to forward the information in the second step of the FSL-write phase. Thus, each database j receives the N answer sets $\{A_{W,2}^{\langle\theta_1\rangle,(j)}, A_{W,2}^{\langle\theta_2\rangle,(j)}, \ldots, A_{W,2}^{\langle\theta_N\rangle,(j)}\}$ from these N randomly selected routing clients. By summing these N answer sets up in an element-wise manner, for all $k \in \Gamma$ and all $d_2 \in [D]$, we have

$$\sum_{j_0\in[N]} A_{W,2}^{\langle\theta_{j_0}\rangle,(j)} = \sum_{j_0\in[N]} \sum_{d_1\in[D]} \psi_j^{d_1-1} \mathcal{X}_{d_1,d_2}^k = \sum_{d_1\in[D]} \psi_j^{d_1-1} \sum_{j_0\in[N]} \mathcal{X}_{d_1,d_2}^k \tag{8.128}$$

because the sum $\sum_{j\in[N]} w_{k,d_2}^{(j)}$ is equal to 0. If X_{d_1,d_2}^k is a submodel symbol, say $M_{k,l}$,

$$\psi_j^{d_1-1} \sum_{j_0\in[N]} \mathcal{X}_{d_1,d_2}^k = \psi_j^{d_1-1}(M_{k,l} + \sum_{j_0\in[N]} (\sum_{i\in\mathcal{C}_{j_0}} (\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle}) + \hat{R}_{k,l})) \tag{8.129}$$

$$= \psi_j^{d_1-1}(M_{k,l} + \sum_{i\in[C]} \Delta_{k,l}^{\langle i\rangle} + \sum_{i\in[C]} w_{k,l}^{\langle i\rangle} + N\hat{R}_{k,l}) \tag{8.130}$$

$$= \psi_j^{d_1-1}(M_{k,l} + \sum_{i\in[C]} \Delta_{k,l}^{\langle i\rangle} + N\hat{R}_{k,l}) \tag{8.131}$$

$$= \psi_j^{d_1-1}(M'_{k,l}+N\hat{R}_{k,l}) \tag{8.132}$$

At this point, each database j is able to derive the value of $\psi_j^{d_1-1}M'_{k,l}$ since the server-side common randomness $\hat{R}_{k,l}$ is known. Otherwise, if $X^k_{d_1,d_2}$ is a randomness symbol,

$$\psi_j^{d_1-1}\sum_{j_0\in[N]}\mathcal{X}^k_{d_1,d_2} = \psi_j^{d_1-1}\mathcal{W} \tag{8.133}$$

where $\mathcal{W}$ is a random value that is completely unknown to the databases. For all $d_2 \in [D]$, by adding the available $\psi_j^{d_1-1}M'_{k,l}$ and $\psi_j^{d_1-1}\mathcal{W}$ over $d_1 \in [D]$, the expected storage in database j for the first B symbols in the submodel k is attained, which can be easily extended to all the submodel information in M_Γ through repetition and time sharing. However, for all $k \in [K]\backslash\Gamma$, as these submodels are not updated at all, the corresponding storage in database j is not changed. That means the required storage $G_j(M'_{[K]}, \mathcal{R}'_S)$ is now achieved in all databases. Meanwhile, the additional plain server-side common randomness $\hat{\mathcal{R}}'_S$ can be directly attained as we expect through the approach in Section 8.6.4. Thus, the reliability constraint is satisfied.

Privacy: For any set of databases with index set $\mathcal{J}$ that meets $|\mathcal{J}| \leq J$, the answer sets $\{A_{U,1}^{\langle\mathcal{C}_j\rangle,(j)}, A_{U,2}^{\langle\theta_{[N]}\rangle,(j)}, A_{W,1}^{\langle\mathcal{C}_j\rangle,(j)}, A_{W,2}^{\langle\theta_{[N]}\rangle,(j)} : j \in \mathcal{J}\}$ are received. With respect to the clients' local data $\mathcal{D}_{[C]}$, it is easy to verify that these answer sets contain less information than the answer sets $\{A_{U,1}^{\langle\mathcal{C}_j\rangle,(j)}, A_{W,1}^{\langle\mathcal{C}_j\rangle,(j)} : j \in [N]\}$. Now, we can use the

answer sets $\{A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)} : j \in [N]\}$ as a base to analyze the privacy constraint. Note that each client-side common randomness symbol is completely unknown to this set of databases as it is generated by J+1 clients collectively. In the answer sets $\{A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)} : j \in [N]\}$, for all $k \in [K]$, the client-side common randomness symbol $w_k^{\langle i \rangle}$ can be used to protect the privacy of $Y_k^{\langle i \rangle}$, and the client-side common randomness symbol c can be used to protect the privacy of $\sum_{i \in [C]} Y_k^{\langle i \rangle}$. As a result, these $|\mathcal{J}|$ databases can only learn the union Γ and nothing beyond that.[8] The concrete proof can be checked in the client's privacy proof in [74, Subsect. V.B]. Then, we turn to the answer sets $\{A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)} : j \in [N]\}$. As a reduced version, for all $k \in [K]$ and $l \in [L]$, the client-side common randomness symbol $w_{k,l}^{\langle i \rangle}$ can be used to protect the privacy of $\Delta_{k,l}^{\langle i \rangle}$. As a result, these $|\mathcal{J}|$ databases can only learn the the full increment sum Δ_Γ and nothing beyond that. Thus, any set of databases with cardinality less than or equal to J cannot gain any additional information about $\mathcal{D}_{[C]}$ beyond Γ and Δ_Γ. Thus, the privacy constraint is satisfied.

Inter-Client Privacy: Only one client from each client group is able to receive the information concerning the other clients' local data. For each client group $\mathcal{C}_j$, the routing client θ_j downloads $D_{U,2}^{\langle \theta_j \rangle,(j)}$ and $D_{W,2}^{\langle \theta_j \rangle,(j)}$ from database j. Due to the existence of the unknown server-side common randomness $\hat{R}_k$ and $\hat{R}_{k,l}$ in the downloads, client θ_j cannot learn any knowledge about the other clients' local data. Thus, the inter-client privacy constraint is satisfied.

[8] The fact that both of $\sum_{i \in [C]} Y_k^{\langle i \rangle}$ and $\sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}$ are always 0 for $k \in [K] \backslash \Gamma$ is implied by the union Γ.

Security Against the Eavesdropper: To verify the security against the eavesdropper, we need to observe the storage information and the transmission information $\mathcal{M}_{\mathcal{E}}$ in any arbitrary E databases whose index set is $\mathcal{E}$. Before an FSL training round begins, the storage information in each database j is the coded submodel information $G_j(M_{[K]}, \mathcal{R}_S)$ and additional server-side common randomness $\hat{\mathcal{R}}_S$, while the latter has nothing to do with the updated full learning model $M'_{[K]}$. During this FSL training round, the transmission information that can be known by each database j is $\{A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{U,2}^{\langle \theta_{[N]} \rangle,(j)}, A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}, A_{W,2}^{\langle \theta_{[N]} \rangle,(j)}\}$. The first two answers only involve the information concerning clients' incidence vectors. The third answer is useless because of the coupled client-side common randomness symbol for each submodel increment symbol. Following the analysis of reliability constraint, the last answer is equivalent to $G_j(M'_\Gamma, \mathcal{R}'_S)$ due to the existence of carefully-designed client-side common randomness. Hence, for the submodels in M_Γ, we have the following relationship by using our specific RSRC-based approach in Section 8.6.3,

$$\frac{1}{|\Gamma|L} I(M'_\Gamma; \mathcal{M}_{\mathcal{E}}) \leq \delta, \quad \forall \mathcal{E} \subseteq [N],\ |\mathcal{E}| = E \tag{8.134}$$

For the submodels in $M_{[K]\backslash\Gamma}$, the identity $M'_{[K]\backslash\Gamma} = M_{[K]\backslash\Gamma}$ is true. Likewise, from the coded submodel information $G_j(M_{[K]\backslash\Gamma}, \mathcal{R}_S)$, we have

$$\frac{1}{(K-|\Gamma|)L} I(M'_{[K]\backslash\Gamma}; \mathcal{M}_{\mathcal{E}}) \leq \delta, \quad \forall \mathcal{E} \subseteq [N],\ |\mathcal{E}| = E \tag{8.135}$$

Because $M'_{[K]}$ and $M'_{[K]\setminus\Gamma}$ are always independent, we are able to derive the following outcome, which is sufficient to guarantee the eavesdropper security,

$$I(M'_{[K]};\mathcal{M}_{\mathcal{E}}) = I(M'_{\Gamma};\mathcal{M}_{\mathcal{E}}) + I(M'_{[K]\setminus\Gamma};\mathcal{M}_{\mathcal{E}}) \leq |\Gamma|L\cdot\delta + (K-|\Gamma|)L\cdot\delta = KL\cdot\delta \tag{8.136}$$

Thus, the security constraint against an eavesdropper is satisfied.

8.6.6 Full Robustness Verification

In this subsection, we analyze the robustness of our proposed achievable scheme in the face of all kinds of non-ideal situations. If multiple such incidents happen simultaneously, we can simply incorporate the idea for each incident into the adjusted scheme one-by-one.

Client Drop-Outs Robustness: In this case, our expectation is that the training process proceeds normally only relying on the data from the active clients. Without loss of generality, for each client group $\mathcal{C}_j$, we assume that a subset of clients with indices $\tilde{\mathcal{C}}_j$ drop-out. In the FSL-PSU phase, for all $j \in [N]$, the response $D_{U,2}^{\langle\theta_j\rangle,(j)}$ to be downloaded by client θ_j becomes

$$D_{U,2}^{\langle\theta_j\rangle,(j)} = \left\{ c \sum_{i\in\mathcal{C}_j\setminus\tilde{\mathcal{C}}_j} (Y_k^{\langle i\rangle} + w_k^{\langle i\rangle}) + \hat{R}_k \colon k\in[K] \right\}, \quad \forall j \in [N] \tag{8.137}$$

With the knowledge of the index set $\tilde{\mathcal{C}}_j$ corresponding to the out-of-operation clients, each routing client θ_j can adjust the answer by additionally appending the value of

$c\sum_{i\in\tilde{\mathcal{C}}_j} w_k^{\langle i\rangle}$ corresponding to the missing client-side common randomness symbols for all $k \in [K]$. Thus, the answer to be forwarded to all the available databases becomes

$$A_{U,2}^{\langle\theta_j\rangle,([N])} = \left\{ c \sum_{i\in\mathcal{C}_j\setminus\tilde{\mathcal{C}}_j} Y_k^{\langle i\rangle} + c\sum_{i\in\mathcal{C}_j} w_k^{\langle i\rangle} + w_k^{(j)} + \hat{R}_k \colon k \in [K] \right\}, \quad \forall j \in [N] \tag{8.138}$$

In this way, each database is still able to decode the union $\cup_{j\in[N]}\Gamma^{\langle\mathcal{C}_j\setminus\tilde{\mathcal{C}}_j\rangle}$ for all the active clients from the element-wise summation results $\{c\sum_{i\in\cup_{j\in[N]}(\mathcal{C}_j\setminus\tilde{\mathcal{C}}_j)} Y_k^{\langle i\rangle} : k \in [K]\}$.

In the FSL-write phase, for all $j \in [N]$, the response $D_{W,2}^{\langle\theta_j\rangle,(j)}$ to be downloaded becomes

$$D_{W,2}^{\langle\theta_j\rangle,(j)} = \left\{ \sum_{i\in\mathcal{C}_j\setminus\tilde{\mathcal{C}}_j} (\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle}) + \hat{R}_{k,l} \colon k \in \Gamma,\ l \in [B] \right\}, \quad \forall j \in [N] \tag{8.139}$$

Using the reliability constraint analysis in the last subsection as reference, if X_{d_1,d_2}^k is a submodel symbol, say $M_{k,l}$, after eliminating the server-side common randomness, for each database j, we have

$$\psi_j^{d_1-1} \sum_{j_0\in[N]} \mathcal{X}_{d_1,d_2}^k = \psi_j^{d_1-1}(M_{k,l} + \sum_{j_0\in[N]} \sum_{i\in\mathcal{C}_{j_0}\setminus\tilde{\mathcal{C}}_{j_0}} (\Delta_{k,l}^{\langle i\rangle} + w_{k,l}^{\langle i\rangle})) \tag{8.140}$$

$$= \psi_j^{d_1-1}(M_{k,l} + \sum_{i\in\cup_{j_0\in[N]}(\mathcal{C}_{j_0}\setminus\tilde{\mathcal{C}}_{j_0})} \Delta_{k,l}^{\langle i\rangle} + \sum_{i\in\cup_{j_0\in[N]}(\mathcal{C}_{j_0}\setminus\tilde{\mathcal{C}}_{j_0})} w_{k,l}^{\langle i\rangle}) \tag{8.141}$$

Otherwise, if X_{d_1,d_2}^k is a randomness symbol, $\psi_j^{d_1-1}\sum_{j_0\in[N]} \mathcal{X}_{d_1,d_2}^k$ is not changed. Hence, for all $k \in \Gamma$ and all $d_2 \in [D]$, each routing client θ_j first selects an empty

value β, i.e., $\beta = 0$. Then, for each d_1 in the set $[D]$, as long as $X^k_{d_1,d_2}$ is some submodel symbol $M_{k,l}$, we add the value of $\psi_j^{d_1-1} \sum_{i\in\cup_{j_0\in[N]}\tilde{\mathcal{C}}_{j_0}} w_{k,l}^{\langle i\rangle}$ to the current β. After finishing the loop over d_1, each working database j asks for the value of β along with $A_{W,2}^{\langle\theta_j\rangle,(j_0)}$ from client θ_j. Although we only consider the first B symbols of each submodel here, this idea can be extended to all the updates in M_Γ through repetition and time-sharing. In this way, each submodel in the union Γ can be updated as desired through the training data from the active clients $\cup_{j\in[N]}\mathcal{C}_j\backslash\tilde{\mathcal{C}}_j$.

Client Late-Arrivals Robustness: If the answers generated by some clients in the first step of the FSL-PSU or FSL-write phases arrive at the server late, which is different from the wrong judgement made by the databases that these clients have dropped-out, the privacy constraint (8.5) is still satisfied, i.e., these late answers do not disclose any extra information about the local data possessed by these late-arriving clients. The fact that the reliability constraint is satisfied is inherited from the one in client drop-outs robustness. For each client group $\mathcal{C}_j$, we assume that a subset of clients with indices $\bar{\mathcal{C}}_j$ cause the answer late-arrivals. In the FSL-PSU phase, for all $j \in [N]$, due to the wrong judgement, the broadcasting answer $A_{U,2}^{\langle\theta_j\rangle,([N])}$ takes the following form according to the analysis in client drop-outs robustness,

$$A_{U,2}^{\langle\theta_j\rangle,([N])} = \left\{ c \sum_{i\in\mathcal{C}_j\backslash\bar{\mathcal{C}}_j} (Y_k^{\langle i\rangle} + w_k^{\langle i\rangle}) + w_k^{(j)} + \hat{R}_k \colon k \in [K] \right\}, \quad \forall j \in [N] \tag{8.142}$$

It is easy to see that no information about the incidence vectors $Y^{\langle\bar{\mathcal{C}}_j\rangle}$ can be extracted by database j from the late answers $A_{U,1}^{\langle\bar{\mathcal{C}}_j\rangle,(j)}$ and the answers $A_{U,2}^{\langle\theta_{[N]}\rangle,(j)}$ in

the form of (8.142) because of the extra client-side common randomness $w_k^{(j)}$. In the FSL-write phase, the privacy constraint can be guaranteed in the same way because of the randomness selected by the routing clients or the extra client-side common randomness $w_{k,d_2}^{(j)}$. This conclusion is still true for any set of databases with size smaller than or equal to J as the randomness truly used here is completely unknown to any set of J colluding databases.

Database Drop-Outs Robustness: Under this situation, we rely on the remaining working databases to complete the training task normally. For instance, say database f drops-out and cannot provide any helpful response to the clients in this FSL round. In the FSL-PSU phase, each working database $j \in [N]\backslash f$ can still receive the answers $A_{U,1}^{\langle \mathcal{C}_j \rangle,(j)}$ and $A_{U,2}^{\langle \theta_{[N]\backslash f} \rangle,(j)}$, but cannot receive any answer from the routing client θ_f in the client group $\mathcal{C}_f$. Thus, our adjusted aim is to derive the desired submodel union $\Gamma\backslash\Gamma^{\langle \mathcal{C}_f \rangle}$ instead. For all $k \in [K]$, in order to obtain the needed $c\sum_{i\in[C]\backslash\mathcal{C}_f} Y_k^{\langle i \rangle}$ from $A_{U,2}^{\langle \theta_{[N]\backslash f} \rangle,(j)}$, each working database j can ask for the value of $c\sum_{i\in\mathcal{C}_f}(w_k^{\langle i \rangle})+w_k^{(f)}$ from the routing client θ_j through one more communication step. After eliminating the known server-side common randomness $\hat{R}_k$, we have the following identity for all $k \in [K]$ as all the client-side common randomness can be cancelled,

$$\sum_{j_0\in[N]\backslash f}(c\sum_{i\in\mathcal{C}_{j_0}}(Y_k^{\langle i \rangle}+w_k^{\langle i \rangle})+w_k^{(j)})+c\sum_{i\in\mathcal{C}_f}(w_k^{\langle i \rangle})+w_k^{(f)} = c\sum_{i\in[C]\backslash\mathcal{C}_f} Y_k^{\langle i \rangle} \tag{8.143}$$

In the FSL-write phase, likewise, each working database $j \in [N]\backslash f$ can obtain the answers $A_{W,1}^{\langle \mathcal{C}_j \rangle,(j)}$ and $A_{W,2}^{\langle \theta_{[N]\backslash f} \rangle,(j)}$ without any answer from the routing client θ_f for the client group $\mathcal{C}_f$. Following the analysis of the reliability constraint in the last subsection, if X_{d_1,d_2}^k is a submodel symbol, say $M_{k,l}$, we have the following result in each database j when the known server-side common randomness is eliminated,

$$\psi_j^{d_1-1} \sum_{j_0 \in [N]\backslash f} \mathcal{X}_{d_1,d_2}^k = \psi_j^{d_1-1}(M_{k,l} + \sum_{j_0 \in [N]\backslash f} \sum_{i \in \mathcal{C}_{j_0}} (\Delta_{k,l}^{\langle i \rangle} + w_{k,l}^{\langle i \rangle})) \tag{8.144}$$

$$= \psi_j^{d_1-1}(M_{k,l} + \sum_{i \in [C]\backslash \mathcal{C}_f} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in [C]\backslash \mathcal{C}_f} w_{k,l}^{\langle i \rangle}) \tag{8.145}$$

Otherwise, if X_{d_1,d_2}^k is a randomness symbol, and we still have

$$\psi_j^{d_1-1} \sum_{j_0 \in [N]\backslash f} \mathcal{X}_{d_1,d_2}^k = \psi_j^{d_1-1} \mathcal{W} \tag{8.146}$$

Hence, for all $k \in \Gamma$ and all $d_2 \in [D]$, each routing client θ_j first selects an empty value $\beta = 0$. Then, for each d_1 in the set $[D]$, provided X_{d_1,d_2}^k is some submodel symbol $M_{k,l}$, the value of $\psi_j^{d_1-1} \sum_{i \in \mathcal{C}_f} w_{k,l}^{\langle i \rangle}$ is added to the current β. When the loop is finished, each working database j asks for the value of β along with $A_{W,2}^{\langle \theta_j \rangle,(j_0)}$ from client θ_j. Again, this idea works for all the submodel updates in the union $\Gamma\backslash\Gamma^{\langle \mathcal{C}_f \rangle}$ via repetition and time-sharing. That means even if database f drops-out, the FSL process can proceed normally without collecting the updates from the clients in the client group $\mathcal{C}_f$. This remedy for single database drop-out can be extended to the case of multiple database drop-outs by replacing index f with an index set.

Database Failure Robustness: Once one of the available databases at the server side fails permanently rather than drops-out temporarily, our solution is to construct a replacement database such that the FSL protocol configured in the beginning is not affected by the database failure at all. If database f fails, for the submodels in M_{Γ}, we can rely on $N-1$ routing clients with indices $\theta_{[N]\setminus f}$ to transmit the submodel update information that is an encoding function G_f of the latest submodels M'_{Γ} and their coupled refreshed server-side common randomness $\mathcal{R}'_S$ to the replacement database. The concrete realization is inherited from the above-mentioned remedy for single database drop-out. Meanwhile, for the other submodels in $M_{[K]\setminus\Gamma}$, the clients can be used to forward the database f repair information that is an encoding function G_f of the previous submodels $M_{[K]\setminus\Gamma}$ and their coupled previous server-side common randomness $\mathcal{R}_S$ to the replacement database according to the database repair part in the proof of Theorem 8.2. Since the size of this repair information is generally large, a large number of clients can work in parallel, i.e., each client forwards a small part of the overall repair information to reduce the communication time. In addition, to make the plain server-side common randomness $\hat{\mathcal{R}}_S$ in the replacement database consistent with the other databases, all the plain server-side common randomness $\hat{\mathcal{R}}_S$ is refreshed through the approach in FSL-CRR phase whether it is coupled with submodels in M_{Γ} or in $M_{[K]\setminus\Gamma}$. If multiple databases fail, the failed databases can be repaired one-by-one for each FSL round.

Active Adversary Robustness: If there is an active adversary taking control of any arbitrary A databases, the responses received by the clients from these A

corrupted databases will not be reliable any more. The core idea of our solution is to force the clients to download more responses than usual and then extract required information from different databases. In the original FSL-PSU phase, for all $j \in [N]$, each routing client θ_j needs to download $D_{U,2}^{\langle\theta_j\rangle,(j)}$ from database j. Now, each client i in the client group $\mathcal{C}_j$ sends the answer in the form of (8.114) to $2A+1$ working databases in an efficient manner. Afterwards, these $2A+1$ working databases individually transmit the response in the form of (8.115) back to client θ_j. Hence, client θ_j is able to decode the correct $D_{U,2}^{\langle\theta_j\rangle,(j)}$ according to the error correcting property of $[2A+1, 1, 2A+1]$ repetition code. In the original FSL-write phase, for all $i \in [C]$, each database i needs to download $D_{W,1}^{\langle i\rangle,(\mathcal{N}_i)}$ from D databases in order to recover the desired submodels M_Γ. If the maximal number of symbols downloaded from a database is η, then $D+2A$ working databases individually transmit η symbols with the same positions to client i. Hence, client i is able to decode M_Γ without any error according to the error correcting property of $[2A+D, D, 2A+1]$ Reed-Solomon code. Then, for all $j \in [N]$, each routing client θ_j needs to download $D_{W,2}^{\langle\theta_j\rangle,(j)}$ from database j. By utilizing $[2A+1, 1, 2A+1]$ repetition code again, the process is the same as the above-mentioned one for downloading $D_{U,2}^{\langle\theta_j\rangle,(j)}$.

For the remaining auxiliary phases, in the FSL-CRR phase, there is no need for the clients to download any information from the databases. Therefore, the existence of active adversary has no influence on this phase. However, in the original FSL-CRG phase, every $J+1$ databases are collaborating with each other to distribute client-side common randomness symbols. As a malicious database can send the same symbol with different values to different clients in the broadcasting process,

the procedure in the current FSL-CRG phase will be a bit more complicated. Now, each database will not broadcast its randomly selected symbol any more. Instead, the server-side partial common randomness is broadcast to the clients. To be more concrete, for the first type of client-side common randomness, by adjusting the FSL-CRR phase such that every pair of clients are forwarding the data to exactly $2A+1$ databases, a server-side partial common randomness symbol $\mathcal{R}_{S,1}$ can be owned by these $2A+1$ databases. Afterwards, these $2A+1$ databases broadcast this symbol to N routing clients. Thus, each routing client can decode the desired $\mathcal{R}_{S,1}$ reliably through $[2A+1, 1, 2A+1]$ repetition code. Likewise, another $2A+1$ databases are employed to broadcast another server-side partial common randomness symbol $\mathcal{R}_{S,2}$ to all the routing clients. Following this way, each routing client can obtain the same set of server-side partial common randomness symbols $\{\mathcal{R}_{S,1}, \mathcal{R}_{S,2}, \ldots, \mathcal{R}_{S,J+1}\}$ from $(J+1)(2A+1)$ different databases. By adding these $(J+1)$ symbols up, N routing clients can share a randomness symbol that is unknown to any J colluding databases. Therefore, the FSL-CRG phase can proceed as we desire even in the presence of an active adversary. By repeating this step $\mathcal{L}-1$ times, the required set $\{w_1, w_2, \ldots, w_{\mathcal{L}-1}\}$ can be shared among these N routing databases. By incorporating this idea into the original FSL-CRG phase, the necessary client-side common randomness can also be obtained by the other clients. The process of distributing the client-side common randomness symbol c is very similar where all the selected clients are equally treated.

8.6.7 Performance Evaluation

We consider the performance of our proposed achievable scheme in this section. As we formulated in Section 8.2, the evaluation consists of communication cost and storage cost.

Communication Cost: First, we consider the basic scheme without any adjustment for additional robustness. In the FSL-CRG phase, for each client-side common randomness symbol in the form of $w_k^{(j)}$ or $w_{k,d_2}^{(j)}$, the communication cost is $(J+1)N$. For each client-side common randomness symbol in the form of $w_k^{\langle i \rangle}$ or $w_{k,l}^{\langle i \rangle}$, the communication cost is at most $(J+1)(N+2)$. The communication cost for the client-side common randomness symbol c is $(J+1)C$, which can be neglected as it is distributed only once. Therefore, the total communication cost in this phase is at most $(J+1)N(N-1)(K+D|\Gamma|L)+(J+1)(N+2)(C-1)(K+|\Gamma|L)$, which is approximately $\mathcal{O}(C(K+|\Gamma|L))$ since the values of D, J, N are small compared with the values of K, L. In the FSL-PSU phase, the communication cost is $(C+N+N^2)K$, which is approximately $\mathcal{O}(CK)$. In the FSL-write phase, the communication cost for the recovery of the desired submodels M_Γ across all the clients is $C \cdot \frac{C_1}{B}|\Gamma|L$ where $\frac{C_1}{B}$ comes from (8.9) in Theorem 8.2. For the remaining transmission, the communication cost is at most $(C+N)|\Gamma|L+N^2D|\Gamma|L$. Therefore, the total communication cost in this phase is approximately $\mathcal{O}(C|\Gamma|L))$. In the FSL-CRR phase, for each server-side common randomness symbol whether it is in the form of $\hat{\mathcal{R}}_k$ or $\hat{R}_{k,l}$, the communication cost is always $2N$. Therefore, the total communication cost in

this phase is $2N(K+|\Gamma|L)$, which is approximately $\mathcal{O}(K+|\Gamma|L)$. By summing the communication cost results from all four phases, the overall communication cost in our one-round distributed FSL achievable scheme is approximately $\mathcal{O}(C(K+|\Gamma|L))$.

When the situations of client drop-outs, client late-arrivals or database drop-outs happen, it is easy to show that the overall communication cost will not be influenced significantly. Further, in the presence of an active adversary, as the value of A is also small, the order of the overall communication cost will not change. It will still be $\mathcal{O}(C(K+|\Gamma|L))$. However, once the situation of database failure happens, for each failed database, the additional communication cost of transmitting the database repair information concerning $M_{[K]\backslash\Gamma}$ to the replacement database is $2\frac{C_2}{B}(K-|\Gamma|)L$, where the coefficient 2 comes from the fact that clients are used to route the information between databases and $\frac{C_2}{B}$ comes from (8.10) in Theorem 8.2. In addition, the additional communication cost of refreshing server-side common randomness symbols $\{\hat{R}_{k,l} : k \in [K]\backslash\Gamma, l \in [L]\}$ is $2N(K-|\Gamma|)L$. Therefore, as the value of $|\Gamma|$ is generally much smaller than the value of K, the additional communication cost for the sake of database failure robustness is approximately $\mathcal{O}(KL)$.

Storage Cost: In order to ensure the security against eavesdropper (8.7) in the presence of a passive eavesdropper, without considering the server-side common randomness, the storage cost in each database is $\frac{S}{B}KL$ where $\frac{S}{B}$ comes from (8.11) in Theorem 8.2. In addition, the storage cost in each database for the plain server-side common randomness is $KL+L$. Therefore, the overall communication cost is $N(SKL+KL+L)$ which is approximately $\mathcal{O}(KL)$.

8.7 Conclusion

In this chapter, we proposed a new RSRC-based distributed FSL scheme that extends our previous two-database FSL scheme in Chapter 7. This new scheme has higher resilience than our previous scheme, while having the same order-wise communication cost and storage cost. More specifically, this new scheme is now fully robust against passive eavesdroppers, active adversaries, database failures, database drop-outs, client drop-outs and client late-arrivals.

In this work, we mainly considered the privacy and security from the perspective of the databases. In reality, it is also possible that the clients collude with each other or with the server [101, 128]. Furthermore, the clients can also be malicious and return arbitrarily erroneous answers to the server. These are interesting research directions. MDS coding or Lagrange coding in [129] across the clients can be utilized for this purpose.

Regarding our RSRC technique that aims to reduce the reconstruction communication cost, repair communication cost, and storage cost simultaneously by allowing information leakage, we did not investigate a converse proof in this chapter. It is quite likely that there exists a better coding scheme that outperforms our RSRC scheme in terms of part or all of the evaluation metrics. Another non-trivial point is how to group the databases and clients to improve the communication efficiency in the two auxiliary phases, i.e., FSL-CRG phase and FSL-CRR phase, in a practical implementation. These are basically optimization problems and would be interesting to explore in an actual distributed FSL configuration.

CHAPTER 9

Conclusions

In this dissertation, we utilized the information-theoretic techniques to explore secure computation and secure learning problems with SPIR serving as a starting point. We developed both new efficient and robust achievable schemes for these problems, and also converse bounds for them.

In Chapter 2, we investigated the two-party PSI problem over a finite set, which is the universal alphabet. We showed that the problem can be recast as an MM-SPIR problem with a fixed message size 1. This is under the assumption that the data sets and their corresponding incidence vectors can be stored in replicated and non-colluding databases. Furthermore, the elements in each data set are generated in an i.i.d. fashion under some probability distribution from the universal alphabet. To that end, we explored the information-theoretic capacity of MM-SPIR as a stand-alone problem. We showed that joint multi-message retrieval does not outperform the successive application of SM-SPIR. For the converse proof, we extended the proof techniques of SM-SPIR to the setting of multi-messages. To unify the query structures of MM-PIR and MM-SPIR, we proposed a new capacity-

achieving scheme as an alternative to the successive usage of the SM-SPIR scheme. Based on these results, we derived the optimal download cost for two-party PSI.

In Chapter 3, we considered an extended version of the SPIR problem, where the user randomly fetches a portion of the available shared common randomness at the databases. This fetched database common randomness can be viewed as a form of side information at the user. We showed that this side information can be utilized as auxiliary randomness data to increase the SPIR rate to the level of PIR rate. The non-trivial use of this new user-side common randomness makes single-database SPIR feasible. Finally, we determined the exact capacity region of the download cost, database-side common randomness and user-side common randomness. According to these results, we obtained the minimum download cost for two-party PSI with this type of auxiliary randomness data, especially in the setting where each party has one database to store the information.

In Chapter 4, we formulated the MP-PSI problem by investigating a specific mode of communication where only a single communication round between the leader party and client parties is needed. Under this assumption, we proposed a novel achievable scheme for MP-PSI. Our scheme hinges on a careful design and sharing of randomness among the client parties prior to starting the MP-PSI process. This is not a straightforward extension to the two-party PSI scheme. By means of this type of auxiliary randomness data that is distributed across the client parties, the download cost of our scheme matches the sum of download cost of pair-wise PSI despite the stringent privacy constraint in MP-PSI. We note that this work provides only an achievable scheme with no claim of optimality.

In Chapter 5, we investigated the overall communication cost of two-database SPIR. As a first work of its kind, by utilizing CDS/CDMS as intermediates, we now understand how to construct the general upload cost for two-database SPIR, and develop a few principles of building two-database SPIR schemes. For a simple two-database SPIR example, we determined its exact optimal overall communication cost by providing a complete upload-download cost achievable region. All the conclusions obtained in this chapter can be applied to PSI as the PSI problem itself involves upload cost and download cost.

In Chapter 6, we investigated the capacity of two-party RSPIR. Even though the capacity of two-database SPIR is independent of the number of messages stored in the databases, the capacity of two-party RSPIR does depend on this value. In addition, a linear download cost for perfect digital blind box delivery can be achieved due to the equivalence between RSPIR and the digital blind box. An important application of RSPIR is the practical implementation of user-side common randomness introduced in Chapter 3. Another potential application arises in Chapter 7. We determined the exact capacity of two-party RSPIR for small number of messages.

In Chapter 7, we proposed a brand-new efficient and robust two-database FSL achievable scheme. The communication cost of our proposed scheme is order-wise similar to the communication cost of existing schemes with much weaker privacy guarantees. Compared to the existing schemes with similar privacy guarantees, our proposed scheme does not require noisy storage of the submodels in the databases. Our scheme is resilient against client drop-outs, client late-arrivals, and database drop-outs. The main ideas of this scheme are based on PSU and its variation

for private write, together with RSPIR and one-time pads for required common randomness generation at the client side. Neither the indices of the submodels updated within the union, nor their updated values are leaked to the databases.

In Chapter 8, we proposed a new RSRC-based distributed FSL achievable scheme that extends our two-database FSL achievable scheme in Chapter 7. This new scheme has higher resilience than our previous scheme, while having the same order-wise communication cost and storage cost. More specifically, this new scheme is now fully robust against passive eavesdroppers, active adversaries, database failures, database drop-outs, client drop-outs and client late-arrivals. In this work, we mainly considered the privacy and security from the perspective of the databases. Regarding our RSRC technique that aims to reduce the reconstruction communication cost, repair communication cost and storage cost simultaneously by allowing information leakage, we did not investigate a converse proof. Another non-trivial point is how to group the databases and clients to improve the communication efficiency during the two auxiliary phases in a practical implementation. These are basically optimization problems.

The contents of Chapter 2 are published in [72, 130], Chapter 3 in [75, 131], Chapter 4 in [74, 132], Chapter 5 in [115], Chapter 6 in [116], Chapter 7 in [125], Chapter 8 in [133].

Bibliography

[1] M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, pages 1–19. Springer, 2004.

[2] E. De Cristofaro and G. Tsudik. Practical private set intersection protocols with linear complexity. In *Proc. Int. Conf. Financ. Cryptogr. Data Security*, pages 143–159. Springer, 2010.

[3] H. Chen, K. Laine, and P. Rindal. Fast private set intersection from homomorphic encryption. In *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, pages 1243–1255. ACM, 2017.

[4] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung. Efficient robust private set intersection. In *Proc. Int. Conf. Appl. Cryptogr. Netw. Security*, pages 125–142. Springer, 2009.

[5] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *Journal of the ACM*, 45(6):965–981, November 1998.

[6] H. Sun and S. A. Jafar. The capacity of private information retrieval. *IEEE Trans. on Info. Theory*, 63(7):4075–4088, July 2017.

[7] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. *Journal of Computer and System Sciences*, 60(3):592–629, June 2000.

[8] H. Sun and S. A. Jafar. The capacity of symmetric private information retrieval. *IEEE Trans. on Info. Theory*, 65(1):322–329, January 2019.

[9] H. Sun and S. A. Jafar. The capacity of robust private information retrieval with colluding databases. *IEEE Trans. on Info. Theory*, 64(4):2361–2370, April 2018.

[10] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, C. Hollanti, and S. El Rouayheb. Private information retrieval schemes for coded data with arbitrary collusion patterns. In *IEEE ISIT*, June 2017.

[11] R. Tajeddine and S. El Rouayheb. Robust private information retrieval on coded data. In *IEEE ISIT*, June 2017.

[12] R. Bitar and S. El Rouayheb. Staircase-PIR: Universally robust private information retrieval. In *IEEE ITW*, pages 1–5, November 2018.

[13] Q. Wang and M. Skoglund. Symmetric private information retrieval from MDS coded distributed storage with non-colluding and colluding servers. *IEEE Trans. on Info. Theory*, 65(8):5160–5175, August 2019.

[14] Q. Wang and M. Skoglund. Linear symmetric private information retrieval for MDS coded distributed storage with colluding servers. In *IEEE ITW*, pages 71–75, November 2017.

[15] Q. Wang, H. Sun, and M. Skoglund. Symmetric private information retrieval with mismatched coded messages and randomness. In *IEEE ISIT*, pages 365–369, July 2019.

[16] Q. Wang and M. Skoglund. Secure symmetric private information retrieval from colluding databases with adversaries. In *Allerton Conference*, October 2017.

[17] T. Guo, R. Zhou, and C. Tian. On the information leakage in private information retrieval systems. *IEEE Trans. on Info. Forensics and Security*, 15:2999–3012, 2020.

[18] K. Banawan and S. Ulukus. The capacity of private information retrieval from coded databases. *IEEE Trans. on Info. Theory*, 64(3):1945–1956, March 2018.

[19] R. Freij-Hollanti, O. Gnilke, C. Hollanti, and D. Karpuk. Private information retrieval from coded databases with colluding servers. *SIAM Journal on Applied Algebra and Geometry*, 1(1):647–664, 2017.

[20] Y. Zhang and G. Ge. A general private information retrieval scheme for MDS coded databases with colluding servers. *Designs, Codes and Cryptography*, 87(11):2611–2623, 2019.

[21] S. Kumar, H.-Y. Lin, E. Rosnes, and A. Graell i Amat. Achieving maximum distance separable private information retrieval capacity with linear codes. *IEEE Trans. on Info. Theory*, 65(7):4243–4273, July 2019.

[22] H. Sun and S. A. Jafar. Private information retrieval from MDS coded data with colluding servers: Settling a conjecture by Freij-Hollanti et al. *IEEE Trans. on Info. Theory*, 64(2):1000–1022, 2018.

[23] K. Banawan and S. Ulukus. Multi-message private information retrieval: Capacity results and near-optimal schemes. *IEEE Trans. on Info. Theory*, 64(10):6842–6862, October 2018.

[24] Y. Zhang and G. Ge. Multi-file private information retrieval from MDS coded databases with colluding servers. Available at arXiv:1705.03186.

[25] K. Banawan and S. Ulukus. The capacity of private information retrieval from Byzantine and colluding databases. *IEEE Trans. on Info. Theory*, 65(2):1206–1219, February 2019.

[26] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, and C. Hollanti. Private information retrieval from coded storage systems with colluding, Byzantine, and unresponsive servers. *IEEE Trans. on Info. Theory*, 65(6):3898–3906, 2019.

[27] R. Tandon. The capacity of cache aided private information retrieval. In *Allerton Conference*, pages 1078–1082, October 2017.

[28] M. Kim, H. Yang, and J. Lee. Cache-aided private information retrieval. In *Asilomar Conference*, October 2017.

[29] Y.-P. Wei, K. Banawan, and S. Ulukus. Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching. *IEEE Trans. on Info. Theory*, 65(5):3215–3232, May 2019.

[30] Y.-P. Wei, K. Banawan, and S. Ulukus. Cache-aided private information retrieval with partially known uncoded prefetching: Fundamental limits. *IEEE JSAC*, 36(6):1126–1139, June 2018.

[31] S. Kumar, A. Graell i Amat, E. Rosnes, and L. Senigagliesi. Private information retrieval from a cellular network with caching at the edge. *IEEE Trans. on Commun.*, 67(7):4900–4912, July 2019.

[32] S. Kadhe, B. Garcia, A. Heidarzadeh, S. El Rouayheb, and A. Sprintson. Private information retrieval with side information. *IEEE Trans. on Info. Theory*, 66(4):2032–2043, April 2020.

[33] Z. Chen, Z. Wang, and S. A. Jafar. The capacity of T-private information retrieval with private side information. *IEEE Trans. on Info. Theory*, 66(8):4761–4773, August 2020.

[34] Y.-P. Wei, K. Banawan, and S. Ulukus. The capacity of private information retrieval with partially known private side information. *IEEE Trans. on Info. Theory*, 65(12):8222–8231, December 2019.

[35] S. P. Shariatpanahi, M. J. Siavoshani, and M. A. Maddah-Ali. Multi-message private information retrieval with private side information. In *IEEE ITW*, pages 1–5, November 2018.

[36] 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. In *Allerton Conference*, pages 180–187, October 2018.

[37] S. Li and M. Gastpar. Single-server multi-message private information retrieval with side information. In *Allerton Conference*, pages 173–179, October 2018.

[38] S. Li and M. Gastpar. Converse for multi-server single-message PIR with side information. In *CISS*, pages 1–6, 2020.

[39] Y.-P. Wei and S. Ulukus. The capacity of private information retrieval with private side information under storage constraints. *IEEE Trans. on Info. Theory*, 66(4):2023–2031, April 2020.

[40] H. Sun and S. A. Jafar. The capacity of private computation. *IEEE Trans. on Info. Theory*, 65(6):3880–3897, June 2019.

[41] M. Mirmohseni and M. A. Maddah-Ali. Private function retrieval. In *IWCIT*, pages 1–6, April 2018.

[42] Z. Chen, Z. Wang, and S. A. Jafar. The asymptotic capacity of private search. *IEEE Trans. on Info. Theory*, 66(8):4709–4721, August 2020.

[43] R. Tandon, M. Abdul-Wahid, F. Almoualem, and D. Kumar. PIR from storage constrained databases - coded caching meets PIR. In *IEEE ICC*, pages 1–7, 2018.

[44] M. A. Attia, D. Kumar, and R. Tandon. The capacity of private information retrieval from uncoded storage constrained databases. *IEEE Trans. on Info. Theory*, 66(11):6617–6634, November 2020.

[45] K. Banawan, B. Arasli, and S. Ulukus. Improved storage for efficient private information retrieval. In *IEEE ITW*, pages 1–5, August 2019.

[46] C. Tian. On the storage cost of private information retrieval. *IEEE Trans. on Info. Theory*, 66(12):7539–7549, December 2020.

[47] Y.-P. Wei, B. Arasli, K. Banawan, and S. Ulukus. The capacity of private information retrieval from decentralized uncoded caching databases. *Information*, 10, December 2019.

[48] K. Banawan, B. Arasli, Y.-P. Wei, and S. Ulukus. The capacity of private information retrieval from heterogeneous uncoded caching databases. *IEEE Trans. on Info. Theory*, 66(6):3407–3416, June 2020.

[49] N. Raviv, I. Tamo, and E. Yaakobi. Private information retrieval in graph-based replication systems. *IEEE Trans. on Info. Theory*, 66(6):3590–3602, June 2020.

[50] K. Banawan and S. Ulukus. Private information retrieval from non-replicated databases. In *IEEE ISIT*, pages 1272–1276, July 2019.

[51] K. Banawan and S. Ulukus. Private information retrieval through wiretap channel II: Privacy meets security. *IEEE Trans. on Info. Theory*, 66(7):4129–4149, July 2020.

[52] Q. Wang and M. Skoglund. On PIR and symmetric PIR from colluding databases with adversaries and eavesdroppers. *IEEE Trans. on Info. Theory*, 65(5):3183–3197, May 2019.

[53] Q. Wang, H. Sun, and M. Skoglund. The capacity of private information retrieval with eavesdroppers. *IEEE Trans. on Info. Theory*, 65(5):3198–3214, May 2019.

[54] H. Yang, W. Shin, and J. Lee. Private information retrieval for secure distributed storage systems. *IEEE Trans. on Info. Forensics and Security*, 13(12):2953–2964, December 2018.

[55] 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, September 2019.

[56] H. Sun and S. A. Jafar. Optimal download cost of private information retrieval for arbitrary message length. *IEEE Trans. Inf. Forensics Security*, 12(12):2920–2932, 2017.

[57] R. Zhou, C. Tian, H. Sun, and T. Liu. Capacity-achieving private information retrieval codes from MDS-coded databases with minimum message size. *IEEE Trans. on Info. Theory*, 66(8):4904–4916, August 2020.

[58] H. Sun and S. A. Jafar. Multiround private information retrieval: Capacity and storage overhead. *IEEE Trans. on Info. Theory*, 64(8):5743–5754, 2018.

[59] K. Banawan and S. Ulukus. Asymmetry hurts: Private information retrieval under asymmetric traffic constraints. *IEEE Trans. on Info. Theory*, 65(11):7628–7645, November 2019.

[60] K. Banawan and S. Ulukus. Noisy private information retrieval: On separability of channel coding and information retrieval. *IEEE Trans. on Info. Theory*, 65(12):8232–8249, December 2019.

[61] R. G. L. D'Oliveira and S. El Rouayheb. One-shot PIR: Refinement and lifting. *IEEE Trans. on Info. Theory*, 66(4):2443–2455, April 2020.

[62] R. Tajeddine, A. Wachter-Zeh, and C. Hollanti. Private information retrieval over random linear networks. *IEEE Trans. Inf. Forensics Security*, 15:790–799, 2020.

[63] S. Vithana, K. Banawan, and S. Ulukus. Semantic private information retrieval. *IEEE Trans. on Info. Theory*, 68(4):2635–2652, April 2022.

[64] M. O. Rabin. How to exchange secrets with oblivious transfer. In *IACR Eprint archive*, 2005. Originally published as: Technical Report TR-81, Aiken Computation Lab, Harvard University, 1981.

[65] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, June 1985.

[66] C. Tian, H. Sun, and J. Chen. Capacity-achieving private information retrieval codes with optimal message size and upload cost. *IEEE Trans. on Info. Theory*, 65(11):7613–7627, November 2019.

[67] I. Samy, M. Attia, R. Tandon, and L. Lazos. Asymmetric leaky private information retrieval. *IEEE Trans. on Info. Theory*, 67(8):5352–5369, August 2021.

[68] L. Kissner and D. Song. Privacy-preserving set operations. In *Advances in Cryptology – CRYPTO 2005*, pages 241–257, 2005.

[69] R. Li and C. Wu. An unconditionally secure protocol for multi-party set intersection. In *Applied Cryptography and Network Security*, pages 226–236, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[70] C. Hazay and M. Venkitasubramaniam. Scalable multi-party private set-intersection. In *Public-Key Cryptography – PKC 2017*, pages 175–203. Springer Berlin Heidelberg, 2017.

[71] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu. Practical multi-party private set intersection from symmetric-key techniques. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1257–1272. Association for Computing Machinery, 2017.

[72] Z. Wang, K. Banawan, and S. Ulukus. Private set intersection: A multi-message symmetric private information retrieval perspective. *IEEE Trans. on Info. Theory*, 68(3):2001–2019, March 2022.

[73] Y. Zhou, Q. Wang, H. Sun, and S. Fu. The minimum upload cost of symmetric private information retrieval. In *IEEE ISIT*, pages 1030–1034, June 2020.

[74] Z. Wang, K. Banawan, and S. Ulukus. Multi-party private set intersection: An information-theoretic approach. *IEEE Jour. on Selected Areas in Info. Theory*, 2(1):366–379, March 2021.

[75] Z. Wang and S. Ulukus. Symmetric private information retrieval at the private information retrieval rate. *IEEE Jour. on Selected Areas in Info. Theory*, 3(2):350–361, June 2022.

[76] J. Cheng, N. Liu, W. Kang, and Y. Li. The capacity of symmetric private information retrieval under arbitrary collusion and eavesdropping patterns. *IEEE Trans. on Info. Forensics and Security*, 17:3037–3050, August 2022.

[77] H.-M. Sun and S.-P. Shieh. Secret sharing in graph-based prohibited structures. In *IEEE Infocom*, April 1997.

[78] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Advances in Cryptology - EUROCRYPT*, May 2005.

[79] B. Applebaum, B. Arkis, P. Raykov, and P. N. Vasudevan. Conditional disclosure of secrets: Amplification, closure, amortization, lower-bounds, and separations. In *Advances in Cryptology – CRYPTO*, 2017.

[80] Z. Li and H. Sun. Conditional disclosure of secrets: A noise and signal alignment approach. *IEEE Trans. on Commun.*, 2022. Early Access.

[81] Z. Li and H. Sun. On the linear capacity of conditional disclosure of secrets. Available at arXiv:2106.04483.

[82] https://en.wikipedia.org/wiki/Gashapon#cite_note-2.

[83] M. Fujihara and A. Shibuya. How is the Gacha system reported on in Japan? In *Digital Games Research Association (DiGRA) International Conference: Play Everywhere*, June 2020.

[84] K. Charnsil, K. Choochuen, et al. 3D-gARt—A new Gachapon 3D-printed toy played with augmented reality and story narration. In *IEEE Global Conf. on Life Sciences and Technologies (LifeTech)*, March 2022.

[85] B. McMahan and D. Ramage. Federated learning: Collaborative machine learning without centralized training data. Available at https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

[86] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, March 2019.

[87] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen. Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[88] O. Goldreich and R. Ostrovsky. Software protection and simulation on oblivious RAMs. *Journal of the ACM*, 43(3):431–473, May 1996.

[89] S. Lu and R. Ostrovsky. Distributed oblivious RAM for secure two-party computation. In *Theory of Cryptography Conference*, page 377–396, March 2013.

[90] E. Stefanov, M. Van Dijk, et al. Path ORAM: An extremely simple oblivious RAM protocol. *Journal of the ACM*, 65(4):1–26, April 2018.

[91] Z. Jia and S. A. Jafar. X-secure T-private information retrieval from MDS coded storage with Byzantine and unresponsive servers. *IEEE Tran. on Info. Theory*, 66(12):7427–7438, December 2020.

[92] Z. Jia and S. A. Jafar. X-secure T-private federated submodel learning with elastic dropout resilience. *IEEE Trans. on Info. Theory*, 68(8):5418–5439, August 2022.

[93] S. Vithana and S. Ulukus. Efficient private federated submodel learning. In *IEEE ICC*, pages 3394–3399, May 2022.

[94] S. Vithana and S. Ulukus. Private read update write (PRUW) in federated submodel learning (FSL): Communication efficient schemes with and without sparsification. Available at arXiv:2209.04421.

[95] S. Vithana and S. Ulukus. Private read-update-write with controllable information leakage for storage-efficient federated learning with top r sparsification. Available at arXiv:2303.04123.

[96] K. Bonawitz, V. Ivanov, et al. Practical secure aggregation for privacy preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, page 1175–1191, 2017.

[97] J. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova. Secure single-server aggregation with (poly)logarithmic overhead. In *Cryptology ePrint Archive*, 2020.

[98] J. So, B. Guler, and A. S. Avestimehr. Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning. In *Cryptology ePrint Archive*, 2020.

[99] S. Kadhe and K. Ramchandran N. Rajaraman abd O. O. Koyluoglu. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. Available at arXiv:2009.11248.

[100] K. Frikken. Privacy-preserving set union. In *Applied Cryptography and Network Security*, pages 237–252, 2007.

[101] Y. Zhao and H. Sun. Information theoretic secure aggregation with user dropouts. *IEEE Trans. on Info. Theory*, 68(11):7471–7484, November 2022.

[102] K. Wan, H. Sun, M. Ji, and G. Caire. Information theoretic secure aggregation with uncoded groupwise keys. Available at arXiv:2204.11364.

[103] Y. Zhao and H. Sun. Secure summation: Capacity region, groupwise key, and feasibility. Available at arXiv:2205.08458.

[104] J. So, C. He, et al. Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. In *Proceedings of Machine Learning and Systems*, pages 694–720, 2022.

[105] S. Vithana, Z. Wang, and S. Ulukus. Private information retrieval and its applications: An introduction, open problems, future directions. Available at arXiv:2304.14397.

[106] K. V. Rashmi, N. B. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran. A solution to the network challenges of data recovery in erasure-coded distributed storage systems: A study on the facebook warehouse cluster. In *Proceedings of the 5th USENIX Conference on Hot Topics in Storage and File Systems*, June 2013.

[107] K. V. Rashmi, N. B. Shah, K. Ramchandran, and P. V. Kumar. Information-theoretically secure erasure codes for distributed storage. *IEEE Trans. on Info. Theory*, 64(3):1621–1646, March 2018.

[108] E. D. Karnin, J. W. Greene, and M. E. Hellman. On secret sharing systems. *IEEE Trans. on Info. Theory*, 29(1):35–41, January 1983.

[109] H. Yamamoto. Secret sharing system using (k, l, n) threshold scheme. *Electronics and Communications in Japan, Part 1*, 69(9):46–54, 1986.

[110] B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. *IACR Cryptology ePrint Archive*, 1998:3, 1997.

[111] S. Kadhe, A. Heidarzadeh, A. Sprintson, and O. O. Koyluoglu. Single-server private information retrieval schemes are equivalent to locally recoverable coding schemes. *IEEE Jour. on Selected Areas in Info. Theory*, 2(1):391–402, 2021.

[112] M. J. Siavoshani, S. P. Shariatpanahi, and M. A. Maddah-Ali. Private information retrieval for a multi-message scenario with private side information. *IEEE Trans. on Commun.*, 69(5):3235–3244, May 2021.

[113] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Thirty-First Annual ACM Symposium on Theory of Computing*, page 245–254. ACM, 1999.

[114] B. Albert-László. *Network Science*. Cambridge university press, 2016.

[115] Z. Wang and S. Ulukus. Communication cost of two-database symmetric private information retrieval: A conditional disclosure of multiple secrets perspective. In *IEEE ISIT*, pages 402–407, June 2022.

[116] Z. Wang and S. Ulukus. Digital blind box: Random symmetric private information retrieval. In *IEEE ITW*, pages 95–100, November 2022.

[117] Y. Lu, Z. Jia, and S. A. Jafar. Double blind T-private information retrieval. *IEEE Jour. on Selected Areas in Info. Theory*, 2(1):428–440, March 2021.

[118] J. Zhu, Q. Yan, and X. Tang. Multi-user blind symmetric private information retrieval from coded servers. *IEEE Jour. on Selected Areas in Commun.*, 40(3):815–831, March 2022.

[119] C. Naim, R. G. L. D'Oliveira, and S. El Rouayheb. Private multi-group aggregation. *IEEE Jour. on Selected Areas in Commun.*, 40(3):800–814, March 2022.

[120] H. Corrigan-Gibbs and D. Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation*, page 259–282, 2017.

[121] M. Kim and J. Lee. Information-theoretic privacy in federated submodel learning. Available at arXiv:2008.07656.

[122] S. Vithana and S. Ulukus. Private read update write (PRUW) with storage constrained databases. In *IEEE ISIT*, pages 2391–2396, June 2022.

[123] S. Vithana and S. Ulukus. Private federated submodel learning with sparsification. In *IEEE ITW*, pages 410–415, November 2022.

[124] S. Vithana and S. Ulukus. Rate distortion tradeoff in private read update write in federated submodel learning. In *Allerton Conference*, October 2022.

[125] Z. Wang and S. Ulukus. Private federated submodel learning via private set union. IEEE Trans. on Info. Theory. Submitted January 2023. Also available at arXiv:2301.07686.

[126] M. Yoshida, T. Fujiwara, and M. P. C. Fossorier. Optimal uniform secret sharing. *IEEE Trans. on Info. Theory*, 65(1):436–443, January 2019.

[127] K. V. Rashmi, N. B. Shah, and P. V. Kumar. Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction. *IEEE Trans. on Info. Theory*, 57(8):5227–5239, August 2011.

[128] Z. Li, Y. Zhao, and H. Sun. Weakly secure summation with colluding users. Available at arXiv:2304.09771.

[129] 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 *International Conference on Artificial Intelligence and Statistics*, pages 1215–1225, 2019.

[130] Z. Wang, K. Banawan, and S. Ulukus. Private set intersection using multi-message symmetric private information retrieval. In *IEEE ISIT*, pages 1035–1040, June 2020.

[131] Z. Wang and S. Ulukus. Symmetric private information retrieval with user-side common randomness. In *IEEE ISIT*, pages 2119–2124, July 2021.

[132] Z. Wang, K. Banawan, and S. Ulukus. An information-theoretic scheme for multi-party private set intersection. In *IEEE ISIT*, pages 1112–1117, July 2021.

[133] Z. Wang and S. Ulukus. Fully robust federated submodel learning in a distributed storage system. IEEE Trans. on Info. Theory. Submitted June 2023. Also available at arXiv:2306.05402.