Private Information Retrieval Protocols Based on Transversal Designs

Master Thesis
Group 5.217d
Mathematics
Aalborg University
June 04, 2021

AALBORG UNIVERSITY
STUDENT REPORT

Title:

Private Information Retrieval Protocols

Based on Transversal Designs

Project:

Master Thesis

Project Period:

February 2020 - June 2020

Project Group:

5.217d

Participants:

Christian Juel Martinsen

Supervisor:

Oliver Wilhelm Gnilke

Page Numbers:

46

Date of Completion:

June 04, 2020

The Faculty of Engineering and Science
Mathematics
Skjernvej 4A
9220 Aalborg Øst
http://www.math.aau.dk/

Abstract:

The goal of this thesis is to explore a private information retrieval scheme based on concepts from transversal designs. The construction of 1-private and $(t-1)$-private PIR protocols will be explained and their properties shown. A construction of transversal designs using orthogonal arrays and GRS codes will be shown, since it leads to good PIR protocols. A discovery of a restriction of the size of the GRS code will be included and explained.
A comparison will be made with a general PIR scheme for coded storage with colluding servers. This protocol has a high information rate and good storage properties. Conversely, the transversal design based PIR protocol has a very low amount of complexity.

Synopsis and Signature

Synopsis

Målet med denne thesis er at undersøge et transversal design baseret private information retrival scheme. Konstruktionen af 1-private og $(t-1)$-private PIR protokoller vil blive forklaret og deres egenskaber vist. Konstruktionen af transversal designs baseret på orthogonal arrays og GRS koder vil blive vist, idet den giver protokoller med gode egenskaber. En opdagelse af en indsnævring af størrelsen på GRS koder vil blive inkluderet og forklaret.

Der vil blive lavet en sammenligning med en generel PIR protokol for kodet opbevaring med samarbejdende servere. Denne protokol har en høj informations rate og gode opbevarings egenskaber. Modsat har PIR protokollen baseret på transversal designs en meget lav grad af kompleksitet.

Signature

Christian Juel Martinsen

Preface and Readers' Guide

Preface

This project is written by a master's student in tenth semester mathematics at Aalborg University during the spring semester 2021. The main theme of this project is private information retrieval, and a particular scheme that is based on transversal designs. Prerequisite knowledge required for reading this project correspond to a bachelor's degree in mathematics. In particular, basic knowledge of linear algebra, abstract algebra, design theory and coding theory is recommended.

I would like to thank Adjunct Oliver Wilhelm Gnilke for his help and supervision during the writing of this project.

Readers' Guide

The Vancouver reference style is used for the bibliography. Definitions, theorems, propositions, lemmas, and examples are numbered consecutively according to the chapter they are in. Figures and tables are numbered in a similar manner. As are equations.

In this project we use the following specific notation:

- The set A^B is the set of n tuples $a = (a_b)_{b \in B}$ of the A-elements indexed by the set B, where $|B| = n < \infty$. This can also be considered as functions from B to A.
- $a_{|T} = (a_t)_{t \in T}$ for $T \subset B$ is the restriction of a to the coordinates of T.
- The expression "PIR scheme" refers to every facet of the PIR scheme, including background and construction. The expression "PIR protocol" refers to the implemented and use-able part.

Contents

1 Introduction

In the modern globalized world, a lot of the most important information is stored on computers. Some of the most important storage mediums are servers, since it is easy to retrieve or manipulate items for people who don't own the servers. An interesting problem presents itself when the servers contain information that should only be accessed by people who should know it. Specifically if one needs to access something on a server, but the one who owns the server shouldn't know exactly what has been accessed. This is called Private Information Retrieval (PIR). The trivial solution to the PIR problem is to download everything on the server, but this is also very impractical. Thus one needs to get creative when designing PIR schemes. The most common PIR protocols uses multiple servers, where the information is encoded and distributed between the servers in some way. The user who wants to retrieve a certain part of the encoded database, then sends a query to each server, with the servers not knowing what the user wants to retrieve. The servers then sends an answer back to the user, based on the received query. The user then takes the answers and uses a reconstruction algorithm to finally get what they wanted.

In this project a PIR scheme based on transversal designs will be explored. The most common PIR schemes get their properties from coding theory, but this scheme takes advantage of a useful property of transversal designs: They partition the set they are defined upon in equally big disjoint sets, with each element in the set appearing in a partition. This partition means that the database can be distributed based on the support of the transversal design, with each server being sent coordinates corresponding to a partition. The queries of the scheme are chosen based on the blocks of the transversal design.

To give a fundamental understanding of the transversal design based PIR scheme, in Chapter 2 the basic theory of PIR schemes will be defined. In particular the concept of replication based PIR protocols and distributed PIR protocols will be explored, since the transversal design based PIR scheme is a distributed PIR protocol. In Chapter 3 design theory and the transversal design will be reviewed. Some very important results of coding theory will also be included, since this is very important for all PIR schemes. In particular, how to construct a code from a design will be shown, since it is very important in creating the PIR scheme.

With the fundamental theory stated, in Chapter 4 the construction of the transversal design based PIR protocol will be explored. First it will be shown how the 1-private PIR protocol based on a transversal design works; how the queries are generated, answer from the servers computed and the reconstruction will be shown in detail. Afterwards, how to construct transversal designs with good properties will be shown. The construction is based on orthogonal arrays, which can be turned into transversal designs. The orthogonal arrays will be defined from codes, in particular GRS codes. The divisibility condition will be presented, since it has implications for the rate of the scheme. Lastly it will be shown how to obtain t-transversal designs from strength t orthogonal arrays. Using this parallel, it will be shown how the 1-private transversal design based PIR protocol can be generalized into the $(t-1)$-private PIR protocol, by using t-transversal designs.

Lastly, to better understand the construction and the properties of the transversal design PIR scheme, an example and a comparison with another scheme will be made in Chapter 5. The example is based on the orthogonal array construction, and this PIR scheme will be 1-private. Afterwards another PIR scheme will be introduced as a comparison point. This scheme is more coding theory based and derives its properties from the star product between two codes. The comparisons will be the rate, storage and complexity between each scheme.

An interesting observation is made in regards to the orthogonal array construction, specifically the implications of using GRS codes as basis for the construction. One should not use GRS codes with prime length, since it results in very small codes. This observation is explained in Chapter 5.

2 Private Information Retrieval

In this chapter some of the most fundamental theory of PIR schemes will be presented, in particular the concept of replication-based and distributed PIR protocols. This chapter is based on [1], [2] and [3].

Let the database used in the PIR scheme be denoted $D = (D_i)_{1 \leq i \leq k} \in \mathbb{F}_q^k$. This database contains $k \log(q)$ bits. This project focuses on PIR protocols that use distributed storage systems. In these types of schemes, the database is in a predetermined way distributed on a number of servers: It is common for PIR schemes to clone the database at the start and then store a copy on all the servers $S_1, \ldots, S_l$. The role of each server S_j is to compute some combination of symbols from D, related to a query sent by the user. Such replication-based PIR protocols can be formally defined by:

Definition 2.0.1. Replication-based PIR protocol
For $1 \leq j \leq l$ assume that every server S_j stores a copy of the database D. An l-server replication-based PIR protocol is a set of three algorithms (Q, A, R) which on input $i \in [1, k]$ runs:

1. Query generation: Q is a randomized algorithm that generates l queries $(q_1, \ldots, q_l) = Q(i)$. Query q_j is sent to server S_j.
2. Answer from the servers: Each server S_j computes an answer $a_j = A(q_j, D)$ and sends it back to the user.
3. Reconstruction: The user computes and outputs $r = R(i, \mathbf{a}, \mathbf{q})$, where $\mathbf{a} = (a_1, \ldots, a_l)$ and $\mathbf{q} = (q_1, \ldots, q_l)$.

Two important properties of the PIR protocol is the correctness and the privacy requirement of the protocol. Correctness is the ability of protocol to reconstruct the desired database entry, based on the query.

Definition 2.0.2. Correctness
Assume that the PIR protocol has been run with input i. If $r = D_i$, then the PIR protocol is said to be correct.

So when the servers follow the PIR protocol it is correct. The privacy of the protocol is the requirement that $Q(i)$ is distributed independently of the index i. This means the servers gain no information about the identity of i, which in practice means the queries sent to the servers must be independent of what one wants to be reconstruct with the help of the servers.

Definition 2.0.3. Privacy
If for every $(i, i') \in [1, k]^2$ and every $T \subseteq [1, l]$ with $|T| \leq t$ chosen uniformly, the distributions $Q(i)_{|T}$ and $Q(i')_{|T}$ contain the same information, the PIR protocol is t-private. Equivalently, as expressed with mutual information:

$$I(Q(i)_{|T}; i) = I(Q(i')_{|T}; i') = 0$$

t-privacy is equivalent to saying that the PIR protocol resists t-colluding servers.

Communication complexity is the number of bits exchanged between the user and the servers. Computational complexity is the maximal number of $\mathbb{F}_q$-operations made by a server in order to compute an answer a_j. The servers must jointly carry l copies of the database, and since $|D| = k \log(q)$ the combined storage of the scheme is $(l-1)k \log(q)$ bits.

A way to reduce the computation cost of PIR protocols is to preprocess the database: A model for which the database can be encoded and distributed over the servers is desired. Let $c = (c_i)_{i \in K}$ denote an encoding of the database D, so the image of D is an injective map $\mathbb{F}_q^k \to \mathbb{F}_q^K$, with $|K| = n \geq k$. For convenience it can be assumed that $K = [1, s] \times [1, l]$, $c_{(i_1,i_2)}$ can be written $c_{i_1}^{(i_2)}$ and $(c_r^{(j)})_{r \in [1,s]}$ can be written $c^{(j)}$. Using this it is possible to define a distributed PIR protocol:

Definition 2.0.4. Distributed PIR protocol
For $1 \leq j \leq l$ assume that the server S_j holds the part $c^{(j)}$ of the encoded database. An l-server distributed PIR protocol is a set of three algorithms (Q, A, R) which on input $i \in I$ runs:

1. Query generation: Same as Definition 2.0.1
2. Answer from the servers: Each server S_j computes an answer $a_j = A(q_j, c^{(j)})$ and

sends it back to the user.

3. Reconstruction: Same as Definition 2.0.1

The database D has now been encoded, and so the storage overhead is defined as the number of redundancy bits stored by the servers. The storage overhead is thus $(sl - k)\log(q)$ bits. Sometimes it is instead taken as the number of symbols of $\mathbb{F}_q$ that holds no information. Then the storage overhead is of course $sl - k$ $\mathbb{F}_q$ symbols.

In regards to the distributed PIR protocol, the concept of the storage overhead leads to a related concept called the rate. The rate of a PIR scheme is taken as the gained information over the downloaded information, and since the data is in $\mathbb{F}_q^k$ and the encoded data is in $\mathbb{F}_q^{sl}$, the rate is $R = \frac{k}{sl}$.

The PIR scheme in this project is based on transversal designs, as opposed to more common schemes, with roots more firmly in coding theory. A more traditional scheme will be included as a comparison point later.

3 Transversal Designs

In this chapter, design theory is introduced and transversal designs are defined. These designs are fundamental to the PIR protocol. Furthermore, a few results of coding theory will be introduced, because coding theory is very important for working with PIR protocols. This chapter is based on [1], [4], [5] [6] and [7].

Design theory has many facets, but at it's heart is the "block design", of which transversal designs are simply a special kind.

Definition 3.0.1. Block design
Let X be a finite, non-empty set of elements and B be a family of subsets of X called blocks. The pair $\mathfrak{D} = (X, B)$ is called a block design.

There exist a myriad of different designs, all characterized by incidence constraints between points and blocks. One of the most simple constraints is that each pair of elements in X must appear in the same number of blocks. In the most simple case, this gives rise to balanced incomplete block designs. This will not be explored in great detail, since the focus is transversal designs, but transversal designs also come from the same basic incidence constraints.

The incidence constraints can be represented by vectors and matrices. The incidence vector $1_b \in \{0,1\}^X$ for $b \subset X$ is the row vector whose x-th coordinate is 1 iff. $x \in b$. Block designs can be represented by an incidence matrix, which is built of incidence vectors.

Definition 3.0.2. Incidence matrix
Let $\mathfrak{D} = (X, B)$ be a block design and let $j \in X$ and $i \in B$ be arbitrary. An incidence matrix $M_{\mathfrak{D}}$ of $\mathfrak{D}$ is a size $|B| \times |X|$ matrix, whose entries m_{ij} are given by:

$$m_{ij} = \begin{cases} 1 & \text{if block i contains point j} \\ 0 & \text{otherwise} \end{cases}$$

The q-rank of $M_{\mathfrak{D}}$ is the rank of $M_{\mathfrak{D}}$ over the field $\mathbb{F}_q$.

An incidence matrix of a design depends on the ordering of the blocks and points and thus is not unique. All incidence matrices representing the same design are permutations of each other, and so have the same q-rank. This means:

Definition 3.0.3. q-rank of a design
The q-rank of a design is the q-rank of any of its incidence matrices.

Transversal designs are block designs where the elements have been partitioned into groups, with a special property of each pair of elements of X appearing in groups and blocks in special ways:

Definition 3.0.4. Transversal design
A transversal design $TD_\lambda(l, s)$, for integers $s, l \geq 2$ and $\lambda \geq 1$, is a block design (X, B) equipped with a partition $\mathfrak{G} = \{G_1, \ldots, G_l\}$ of X called the set of groups, for which:

- $|X| = ls$,
- $|G_i| = s$, $\forall G_i \in \mathfrak{G}$
- $|b_i| = l$, $\forall b_i \in B$
- Any pair of elements from X is contained in either one group and no block or in no group and λ blocks.

Sometimes the transversal design is represented as $\mathfrak{T} = (X, B, \mathfrak{G})$. If $\lambda = 1$, then the notation $TD(l, s)$ is used. Definition 3.0.4 implies that a block cannot be secant to a group in more than one point. Furthermore, any block must meet any group, by which:

$$\forall (b, G) \in B \times \mathfrak{G}, |b \cap G| = 1$$

There must be exactly λs^2 blocks in B, which can be established by counting pairs of elements: There are $\binom{sl}{2}$ unordered pairs in X, l groups and each group contains $\binom{s}{2}$ groups. Thus there are $\left(\binom{sl}{2} - l\binom{s}{2}\right)$ unordered pairs in X contained in a block $b \in B$. This set is now denoted S. By using Definition 3.0.4, the sum

$$\sum_{\{x,y\} \in S} \sum_{b \in B} 1_{\{x,y\} \subset b} = \sum_{\{x,y\} \in S} \lambda = \sum_{b \in B} \binom{l}{2},$$

can be established. That means:

$$\sum_{\{x,y\} \in S} \lambda = \lambda \left(\binom{sl}{2} - l\binom{s}{2} \right) = \lambda s^2 \binom{l}{2}$$

$$\sum_{b \in B} \binom{l}{2} = |B| \binom{l}{2}$$

by which $|B| = \lambda s^2$.

3.1 Linear Codes From Block Designs

To build the PIR protocol, it is needed to know how a linear code can be associated to a transversal design. A linear code is defined by:

Definition 3.1.1. Linear code
Let $\mathbb{F}_q$ be the finite field of q elements. A linear code C is a k-dimensional subspace of the vector space $\mathbb{F}_q^n$. It has has codeword length n, message length k and is denoted $[n, k]$.

The size of the code is the number of codewords, which equals q^k.

One of the most important concepts in coding is distance and weight.

Definition 3.1.2. Distance
The distance between two elements in $x, y \in C$, denoted $d(x, y)$ is the number of coordinate places in which they differ:

$$d(x, y) = |\{i \mid 1 \leq i \leq n, x_i \neq y_i\}|$$

This is a distance function, that fulfills all the normal criteria of distance functions:

Definition 3.1.3. Distance function
A distance function, or a metric, on a set X is a function

$$d : X \times X \to [0, \infty),$$

for which, with $x, y, z \in X$, the following three axioms are satisfied:

- $d(x, y) = 0 \Leftrightarrow x = y$,
- $d(x, y) = d(y, x)$,
- $d(x, y) \leq d(x, z) + d(z, y))$.

The weight of a codeword is a natural extension of the distance, and vice versa:

Definition 3.1.4. Weight
The weight $w(x)$ of an element $x \in C$ is $d(x, \vec{0})$ with $\vec{0}$ being the zero vector, or equivalently:

$$w(x) = |\{i \mid 1 \leq i \leq n, x_i \neq 0\}|$$

Most often these quantities are called Hamming distance and Hamming weight.

The minimum distance of an entire code has many formulations, but a simple one is:

Definition 3.1.5. Minimum distance of a code
The minimum distance of a code is the minimum Hamming distance between any pair of different codewords.

A simple way of finding the minimum distance of a linear code is to use the weight:

Proposition 3.1.6.
The minimum distance of a linear code C is equal to the minimum weight among all nonzero codewords.

Proof:
C is a linear subspace, so if $x, y \in C$, then $x - y \in C$. Therefore $d(x, y) = d(x - y, 0) = w(x - y)$. ■

A generator matrix of a code C is matrix, for which the rows are a set of basis vectors of C:

Definition 3.1.7. Generator matrix
Let C be a $[n, k]$ code. A generator matrix G for the code C is a $k \times n$ matrix whose rows are taken from C and are linearly independent.

The association of a linear code with a transversal design in this project is based on the dual of a code.

Definition 3.1.8. Dual code
The dual $C^{\perp}$ of the code $C \in \mathbb{F}_q^n$ is the vector space consisting of all vectors $h \in \mathbb{F}_q^n$ such that $\forall c \in C$:

$$\sum_{i=1}^{n} c_i h_i = 0 \tag{3.1}$$

The dual code $C^{\perp}$ is an $[n, n - k]$ code. If H is a generator matrix for $C^{\perp}$, it is called a parity-check matrix for C. A parity-check or parity-check vector is a vector h that is

orthogonal to all the words of the code.

Definition 3.1.9. Parity check matrix
A parity-check matrix H for a $[n, k]$ code C is a $(n-k) \times n$ matrix whose rows are linearly independent parity-checks.

A linear code can be built from a block design, by simply taking a block from the design and considering it as a parity-check vector of the code. The parameters of the code depend on the q-rank of design, since the parity-check matrix is derived from the incidence matrix.

Definition 3.1.10.
Let $\mathbb{F}_q$ be a finite field and $\mathfrak{D} = (X, B)$ be a block design and $n = |X|$. The code $C_q(\mathfrak{D})$ is the $\mathbb{F}_q^n$-linear code, where the designs incidence matrix $M_{\mathfrak{D}}$ is a matrix consisting of parity checks of the code. The dimension of $C_q(\mathfrak{D})$ over $\mathbb{F}_q$ equals $|X| - \text{rank}_q(M_{\mathfrak{D}})$.

$C_q(\mathfrak{D})$ is uniquely defined up to a chosen ordering of X and the ordering of blocks does not affect the code. Furthermore, since $M_{\mathfrak{D}}$ has coefficients in $\{0, 1\}$, $\text{rank}_q(M_{\mathfrak{D}}) = \text{rank}_p(M_{\mathfrak{D}})$, where p is the characteristic of the field $\mathbb{F}_q$.

To make distribution easier, a systematic encoding of the database is used at the start of the PIR protocol. A systematic encoding is useful for retrieving the message m from the codeword c efficiently.

Definition 3.1.11. Systematic encoding
Let $C \subseteq \mathbb{F}_q^n$ be a linear code of dimension $k \leq n$. A systematic encoding of C is a one-to-one map $\varphi : \mathbb{F}_q^k \to C$, such that there exists an injective map $\sigma : [1, k] \to [1, n]$, which $\forall m \in \mathbb{F}_q^k$ and $\forall i \in [1, k]$, satisfies:

$$m_i = \varphi(m)_{\sigma(i)}$$

The set $\sigma([1, k]) \subseteq [1, n]$ is called an information set of C.

With all the basic theory presented, the PIR protocol will now be introduced.

4 PIR Protocol Construction From Transversal Designs

In this section the common construction of the PIR protocol based on transversal designs is presented. The standard construction results in 1-private schemes, and constructions resulting in stronger privacy will be explored later. This section is based on [1], [4], [8] and [9].

4.1 1-private Distributed PIR Protocol

The basic PIR protocol is 1-private, i.e. with no colluding servers the scheme guarantees perfect security. This is because the construction exploits a useful property of transversal designs: The knowledge of one point in a block from a transversal design gives almost no information on the other points in that block.

As the starting point let $\mathfrak{T} = (X, B, \mathfrak{G})$ be a transversal design with parameters λ, l, s and $n = |X| = ls$. Let the associated $\mathbb{F}_q$-linear code be denoted $\mathfrak{C} = C_q(\mathfrak{T}) \subseteq \mathbb{F}_q^{ls=n}$ and let $k = \dim_{\mathbb{F}_q} \mathfrak{C}$. The overall construction of the PIR scheme based on a transversal design can be summarized in three steps:

Construction 4.1.1.

$$TD(l,s) \underset{\text{Incidence matrix}}{\rightarrow} C_q(\mathfrak{T}) \subseteq \mathbb{F}_q^n \underset{\text{Database encoding}}{\rightarrow} \text{Distributed PIR protocol}$$

Each step of the construction of the protocol itself will now be explained. For a summation, see Construction 4.1.2. The construction of the distributed PIR protocol should follow along the lines of 2.0.4. The first step of the construction is the initialization, consisting of database encoding and distribution to the servers:

- The encoding consists of the user computing a systematic encoding, like Definition 3.1.11, of the database $D \in \mathbb{F}_q^k$, resulting in the codeword $c \in \mathfrak{C}$.

- For the distribution, denote by $c^{(j)} = c_{|G_j}$ the symbols of c whose support is the group $G_j \in \mathfrak{G}$. Each server S_j receives the corresponding $c^{(j)}$, for $1 \leq j \leq l$.

Next is the retrieving step. Here the goal is to retrieve the symbol c_i for $i \in X$. The groups in the transversal design $\mathfrak{T}$ contain each element once, so the index j^* will denote the unique group G_{j^*} that contains i, i.e. $c_i = c_r^{j^*}$ where $r \in [1, s]$. Furthermore, the subset of blocks containing i are denoted B^*. The next part is the query generation, the answer from the servers and the reconstruction. These steps can be explained as:

- *Query generation:* As derived from Definition 3.0.4, it is known that each group and each block has one common element. The blocks of B^* consists of the blocks that intersects Q_{j^*} in i. So the user now picks at random a block $b \in B^*$. For $j \neq j^*$ there is a unique element in the intersection $b \cap G_j$ that is not contained in G_{j^*}. This is the index $q_j \in b \cap G_j$ and this index is sent to server S_j. The remaining server S_{j^*} receives a random query q_{j^*} uniformly picked in G_{j^*}. So all elements of the design has a chance to be picked as a query, and each query is unique.
- *Answer from the servers:* Each server S_j (including S_{j^*}) simply reads the query q_j sent to it and sends back $a_j = c_{q_j}$ as the answer.
- *Reconstruction:* The reconstruction relies on Definition 3.0.4, from which it is known that the incidence vector 1_b belongs to the dual code $\mathfrak{C}^\perp$. That means for a $c \in \mathfrak{C}$ $\sum_{x \in b} c_x = 0$. So, since the servers S_j for $j \neq j^*$ receive queries correspond to a b that contains i, it must be true that:

$$c_i = -\sum_{x \in b \setminus \{i\}} c_x = -\sum_{j \neq j^*} c_{q_j}$$

So in the reconstruction, the user computes

$$r = -\sum_{j \neq j^*} c_{q_j} = -\sum_{j \neq j^*} a_j$$

and outputs r.

Construction 4.1.2. 1-private distributed PIR protocol
Input: A transversal design $TD_\lambda(l, s)$ called $\mathfrak{T} = (X, B, \mathfrak{G})$. A code $C_q(\mathfrak{T})$ with length $n = ls$ and dimension $k = \dim_{\mathbb{F}_q} \mathfrak{C}$.

1. **Initialization step**.
   (a) *Encoding.* Compute a systematic encoding of the database $D \in \mathbb{F}_q^k$, resulting in the codeword $c \in \mathfrak{C}$.
   (b) *Distribution.* Set $c^{(j)} = c_{|G_j}$ as the symbols of c whose support is the group $G_j \in \mathfrak{G}$. For $1 \leq j \leq l$, send each $c^{(j)}$ to the corresponding server S_j.

2. **Retrieving step for symbol c_i for $i \in X$.** Pick $c_i = c_r^{(j^*)}$, where $j^* \in [1, l]$ is the index of the unique group G_{j^*} containing i and $r \in [1, s]$. Set B^* as the subset of blocks containing i. Run the three algorithms (Q, A, R), given by:

   (a) *Q: Queries generation.* Pick uniformly at random a block $b \in B^*$. For $j \neq j^*$ send the unique index $q_j \in b \cap G_j$ to server S_j. Send a random query q_{j^*} uniformly picked in G_{j^*} to S_{j^*}.

   (b) *A: Answer from the servers.* Each server S_j reads $a_j := c_{q_j}$ and sends it back to the user:

   $$A(q_j, c^{(j)}) = c_{q_j}$$

   (c) *R: Reconstruction.* Set $\mathbf{a} = \{a_1, \ldots, a_l\}$ and $\mathbf{q} = \{q_1, \ldots, q_l\}$. Compute

   $$r = R(i, \mathbf{a}, \mathbf{q}) := -\sum_{j \neq j^*} a_j = -\sum_{j \neq j^*} c_{q_j}.$$

   Output: The reconstruction r

As noted in Chapter 3, there are five fundamental factors in the analysis of the scheme: Correctness, security, communication complexity, computation complexity and storage overhead.

The correctness has already been considered in the explanation of the reconstruction step in the algorithm. The protocol is correct as long as there are no errors in the symbols $a_j := c_{q_j}$ returned by the servers.

In the case of the security, consider the uniformly random pick of $b \in B^*$. It suffices to prove that $\mathbb{P}(i \mid q_j) = \mathbb{P}(i)$ for all $j \in [1, l]$, where the probabilities are taken over the randomness. By the law of total probability:

$$\mathbb{P}(i \mid q_j) = \mathbb{P}(i \mid q_j \wedge i \in G_j)\mathbb{P}(i \in G) + \mathbb{P}(i \mid q_j \wedge i \notin G_j)\mathbb{P}(i \notin G)$$

q_j can be eliminated by the following two observations:

- When $i \in G_j$, the definition of the protocol means that q_j is uniformly random, making q_j and i independent.
- When $i \notin G_j$. Definition 3.0.4 implies that there are as many blocks containing both q_j and i as there are blocks containing both q_j and any i' in $X \setminus G_j$. Thus q_j and i are again independent.

These observations mean:

$$\mathbb{P}(i \mid q_j) = \mathbb{P}(i \mid i \in G_j)\mathbb{P}(i \in G) + \mathbb{P}(i \mid i \notin G_j)\mathbb{P}(i \notin G) = \mathbb{P}(i),$$

by which the scheme is 1-private, or equivalently it protects against a single non-colluding server.

The complexities are simple: For communication complexity, exactly one index in $[1, s]$ and one symbol in $\mathbb{F}_q$ are exchanged between each server and the user, thus the overall communication complexity $l(\log(s)+\log(q)) = l\log(sq)$ bits. The computation complexity of the protocol is very low, since each server S_j only needs to read the symbol defined by query q_j. There is thus no extra computational cost for the protocol.

Finally, in the case of the storage overhead, the number of bits stored on a server is $s\log(q)$, giving a total storage overhead of $(ls-k)\log(q)$ bits.

In conclusion, the following theorem holds:

Theorem 4.1.3.
Let D be a database with k entries over $\mathbb{F}_q$ and $\mathfrak{T} = TD(l, s)$ be a transversal design, whose incidence matrix has rank $ls-k$ over $\mathbb{F}_q$. Then there exists a distributed l-server 1-private PIR protocol with:

- Only one $\mathbb{F}_q$ symbol to read for each server,
- $l-1$ field operations over $\mathbb{F}_q$ for the user,
- $l\log(sq)$ bits of communication, since $l\log(s)$ are uploaded and $l\log(q)$ are downloaded,
- a total storage overhead of $(ls-k)\log(q)$ bits on the servers.

Theorem 4.1.3 implies that, when optimizing the practical parameters of the PIR scheme, a small number of groups in the transversal design is desired, i.e. small values of l in $TD_\lambda(l, s)$. Conversely, the dimension of $\mathfrak{C}$ strongly depends on l and n, so too small values of l can lead to trivial or very small codes. Thus one should find codes with large dimension compared to their lengths, arising from transversal designs with few groups. The characteristic of the field $\mathbb{F}_q$ should be chosen carefully in order to obtain non-trivial codes, as described by the following proposition:

Proposition 4.1.4.
Let $\mathfrak{T} = (X, B, \mathfrak{G})$ be a transversal design given by $TD_\lambda(l, s)$ and let $q = p^e$ with p prime. If p does not divide λs then

$$\mathfrak{C} \subseteq \{c \in \mathbb{F}_q^{ls}, \forall G \in \mathfrak{G}, c_{|G} \in \mathrm{Rep}(s)\},$$

where $\mathrm{Rep}(s)$ is the repetition code of length s. In particular if p does not divide λs, then $\dim_{\mathbb{F}_q} \mathfrak{C}$ is at most l.

Proof:
For $x \in X$ one has $B_x = \{b \in B, x \in b\}$, so define $a^{(x)} = \sum_{b \in B_x} 1_b$. Since $C^\perp$ is generated by $\{1_b, b \in B\}$, it follows that $a^x \in C^\perp$. Denote $G_x \in \mathfrak{G}$ as the only group that contains x. Then:

$$\begin{array}{ll} a^x_x = \lambda s & \\ a^x_i = 0 & \forall i \in G_x \setminus \{x\} \\ a^x_j = \lambda & \forall j \in X \setminus G_x \end{array}$$

This means $a^{(x)} - a^{(y)} = \lambda s \left(1_{\{x\}} - 1_{\{y\}}\right)$ if x and y are in the same group. If p does not divide λs, then $1_{\{x\}} - 1_{\{y\}} \in C^\perp$. Now, let

$$\mathfrak{C} = \mathrm{Span}_{\mathbb{F}_q} \left\{1_{\{x\}} - 1_{\{y\}}, \forall x, y \in X; \{x, y\} \subset G \in \mathfrak{G}\right\},$$

by which:

$$\mathfrak{C} \subseteq \{c \in \mathbb{F}_q^{ls}, \forall G \in \mathfrak{G}, c_{|G} \in \mathrm{Rep}(s)\},$$

and so the desired result is obtained. ■

Constructions of transversal designs using orthogonal arrays will now be presented.

4.2 Explicit Construction of 1-private TD-based PIR Protocols

Let $l(k)$ denote the number of servers involved in a given PIR protocol running on a database with k entries and $n(k)$ the actual number of symbols stored by all servers. These parameters correspond respectively to the block size and the number of points of the transversal design used in the construction. Small values of l and n are desired.

A good way to produce transveral designs which result in high rate codes is to base them on special geometries or other combinatorial constructions. Two special geometries that can result in good transversal designs are affine geometries and projective geometries, but these will not be explored in this project. This project will focus on a construction called the orthogonal array from which a transversal design can be derived in an interesting way. Furthermore, the concept of strength of an orthogonal array, leads to the concept of t-transversal designs.

4.2.1 Orthogonal Arrays

There are many ways of defining an orthogonal array, but the one that uses the sub-array notation is preferred in this project.

Definition 4.2.1. Orthogonal Array
Let $\lambda, s \geq 1$ and $l \geq t \geq 1$. Define A as an $\lambda s^t \times l$ array with entries being elements of set S with $|S| = s$. A is called an orthogonal array $OA_\lambda(t, l, s)$ if, in any $\lambda s^t \times t$ sub-array A' of A, each ordered size t subset of S appears exactly λ times in the rows of A'.

For an orthogonal array $OA_\lambda(t, j, s)$, λ is called the index, t is called the strength, and l is called the degree. If the strength is omitted, then $t = 2$ and if the index is omitted then $\lambda = 1$. In the case of both being omitted the orthogonal array is denoted $A = OA(l, s)$. These are the most common orthogonal arrays, but higher strength orthogonal arrays will be important for further developing the PIR scheme. For convenience, this project restricts Definition 4.2.1 to orthogonal arrays with no repeated column or row.

The following proposition shows how to construct transversal designs from orthogonal arrays:

Proposition 4.2.2.
A transversal design $TD(l, s)$ can be constructed from an orthogonal array $OA(l, s)$.

Proof:
Let A be an $OA(l, s)$ orthogonal array with $R(A)$ denoting the s^2 rows of A. A transversal design is defined by its point set X, its block set B and its group set $\mathfrak{G}$. First, the point set can be constructed as $X = S \times [1, l]$. For the block set, a block can be defined by associating each row $c \in R(A)$ to a block:

$$b_c := \{(c_i, i), i \in [1, l]\}$$

The block set can then be constructed as:

$$B := \{b_c, c \in R(A)\}$$

Finally for the group set, $\mathfrak{G} := \{S \times \{i\}, i \in [1, l]\}$ will partition X into l groups. Thus Definition 3.0.4 means that the sets $(X, B, \mathfrak{G})$ defines a transversal design $TD(l, s)$. ■

To expand the exploration of orthogonal arrays, a concept is introduced called a generic code $C_0 \subset S^l$, which is just some code that can be defined as necessary. Listed in rows, all the codewords of such a generic code C_0 forms an orthogonal array, with t derived from the dual distance d' of C_0 by $t = d' - 1$. For linear codes, the dual distance is the distance of the dual code. For a deeper exploration of this concept, see $[4, p.698 - 699]$.

Given a code C_0, the orthogonal array it defines is denoted A_{C_0} and the transversal design built from this orthogonal array is denoted $\mathfrak{T}_{C_0}$. Finally, $C_q(\mathfrak{T}_{C_0})$ denotes the code obtained

from the $\mathfrak{T}_0$, and thus it is possible to construct a PIR scheme. The parity-check matrix $M_{T_{C_0}}$ of $C_q(\mathfrak{T}_{C_0})$ stores incidence relations between all the codewords in C_0 so the code is called the incidence code of C_0:

Definition 4.2.3. Incidence code
Let C_0 be a generic code $C_0 \subset S^l$ with $|S| = s$. The incidence code of C_0 over $\mathbb{F}_q$, is the $\mathbb{F}_q$-linear code of length $n = sl$ built from the transversal design $\mathfrak{T}_{C_0}$:

$$IC_q(C_0) := C_q(\mathfrak{T}_{C_0})$$

The alphabets S and $\mathbb{F}_q$ need not be equivalent.

The point of introducing incidence codes is to link C_0 more directly with PIR schemes, since the overall construction can be expanded to:

Construction 4.2.4.

$$C_0 \underset{\text{Equivalence}}{\leftrightarrow} OA(l,s) \underset{\textit{Proposition } 4.2.2}{\rightarrow} TD(l,s) \underset{\text{Incidence matrix}}{\rightarrow} IC_q(C_0) \underset{\text{Database encoding}}{\rightarrow} \text{Distributed PIR Scheme}$$

The link between generic codes, incidence codes and PIR protocols is very interesting: Since C_0 is a generic code, the incidence code $IC(C_0)$ has an innumerable amount of definitions and thus an incredibly large family of PIR protocols is obtained. Most of these protocols are not practical though, essentially because the kernel of the incidence matrix of $IC(C_0)$ is too small. An easy way to simplify the search is to notice that the more blocks a transversal design contains, the larger its incidence matrix is. This means that the associated code has a smaller dimension for large amounts of blocks. Conversely, the number of blocks of $\mathfrak{T}_{C_0}$ is the number of codewords of C_0. So the search is restricted to small generic codes since this should result in large incidence codes. These observations lead to the consideration of using maximum distance separable, also called MDS, codes.

Maximum Distance Separable codes

MDS codes are very important for many different PIR schemes, but in this case they will be used to explore incidence codes. They arise from the singleton bound, which is given by:

Theorem 4.2.5. Singleton bound
Let C be an $[n, k]$ code with minimum distance d. Then

$$d \leq n - k + 1$$

Proof:
Delete $d - 1$ fixed positions from all the q^k codewords. They must still be different since each pair differ in at least d positions. There are q^{n-d+1} vectors with the remaining positions and thus $k \leq n - d + 1$. ■

An MDS code is called as such, because it meets the singleton bound.

Definition 4.2.6. MDS code
A [n,k,d] linear code is called maximum distance separable, if it reaches the singleton bound:

$$d = n - k + 1 \Leftrightarrow d + k = n + 1$$

The dual code of an MDS code is also an MDS code:

Theorem 4.2.7.
A linear code C is an MDS code iff. $C^{\perp}$ is an MDS code.

Proof:
Suppose C is a $[n, k, d]$ MDS code. A parity-check matrix H for C is a generator matrix for $C^{\perp}$, and any non-zero codeword $c \in C^{\perp}$ can be taken as a row of H. Every set of $d - 1 = n - k$ columns H are linearly independent, and so c must have less than $d - 1$ entries equal to zero. Thus $w(c)$ must be at least $n - (d - 1) + 1 = k + 1$ and since the singleton bound implies that this the minimum weight, $C^{\perp}$ is an MDS code. ■

By the proof of Theorem 4.2.7, the dual distance of an MDS code is $k + 1$.

The best known family of MDS codes are generalized Reed-Solomon codes.

Definition 4.2.8. Generalized Reed-Solomon code
Let $l \geq k \geq 1$. Let $x = (x_1, \ldots, x_l) \in \mathbb{F}_q^l$ be a tuple with $x_i \neq x_j$ for $i \neq j$, called

evaluation points and let $y = (y_1, \ldots, y_n) \in (\mathbb{F}_q^\times)^l$ be the column multipliers. The generalized Reed-Solomon code is given by:

$$GRS_k(x, y) = \{y_1 f(x_1), \ldots, y_l f(x_l), f \in \mathbb{F}_q[X], \deg f < k\}.$$

$GRS_k(x, y)$ codes are linear MDS codes of dimension k over $\mathbb{F}_q$. If $y = (1, \ldots, 1)$, then a standard Reed-Solomon code is obtained.

The interest in GRS codes comes from the fact, that they are more or less the only instances of MDS codes of dimension 2.

Lemma 4.2.9.
Let $2 \leq l \leq q$. All $[l, 2, l-1]$ MDS codes over $\mathbb{F}_q$ are GRS codes.

Proof:
Let C denote a $[l, 2, l-1]_q$ code with $2 \leq l \leq q$. C is a MDS code and thus has dual distance $d' = 3$. It is now claimed that there exists a codeword $c \in C$ with Hamming weight l. Let $G = (P_1, \ldots, P_l)$ be a generator matrix of C, where $P_i \in \mathbb{F}_q^2$ are columns. This leads to two observations: First, each point of P_i is non-zero, otherwise the dual distance would be 1. Second, $0, P_i, P_j$ are not the same for $i \neq j$, otherwise the dual distance would be 2. Notice that codewords in C are evaluations of bi-linear maps $\mu : \mathbb{F}_q^2 \to \mathbb{F}_q$ over $(P_1, \ldots, P_l)$, i.e. C can be written:

$$C = \left\{(\mu(P_1), \ldots, \mu(P_l)), \mu \in L(\mathbb{F}_q^2, \mathbb{F}_q)\right\},$$

and the P_i's are not all in the same line, otherwise $\dim C \leq 1$.

Since $l \leq q$, there exists $Q = (Q_0, Q_1) \in \mathbb{F}_q^2 \setminus \{0\}$ such that Q is not in the line defined by any of the P_i's. Define $\mu_Q(X, Y) = Q_1 X - Q_0 Y$, which is a non-zero bi-linear form and must vanish on a line of $\mathbb{F}_q^2$. Since $\mu_Q(Q) = 0$, it vanishes on the line spanned by Q. This means $\mu_Q(P_i) \neq 0$ for every $i \in [1, l]$ and $c = (\mu_Q(P_1), \ldots, \mu_Q(P_l))$ belongs to C and has Hamming weight l.

Finally, pick $u \in C$ such that $\{c, u\}$ spans C. The coordinate-wise product $(c_1 \times u_1, \ldots, c_l u_l)$ is denoted $c * u$ and $\vec{1}$ is the vector with all entries 1. Then $c = \vec{1} * c$ and $u = c * (c^{-1} * u)$, where c^{-1} is the coordinate wise inverse of c through $*$. Then C can be written $c * C'$ where C' has $G' = \binom{\vec{1}}{c^{-1} * u}$ as generator matrix. In conclusion, C is the GRS code with evaluation points $x = c^{-1} * u$, multiplies $y = c$ and dimension 2. ■

Lemma 4.2.9 has some interesting implications for transversal design and to explore this, an isomorphic map is defined: A map $\varphi : X \to X'$ is an isomorphism between the transversal

designs $(X, B, \mathfrak{G})$ and $(X', B', \mathfrak{G}')$ if it is one-to-one and preserves the incidence relations, i.e. if φ is invertible on the points, blocks and codes:

$$\varphi(X) = X' \quad \varphi(B) = B' \quad \varphi(\mathfrak{G}) = \mathfrak{G}'$$

The following lemma provides the characterization of isomorphisms between two transversal designs.

Lemma 4.2.10.
Let C, C' be two codes such that $C' = y * C$ for some $y \in (\mathbb{F}_q^{\times})^l$. Then the transversal designs they define, $\mathfrak{T}_C$ and $\mathfrak{T}_{C'}$ respectively, are isomorphic.

Proof:
Let the transversal designs from the two codes be written $\mathfrak{T}_C = (X, B, \mathfrak{G})$ and $\mathfrak{T}_{C'} = (X', B', \mathfrak{G}')$, by which $X = X' = \mathbb{F}_q \times [1, l]$ and $\mathfrak{G} = \mathfrak{G}' = \{\mathbb{F}_q \times \{i\}, 1 \leq i \leq l\}$. For the block sets, it is clear that $B = \{\{(c_i, i), 1 \leq i \leq l\}, c \in C\}$ and $B' = \{\{(y_i c_i, i), 1 \leq i \leq l\}, c \in C\}$. Consider:

$$\varphi_y : \quad \mathbb{F}_q \times [1, l] \to \mathbb{F}_q \times [1, l],$$

which maps:

$$(x_i, i) \mapsto (y_i x, i).$$

y is invertible over $*$ by which φ_y is one-to-one on X. Furthermore, since φ_y only acts on the first coordinate, it maps $\mathfrak{G}$ to itself. Finally $\varphi_y(B)$ is exactly B' since $C' = y * C$. ■

It is now possible to show that the study of two dimensional MDS codes can be restricted to $RS_2(x)$ Reed-Solomon codes.

Proposition 4.2.11.
Let $2 \leq l \leq q$ and $\mathbb{F}_p$ be any finite field. Any incidence code $IC(C_0)$ built from a $[l, 2, l-1]_q$ linear MDS code C_0 is permutation-equivalent to the incidence code $IC_p(RS_2(x))$, with $x \in \mathbb{F}_q^l, x_i \neq x_j$.

Proof:
By Lemma 4.2.9 all $[l, s, l-1]_q$ linear codes C_0 can be written $y \times RS_2(x)$ for $x \in \mathbb{F}_q^l$. Using the notation of the mapping φ_y, one can write $\varphi_y(\mathfrak{T}_{RS_2(x)}) = \mathfrak{T}_{y * RS_2(x)}$. That means $u \in IC_p(y * RS_2(x))$ iff. $u \in C_p(\varphi_y(\mathfrak{T}_{RS_2(x)}))$. Consider a new map given by:

$$\tilde{\varphi}_y : \quad \mathbb{F}_p^X \to \mathbb{F}_p^X,$$

which maps:

$$u = (u_x)_{x \in X} \mapsto (u_{\varphi_y(x)})_{x \in X}$$

It is not hard to observe that $\tilde{\varphi}_y(IC_p(RS_2(x))) = C_p(\varphi_y(\mathfrak{T}_{RS_2(x)}))$ and $\tilde{\varphi}_y$ is a permutation of coordinates. Thus $IC_p(C_0)$ is permutation-equivalent to $IC_p(RS_2(x))$ and the proof is complete. ■

It turns out that the incidence code $IC_p(RS_2(\mathbb{F}_q))$ is equal to another type of code; codes based on transversal designs built from affine geometries. The construction of transversal designs from affine geometries has not been explored and thus only some simple observations will be made. Firstly, the actual proposition that contains the equivalence:

Proposition 4.2.12.
Let $\mathfrak{T}_A(2, q)$ be the transversal design built from the affine space of dimension 2 over $\mathbb{F}_q$. Such a design is called an affine design. See [1, p. 7] for a formal definition. Let $IC_q(RS_2(\mathbb{F}_q))$ be the incidence code over $\mathbb{F}_q$ of the full-length Reed-Solomon code of dimension 2 over $\mathbb{F}_q$.
The codes

- $C_1 = IC_q(RS_2(\mathbb{F}_q))$,
- C_2, the code over $\mathbb{F}_q$ based on the transversal design build from the affine plane $A^2(\mathbb{F}_Q)$ $\mathfrak{T}_A(2, q)$,

are equal up to permutation.

Proof:
Proof omitted. ■

This means that C_1 and C_2 have the same rate and dimension considerations. For a further exploration of the affine geometry construction, see [1, p. 7-8].

4.2.2 Divisible Codes

As it turns out, some very good incidence codes arise from linear codes C_0 that satisfy a certain divisibility condition. The important part is that the incidence codes will have rate approximately greater than $\frac{1}{2}$. The divisibility condition that C_0 must follow is given by:

Definition 4.2.13. Divisibility of a code
Let $p \geq 2$. A linear code is p-divisible if p divides the Hamming weight of all its codewords.

By considering the incidence matrix which defines an incidence code, the following lemma can be derived:

Lemma 4.2.14.
Let C_0 be a code of length l over a set S, $\mathfrak{T}_{C_0}$ be the transversal design associated to C_0, and $d(x, y)$ be the Hamming distance between x and y. Denote by M_{C_0} the incidence matrix of $\mathfrak{T}_{C_0}$, where the rows of M_{C_0} are indexed by codewords from C_0. Then:

$$(MM^T)_{c,c'} = l - d(c, c') \quad \forall c, c' \in C_0$$

Proof:
Let the notation $M[c, (\alpha, i)]$ be the entry of M indexed by the codeword $c \in C_0$ for the row and $(\alpha, i) \in S \times [1, l]$ for the column. Further, let $\vec{1}_{u(c,i,\alpha)} \in \{0, 1\}$ denote the Boolean value of the property u. For $c, c' \in C_0$, then:

$$(MM^T)_{c,c'} = \sum_{\alpha \in S, i \in [1,l]} M[c, (\alpha, i)] M[c', (\alpha, i)] = \sum_{\alpha \in S, i \in [1,l]} \vec{1}_{c_i = \alpha} \vec{1}_{c'_i = \alpha}$$
$$= \sum_{i=1}^{l} \sum_{\alpha \in S} \vec{1}_{c_i = c'_i = \alpha} = \sum_{i=1}^{l} \vec{1}_{c_i = c'_i} = l - d(c, c').$$

■

Lemma 4.2.14 means that, if some prime p divides l as well as the Hamming weight of all the codewords in C_0, then MM^T vanishes over any extension of $\mathbb{F}_p$, and M is a parity-check matrix of a code containing its dual. In more general terms, the following proposition holds:

Proposition 4.2.15.
Let C_0 be a linear code of length l over S, with $|S| = s$. Furthermore let C be the incidence code $IC_q(C_0)$, where $\mathbb{F}_q$ has characteristic p. Denote by $n = ls$ the length of C. If C_0 is p-divisible and C_{par} denotes the parity-check code of length n over $\mathbb{F}_q$, then

$$C^{\perp} \cap C_{par} \subseteq C.$$

This means $\dim C \geq \frac{n-1}{2}$ and if p divides l, then $C^{\perp} \subseteq C$ and $\dim C \geq \frac{n}{2}$.

Proof:
Let M be the incidence matrix of $\mathfrak{T}_{C_0}$ and denote by J and J' the matrices with all entries

1, with sizes $|C_0| \times n$ and $|C_0| \times |C_0|$ respectively. If C_0 is assumed to be p-divisible, then by Lemma 4.2.14:

$$MM^T = lJ' \text{mod}(p)$$

Furthermore, by an easy calculation:

$$MJ^T = lJ'$$

Over $\mathbb{F}_q$ this means

$$M(M-J)^T = 0,$$

by which it is natural to consider the code A of length n generated over $\mathbb{F}_q$ by the matrix $M - J$, where $A \subseteq C$. Let $C_{par} := \left\{c \in \mathbb{F}_q^n, \sum_i c_i = 0\right\}$ be the parity-check code of length n over $\mathbb{F}_q$. It follows that $c \in C_{par} \Leftrightarrow cJ^T = 0$ and $uJ = 0 \Leftrightarrow uJ' = 0$. If p does not divide l, then:

$$\begin{aligned} C^\perp \cap C_{par} &= \left\{c = uM \in \mathbb{F}_q^n, cJ^T = 0\right\} = \left\{c = uM \in \mathbb{F}_q^n, luJ' = 0\right\} \\ &= \left\{c = uM \in \mathbb{F}_q^n, uJ = 0\right\} = \left\{u(M-J) \in \mathbb{F}_q^n, uJ = 0\right\} \\ &\subseteq A \subseteq C \end{aligned}$$

The dimension is a simple calculation:

$$\dim C \geq \dim(C^\perp \cap C_{par}) \geq \dim C^\perp - 1 = n - \dim C - 1 \Leftrightarrow \dim C \geq \frac{n-1}{2}$$

In the case of p dividing l, then $lJ' \text{mod}(p) = 0$, so $MM^T = 0$ by which $C^\perp \subseteq C$. For the dimension:

$$\dim C \geq \dim C^\perp = n - \dim C \Leftrightarrow \dim C \geq \frac{n}{2}$$

■

For PIR protocols, the following corollary can be derived:

Corollary 4.2.16.
Assume there exists a p-divisible code of length l_0 over $\mathbb{F}_q$ with p a prime. Then a distributed PIR protocol can be build for a k-entries database over F_q with $k \geq (l_0 q - 1)/2$. The Protocol has parameters $l(k) = l_0$ and $n(k) = l_0 q \leq 2k + 1$.

The conclusion is that one would like to find divisible codes C_0 defined over large alphabets compared to the code length.

4.3 (t-1)-private PIR Protocol Construction From t-transversal Designs

Simple transversal designs do not suffice in constructing PIR protocols that can protect against colluding servers. The solution is orthogonal arrays: As shown in Proposition 4.2.2, it is easy to build a transversal design from a strength two orthogonal array. The idea can be used higher strength orthogonal arrays, which leads to the concept of t-transversal designs:

Definition 4.3.1. t-transversal designs
A t-transversal design is a block design $\mathfrak{D} = (X, B)$ equipped with a group set $\mathfrak{G} = \{G_1, \ldots, G_l\}$, with $l \geq t \geq 1$, for which:

- $|X| = sl$,
- ans group has size s and any block has size l,
- for any $T \subseteq [1, l]$ with $|T| = t$ and for any $(x_1, \ldots, x_t) \in G_{T_1} \times \cdots \times G_{T_t}$, there exists exactly λ blocks $b \in B$ such that $\{x_1, \ldots, x_t\} \subset b$.

Such a design is denoted $t\text{-}TD_\lambda(l, s)$ or $t\text{-}TD(l, s)$ with $\lambda = 1$.

The way to derive a t-transversal design from a strength t orthogonal array is shown in Section 4.3.1. Clearly, Definition 3.0.4 is the definition of a 2-transversal design, and since such a design makes a 1-private protocol, a t-design will make a $(t-1)$-private PIR protocol. The construction of the $(t-1)$-private PIR protocol is identical with 1-private PIR protocol, i.e. define the code $C_q(\mathfrak{T})$ associated to the t-transveral design and run Construction 4.1.2. A t-transversal design is also a 2-transversal design for $t \geq 2$, by which, correctness, communication complexity, computation complexity and storage overhead is identical for 1-private and $(t-1)$-private PIR protocols. This means only security remains, but it is fairly simple:

By using the same approach as establishing that there are λs^2 blocks in B of the 2-transversal design, there must be λs^t blocks in B of t-transversal design. Now, let T be a collusion of servers of size $|T| \leq t - 1$. For varying $i \in K$, the distributions $Q(i)_{|T}$ are the same because there are exactly $\lambda s^{t-1-|T|} \geq \lambda \neq 0$ blocks containing both i and the queries known by the servers in T.

In summation, the following theorem holds:

Theorem 4.3.2.
Let D be a database with k entries over $\mathbb{F}_q$, and $\mathfrak{T} = t\text{-}TD(l, s)$ be a t-transversal design, whose incidence matrix has rank $ls - k$ over $\mathbb{F}_q$. Then, there exist an l-server

$(t-1)$-private PIR protocol with:

- Only one $\mathbb{F}_q$ symbol to read for each server,
- $l-1$ field operations over $\mathbb{F}_q$ for the user,
- $l\log(sq)$ bits of communication,
- a storage overhead of $(ls-k)\log q$ bits on the servers.

4.3.1 t-transversal Designs from Strength t Orthogonal Arrays

Since the strength t orthogonal arrays gave rise to the idea of the t-transversal design, it is only appropriate to show how to actually obtain such a design. First is a definition:

Definition 4.3.3.
Let A be an orhogonal array $OA_\lambda(t,l,s)$ on a symbol set S. Here, A is composed of rows $a_i = (a_{i.j})_{1\leq j\leq l}$ for $1 \leq i \leq \lambda s^t$. From this comes a design $\mathfrak{T} = (X, B, \mathfrak{G})$, where each set is given by:

- $X = S \times [1,l]$,
- $b_i = \{(a_{i,j}, j), 1 \leq j \leq l\}$ for all $a_i \in \text{Rows}(A)$,
- $\mathfrak{G} = \{S \times \{i\}, 1 \leq i \leq l\}$.

To show that the design defined from the orthogonal array A by Definition 4.3.3, one simply analyzes each set X, B and $\mathfrak{G}$:

Proposition 4.3.4.
If A is an $OA_\lambda(t,l,s)$ orthogonal array, then the design defined with A by Definition 4.3.3 is a t-$TD_\lambda(l,s)$ design.

Proof:
It is easy to see that $\mathfrak{G}$ is a partition of X and the blocks and groups have the claimed size. For the incidence property, let $T \subset [1,l]$ with $|T| = t$ and let $(x_1,\ldots,x_t) \in G_{T_1} \times \cdots \times G_{T_t}$. There must be exactly λ blocks $b \in B$ such that $\{x_1,\ldots,x_t\} \subset b$. Consider the map:

$$\psi : B \to \text{Rows}(A)$$

Given by the mapping:

$$b_i = \{(a_{i,j}, j), 1 \leq j \leq l\} \mapsto (a_{i,1},\ldots,a_{i,l})$$

By the assumption that the orthogonal arrays have no repeated rows, ψ is one-to-one. Now, denote by $x' = (x'_1, \ldots, x'_t) \in S^t$ the vector formed by the first coordinates of $(x_1, \ldots, x_t) \in X^t$. By Definition 4.2.1 x' appears exactly λ times in the submatrix of A defined by the columns indexed by T. So x' defines λ pre-images in B, and the proof is complete. ■

Using the orthogonal arrays, a very interesting corollary can be established:

Corollary 4.3.5.
Let C_0 be a code of length l and dual distance $t + 2 \leq l$ over a set S, with $|S| = s$. Then the incidence code $IC_q(C_0)$ defines a t-private PIR protocol.

Proof:
Let A be the orthogonal array defined by C_0. A has strength $t+1$, hence Proposition 4.3.4 implies that the associated transversal design is a $(t+1)$-$TD(l, s)$ design. Theorem 4.3.2 ensures that the PIR protocol induced by this transversal design is t-private. ■

5 Scheme Analysis

In this chapter a comparison will be made between the transversal design based PIR scheme and another PIR scheme, to illustrate how the design based PIR scheme differentiates itself. An example of the design based PIR scheme will also be constructed, and from this an interesting property of the orthogonal array construction will be derived. This chapter is based on [1] and [2].

5.1 Construction Example

To illustrate the PIR scheme construction, an example that follows Construction 4.2.4 will be made. In this example the generic code C_0 will be the dimension 2 Reed-Solomon code over $\mathbb{F}_5$.

Example 5.1.1. Scheme Construction
Let C_0 be the linear [5,2,4] Reed-Solomon code over the field $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$. The orthogonal array from this code, is simply all the codewords listed in a matrix.

This results in an $OA(5,5)$ orthogonal array, with entries:

$$\begin{pmatrix}
0, & 0, & 0, & 0, & 0, \\
1, & 1, & 1, & 1, & 1, \\
2, & 2, & 2, & 2, & 2, \\
3, & 3, & 3, & 3, & 3, \\
4, & 4, & 4, & 4, & 4, \\
0, & 1, & 2, & 3, & 4, \\
0, & 2, & 4, & 1, & 3, \\
0, & 3, & 1, & 4, & 2, \\
0, & 4, & 3, & 2, & 1, \\
1, & 2, & 3, & 4, & 0, \\
1, & 3, & 0, & 2, & 4, \\
1, & 4, & 2, & 0, & 3, \\
1, & 0, & 4, & 3, & 2, \\
2, & 3, & 4, & 0, & 1, \\
2, & 4, & 1, & 3, & 0, \\
2, & 0, & 3, & 1, & 4, \\
2, & 1, & 0, & 4, & 3, \\
3, & 4, & 0, & 1, & 2, \\
3, & 0, & 2, & 4, & 1, \\
3, & 1, & 4, & 2, & 0, \\
3, & 2, & 1, & 0, & 4, \\
4, & 0, & 1, & 2, & 3, \\
4, & 1, & 3, & 0, & 2, \\
4, & 2, & 0, & 3, & 1, \\
4, & 3, & 2, & 1, & 0,
\end{pmatrix}$$

By Proposition 4.2.2 this orthogonal array can be turned into an $\mathfrak{T}_{C_0} = TD(5,5)$ transversal design, where the sets $(X, B, \mathfrak{G})$ are:

$$\begin{aligned}
X &= \{(\beta, i), i \in [1,5], \beta \in \mathbb{F}_5\} \\
B &= \{\{(c_i, i), i \in [1,5]\}, c \in \text{Rows}(OA(5,5))\} \\
\mathfrak{G} &= \{\{0,1,2,3,4\} \times \{i\}, i \in [1,5]\}
\end{aligned}$$

As a simple example, the ninth row $a_{10} = (1,2,3,4,0)$ is turned into the block $\{(1,1),(2,2),(3,3),(4,4),(0,5)\}$. Each column correspond to a group, with the first column corresponding to the group with the elements $\{(0,1),(1,1),(2,1),(3,1),(4,1)\}$.

The incidence code of this transversal design has length 25, and so to consider the blocks as words in $\{0,1\}^{25}$, the elements of X are ordered like this $(0,1),(1,1)...(4,1),(0,2)(1,2)....$ Thus the ninth row can be turned into the incidence

vector:

$$1_{b_9} = (0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,1,0,0,0,0)$$

The incidence matrix of $\mathfrak{T}_{C_0}$ is:

$$M_{\mathfrak{T}_{C_0}} = \begin{pmatrix}
10000 & 10000 & 10000 & 10000 & 10000 \\
01000 & 01000 & 01000 & 01000 & 01000 \\
00100 & 00100 & 00100 & 00100 & 00100 \\
00010 & 00010 & 00010 & 00010 & 00010 \\
00001 & 00001 & 00001 & 00001 & 00001 \\
10000 & 01000 & 00100 & 00010 & 00001 \\
10000 & 00100 & 00001 & 01000 & 00010 \\
10000 & 00010 & 01000 & 00001 & 00100 \\
10000 & 00001 & 00010 & 00100 & 01000 \\
01000 & 00100 & 00010 & 00001 & 10000 \\
01000 & 00010 & 10000 & 00100 & 00001 \\
01000 & 00001 & 00100 & 10000 & 00010 \\
01000 & 10000 & 00001 & 00010 & 00100 \\
00100 & 00010 & 00001 & 10000 & 01000 \\
00100 & 00001 & 01000 & 00010 & 10000 \\
00100 & 10000 & 00010 & 01000 & 00001 \\
00100 & 01000 & 10000 & 00001 & 00010 \\
00010 & 00001 & 10000 & 01000 & 00100 \\
00010 & 10000 & 00100 & 00001 & 01000 \\
00010 & 01000 & 00001 & 00100 & 10000 \\
00010 & 00100 & 01000 & 10000 & 00001 \\
00001 & 10000 & 01000 & 00100 & 00010 \\
00001 & 01000 & 00010 & 10000 & 00100 \\
00001 & 00100 & 10000 & 00010 & 01000 \\
00001 & 00010 & 00100 & 01000 & 10000
\end{pmatrix}$$

This is a 25×25 matrix. An interesting observation is that the incidence matrix can be obtained directly from $OA(5,5)$ by substituting $0,1,2,3,4$ with the tuples $(10000),(01000),(00100),(00010),(00001)$ respectively.

The incidence code has the codewords of M as the dual code, and M has rank 21. Then the incidence code has a rank 4 generator matrix, and by either using gaussian elimination or the fact that it is a transversal design, the incidence code has the following

generator matrix:

$$G^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

So this is a $[25, 4]$ code. It can encode a $D \in \mathbb{F}_2^4$ database.

This example brings out the importance of how the field characteristic affects the code dimension, as shown by Proposition 4.1.4. The characteristic of $\mathbb{F}_2$ is of course 2 and this does not divide $\lambda s = 5$. Hence this code having only dimension 4, compared to a length of 25. This means the storage overhead becomes $21 \log(2)$ bits, which is unnecessarily large for such a small encodable database.

An interesting counterpoint exist with the PIR protocol built from the Reed-Solomon code of dimension 2 over $F_4 = \{0, 1, \alpha, \alpha + 1\}$. The incidence matrix of the transversal

design resulting from the corresponding orthogonal array is:

$$M = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}$$

This matrix has rank 9, so the dual code has dimension 9, by which the incidence code has dimension 7. The generator matrix of the incidence code over $\mathbb{F}_2$ is:

$$G = \begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}$$

This can encode a $D \in \mathbb{F}_2^7$ database and has a storage overhead of $9\log(2)$ bits, which is a big improvement.

So when working with the generic-to-incidence construction, it is important to consider the transversal design obtained from the generic code, since it has important implications on the incidence code.

There is an interesting observation from Example 5.1.1, about the orthogonal code construction that uses GRS code: The two transversal designs have group size s equal to the the size of the field the GRS code is defined over: The $\mathbb{F}_4$ GRS code gives a design with size four groups, while the $\mathbb{F}_5$ GRS code design has size five groups. Thus Proposition 4.1.4 comes into play: If one wants to encode a database with entries in $\mathbb{F}_2$, one then shouldn't choose a prime field for the Reed-Solomon code, since the characteristic of $\mathbb{F}_2$ never divides

a prime. This means that one should not use GRS codes defined over a prime field when using the orthognal array construction.

5.2 A General PIR Scheme for Coded Storage with Colluding Servers

In the interest of comparing the transversal design PIR scheme, a general PIR scheme for coded storage with colluding servers is introduced. This scheme will be called the general scheme for brevity. The general scheme is very heavily based on GRS codes and how the database is coded. An important part of this scheme, and many other schemes, is the star product between two vector spaces:

Definition 5.2.1. Star product
Let $V, W \subseteq \mathbb{F}_q^n$ be vector spaces. The star product between V and W is the subspace of $\mathbb{F}_n^q$ generated by the Hadamard products:

$$v \star w = [v_1 w_1, \cdots, v_n w_m],$$

for all pairs $v \in V$ and $w \in W$.

The star product is very important due to the following proposition:

Proposition 5.2.2.
Let $V, W \subseteq \mathbb{F}_q^n$ be linear codes. The star product $V \star W$ satisfies:

(i) If $V \subseteq \mathbb{F}_q^n$ and $W = \text{Rep}(n) \subseteq \mathbb{F}_q^n$, then $V \star W = V$.

(ii) If $V, W \subseteq \mathbb{F}_q^n$ with $\text{supp}(V) = \text{supp}(W) = [n]$ and $(V \star W)^\perp = H$, then $d_H \geq d_{V^\perp} + d_{W^\perp} - 2$.

(iii) If V is an MDS code, then $(V \star V^\perp)^\perp = \text{Rep}(n)$

(iv) If $V = \text{GRS}_i(\alpha, v)$ and $W = \text{GRS}_j(\alpha, w)$, then $V \star W = \text{GRS}_{\min\{i+j-1,n\}}(\alpha, v \star w)$ for any parameters v, w.

Proof:
Each property is proved in turn:

(i) This follows directly from Definition 5.2.1.

(ii) See [10].

(iii) To show this, let $H = (V \star V^\perp)^\perp$. Obviously $\text{Rep}(n) \subseteq H$, and by using that C is

an $[n, k]$ MDS code, $C^{\perp}$ is an $[n, n-k]$ MDS code, the previous property, and the singleton bound, it can be seen that $H = \text{Rep}(n)$.

(iv) This property is proved by first showing that $\text{GRS}_i(\alpha, v) \star \text{GRS}_j(\alpha, w) \subseteq \text{GRS}_{\min\{i+j-1,n\}}(\alpha, v \star w)$, and then the converse. The first part is done simply by taking two arbitrary code words from each $V = \text{GRS}_i(\alpha, v)$ and $W = \text{GRS}_j(\alpha, w)$, and then showing that their star product is in $\text{GRS}_{\min\{i+j-1,n\}}(\alpha, v \star w)$. The converse comes from the fact $\text{GRS}_{\min\{i+j-1,n\}}(\alpha, v \star w)$ is generated as an $\mathbb{F}_q$-vector space by codewords containing a monomial of degree $m \leq i + j - 1$. These codewords can then be decomposed into the star product between two differnt codewords, each containing a monomial of degree $a < i, b < j$ and each an element of $\text{GRS}_i(\alpha, v)$ and $\text{GRS}_j(\alpha, w)$ respectively.

■

The main benefit of the general scheme is its high rate. To make clear notation, from this point forward subscripts refer to servers and superscripts refer to files. Assume that the files of the database are $c^1, \ldots, c^m \in \mathbb{F}_q^{b \times k}$, where b is a parameter that will be adjusted to enable the users retrieval of exactly one whole file. The data is stored in a $bm \times k$ matrix D. The files are encoded with a linear $[n, k, d]$-code C by using its generator matrix G_C, by which the encoded database is:

$$Y = DG_C = \begin{bmatrix} y_1^1 & \cdots & y_n^1 \\ \vdots & \ddots & \vdots \\ y_1^m & \cdots & y_n^m \end{bmatrix}$$

The column $y_j \in \mathbb{F}_q^{bm \times 1}$ is sent to the j'th server. This encoding type has a very good benefit: Any $d_C - 1$ servers can fail, and a user can still successfully retrieve any file x^i.

The scheme itself is iterative, and the number of iterations of such a scheme is given by z. The rate of the PIR scheme is then given by:

Definition 5.2.3. Rate
The rate of the PIR scheme for coded storage with colluding servers is $\frac{bk}{nz}$.

The scheme starts with two codes:

- The linear $[n, k, d]$-code C over $\mathbb{F}_q$ with generator matrix G_C, from which a distributed storage system $Y = DG_C$ can be defined.
- A linear code $R \subseteq \mathbb{F}_q^n$ called the retrieval code. This code will determine the privacy properties of the scheme.

The scheme is iterative, where with each iteration a certain number of symbols are downloaded. Fix a subset $J \subseteq [n]$ of servers with constant size $|J| = \max\{c, k\}$, where

$c = d_{D\star R} - 1$. J is the set of all servers from which the encoded symbols are obtained. During the u'th iteration of the procedure the symbol $y_j^i(a)$ is obtained from every server $j \in J_u^a$, where $J_u^a \subseteq J$, with $u \in [z] = \left[\frac{\text{lcm}(c,k)}{c}\right]$ and $a \in [b] = \left[\frac{\text{lcm}(c,k)}{k}\right]$. c is defined as $d_{C\star R} - 1$. The scheme construction works in the following steps:

Query generation, Q: mb codewords $d^{h,a} = [d^{h,a}(1) \dots d^{h,a}(n)]$ are selected uniformly at random from R for $h \in [m]$ and $a \in [b]$, with m being the number of files in the database. For $h \in [m]$ and $j \in [n]$, define:

$$d_j^h = [d^{h,1}(j) \dots d^{h,b}(j)] \in \mathbb{F}_q^{1\times b}$$
$$d_j = [d_j^1 \dots d_j^m] \in \mathbb{F}_q^{1\times mb}$$

At each iteration, $J_u := [c] \in J$ is partitioned into b subsets, where the first iteration is:

$$J_1^1 = \left\{1, \dots, \frac{c}{b}\right\}, J_1^2 = \left\{\frac{c}{b} + 1, \dots, \frac{2c}{b}\right\}, \dots, J_1^b = \left\{(b-1)\frac{c}{b}, \dots, b\frac{c}{b} = c\right\}$$

For the other iterations $u = 2, \dots, z$, $J_u^a \subseteq J$ is recursively defined to be the cyclic shift of J_{u-1}^a within J to the right by $g = \frac{c}{b}$ indices. So, if $J_{u-1}^a = \{j_1, \dots, j_g\}$, then:

$$J_u^a = \{j_1 + g, j_2 + g, \dots, g\}$$

Lastly, let $J_u = J_u^1 \cup \dots \cup J_u^b$, by which the queries are given by:

$$q_j^i = \begin{cases} d_j + e_{b(i-1)+a} & \text{if } j \in J_u^a \\ d_j & \text{if } j \notin J_u \end{cases}$$

Here, $e_{b(i-1)+a} \in \mathbb{F}_q^{1\times mb}$ is the $(b(i-1)+a)$'th standard basis vector. For $j \in J_u^a$, the query q_j^i is d_j, but with entry $d^{i,j}(j)$ replaced with $d^{i,a}(j) + 1$.

Answer from the servers, A: A response vector is calculated:

$$r^i = (\text{codeword of } C \star R) + y_{J_u}^i$$

$y_{J_u}^i$ is a vector with entries $y_j^i(a)$ in known positions for all $j \in J_u^a$ and all $a \in [b]$, and zeroes elsewhere.

Reconstruction, R: Let S be a generator matrix of $(C \star R)^\perp$. Every c columns of S are linearly independent, by the definition of c. Assume the file x^i must be reconstructed, so consider the response vector r^i from the first iteration, and then compute:

$$Sr^i = S(\text{codeword of } C \star R) + Sy^i = Sy^i$$

This way, the values $y_1^i(1), \dots, y_c^i(b)$ are obtained. For the u'th iteration of the scheme all entries of the form $y_j^i(a)$ for $j \in J_u^a$ and $a \in [b]$ are obtained.

5.2.1 Analysis

Analysis of this scheme follows along the lines of the design based PIR scheme, but it is a bit different.

During each iteration of the scheme,

$$g := \frac{k}{z} = \frac{c}{b}$$

symbols are downloaded from every row of y^i. So after z iterations the scheme will have downloaded $zg = k$ symbols of the row $y^{i,a}$ of y^i for all $a \in [b]$.

For correctness the following theorem applies:

Theorem 5.2.4. Correctness
Let C be a $[n, k, d]$-code and R be retrieval code, such that:

- $d_{C\star R} - 1 \leq k$, or
- there exists $J \subseteq [n]$ of $|J| = \max\{d_{C\star R} - 1.k\}$, where every $K \subset J$ with $|K| = k$ is an information set of C.

Then the general PIR scheme is correct, i.e. the desired file is retrieved with rate $\frac{(d_{C\star D}-1)}{n}$.

Proof:
If the first case is satisfied, one simply chooses $J \subseteq [n]$ of size k to be any information set of C, and so a proof that shows the second condition satisfies both. During the reconstruction algorithm, k symbols from each row $y^{i,a}$ of y^i are retrieved. Since every $K \subseteq J$ of $|K| = k$ is an information set, it suffices to recover every $y^{i,a}$ and therefore all of x^i. The rate is:

$$\frac{k \cdot \frac{\text{lcm}(c,k)}{k}}{n \cdot \frac{\text{lcm}(c,k)}{c}} = \frac{d_{C\star R} - 1}{n}$$

■

The security is a bit more involved.

Theorem 5.2.5. Privacy
The general PIR scheme protects against $d_{R^\perp} - 1$ colluding servers.

Proof:
Let $T = \{j_1, \ldots, j_t\} \subseteq [n]$ be a set of severs with $|T| = t \leq d_{R^\perp} - 1$. The first step is

to show that during a single iteration of the scheme, the mutual information between the queries and the index is zero, i.e. $I(q^i_{j_1}, \ldots, q^i_{j_t}; i) = 0$. Since $t \leq d_{D^\perp} - 1$ every t columns of R's generator matrix are linearly independent, so the code R_T is the entire space $\mathbb{F}_q^t$. Now, consider the distribution of the vectors

$$d_j = [d^{1,1}(j) \ldots d^{1,b}(j) \ldots d^{m,1}(j), \ldots d^{m,b}(j)] \in \mathbb{F}_q^{1 \times bm}$$

for a single $j \in T$. $R_{\{j\}}$ is distributed uniformly on $\mathbb{F}_q$ and the codewords $d^{h,a}$ are selected uniformly at random from R, so d_j is uniform on $\mathbb{F}_q^{1 \times bm}$. Furthermore, since R_T is all of $\mathbb{F}_q^t$, the joint distribution $\{d_j \mid j \in T\}$ is uniform over $(\mathbb{F}_q^{1 \times bm})^t$. If $f(i,j)$ denotes the index of the standard basis vector, then the queries

$$\{q^i_{j_1}, \ldots, q^i_{j_t}\} = \{d_j + e_{f(i,j)} \mid j \in T \cap J\} \cup \{d_j \mid j \in T \setminus J\}$$

are uniformly distributed for all i, since translating the uniform distribution by a vector gives the uniform distribution again. That means the distribution of queries is independent of i, by which $I(q^i_{j_1}, \ldots, q^i_{j_t}; i) = 0$ for a single iteration.

For all iterations of the scheme, consider the joint distribution Q^i_T of all queries to all servers in T, as all iterations are gone through. During each iteration, the vectors $d^{h,a}$ are chosen independently of all other iterations, by which Q^i_T is uniform $(\mathbb{F}_q^{1 \times bm})^{tz}$. Then the joint distribution is independent of the index, i.e. $I(Q^i_T; i) = 0$ and the proof is complete. ■

5.3 Scheme Comparison

Besides being based on two different ideas, there are four main points in which the two schemes are different: Rate, Storage, complexity and server error handling. Server error handling will not be explored in detail, but is still touched upon.

5.3.1 Rate

A PIR schemes rate is very important, since high rate schemes result in very efficient schemes, as the entire point of PIR schemes is to retrieve information. The rate of the design based PIR schemes is of course very simple, it is simply the retrieved information over the encoded information $\frac{k}{n}$. Since b and s can be chosen as pleased in the general PIR scheme, this is of course similar to Definition 5.2.3. But as shown in Theorem 5.2.4, the rate of the general PIR scheme can be expressed as $\frac{d_{C \star R} - 1}{n}$.

As shown in Example 5.1.1 the rate of the transversal design based PIR scheme is very dependent on the interaction between the block size and the characteristic of the field the database will be encoded in. If the characteristic does not divide the group size, the rate will be at most $\frac{l}{ls} = \frac{1}{s}$. There is also the divisibility condition as shown in Section 4.2.2, with the condition satisfied ensuring a rate above $\frac{1}{2}$.

To get high rate PIR protocols then requires a different approach for each scheme: The design based scheme should be build around the divisibility condition and the characteristic of the databases field, and the general scheme should be build around the star product. It is interesting that they both center on GRS-codes: For the 1-private transversal design PIR protocol, all generic codes are dimension 2 RS codes. Similarly, because of Proposition 5.2.2, some very good PIR protocols can be achieved when C is a GRS-code.

5.3.2 Storage

Storage overhead is the amount of information stored on the servers to make the PIR protocol work. This is important, since the storage could potentially be so large, that the protocol is rendered moot by the effort required to keep these servers running. An equivalent concept is the storage rate.

The general scheme has an equal amount of servers to the length of the encoding code n, because each column in the encoded matrix Y represents a server. Each row of y contains k symbols of information, which means that the amount of extra symbols of a single row is $n-k$. The number of entries on each server is shaped by bm. b is negligible, and thus only the number of files describe how many entries there are on each server. The system then stores a total of mn F_q symbols, of which km are information symbols. So the total storage overhead is $m(n-k)$ of symbols in $\mathbb{F}_q$ and equivalently $n-k$ storage overhead per file.

As described in Theorem 4.1.3 and Theorem 4.3.2 the storage overhead of $(ls-k)\log(q)$ bits. There are $ls = n$ elements in the transversal design, which means that the stored codeword has length n, and of these k are information symbols on the servers. The problem is that this is not really comparable with the general PIR scheme, since each file is a symbol in $\mathbb{F}_q^k$. So to make the comparison, one takes the files to be $\mathbb{F}_{q^k}$ symbols, such that the database is a string in $\mathbb{F}_{q^k}^m$. To make sure that the schemes are exactly the same, it would be needed that the number of servers used are the same for each scheme. It is very hard to determine if this is possible, but for the sake of argument, this is assumed. There are l servers each storing s $\mathbb{F}_{q^k}$ symbols. This means each server stores sk $\mathbb{F}_q$ symbols, and so the system stores a total of lsk symbols. The system encodes mk symbols of information, and thus the total storage overhead is $lsk - mk$ symbols of $\mathbb{F}_q$ and equivalently a $\frac{lsk}{m} - k$ storage overhead per file.

These can also be represented as rates, with the general scheme having a storage rate of $\frac{k}{n}$ and the TD based scheme has storage rate $\frac{m}{ls}$. Both have fairly good storage capabilities, but it is easier to store larger files in the general scheme, so it is considered slightly superior.

5.3.3 Complexity

Complexity is very important in PIR schemes, since simple PIR schemes are preferable to other schemes. The transversal design scheme has a very low amount of computational

complexity for the servers: Each server simply reads it's query and sends back a response without any further calculations. The user needs only compute the queries and that is very easy, based on the properties of the transversal design.

The general scheme's servers need to compute a response vector:

$$r^i = (\text{codeword of } C \star R) + y^i_{J_u}$$

r^i_j is computed differently based on whether $j \in J^{a_0}_u$ or $j \notin J_u$. In the latter case:

$$r^i_j = \langle q^i_j, y_j \rangle = \sum_{h=1}^{m} \langle d^h_j, y^h_j \rangle = \sum_{h=1}^{m} \sum_{a=1}^{b} d^{h,a}(j) y^h_j(a).$$

And in the other case:

$$r^i_j = \sum_{h=1}^{m} \sum_{a=1}^{b} d^{h,a}(j) y^h_j(a) + y^i_j(a_0)$$

This means that each server performs at least one star product operation and if $j \in J^{a_0}_u$ the server also performs an addition operation. The conclusion is that the design based scheme has a very low amount of complexity.

5.3.4 Operation

Operation of the scheme is also an important consideration. Here the focus is on how easy or hard it is to change, add or delete files of the original database. It is very easy to do in the general scheme, since the encoding is very simple: If a file is added, a file can easily be encoded and added to the servers. If a file is deleted, it is known exactly which row is deleted in the encoding, and these symbols are simply deleted from the servers. Changing a file is simply changing the encoding.

It is not simple to do the same for the TD based PIR scheme. When a file is changed, a new encoding of the entire database needs to be computed. Deleting and adding a file is very complicated: It is possible to delete a file by simply substituting it with a file of no information and retain the scheme. But if a file is to be added, the scheme needs to be redesigned.

5.3.5 Server Failures

One of the main benefits of distributed PIR schemes, is that they can be designed, such that if a server fails, the information can be reconstructed from the other servers. The general scheme excels at this, where $n - k$ servers can fail, and it is still possible to use the PIR protocol. The transversal design based PIR scheme has a very limited capability in this regard. It would be possible to reconstruct one server if one fails, since it would be obvious what is missing from the transversal designs set. But resistance against a larger amount of server failures haven't been explored.

6 | Discussion and Conclusion

Discussion

Construction Considerations

In this project extraneous ways of creating transversal designs from other mathematical structures have not been included. The two most important of these are the construction using affine planes and the construction using projective geometries. These have not been concluded because so as to not clutter the project, as well as the orthogonal array construction being more interesting. As Proposition 4.2.12 describes, 1-private PIR schemes based on the transversal design the affine and orthogonal constructions are essentially the same. The projective geometry allows one to use a result from Hadamard, with which one can compute the p-rank of the code based on the projective geometry design. The two constructions are asymptotically the same. as can be proven by experimentation.

The divisibility condition could also be explored more, since it has important implications for the rate of the orthogonal array construction. A direct link between the divisibility condition and the GRS code length observation would be an interesting further investigation.

Server Error Considerations

Server errors are broad, but to simplify, in this project they are when a server fails in some way. The general PIR scheme has great measures against server errors, as described in Chapter 5. The general PIR scheme is maximally robust against failures, if the encoding code C is an MDS code, i.e. it protects against $d_{C\star D} - 1$ failures. Furthermore, it also protects against $\frac{d_{C\star D}-1}{2}$ byzantine errors. The reason for this is because the user downloads more information than strictly required when using the protocol. This way the lost information can be reconstructed using the extra information.

Here is presented two directions to proof against server errors for the transversal design based PIR scheme:

Extra servers: If a server doesn't respond, one simply needs to acquire the information

that the server contained. Thus one could introduce additional servers, that contain that information. The problem here is that these servers need to be designed with great care. If they are simply extra cloned parts of the database that are consulted in the case of a failure, it would be quite easy for the servers to find out what was asked for.

Redundancy introduction: One of the reasons the general scheme protects against failures is because extra redundant information is downloaded. It could be interesting to modify the database encoding of the transversal design based PIR scheme, such that extra redundancy is introduced. This would limit the rate of the scheme, and so one needs to be careful when introducing redundancy.

Conclusion

In this project a transversal design based private information retrieval protocol has been explored. The privacy of this PIR protocol comes from the size of t in t-transversal designs. The protocol is correct as long as each server doesn't send a wrong answer. This can be a problem, since if a server fails, the protocol might fail. It has been shown how transversal designs are built from orthogonal arrays and GRS codes. In this connection, it has been found out that one should not use GRS codes defined over a prime field, since the PIR protocol will have a very low rate. The main benefit of the scheme is it's remarkably low complexity, where each server needs only read a single symbol, and make no other calculations. The protocol also has a fairly good rate if the divisibility condition is satisfied. The storage overhead of scheme differs depending on the database: In the case of the database being $D \in \mathbb{F}_q^k$ the protocol stores $(ls-k)\log(q)$ bits or $ls-k$ $\mathbb{F}_q$ symbols. If the database instead is $D \in \mathbb{F}_{q^k}^m$ it stores $lsk-mk$ $\mathbb{F}_q$ symbols. The operation of the protocol may cause problems, since it is hard to manipulate the entries of the original database after encoding.

7 | Bibliography

[1] Lavauzelle J. Private Information Retrieval From Transversal Designs. IEEE Transactions on Information Theory. 2019 Feb;65(2):1189–1205. Available from: http://dx.doi.org/10.1109/TIT.2018.2861747.

[2] Freij-Hollanti R, Gnilke OW, Hollanti C, Karpuk DA. Private Information Retrieval from Coded Databases with Colluding Servers. SIAM Journal on Applied Algebra and Geometry. 2017 Jan;1(1):647–664. Available from: http://dx.doi.org/10.1137/16M1102562.

[3] Raviv N, Karpuk DA. Private Polynomial Computation from Lagrange Encoding; 2019.

[4] Colbourn CJ, Dinitz JH. Handbook of Combinatorial Designs. Discrete Mathematics and Its Applications. CRC Press; 2006. Available from: https://books.google.dk/books?id=S9FA9rq1BgoC.

[5] Justesen J, Høholdt T. A Course In Error-Correcting Codes. European Mathematical Society; 2004.

[6] Cameron PJ, Van Lint JH. Designs, graphs, codes and their links. 1st ed. London Mathematical Society Student Texts 22.. Cambridge, U.K: Cambridge University Press; 1991.

[7] Hughes DR, Piper FC. Design theory. Cambridge: Cambridge University Press; 1985.

[8] Lavauzelle J. Codes with locality : constructions and applications to cryptographic protocols; 2018.

[9] Assmus EF, Key JD. Designs and their Codes. Cambridge Tracts in Mathematics. Cambridge University Press; 1992.

[10] van Lint J, Wilson R. On the minimum distance of cyclic codes. IEEE Transactions on Information Theory. 1986;32(1):23–40.