Simplified PIR and CDS Protocols and Improved Linear Secret-Sharing Schemes

Bar Alon
Georgetown University

Amos Beimel Or Lasri
Ben Gurion University

October 1, 2025

Abstract

We consider 3 related cryptographic primitives, private information retrieval (PIR) protocols, conditional disclosure of secrets (CDS) protocols, and secret-sharing schemes; these primitives have many applications in cryptography. We study these primitives requiring information-theoretic security. The complexity of the three primitives has been dramatically improved in the last few years and they are closely related, i.e., the 2-server PIR protocol of Dvir and Gopi (J. ACM 2016) was transformed to construct the CDS protocols of Liu, Vaikuntanathan, and Wee (CRYPTO 2017, Eurocrypt 2018) and these CDS protocols are the main ingredient in the construction of the best-known secret-sharing schemes. To date, the message size required in PIR and CDS protocols and the share size required in secret-sharing schemes are not understood and there are big gaps between their upper bounds and lower bounds. The goal of this paper is to try to better understand the upper bounds by simplifying current constructions and supplying tools for improving their complexity. We obtain the following results:

- We simplify, abstract, and generalize the 2-server PIR protocol of Dvir and Gopi (J. ACM 2016), the recent multi-server PIR protocol of Ghasemi, Kopparty, and Sudan (STOC 2025), and the 2-server and multi-server CDS protocols of Liu et al. (CRYPTO 2017, Eurocrypt 2018) and Beimel, Farràs, and Lasri (TCC 2023). In particular, we present one PIR protocol generalizing both the 2-server and multi-server PIR protocols using general share conversions. For $k \geq 3$ servers, our main contribution is a simpler proof the correctness of the protocol, avoiding the partial derivatives and interpolation polynomials used by Ghasemi et al.
- In addition to simplifying previous protocols, our 2-server protocols can use a relaxed variant of matching vectors over any m that is the product of two distinct primes. Our constructions do not improve the communication complexity of PIR and CDS protocols; however, construction of better relaxed matching vectors over *any* m that is the product of two distinct primes will improve the communication complexity of 2-server PIR and 2-server CDS protocols.
- In many applications of secret-sharing schemes it is important that the scheme is linear, e.g., by using the fact that parties can locally add shares of two secrets and obtain shares of the sum of the secrets. In an independent result, we provide a construction of linear secret-sharing schemes for n-party access structures with improved share size of $2^{0.7563n}$. Previously, the best share size for linear secret-sharing schemes was $2^{0.7576n}$ and it is known that for most n-party access structures the shares' size is at least $2^{0.5n}$. This result is achieved by a reduction to unbalanced CDS protocols (compared to balanced CDS protocols in previous constructions).

1 Introduction

Private information retrieval (PIR) protocols, conditional disclosure of secrets (CDS) protocols, and secret-sharing schemes are cryptographic primitives that have many applications. We study these primitives requiring information-theoretic security. The three primitives are closely related, e.g., the same techniques are used to construct PIR and CDS protocols, and CDS protocols are used to construct secret-sharing schemes for arbitrary access structures. Furthermore, the goal of these primitives is to protect the secrecy of some inputs (i.e., a database, private inputs of servers, or a secret), and they are non-interactive.

The complexity of PIR and CDS protocols and secret-sharing schemes has been dramatically improved in the last few years; yet their complexity is far from being understood. For example, the share size in the best-known secret-sharing schemes for arbitrary n-party access structure is exponential, i.e., $2^{O(n)}$, while the best lower bound is $\Omega(n^2/\log n)$ [31, 30]. Determining the optimal complexity of these primitives is a major open problem. Improving the known upper bounds will lead to better complexity in the protocols that use them, and proving better lower bounds will show their limitations. Furthermore, understanding the exact complexity of these primitives may lead to understanding the optimal complexity of more complex interactive cryptographic primitives, e.g., the communication complexity required for information-theoretic secure multi-party computation (MPC) protocols. Our goal in this paper is to advance the understanding of the upper bounds for PIR and CDS protocols and secret-sharing schemes, especially linear secret-sharing schemes. Towards this goal, we will try to simplify and generalize the current constructions, provide new tools for better constructions, and construct better schemes.

1.1 PIR, CDS, and Linear Secret Sharing

Before presenting our results, we informally discuss the primitives we study in this work.

Private information retrieval. Private information retrieval (PIR) protocols enable a user to obtain an item from a database held by two or more servers such that each server does not learn information on the retrieved item. Specifically, in a k-server PIR protocol, a user holds an index $i \in [N]$, and the servers hold the same database $D \in \{0,1\}^N$. The goal is for the user to learn D_i, using one round of communication, without revealing any information about i to each server. To achieve this goal, the user computes queries $q_1, \ldots, q_k$ and sends q_j to the j-th server (for $1 \leq j \leq k$). Each server computes an answer, based on the query that it got and the database, and sends the answer to the user, who reconstructs D_i from the index, its randomness, and the answers.

Private information retrieval protocols were introduced by Chor, Goldreich, Kushilevitz, and Sudan [27] in 1995. Since then, the communication complexity of private information retrieval protocols has been studied in a line of works [2, 47, 50, 17, 56, 72, 73, 39, 52, 26, 36, 45]. Specifically, Efremenko [39] constructed a 3-server PIR protocol with query length $2^{\tilde{O}(\sqrt{\log N})}$ and 2^r-server PIR protocols with query length $2^{\tilde{O}(\sqrt[r]{\log N})}$; the answer length in these protocols is $O(1)$. Dvir and Gopi [36] broke the $N^{1/3}$ barrier for the communication complexity for 2-server PIR protocols, showing a construction with communication complexity of $2^{\tilde{O}(\sqrt{\log N})}$. Very recently, Ghasemi, Kopparty, and Sudan [45] constructed a 3-server PIR protocol with query and answer length $2^{\tilde{O}(\sqrt[3]{\log N})}$. The communication complexity of k-server PIR protocols for $k \geq 6$ was improved by Itoh and Suzuki [51, 52], Chee, Feng, Ling, Wang, and Zhang [26], DG, and GKS. For example,

there is a 6-server PIR protocol with communication complexity $2^{\tilde{O}(\sqrt[4]{\log N})}$ (this is implied by the results of [51, 3, 45] and was overlooked in previous papers). See Theorem 6.6 for the best-known PIR protocols. The PIR protocols in these works are based on matching vector families. Beimel, Ishai, Kushilevitz, and Orlov [20] generalize the 3-server protocol of Efremenko by using share conversions; specifically, they can use matching vectors over $\mathbb{Z}_m$ for a larger set of products of primes. The best known lower bound on the total communication complexity of 2-server PIR protocols is $5 \log n$, proved by Wehner and de Wolf [71] (improving on [62, 57, 55]).

We next discuss an application of the new PIR protocol of Ghasemi et al. [45]. Beimel, Ishai, Kumaresan, and Kushilevitz [18] presented constructions of secure two-party computation protocols in the correlated randomness model from PIR protocols. In this model, before the inputs are known to the parties, during an offline phase, a pair of two correlated random strings (r_1, r_2) is drawn from some fixed joint distribution and each party receives one of the strings (e.g., r_1, r_2 are random bits such that $r_1 \wedge r_1 = 1$). During the online phase, the parties can use their correlated random strings to evaluate the function. Beimel et al. [18] constructed a secure two-party computation protocol for an arbitrary function $f : [N] \times [N] \to \{0, 1\}$ in the correlated randomness model from a 3-server PIR protocol with a database of size N^2; provided that the PIR protocol satisfies some properties, the communication complexity and the size of the correlated randomness in the secure two-party computation protocol is polynomial in the communication complexity of the PIR protocol. As the 3-server PIR protocol of Ghasemi et al. [45] satisfies these properties, we get a secure two-party computation protocol for a function $f : [N] \times [N] \to \{0, 1\}$ with communication complexity and the size of the correlated randomness of $2^{\tilde{O}(\sqrt[3]{\log(N)})}$.

Conditional disclosure of secrets. Conditional disclosure of secrets (CDS) protocols are a cryptographic primitive, introduced by Gertner, Ishai, Kushilevitz, and Malkin [44]. Their motivation was to construct symmetric private information retrieval protocols. CDS protocols were later used in constructions of other cryptographic applications, such as attribute-based encryption [43, 11, 70] and priced oblivious transfer [1]; they are a central tool in the construction of secret-sharing schemes for arbitrary access structures [59, 7, 10, 21].

In a CDS protocol, several servers hold the same secret and a common random string, and each server holds a private input. Additionally, there is a referee who knows the private inputs of all servers. The referee should learn the secret if and only if the private inputs of the servers satisfy some condition, specified by a predicate f. To achieve this goal, each server sends a single message to the referee; the message of each server is a function of the secret, the common random string, and its private input. The referee can reconstruct the secret from the messages if and only if the inputs satisfy the condition.

Constructions of CDS protocols were given in [44, 18, 43, 60, 61, 22, 5, 4]. The best known 2-server CDS protocol has message length $2^{\tilde{O}(\sqrt{\log N})}$ [60]. The best known lower bounds for 2-server CDS protocols is $\Omega(\log N)$, proved by Applebaum, Arkis, Raykov, and Vasudevan [6] (see also [43, 9]). The best k-server CDS construction is due to Liu et al. [61] and has a message size of $2^{\tilde{O}(\sqrt{k \cdot \log N})}$. Applebaum and Arkis [4] (improving on [5]) constructed a CDS protocol for long secrets, where the message length is only 4 times the length of the secret.

Secret sharing. Secret-sharing schemes, introduced by Shamir [69] and Blakley [24] for the threshold case and Ito, Saito, and Nishizeki [49] for the general case, allow a dealer holding a secret to distribute strings (called shares) to parties, such that only authorized sets of parties

can reconstruct the secret, while unauthorized sets learn nothing about the secret. The collection of authorized sets is called an access structure. Secret sharing has found many applications in cryptography, distributed computing, and complexity theory (see [13]). Identifying the necessary and sufficient share size of secret-sharing schemes for general access structures is a major open problem. The best-known schemes for n-parties access structure achieve share complexity of 2^{cn} for a constant $c < 1$ [59, 7, 8, 10], with Applebaum and Nir [10] constructing the current best scheme, which achieves share size $1.5^n < 2^{0.585n}$. On the negative side, the best lower bound on the total share size is $\Omega(n^2/\log n)$ due to Csirmaz [31, 30].

Linear secret sharing. Linear secret-sharing are schemes in which the shares are computed by applying a linear function (over some finite field) on the secret and some random elements from the field. Alternatively, these are schemes in which each share is a vector over the field and every authorized set reconstructs the secret by applying a linear function on its shares. In many applications of secret-sharing schemes, it is important that the scheme is linear, e.g., they use the fact that parties can locally add shares of two secrets and obtain shares of the sum of the secrets. Such applications include the secure multi-party computation protocol that are secure against an arbitrary (Q2) adversary structure [29] and the construction of public-key (multi-user) attribute-based encryption [11, 70]. Linear secret-sharing schemes have additional advantages, e.g., they are homomorphic, the sharing and reconstruction are efficient, and they are closed under duality. Prior to our work, the best previous linear scheme of [10] has share size $2^{0.7576n}$ for any n-party access structure. Until recently, most of the constructions of secret-sharing schemes were linear, e.g., [69, 25, 49, 23, 54].

Linear secret-sharing schemes are equivalent to monotone span programs, a linear-algebraic model of computation introduced by Karchmer and Wigderson [54]. Lower bounds on the share size in linear secret-sharing schemes and monotone span programs for explicit access structures were proven in [16, 12, 41, 42, 28, 66, 67]; the best result is that there exist explicit access structures for which every linear secret-sharing scheme realizing the access structures has shares of length at least 2^{cn} for some constant $0 \leq c < 0.5$, proved by Pitassi and Robere [67]. Lower bounds for share size in linear secret-sharing schemes for almost all access structures were proven in [12, 68, 14], in particular, almost all access structures require shares of length at least $2^{0.5n}$ in any linear secret-sharing scheme realizing them [12].

All the above primitives are closely related, e.g.,

- The 2-server PIR protocol of Dvir and Gopi [36] was transformed to construct the CDS protocols of Liu et al. [60, 61].
- CDS protocols are basically a special case of secret-sharing schemes, i.e., 2-server CDS protocols are equivalent to secret-sharing schemes for forbidden bipartite graph access structures.
- CDS protocols are a central ingredient in constructing the best known secret-sharing schemes for arbitrary access structures, see [59, 8, 10]. Similarly, the best known linear secret-sharing schemes for arbitrary access structures are constructed from linear CDS protocols.

Furthermore, in all the above primitives, the optimal communication complexity/share-size is not known, and there are large gaps between the known lower bounds and upper bounds.

1.2 Our Results

We provide two new techniques addressing the upper bounds for the above primitives.

Abstraction of the DG and GKS PIR protocols and the LVW and BFL CDS protocols. Our first result is an abstraction of the 2-server PIR protocol of Dvir and Gopi [36], the k-server PIR protocol of Ghasemi, Kopparty, and Sudan [45], and the CDS protocols of Liu et al. [60, 61] and its generalization by Beimel, Farràs, and Lasri [15], henceforth the DG, GKS, LVW17, LVW18 and BFL protocols, respectively. Although we do not obtain (asymptotically) better communication, our constructions have the benefit of being both simpler (e.g., we do not use partial derivatives) and more general. The generalization could potentially lead to improvements in the future. Furthermore, the proof of the correctness in this paper is simpler than the proof in GKS and does not require partial derivatives and interpolation theorems.

In more detail, the DG, LVW, and BFL protocols use matching vectors [46] – a combinatorial object that was used to construct explicit Ramsey graphs [46], and later found other applications in computer science such as error-correcting codes [34], PIR protocols [39, 36], and CDS protocols LVW17 ,LVW18. Roughly speaking, an (S,t)-matching vector family over $\mathbb{Z}_m^h$ is a collection of vectors $((\mathbf{u}_i,\mathbf{v}_i))_{i=1}^N$, each in $\mathbb{Z}_m^h$, such that for all $i \neq j$ it holds that $\langle \mathbf{u}_i,\mathbf{v}_i\rangle \in S \quad \text{and} \quad \langle \mathbf{u}_i,\mathbf{v}_j\rangle = t$, where $S \subseteq \mathbb{Z}_m \setminus \{t\}$, and the inner products are done $\bmod m$.

We present one PIR protocol generalizing both the 2-server PIR protocol of DG and the k-server PIR protocol of GKS. This protocol uses share conversion, which is a method, introduced by [20], in which parties, holding shares of a secret s in one secret-sharing scheme, can transform them without interaction to shares of a related secret s' in another secret-sharing scheme. Specifically, we require that shares of the secret 1 are transformed to shares of a non-zero secret s' and shares of a secret $s \in S_{\text{can},m}$ are transformed to shares of the secret $s'=0$, where for $m = p_1 \cdot \ldots \cdot p_r$, a product of r distinct primes, $S_{\text{can},m} = \{a \in \mathbb{Z}_m : \forall 1 \leq i \leq r, a \bmod p_i \in \{0,1\}\} \setminus \{1\}$; for example $S_{\text{can},2} = \{0\}$ and $S_{\text{can},6} = \{0,3,4\}$.

Example 1.1. *We give a simple example of a two-party share conversion from additive sharing over $\mathbb{F}_2$ to additive sharing over $\mathbb{F}_3$, where in an additive sharing of a secret $s \in \mathbb{F}_q$, the shares are $\mathsf{sh}_1 = r$ and $\mathsf{sh}_2 = s + r \bmod q$, for a random $r \in \mathbb{F}_q$. Each server converts its share using the conversion functions $C_1, C_2 : \mathbb{F}_2 \to \mathbb{F}_3$, where $C_1(x) = x$ and $C_2(x) = -x \bmod 3$. We next explain why this is a share conversion, i.e., shares of 0 are converted to shares 0, and shares of 1 are converted to shares of 1 or 2. Starting with a sharing of $s = 0$ results in a sharing of $C_1(r) + C_2(0+r) \equiv r - r \equiv 0 \pmod 3$, and starting with a sharing of $s = 1$ results in a sharing of $C_1(r) + C_2(r+1 \bmod 2) = r - (r+1 \bmod 2)$; this value is 2 if $r = 0$ and is 1 if $r = 1$.*

Generally, our share conversions will convert sharing over one domain to a sharing of a different domain such that a sharing of 0 will be converted to a sharing of 0, and a sharing of 1 will be converted to a sharing of $\neq 0$. Ideally, we would have wanted a share conversion that converts a sharing of 1 to a sharing of 1; however, as [20] showed, no such share conversion exists.

We provide the following general construction, generalizing the results of DG and GKS.

Theorem 1.2 (Informal, simple k-server PIR protocol)**.** *Let $m = p_1 \cdot \ldots \cdot p_r$ be a product of r distinct primes and let $m_2 = p_2 \cdot \ldots \cdot p_r$. Assume there is a family of $S_{\text{can},m}$-matching vectors in $\mathbb{Z}_m^h$ of size N and a k-party S_{can,m_2}-share conversion from $\mathbb{Z}_{m_2}$ to $\mathbb{F}_{p_1^\ell}$ for some $\ell \in \mathbb{N}$. Then, there is a k-server PIR protocol for a database of length N with message size $h \cdot \log(m_2 \cdot p_1^\ell)$.*

The k-server PIR protocol for $k \geq 3$ is constructed by using decoding polynomials of [39] as a share conversion and the matching vectors of [46]. For example, this results in a 3-server PIR with communication complexity $2^{\tilde{O}(\sqrt{\log N})}$. The improvement in the communication complexity of the k-server PIR protocol of [45] and our protocol compared to the protocol of [39] is achieved by using matching vectors for an m that is a product of r primes and using a share conversion (i.e., decoding polynomials) only for m_2 that is a product of $r-1$ primes.

The construction of GKS uses the decoding polynomial of [38]. We generalize their protocol by using the abstraction of share conversion (using the techniques of [20]). The abstraction given by share conversion makes the proof of our protocol modular as it is independent of the proof that a certain set of functions gives a share conversion, and gives rise to a potential improvement by constructing a better share conversion. Alas, currently, for $k \geq 3$ we do not have any suitable share conversions other than the conversion of [38].

More importantly, our protocol and its proof are simpler, and in particular, it is solely based on elementary modular arithmetic and does not use partial derivatives or interpolation theorems.

2-server PIR protocols and 2-server and multi-server CDS protocols. The 2-server PIR protocol implied by our construction is of special interest; this 2-server protocol was published prior to the multi-server PIR protocol of Ghasemi et al. [45]. This protocol uses a more relaxed S-matching vectors where $S_{\text{zero},m} = \mathbb{Z}_m \setminus \mathbb{Z}_m^*$.

Theorem 1.3 (Informal, simple PIR protocols)**.** *Let $m = p_1 \cdot p_2$ a product of two distinct primes. Assume there exists a family of $S_{\text{zero},m}$-matching vectors in $\mathbb{Z}_m^h$ of size N. Then there is a 2-server PIR protocol for a database of length N with message size $O(h \cdot \log m)$.*

This protocol can be transformed to a CDS protocol using the transformation of LVW17.

Theorem 1.4 (Informal, simple CDS protocols)**.** *Let $f : [N]^2 \to \{0,1\}$, let $m = p_1 p_2$ be aproduct of tow distinct primes. Assume there exists a family of $S_{\text{zero},m}$-matching vectors in $\mathbb{Z}_m^h$ of size N. Then there is a 2-server CDS protocol for f with message size $O(h \cdot \log m)$.*

We next compare our 2-server PIR and CDS protocols to the 2-server CDS protocols of DG, LVW and BFL.

- Compared to the DG and LVW constructions, our construction uses a relaxed version of matching vectors with the bigger set; thus, it may be possible to construct $S_{\text{zero},m}$-matching vectors that are shorter than the previous constructions of the more stringent $S_{\text{can},m}-$matching vectors. Any improved construction of $S_{\text{zero},m}$-matching vectors (for some m, such that $\log m = 2^{o(\sqrt{\log N \log\log N})}$) immediately implies an improved construction of 2-server PIR and CDS protocols.

- In this paper, for constructing both PIR and CDS protocols, we use $S_{\text{zero},m}$-matching vectors as oppose to $(S_{\text{one},m}, 0)$-matching vectors, where $S_{\text{one},m} = \{a \in \mathbb{Z}_m : a \equiv 1 \pmod{p_1} \vee a \equiv 1 \pmod{p_2}\}$. These two types of matching vectors are similar, in particular, the length of $S_{\text{zero},m}$-matching vectors and $S_{\text{one},m}$-matching vectors differs by at most 1as we show in Claim 3.10. However, using $S_{\text{zero},m}$-matching vectors results in a simpler presentation of the protocol and simplifies its proof as well.

- The DG and LVW constructions use $m = 6$ and the BFL construction uses $m = p_1p_2$ for any two primes such that $p_2 \mid p_1 - 1$. Our construction of 2-server PIR and CDS protocols, on the other hand, works for any $m = p_1p_2$, where $p_2 \neq p_1$ are prime numbers.

- We use general 2-party share conversion; this helps us generalize the protocol for every pair of primes p_1 and p_2, in particular we can use a share conversion that is much simpler and more intuitive (see Protocol 2.1).

As a result, our protocols abstract and generalize the previous protocols and are simpler. We show a similar result for k-server CDS protocols, i.e., if there is a family of *decomposable* matching vectors in $\mathbb{Z}_m^h$ (for m a product of two distinct primes), then there is a k-server CDS protocol with message length $O(k^2 h \log m)$

(see Definition 7.4 for the definition of decomposable matching vectors).

As explained above, our 2-server protocol use $S_{\text{zero},m}$-matching vectors; these matching vectors might be easier to construct than the $S_{\text{can},m}$-matching vectors, that were studied previously. We note that, by a construction of [39], every family of $S_{\text{zero},m}$-matching vectors of lengthy h can be transformed to a family of $S_{\text{can},m}$-matching vectors of length h^m; thus to construct families of $S_{\text{zero},m}$-matching vectors with shorter vectors one should consider using big values of m (that depend on the number of required vectors N).

Improved results for linear secret sharing. In an independent result, we generalize the reduction from secret-sharing schemes to CDS protocols. This generalization allows us to obtain better *linear* secret-sharing schemes, where the sharing and reconstruction algorithms are both linear mappings. Specifically, while the best previous linear scheme due to [10] has share size $2^{0.7576n}$ for any n-party access structure, we reduce the exponent down to $0.7563n$. Recall that the best-known share size for general secret-sharing schemes for arbitrary access structures is $2^{0.585n}$.

Theorem 1.5 (Informal, linear secret-sharing schemes)**.** *Every n-party access structure can be realized by a linear secret-sharing scheme with share size $2^{0.7563n}$.*

2 Our Techniques

2.1 A Simple 2-Server PIR Protocol

DG presented a 2-server PIR protocol with communication $2^{O(\sqrt{\log N \log\log N})}$; their protocol uses matching vectors over $\mathbb{Z}_6$. BFL implicitly generalized this protocol by using matching vectors over $\mathbb{Z}_m$, where m is a product of two primes p_2, p_1 such that $p_2 | p_1 - 1$. We simplify and generalize these protocols using matching vectors over $\mathbb{Z}_m$ for every m that is a product of two distinct primes. In this section, we will refer to the two servers as Alice and Bob.

Warm-up. We start with describing an inefficient two-server PIR protocol that will provide the motivation for our protocol. Let $(\mathbf{u}_i)_{i=1}^N \in \mathbb{F}_p^h$ be a set of orthonormal vectors over some finite field $\mathbb{F}_p$ and $h \in \mathbb{N}$; that is, $\langle \mathbf{u}_i, \mathbf{u}_i \rangle \equiv 1 \pmod p$ and $\langle \mathbf{u}_i, \mathbf{u}_j \rangle \equiv 0 \pmod p$ for every $i \neq j$. The user with index i chooses a random $\mathbf{r} \in \mathbb{F}_p^h$ and sends $\mathbf{q}_A = \mathbf{r}$ to Alice and $\mathbf{q}_B = \mathbf{r} + \mathbf{u}_i \bmod p$ to Bob. A server with query $\mathbf{q}$ and database D computes the answer $\sum_{j=1}^N \langle \mathbf{q}, \mathbf{u}_j \rangle D_j$ and sends the answer

to the user where the inner product is computed modulo p; the user subtracts the answer of Alice from the answer of Bob and obtains D_i as we next explain.

$$\sum_{j=1}^{N}\langle \mathbf{r}+\mathbf{u}_i,\mathbf{u}_j\rangle D_j - \sum_{j=1}^{N}\langle \mathbf{r},\mathbf{u}_j\rangle D_j \equiv \sum_{j=1}^{N}(\langle \mathbf{r}+\mathbf{u}_i,\mathbf{u}_j\rangle - \langle \mathbf{r},\mathbf{u}_j\rangle)D_j$$
$$\equiv \sum_{j=1}^{N}\langle \mathbf{u}_i,\mathbf{u}_j\rangle D_j \equiv D_i \pmod{p},$$

where the last equality follows from the orthonormality of the vectors. The obvious problem with this construction is that the length of N orthonormal vectors is at least N and the protocol is not efficient. Following [39], we will work with vectors over $\mathbb{Z}_m$ for a composite m; specifically, m is a product of two distinct primes p_2, p_1. As every set of orthonormal vectors over $\mathbb{Z}_m$ is also orthonormal over $\mathbb{F}_{p_2}$, we need to relax the orthonormality requirements.

Matching vectors. We use an $S_{\text{zero},m}$-matching vector family $((\mathbf{u}_i,\mathbf{v}_i))_{i=1}^{N}$ over $\mathbb{Z}_m^h$, where $m = p_1p_2$ for primes $p_2 < p_1$, i.e., for all $i \neq j$ it holds that

$$\langle \mathbf{u}_i,\mathbf{v}_i\rangle \bmod m = 1 \quad \text{and} \quad \langle \mathbf{u}_i,\mathbf{v}_j\rangle \bmod m \in \mathbb{Z}_m \setminus \mathbb{Z}_m^*.$$

Observe that this is equivalent to

$$\langle \mathbf{u}_i,\mathbf{v}_i\rangle \bmod p_2 = 1 \quad \text{and} \quad \langle \mathbf{u}_i,\mathbf{v}_i\rangle \bmod p_1 = 1$$

and

$$\langle \mathbf{u}_i,\mathbf{v}_j\rangle \bmod p_2 = 0 \quad \text{or} \quad \langle \mathbf{u}_i,\mathbf{v}_j\rangle \bmod p_1 = 0.$$

Such a matching vector family with $h = 2^{O(\sqrt{\log N \log\log N})}$ can be constructed from the matching vector families constructed in [46, 58]. Note that this definition is a modification of the definition of matching vectors in previous papers, where it is required that $\langle \mathbf{u}_i,\mathbf{v}_i\rangle \bmod m = 0$ and the requirement for $i \neq j$ is also different. This modification allows us to simplify the protocol.

In the following, for a prime p let $\langle \mathbf{u},\mathbf{v}\rangle_p = \sum_{\ell=1}^{h}\mathbf{u}[\ell]\mathbf{v}[\ell] \bmod p$.

Protocol 2.1 (A 2-server PIR protocol)**.**
Public parameters: *$S_{\text{zero},m}$-matching vectors $((\mathbf{u}_i,\mathbf{v}_i))_{i=1}^{N}$ over $\mathbb{Z}_m^h$, where $m = p_1p_2$ for two primes $p_2 < p_1$.*
Alice's and Bob's input: $D \in \{0,1\}^N$.
The user's input: $i \in [N]$.

- *The user chooses $\mathbf{r} \leftarrow \mathbb{Z}_{p_2}^h$ with uniform distribution and sends $\mathbf{q}_A = \mathbf{r}$ to Alice and $\mathbf{q}_B = \mathbf{u}_i + \mathbf{r} \bmod p_2$ to Bob.*
- *Alice and Bob compute $\mathbf{a}_A = \sum_{j=1}^{N}(\langle \mathbf{q}_A,\mathbf{v}_j\rangle_{p_2}\cdot D_j)\mathbf{v}_j \bmod p_1$ and $\mathbf{a}_B = \sum_{j=1}^{N}(\langle \mathbf{q}_B,\mathbf{v}_j\rangle_{p_2}D_j)\cdot \mathbf{v}_j \bmod p_1$ respectively and send the answers to the user (each answer is a vector in $\mathbb{Z}_{p_1}^h$).*
- *The user outputs 1 if*
$$\langle \mathbf{u}_i, \mathbf{a}_B - \mathbf{a}_A\rangle \not\equiv 0 \pmod{p_1}, \tag{1}$$
and 0 otherwise.

For comparison, we describe the simplest version of the PIR protocol of DG in Appendix B. We next prove that Protocol 2.1 is a PIR protocol. To emphasize that the proof is simple, we provide the entire proof in this section. Each query to a server is uniformly distributed in $\mathbb{Z}_{p_2}^h$ regardless of i and the privacy clearly holds. We next show that correctness holds; this should be carefully analyzed as we use inner product over $\mathbb{Z}_{p_2}$ and $\mathbb{Z}_{p_1}$. The user computes

$$\begin{aligned}\langle \mathbf{u}_i, \mathbf{a}_B - \mathbf{a}_A \rangle_{p_1} &\equiv \left\langle \mathbf{u}_i,\ \sum_{j=1}^{N} \langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \cdot D_j \mathbf{v}_j - \sum_{j=1}^{N} \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \cdot D_j \mathbf{v}_j \right\rangle_{p_1} \\ &\equiv \sum_{j=1}^{N} \left(\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_j \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \right) \cdot \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot D_j \pmod{p_1}. \end{aligned} \tag{2}$$

We claim that this sum is equal to $\alpha \cdot D_i$ for some $\alpha \neq 0$, that is, in the sum in (2) each term for $j \neq i$ is zero and the term for $j = i$ is non-zero if and only if $D_i = 1$.

We first claim that

$$\left(\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_j \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \right) \cdot \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot D_j \equiv 0 \pmod{p_1}, \text{ for } i \neq j. \tag{3}$$

Clearly, this is true if $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} = 0$. Otherwise, $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_2} = 0$; thus,

$$\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \equiv \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_2} + \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \equiv \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \pmod{p_2}$$

and (3) holds.

Now, we consider the case $i = j$.

$$\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \equiv \left(\langle \mathbf{u}_i, \mathbf{v}_i \rangle_{p_2} + \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \right) \bmod p_2 \equiv 1 + \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \pmod{p_2}.$$

If $\langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} < p_2 - 1$ then $\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_i \rangle_{p_2} = 1 + \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2}$ over $\mathbb{F}_{p_1}$ and $\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_i \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \equiv 1 \pmod{p_1}$. Otherwise, $\langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} = p_2 - 1$, and over $\mathbb{F}_{p_1}$

$$\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_i \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \equiv 0 - \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \equiv 1 - p_2 \not\equiv 0 \pmod{p_1}, \tag{4}$$

since $0 < p_2 - 1 < p_1$.

Therefore, by (3) and (4)

$$\begin{aligned}\langle \mathbf{u}_i, \mathbf{a}_B - \mathbf{a}_A \rangle_{p_1} &\equiv \sum_{j=1}^{N} \left(\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_j \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_j \rangle_{p_2} \right) \cdot \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot D_j \\ &\equiv \left(\langle \mathbf{u}_i + \mathbf{r}, \mathbf{v}_i \rangle_{p_2} - \langle \mathbf{r}, \mathbf{v}_i \rangle_{p_2} \right) \cdot \langle \mathbf{u}_i, \mathbf{v}_i \rangle_{p_1} \cdot D_i \equiv \alpha \cdot D_i \pmod{p_1}. \end{aligned}$$

for some $\alpha \neq 0$. Thus, the reconstruction of D_i is correct.

Our 2-server CDS protocol is obtained from the 2-server PIR protocol using the transformation of LVW17. See details in Section 7.1.

In Section 5, we show a more general PIR protocol that does not assume $p_2 < p_1$. Note that if $p_1 | p_2 - 1$, then the inequality in (4) does not necessarily hold. To resolve this, we abstract the properties of the operator $\langle \mathbf{q}, \mathbf{v} \rangle_{p_2}$ that we use in Protocol 2.1; instead, the servers apply a "so-called" *share conversion* $C(\langle \mathbf{q}, \mathbf{v} \rangle)$. There are a few reasons to describe the protocols using a general share conversion:

- This unifies the description of the 2-server and multi-server protocols.
- The proofs are somewhat cleaner using this notation.
- This provides a more general protocol; in particular, it captures a simplified variation of the BFL protocol (which generalizes the DG protocol and the LVW protocol).
- In some scenarios, we will want the servers in Protocol 2.1 to send answers over a specific field, e.g., given matching vectors over $\mathbb{Z}_{21}$, we will want to work in $\mathbb{F}_7$. We construct share conversions that enable such a property.

Remark 2.2. *In [61], they use the notion of PIR encoding to describe their PIR protocol, which looks similar to our protocol. However, the PIR encoding they construct for 2-servers is actually the same as in DG and LVW and their construction and proofs are more complicated than our construction and proofs.*

2.2 A Simple 3-Server PIR Protocol

In this paper, we present a somewhat simplified version of the recent k-server PIR protocol of GKS; our proof of the protocol is simpler and does not use partial derivations. To simplify the notations, we will describe in this section the 3-server PIR protocol with communication complexity $2^{\tilde{O}(\sqrt[3]{\log n})}$. Furthermore, we will use specific parameters (in Section 6, these choices are generalized). The protocol is a generalization of the above 2-server PIR protocol, using decoding polynomials – a tool from the 3-server PIR protocol of Efremenko [39]. As in the above 2-server PIR protocol, the 3-server PIR protocol uses a family of matching vectors; the matching vectors are over $\mathbb{Z}_m$, where $m = p_1 \cdot p_2 \cdot p_3$ is a product of 3 distinct primes. Furthermore, the queries of the user on index i are a secret sharing of $\mathbf{u}_i$ and the answer of the t-th server is of the form $\sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) \cdot C_j(\langle \mathbf{q}_t, \mathbf{v}_j \rangle)$ for some function C_j, where $\mathbf{q}_t$ is the query sent to the t-th server.

We next elaborate. Let $m = p_1 \cdot p_2 \cdot p_3$, $m_2 = p_2 \cdot p_3$ and assume that $m_2 + 1 = p_1^\ell$ for some ℓ (e.g., $p_1 = 2$, $m_2 = 7 \cdot 73 = 2^9 - 1$). The protocol uses an $S_{\text{can},m}$-matching vector family as defined in Section 1.2; it can be constructed with vectors of length $h = 2^{\tilde{O}(\sqrt[3]{\log N})}$ [46, 58].

The last ingredient we need is decoding polynomials, introduced in [39]. Consider the field $\mathbb{F}_{p_1^\ell}$; its multiplicative group has a generator γ of order $p_1^\ell - 1 = m_2$. A decoding polynomial $P(X)$ over $\mathbb{F}_{p_1^\ell}$ is a polynomial such that $P(\gamma) \neq 0$ and $P(\gamma^s) = 0$ for every $s \in S_{\text{can},m_2}$. For example, if $m_2 = 7 \cdot 73 = 511$, then $S_{\text{can},m_2} = \{0, 147, 365\}$, thus we can take the polynomial $P(x) = (x - \gamma^0)(x - \gamma^{147})(x - \gamma^{365})$; this polynomial has 4 non-zero coefficients. For the 3-server PIR protocol, we will need a decoding polynomial $P(X) = b_3 x^{\alpha_3} + b_2 x^{\alpha_2} + b_0 x^{\alpha_0}$ with 3 non-zero coefficients; such polynomial is implied by [39].[1] We describe the 3-server PIR protocol in Protocol 2.3.

Protocol 2.3 (A 3-server PIR protocol)**.**
Public parameters: *An S_{can}-matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ over $\mathbb{Z}_m^h$ for $m = p_1 \cdot p_2 \cdot p_3$, where $m_2 = p_2 \cdot p_3 = p_1^\ell - 1$ for some $\ell \geq 1$, and a generator $\gamma \in \mathbb{F}_{p_1^\ell}^*$ (of order m_2).*
A decoding polynomial $P(x) = \sum_{t=1}^{3} b_t x^{\alpha_t}$ over $\mathbb{F}_{p_1^\ell}$
Servers' Input: $D \in \{0,1\}^N$.

[1] Take the polynomial $P(X) = \gamma^{269} x^{499} + \gamma^{488} x^{446} + \gamma^{342}$ over $\mathbb{F}_{2^{512}} = \mathbb{F}[\gamma]/(\gamma^9 + \gamma^4 + 1)$.

The user's input: $i \in [N]$.
The user's randomness: $\mathbf{r} \in \mathbb{F}_{m_2}^h$

- *The user sends* $\mathbf{q}_t \leftarrow \mathbf{r} + \alpha_t \cdot \mathbf{u}_i \bmod m_2$ *to the* t*-th server for* $1 \le t \le 3$ *(where* $\alpha_1, \alpha_2, \alpha_3$ *are the exponents in the decoding polynomial).*
- *The* t*-th server sends to the user an answer* $\mathbf{a}_t \in \mathbb{F}_{p_1^\ell}^h$*, where*

$$\mathbf{a}_t \leftarrow \sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) \cdot b_t \gamma^{\langle \mathbf{q}_t, \mathbf{v}_j \rangle}.$$

- *The user computes* $\sum_{t=1}^{k} \langle (\mathbf{u}_i \bmod p_1), \mathbf{a}_t \rangle$ *over* $\mathbb{F}_{p_1^\ell}$ *and returns* 0 *iff the expression equals* 0.

We next prove that Protocol 2.3 is a PIR protocol. Each query to a server is uniformly distributed in $\mathbb{Z}_{m_2}^h$ regardless of i and the privacy clearly holds. We next show that correctness holds; this should be carefully analyzed as we use inner product over $\mathbb{Z}_{m_2}$ and $\mathbb{F}_{p_1^\ell}$. Observe that $\gamma^{(a+b) \bmod m_2} = \gamma^{a \bmod m_2} \cdot \gamma^{b \bmod m_2}$ since the order of γ is m_2. The user computes the following expression in $\mathbb{F}_{p_1^\ell}$.

$$\begin{aligned}
\sum_{t=1}^{3} \langle (\mathbf{u}_i \bmod p_1), \mathbf{a}_t \rangle &= \langle (\mathbf{u}_i \bmod p_1), \sum_{t=1}^{3} \sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) \cdot b_t \gamma^{\langle \mathbf{r} + \alpha_t \cdot \mathbf{u}_i, \mathbf{v}_j \rangle} \rangle \\
&= \sum_{t=1}^{3} \sum_{j=1}^{N} D_j \langle (\mathbf{u}_i \bmod p_1), (\mathbf{v}_j \bmod p_1) \rangle \cdot b_t \gamma^{\langle \mathbf{r} + \alpha_t \cdot \mathbf{u}_i, \mathbf{v}_j \rangle} \\
&= \sum_{j=1}^{N} D_j \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot \gamma^{\langle \mathbf{r}, \mathbf{v}_j \rangle} \sum_{t=1}^{3} b_t \gamma^{\langle \alpha_t \cdot \mathbf{u}_i, \mathbf{v}_j \rangle} \\
&= \sum_{j=1}^{N} D_j \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot \gamma^{\langle \mathbf{r}, \mathbf{v}_j \rangle} \sum_{t=1}^{3} b_t \left(\gamma^{\langle \mathbf{u}_i, \mathbf{v}_j \rangle} \right)^{\alpha_t} \\
&= \sum_{j=1}^{N} D_j \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot \gamma^{\langle \mathbf{r}, \mathbf{v}_j \rangle} P\left(\gamma^{\langle \mathbf{u}_i, \mathbf{v}_j \rangle} \right).
\end{aligned}$$

We next analyze the terms in the above expression.

- For $i = j$, by the properties of matching vectors and the decoding polynomial, $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m = 1$, therefore, $\langle \mathbf{u}_i, \mathbf{v}_i \rangle_{m_2} = 1$ and $\langle \mathbf{u}_i, \mathbf{v}_i \rangle_{p_1} = 1$, and $P(\gamma^{\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{m_2}}) = P(\gamma) \neq 0$. Thus, $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} \cdot \gamma^{\langle \mathbf{r}, \mathbf{v}_j \rangle} P\left(\gamma^{\langle \alpha_t \mathbf{u}_i, \mathbf{v}_j \rangle_{m_2}} \right) \neq 0$ over $\mathbb{F}_{p_1^\ell}$.
- If $i \neq j$, then $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m \neq 1$ and $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_d} \in \{0, 1\}$ for $1 \le d \le 3$. If $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} = 0$ then $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1} = 0$ over $\mathbb{F}_{p_1^\ell}$. Otherwise, $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{m_2} \neq 1$ and $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_d} \in \{0, 1\}$ for $2 \le d \le 3$, i.e., $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{m_2} \in S_{\text{can},m_2}$; since P is decoding polynomial, $P(\gamma^{\langle \mathbf{u}_i, \mathbf{v}_j \rangle_{m_2}}) = 0$ over $\mathbb{F}_{p_1^\ell}$.

We conclude that $\sum_{t=1}^{3} \langle (\mathbf{u}_i \bmod p_1), \mathbf{a}_t \rangle$ is equal to some non-zero term times D_i and the reconstruction is correct.

We can instantiate Protocol 2.3 with $m = 2 \cdot 7 \cdot 73 = 511$, the decoding polynomial implied by [39], and the $S_{\text{can},m}$-matching vector family implied by [46, 58]; the communication complexity of the resulting polynomial is $h = 2^{\tilde{O}(\sqrt[3]{\log N})}$.

2.3 An Improved Linear Secret Sharing

Our construction of an improved linear secret-sharing scheme follows the ideas of the recent constructions of linear secret-sharing schemes [59, 7, 8, 10], specifically, we use the blueprint of Applebaum and Nir [10]. This construction uses robust CDS protocols to realize downslice access structures (see definition below). It then uses covering and bootstrapping techniques to construct better linear secret-sharing schemes for downslice access structures. As every n-party access structure can be written as an intersection of n downslice access structures, this implies a linear secret-sharing scheme for every access structure.

Let $0 < b < n$. A b-downslice access structure Γ is an access structure where all maximal unauthorized sets are of size b, that is there are some sets $A_1, \ldots, A_\ell$ of size b such that $A \notin \Gamma$ if and only if there is some $1 \leq i \leq \ell$ such that $A \subseteq A_i$, i.e., $\Gamma = \{B : \forall 1 \leq i \leq \ell, B \not\subseteq A_i\}$.

Example 2.4. *Consider the access structure with n parties $\{p_1, \ldots, p_n\}$, where a set is A authorized if and only if it contains at least one party from $\left\{p_1, \ldots, p_{n/2}\right\}$ and at least one party from $\left\{p_{n/2+1}, \ldots, p_n\right\}$; this is an $n/2$-downslice in which the maximal unauthorized sets are $\left\{p_1, \ldots, p_{n/2}\right\}$ and $\left\{p_{n/2+1}, \ldots, p_n\right\}$ and the minimal authorized set are all sets B of size 2 s.t. $B = \{p_{i_1}, p_{i_2}\}$ where $1 \leq i_1 \leq \frac{n}{2}$ and $\frac{n}{2} + 1 \leq i_2 \leq n$.*

Every access structure Γ can be written as $\cap_{1 \leq b \leq n} \Gamma_b$, where Γ_b is the b-downslice access structure whose maximal unauthorized sets are the maximal unauthorized sets of Γ of size b.

Our construction is implied by a construction of linear secret-sharing schemes for n-party b-downslice access structures for $b > n/2$ that is better than previous constructions. We next explain the idea of our improvement. We achieve this goal by a reduction to CDS protocols; specifically to robust CDS protocols (as introduced in [8]). Previous constructions of linear secret-sharing schemes using robust CDS protocols use either reductions to 2-server protocols [8] or to $\sqrt{n}$-server protocols [8, 10]; our improvement is via reduction to 2-server CDS protocols.

We first recall the notion of robust 2-server CDS protocols; again we reffer to the servers as Alice and Bob. The security requirement of a CDS protocol for a function f ensures that if Alice sends its message for an input x and Bob sends its message for an input y such that $f(x, y) = 0$, then the referee does not learn any information on the secret from these messages. Assume that Alice also sends a message for an input $x' \neq x$ with the same shared randomness, such that also $f(x', y) = 0$; the CDS protocol does not guarantee that the referee does not learn any information on the secret from the 3 messages. In a (t_1, t_2)-robust CDS protocol, the security is guaranteed even when using the same shared randomness, Alice sends messages for a set X_1 with at most t_1 inputs and Bob sends messages for a set X_2 with at most t_2 inputs such that $f(x, y) = 0$ for every $x \in X_1, y \in X_2$; i.e., the referee does not learn information on the secret from these $t_1 + t_2$ messages.

Warm-up. We show in Scheme 2.5 that, given an access structure Γ, an efficient (N, N)-robust CDS protocol for a function f_Γ defined below implies a good secret-sharing scheme for Γ. Let $\{p_1, \ldots, p_n\}$ be the parties of Γ, $P_1 = \left\{p_1, \ldots, p_{n/2}\right\}$ and $P_2 = \left\{p_{n/2+1}, \ldots, p_n\right\}$. Define $f_\Gamma : 2^{P_1} \times 2^{P_2} \to \{0, 1\}$, where for $A_1 \subseteq P_1, A_2 \subseteq P_2$

$$f_\Gamma(A_1, A_2) = 1 \quad \iff \quad A_1 \cup A_2 \in \Gamma;$$

the size of the domain of each server is $N = 2^{n/2}$.

Scheme 2.5 (A simple secret-sharing scheme from a fully robust CDS protocol)**.**
Public parameters: *An (N, N)-robust CDS protocol $\mathcal{P}$ for f_Γ.*
The secret: $s \in \{0,1\}$.

- *Choose a random string r for the CDS protocol.*
- *For every $A_1 \in 2^{P_1}$, compute m_{A_1} – the message in $\mathcal{P}$ of Alice with input A_1 and random string r, share m_{A_1} in a $|A_1|$-out-of-$|A_1|$ secret-sharing scheme, and give each share to a party in A_1.*
- *For every $A_2 \in 2^{P_2}$, compute m_{A_2} – the message in $\mathcal{P}$ of Bob with input A_2 and random string r, share m_{A_2} in a $|A_2|$-out-of-$|A_2|$ secret-sharing scheme, and give each share to a party in A_2.*

For a set A, let $A_1 = A \cap P_1, A_2 = A \cap P_2$. If $A \in \Gamma$, $f_\Gamma(A_1, A_2) = 1$, thus the secret can be reconstructed from m_{A_1}, m_{A_2}. As the parties in A can reconstruct these messages, the correctness follows. For the security, consider a set $A \notin \Gamma$. The parties in A can only reconstruct the messages M_{B_1}, M_{B_2} for every $B_1 \subseteq A_1$ and $B_2 \subseteq A_2$. As Γ is monotone, $B_1 \cup B_2 \notin \Gamma$ for every such B_1, B_2, thus, $f_\Gamma(B_1, B_2) = 0$. By the (N, N)-robustness of the CDS protocol, the parties in A do not know any information on the secret.

The problem in the above construction is the share size. The share of each party $p_i \in P_1$ is a share of the message of each subset $A_1 \in P_1$ such that $p_i \in A_1$, there are $2^{n/2-1}$ such sets. The best known (N, N)-robust CDS protocol for a function $f : [N] \times [N] \to \{0,1\}$ has message size $O(N/\log N)$ [40]; in our case $N = 2^{n/2}$. Thus, the total share of each party is $O(2^{n/2-1} \cdot N/\log N) = 2^{n-o(n)}$. This share size is too big as our goal is share size 2^{cn} for some constant $c < 1$.

Linear secret-sharing schemes for somewhat regular access structures. As in previous papers, to improve the share size we reduce the question of constructing a linear secret-sharing scheme for an arbitrary access structure Γ to the question of constructing a (t, N_2)-robust CDS protocol for functions $f : [N_1] \times [N_2] \to \{0,1\}$ for some $t = o(N_1)$; the message size of Alice in the best known linear CDS protocol for such f is $\tilde{O}(N_1/t)$ and the message size of Bob is $\tilde{O}(t)$ (up to polylogarithmic factors).[2] Notice that the message sizes of Alice and Bob are not the same when $t \not\approx \sqrt{N_2}$.

We use a different approach then previous papers, which results in a better share size. Rather than consider $|P_1| = |P_2| = n/2$ as in the warm-up and [8], we take P_1, P_2 such that $|P_1| = \mu n$ and $|P_2| = (1-\mu)n$ for some parameter $0 < \mu < 1$ and define $f_\Gamma : \{0,1\}^{\mu n} \times \{0,1\}^{(1-\mu)n} \to \{0,1\}$ with respect to this partition. This is our main contribution for constructing linear secret-sharing schemes; this unbalanced partition is the tool that enables us to improve the share size. Following [8], we show in Section 8.3 that to realize b-downslice access structures it suffices to consider somewhat regular access structures, which are access structures in which every maximal unauthorized set A is "well partitioned", i.e., $|A \cap P_1| \leq \mu b$.[3] We execute Scheme 2.5 with the modified f_Γ using a (t, N)-robust CDS protocol for $t = 2^{\mu b}$.

[2] To get this message size, we need a more refined notion of robustness called $(\mathcal{Z}, N)$-robustness. See Definition 8.4 for a definition of this notion and Section 8.3 for the details why we can use it.

[3] Actually, the bounds on the sizes that we get are bigger by a factor of $n^{0.2}$; we ignore this factor in this section. Furthermore, following [8], we will also require that minimal authorized sets are well partitioned; we do not use this property in this paper.

The correctness follows as in the warm-up. For the security, consider an unauthorized set A; recall that $|A \cap P_1| \leq \mu b$, thus, the parties in A learn at most $t = 2^{\mu b}$ messages in the CDS protocol $\mathcal{P}$ (one message for each $A' \subseteq A$). The share of each $p_i \in P_1$ contains $2^{\mu n}$ shares of messages of Alice, each message of size $\tilde{O}(N_1/t) = 2^{\mu n}/2^{\mu b}$; i.e., the share size is $\tilde{O}(2^{\mu(2n-b)})$. Similarly, the share of each $p_i \in P_2$ contains $2^{(1-\mu)n}$ shares of messages of Bob, each message of size $\tilde{O}(t) = 2^{\mu b}$; i.e., the share size is $\tilde{O}(2^{n-\mu(n-b)})$. To minimize the maximum share size, we take μ such that $2^{\mu(2n-b)} = 2^{n-\mu(n-b)}$, i.e., $\mu = \frac{n}{3n-2b}$; this results in share size $\tilde{O}(2^{n(2n-b)/(3n-2b)})$. For $b > n/2$ our scheme improves on the scheme of [10].

Example 2.6. *Consider $b = 0.5412n$ (this is the value that we will use in our construction). In this case $\mu = \frac{1}{3-2\cdot 0.5412} = 0.5214$ and the share size is $\tilde{O}(2^{0.7607n})$. In this case more parties are in P_1, the set of parties attached to Alice.*

Consider $b = 0.4n$. In this case $\mu = 0.4545 < 0.5$ and the share size is $\tilde{O}(2^{0.7272n})$. In this case less parties are in P_1, the set of parties attached to Alice.

Linear secret-sharing schemes for arbitrary access structures. We use the fact that any access structure Γ is the intersection of n downslices to realize an arbitrary access structure; however as the share size for b-downslices approaches 2^n as b approaches n, we cannot use the above schemes for downslices as is. We use two techniques from previous papers to reduce the share size. The first technique is the covering technique [7, 10], which shows that for every $a < b$, every b-downslice access structure with n parties can be realized as the intersection of roughly $\binom{n}{n-b}/\binom{n-a}{n-b}$ access structures, each one of them is a $b-a$-downslice access structure with $n-a$ parties. The second technique is the boosting technique from [10] showing that for linear secret-sharing schemes each b-downslice with n parties can be realized by realizing b-downslice access structures with fewer parties (and some multislice access structures). This results in a recursive construction that uses a scheme for b-downslice access structures and results in a scheme for b-downslice access structures with better share size.

Following [10], our scheme for an arbitrary access structure will have the following structure:

- Write Γ as $\cap_{b=1}^{n}\Gamma_b$, where Γ_b is a b-downslice access structure.
- For every $b < 0.5n$, use the linear scheme of [10] to realize the b-downslice Γ_b.
- Use the boosting technique to realize $0.5n$ downslice access structures.
- For every $0.5n \leq b \leq 0.554n$, use the covering technique to realize the b-downslice Γ_b using the scheme for $0.5n$ downslice access structures.
- Use the covering technique to realize 0.554-downslice access structures using our scheme for $0.5214n$-downslice access structures.
- Use the resulting scheme for $0.554n$-downslice access structures and the boosting technique to get a better scheme for $0.554n$-downslice access structures.
- For every $0.554n \leq b \leq n$, use the covering technique to realize the b-downslice Γ_b using the scheme for $0.554n$ downslice access structures.

The scheme in [10] had different constant instead of 0.554 and $0.5214n$. Finding the exact constants that optimize our scheme was done using a computer programs; this program also found the parameters in the boosting.

Remark 2.7. *In [10] and in this paper, we partition the possible values of* $1 \leq b \leq n$ *to 3 intervals. It might seem that taking more intervals can reduce the share size; however, this is not true. For example, the construction for* $0.592n$*-downslice access structures (via covering) results in the highest share size. For* $b = 0.592n$*, using the covering technique with* $0.554n$ *downslice access structures is optimal, i.e., using the covering technique with a different* a*-downslice will result is bigger share size. Although for other values* $0.554n < b < n$ *covering to a different downslice may improve the share size for* b*-downslice access structure, it will not improve the maximum share size for* Γ*.*

3 Preliminaries

3.1 Notations

We start with some notation used in this paper. For strings $x, y \in \{0,1\}^n$ we write $x \leq y$ if $x_i \leq y_i$ for every $i \in [n]$. We denote the concatenated string as $x||y$. We let $\mathrm{wt}(x)$ denote the Hamming weight of x. We denote the *binary entropy function* by $h : [0,1] \to [0,1]$ and it is defined as $h(x) = -x \log x - (1-x)\log(1-x)$ for all $x \in (0,1)$, and $h(0) = h(1) = 0$. For every $p \in \mathbb{Z}$, we denote $\langle \cdot, \cdot \rangle_p$ the inner product modulo p, i.e., for every two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_p^h$, $\langle \mathbf{x}, \mathbf{y} \rangle_p = \sum_{\ell=1}^{h} \mathbf{x}[\ell] \cdot \mathbf{y}[\ell] \bmod p$. For every $m \in \mathbb{N}$, we define the equivalence relation over $\mathbb{Z} \times \mathbb{Z}$, where for every $a, b \in \mathbb{Z}$, we say that $a \equiv b \pmod m$ if $a \bmod m = b \bmod m$. In contrast, $a \bmod m$ denotes the operator that outputs the remainder of the division of a by m.

3.2 Secret-Sharing Schemes

We start by defining (perfect) secret-sharing schemes.

Definition 3.1 (Access structures)**.** *Let* $P = \{p_1, \ldots, p_n\}$ *be a set of parties. A collection* $\Gamma \subseteq 2^{\{p_1,\ldots,p_n\}}$ *is monotone if* $B \in \Gamma$ *and* $B \subseteq C$ *imply that* $C \in \Gamma$*. An access structure* $\Gamma \subseteq 2^{\{p_1,\ldots,p_n\}}$ *is a monotone collection of non-empty sets. Sets in* Γ *are called authorized, and sets not in* Γ *are called unauthorized. We will also represent an* n*-party access structure by a function* $f : \{0,1\}^n \to \{0,1\}$*, where an input (i.e., a string)* $\sigma = \sigma_1, \sigma_2, \ldots, \sigma_n \in \{0,1\}^n$ *represents the set* $A_\sigma = \{p_i : i \in [n], \sigma_i = 1\}$*, and* $f(\sigma) = 1$ *if and only if* $A \in \Gamma$*. We will also call* f *an access structure.*

A secret-sharing scheme defines a way to distribute shares to parties. Such a scheme is said to realize an access structure Γ if the shares held by any authorized set of parties (i.e., a set in the access structure) can be used to reconstruct the secret, and the shares held by any unauthorized set of parties reveal nothing about the secret. The formal definition is given as follows.

Definition 3.2 (Secret-sharing schemes)**.** *A secret-sharing scheme* Π *over a set of parties* $P = \{p_1, \ldots, p_n\}$ *with domain of secrets* S *and domain of random strings* R *is a mapping from* $S \times R$ *to a set of* n*-tuples* $S_1 \times S_2 \times \cdots \times S_n$ *(the set* S_j *is called the domain of shares of* p_j*). A dealer distributes a secret* $s \in S$ *according to* Π *by first sampling a random string* $r \in R$ *with uniform distribution, computing a vector of shares* $\Pi(s; r) = (\mathsf{sh}_1, \ldots, \mathsf{sh}_n)$*, and privately communicating each share* sh_j *to party* p_j*. For a set* $A \subseteq \{p_1, \ldots, p_n\}$*, we denote* $\Pi_A(s; r)$ *as the restriction of* $\Pi(s; r)$ *to its* A*-entries (i.e., the shares of the parties in* A*).*

A secret-sharing scheme Π *with domain of secrets* S *realizes an access structure* Γ *if the following two requirements hold:*

Correctness. *The secret s can be reconstructed by any authorized set of parties. That is, for any authorized set $B = \left\{p_{i_1}, \ldots, p_{i_{|B|}}\right\} \in \Gamma$, there exists a reconstruction function $\mathrm{Recon}_B : S_{i_1} \times \cdots \times S_{i_{|B|}} \to S$ such that for every secret $s \in S$ and every random string $r \in R$, it holds that $\mathrm{Recon}_B\left(\Pi_B(s;r)\right) = s$.*

Security. *Every unauthorized set cannot learn anything about the secret from its shares. Formally, for any set $T \notin \Gamma$, every two secrets $s_1, s_2 \in S$, and for a uniformly distributed $r \leftarrow R$, the distributions $\left(\Pi_T(s_1;r) = \langle \mathsf{sh}_j \rangle_{p_j \in T}\right)$ and $\left(\Pi_T(s_2;r) = \langle \mathsf{sh}_j \rangle_{p_j \in T}\right)$ are equivalent.*

The size of the share of party p_j is defined as $\log |S_j|$ and the size of the shares of Π is defined as $\max_{1 \le j \le n} \log |S_j|$. The total share size of Π is defined as $\sum_{j=1}^{n} \log |S_j|$.

Definition 3.3. *A scheme is called* linear over a finite field $\mathbb{F}$ *if $S = \mathbb{F}$, $R = \mathbb{F}^\ell$ for some integer $\ell \ge 1$, the sets $S_1, \ldots, S_n$ are vector fields over $\mathbb{F}$, and the mapping Π is a linear mapping of the secret and the coordinates of the random string $r = (r_1, \ldots, r_\ell)$. Finally, for an access structure Γ, let $\mathsf{LSS}(\Gamma)$ denote the minimum total share size, where the minimum is taken over all linear secret-sharing schemes realizing Γ (over all finite fields).*

3.3 Conditional Disclosure of Secrets

Informally, in a CDS protocol there are k servers $S_1, \ldots, S_k$, each holding a private input x_i, the secret s, and a common random string r, and there is a referee holding $x_1, \ldots, x_k$. Each server S_i sends the message $m_i = \textsc{Enc}(x_i, s; r)$ to the referee, and the referee can reconstruct s if and only if $f(x_1, \ldots, x_n) = 1$.

Definition 3.4 (Conditional disclosure of secrets (CDS) protocols)**.** *Let $f : X_1 \times \cdots \times X_k \to \{0,1\}$ be a k-input function. A k-*server CDS protocol *$\mathcal{P}$ for f, with domain of secrets S, domain of common random strings R, and finite message domains $M_1, \ldots, M_k$, consists of k encoding functions $\textsc{Enc}_1, \ldots, \textsc{Enc}_k$, where $\textsc{Enc}_i : X_i \times S \times R \to M_i$ for every $i \in [k]$. For an input $x = (x_1, \ldots, x_k) \in X_1 \times \cdots \times X_k$, secret $s \in S$, and randomness $r \in R$, we let $\textsc{Enc}(x, s; r) = (\textsc{Enc}_1(x_1, s; r), \ldots, \textsc{Enc}_k(x_k, s; r))$. We say that $\mathcal{P}$ is a CDS protocol for f if it satisfies the following properties:*

Correctness. *There is a deterministic reconstruction function $\textsc{Dec} : X_1 \times \cdots \times X_k \times M_1 \times \cdots \times M_k \to S$ such that for every input $x = (x_1, \ldots, x_k) \in X_1 \times \cdots \times X_k$ for which $f(x_1, \ldots, x_k) = 1$, every secret $s \in S$, and every common random string $r \in R$, it holds that $\textsc{Dec}(x, \textsc{Enc}(x, s; r)) = s$.*

Security. *For every input $x = (x_1, \ldots, x_k) \in X_1 \times \cdots \times X_k$ satisfying $f(x_1, \ldots, x_k) = 0$ and every pair of secrets $s, s' \in S$, and for a uniformly distributed $r \leftarrow R$, the distributions $\textsc{Enc}(x, s; r)$ and $\textsc{Enc}(x, s'; r)$ are equivalent.*

The message size *of a CDS protocol $\mathcal{P}$ is defined as the size of the largest message sent by the servers, i.e., $\max_{1 \le i \le k} \log |M_i|$.*

In two-server CDS protocols, we sometimes refer to the servers as Alice and Bob (instead of S_1 and S_2, respectively) and to the referee as Charlie.

Similarly to secret-sharing schemes, all the CDS protocols presented in this paper are with domain of secrets $S = \{0,1\}$, unless stated otherwise.

Definition 3.5 (The predicate INDEX_N^k)**.** *We define the k-input function* $\mathrm{INDEX}_N^k : \{0,1\}^{N^{k-1}} \times [N]^{k-1} \to \{0,1\}$ *where for every* $D \in \{0,1\}^{N^{k-1}}$ *(a* $(k-1)$ *dimensional array called the database) and every* $(i_2, \ldots, i_k) \in [N]^{k-1}$ *(called the index),* $\mathrm{INDEX}_N^k(D, i_2, \ldots, i_k) = D_{i_2,\ldots,i_k}$.

Observation 3.6 ([43])**.** *If there is a k-server CDS protocol for* INDEX_N^k *with message size M, then for every* $f : [N]^k \to \{0,1\}$ *there is a k-server CDS protocol with message size M.*

We obtain the above CDS protocol for f in the following way: Server S_1 with input x_1 constructs a database $D_{i_2,\ldots,i_k} = f(x_1, i_2, \ldots, i_k)$ for every $i_2, \ldots, i_k \in [N]$ and servers $S_2, \ldots, S_k$ treat their inputs $(x_2, \ldots, x_k) \in [N]^{k-1}$ as the index, and execute the CDS protocol for $\mathrm{INDEX}_N^k(D, x_2, \ldots, x_k) = f(x_1, x_2, \ldots, x_k)$.

3.4 Private Information Retrieval (PIR)

A k-server PIR protocol involves k servers $S_1, \ldots, S_k$ and a user. Each server holds the same database $D \in \{0,1\}^N$; the user holds an index $i \in [N]$ and wants to retrieve the bit D_i without revealing i. We consider a simple model of communication (we do not know of information-theoretic PIR protocols where interaction improves the communication) where the user sends a query to each server and each server sends back an answer to the user; this answer is computed from the query the server received and the database D. Given the answers from the servers, the user reconstructs D_i. We next provide a formal definition.

Definition 3.7 (Private information retrieval (PIR) protocols)**.** *A k-server PIR protocol* $\mathcal{P}$ *for a database* $D \in \{0,1\}^N$ *and an index* $i \in [N]$ *consists of k randomized query algorithms* $\mathcal{Q}_1, \ldots, \mathcal{Q}_k$*, k answer algorithms* $\mathcal{A}_1, \ldots, \mathcal{A}_k$ *and a reconstruction algorithm* $\mathcal{C}$*. The user holds the index i, chooses a random string r and sends a query to the servers* $q_1 \leftarrow \mathcal{Q}_1(i; r), \ldots, q_k \leftarrow \mathcal{Q}_k(i; r)$ *respectively. The servers hold the database D, receive a query from the user, and the servers send an answer* $a_1 \leftarrow \mathcal{A}_1(q_1, D), \ldots, a_k \leftarrow \mathcal{A}_k(q_k, D)$ *respectively. Finally, the user computes his output by applying the reconstruction function,* $\mathcal{C}(r, a_1, \ldots, a_k, i)$*. We say that a protocol* $\mathcal{P} = (\mathcal{Q}_1, \ldots, \mathcal{Q}_k, \mathcal{A}_1, \ldots, \mathcal{A}_k, \mathcal{C})$ *is a k-server PIR protocol if it satisfies the following properties.*

Correctness: *The user outputs the correct value. Formally, for every database* $D \in \{0,1\}^N$*, index* $i \in [N]$*, and a random string* $r \in R$*,*

$$\mathcal{C}(r, \mathcal{A}_1(\mathcal{Q}_1(i; r), D), \ldots, \mathcal{A}_k(\mathcal{Q}_k(i; r), D), i) = D_i.$$

Security: *Each server learns no information about i from its query. Formally, for every* $1 \le j \le k$*, every pair of indexes* i_1, i_2*, and for a uniformly distributed* $r \leftarrow R$*, the distributions* $\mathcal{Q}_j(i_1; r)$ *and* $\mathcal{Q}_j(i_2; r)$ *are equivalent.*

The communication complexity of a k-server PIR protocol is the largest message communicated between the servers and the user, i.e., $\max\{|q_1|, \ldots, |q_k|, |a_1|, \ldots, |a_k|\}$*, where the maximum is over all* $D \in \{0,1\}^N, i \in [N], r \in R$*.*

3.5 Matching Vectors

We next define matching vectors (MV). Matching vectors are the key tool for the constructions of the best-known non-linear CDS protocols in LVW17, LVW18, BFL and PIR protocols in [39] and

DG. Here, we present a definition of matching vectors that generalizes the known definition from Efremenko [37], LVW17, LVW18 and DG. In our definition, the matching vector family is defined by two disjoint sets S and T in which the inner products of the vectors lie, whereas in the former definitions the family is only defined by one set S and the inner products lie in $S \cup \{0\}$.

Afterwards, we specify the type of matching vector which we are going to use in our PIR and CDS construction, and show the equivalence to the previous definitions, and henceforth derive an efficient construction for our definition.

Definition 3.8 (Matching vectors)**.** *Let $N, m, h > 0$ be positive integers, and let $S, T \subseteq \mathbb{Z}_m$ be subsets such that $S \cap T = \emptyset$. The vectors family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$, where $\mathbf{u}_i, \mathbf{v}_i \in \mathbb{Z}_m^h$, is called* (S,T)-matching vectors *if the following holds.*

1. $\langle \mathbf{u}_i, \mathbf{v}_i \rangle_m \in T$ *for every* $i \in [N]$.
2. $\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m \in S$ *for every* $i \neq j \in [N]$.

If $T = \{1\}$ then we call the vector family S-matching vectors.

We next specify the matching vectors set that we will use in our CDS and PIR protocols in this paper. Let $m = p_1 \cdot m_2$ and $m_2 = p_2 \cdot \ldots \cdot p_r$ where $p_1, \ldots, p_r$ are r distinct primes. In most previous constructions of CDS and PIR protocols based on matching vector families [39, 36, 60] a variant of this set is used.

$$S_{\text{can},m} = \{a \in \mathbb{Z}_m : \forall i \in [r], a \bmod p_i \in \{0,1\}\} \setminus \{1\}\,.$$

For $r = 2$, BFL use in their construction of 2-server CDS protocol an $S_{\text{one},m}$-matching vectors where

$$S_{\text{one},m} = \{a \in \mathbb{Z}_m : a \equiv 1 \pmod{p_2} \vee a \equiv 1 \pmod{p_1}\}\,.$$

To simplify the construction of k-server CDS and PIR protocols, we use a slightly different matching vectors family,

$$S_{\text{zero},m} = \mathbb{Z}_m \setminus \mathbb{Z}_m^* = \left\{ a \in \mathbb{Z}_m : \bigvee_{i \in [r]} a \equiv 0 \pmod{p_i} \right\}.$$

Finally, we define a "combination" of these sets

$$S^*_{\text{can},m} = \{a \in \mathbb{Z}_m : a \neq 1, a \bmod p_1 = 0 \vee a \bmod m_2 \in S_{\text{can},m_2}\}\,.$$

Note that when m is a product of two primes, $S^*_{\text{can},m} = S_{\text{zero},m}$.

Example 3.9. *Let $m = 2 \cdot 7 \cdot 73$. In this case,*

$$S_{\text{can},m} = \{0, 147, 365, 511, 512, 658, 876\}\,,$$

$$S^*_{\text{can},m} = \{0, 2, 4, \ldots, 1022\} \cup \{147, 365, 511\}\,,$$

$$S_{\text{zero},m} = \{0, 2, 4, \ldots, 1022\} \cup \{0, 7, 14, \ldots, 1017\} \cup \{0, 73, 146, \ldots, 951\}\,.$$

We next prove that the lengths of $(S_{\text{one},m}, 0)$-matching vectors and $S_{\text{zero},m}$-matching vectors are equivalent up to an additive factor of 1.

Claim 3.10. *If there is an $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length h, then there is an $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length $h+1$. If there is an $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length h, then there is an $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length $h+1$.*

Proof. Given $((\mathbf{u}_i, \mathbf{v}_i))_{i\in[N]}$, define $\mathbf{u}'_i = (1, -\mathbf{u}_i), \mathbf{v}'_i = (1, \mathbf{v}_1)$. Then, if $((\mathbf{u}_i, \mathbf{v}_i))_{i\in[N]}$ is an $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length h, then, $((\mathbf{u}'_i, \mathbf{v}'_i))_{i\in[N]}$ is an $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length $h+1$, and if $((\mathbf{u}_i, \mathbf{v}_i))_{i\in[N]}$ is an $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length h, then $((\mathbf{u}'_i, \mathbf{v}'_i))_{i\in[N]}$ is an $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length $h+1$. This follows since for every i, j, $\langle \mathbf{u}'_i, \mathbf{v}'_j \rangle = 1 - \langle \mathbf{u}_i, \mathbf{v}_j \rangle$, and

$$\begin{aligned}
&\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m = 0 \Rightarrow \langle \mathbf{u}'_i, \mathbf{v}'_j \rangle_m = 1,\\
&\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m = 1 \Rightarrow \langle \mathbf{u}'_i, \mathbf{v}'_j \rangle_m = 0,\\
&\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m \in S_{\text{one},m} \Rightarrow \langle \mathbf{u}'_i, \mathbf{v}'_j \rangle_m \in S_{\text{zero},m}, \text{ and}\\
&\langle \mathbf{u}_i, \mathbf{v}_j \rangle_m \in S_{\text{zero},m} \Rightarrow \langle \mathbf{u}'_i, \mathbf{v}'_j \rangle_m \in S_{\text{one},m}.
\end{aligned}$$

□

We can similarly show that the length of $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-matching vectors and $S_{\text{can},m}$-matching vectors are equivalent up to an addition of one entry.

Since $S_{\text{can},m} \cup \{1\} \setminus \{0\} \subset S_{\text{one},m}$, a family of $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-matching vectors is a family of $(S_{\text{one},m}, 0)$-matching vectors; Claim 3.10 and the known results for $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$ matching vectors [46, 58] imply the following theorem.

Theorem 3.11 ([46, 58])**.** *For every r distinct primes $p_1, p_2, \ldots, p_r$, there is an $S_{\text{can},m}$ matching vector family over $\mathbb{Z}_m$ for $m = p_1 \cdot \ldots \cdot p_r$, of size N and length $2^{O(\sqrt[r]{\log N (\log\log N)^{r-1}})}$.*[4]

Trivially, $S_{\text{can},m}$-matching vectors imply $S^*_{\text{can},m}$ and $S_{\text{zero},m}$-matching vectors of the same length. We do not know if such implication exists in the other direction or there is a better construction for $S_{\text{zero},m}$-matching vectors, which would improve the communication of 2-server PIR and CDS protocols.

4 Simplified k-Server PIR Using Share Conversion

In this section, we will describe Protocol 4.4 – a k-server PIR protocol; this protocol is a simplification and generalization of the 2-server PIR protocol of DG and the recent k-server PIR protocol of GKS. The PIR protocols presented in Sections 2.1 and 2.2 are instantiations of this protocol.

Our protocol modifies the previous protocols in two ways. First, we use matching vectors such that $\langle \mathbf{u}_i, \mathbf{v}_i \rangle = 1$ (in previous papers, $\langle \mathbf{u}_i, \mathbf{v}_i \rangle = 0$); this modification simplifies the protocol. The second generalization is the use of a general share conversion from a k-party secret sharing as defined in Section 4.1. Share conversion was defined in [20] and was used to abstract the PIR protocol of [39]; we use the same abstraction in this paper. Using various share conversions in Protocol 4.4, we obtain varios PIR protocols. In Section 5, we show how to use Protocol 4.4 to construct 2-server PIR protocols and in Section 6 we show how to use it to construct k-server PIR protocols for $k \geq 3$. Furthermore, in Section 7, we show that transformations of the 2-server PIR protocol in LVW17, LVW18 yield 2-server and k-server CDS protocols.

[4] The constant in the exponent depends on m.

4.1 Share Conversion

Beimel et al. [20] abstracted the PIR protocol of Efremenko [39] by using share conversion. They described how the notion of share conversion generalizes S-decoding polynomials used in [39] and demonstrated how to construct k-server PIR protocols with short answers using share conversion.

Next, we present the definition of share conversion from a k-party secret-sharing scheme over $\mathbb{Z}_m$ to additive sharing over $\mathbb{F}_{p^\ell}$; in such conversion, the parties are given shares of a secret s, each party locally computes a function of its share to produce new shares of a related secret s'.[5] For simplicity of the definition, we assume that each share in the original scheme is one element from $\mathbb{Z}_m$.[6]

Definition 4.1 (S-share conversion [20])**.** *Let Π be a k-party sharing over $\mathbb{Z}_m$, and let $S \subseteq \mathbb{Z}_m \setminus \{1\}$. The functions $C_1, \ldots, C_k : \mathbb{Z}_m \to \mathbb{F}_{p^\ell}$ are a k-party S-share conversion if the following requirements hold.*

- *If $\mathsf{sh}_1, \ldots, \mathsf{sh}_k = \Pi(1; r)$ for some r, then $C_1(\mathsf{sh}_1) + \cdots + C_k(\mathsf{sh}_k) \neq 0$.*
- *If $\mathsf{sh}_1, \ldots, \mathsf{sh}_k = \Pi(s; r)$ for some r and $s \in S$, then $C_1(\mathsf{sh}_1) + \cdots + C_k(\mathsf{sh}_k) = 0$.*

Example 4.2. *Let $m = 2$ and consider the secret-sharing scheme over $\mathbb{Z}_2$, where the dealer chooses a uniformly distributed bit r, $\mathsf{sh}_1 = r$, and $\mathsf{sh}_2 = s \oplus r$. We describe a 2-party $\{0\}$-share conversion from this scheme to the additive scheme over $\mathbb{F}_3$, where $C_1(a) = a$ and $C_2(a) = -a \bmod 3$ for every $a \in Z_2$. This conversion satisfies the requirements of Definition 4.1:*

- *If $\mathsf{sh}_1, \mathsf{sh}_2 = \Pi(1; r)$ for some r, then either $\mathsf{sh}_1 = 1$ and $\mathsf{sh}_2 = 0$ and $C_1(1) + C_2(0) \equiv 1 - 0 \equiv 1 \pmod 3$, or $\mathsf{sh}_1 = 0$ and $\mathsf{sh}_2 = 1$ and $C_1(0) + C_2(1) \equiv 0 + (-1) \equiv 2 \pmod 3$.*
- *If $\mathsf{sh}_1, \mathsf{sh}_2 = \Pi(0; r)$ for some r, then $\mathsf{sh}_1 = \mathsf{sh}_2$ and $C_1(\mathsf{sh}_1) + C_2(\mathsf{sh}_2) \equiv \mathsf{sh}_1 - \mathsf{sh}_2 \equiv 0 \pmod 3$.*

In our PIR protocols, we will use share conversion from a linear secret-sharing scheme. Recall that a secret-sharing scheme over $\mathbb{Z}_m$ is linear if each share is a linear combination of the secret and uniformly distributed random elements in $\mathbb{Z}_m$. For example of a linear scheme, let $\alpha_1, \ldots, \alpha_k$ be some constants in $\mathbb{Z}_m^*$ and consider the scheme where the dealer shares s by choosing $r \in \mathbb{Z}_m$ with uniform distribution and outputting the shares $\mathsf{sh}_1 = r + \alpha_1 s, \ldots, \mathsf{sh}_k = r + \alpha_k s$.

The following observation follows from the linearity of a secret-sharing scheme.

Observation 4.3. *Let Π be a linear secret-sharing scheme over $\mathbb{Z}_m$. If $\mathsf{sh}_1, \ldots, \mathsf{sh}_k = \Pi(s; r)$ and $\mathsf{sh}'_1, \ldots, \mathsf{sh}'_k = \Pi(s'; r')$, then $\mathsf{sh}_1 + \mathsf{sh}'_1, \ldots, \mathsf{sh}_k + \mathsf{sh}'_k$ are shares of the secret $s + s' \bmod m$ (using the randomness $\mathbf{r} + \mathbf{r}'$).*

Furthermore, for the security of the PIR protocol we use a scheme that is 1-private, i.e., the scheme satisfies the security requirement of Definition 3.2with respect to all singletons. We do not need that bigger sets can reconstruct the secret.

[5] Beimel et al. gave a broader definition of a share conversion with respect to a relation $R \subseteq \mathbb{Z}_m \times \mathbb{F}_{p^\ell}$ in which k parties hold shares of a secret s given by a secret-sharing scheme over $\mathbb{Z}_m$ and each party locally computes a share of a secret s' over a field $\mathbb{F}_{p^\ell}$ such that $(s, s') \in R$. We only consider a specific relation.

[6] The definition can be easily generalized to the case that each share is composed of more than 1 element (e.g., CNF sharing).

4.2 The PIR Protocol Using Share Conversion

Now, we present the k-server PIR protocol that uses share conversion.

Protocol 4.4 (A k-server PIR protocol)**.**
Public parameters: *An $S^*_{\mathrm{can},m}$-matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ over $\mathbb{Z}_m^h$ for $m = p_1 \cdot \ldots \cdot p_r$ a product of r distinct primes.*
A k-party 1-private linear secret-sharing scheme Π over $\mathbb{Z}_{m_2}$, where $m_2 = p_2 \cdot \ldots \cdot p_r$.
A k-party S_{can,m_2}-share conversion $C_1, \ldots, C_k$ from Π to additive sharing over $\mathbb{F}_{p_1^\ell}$.
Servers' Input: $D \in \{0,1\}^N$.
User's input: $i \in [N]$.

- *For every $\ell \in [h]$, the user shares $\mathbf{u}_i[\ell]$ using Π, with shares denoted as $\mathsf{sh}_1^\ell, \ldots, \mathsf{sh}_k^\ell$, and sends $\mathbf{q}_t \leftarrow (\mathsf{sh}_t^1, \ldots, \mathsf{sh}_t^h) \in \mathbb{F}_{m_2}^h$ to the t-th server for $1 \le t \le k$.*
- *The t-th server sends to the user $\mathbf{a}_t \in \mathbb{F}_{p^\ell}^h$, where*

$$\mathbf{a}_t \leftarrow \sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) \cdot C_t(\langle \mathbf{q}_t, \mathbf{v}_j \rangle_{m_2}).$$

- *The user computes $\sum_{t=1}^k \langle (\mathbf{u}_i \bmod p_1), \mathbf{a}_t \rangle$ over $\mathbb{F}_{p_1^\ell}$ and return 0 iff the expression equals to 0.*

Theorem 4.5. *Let $m = p_1 \cdot \ldots \cdot p_r$ be a product of r distinct primes and $m_2 = p_2 \cdot \ldots \cdot p_r$. Assume there is a family of $S^*_{\mathrm{can},m}$-matching vectors in $\mathbb{Z}_m^h$ of size N and a k-party S_{can,m_2}-share conversion from $\mathbb{Z}_{m_2}$ to $\mathbb{F}_{p_1^\ell}$ for some $\ell \in \mathbb{N}$. Then, Protocol 4.4 is a k-server PIR protocol for a database of size N with message size $h \cdot \log(m_2 \cdot p_1^\ell)$.*

Proof. In order to prove the theorem, we need to show that Protocol 4.4 satisfies correctness and security.

Correctness. First, note that $(\mathbf{u}_i \bmod p_1)$ is a vector over $\mathbb{F}_{p_1}$ and therefore also over $\mathbb{F}_{p_1^\ell}$. Also, $\sum_{j=1}^N D_j(\mathbf{v}_j \pmod{p_1}) \cdot C_t(\langle \mathbf{q}_t, \mathbf{v}_j \rangle_{m_2})$ is a linear combination of vectors over $\mathbb{F}_{p_1}$ with coefficients in $\mathbb{F}_{p_1^\ell}$. Thus, when applying an inner product over $\mathbb{F}_{p_1^\ell}$ we get

$$\begin{aligned}\langle (\mathbf{u}_i \bmod p_1), \mathbf{a}_t \rangle &= \langle (\mathbf{u}_i \bmod p_1), \sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) \cdot C_t(\langle \mathbf{q}_t, \mathbf{v}_j \rangle_{m_2}) \rangle \\ &= \sum_{j=1}^{N} \left(D_j \cdot C_t(\langle \mathbf{q}_t, \mathbf{v}_j \rangle_{m_2})\right) \langle \mathbf{u}_i, \mathbf{v}_j \rangle_{p_1}.\end{aligned}$$

The user computes

$$\begin{aligned}\sum_{t=1}^{k}\langle(\mathbf{u}_i \bmod p_1),\mathbf{a}_t\rangle &= \sum_{t=1}^{k}\sum_{j=1}^{N}\left(D_j \cdot C_t(\langle \mathbf{q}_t,\mathbf{v}_j\rangle_{m_2})\right)\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{p_1}\\ &= \sum_{j=1}^{N}\sum_{t=1}^{k}\left(D_j \cdot C_t(\langle \mathbf{q}_t,\mathbf{v}_j\rangle_{m_2})\right)\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{p_1}\\ &= \sum_{j=1}^{N} D_j \cdot \langle \mathbf{u}_i,\mathbf{v}_j\rangle_{p_1}\sum_{t=1}^{k} C_t(\langle \mathbf{q}_t,\mathbf{v}_j\rangle_{m_2}). \end{aligned} \tag{5}$$

Recall that $\mathbf{q}_1[\ell],\ldots,\mathbf{q}_k[\ell]$ are shares of $\mathbf{u}_i[\ell]$, thus, by Observation 4.3, $\langle \mathbf{q}_1,\mathbf{v}_j\rangle_{m_2},\ldots,\langle \mathbf{q}_k,\mathbf{v}_j\rangle_{m_2}$ are shares of the secret $\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{m_2}$. We next prove that the j-th term in Equation (5), for $i \neq j$, is 0, while the i-th term is some non-zero number times D_i:

$j = i$. In this case, $\langle \mathbf{u}_i,\mathbf{v}_i\rangle_{p_1} = 1$ and $\langle \mathbf{q}_1,\mathbf{v}_i\rangle_{m_2},\ldots,\langle \mathbf{q}_k,\mathbf{v}_j\rangle_{m_2}$ are shares of $\langle \mathbf{u}_i,\mathbf{v}_i\rangle_{m_2} = 1$, thus, by the definition of share conversion, $\sum_{t=1}^{k} C_t(\langle \mathbf{q}_t,\mathbf{v}_j\rangle_{m_2}) \neq 0$.

$j \neq i$. In this case, $\langle \mathbf{u}_i,\mathbf{v}_j\rangle_m \in S^*_{\mathrm{can},m}$. Therefore, either $\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{p_1} = 0$ and the term is zero, or $\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{m_2} \in S_{\mathrm{can},m_2}$. In the latter case, $\langle \mathbf{q}_1,\mathbf{v}_i\rangle_{m_2},\ldots,\langle \mathbf{q}_k,\mathbf{v}_j\rangle_{m_2}$ are shares of $\langle \mathbf{u}_i,\mathbf{v}_j\rangle_{m_2} \in S_{\mathrm{can},m_2}$, thus, by the definition of S_{can,m_2}-share conversion $\sum_{t=1}^{k} C_t(\langle \mathbf{q}_t,\mathbf{v}_j\rangle_{m_2}) = 0$. We conclude that for $j \neq i$, the value of the j-th term is 0.

Thus,

$$\sum_{t=1}^{k}\langle(\mathbf{u}_i \bmod p_1),\mathbf{a}_t\rangle = \alpha \cdot D_i \tag{6}$$

for some non-zero $\alpha \in \mathbb{F}_{p_1^\ell}$ and the user returns D_i.

Security. By the 1 privacy of Π, each query is identically distributed for every for every index i. □

5 A Simplified 2-Server PIR protocol

We next show how to construct a 2-server PIR protocol using Protocol 4.4, that is, we need to construct 2-party share conversions and explain which family of matching vectors we use. The 2-server PIR protocols that we get are simplifications and generalizations of the 2-server PIR protocols of DG, LVW, and BFL. Our protocols can use an $S_{\mathrm{zero},m}$-matching vector family over any m that is a product of two distinct primes. The protocols of DG and LVW use $(S_{\mathrm{can},m},0)$-matching vector family over $\mathbb{Z}_6$ and the protocol of BFL uses $(S_{\mathrm{one},m},0)$-matching vector family for $m = p_1p_2$ s.t. $p_2|p_1-1$. Thus, our protocols offer full flexibility in choosing m as a product of two primes.

Let $m = p_1p_2$; in this case $m_2 = p_2$ and $S_{\mathrm{can},m_2} = \{0\}$. Consider the following 2-party secret-sharing scheme Π over $\mathbb{F}_{p_2}$, where for every $s \in \mathbb{F}_{p_2}$ the dealer chooses a uniformly distributed $r \in \mathbb{Z}_{p_2}$ and computes $\mathsf{sh}_1 = r$ and $\mathsf{sh}_2 = r + s \bmod p_2$. The reconstruction of the secret is $\mathsf{sh}_2 - \mathsf{sh}_1 \bmod p_2 = s$. We construct several $\{0\}$-share conversions from this secret-sharing scheme to an additive sharing over $\mathbb{F}_{p_1}$ (i.e., $\ell = 1$). Let $C_1, C_2 : \mathbb{F}_{p_2} \to \mathbb{F}_{p_1}$ be such share conversion; for a sharing of $s = 0$, i.e., $\mathsf{sh}_1 = \mathsf{sh}_2$, we require $C_1(\mathsf{sh}_1) + C_2(\mathsf{sh}_1) \equiv 0 \pmod{p_1}$, i.e., $\Rightarrow C_1(\mathsf{sh}_1) \equiv -C_2(\mathsf{sh}_1) \pmod{p_1}$. Therefore, we define 2-party share conversion by one function C that satisfies that for a sharing of $s = 1$, i.e., $\mathsf{sh}_2 - \mathsf{sh}_1 \equiv 1 \pmod{p_2}$, $C(\mathsf{sh}_2) - C(\mathsf{sh}_1) \not\equiv 0 \pmod{p_1}$.

5.1 Constructing 2-Party Share Conversions

We next describe three 2-party $\{0\}$-share conversions.

5.1.1 The First Share Conversion

The first share conversion is used in Section 2; it assumes that $p_1 \nmid p_2 - 1$, in particular, it can be applied when $p_2 < p_1$.

Claim 5.1. *Let p_1, p_2 be primes such that $p_1 \nmid p_2 - 1$, and let $C^1 : \mathbb{F}_{p_2} \to \mathbb{F}_{p_1}$, where for every $x \in \mathbb{F}_{p_2}$, $C^1(x) = x \bmod p_1$. Then, C^1 is a 2-party share conversion.*

Proof. We need that for every $x \in \mathbb{F}_{p_2}$, $C^1(x + 1 \bmod p_2) - C^1(x) \not\equiv 0 \pmod{p_1}$. Let $x \in \mathbb{F}_{p_2}$,

- If $x \not\equiv p_2 - 1 \pmod{p_2}$, then

$$C^1(x + 1 \bmod p_2) - C^1(x) \equiv C^1(x + 1) - C^1(x) \equiv x + 1 - x \equiv 1 \not\equiv 0 \pmod{p_1}.$$

- Otherwise, if $x = p_2 - 1$, then, since $p_1 \nmid p_2 - 1$

$$\begin{aligned} C^1(p_2 - 1 + 1 \bmod p_2) - C^1(p_2 - 1) &\equiv C^1(0) - C^1(p_2 - 1) \\ &\equiv 0 - (p_2 - 1) \equiv 1 - p_2 \not\equiv 0 \pmod{p_1}. \end{aligned}$$

□

In particular, if $p_2 < p_1$ then $C^1(x) = x$ is a share conversion. We note that when $p_1 | p_2 - 1$, $C^1(x) = x \bmod p_1$ is not a share conversion since

$$C^1(p_2 - 1 + 1) - C^1(p_2 - 1) \equiv 0 - p_2 + 1 \equiv 0 \pmod{p_1}.$$

5.1.2 The Second Share Conversion

The second share conversion assumes that $p_2 | p_1 - 1$. This is the share conversion used in BFL and its generalization is used in [39] and GKS and in our k-server PIR protocol for $k \geq 3$. The share conversion uses an element $\gamma \in \mathbb{F}^*_{p_1}$ whose order is p_2, i.e., p_2 is the smallest positive integer s.t. $\gamma^{p_2} \equiv 1 \pmod{p_2}$. Such an element exists if and only if $p_2 | p_1 - 1$.

Claim 5.2. *Let p_1, p_2 primes s.t. $p_2 | p_1 - 1$, and let $\gamma \in \mathbb{F}^*_{p_1}$ be an element of order p_2. Then the following function, $C^2(x) = \gamma^x \bmod p_1$ is a share conversion.*

Proof. Since the order of γ is p_2, the mapping C^2 is indeed from $\mathbb{F}_{p_2}$ to $\mathbb{F}_{p_1}$. For $x \in \mathbb{F}_{p_2}$,

$$C^2(x + 1 \bmod p_2) - C^2(x) \equiv \gamma^{x+1 \bmod p_2} - \gamma^x \equiv \gamma^{x+1} - \gamma^x \equiv \gamma^x(\gamma - 1) \pmod{p_1}.$$

Since $\gamma \in \mathbb{F}^*_{p_1}$, we know that $\gamma \neq 0$; since the order of γ is $p_2 > 1$, we know that $\gamma \neq 1$, therefore $C^2(x + 1) - C^2(x) \not\equiv 0 \pmod{p_1}$. □

5.1.3 The Third Share Conversion

The third share conversion only assumes that $p_1 \neq 2$. The size of the range of this share conversion is 3.

Claim 5.3. *Let p_1, p_2 primes, such that $p_1 > 2$ and define C^3 as follows:*

$$C^3(x) = \begin{cases} x \bmod 2 & 0 \leq x < p_2 - 1 \\ 2 & x = p_2 - 1 \end{cases}.$$

Then, C^3 is a share conversion.

Proof. Let $x \in \mathbb{F}_{p_2}$,

- $0 \leq x < p_2 - 1$, then $C^3(x+1) - C^3(x) \in \{-1, 1\}$, i.e., the difference is non-zero.
- Otherwise, if $x = p_2 - 1$, then, since $p_1 > 2$,

$$C^3(x + 1 \bmod p_2) - C^3(x) \equiv C^3(0) - C^3(p_2 - 1) \equiv 0 - 2 \not\equiv 0 \pmod{p_1}.$$

□

5.2 The 2-Server PIR Protocol

We next present the simplified 2-server PIR protocol using share conversion; this is an instantiation of Protocol 4.4. A concrete implementation of this 2-server PIR protocol is given in Protocol 2.1.

Protocol 5.4 (A 2-server PIR protocol from share conversion)**.**
Public parameters: *An $S_{\text{zero},m}$-matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ over $\mathbb{Z}_m^h$ for $m = p_1 p_2$.*
A 2-party $\{0\}$-share conversion $C : \mathbb{F}_{p_2} \to \mathbb{F}_{p_1}$.
Servers' Input: $D \in \{0,1\}^N$.
User's input: $i \in [N]$.
User's randomness: $\mathbf{r} \in \mathbb{F}_{p_2}^h$.

- *The user sends queries $\mathbf{q}_1 \leftarrow \mathbf{r} \bmod p_2$ and $\mathbf{q}_1 \leftarrow \mathbf{r} + \mathbf{u}_i$ to the first server and second server, respectively.*
- *The t-th server (for $t \in \{1,2\}$) sends to the user $\mathbf{a}_t \in \mathbb{F}_{p_1}^h$, where*

$$\mathbf{a}_t \leftarrow \sum_{j=1}^{N} D_j(\mathbf{v}_j \bmod p_1) C(\langle \mathbf{q}_t, \mathbf{v}_j \rangle_{p_2}).$$

- *The user outputs 0 iff $\langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 \rangle_{p_1} = 0$.*

By using the 2-party share conversion C^1 of Claim 5.1 (taking p to be the bigger factor of m) and Theorem 4.5 we obtain the following result. Recall that for $S^*_{\text{can},m} = S_{\text{zero},m}$ when m is a product of two primes.

Theorem 5.5. *Let p_1, p_2 be two distinct primes and let $m = p_1 p_2$. Assume there is a family of $S_{\text{zero},m}$-matching vectors in $\mathbb{Z}_m^h$ of size N. Then, there exists a 2-server PIR protocol with database of length N and with message size $h \cdot \log m$.*

Using the matching vector family of [46, 58] over $\mathbb{Z}_6$, we get the result of DG [35].

Corollary 5.6. *There is a 2-server PIR protocol with message size $2^{\tilde{O}(\sqrt{\log N})}$.*

6 k-Server PIR Protocols from Sparse S-Decoding Polynomials

In this section, we show how to instantiate Protocol 4.4 to obtain k-server PIR protocols for $k > 2$. The protocols we obtain are a simplification of the recent PIR protocol of GKS, obtaining the same communication complexity. To instantiate the protocol, we use the share conversion via sparse decoding polynomials, introduced by Efremenko [39]. For the sake of completeness, we describe this share conversion below.

6.1 k-Sparse Decoding Polynomials

As we previously described, Efremenko [39] presented a construction for k-server PIR protocol. Beimel et al. [20] defined share conversion and observed that Efremenko implicitly constructed share conversions via sparse decoding polynomials. We present a definition of decoding polynomials that generalizes the definition from [39], defining them by a subset S and an element $t \notin S$, where the former definition only was defined by S and $t = 0$.

Definition 6.1 (Decoding polynomials)**.** *Let $m > 0$, $t \in \mathbb{Z}_m$, $S \subseteq \mathbb{Z}_m \setminus \{t\}$, and p be a prime s.t. $m \mid p^\ell - 1$ for some $\ell \geq 1$. Let $\gamma \in \mathbb{F}_{p^\ell}$ be an element of order m. A polynomial $P(X) \in \mathbb{F}_{p^\ell}[X]$ is called an (S, t)-decoding polynomial if:*

$$P(\gamma^s) = \begin{cases} \alpha \neq 0 & \text{if } s = t \\ 0 & \text{if } s \in S. \end{cases}$$

If $t = 1$, we call the polynomial an S-decoding polynomial. A polynomial is k-sparse if it has at most k non-zero coefficients.

In our construction, we will use $S_{\text{can},m}$-decoding polynomial for some $m = p_1 \cdot \ldots \cdot p_r$, where the known constructions of decoding polynomials in [52, 26] are $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-decoding polynomial. We now show how to construct $S_{\text{can},m}$-decoding polynomial from an $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-decoding polynomial with the same sparsity.

Lemma 6.2. *If there is a k-sparse $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-decoding polynomial over $\mathbb{F}_{p^\ell}$, then there is a k-sparse $S_{\text{can},m}$-decoding polynomial.*

Proof. Let $P(X)$ be an $(S_{\text{can},m} \cup \{1\} \setminus \{0\}, 0)$-decoding polynomial over $\mathbb{F}_{p^\ell}$ for the element $\gamma \in \mathbb{F}_{p^\ell}$ of order m. We define $P'(X) = P(\gamma \cdot X^{m-1})$. For every s, $P'(\gamma^s) = P(\gamma \cdot (\gamma^s)^{m-1}) = P(\gamma \cdot \gamma^{s(m-1)}) = P(\gamma^{(1-s) \bmod m})$. Therefore,

1. $P'(\gamma^1) = P(\gamma^0) = 1$.
2. If $s \in S_{\text{can},m}$, then, by the CRT $(1 - s) \bmod m \in S_{\text{can},m} \cup \{1\} \setminus \{0\} \Rightarrow P'(\gamma^s) = P(\gamma^{1-s}) = 0$.

□

We next show that a k-sparse decoding polynomial implies a k-party share conversion (as discussed in [20]).

Lemma 6.3. *Let $m = p_1 \cdot \ldots \cdot p_r$ be a product of r distinct primes, p be a prime and $\ell \in \mathbb{N}$ such that $m \mid p^\ell - 1$. If there is a k-sparse $S_{\text{can},m}$-decoding polynomial over $\mathbb{F}_{p^\ell}$, then there is a k-party $S_{\text{can},m}$-share conversion from $\mathbb{Z}_m$ to $\mathbb{F}_{p^\ell}$.*

Proof. Let $P(x) = \sum_{t=1}^{k} b_t x^{\alpha_t}$ be an explicit representation of a k-sparse $S_{\text{can},m}$-decoding polynomial over $\mathbb{F}_{p^\ell}$, and $\gamma \in \mathbb{F}_{p^\ell}^*$ be an element of order m in $\mathbb{F}_{p^\ell}$ (such γ exists since m divides $2^\ell - 1$ – the order of the multiplicative group of $\mathbb{F}_{p^\ell}$). We consider the following linear secret-sharing scheme over $\mathbb{Z}_m$: To share a secret $s \in \mathbb{Z}_m$, the dealer chooses $r \in \mathbb{Z}_m$ with uniform distribution and computes the shares $\mathsf{sh}_t \leftarrow r + \alpha_t \cdot s \bmod m$ for $1 \le t \le k$. This is a variant of Shamir's 2-out-of-k secret-sharing scheme; in this scheme, the evaluation points are determined by the decoding polynomial P. We define the share conversion $C_t : \mathbb{Z}_m \to F_{p^\ell}$ for $1 \le t \le k$ as follows:

$$C_t(a) = b_t \cdot \gamma^a.$$

We next prove that this is a share conversion. Let $\mathsf{sh}_1 = r + \alpha_1 \cdot s \bmod m, \ldots, \mathsf{sh}_k = r + \alpha_k \cdot s \bmod m$ be shares of a secret s. Then,

$$\sum_{t=1}^{k} C_t(\mathsf{sh}_t) = \sum_{t=1}^{k} b_t \cdot \gamma^{r+\alpha_t \cdot s \bmod m} = \sum_{t=1}^{k} b_t \cdot \gamma^{r+\alpha_t \cdot s} = \gamma^r \sum_{t=1}^{k} b_t \cdot (\gamma^s)^{\alpha_t} = \gamma^r P(\gamma^s).$$

On the one hand, $P(\gamma) \neq 0$, thus $\sum_{t=1}^{k} C_t(\mathsf{sh}_t) \neq 0$ for $s = 1$. On the other hand, $P(\gamma^s) \neq 0$ for every $s \in S_{\text{can},m}$, thus $\sum_{t=1}^{k} C_t(\mathsf{sh}_t) = 0$ for $s \in S_{\text{can},m}$. □

Combining Lemma 6.3 with Theorem 4.5, we obtain the following result.

Theorem 6.4. *Let $m = p_1 \cdot \ldots \cdot p_r$ be a product of r distinct primes and let $m_2 = p_2 \cdot \ldots \cdot p_r$ such that $m_2 | p_1^\ell - 1$ for some $\ell \ge 1$. Assume there is a family of $S^*_{\text{can},m}$-matching vectors in $\mathbb{Z}_m^h$ and a k-sparse S_{can,m_2}-decoding polynomial. Then, there is a k-server PIR protocol over $\mathbb{F}_{p_1}^\ell$ with message size $O(h \log(m_2 \cdot p_1^\ell))$.*

The best known decoding polynomials. Efremenko [39] showed that for every $m = p_1 \cdot \ldots \cdot p_r$ that is a product of r distinct primes, there is some $\ell \in \mathbb{N}$ and a 2^r-sparse S_{can,m_2}-decoding polynomial over $\mathbb{F}_{p_1^\ell}$. Efremenko also presented a 3-sparse $S_{\text{can},7\cdot73}$-decoding polynomial over $\mathbb{F}_{2^9}$ (as opposed to the trivial polynomial that is 4-sparse). Itoh and Suzuki [51, 52] improved on Efremenko's work by giving a recursive method to construct $S_{\text{can},m}$-decoding polynomials by composing decoding polynomials over fewer primes.

Theorem 6.5 ([51, 52])**.** *Let m_1, m_2 be products of r_1, r_2 distinct primes respectively and p be a prime s.t. m_1, m_2, p, are co-prime. Furthermore, let $\ell_1, \ell_2 \in \mathbb{N}$. Assume there is a k_1-sparse S_{can,m_2}-decoding polynomial $P_1(x) \in \mathbb{F}_{p^{\ell_1}}[x]$ and a k_2-sparse S_{can,m_1}-decoding polynomial $P_2(x) \in \mathbb{F}_{p^{\ell_2}}[x]$. Let $m = m_1 \cdot m_2$ and $\ell = \operatorname{lcm}(\ell_1, \ell_2)$. Then, there is a $k_1 k_2$-sparse $S_{\text{can},m}$-decoding polynomial $P(x) \in \mathbb{F}_{p^\ell}[x]$.*

For example, taking Efremenko [39] 3-sparse $S_{\text{can},7\cdot73}$-decoding polynomial over $\mathbb{F}_{2^9}$ and a trivial 2-sparse $S_{\text{can},3}$-decoding polynomial over $\mathbb{F}_{2^2}$, namely the polynomial $x-1$, Itoh and Suzuki [51] got a 6-sparse $S_{\text{can},3\cdot7\cdot73}$ decoding polynomial over $\mathbb{F}_{2^{18}}$. Other 6-sparse $S_{\text{can},m}$ decoding polynomials, where m is a product of 3 primes, were discovered by Amran [3]. (see Table 1).Using Theorem 6.4 with these polynomials and the matching vector family of [46, 58] results in a 6-server PIR protocol with communication complexity $2^{\tilde{O}(\sqrt[4]{\log N})}$ (this decoding polynomial was overlooked in [26, 45]).

To use Theorem 6.5, Itoh and Suzuki [52] and Chee et al. [26] constructed additional 3-sparse decoding polynomials. The best-known decoding polynomials are described in the following theorem; it is slightly better than the result described by Chee et al. [26] using the 6-sparse polynomials.

m	$\mathbb{F}_{p^\ell}$	$S_{\text{can},m}$-decoding polynomial
$3 \cdot 5 \cdot 13$	$\mathbb{F}_{2^{12}}$	$P(X) = \gamma^{140}X^{194} + \gamma^{145}X^{187} + \gamma^{117}X^{156} + \gamma^{188}X^{131} + \gamma^{115}X^{73} + \gamma^{82}$
$3 \cdot 5 \cdot 17$	$\mathbb{F}_{2^{18}}$	$P(X) = X^{254} + \gamma^{45}X^{254} + \gamma^{206}X^{228} + \gamma^{96}X^2 + \gamma^{88}X^{79} + \gamma^{20}$
$3 \cdot 7 \cdot 13$	$\mathbb{F}_{2^{12}}$	$P(X) = \gamma^{257}X^{270} + \gamma^{91}X^{232} + \gamma^{133}X^{181} + \gamma^{26}X^{107} + \gamma 183X^{59} + \gamma^{110}$
$3 \cdot 11 \cdot 31$	$\mathbb{F}_{2^{10}}$	$P(x) = \gamma^{825}X^{1022} + \gamma^{894}X^{1012} + \gamma^{600}X^{970} + \gamma^{584}X^{914} + \gamma^{810}X^{300} + \gamma^{775}$
$5 \cdot 7 \cdot 13$	$\mathbb{F}_{2^{12}}$	$P(X) = \gamma^{414}X^{454} + \gamma^{384}X^{438} + \gamma^{169}X^{372} + \gamma^{47}X^{362} + \gamma^{296}X^{205} + \gamma^{434}$

Table 1: 6-sparse $S_{\text{can},m}$-decoding polynomials from [3].

Theorem 6.6. *There exist k-sparse $S_{\text{can},m}$-decoding polynomials, for some m that is a product of r distinct primes, for the following values:*

- *For an even $2 < r < 104$ and is even and $k \leq 3^{r/2}$,*
- *For an odd $1 < r < 103$ and $k = 2 \cdot 3^{\frac{r-1}{2}}$, and*
- *For $r \geq 104$ and $k = (3/4)^{51} \cdot 2^r$.*

For all such values k, r, there is a k-server PIR protocol with communication complexity $2^{\tilde{O}(\sqrt[r]{\log(N)})}$.

Next, we present a result by Amran [3], constructing 6-sparse $S_{\text{can},m}$-decoding polynomials; this construction does not use the composition method and was found using a computer search. We present the $S_{\text{can},m}$-decoding polynomials over $\mathbb{F}_{p^\ell}$ in Table 1:

6.2 $S_{\text{zero},m}$-Decoding Polynomials

When m is a product of r distinct primes and p is co-prime to m, there is a trivial 2^r-sparse $S_{\text{can},m}$-decoding polynomial over $\mathbb{F}_{p^\ell}$, namely $\prod_{s \in S_{\text{can},m}}(x - \gamma^s)$, where $\ell < m$ satisfies $m \mid p^\ell - 1$ (such ℓ exists by Euler's theorem) and γ is an element of order m in $\mathbb{F}^*_{p^\ell}$. In this section, we show that there is a 2^r-sparse $S_{\text{zero},m}$-decoding polynomial when m is a product of r distinct primes, that is, this polynomial is a decoding polynomial for a bigger set than the trivial $S_{\text{can},m}$-decoding polynomial. As a result, if one constructs "short" $S_{\text{zero},m}$-matching vectors for m that is a product of r primes (compared to the known $S_{\text{can},m}$-matching vectors), then we will get improved 2^r-server PIR protocols.

Claim 6.7. *Let $m = p_1 \cdot \ldots \cdot p_r$ be a product of r distinct primes. Then, there is some $\ell \in \mathbb{N}$ such that the polynomial $P(x) = \prod_{i=1}^{r}(x^{\frac{m}{p_i}} - 1)$ over $\mathbb{F}_{p^\ell}$ is a 2^r-sparse $S_{\text{zero},m}$-decoding polynomial.*

Proof. Take $\ell < m$ such that $m \mid p^\ell - 1$ (such ℓ exists by Euler's theorem), g a generator of $\mathbb{F}^*_{p^\ell}$, and $\gamma = g^{(2^\ell - 1)/m}$; the order of γ is m. First, it is clear that the number of monomials in $P(x)$ is at most 2^r. Next, we show that it is an $S_{\text{can},m}$-decoding polynomial. Since the order of γ is m, $\gamma^{m/p_i} \neq 1$ for every $1 \leq i \leq r$, thus $P(\gamma) \neq 0$. Let $s \in S_{\text{zero},m}$, i.e., there exists $j \in [r]$ s.t. $s \equiv 0 \pmod{p_j}$ and $s = q \cdot p_j$ for some $q > 0$. Then,

$$(\gamma^s)^{\frac{m}{p_j}} = \gamma^{(qp_j)\cdot\frac{m}{p_j}} = (\gamma^m)^q = 1.$$

Therefore, $P(\gamma^s) = 0$. □

We can replace in Theorem 6.4 the $S^*_{\text{can},m}$-decoding polynomial with the $S_{\text{zero},m}$-decoding polynomial of Claim 6.7 and get the following theorem.

Theorem 6.8. *Let $m = p_1 \cdot m_2$ and $m_2 = p_2 \cdot \ldots \cdot p_r$. If there is an $S_{\text{zero},m}$-matching vector family with vectors of length h, then there is a 2^r-server PIR protocol with communication complexity $h \cdot \log(m_2 \cdot p_1^\ell)$ for some $\ell < m$.*

We note that Theorem 6.8 does not yield better PIR protocols than the protocol of GKS since we do not know any $S_{\text{zero},m}$-matching vector families that are asymptotically shorter than the best known $S_{\text{can},m}$-matching vectors. Theorem 6.8 gives the motivation to find better $S_{\text{zero},m}$-matching vectors in order to improve the communication for PIR protocols.

6.3 Are There Better Share Conversions?

To construct a k-server PIR protocol using Theorem 4.5, we need a k-server S_{can,m_2} share conversion to $\mathbb{F}_{p_1^\ell}$, where m_2 is a product of $r-1$ distinct primes and p_1 is co-prime to m_2. Efremenko [39] has (implicitly) shown that such share conversions exist when k-sparse decoding polynomials exist (see Lemma 6.3). Constructing share conversions that do not use decoding polynomials can improve the communication complexity of PIR protocols.

Constructions of 3-server S_{can,m_2} share conversion to $\mathbb{F}_{p_1^\ell}$, where m_2 is a product of 2 distinct primes were given by Beimel et al. [20] (using a computerized search), Paskin-Cherniavsky and Schmerler [65], and Paskin-Cherniavsky and Nissenbaum [64]. However, in all these constructions p_1 divides m_2; thus, these share conversions cannot be used in Theorem 4.5.

The question of constructing suitable share conversions (for small values of k, r) without using decoding polynomials is left as an open problem.

7 Simplified CDS protocols

In this section, we present our simplified CDS protocols. In Section 7.1 we present the 2-server protocol, and in Section 7.2 we present our multi-server protocol.

7.1 The 2-Server CDS Protocol

We next present Protocol 7.1 – our simplified 2-server CDS protocol using 2-server share conversion as in Section 5. Protocol 7.1 is obtained, as in LVW17 by exchanging the roles of the parties in Protocol 5.4 – our 2-server PIR protocol. In more details, the randomness of the user is now the shared random string of the servers; Alice computes the same message as her answer in the PIR protocol and Bob computes the query q_B and sends it to Charlie, and Charlie computes the answer of Bob in the PIR protocol (Charlie holds the database, while Bob does not hold it). Furthermore, some masking is added to prevent Charlie from learning information on the common random string.

Protocol 7.1 (A 2-server CDS protocol)**.**
Public parameters: *An $S_{\text{zero},m}$-matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ over $\mathbb{Z}_m^h$ for $m = p_1 p_2$.*
Alice's input: $D \in \{0,1\}^N$.
Bob's input: $i \in [N]$.
The secret: $s \in \{0,1\}$.

Shared randomness: $\mathbf{r}_1 \in \mathbb{F}_{p_2}^h$, $\mathbf{r}_2 \in \mathbb{F}_{p_1}^h$.
Notations: *Let* $C : \mathbb{F}_{p_2} \to \mathbb{F}_{p_1}$ *be a share conversion.*

- *Alice sends* $\mathbf{m}_A \leftarrow \sum_{j=1}^{N} C(\langle \mathbf{r}_1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j + \mathbf{r}_2 \bmod p_1$.
- *Bob sends* $\mathbf{m}_B^1 \leftarrow (s\mathbf{u}_i + \mathbf{r}_1 \bmod p_2)$, $m_B^2 \leftarrow \langle \mathbf{u}_i, \mathbf{r}_2 \rangle_{p_1}$.
- *Charlie outputs 1 if*

$$\langle \mathbf{u}_i, \sum_{j=1}^{N} C(\langle \mathbf{m}_B^1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j - \mathbf{m}_A \rangle_{p_1} + m_B^2 \not\equiv 0 \pmod{p_1}, \tag{7}$$

and 0 otherwise.

Theorem 7.2. *Let* p_1, p_2 *be two distinct primes and let* $m = p_1 p_2$. *Protocol 7.1 is a 2-server CDS protocol over* $\mathbb{F}_{p_1}$ *for* INDEX_N^2 *with message size* $h \cdot \log m$, *where* h *is the length of the matching vectors used in the protocol.*

Proof. First, denote $\mathbf{a}_1 = \sum_{j=1}^{N} C(\langle \mathbf{r}_1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j$, and $\mathbf{a}_2 = \sum_{j=1}^{N} C(\langle \mathbf{m}_B^1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j$. Then, $\mathbf{m}_A = \mathbf{a}_1 + \mathbf{r}_2$. Note, that $\mathbf{a}_1$ is the same expression as the answer of the first server in Protocol 5.4. Then, in Protocol 7.1, Charlie outputs 1 iff

$$\begin{aligned} \langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 - \mathbf{r}_2 \rangle_{p_1} + m_B^2 &\equiv \langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 - \mathbf{r}_2 \rangle_{p_1} + \langle \mathbf{u}_i, \mathbf{r}_2 \rangle_{p_1} \\ &\equiv \langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 \rangle_{p_1} \not\equiv 0 \pmod{p_1} \end{aligned}$$

Before proving the correctness and security we will show that for some $\alpha \not\equiv 0 \pmod{p_1}$

$$\langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 \rangle_{p_1} = s \cdot D_i \cdot \alpha. \tag{8}$$

This follows from the following analysis.

- If $s = 0$, then $\mathbf{m}_B^1 = \mathbf{r}_1$, and $\mathbf{a}_2 = \mathbf{a}_1$. In this case $\langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 \rangle_{p_1} = 0$.
- Otherwise, if $s = 1$, then $\mathbf{a}_2$ is equal to the answer of the second server in Protocol 5.4. Therefore from Equation (6) in the proof of Theorem 4.5

$$\langle \mathbf{u}_i, \mathbf{a}_2 - \mathbf{a}_1 \rangle_{p_1} = D_i \cdot \alpha$$

for $\alpha \not\equiv 0 \pmod{p_1}$.

Charlie computes $s \cdot D_i \cdot \alpha \bmod p_1$ for $\alpha \not\equiv 0 \pmod{p_1}$ and return 1 iff it is not equal to 0.

Now, we can continue to prove the correctness and security.

Correctness. Correctness should hold when $D_i = 1$. In this case by (8), Charlie outputs 1 if $s \cdot \alpha \bmod p_1$ is not equal to 0 for some $\alpha \not\equiv 0 \pmod{p_1}$, thus Charlie returns s.

Security. We prove that when $D_i = 0$, Charlie learns no information on the secret.

- The joint distribution of $\mathbf{m}_A, \mathbf{m}_B^1$ is uniformly distributed since they are masked by $\mathbf{r}_2, \mathbf{r}_1$ respectively.
- By (8),
$$\langle \mathbf{u}_i, \sum_{j=1}^{N} C(\langle \mathbf{m}_B^1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j - \mathbf{m}_A \rangle_{p_1} + m_B^2 = s \cdot D_i \cdot \alpha = 0.$$
Therefore, m_B^2 is independent of s and can be computed from $\mathbf{m}_A, \mathbf{m}_B^2, i, D$.

From the observations, when $D_i = 0$, Charlie can simulate $\mathbf{m}_A, \mathbf{m}_B^1, m_B^2$ given i, D. □

7.2 The Multi-Server CDS Protocol

In this section, we generalize the 2-server CDS protocol in Protocol 7.1 to a $(k+1)$-server CDS protocol (for $k \geq 2$), similarly to the generalization done by LVW18; this is done using decomposable matching vectors. We start with the definitions of point-wise product of vectors and decomposable matching vectors.

Definition 7.3 (Pointwise product)**.** *Let $m, h > 0$ be two positive integers and let $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_m^h$. The point-wise product (or Hadamard product) of $\mathbf{x}, \mathbf{y}$, denoted by $\mathbf{x} \odot \mathbf{y}$, is a vector in $\mathbb{Z}_m^h$ whose ℓ-th element is the product of the ℓ-th elements of $\mathbf{x}, \mathbf{y}$, i.e. $(\mathbf{x} \odot \mathbf{y})[\ell] = \mathbf{x}[\ell] \cdot \mathbf{y}[\ell] \mod m$.*

Definition 7.4 (k-decomposability [61])**.** *A family of vectors $(\mathbf{u}_i)_{i=1}^{N^k}$ over $\mathbb{Z}_m^h$ is k-decomposable if there exist vector families $(\mathbf{u}_{1,i})_{i=1}^{N}, \ldots, (\mathbf{u}_{k,i})_{i=1}^{N}$ over $\mathbb{Z}_m^h$ such that under the natural mapping $i \mapsto (i_1, \ldots, i_k) \in [N]^k$*
$$\mathbf{u}_i = \mathbf{u}_{1,i_1} \odot \cdots \odot \mathbf{u}_{k,i_k} \mod m$$
for all $i \in [N]^k$. That is, $\mathbf{u}_i$ is the pointwise product of k vectors $\mathbf{u}_{1,i_1}, \ldots, \mathbf{u}_{k,i_k}$, where each $\mathbf{u}_{j,i_j}$ can be computed from i_j.

Definition 7.5 (Decomposable matching vector families)**.** *For integers $N, m, h, k > 0$, $t \in \mathbb{Z}_m$ and $S \subseteq \mathbb{Z}_m \setminus \{t\}$, a collection of vectors $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^{N^k}$ over $\mathbb{Z}_m^h$ is a k-decomposable (S,t)-matching vector family if it is an (S,t)-matching vector family, and $(\mathbf{u}_i)_{i=1}^{N^k}$, $(\mathbf{v}_i)_{i=1}^{N^k}$ are k-decomposable.*

Next, we present a claim similar to Claim 3.10 showing the equivalence between the length of k-decomposable $(S_{\text{one},m}, 0)$-matching vectors to the length k-decomposable $S_{\text{zero},m}$-matching vectors.

Claim 7.6. *If there is a k-decomposable $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length h, then there is a k-decomposable $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length $h+1$. If there is an k-decomposable $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ of length h, then there is a k-decomposable $(S_{\text{one},m}, 0)$-matching vector family over $\mathbb{Z}_m$ of length $h+1$.*

Proof. Given $((\mathbf{u}_i, \mathbf{v}_i))_{i \in [N]^k}$ define $\mathbf{u}_i' = (1, -\mathbf{u}_i), \mathbf{v}_i' = (1, \mathbf{v}_i)$. We have seen in the proof of Claim 3.10 that if $((\mathbf{u}_i, \mathbf{v}_i))_{i \in [N]^k}$ is an $(S_{\text{one},m}, 0)$-matching vector family then $((\mathbf{u}_i', \mathbf{v}_i'))_{i \in [N]}$ is an $S_{\text{zero},m}$-matching vector family, and if $((\mathbf{u}_i, \mathbf{v}_i))_{i \in [N]}$ is an $S_{\text{zero},m}$-matching vector family then $((\mathbf{u}_i', \mathbf{v}_i'))_{i \in [N]^k}$ is an $(S_{\text{one},m}, 0)$-matching vector family. It is left to show that if $((\mathbf{u}_i, \mathbf{v}_i))_{i \in [N]^k}$ is k-decomposable matching vector family then $((\mathbf{u}_i', \mathbf{v}_i'))_{i \in [N]^k}$ is a k-decomposable matching vector family. Let $(\mathbf{u}_{1,i})_{i=1}^{N}, \ldots, (\mathbf{u}_{k,i})_{i=1}^{N}$ be the decomposition of $(\mathbf{u}_i)_{i \in [N]^k}$. For every $i_1 \in [N]$,

define $\mathbf{u}'_{1,i_1} = (1, -\mathbf{u}_{1,i_1})$, and for every $2 \leq t \leq k$, $i_t \in [N]$, define $\mathbf{u}'_{t,i_t} = (1, \mathbf{u}_{t,i_i})$, then $(\mathbf{u}'_{1,i})_{i=1}^{N}, \ldots, (\mathbf{u}'_{k,i})_{i=1}^{N}$ is a decomposition of $(\mathbf{u}'_i)_{i\in[N]}$. Since, for every $i \in [N]^k$

$$\mathbf{u}'_i[1] = 1 = \Pi_{t=1}^{k} \mathbf{u}'_{t,i_t}[1]$$

and for every $2 \leq \ell \leq h+1$,

$$\mathbf{u}'_i[\ell] = -\mathbf{u}_i[\ell-1] = -\Pi_{t=1}^{k} \mathbf{u}_{t,i_t}[\ell-1] = \Pi_{t=1}^{k} \mathbf{u}'_{t,i_t}[\ell].$$

□

Since $S_{\text{can},m} \subset S_{\text{one},m}$, a k-decomposable family of $(S_{\text{can},m}, 0)$-matching vectors is a k-decomposable family of $(S_{\text{one},m}, 0)$-matching vectors. Since LVW18 showed that the known results for $S_{\text{can},m}$ matching vectors [46] are k-decomposable, combining those results with Claim 7.6 imply the following corollary.

Corollary 7.7. *For every distinct primes p_1, p_2, there is a k-decomposable $S_{\text{zero},m}$-matching vector family over $\mathbb{Z}_m$ for $m = p_1 p_2$, of size N^k and length $2^{O(\sqrt{k \log N} \log(k \log N))}$.*

The multi-server CDS protocol that we present is a generalization of Protocol 7.1 to k servers. In the protocol, one server, we call Alice holds the database $D \in \{0,1\}^{N^{k-1}}$ and sends the exact same message as in Protocol 7.1. The other $k-1$ servers will jointly hold an index $i \in [N]^{k-1}$ i.e. the servers $S_2, \ldots, S_k$ will hold $i_2, \ldots, i_k \in [N]$ respectively. The $k-1$ parties will simulate Bob from Protocol 7.1 using a PSM protocol whose functionality is Bob's messages in the 2-server CDS protocol (for a formal definition of PSM see [48]).

In a PSM protocol there are $k-1$ parties each holding an input x_i and a common randomness. There is also a referee we call Charlie who wants to learn $F(x_1, \ldots, x_{k-1})$ for a fixed functionality F. Each server sends a message to Charlie based on their private input and the common randomness. Given these messages Charlie should be able to learn $F(x)$ without learning anything else about $x_1, \ldots, x_{k-1}$.

In the k-server CDS protocol each server $S_2, \ldots, S_k$ will send a message to Charlie according to the PSM, and Charlie would be able to reconstruct the messages of Bob from Protocol 7.1 as if he was holding the full index i, without learning any information about the secret. That special purpose PSM which is presented by LVW18 uses the $(k-1)$-decomposablity of the matching vectors.

Next we generalize the special purpose PSM protocol from LVW18 to any two distinct primes p_1, p_2.

Theorem 7.8 ([61])**.** *For integers N, h, $k \leq \log N$ and $m = p_1 p_2$ for distinct primes p_1, p_2, if $(\mathbf{u}_i)_{i=1}^{N^{k-1}}$ is $(k-1)$-decomposable, then there is a PSM for the functionality*

$$F_{\mathsf{aux}} : [N] \times \cdots \times [N] \times \{0,1\} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_1} \to \mathbb{F}_{p_2}^{h} \times \mathbb{F}_{p_1}$$

$$\textit{where } F_{\mathsf{aux}}(i_1, \ldots, i_k; s, \mathbf{r}_1, \mathbf{r}_2) \mapsto (s\mathbf{u}_i + \mathbf{r}_1, \langle \mathbf{u}_i, \mathbf{r}_2 \rangle_{p_1})$$

with communication complexity $O(h \cdot k^2 \log m)$.

Finally, we present our k-server CDS protocol.

Protocol 7.9.
Public parameters: *A decomposable $S_{\text{zero},m}$-matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^{N^{k-1}}$ over $\mathbb{Z}_m^h$ for $m = p_1p_2$ and a PSM protocol $(\mathsf{PSM.B}_2, \ldots, \mathsf{PSM.B}_k, \mathsf{PSM.C})$ for F_{aux}.*
Input of S_1 (Alice): $D \in \{0,1\}^{N^{k-1}}$.
Input of S_t $(2 \le t \le k)$: $i_t \in [N]$.
The secret: $s \in \{0,1\}$.
Shared randomness: $\mathbf{r}_1 \in \mathbb{F}_{p_2}^h$, $\mathbf{r}_2 \in \mathbb{F}_{p_1}^h$ *and randomness* $\mathbf{r}_{\mathsf{PSM}}$ *of the* PSM.
Notations: *Let $C : \mathbb{F}_{p_2} \to \mathbb{F}_{p_1}$ be a share conversion, and let $V : \mathbb{F}_{p_2}^h \to \mathbb{F}_{p_1}^h$ where $V(\mathbf{w}) = \sum_{j=1}^N C(\langle \mathbf{w}, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j \bmod p_1$.*

- *For $2 \le t \le k$, the t-th party sends $\mathbf{m}_{\mathsf{PSM},t} \leftarrow \mathsf{PSM.B}_t(i_t, s, \mathbf{r}_1, \mathbf{r}_2; \mathbf{r}_{\mathsf{PSM}})$*
- *Alice sends $\mathbf{m}_A \leftarrow \sum_{j=1}^N C(\langle \mathbf{r}_1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j + \mathbf{r}_2 \bmod p_1$.*
- *Charlie computes $(\mathbf{m}_B^1, m_B^2) \leftarrow \mathsf{PSM.C}(\mathbf{m}_{\mathsf{PSM},2}, \ldots, \mathbf{m}_{\mathsf{PSM},k})$.*
- *Charlie outputs 1 if*

$$\langle \mathbf{u}_i, \sum_{j=1}^N C(\langle \mathbf{m}_B^1, \mathbf{v}_j \rangle_{p_2}) \cdot D_j \mathbf{v}_j - \mathbf{m}_A \rangle_{p_1} + m_B^2 \not\equiv 0 \pmod{p_1}, \tag{9}$$

and 0 otherwise.

Theorem 7.10. *Let p_1, p_2 be two distinct primes and let $m = p_1p_2$. Protocol 7.9 is a k-server CDS protocol over $\mathbb{F}_{p_1}$ for INDEX_N^k with message size $O(h \cdot k^2 \log m)$, where h is the length of the matching vectors used in the protocol.*

Proof. The correctness and security follow from the correctness and security of Protocol 7.1 and of the PSM protocol for F_{aux} in Theorem 7.8. From the correctness of the PSM for F_{aux}, given $\mathbf{m}_{\mathsf{PSM},2}, \ldots, \mathbf{m}_{\mathsf{PSM},k}$, Charlie computes correctly $\mathbf{m}_B^1 \leftarrow s\mathbf{u}_i + \mathbf{r}_1, m_B^2 \leftarrow \langle \mathbf{u}_i, \mathbf{r}_2 \rangle_{p_1}$. Thus, from the proof of Theorem 7.2 the correctness follows. The security follows from the security of Protocol 7.1; the joint distribution $\mathbf{m}_A, \mathbf{m}_B^1, \mathbf{m}_B^2$ is equally distributed for $s = 0$ and $s = 1$. Finally, from the security of the PSM protocol, the joint distribution of $\mathbf{m}_{\mathsf{PSM},1}, \ldots, \mathbf{m}_{\mathsf{PSM},k}$ can be simulated from $(\mathbf{m}_B^1, m_B^2)$.

Communication. Alice sends a vector over $\mathbb{F}_{p_1}^h$ (of length $h \log p_1$), while all the other k servers send a PSM message, thus from Theorem 7.8 their message length is $O(h \cdot k^2 \log m)$. Thus the message size of the protocol is $(h \cdot k^2 \log m)$. □

8 Improved Linear Secret-Sharing Schemes for Arbitrary Access Structures

In this section, we show an improved construction of linear secret-sharing schemes for arbitrary access structures. We start with some preliminaries, followed by the main result of this section.

8.1 Preliminaries

8.1.1 Multislice and Downslice Access Structures

We next introduce the multislice and downslice of an access structure. They have found use in constructing secret-sharing schemes for general access structures [59, 7, 8, 10]. We first define multislices. Roughly, the $(a:b)$-multislice of an access structure Γ agrees with Γ on all sets of size between a and b, all sets of size less than a are unauthorized, and all sets of size greater than b are authorized. Formally, it is defined as follows.

Definition 8.1 (Multislices)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be an n-party access structure and let $a \leq b \in [n]$. The $(a:b)$-multislice of f is the access structure $F : \{0,1\}^n \to \{0,1\}$ defined as*

$$F(x) = \begin{cases} 0 & \text{if } \mathrm{wt}(x) < a \\ f(x) & \text{if } \mathrm{wt}(x) \in [a,b] \\ 1 & \text{if } \mathrm{wt}(x) > b \end{cases}.$$

An access structure is called an $(a:b,n)$-multislice access structure *if it is the $(a:b)$-multislice of some access structure over n parties. For $\alpha < \beta \in [0,1]$, we denote the linear exponent of $(\alpha n : \beta n, n)$-multislice access structure by*

$$\mathbf{M}_\ell(\alpha:\beta) = \limsup_{n\to\infty} \max_{F \in \mathcal{M}(\alpha:\beta,n)} \frac{1}{n} \log \mathsf{LSS}(F),$$

where $\mathcal{M}(\alpha:\beta,n)$ is the set of all $(\alpha n : \beta n, n)$-multislice access structures.

We next define a b-downslice of an access structure Γ. It is defined as the access structure that agrees with Γ on all sets of size b, and whose max-terms are all of size b. That is, a set is unauthorized if and only if it is contained in an unauthorized set of Γ of size b.

Definition 8.2 (Downslices)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be an n-party access structure and let $b \in [n]$. The b-downslice of f is the access structure $F : \{0,1\}^n \to \{0,1\}$ defined as*

$$F(x) = \begin{cases} 0 & \text{if } \exists x' : \mathrm{wt}(x') = b, x \leq x', f(x') = 0 \\ 1 & \text{otherwise} \end{cases}.$$

An access structure is called a (b,n)-downslice access structure *if it is the b-downslice of some access structure over n parties. For $\beta \in [0,1]$, we denote the linear exponent of $(\beta n, n)$-downslice access structure by*

$$\mathbf{D}_\ell(\beta) = \limsup_{n\to\infty} \max_{F \in \mathcal{D}(\beta,n)} \frac{1}{n} \log \mathsf{LSS}(F),$$

where $\mathcal{D}(\beta,n)$ is the set of all $(\beta n, n)$-downslice access structures.

8.1.2 Robust Conditional Disclosure of Secrets

Observe that in the security definition of a CDS protocol, if a server sends several messages for different inputs with the same randomness, then it is not guaranteed that the referee does not learn anything about the secret. Toward constructing better secret-sharing schemes, Applebaum et al. [8] introduced the notion of robust CDS (RCDS) protocol, where the security is guaranteed to

hold even if the servers send several messages for different inputs that do not satisfy the predicate. Before formally defining RCDS protocols, we first define the notion of a zero set, which is simply a subset of inputs that are mapped to 0.

Definition 8.3 (Zero sets)**.** *Let $f : X \times Y \to \{0,1\}$ be a function. We say that a set $Z \subseteq X \times Y$ is a* zero set of f *if $f(x,y) = 0$ for all $(x,y) \in Z$.*

Definition 8.4 ($\mathcal{Z}$-robust CDS protocols)**.** *Let $\mathcal{P}$ be a CDS protocol for a function $f : X \times Y \to \{0,1\}$, and let $Z = Z_1 \times Z_2 \subseteq X \times Y$ be a zero set of f. Let $\mathsf{A}(Z_1, s; r) = (A(x, s; r))_{x \in Z_1}$ and $\mathsf{B}(Z_2, s; r) = (\mathsf{B}(y, s; r))_{y \in Z_2}$. We say that $\mathcal{P}$ is robust for the set Z if for every pair of secrets $s_1, s_2 \in S$ and for $r \leftarrow R$ it holds that*

$$(\mathsf{A}(Z_1, s_1; r), \mathsf{B}(Z_2, s_1; r)) \equiv (\mathsf{A}(Z_1, s_1; r), \mathsf{B}(Z_2, s_2; r)).$$

Let $\mathcal{Z}_1 \subseteq 2^X, \mathcal{Z}_2 \subseteq 2^Y$. We say that $\mathcal{P}$ is $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust if it is robust for every zero set $Z = Z_1 \times Z_2$ such that $Z_1 \in \mathcal{Z}_1, Z_2 \in \mathcal{Z}_2$. Furthermore, for an integer t_1 we say that $\mathcal{P}$ is $(\mathcal{Z}_1, t_2)$-robust if it is $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust for $\mathcal{Z}_2 = \{A \subseteq\in Y : |A| \leq t_2\}$; (t_1, t_2)-robustness is defined similarly. Finally, for an integer t, we say that $\mathcal{P}$ is t-robust if it is (t, t)-robust.

8.2 Statement of Our Result

In this section, we prove new upper bounds on the share size of linear secret-sharing schemes for all access structures.

Theorem 8.5. *Any n-party access structure can be realized with a linear secret-sharing scheme by share size $2^{0.7563n+o(n)}$.*

To simplify notations, if a scheme has share size $2^{S \cdot n + o(n)}$, we will say that it has exponent S. As observed in [10], it is enough to prove the bound for all downslices. Recall that a downslice is an access structure where all maximal authorized sets are of the same size. Applebaum and Nir [10]constructed a scheme for a downslice by a bootstrapping algorithm, taking a linear secret-sharing scheme for a downslice and constructing a better scheme. Our improvement is done by constructing a better scheme for downslices than [10], before applying bootstrapping. The next result states the share size for our basis scheme. In Section 8.5we apply the bootstrapping, which proves Theorem 8.5.

We say that that $\mathbf{D}_\ell(\beta) < \lambda$ means that every n-party $(\beta n, n)$-downslice can be realized by a secret-sharing scheme with share size $\leq 2^{(\lambda + o(1))n}$. as formally defined in Definition 8.2

Lemma 8.6. *Let $\alpha_0 > 0.5412$ denote the unique solution to the equation $(2x-3)^2 \log(x) + 2x^2 - 8x + 7 = 0$ in the interval $[0,1]$. Then for all $\beta \in (0,1)$,*

$$\mathbf{D}_\ell(\beta) \leq \begin{cases} \frac{1}{2} + \frac{\beta}{2} & \text{if } \beta \leq 1/2 \\ \frac{2-\beta}{3-2\beta} & \text{if } 1/2 < \beta \leq \alpha_0 \\ h(\beta) - 0.51079 \cdot (1-\beta) & \text{otherwise} \end{cases}.$$

For example, we get $\mathbf{D}_\ell(0.54) \leq 146/192 \approx 0.7604$. On the other hand, before applying the bootstrapping, Applebaum and Nir [10] proved the upper bound of $\mathbf{D}_\ell(0.54) \leq h(0.54) - 0.5 \cdot (1 - 0.54) \approx 0.7654$.

The bound for $\beta \leq 1/2$ was shown in [10]. The proof for the case where $1/2 < \beta \leq \alpha_0$ is given in Lemma 8.7, and the proof for the case where $\alpha_0 < \beta \leq 1$ is given in Lemma 8.18.

8.3 A Better Linear Secret-Sharing Scheme for Downslices With Low Density

We next show an improved linear secret-sharing scheme for all downslices with low density. That is, we prove the following.

Lemma 8.7 (Low density)**.** *Every (b,n)-downslice access structure can be realized by a linear secret-sharing scheme with share size of at most $2^{\frac{n(2n-b)}{3n-2b}+o(n)}$. Consequently, $\mathbf{D}_\ell(\beta) \leq \frac{2-\beta}{3-2\beta}$ for any $\beta \in [0,1]$.*

Towards constructing such a scheme, we first construct a new linear secret-sharing scheme for all multislices. For this, we use the following 2-server linear CDS for an arbitrary function; this CDS protocol is already fully robust for Bob.

Theorem 8.8 ([43, 8])**.** *Let $f : [N_1] \times [N_2] \to \{0,1\}$. Then there exists a linear $(1, N_2)$-robust CDS protocol for f, where the message size of Alice is $N_1 - 1$, and the message of Bob is a single bit.*

By the immunization theorem of [8] we obtain the following.

Corollary 8.9. *Let $f : [N_1] \times [N_2] \to \{0,1\}$ and let $t \leq N_1$ and $\mathcal{Z}_1 \subseteq 2^{[N_1]}$ such that $|Z| \leq t$ for every $Z \in \mathcal{Z}_1$ and $\mathcal{Z}$ contains at most u maximal sets. Then there exists a linear $(\mathcal{Z}_1, N_2)$-robust CDS protocol for f, where the message size of Alice is $O(\frac{N_1}{t} \log^3 t \log N_1 \log u)$ and the message size of Bob is $O(t \log^3 t \log N_2 \log u)$.*

Our construction of a linear secret-sharing scheme for multislice access structures follows a similar approach to previous constructions [8, 10]. That is, we first reduce the construction to realizing somewhat regular access structures. These are partial access structures where we partition the parties into two sets and every authorized and every unauthorized set are partitioned "nicely" between the two sets. Unlike previous constructions, the partition we consider does not split each set into roughly equal sizes. This uneven partition is the main source for the improved share size in our construction. We next formally define somewhat regular access structures.

Definition 8.10 ((I,a,b)-somewhat regular access structures)**.** *Let $1 \leq a \leq b \leq n$ and let $I \subset P$ of size $|I| = \mu n$. A (partial) access structure $\Gamma = (\Gamma_{\text{yes}}, \Gamma_{\text{no}})$ over n parties is called (I,a,b)-somewhat regular if for every $A \in \Gamma_{\text{yes}} \cup \Gamma_{\text{no}}$ it holds that $\mu a \leq |A \cap I| \leq \mu b$ and $(1-\mu)a \leq |A \cap (P \setminus I)| \leq (1-\mu)b$.*

Note that previous papers considered the case where $\mu = 1/2$. We next construct a linear secret-sharing scheme for any (I,a,b)-somewhat regular access structure using a robust CDS protocol.

Lemma 8.11. *Let $\Gamma = (\Gamma_{\text{yes}}, \Gamma_{\text{no}})$ be an (I,a,b)-somewhat regular access structure and let $\mu = |I|/n$, $t_1 = \sum_{j=\mu a}^{\mu b} \binom{\mu b}{j}$, and $t_2 = \sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)b}{j}$. Assume that for all functions $f : \{0,1\}^{\mu n} \times \{0,1\}^{(1-\mu)n} \to \{0,1\}$ and every $\mathcal{Z}_1 \subseteq \{0,1\}^{\mu n}, \mathcal{Z}_2 \subseteq \{0,1\}^{(1-\mu)n}$ such that every set in $\mathcal{Z}_1$ and $\mathcal{Z}_2$ has size at most t_1 and t_2 respectively and $\mathcal{Z}_1$ and $\mathcal{Z}_2$ contain at most $\binom{\mu n}{\mu b}$ and $\binom{(1-\mu)n}{(1-\mu)b}$ maximal sets respectively there exists a 2-server $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust linear CDS protocol, such that the message sizes of Alice and Bob are s_{A} and s_{B} respectively.*

Then there exists a linear secret-sharing scheme realizing Γ such that the share size of every party $p_i \in I$ is less than $s_{\mathsf{A}} \cdot \sum_{j=\mu a}^{\mu b} \binom{\mu n}{j}$ and the share size of every party $p_i \in P \setminus I$ is less than $s_{\mathsf{B}} \cdot \sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)n}{j}$.

Proof. In the following, we let $X = \{0,1\}^{\mu n}$, let $Y = \{0,1\}^{(1-\mu)n}$, and let $I = \{p_{i_1} < \cdots < p_{i_{\mu n}}\}$ and $P \setminus I = \{p_{i_{\mu n+1}} < \cdots < p_{i_n}\}$. For a string $x \in X$ we denote $A^1_x = \left\{p_{i_j} : j \in [\mu n], x_j = 1\right\}$, and for a string $y \in Y$ we denote $A^2_y = \left\{p_{i_{j+\mu n}} : j \in [(1-\mu)n], y_j = 1\right\}$. We next present the construction.

Scheme 8.12.
Input: *The dealer holds a secret $s \in \{0,1\}$.*

1. *Let $f : X \times Y \to \{0,1\}$ be some monotone function that agrees with Γ, that is, for every $x \in X$ and $y \in Y$ such that $A^1_x \cup A^2_y \in \Gamma_{\text{yes}} \cup \Gamma_{\text{no}}$, it holds that $f(x,y) = 1$ if and only if $A^1_x \cup A^2_y \in \Gamma_{\text{yes}}$,[7] and let*

$$\mathcal{Z}_1 = \{\{x^* \leq x : \text{wt}(x^*) \geq \mu a\} : \text{wt}(x) \leq \mu b\}$$

and

$$\mathcal{Z}_2 = \{\{y^* \leq y : \text{wt}(^*) \geq (1-\mu)a\} : \text{wt}(y) \leq (1-\mu)b\}\,.$$

2. *Sample a random string r for a 2-server $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust linear CDS protocol $\mathcal{P} = (\mathsf{A}, \mathsf{B})$ for f.*

3. *For every $x \in X$ whose Hamming weight w is in the interval $[\mu a, \mu b]$, compute $\mathsf{sh}_{\mathsf{A}}(x,s) = \mathsf{A}(x,s;r)$ and share it between the parties in A^1_x (i.e., the parties that correspond to the string x) using a w-out-of-w additive secret-sharing scheme. We denote the share of $p_i \in A^1_x$ by $\mathsf{sh}_{\mathsf{A}}(x,s,i)$.*

4. *For every $y \in Y$ whose Hamming weight w is in the interval $[(1-\mu)a, (1-\mu)b]$, compute $\mathsf{sh}_{\mathsf{B}}(y,s) = \mathsf{B}(y,s;r)$ and share it between the parties in A^2_y using a w-out-of-w additive secret-sharing scheme. We denote the share of $p_i \in A^2_y$ by $\mathsf{sh}_{\mathsf{B}}(y,s,i)$.*

5. *The share of $p_i \in I$ is $(\mathsf{sh}_{\mathsf{A}}(x,s,i))_{x \in X, p_i \in A^1_x}$ and the share of $p_i \in P \setminus I$ is $(\mathsf{sh}_{\mathsf{B}}(y,s,i))_{y \in Y, p_i \in A^2_y}$.*

We first show that Scheme 8.12 is correct. Fix $A \in \Gamma_{\text{yes}}$. Since Γ is (I,a,b)-somewhat regular, $\mu a \leq |A \cap I| \leq \mu b$ and $(1-\mu)a \leq |A \cap (P \setminus I)| \leq (1-\mu)b$. Therefore, for the strings $x \in X$ and $y \in Y$ such that $A^1_x = A \cap I$ and $A^2_y = A \cap (P \setminus I)$, it holds that $f(x,y) = 1$ and that their Hamming weights are in the intervals $[\mu a, \mu b]$ and $[(1-\mu)a, (1-\mu)b]$, respectively. Hence, the parties in A^1_x can reconstruct $\mathsf{sh}_{\mathsf{A}}(x,s)$ and the parties in A^2_y can reconstruct $\mathsf{sh}_{\mathsf{B}}(y,s)$. By the correctness of the $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust CDS protocol, they can reconstruct the secret s.

We next show privacy. Fix $A \in \Gamma_{\text{no}}$, let $x \in X$ denote the string such that $A^1_x = A \cap I$, and let $y \in Y$ denote the string such that $A^2_y = A \cap (P \setminus I)$. Consider two secrets s_0 and s_1. To show privacy, we prove that the corresponding shares of the parties in A, denoted $\mathcal{D}(s_0)$ and $\mathcal{D}(s_1)$ respectively, are identically distributed. Since for every $x' \not\leq x$ and every $y' \not\leq y$, the parties in $A^1_x \cup A^2_y$ miss at least one share of $\mathsf{sh}_{\mathsf{A}}(x',s)$ and one share of $\mathsf{sh}_{\mathsf{B}}(y',s)$, by the security of the additive secret-sharing scheme, the shares $\mathsf{sh}_{\mathsf{A}}(x',s)$ and $\mathsf{sh}_{\mathsf{B}}(y',s)$ held by $A^1_x \cup A^2_y$ are uniformly distributed and independent of the secret and the other shares and can be ignored.

[7] Since $\Gamma_{\text{yes}} \cup \Gamma_{\text{no}}$ is a partial access structure, such monotone f exists.

The remaining shares are

$$\left\{\mathsf{sh}_{\mathsf{A}}(x^*, s, j) : x^* \leq x, p_j \in A^1_{x^*}\right\} \cup \left\{\mathsf{sh}_{\mathsf{B}}(y^*, s, j) : y^* \leq y, p_j \in A^2_{y^*}\right\},$$

i.e., the shares of the messages corresponding to the inputs

$$Z_1 = \{x^* \leq x : \mathrm{wt}(x^*) \geq \mu a\} \text{ and } Z_2 = \{y^* \leq y : \mathrm{wt}(y^*) \geq (1-\mu)a\}.$$

Since f is monotone and $f(x, y) = 0$, the set $Z_1 \times Z_2$ is a zero set. By the robustness of the $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust CDS scheme, it follows that $\mathcal{D}(s_0) \equiv \mathcal{D}(s_1)$.

Finally, we analyze the share size. The set $\mathcal{Z}_1$ contains $\binom{\mu n}{\mu b}$ maximal sets (one such set for each string of weight μb) and each $Z \in \mathcal{Z}_1$ contains at most $t_1 = \sum_{j=\mu a}^{\mu b} \binom{\mu b}{j}$ inputs (one input for every string $x^* \leq x$ such that $\mathrm{wt}(x^*) \geq \mu a$ and $\mathrm{wt}(x) \leq \mu b$). Similarly, the set $\mathcal{Z}_2$ contains $\binom{(1-\mu)n}{(1-\mu)b}$ maximal sets and each $Z \in \mathcal{Z}_2$ contains at most $t_2 = \sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)b}{j}$ inputs. Thus, the message sizes of Alice and Bob in the $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust CDS protocol are s_{A} and s_{B} respectively. Every party $p_i \in I$ receives a share of Alice's message in the $(\mathcal{Z}_1, \mathcal{Z}_2)$-robust CDS protocol for every $x \in X$ of Hamming weight in $[\mu a, \mu b]$, where $p_i \in A^1_x$. Since there are $\sum_{j=\mu a}^{\mu b} \binom{\mu n - 1}{j-1}$ such strings $x \in X$ where $p_i \in A^1_x$, its share size is less than $s_{\mathsf{A}} \cdot \sum_{j=\mu a}^{\mu b} \binom{\mu n}{j}$. Similarly, the share size of every party $p_i \in P \setminus I$ is less than $s_{\mathsf{B}} \cdot \sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)n}{j}$. □

We next construct a linear secret-sharing scheme for a multislice access structure using linear secret-sharing schemes for somewhat regular access structures. Fix $a, b \in [n^{0.8}, n - n^{0.8}]$ such that $a < b$ (we deal with the general case below) and let $\varepsilon = n^{-0.2}$ be a proximity parameter. Let $I \subseteq P$ and let $\mu = |I|/n$ and let $A \subseteq P$. We say that A is *good* for I if

$$\mu a - \varepsilon \leq |A \cap I| \leq \mu b + \varepsilon \quad \text{and} \quad (1-\mu)a - \varepsilon \leq |A \cap (P \setminus I)| \leq (1-\mu)b + \varepsilon.$$

If A is not good, then we call it bad. We will use the following lemma.

Lemma 8.13. *For all $a, b \in [n^{0.8}, n - n^{0.8}]$ such that $a < b$ and every constant $\mu \in (0, 1)$, there exists a collection of $\lambda = O(n)$ subsets $I_1, \ldots, I_\lambda \subseteq P$, each of size μn, such that for all $A \subseteq P$ satisfying $a \leq |A| \leq b$, the set A is good for at least 0.7λ of the subsets.*

Proof. We prove the lemma using the probabilistic method. We sample a collection of $\lambda = O(n)$ subsets, each of size μn, uniformly at random, and show that with positive probability all inputs are good for at least 0.7λ of the subsets.

Let us first analyze this probability with respect to a single subset I sampled uniformly at random from all subsets of P of size μn. Fix $A \subseteq P$ such that $a \leq |A| \leq b$. We next show that

$$\mu a - \varepsilon \leq |A \cap I| \leq \mu b + \varepsilon \quad \text{and} \quad (1-\mu)a - \varepsilon \leq |A \cap (P \setminus I)| \leq (1-\mu)b + \varepsilon$$

with overwhelming probability. We only prove the former as the latter can be proved using an analogous argument.

For every $i \in A$ let X_i denote the indicator for the event $i \in A \cap I$. Then these random variables are negatively associated (see Claim A.4). We denote by $X = \sum_{i \in A} X_i$ the random variable that

is equal to $|A \cap I|$. By linearity of expectation, $\mathrm{E}[X] = \mu|A|$, hence $\mu a \leq \mathrm{E}[X] \leq \mu b$. Now, since $\varepsilon = n^{-0.2} < \mu b$, by Chernoff's inequality (see Theorem A.2),

$$\begin{aligned}\Pr[X > \mu b + \varepsilon] &\leq \Pr[X > \mu|A| + \varepsilon] = \Pr\left[X > \mu|A| \cdot \left(1 + \varepsilon \cdot \frac{1}{\mu|A|}\right)\right] \\ &\leq e^{-\varepsilon^2 \cdot \frac{\mu|A|}{3}} \leq e^{-n^{-0.4} \cdot \frac{\mu n^{0.8}}{3}} = e^{-\frac{\mu n^{0.4}}{3}}.\end{aligned}$$

Similarly,

$$\Pr[X < \mu a - \varepsilon] \leq e^{-\frac{\mu n^{0.4}}{3}}.$$

Therefore, by the union bound,

$$\mu a - \varepsilon \leq |A \cap I| \leq \mu b + \varepsilon \quad \text{and} \quad (1-\mu)a - \varepsilon \leq |A \cap (P \setminus I)| \leq (1-\mu)b + \varepsilon$$

except with probability $o(1)$.

Finally, if we independently sample λ subsets, the probability that A is bad for at least 0.3λ of them is, by a Chernoff bound, at most $2^{-\Omega(\lambda)}$. By taking $\lambda = Cn$ for a sufficiently large constant C, the latter probability is smaller than 2^{-n}. Applying the union bound over all possible subsets A, the probability that every A is good for at least 0.7λ of the sets is strictly greater than 0. Therefore, there exists λ subsets such that A is good for at least 0.7λ of them. □

We can now realize $(a : b, n)$-multislice access structures. We first realize them for all $n^{0.8n} \leq a < b \leq n - n^{0.8}$.

Lemma 8.14. *Let $n^{0.8} \leq a < b \leq n - n^{0.8}$ and let Γ be an $(a : b, n)$-multislice access structure. Let $c = a - n^{-0.2}$, let $d = b + n^{-0.2}$, and let $\mu \in (0, 1)$. Assume that for all $I \subset P$, where $|I| = \mu n$, any (I, c, d)-somewhat regular access structure can be realized with a linear secret-sharing scheme where the share size is at most m. Then Γ can be realized with a linear secret-sharing scheme with share size at most $O(mn \log n)$.*

Proof. We start by considering an $(a : b, n)$ partial multislice access structure, namely, the access structure is defined only over the inputs whose Hamming weight is in the interval $[a, b]$. The scheme is as follows.

1. Let $I_1, \ldots, I_\lambda$ be the collection of $\lambda = O(n)$ subsets of P guaranteed by Lemma 8.13.
2. Share the secret s using a $\lambda/2$-out-of-λ Shamir's secret-sharing scheme. Let $\sigma_1, \ldots, \sigma_\lambda$ denote the shares.
3. For every $j \in [\lambda]$, share σ_j with fresh randomness using the secret-sharing scheme realizing the (I_j, c, d)-somewhat regular access structure $\Gamma_{I_j,c,d}$ that agrees with Γ.

We now analyze the construction. We start with showing correctness. Let $A \in \Gamma_{\mathrm{yes}}$. By Lemma 8.13, the set $J = \{j : A \text{ is good for } I_j\}$ is of size $|J| \geq 0.7\lambda$. Therefore, at least 0.7λ of the shares σ_j can be reconstructed by the parties in A. Thus, they can reconstruct s.

We next show privacy. Let $A \in \Gamma_{\mathrm{no}}$. Since A is good for at least 0.7λ of the subsets, by the privacy of each $\Gamma_{I_j,c,d}$, it holds that at least 0.7λ of the shares σ_j remain perfectly hidden. Therefore, the secret s remains perfectly hidden.

As for the share size, first, note that each σ_j is of size $O(\log \lambda)$. Therefore, the share size in the above scheme is at most $O(m\lambda \log \lambda) = O(mn \log n)$.

We now handle the fully defined access structure, that is, we need to consider the "big" and "small" sets. Recall that for all $A \subseteq P$ where $|A| < a$ it holds that $A \notin \Gamma$, and for all $B \subseteq P$ where $|B| > b$ it holds that $B \in \Gamma$. The scheme is as follows.

1. Share s using a $(b+1)$-out-of-n Shamir's secret-sharing scheme, and give the i^{th} share to p_i.
2. Share s into s_0 and s_1 using a 2-out-of-2 secret-sharing scheme.
3. Share s_0 using an a-out-of-n Shamir's secret-sharing scheme and give the i^{th} share to the p_i.
4. Share s_1 using the secret-sharing scheme for the partial access structure shown above, and give the i^{th} share to the p_i.

For correctness, note that if $A \in \Gamma$ is of size at least $b+1$ then the parties can reconstruct s using the Shamir shares of the first scheme. Otherwise, it must be the case where $a \leq |A| \leq b$, hence the parties can reconstruct s_0 and s_1, and thus can reconstruct s.

For privacy, observe that if $A \notin \Gamma$ is of size $|A| < a$, the parties cannot reconstruct s_0, hence s is perfectly hidden. Otherwise, it must be the case where $a \leq |A| \leq b$, hence the Shamir shares of Step 1 reveal no information to the parties, and they cannot reconstruct s_1 due to the privacy of the underlying scheme.

Finally, note that the share size is only $O(\log n)$ additively longer than the share size for the partial access structure. Thus, the share is at most $O(mn \log n)$. □

We now put everything together and construct a secret-sharing scheme for all multislice access structures. As a corollary to this theorem, we obtain Lemma 8.7. Note that in the bootstrapping step, we need to realize multislices(see Lemma 8.19). We note that the result is not optimal for all parameters, and for some parameters better upper bounds can be obtained by either using the cover lemma (see Lemma 8.18 below) or the results of [10]. We use the following results due to Liu and Vaikuntanathan [59], which decomposes an access structure.

Proposition 8.15 ([59])**.** *Let Γ be an access structure and let $a < b \in [n]$. Define three access structures $\Gamma_{\text{bot}}(a)$, $\Gamma_{\text{mid}}(a,b)$, and $\Gamma_{\text{top}}(b)$ as follows*

$$\begin{aligned}
&\Gamma_{\text{bot}}(a) : A \in \Gamma_{\text{bot}}(a) \iff \exists A' \in \Gamma \text{ s.t. } A' \subseteq A \text{ and } |A'| < a \\
&\Gamma_{\text{mid}}(a,b) : A \in \Gamma_{\text{mid}}(a,b) \iff A \in \Gamma \text{ and } a \leq |A| \leq b, \text{ or } |A| > b \\
&\Gamma_{\text{top}}(b) : A \notin \Gamma_{\text{top}}(b) \iff \exists A' \notin \Gamma \text{ s.t. } A \subseteq A' \text{ and } |A'| > b.
\end{aligned}$$

Then $\Gamma = (\Gamma_{\text{bot}}(a) \cup \Gamma_{\text{mid}}(a,b)) \cap \Gamma_{\text{top}}(b)$. Consequently, if $\Gamma_{\text{bot}}(a)$, $\Gamma_{\text{mid}}(a,b)$, and $\Gamma_{\text{top}}(b)$ can be realized with linear secret-sharing schemes with exponent S, then so is Γ.

Theorem 8.16 (Multislice theorem)**.** *Let $1 \leq a < b \leq n$. Then every $(a : b, n)$-multislice access structure can be realized by a linear secret-sharing scheme with share size at most $2^{\frac{n(2n-h(a/b)\cdot b)}{3n-2h(a/b)\cdot b}+o(n)}$ if $a > b/2$, and share size at most $2^{\frac{n(2n-b)}{3n-2b}+o(n)}$ otherwise. Consequently, if $a = \alpha n$ and $b = \beta n$, then the exponent $\mathbf{M}_\ell(\alpha : \beta)$ satisfies.*

$$\mathbf{M}_\ell(\alpha : \beta) \leq \begin{cases} \frac{2-h(\alpha/\beta)\cdot\beta}{3-2h(\alpha/\beta)\cdot\beta} & \text{if } \alpha > \beta/2 \\ \frac{2-\beta}{3-2\beta} & \text{otherwise.} \end{cases}$$

Proof. Let Γ be an $(a : b, n)$-multislice access structure. Let $a' = \max\{a, n^{0.8}\}$ and let $b' = \min\{b, n - n^{0.8}\}$. Consider the three access structures $\Gamma_{\text{bot}}(a')$, $\Gamma_{\text{mid}}(a', b')$, and $\Gamma_{\text{top}}(b')$. Then by Proposition 8.15 we can write $\Gamma = (\Gamma_{\text{bot}}(a') \cup \Gamma_{\text{mid}}(a', b')) \cap \Gamma_{\text{top}}(b')$. We realize $\Gamma_{\text{bot}}(a')$ using a DNF scheme, and realize $\Gamma_{\text{top}}(b')$ using a CNF scheme, respectively. The cost of these schemes is at most $n\binom{n}{n^{0.8}} = 2^{o(n)}$.

It is left to realize $\Gamma_{\text{mid}}(a', b')$. Observe that $\Gamma_{\text{mid}}(a', b')$ is an $(a' : b', n)$-multislice access structure where $n^{0.8} \leq a' < b' \leq n - n^{0.8}$. We realize Γ_{mid} using the scheme given by Lemma 8.14, and using the scheme from Lemma 8.11 to realize the somewhat regular access structures. The total share size is $s \cdot O(n \log n)$, where

$$s = \max\left\{ s_{\mathsf{A}} \cdot \sum_{j=\mu a}^{\mu b} \binom{\mu n}{j},\ s_{\mathsf{B}} \cdot \sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)n}{j} \right\},$$

where s_{A} and s_{B} are the message sizes of Alice and Bob in the $(\mathcal{Z}_1, 2^{(1-\mu)n})$-robust linear CDS protocol guaranteed by Corollary 8.9, where $t = \sum_{j=\mu a}^{\mu b} \binom{\mu b}{j}$ and $u = \binom{\mu n}{\mu b}$. Then $s_{\mathsf{A}} = O(\frac{2^{\mu n}}{t} \log^3 t \cdot \mu n \cdot \log \binom{\mu n}{\mu b})$, and $s_{\mathsf{B}} = O(t \log^3 t \cdot (1-\mu)n \cdot \log \binom{\mu n}{\mu b})$. Since $\sum_{j=\mu a}^{\mu b} \binom{\mu n}{j} < 2^{\mu n}$ and $\sum_{j=(1-\mu)a}^{(1-\mu)b} \binom{(1-\mu)n}{j} < 2^{(1-\mu)n}$, the share size is

$$s = \max\left\{ O\left(\frac{2^{\mu n}}{t} \log^3 t \cdot \mu n \cdot \log \binom{\mu n}{\mu b}\right) \cdot 2^{\mu n},\right.$$
$$\left. O\left(t \log^3 t \cdot (1-\mu)n \cdot \log \binom{\mu n}{\mu b}\right) \cdot 2^{(1-\mu)n} \right\},$$

In order to optimize the share complexity (up to polynomial factors in n), we choose μ such that

$$\frac{2^{2\mu n}}{t} = t \cdot 2^{(1-\mu)n}. \tag{10}$$

Observe that

$$t = \begin{cases} O(2^{h(a/b)\cdot \mu b}) & \text{if } a > b/2 \\ O(2^{\mu b}) & \text{otherwise.} \end{cases}$$

Therefore, to ensure that (10) holds we take

$$\mu = \begin{cases} \frac{n}{3n - 2h(a/b)\cdot b} & \text{if } a > b/2 \\ \frac{n}{3n-2b} & \text{otherwise.} \end{cases}$$

Observe that in both cases it holds that $\mu \in (0, 1)$. Thus, if $a > b/2$ then, up to polynomial factors, the share size is $\frac{2^{2\mu n}}{t} = 2^{2\mu n - h(a/b)\cdot \mu b} = 2^{\frac{n(2n - h(a/b)\cdot b)}{3n - 2h(a/b)\cdot b}}$, and if $a \leq b/2$ then, up to polynomial factors, the share size is $\frac{2^{2\mu n}}{t} = 2^{2\mu n - \mu b} = 2^{\frac{n(2n-b)}{3n-2b}}$. □

As a corollary, we obtain Lemma 8.7.

Proof of Lemma 8.7. Observe that for every $b \in [n]$, every $(0 : b, n)$-multislice access structure is also a (b, n)-downslice access structure. Therefore, any (b, n)-downslice can be realized with a linear secret-sharing scheme with share size at most $2^{\frac{n(2n-b)}{3n-2b} + o(n)}$. □

8.4 Constructing a Linear Secret-Sharing for Downslices With High Density

Similarly to [10], for downslices with high density, we reduce this to the case of low density via the covering lemma.

Lemma 8.17 (Cover reduction lemma [7, 10])**.** *Let $a < b \leq n$ be positive integers. If $(b-a, n-a)$-downslices can be linearly realized with share size $z(b-a, n-a)$ then (b, n)-downslices can be linearly realized with share size of*

$$\left[\binom{n}{n-b} / \binom{n-a}{n-b}\right] \cdot \left[1 + \log \binom{n-a}{n-b}\right] \cdot z(b-a, n-a).$$

Consequently, for all constants $0 \leq \alpha < \beta \leq 1$ it holds that if $(\alpha m, m)$-downslices can be linearly realized with exponent $\tilde{z}(\alpha)$, then $(\beta n, n)$-downslices can be linearly realized with exponent at most

$$h(\beta) - (1-\beta) \cdot \frac{h(\alpha) - \tilde{z}(\alpha)}{1-\alpha}.$$

Combined with Lemma 8.7, we obtain the following result for downslices with high density. We note that in [10], the reduction of a downlice with density β was to a downslice with density $1/2$. Since we improve the results for slices with low density, we reduce to a different value.

Lemma 8.18 (High density)**.** *Let $\alpha_0 > 0.5412$ denote the unique solution to the equation $(2x-3)^2 \log(x) + 2x^2 - 8x + 7 = 0$ in the interval $[0, 1]$. For every integers n and $b \in (\alpha_0 n, n]$, every (b, n)-downslice can be linearly realized with share size at most*

$$\left[\binom{n}{n-b} / \binom{\frac{n-b}{1-\alpha_0}}{n-b}\right] \cdot \left[1 + \log \binom{\frac{n-b}{1-\alpha_0}}{n-b}\right] \cdot 2^{\frac{2-\alpha_0}{(3-2\alpha_0)(1-\alpha_0)} \cdot (n-b)}.$$

Consequently, for all $\beta \in (\alpha_0, 1]$,

$$\mathbf{D}_\ell(\beta) \leq h(\beta) - (1-\beta) \cdot \frac{h(\alpha_0) - \frac{2-\alpha_0}{3-2\alpha_0}}{1-\alpha_0} < h(\beta) - 0.51079 \cdot (1-\beta).$$

Proof. By Lemmas 8.7 and 8.17, for every $a \in (0, b)$, every (b, n)-downslices can be linearly realized with share size of

$$\begin{aligned}
&\left[\binom{n}{n-b} / \binom{n-a}{n-b}\right] \cdot \left[1 + \log \binom{n-a}{n-b}\right] \cdot 2^{\frac{2n-b-a}{3n-2b-a} \cdot (n-a) + o(n)} \\
&\qquad \leq 2^{h(b) \cdot n - h\left(\frac{n-b}{n-a}\right) \cdot (n-a) + \frac{2n-b-a}{3n-2b-a} \cdot (n-a) + o(n)} \\
&\qquad = 2^{h(b) \cdot n - \left(h\left(\frac{n-b}{n-a}\right) - \frac{2n-b-a}{3n-2b-a}\right) \cdot (n-a) + o(n)}.
\end{aligned}$$

Let $a' = \frac{b-a}{n-a} \cdot n \in (0, b)$ and let $\alpha' = a'/n$. Then $a = \frac{b-a'}{n-a'} \cdot n$ and $n - a = \frac{n-b}{n-a'} \cdot n$, hence the share size is at most

$$\begin{aligned}
2^{h(b) \cdot n - \left(h\left(\frac{n-b}{n-a}\right) - \frac{2n-b-a}{3n-2b-a}\right) \cdot (n-a) + o(n)} &= 2^{h(b) \cdot n - \left(h(\alpha') - \frac{n-b+\frac{n-b}{n-a'} \cdot n}{2(n-b)+\frac{n-b}{n-a'} \cdot n}\right) \cdot \frac{n-b}{n-a'} \cdot n + o(n)} \\
&= 2^{\left(h(b) - (n-b) \cdot \frac{h(\alpha') - \frac{2n-a'}{3n-2a'}}{n-a'}\right) \cdot n + o(n)} \\
&= 2^{\left(h(b) - (n-b) \cdot \frac{h(\alpha') - \frac{2-\alpha'}{3-2\alpha'}}{(1-\alpha')n}\right) \cdot n + o(n)}.
\end{aligned}$$

The share size is minimized when $\frac{h(\alpha')-\frac{2-\alpha'}{3-2\alpha'}}{1-\alpha'}$ is maximized. Taking the derivative, we obtain that the maximum occurs when $(2\alpha'-3)^2\log(\alpha')+2(\alpha')^2-8\alpha'+7=0$. By definition, this holds for $\alpha'=\alpha_0$. Thus, the claim follows. □

8.5 Applying the Bootstrapping of Applebaum and Nir

Applebaum and Nir [10] showed a bootstrapping algorithm for linear secret-sharing schemes by exploiting duality. They proved the following.

Lemma 8.19 (Bootstrapping)**.** *Given an integer n and a target slice $b \in [n]$ let $\mathbf{D}_\ell(b,n)[0]$ denote the share size given by Lemma 8.6. For every $i \geq 0$ define*

$$\mathbf{D}_\ell(b,n)[i+1] = \min_{a \leq b}\left(\max\left(\mathbf{M}_\ell(a:b,n),\ \max_{c \leq a}\left(h(c/n)-(c/n)\cdot\frac{h(b/n)-\mathbf{D}_\ell(b,n)[i]}{1-b/n}\right)\right)\right)$$

Then for every $i \geq 0$, every (b,n)-downslice can be realized with a linear secret-sharing scheme with share size at most $\mathbf{D}_\ell(b,n)[i]$.

We next define the function that captures the exponent of the construction. For every $i \geq 0$ we define a sequence of functions $d_i : [0,1] \to [0,1]$ as follows. Let

$$d_1(\beta) = \begin{cases} \frac{1}{2}+\frac{\beta}{2} & \text{if } \beta \leq 1/2 \\ \frac{2-\beta}{3-2\beta} & \text{if } 1/2 < \beta \leq 0.5412 \\ h(\beta) - 0.51079\cdot(1-\beta) & \text{otherwise} \end{cases}$$

be the exponent given by Lemma 8.6. Next, for every $i \geq 0$ let

$$d_{i+1}(\beta) = \min_{\alpha \leq \beta}\left(\max\left(m(\alpha,\beta),\ \max_{\gamma \leq \alpha}(u_i(\gamma,\beta))\right)\right), \tag{11}$$

where $u_i, m : [0,1]^2 \to [0,1]$ are defined as

$$u_i(\gamma,\beta) = h(\gamma) - \gamma\cdot\frac{h(\beta)-d_i(\beta)}{1-\beta}$$

and

$$m(\alpha,\beta) = \begin{cases} \frac{2-h(\alpha/\beta)\cdot\beta}{3-2h(\alpha/\beta)\cdot\beta} & \text{if } \alpha > \beta/2 \\ \frac{2-\beta}{3-2\beta} & \text{otherwise} \end{cases}$$

is the exponent given by Theorem 8.16. By Lemma 8.19, we obtain the following.

Lemma 8.20. *For every $i \geq 0$ and every $\beta \in [0,1]$, it holds that $\mathbf{D}_\ell(\beta) \leq d_i(\beta)$.*

We next show how to simplify Equation (11). Since

$$\frac{\partial u_i(\gamma,\beta)}{\partial\gamma} = \log\left(\frac{1}{\gamma}-1\right) - \frac{h(\beta)-d_i(\beta)}{1-\beta},$$

it follows that for every β, the function $u_i(\gamma, \beta)$ is maximized at

$$\gamma_i := \left(1 + 2^{\frac{h(\beta) - d_i(\beta)}{1-\beta}}\right)^{-1}.$$

Let $\widetilde{u}_i : [0,1]^2 \to [0,1]$ be defined as $\widetilde{u}_i(\alpha, \beta) = \max_{\gamma \leq \alpha} (u_2(\gamma, \beta))$. Then

$$\widetilde{u}_i(\alpha, \beta) = \begin{cases} u_i(\gamma_i, \beta) & \text{if } \gamma_i \leq \alpha \leq \beta \\ u_i(\alpha, \beta) & \text{otherwise} \end{cases}.$$

Therefore,

$$d_{i+1}(\beta) = \min_{\alpha \leq \beta} \left(\max\left(m(\alpha, \beta),\ \widetilde{u}_i(\alpha, \beta)\right)\right).$$

Finally, observe that $m(\alpha, \beta)$ is a decreasing function of α, while $\widetilde{u}_i(\alpha, \beta)$ is an increasing function of α. Since both are continuous, they intersect at exactly one value $\widetilde{\alpha}_i$. This value also minimizes $\max\left(m(\alpha, \beta),\ \widetilde{u}_i(\alpha, \beta)\right)$. Therefore,

$$d_{i+1}(\beta) = m(\widetilde{\alpha}_i, \beta).$$

8.6 A Linear Secret-Sharing Scheme for All Access Structures

We can now show that every access structure can be realized with a linear secret-sharing scheme with an exponent at most 0.7563, thus proving Theorem 8.5. The idea for the construction is roughly as follows. First, as observed by [10], it suffices to consider downslice accesss structures. Similarly to [10], we realize each $(\beta n, n)$-downslice, for $\beta \in [0,1]$, by partitioning $[0,1]$ into 3 segments and deal with each segment using a different scheme. We partition $[0,1]$ differently than [10]. Specifically, we consider the segments $[0, 1/2]$, $[1/2, 0.554]$, and $[0.554, 1]$ ([10] used 0.535 instead of 0.554). Now, for every $\beta < 1/2$, we use the scheme of Applebaum and Nir [10] as stated in Lemma 8.6, for every $\beta \in [1/2, 0.554]$ we reduce it to $(n/2, n)$-downslice, and for every $\beta \in [0.554, 1]$ we reduce it to $(0.554 \cdot n, n)$-downslice. We then apply the bootstrapping construction (Lemma 8.19)for the two values $n/2$ and $0.554 \cdot n$ for 7 iterations. We stress that the choice of our parameters was done using a computer to approximate the optimal choice.

We now formalize the proof. We use the following claim proved by [10], stating that every n-party access structure is the conjunction of at most n downslices.

Claim 8.21. *Let Γ be an n-party access structure. For every $b \in [n]$ let Γ_b denote the b-downslice of Γ. Then $\Gamma = \bigcap_{b=1}^{n} \Gamma_b$. In particular, if every Γ_b can be realized with a linear secret-sharing scheme with exponent S, then so can Γ.*

Proof of Theorem 8.5. By Claim 8.21, it suffices to construct a scheme for every downslice. The proof is done by applying Lemma 8.19for $b_1 = n/2$ and $b_2 = 0.554 \cdot n$ for 7 iterations, and then use the cover reduction lemma (Lemma 8.17) to reduce every other downslice to one these two values. Using a computer to compute the exponents, we get that $\mathbf{D}_\ell(0.5) < 0.736$ and $\mathbf{D}_\ell(0.554) < 0.752$. Using the cover reduction lemma, we obtain the following bounds on the share size of linear secret-sharing schemes for downslices, as explained below.

1. If $0 \leq \beta < 1/2$ then by Lemma 8.6 it holds that $\mathbf{D}_\ell(\beta) \leq \frac{1}{2} + \frac{\beta}{2} < 0.75$.

2. If $1/2 \leq \beta < 0.554$ then we use the covering lemma with $\alpha = 1/2$ to obtain

$$\mathbf{D}_\ell(\beta) < h(\beta) - (1-\beta) \cdot \frac{h(1/2) - \mathbf{D}_\ell(0.5)}{1-0.5} < 0.7561.$$

3. If $0.554 \leq \beta \leq 1$ then we use the covering lemma with $\alpha = 0.554$ to obtain

$$\mathbf{D}_\ell(\beta) < h(\beta) - (1-\beta) \cdot \frac{h(0.554) - \mathbf{D}_\ell(0.554)}{1-0.554} < 0.7563.$$

□

Bibliography

[1] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 118–134, 2001.

[2] Andris Ambainis. Upper bound on the communication complexity of private information retrieval. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proc. of the 24th International Colloquium on Automata, Languages and Programming*, volume 1256 of *LNCS*, pages 401–407, 1997.

[3] Hadar Amran. Constructing multi-servers private information retrieval protocols. Master's thesis, Ben-Gurion University, 2016.

[4] Benny Applebaum and Barak Arkis. On the power of amortization in secret sharing: d-uniform secret sharing and CDS with constant information rate. *ACM Trans. Comput. Theory*, 12(4):24:1–24:21, 2020.

[5] Benny Applebaum, Barak Arkis, Pavel Raykov, and Prashant Nalini Vasudevan. Conditional disclosure of secrets: Amplification, closure, amortization, lower-bounds, and separations. In *CRYPTO 2017*, volume 10401 of *LNCS*, pages 727–757, 2017.

[6] Benny Applebaum, Barak Arkis, Pavel Raykov, and Prashant Nalini Vasudevan. Conditional disclosure of secrets: Amplification, closure, amortization, lower-bounds, and separations. *SIAM J. Comput.*, 50(1):32–67, 2021.

[7] Benny Applebaum, Amos Beimel, Oriol Farràs, Oded Nir, and Naty Peter. Secret-sharing schemes for general and uniform access structures. In *EUROCRYPT 2019*, volume 11478 of *LNCS*, pages 441–471, 2019.

[8] Benny Applebaum, Amos Beimel, Oded Nir, and Naty Peter. Better secret sharing via robust conditional disclosure of secrets. In *52nd STOC*, pages 280–293, 2020.

[9] Benny Applebaum, Thomas Holenstein, Manoj Mishra, and Ofer Shayevitz. The communication complexity of private simultaneous messages, revisited. In *EUROCRYPT 2018*, volume 10401 of *LNCS*, pages 261–286, 2018.

[10] Benny Applebaum and Oded Nir. Upslices, downslices, and secret-sharing with complexity of 1.5^n. In *CRYPTO 2021*, volume 12827 of *LNCS*, pages 627–655, 2021.

[11] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 557–577, 2014.

[12] László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999.

[13] Amos Beimel. Secret-sharing schemes for general access structures: An introduction. Cryptology ePrint Archive, Paper 2025/518, 2025.

[14] Amos Beimel and Oriol Farràs. The share size of secret-sharing schemes for almost all access structures and graphs. In *TCC 2020*, volume 12552 of *LNCS*, pages 499–529, 2020.

[15] Amos Beimel, Oriol Farràs, and Or Lasri. Improved polynomial secret-sharing schemes. Cryptology ePrint Archive, Paper 2023/1158, 2023.

[16] Amos Beimel, Anna Gál, and Mike Paterson. Lower bounds for monotone span programs. Research Series BRICS-RS-94-46, BRICS, Department of Computer Science, University of Aarhus, December 1994.

[17] Amos Beimel and Yuval Ishai. Information-theoretic private information retrieval: A unified construction. In *ICLAP 2001*, volume 2076 of *LNCS*, pages 912–926, 2001. Journal version in [19].

[18] Amos Beimel, Yuval Ishai, Ranjit Kumaresan, and Eyal Kushilevitz. On the cryptographic complexity of the worst functions. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 317–342. Springer-Verlag, 2014.

[19] Amos Beimel, Yuval Ishai, and Eyal Kushilevitz. General constructions for information-theoretic private information retrieval. *J. of Computer and System Sciences*, 71(2):213–247, 2005.

[20] Amos Beimel, Yuval Ishai, Eyal Kushilevitz, and Ilan Orlov. Share conversion and private information retrieval. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012*, pages 258–268, 2012.

[21] Amos Beimel, Hussien Othman, and Naty Peter. Quadratic secret sharing and conditional disclosure of secrets. In *CRYPTO 2021*, volume 12827 of *LNCS*, pages 748–778, 2021.

[22] Amos Beimel and Naty Peter. Optimal linear multiparty conditional disclosure of secrets protocols. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 332–362. Springer, 2018.

[23] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35, 1988.

[24] George Robert Blakley. Safeguarding cryptographic keys. In *Managing requirements knowledge, international workshop on*, pages 313–313. IEEE Computer Society, 1979.

[25] George Robert Blakley. Safeguarding cryptographic keys. In *Proc. of the 1979 AFIPS National Computer Conference*, volume 48, pages 313–317, 1979.

[26] Yeow Meng Chee, Tao Feng, San Ling, Huaxiong Wang, and Liang Feng Zhang. Query-efficient locally decodable codes of subexponential length. *Computational Complexity*, 22(1):159–189, 2013.

[27] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *Proc. of the 36th IEEE Symp. on Foundations of Computer Science*, pages 41–51, 1995. Journal version: *J. of the ACM*, 45:965–981, 1998.

[28] Stephen A. Cook, Toniann Pitassi, Robert Robere, and Benjamin Rossman. Exponential lower bounds for monotone span programs. In *57th FOCS*, pages 406–415, 2016.

[29] Ronald Cramer, Ivan Damgård, and Ueli Maurer. General secure multi-party computation from any linear secret-sharing scheme. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 316–334. Springer-Verlag, 2000.

[30] László Csirmaz. The dealer's random bits in perfect secret sharing schemes. *Studia Sci. Math. Hungar.*, 32(3–4):429–437, 1996.

[31] László Csirmaz. The size of a share must be large. *J. of Cryptology*, 10(4):223–231, 1997.

[32] Devdatt P Dubhashi, Volker Priebe, and Desh Ranjan. Negative dependence through the fkg inequality. *BRICS Report Series*, 1996.

[33] Devdatt P Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *BRICS Report Series*, 3(25), 1996.

[34] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 705–714. IEEE Computer Society, 2010.

[35] Zeev Dvir and Sivakanth Gopi. 2-server PIR with sub-polynomial communication. In *47th STOC*, pages 577–584, 2015.

[36] Zeev Dvir and Sivakanth Gopi. 2-server PIR with subpolynomial communication. *J. ACM*, 63(4):39:1–39:15, 2016.

[37] K. Efremenko. 3-query locally decodable codes of subexponential length. Technical Report TR08-069, Electronic Colloquium on Computational Complexity, `www.eccc.uni-trier.de/eccc/`, 2008.

[38] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *STOC'09*, pages 39–44, November 2009.

[39] Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012.

[40] Paul Erdös and László Pyber. Covering a graph by complete bipartite graphs. *Discrete Mathematics*, 170(1–3):249–251, 1997.

[41] Anna Gál. A characterization of span program size and improved lower bounds for monotone span programs. *Computational Complexity*, 10(4):277–296, 2002.

[42] Anna Gál and Pavel Pudlák. Monotone complexity and the rank of matrices. *Inform. Process. Lett.*, 87:321–326, 2003.

[43] Romain Gay, Iordanis Kerenidis, and Hoeteck Wee. Communication complexity of conditional disclosure of secrets and attribute-based encryption. In *CRYPTO 2015*, volume 9216 of *LNCS*, pages 485–502, 2015.

[44] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. In *Proc. of the 30th ACM Symp. on the Theory of Computing*, pages 151–160, 1998. Journal version: *J. of Computer and System Sciences*, 60(3):592–629, 2000.

[45] Fatemeh Ghasemi, Swastik Kopparty, and Madhu Sudan. Improved pir schemes using matching vectors and derivatives. Technical Report 2411.11611, Cryptology ePrint Archive, 2024. To appear at STOC '25.

[46] Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, 20:71–86, 2000.

[47] Yuval Ishai and Eyal Kushilevitz. Improved upper bounds on information theoretic private information retrieval. In *Proc. of the 31st ACM Symp. on the Theory of Computing*, pages 79 – 88, 1999. Journal version in [19].

[48] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st FOCS*, pages 294–304, 2000.

[49] Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing schemes realizing general access structure. In *Globecom 87*, pages 99–102, 1987. Journal version: Multiple assignment scheme for sharing secret. *J. of Cryptology*, 6(1), 15-20, 1993.

[50] Toshiya Itoh. Efficient private information retrieval. *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, E82-A(1):11–20, 1999.

[51] Toshiya Itoh and Yasuhiro Suzuki. New constructions for query-efficient locally decodable codes of subexponential length. *CoRR*, abs/0810.4576, 2008.

[52] Toshiya Itoh and Yasuhiro Suzuki. Improved constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions on Information and Systems*, E93-D(2):263–270, 2010.

[53] Kumar Joag-Dev and Frank Proschan. Negative association of random variables with applications. *The Annals of Statistics*, pages 286–295, 1983.

[54] Mauricio Karchmer and Avi Wigderson. On span programs. In *8th Structure in Complexity Theory*, pages 102–111, 1993.

[55] J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. of the 32nd ACM Symp. on the Theory of Computing*, pages 80–86, 2000.

[56] Kiran S. Kedlaya and Sergey Yekhanin. Locally decodable codes from nice subsets of finite fields and prime factors of Mersenne numbers. Technical Report TR07-040, Electronic Colloquium on Computational Complexity, `www.eccc.uni-trier.de/eccc/`, 2007.

[57] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *J. of Computer and System Sciences*, 69(3):395–420, 2004.

[58] Samuel Kutin. Constructing large set systems with given intersection sizes modulo composite numbers. *Combinatorics, Probability* Computing, Sep 2002.

[59] Tianren Liu and Vinod Vaikuntanathan. Breaking the circuit-size barrier in secret sharing. In *50th STOC*, pages 699–708, 2018.

[60] Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Conditional disclosure of secrets via non-linear reconstruction. In *CRYPTO 2017*, volume 10401 of *LNCS*, pages 758–790, 2017.

[61] Tianren Liu, Vinod Vaikuntanathan, and Hoeteck Wee. Towards breaking the exponential barrier for general secret sharing. In *EUROCRYPT 2018*, volume 10820 of *LNCS*, pages 567–596, 2018.

[62] Eran Mann. Private access to distributed information. Master's thesis, Technion – Israel Institute of Technology, Haifa, 1998.

[63] Colin McDiarmid. On the method of bounded differences. *Surveys in combinatorics*, 141(1):148–188, 1989.

[64] Anat Paskin-Cherniavsky and Olga Nissenbaum. New bounds and a generalization for share conversion for 3-server pir. *Entropy*, 24(4), 2022. Cited by: 3; All Open Access, Gold Open Access, Green Open Access.

[65] Anat Paskin-Cherniavsky and Leora Schmerler. On share conversions for private information retrieval. *Entropy*, 21(9), 2019.

[66] Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *49th STOC*, pages 1246–1255, 2017.

[67] Toniann Pitassi and Robert Robere. Lifting Nullstellensatz to monotone span programs over any field. In *50th STOC*, pages 1207–1219, 2018.

[68] Lajos Rónyai, László Babai, and Murali K. Ganapathy. On the number of zero-patterns of a sequence of polynomials. *Journal of the AMS*, 14(3):717–735, 2001.

[69] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[70] Hoeteck Wee. Dual system encryption via predicate encodings. In *TCC 2014*, volume 8349 of *LNCS*, pages 616–637, 2014.

[71] Stephanie Wehner and Ronald de Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *Proc. of the 32nd International Colloquium on Automata, Languages and Programming*, volume 3580 of *LNCS*, pages 1424–1436, 2005.

[72] David Woodruff and Sergey Yekhanin. A geometric approach to information-theoretic private information retrieval. In *Proc. of the 20th IEEE Conf. on Computational Complexity*, pages 275–284, 2005.

[73] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. In *Proc. of the 39th ACM Symp. on the Theory of Computing*, pages 266–274, 2007.

A Negatively Associated Random Variables

Definition A.1 (Negative association [53])**.** *Let $X_1, \ldots, X_n$ be random variables. The random variables are negatively associated if for every two disjoint index sets $I, J \subseteq [n]$,*

$$\mathrm{E}\left[f((X_i)_{i\in I}) \cdot g((X_j)_{j\in J})\right] \leq \mathrm{E}\left[f((X_i)_{i\in I})\right] \cdot \mathrm{E}\left[g((X_j)_{j\in J})\right],$$

for all functions $f : \mathbb{R}^{|I|} \to \mathbb{R}$ and $g : \mathbb{R}^{|J|\to\mathbb{R}}$ that are both non-decreasing or both non-increasing.

It is known that the Chernoff-Hoeffding bounds are applicable to sums of variables that satisfy the negative association [63] (see also [33, Proposition 5]).

Theorem A.2 (Chernoff-Hoeffding bounds)**.** *Let $X_1, \ldots, X_n$ be negatively associated random variables taking values in $\{0, 1\}$, let $X = \sum_{i=1}^{n} X_i$, and let $\mu = \mathrm{E}[X]$. Then for every for every $\delta \in (0, 1)$,*

$$\Pr\left[X > (1+\delta)\mu\right] \leq e^{-\delta^2\mu/3} \quad \textit{and that} \quad \Pr\left[X < (1-\delta)\mu\right] \leq e^{-\delta^2\mu/3}.$$

Finally, it is also known that when sampling a subset of $[n]$ uniformly at random, the random variables that correspond to the indicator of whether an element is in the sampled set, are negatively correlated.

Claim A.3 ([32, Theorem 10])**.** *Let $X_1, \ldots, X_n$ be random variables that take values in $\{0, 1\}$ and are distributed uniformly over m-weight vectors where $m \leq n$. That is, for every $x \in \{0, 1\}^n$ with Hamming weight m,*

$$\Pr\left[X = x\right] = \binom{n}{m}^{-1},$$

and $\Pr[X = x] = 0$ for any other $x \in \{0, 1\}^n$. Then the random variables in $X_1, \ldots, X_n$ are negatively correlated.

Since any subset of negatively associated variables are also negatively associated [53, Property 4] we obtain the following.

Claim A.4. *Let I be a random subset of $[n]$ of size μn, sampled uniformly at random, and let $A \subseteq [n]$. For every $i \in A$ let X_i denote the indicator for the event $i \in A \cap I$. Then $(X_i)_{i\in A}$ are negatively correlated.*

B The PIR Protocol of DG

To be able to compare the PIR/CDS protocols that we present in this paper to previous protocols, we present the PIR protocol of DG. They use a matching vector family $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ over $\mathbb{Z}_6^h$ such that for all $i \neq j$ it holds that

$$\langle \mathbf{u}_i, \mathbf{v}_i \rangle \bmod m = 0 \quad \text{and} \quad \langle \mathbf{u}_i, \mathbf{v}_j \rangle \bmod 6 \in \{1, 3, 4\}\,.$$

(Note that in our construction over $\mathbb{Z}_6$, $\langle \mathbf{u}_i, \mathbf{v}_i \rangle \bmod m = 1$ and $\langle \mathbf{u}_i, \mathbf{v}_j \rangle \bmod 6 \in \{0, 2, 3, 4\}$.) Such a matching vector family with $h = 2^{O(\sqrt{\log N \log\log N})}$ is constructed in [46, 58]. In the following, for a prime p let $\langle \mathbf{u}, \mathbf{v} \rangle_p = \sum_{\ell=1}^h \mathbf{u}[\ell]\mathbf{v}[\ell] \bmod p$.

Protocol B.1.
Public parameters: *A matching vector family* $((\mathbf{u}_i, \mathbf{v}_i))_{i=1}^N$ *over* $\mathbb{Z}_6^h$.
Alice's and Bob's input: $D \in \{0,1\}^N$.
The user's input: $i \in [N]$.

- *The user chooses* $\mathbf{r} \leftarrow \mathbb{Z}_2^h$ *with uniform distribution and sends* $\mathbf{q}_a = \mathbf{r}$ *to Alice and* $\mathbf{q}_b = \mathbf{u}_i + \mathbf{r} \bmod 2$ *to Bob.*

- *Alice computes*

$$\mathbf{m}_A^1 = \sum_{j=1}^N (-1)^{\langle \mathbf{q}_A, \mathbf{v}_j \rangle_2} \cdot D_j \bmod 3, \quad \mathbf{m}_A^2 = \sum_{j=1}^N ((-1)^{\langle \mathbf{q}_A, \mathbf{v}_j \rangle_2} \cdot D_j)\mathbf{v}_j \bmod 3$$

  and Bob computes

$$\mathbf{m}_B^1 = \sum_{j=1}^N (-1)^{\langle \mathbf{q}_B, \mathbf{v}_j \rangle_2} \cdot D_j \bmod 3, \quad \mathbf{m}_B^2 = \sum_{j=1}^N ((-1)^{\langle \mathbf{q}_B, \mathbf{v}_j \rangle_2} \cdot D_j)\mathbf{v}_j \bmod 3.$$

  Alice and Bob send m_A^1, m_A^2 *and* m_A^1, m_A^2 *respectively to the user (each answer is a vector in* $\mathbb{Z}_3^h$ *and an element in* $\mathbb{Z}_3$*).*

- *The user outputs* $D_i = 1$ *if*

$$\langle \mathbf{u}_i, \mathbf{m}_B^2 - \mathbf{m}_A^2 \rangle - \mathbf{m}_B^1 + \mathbf{m}_A^1 \not\equiv 0 \pmod 3, \tag{12}$$

  and $D_i = 0$ *otherwise.*

Notice that in Protocol B.1 Alice and Bob need to send an additional element compared to Protocol 5.4 and the reconstruction and the proof of correctness are more involved. Furthermore, writing $(-1)^{\langle \mathbf{q}_A, \mathbf{v}_j \rangle_2}$ instead of $\langle \mathbf{q}_A, \mathbf{v}_j \rangle_2$ further complicated the protocol.