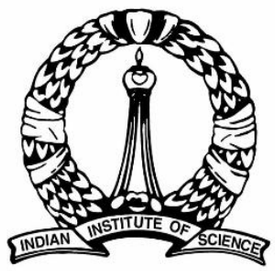
Some problems in Private Information Retrieval

A final project report
submitted in partial fulfilment of the
requirements for the degree of
Master of Technology
in
Communication and Networks
by

Adway Patra
SR No.: 04-02-04-10-42-17-1-14681

Under the guidance of

Prof. Navin Kashyap



Department of Electrical Communication Engineering
Indian Institute of Science
Bangalore - 560 012
June 2019

Abstract

Two latest variations of the private information retrieval (PIR) problem are considered in this work. In the first one, we consider the problem of PIR from multiple storage nodes when the underlying database is encoded using regenerating codes, i.e., the database has the ability to recover from individual node failures. More specifically, we intend to design PIR schemes for such databases that optimize the popular performance measure of download rate of a retrieval scheme. We wish to maximize this rate when a user wants to retrieve a message privately (in an information-theoretic sense) from the database. In this regard, firstly, we provide some sub-optimal PIR schemes for a specific subclass of regenerating codes. Afterwards, we focus on the more extensively explored Minimum Storage Regenerating (MSR) codes and formally state and prove the PIR capacity of such codes. We also give a simple scheme to achieve this capacity which is based on the similarly motivated construction for MDS codes. Furthermore, we give a PIR scheme for a specific class of Minimum Bandwidth Regenerating (MBR) codes that achieves the same rate as its minimum storage counterpart. In the second part of this thesis, we take on the problem of PIR in graph-based replication systems with possible node collusion. We discuss a recently proposed PIR scheme that provides perfect privacy provided that the girth of the graph is larger than the degree of collusion. We give an extension to this scheme that allows the user to circumvent this restriction by incurring a reduction in the download rate. Finally, for any degree of collusion, we give an upper bound on the download rate of linear PIR for a large family of graphs.

Contents

Chapter 1

Introduction

Private Information Retrieval (PIR) is the problem of retrieval of a message among several messages stored in a storage system without revealing any information about the identity of the desired message to the database. The problem of PIR was first introduced in the seminal paper by Chor *et al.* [2]. Since then, a considerable amount of research has been done in the field of PIR. In the classical PIR problem, a user trying to download one message out of many contacts multiple non-communicating storage nodes (or servers) that store the messages in a replicative format (using repetition coding) such that no information about the identity of the message is leaked to any of the nodes. The requirement of multiple storing entities is justified by the observation that for the case of a single-server database, the only possible privacy-preserving download scheme is the trivial one of downloading everything. On the other hand, the efficiency of a retrieval scheme while maintaining user privacy becomes a non-trivial concern in case of a database consisting of multiple servers in a distributed storage setting. A well-accepted measure of efficiency for information-theoretic PIR schemes is the download rate which is the number of desired bits retrieved per downloaded bit.

It was shown in [3] that the maximum rate of download for such a scheme termed the PIR capacity is $\frac{1-\frac{1}{N}}{1-\frac{1}{N^m}}$ for m equal-length messages stored in a replicative manner across N nodes. The authors also presented an achievable scheme based on the idea of blind interference alignment technique which was introduced for wireless networks in [4]. Although, the redundancy introduced by the repetition coding scheme allows the system to tolerate the maximum possible number of simultaneous node failures, it is somewhat wasteful of the storage resources. To offer more efficient use of storage space, Maximum Distance Separable (MDS) coded databases were considered for the PIR setting in [1] and the capacity expression was derived to be $\frac{1-R_c}{1-{R_c}^m}$ where R_c is the code-rate $\frac{k}{n}$ of the underlying MDS $[n, k]$ code.

Since distributed storage systems are prone to frequent node failures, regenerating codes were proposed in [5]. These codes have the capability of replacing a failed node with an exact or functional replica by contacting a subset of the surviving nodes while minimizing the amount of data downloaded for repair. In this work, we have considered the problem of PIR while the underlying distributed storage system is of a regenerating nature. It turns out that for storage-optimal regenerating codes (i.e., the Minimum Storage Regenerating codes to be formally defined later) the PIR capacity is exactly that of an MDS coded database.

An alternative way of reducing the storage overhead without sacrificing the simplicity of replication coding is to use partial replication where a message is replicated in only a subset of the nodes instead of all of them. This leads to a hypergraph structure in the DSS where nodes denote vertices and edges denote messages. The most storage efficient of such schemes is with overhead of 2 which leads to a graph structure. In [6], the authors considered the problem of PIR on such graph-based replication systems (GPIR) with possible node collusions. They gave a PIR scheme that provides perfect privacy achieving the rate of $\frac{1}{n}$, n being the number of nodes in the system, as long as there are no cycles in the subgraph induced by the colluding nodes. Hence, the scheme is restricted to work only when the maximum degree of collusion is less than the girth of the graph. In this work, an extension of their scheme is given which allows the user to achieve perfect privacy even when the degree of collusion is more than the girth of the graph by compromising on the download rate. In fact, the scheme provides a trade-off between the maximum degree of collusion that needs to be tolerated and the download rate. Additionally, we give the first generic upper bound on the rate of linear PIR for a large class of graphs and any value of the degree of collusion.

Chapter 2

Related Work

Having been introduced in [2], the basic PIR problem has seen a lot of practical variations throughout the past few years. As stated before, the PIR capacity for replicative database was found in [4] and the same for the MDS coded database in [1]. The problem of PIR with colluding servers (TPIR) where at most T of the total number of servers may collude with each other, was considered in [7] and later extended to byzantine and adversarial servers with possible node collusions in [8] and [9]. The idea of symmetric PIR (SPIR), which ensures the privacy of the messages other than the desired one from the retriever as well as the privacy of the desired message from the individual nodes, was investigated in [10] for replicated databases and was later extended to MDS coded databases in [11]. A further extension of this SPIR problem was considered in [12] with different types of adversaries and the capacity was derived in each scenario. In [13], the authors consider the problem of PIR with available private side information (PIR-PSI) at the user end. The capacity of PIR-PSI was derived in [14]. Further variations of this work are cache-aided PIR with uncoded partial side information [15], single-server single-message PIR with coded side information [16], single-server multi-message PIR with side information [17] and PIR-PSI under storage constraints [18]. Two recent variations of the PIR scheme are asymmetrical PIR schemes [19] which download at different rates from different servers and staircase PIR [20] which presents an universally robust solution of the problem of unresponsive servers.

Regenerating codes were introduced in [5] to address the issue of frequent node failures in erasure coded databases and to expedite the subsequent repairs. An $[n, k, d]$ regenerating code for n nodes allows the user to reconstruct the data by contacting any k of the n nodes and allows any failed node to be regenerated by contacting any d of the remaining $n-1$ nodes. Two types of node repairs are considered in the literature: functional repair and exact repair. The problem of functional repair was shown to be connected to network multi-casting problem in a directed graph [5]. On the other hand, the exact repair problem has a trade-off between the amount of storage per node and the amount of download required for repair of a node. Codes at the two extreme points: the Maximum Storage Regenerating (MSR) codes and the Maximum Bandwidth Regenerating (MBR) codes are of fair interest and have been vastly studied. The achievability of the MSR point was first shown in [21] for $k = 2$ and $d = n-1$. An MBR construction with $d = n-1$ and an MSR code for $d = k+1$ performing approximately-exact regeneration was given in [22]. A combination of functional and exact regeneration i.e. a hybrid regeneration at the MSR point was proposed in [23]. All these constructions require large field sizes. The Product-Matrix (PM) construction for both the MSR and MBR points, proposed in [24], provides codes for almost all parameters (all n, k, d values for MBR and all $n, k, d \geq 2k-1$ for MSR) and requires considerably smaller field sizes. In [25], the authors showed that all other points except the MSR and MBR on the storage-bandwidth trade-off are not achievable under exact repair.

The problem of PIR for regenerating codes was first considered in [26]. However, the construction requires the number of nodes in the system to be exponential in the number of messages, which, in real life scenarios, is not practical to implement. Very recently, there has been a growing interest in the research community on this topic. Recent PIR scheme constructions on the widely popular PM framework can be found in [27]-[28]. We introduce some sub-optimal PIR schemes for the PM framework on both the MSR and MBR points of interest. These schemes have the advantage of having rates $1 - \frac{d}{n}$ that are independent of the number of messages in the database. Next, we give an explicit algorithm to construct PIR schemes for achieving the PIR capacity of MSR codes which is the same as that of the

MDS codes. Our construction is based upon the MDS PIR capacity achieving scheme and works by treating MSR codes as vector MDS codes. A similar line of work can be found in [29]. We also give a scheme for the specific PM MBR codes that achieves the same rate as its MSR counterpart.

The notion of PIR for graph based replication systems was first proposed in [6]. This line of research is more closely related to the problem of storage constrained PIR [30] where each node is allowed to store only a fraction of the total data. Many real life practical data storage systems such as Hadoop and Cassandra use such graph based storage techniques. To the best of our knowledge, the question of the maximum possible PIR rate for such databases, with possible node collusions, is, in general, not explored. A recent work of Jia *et al.* [31] has answered the question of asymptotic capacity (when the number of messages becomes large) for certain scenarios. In this work, we follows the model of [6] which primarily considers the lowest storage overhead with a replication factor of 2. We first address a limitation of a PIR scheme proposed by the authors and give a technique to make their scheme applicable to wider scenarios, although by compromising in the download rate. Afterwards, we give an upper bound on the linear TPIR rate for any value of T for a large family of graphs.

The following chapters are organized as follows. Chapter 3 gives some preliminaries about regenerating codes that are useful in subsequent sections. We introduce our model for PIR for regenerating codes and PIR for graph-based replication systems in Chapter 4. In Chapter 5, we introduce the sub-optimal schemes for the MSR and MBR PM codes. In Chapter 6, we state the PIR capacity for MSR codes and give an explicit algorithm for constructing a capacity achieving PIR scheme for any MSR code. We also show that same rates can be achieved for the MBR PM codes as well. In Chapter 7, we briefly discuss the PIR scheme of [6] and also provide our proposed extension. We then give the upper bound on linear TPIR rates for a family of graphs. Finally, we draw conclusion in Chapter 8.

Chapter 3

Preliminaries

3.1 Regenerating Codes

Distributed Storage Systems (DSS) are prone to frequent node failures. To combat this random phenomenon, controlled redundancy is introduced in the system such that even under the event of a certain number of node failures, the performance of the DSS from the user perspective is not affected. Erasure codes such as Reed-Solomon codes are widely used in DSS. An $[n, k]$ erasure code allows the user to retrieve a message allowing at most $n - k$ arbitrary nodes to fail. To keep such failure tolerance property invariant with time, there is a natural necessity of regeneration of the failed nodes. For MDS codes such as Reed-Solomon codes, a trivial technique for regeneration of a failed node can be to download the data contained in any k surviving nodes, decode the message from the downloaded data and then re-encode it for the failed nodes. This asks for huge communication overheads and hence more efficient regeneration is expected. In their seminal paper, Dimakis *et al.* [5] first showed, by use of network coding arguments, that such efficient codes do indeed exist and perform strictly superior to existing techniques.

A regenerating code $[n, k, d, \alpha, \beta, B]$ is used to store a message of size B, whose symbols belong to a certain finite field $\mathbb{F}_q$ among n identical servers storing α coded symbols each such that the following two conditions are satisfied.

- The original message can be retrieved by contacting any k out of n nodes and downloading their stored data.
- If a node fails, it can be regenerated by contacting any d out of $n-1$ remaining nodes and downloading β ($\leq \alpha$) symbols from each of them. Hence, we have the parametric relation

$$k \leq d \leq n - 1 \tag{3.1}$$

Using the cut set bound of network coding, the authors in [5] showed that the parameters of regenerating codes are constrained by the equation

$$B \leq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\} \tag{3.2}$$

This inequality not only provides an upper bound on the maximum possible information that can be stored using a fixed regenerating code, but also gives a trade-off between storage per node α and regeneration bandwidth $\gamma = d\beta$. Along the storage-bandwidth trade-off curve, the two extremal points are of significant interest. The point corresponding to the Minimum Storage Regenerating (MSR) code has parameters

$$(\alpha_{MSR}, \beta_{MSR}) = \left(\frac{B}{k}, \frac{B}{k(d-k+1)}\right) \tag{3.3}$$

It is clear from the expressions that having $d = k$ at the MSR point does not provide us with any savings in terms of download. To repair a failed node, a newcomer, restricted to contact only k nodes, can only perform the trivial technique of regeneration for MDS codes. The regeneration bandwidth turns out to be a decreasing function of d and hence is minimized

when all the other nodes are contacted for repairing a failed node, i.e., at $d = n - 1$, having the value of

$$\gamma_{MSR}^{min} = \alpha \left(\frac{n-1}{n-k}\right) = \left(\frac{B}{k}\right)\left(\frac{n-1}{n-k}\right) \tag{3.4}$$

A bandwidth expansion factor is necessary for the reliability-redundancy optimality of the MDS property.

The point on the other extreme of the curve, the Minimum Bandwidth Regenerating (MBR) code has the parameters

$$(\alpha_{MBR}, \beta_{MBR}) = \left(\frac{2Bd}{2kd - k^2 + k}, \frac{2B}{2kd - k^2 + k}\right) \tag{3.5}$$

At this operating point, we have $\alpha = d\beta$ and so MBR codes do not incur any repair bandwidth expansion and download exactly the amount of information stored in the failed node. However, at the point $d = n - 1$, we have

$$\gamma_{MBR}^{min} = \alpha_{MBR}^{min} = \left(\frac{B}{k}\right)\left(\frac{2n-2}{2n-k-1}\right) \tag{3.6}$$

which shows that a storage expansion has been incurred for minimizing the bandwidth and the codes are no longer optimal in the MDS sense.

3.2 Product-Matrix (PM) Codes

The product-matrix framework [24] provides regenerating code constructions for almost all parameters of interest at the two extreme points of the trade-off curve (all $[n, k, d]$ values at the MBR point and all $[n, k, d \geq 2k - 1]$ values at the MSR point). An $[n, k, d, \alpha, \beta, B]$ MSR PM code is described by an $n \times \alpha$ code matrix C whose i^{th} row corresponds to the α symbols stored by the i^{th} node. The code matrix is the product of an $n \times d$ encoding matrix ψ and $d \times \alpha$ message matrix M :

$$C = \psi M \tag{3.7}$$

Note that M possesses certain symmetry properties to facilitate regeneration and does not necessarily contains $d\alpha$ independent message symbols. If a user contacts any set K of nodes $\{i_1, i_2, \ldots, i_k\}$ such that the cardinality, $|K|$, of the set equals k and downloads the stored data, the information available to the user can be represented as $\psi_K M$ where the matrix ψ_K is a sub-matrix of ψ corresponding to these rows. The user must be able to decode the original message from these $k\alpha$ symbols. If now a node, say f, fails, let the newcomer node contact a set D of surviving nodes $\{h_1, h_2, \ldots, h_d\}$ such that $|D| = d$. A node h_j sends back $\psi_{h_j} M \mu_f$ where μ_f is a length α column vector that is dependent upon the failed node index f. From the data sent back by the d helper databases, the newcomer should be able to exactly reproduce the lost α symbols of the node f. In this framework, β is assumed to be 1 and the authors in [24] showed that this assumption is without loss of generality.

3.2.1 Minimum Storage Regenerating Codes

The parameter set of the MSR PM code can be expressed as

$$[n, k, d = 2k - 2, \alpha = d - k + 1 = k - 1, \beta = 1, B = k\alpha] \tag{3.8}$$

The message matrix M is a $d \times \alpha$ matrix

$$M = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{3.9}$$

where each of S_1 and S_2 is a $\alpha \times \alpha$ symmetric matrix containing $\binom{\alpha+1}{2}$ distinct independent symbols each. Hence the total message symbols in the two matrices together is $\alpha(\alpha + 1)$ which is exactly B. The encoding matrix ψ is defined as

$$\psi = \begin{bmatrix} \phi & \Delta\phi \end{bmatrix} \tag{3.10}$$

where ϕ is an $n \times \alpha$ matrix having any α rows linearly independent and Δ is an $n \times n$ diagonal matrix with distinct elements. Also the matrix ψ needs to be such that any d rows are linearly independent. It was shown in [24] that one can construct codes for all $n - 1 \geq d \geq 2k - 2$ by using an MSR code at $d = 2k - 2$ (Theorem 6 in [24]).

3.2.2 Minimum Bandwidth Regenerating Codes

The parameter set of the MBR PM code can be expressed as

$$[n, k, d \geq k, \alpha = d, \beta = 1, B = \binom{k+1}{2} + k(d-k)] \tag{3.11}$$

The message matrix M is defined as

$$M = \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \tag{3.12}$$

where S is a symmetric matrix of dimension $k \times k$ containing the first $\binom{k+1}{2}$ symbols of the message and the matrix T is of dimension $k \times (d-k)$ containing the other $k(d-k)$ message symbols. The encoding matrix ψ has the structure

$$\psi = \begin{bmatrix} \phi & \Delta \end{bmatrix} \tag{3.13}$$

where ϕ is an $n \times k$ matrix having any k rows linearly independent and Δ is an $n \times (d-k)$ matrix such that, overall, any d rows of ψ are linearly independent.

Chapter 4

System Model

In this section we describe the system model that we will be following. Since we have two different problems to discuss, the two system models are also different and described separately. Let $[a]$, for some positive integer a, denote the set $\{1, 2, \ldots, a\}$ and $[a_1 : a_2]$, for some positive integers a_1, a_2 and $a_2 \geq a_1$, denote the set $\{a_1, a_1 + 1, \ldots, a_2 - 1, a_2\}$. For a set of random variables $\{A_i : i \in [a]\}$, $A_{j_1:j_2}$ for $1 \leq j_1 \leq j_2 \leq a$ denotes the set of random variables $\{A_{j_1}, A_{j_1+1}, A_{j_1+2}, \ldots, A_{j_2}\}$ and A_K for $K \subseteq [a]$ denotes $\{A_i : i \in K\}$. For any matrix X, the r^{th} row of matrix X is denoted by $X(r)$ and for a set of indices K, X_K denotes the sub-matrix of X restricting its rows to those with indices in K. The transpose of the matrix X is denoted by X^t.

4.1 PIR for Regenerating Codes

We have n identical servers denoted by the set $[n] = \{1, 2, \ldots, n\}$ and they store m messages $\{W_1, W_2, \ldots, W_m\}$. We fix an $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B}]$ regenerating code and take the size of each message to be $L = \tilde{L}\tilde{B}$ where $\tilde{L}$ is some positive integer. The L symbols of each message are independently chosen at random from a finite field $\mathbb{F}_q$:

$$H(W_i) = L \ \forall \ i \ \in \ [m] \tag{4.1}$$

$$H(W_1, W_2, \ldots, W_m) = \sum_{i=1}^{m} H(W_i) = mL \tag{4.2}$$

where $H()$ is the entropy function with logarithm taken to the base q. The encoding of a message of L symbols is done in the following way. Each block of $\tilde{B}$ message symbols is separately encoded using the same encoding procedure of the above regenerating code. Since the encoding of each such block is carried out independently of the other blocks, the regeneration and reconstruction (i.e., decoding) of each such block can also be carried out independently of all the other blocks of the same message. The resulting encoding of the message with L symbols can be thought of as the encoding obtained using an $[n, k, d, \alpha, \beta, B = L]$ regenerating code that stores $\alpha = \tilde{L}\tilde{\alpha}$ symbols per node and requires $\beta = \tilde{L}\tilde{\beta}$ downloads from each helper node during node regeneration. This idea is inspired by the concept of "striping of data" as discussed in Section I-C of [24]. Such an encoding scheme gives us the flexibility to increase L in multiples of $\tilde{B}$ by simply increasing $\tilde{L}$. This causes the parameters α, β, B to increase accordingly while the *underlying* regenerating code parameters $\tilde{\alpha}, \tilde{\beta}, \tilde{B}$ do not change. In the limit, the file size L can be made to go to infinity simply by increasing $\tilde{L}$. As $\tilde{L} \to \infty$, the parameters $\alpha, \beta, L \ (= B) \to \infty$ while the *underlying* regenerating code parameters $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B}]$ remain fixed.

Let Y_i denote the data stored at node i. Since the messages themselves are independent and the encoding of each message is performed independently of the others, it follows that Y_i consists of m independent components $\{Y_i^1, Y_i^2, \ldots, Y_i^m\}$ with Y_i^j being the component corresponding to the j^{th} message.

$$H(Y_i^j) = \alpha \ \ \forall \ j \in [m] \tag{4.3}$$

$$H(Y_i) = m\alpha \tag{4.4}$$

A user should be able to recover a message by accessing any subset $K \subset N$ of cardinality k and downloading the data corresponding to that message. So,

$$H(W_j | Y_K^j) = 0 \ \ \forall \ j \in [m] \tag{4.5}$$

which when combined with Equations (4.1)-(4.4) gives:

$$H(W_1, W_2, \ldots, W_m | Y_K) = 0 \tag{4.6}$$

In the event of a node failure, the newcomer node contacts a subset $D \subset N$ of nodes of cardinality d and each of these helper nodes sends β symbols for the repair. Let us use $S_{h\to f}$ to denote the random variable sent by helper node h for the repair of node f. Again using independence arguments we can say that the variable $S_{h\to f}$ can be split into m independent components $\{S^1_{h\to f}, S^2_{h\to f}, \ldots, S^m_{h\to f}\}$ where $S^j_{h\to f}$ is sent by helper node h for regeneration of the data stored at node f corresponding to the j^{th} message. From the regenerating code parameters it follows that :

$$H(S^j_{h\to f}) = \beta \quad \forall \ j \in [m] \tag{4.7}$$

$$I(S^{j_1}_{h\to f}; S^{j_2}_{h\to f}) = 0 \quad \forall \quad j_1, j_2 \in [m] \ , \ j_1 \neq j_2 \tag{4.8}$$

which implies

$$H(S_{h\to f}) = m\beta \tag{4.9}$$

The regeneration property further states that

$$H(S^j_{h\to f} | Y^j_h) = 0 \quad \forall \ j \in [m] \tag{4.10}$$

$$H(Y^j_f | S^j_{D\to f}) = 0 \quad \forall \ j \in [m] \tag{4.11}$$

where $S^j_{D\to f} \equiv \{S^j_{h\to f} : h \in D\}$. Now let us suppose that an user is interested in retrieving a message W_j privately from the n nodes. The user has a realization of j private to him. A set of queries $\{Q^{(j)}_1, Q^{(j)}_2, \ldots, Q^{(j)}_n\}$ is prepared by the user such that individually the queries reveal no information about the desired index j to any of the nodes :

$$I(j; Q^{(j)}_i) = 0 \quad \forall \, i \ \in \ [n] \tag{4.12}$$

Since the queries are prepared with no information about the original messages, we have

$$I(Q^{(j)}_1, Q^{(j)}_2, \ldots, Q^{(j)}_n; W_1, W_2, \ldots, W_m) = 0 \quad \forall \ j \ \in \ [m] \tag{4.13}$$

The i^{th} node, upon receiving the query $Q^{(j)}_i$ returns an answer variable $A^{(j)}_i$ which is a deterministic function of the data stored Y_i and the received query $Q^{(j)}_i$:

$$H(A^{(j)}_i | Q^{(j)}_i, Y_i) = 0 \quad \forall \, i \ \in \ [n] \tag{4.14}$$

Getting back all the answers $(A^{(j)}_1, A^{(j)}_2, \ldots, A^{(j)}_n)$ from the n nodes, the user should be able to recover the desired message with very small probability of error. From Fano's inequality it follows that :

$$H(W_j | A^{(j)}_1, \ldots, A^{(j)}_n, Q^{(j)}_1, \ldots, Q^{(j)}_n) = o(L) \quad \forall \, j \ \in \ [m] \tag{4.15}$$

where $\frac{o(L)}{L} \to 0$ as $L \to \infty$.

The rate of the PIR scheme is defined as the ratio of the size of the desired message to the total download cost under the two constraints of equations (4.12) and (4.15) of privacy and reliability respectively. The rate R is defined as

$$R = \frac{H(W_j)}{\sum_{i=1}^{n} H(A^{(j)}_i)} \tag{4.16}$$

The PIR capacity C is the supremum of the rate over all PIR schemes. As mentioned before, since we are following the standard setting of information-theoretic PIR, we will be neglecting the upload cost related to the queries and only focus on the download cost.

4.2 PIR for Graph-based Replication Systems

We consider a distributed storage system (DSS) with n identical storage nodes (or servers), denoted by the set $[n]$. There are m messages in the system denoted by $W_1, W_2, \ldots, W_m$. Each message is of length L, for some integer L. The L symbols of each message are chosen uniformly, randomly and independently from some underlying field $\mathbb{F}_q$.

$$H(W_j) = L \ \forall\, j \in [m] \tag{4.17}$$

$$H(W_1, W_2, \ldots, W_m) = \sum_{j=1}^{m} H(W_j) = mL \tag{4.18}$$

where $H()$ is the entropy function with logarithm taken to the base q. Instead of conventional replication, where each message is replicated at every node, we assume that each message W_j is stored in a certain subset $\mathcal{R}_j \subset [n]$ of size 2. Hence, each message is replicated at exactly two nodes. Further, we assume that for $j_1 \neq j_2$ $, \mathcal{R}_{j_1} \neq \mathcal{R}_{j_2}$ $\forall j_1, j_2 \in [m]$, i.e., two nodes can share at most one message.

Based upon this storage pattern, we can define an undirected graph $G = (V, E)$ with $|V| = n, |E| = m$, where an edge $(i_1, i_2) \in E$ iff $\exists\ \mathcal{R}_j = \{i_1, i_2\}$ for some $j \in [m]$. Clearly, our previous assumption excludes the possibililty of any double edges in this graph. So an edge (i_1, i_2) uniquely identifies the message W_j for which $\mathcal{R}_j = \{i_1, i_2\}$ and we will, alternatively, call this the j^{th} edge (message). The degree and the stored content of a vertex (node) i are denoted by d_i and Y_i. So, if the edges incident on node i are $\{j_1, j_2, \ldots, j_{d_i}\}$, then

$$Y_i = \begin{bmatrix} W_{j_1} & W_{j_2} & \ldots & W_{j_{d_i}} \end{bmatrix}^t \tag{4.19}$$

To retrieve desired message $W_{\hat{m}}$, the user prepares queries $\{Q_1^{(\hat{m})}, Q_2^{(\hat{m})}, \ldots, Q_n^{(\hat{m})}\}$ and sends $Q_i^{(\hat{m})}$ to node i. Since, we will only be considering linear PIR schemes, we can think of $Q_i^{(\hat{m})}$ to be a $l_i \times d_i L$ matrix

$$Q_i^{(\hat{m})} = \begin{bmatrix} Q_{i,j_1}^{(\hat{m})} & Q_{i,j_2}^{(\hat{m})} & \ldots & Q_{i,j_{d_i}}^{(\hat{m})} \end{bmatrix} \tag{4.20}$$

where each $Q_{i,j}^{(\hat{m})}$ is a $l_i \times L$ matrix. The i^{th} node, upon receiving $Q_i^{(\hat{m})}$, returns the answer string

$$A_i^{(\hat{m})} = Q_i^{(\hat{m})} Y_i = \begin{bmatrix} Q_{i,j_1}^{(\hat{m})} & Q_{i,j_2}^{(\hat{m})} & \ldots & Q_{i,j_{d_i}}^{(\hat{m})} \end{bmatrix} \begin{bmatrix} W_{j_1} & W_{j_2} & \ldots & W_{j_{d_i}} \end{bmatrix}^T \tag{4.21}$$

which is of length l_i. The total download is $D = \sum_{i=1}^{n} l_i$ and the rate of the PIR scheme is $R = \frac{L}{D}$.

To model the T-privacy constraint, we say that a set $\mathcal{T} \subset [n]$ of nodes of size at most T can collude among themselves, i.e., they can exchange information about the queries sent to them and the data they store. Alternatively, we model this by an adversary who has complete knowledge of the storage pattern and has access to the queries and stored contents of a set $\mathcal{T} \subset [n], |\mathcal{T}| = T$. We shall use these two terminologies interchangeably. We have the following information-theoretic privacy constraint:

$$I(\hat{m}; Q_{\mathcal{T}}^{(\hat{m})}) = 0 \quad \forall\, \mathcal{T} \subset [n],\ |\mathcal{T}| = T,\ \forall\, \hat{m} \in [m] \tag{4.22}$$

For notational simplicity, we will sometimes suppress the use of the desired message index in the superscripts of all random variables. The intention should be clear from the context. We write all the queries $\{Q_i : i \in [n]\}$ in the $n \times m$ matrix Q. The $(i,j)^{th}$ entry of Q is itself a matrix $Q_{i,j} \in \mathbb{F}_q^{l_i \times L}$ if in the incidence matrix I of the graph G the $(i,j)^{th}$ entry is 1 and an all-zero matrix of appropriate dimensions otherwise. The complete answer vector would then be

$$A = \begin{bmatrix} A_1 & A_2 & \ldots & A_n \end{bmatrix}^T = Q \begin{bmatrix} W_1 & W_2 & \ldots & W_m \end{bmatrix}^T \tag{4.23}$$

For example, the query matrix and answer vector for K^3, would be

$$A = \begin{bmatrix} Q_{1,1} & 0 & Q_{1,3} \\ Q_{2,1} & Q_{2,2} & 0 \\ 0 & Q_{3,2} & Q_{3,3} \end{bmatrix} \begin{bmatrix} W_1 \\ W_2 \\ W_3 \end{bmatrix} \tag{4.24}$$

For a set $\mathcal{T} \subset [n]$ and $\mathcal{S} \subset [m]$, $Q_{\mathcal{T},\mathcal{S}}$ will denote the submatrix of Q restricted to the rows and columns corresponding to the nodes of $\mathcal{T}$ and messages of $\mathcal{S}$. The notations $Q_{\mathcal{T},:}$

and $Q_{:,\mathcal{S}}$ will be abbreviated to $Q_{\mathcal{T}}$ and $Q_{\mathcal{S}}$ whenever there is little chance of confusion. The subgraph spanned by the nodes in $\mathcal{T}$ will be denoted by $G_{\mathcal{T}}$. The ranks of all the matrices will be taken over the field $\mathbb{F}_q$.

From the answer strings, the desired message $W_{\hat{m}}$ would be recoverable iff there is a linear transformation over $\mathbb{F}_q$ consisting of only row operations of the matrix Q to some other matrix $\overline{Q}$ of the form

$$\overline{Q} = \begin{bmatrix} 0 & \dots & \tilde{Q}_{\hat{m}} & \dots & 0 \\ & & X & & \end{bmatrix} \tag{4.25}$$

where $\tilde{Q}_{\hat{m}}$ is an $L \times L$ full rank matrix and X is an arbitrary matrix. Since there are only row operations allowed, we can say that

$$rank(\begin{bmatrix} Q_{x,\hat{m}} \\ Q_{y,\hat{m}} \end{bmatrix}) = L \tag{4.26}$$

where $\mathcal{R}_{\hat{m}} = \{x, y\}$. We argue that for preserving privacy, the above rank condition has to be true not only for the desired message $W_{\hat{m}}$ but for all the messages. In other words, the following has to be satisfied

$$rank(\begin{bmatrix} Q_{i_1,j} \\ Q_{i_2,j} \end{bmatrix}) = L \quad \forall\, j \in [m], \mathcal{R}_j = \{i_1, i_2\} \tag{4.27}$$

The reason is as follows. The adversary has full knowledge of the underlying graph, i.e., the storage pattern of messages, and he is in control of any T nodes for $2 \leq T \leq n$, the identities of which are not known to the user. If for any message index $j \in [m]$, the above rank condition is not satisfied then an adversary who has control over the nodes of $\mathcal{R}_j = \{i_1, i_2\}$ can calculate the rank of $\begin{bmatrix} Q_{i_1,j} \\ Q_{i_2,j} \end{bmatrix}$ and hence deduce that W_j can not be the desired message. This is a breach of perfect privacy and hence the above relationship must hold true.

Chapter 5

Sub-optimal PIR Schemes for PM Codes

5.1 PIR Scheme for MSR PM Codes

We propose a scheme that achieves the rate $1 - \frac{d}{n}$ for when the underlying code is an MSR PM code with parameters $[n = d + k, k, d \geq k, \tilde{\alpha} = d - k + 1, \tilde{\beta} = 1, \tilde{B} = k\tilde{\alpha}]$. The m messages stored in the database each is of size $L = k\tilde{\alpha}$ (i.e., $\tilde{L} = 1$ and hence $\alpha = \tilde{\alpha}, \beta = \tilde{\beta}$) and encoded using the product-matrix MSR code across n nodes. The user wants to retrieve message $W_{\hat{m}}$ where $\hat{m} \in [m]$. User prepares queries $\{Q_1^{(\hat{m})}, \ldots, Q_n^{(\hat{m})}\}$ where each $Q_i^{(\hat{m})}$ is a $\alpha \times m\alpha$ matrix and can be written as

$$Q_i^{(\hat{m})} = U + V_i^{(\hat{m})} \quad \forall\, i \in\ [n] \tag{5.1}$$

U is a randomly chosen matrix from the space of all $\alpha \times m\alpha$ matrices with elements from the field $\mathbb{F}_q$. The matrix $V_i^{(\hat{m})}$ has the structure

$$V_i^{(\hat{m})} = \begin{bmatrix} O_{\alpha\times(\hat{m}-1)\alpha} & X_i^{(\hat{m})} & O_{\alpha\times(m-\hat{m})\alpha} \end{bmatrix} \tag{5.2}$$

where O is the all-zero matrix having the dimensions in the sub-script and the $\alpha \times \alpha$ matrix $X_i^{(\hat{m})}$ is chosen as follows

1. Choose any d number of nodes D from the n nodes $\{1, 2, \ldots, n\}$ and set $X_i^{(\hat{m})} = O_{\alpha\times\alpha}\ \ \forall\ \ i \in D$.
2. For the remaining $(n - d)$ nodes, set $X_i^{(\hat{m})} = I_{\alpha\times\alpha}\ \ \forall\ \ i \in [n] \setminus D$

The i^{th} node upon receiving query matrix $Q_i^{(\hat{m})}$ will return answer string $A_i^{(\hat{m})}$ as

$$A_i^{(\hat{m})} = Q_i^{(\hat{m})} Y_i \tag{5.3}$$

The following claims further study and verify the properties of this scheme.

Claim 5.1.1: *From all the $\{A_i^{(\hat{m})} : i \in [n]\}$, the user is able to recover $W_{\hat{m}}$.*

Proof: Consider the set D that was randomly chosen at the users end for which all-zero matrix was set as $X_i^{(\hat{m})}$. For these nodes

$$A_i^{(\hat{m})}(r) = Q_i^{(\hat{m})}(r) Y_i = U(r) Y_i \quad \forall\, r\ \in\ [\alpha] \tag{5.4}$$

Following the notations introduced in section 3.2.1, we can express Y_i as

$$Y_i = \overline{M}\psi(i)^t \tag{5.5}$$

where $\overline{M}$ is the $m\alpha \times d$ matrix $[\ M_1 \mid M_2 \mid \ \ldots\ \mid M_m\ ]^t$ where each $M_j\ \forall\ \ j \in [m]$ is the $d \times \alpha$ message matrix corresponding to to the j^{th} message W_j according to the MSR format of Equation (3.9). Note that $A_i^{(\hat{m})}(r)$ is the inner product of two equal-length vectors and hence it is a single element from $\mathbb{F}_q$. If we now take the set $\{A_i^{(\hat{m})}(r) : i \in D\}$, we can write this in matrix form

$$A_D^{(\hat{m})}(r) = U(r)\overline{M}\psi_D^t \tag{5.6}$$

where ψ_D is the sub-matrix of ψ having rows indexed by D. From the properties of ψ, we know that ψ_D is invertible and so we can calculate the length-d vector $U(r)\overline{M}$. Now take any node $i' \in [n] \setminus D$ for which $X_i^{(\hat{m})} = I_\alpha$. For these nodes

$$\begin{aligned} A_{i'}^{(\hat{m})}(r) = Q_{i'}^{(\hat{m})}(r)Y_{i'} &= U(r)Y_{i'} + V_{i'}^{(\hat{m})}(r)Y_{i'} \\ &= U(r)\overline{M}\psi_{i'}^t + e_r Y_{i'}^{\hat{m}} \end{aligned} \tag{5.7}$$

where e_r is the α length unit vector having a 1 in the rth position. Since we know the encoding matrix ψ and we have already calculated $U(r)\overline{M}$, from Equation (5.7) we can cancel the interference term $U(r)\overline{M}\psi_{i'}^t$ to get only $e_r Y_{i'}^{\hat{m}}$ which is nothing but the r^{th} coded symbol in the i'^{th} node corresponding to the user's desired message $W_{\hat{m}}$. Doing this for all $i' \in [n] \setminus D$, and all $r \in [\alpha]$, we have a total of $k\alpha$ coded symbols corresponding to $W_{\hat{m}}$ from k databases. This information is sufficient for decoding the desired message as per the properties of the underlying code. ■

Claim 5.1.2: The rate of this PIR scheme is $1 - \frac{d}{n}$.

Proof: The rate of a PIR scheme, as defined in Section 4, is

$$R = \frac{H(W_{\hat{m}})}{\sum_{i=1}^n H(A_i^{(\hat{m})})} \tag{5.8}$$

Considering one term in the denominator sum,

$$\begin{aligned} H(A_i^{(\hat{m})}) &= \sum_{r=1}^{\alpha} H(A_i^{(\hat{m})}(r) | A_i^{(\hat{m})}(r-1), \ldots, A_i^{(\hat{m})}(1)) \\ &= \sum_{r=1}^{\alpha} H(U(r)Y_i + V_i^{(\hat{m})}(r)Y_i | U(r-1)Y_i + V_i^{(\hat{m})}(r-1)Y_i, \ldots, U(1)Y_i + V_i^{(\hat{m})}(1)Y_i) \\ &= \sum_{r=1}^{\alpha} H(U(r)Y_i + V_i^{(\hat{m})}(r)Y_i) \\ &= \sum_{r=1}^{\alpha} H(A_i^{(\hat{m})}(r)) \\ &= \alpha \end{aligned} \tag{5.9}$$

where the third equality in Equation (5.9) follows from the independence of the rows of the random matrix U. So, the rate of the scheme becomes

$$\begin{aligned} R &= \frac{H(W_{\hat{m}})}{\sum_{i=1}^n H(A_i^{(\hat{m})})} \\ &= \frac{k\alpha}{n\alpha} \\ &= \frac{(n-d)\alpha}{n\alpha} \\ &= 1 - \frac{d}{n} \end{aligned} \tag{5.10}$$

■

Claim 5.1.3: The retrieval scheme discussed here is private in the sense described in Section 4.

Proof: We have to prove that the condition of privacy of Equation (4.12) is satisfied for this retrieval scheme. For a node $i \in [n]$,

$$I(j; Q_i^{(j)}) = H(Q_i^{(j)}) - H(Q_i^{(j)} | j) \tag{5.11}$$

Consider the second term.

$$\begin{aligned}
H(Q_i^{(j)}|j) &= -\sum_{R\in\mathbb{F}_q^{(\alpha\times m\alpha)}}\sum_{\hat{m}\in[m]} Pr(Q_i^{(j)}=R, j=\hat{m})\log Pr(Q_i^{(j)}=R|j=\hat{m}) \\
&= -\sum_{R\in\mathbb{F}_q^{(\alpha\times m\alpha)}}\sum_{\hat{m}\in[m]} Pr(U=R-V_i^{(\hat{m})})Pr(j=\hat{m})\log Pr(U=R-V_i^{(\hat{m})}) \\
&= -\sum_{R'\in\mathbb{F}_q^{(\alpha\times m\alpha)}}\sum_{\hat{m}\in[m]} Pr(U=R')Pr(j=\hat{m})\log Pr(U=R') \\
&= -\sum_{R'\in\mathbb{F}_q^{(\alpha\times m\alpha)}} Pr(U=R')\log Pr(U=R') \qquad (5.12)\\
&= -\sum_{R'\in\mathbb{F}_q^{(\alpha\times m\alpha)}} Pr(Q_i^{(j)}-V_i^{(j)}=R')\log Pr(Q_i^{(j)}-V_i^{(j)}=R') \\
&= -\sum_{R''\in\mathbb{F}_q^{(\alpha\times m\alpha)}} Pr(Q_i^{(j)}=R'')\log Pr(Q_i^{(j)}=R'') \\
&= H(Q_i^{(j)})
\end{aligned}$$

which gives us $I(j;Q_i^{(j)})=0$ and hence proving the privacy of the discussed scheme. ■

Remark: The above described scheme puts a constraint on the number of servers present in the system i.e. $n=d+k$. Note that, even for the values of $n>d+k$, we can still make use of the scheme by choosing any random subset of $d+k$ nodes among n and operating only on these nodes and leaving the other nodes idle. Since, these working nodes can be chosen at random from the whole set, the privacy requirements are not violated.

5.2 PIR Scheme for MBR PM Codes

Now, we propose a similar scheme for the Product-Matrix MBR code that achieves the same rate as its counterpart. We take the parameters $[n=d+k,k,d\geq k,\tilde{\alpha}=d,\tilde{\beta}=1,\tilde{B}=k(d-k)+\binom{k+1}{2}]$ and choose k to be odd. Same as Section 5.1, $\tilde{L}$ is taken to be 1 which implies that the m messages each are of length $L=k(d-k)+\binom{k+1}{2}$ and $\alpha=\tilde{\alpha},\beta=\tilde{\beta}$. It is easy to see that the query matrices $\{Q_i^{(\hat{m})}: i\in[n]\}$ described by Equations (5.1)-(5.2) can also be used in this setting to have a PIR scheme. However, the rate of this scheme is now less then $1-\frac{d}{n}$ simply because the numerator in (5.10) is now less than $k\alpha$. To get the same rate, we now have to reduce the number of sub-queries per node, i.e., reduce the number of rows in the matrices $\{Q_i^{(\hat{m})}: i\in[n]\}$. This is done in the following manner.

The query matrices $\{Q_i^{(\hat{m})}: i\in[n]\}$ have the same structure as in Equation (5.1) but the number of rows are limited to $d-\frac{k-1}{2}$. The matrices $V_i^{(\hat{m})}$ for $i\in[n],\hat{m}\in[m]$ have similar structure to Equation (5.2) and the dimensions are changed to $(d-\frac{k-1}{2})\times md$.

$$V_i^{(\hat{m})}=\begin{bmatrix}O_{(d-\frac{k-1}{2})\times(\hat{m}-1)d} & X_i^{(\hat{m})} & O_{(d-\frac{k-1}{2})\times(m-\hat{m})d}\end{bmatrix} \qquad (5.13)$$

The main difference lies in choosing the matrix $X_i^{(\hat{m})}$. The $(d-\frac{k-1}{2})\times d$ matrix $X_i^{(\hat{m})}$ for this scheme is chosen as

- For a set $D\subset[n]$ of cardinality d, set $X_i^{(\hat{m})}=O_{((d-\frac{k-1}{2})\times d)}$.
- For the other k nodes of $K\equiv[n]\setminus D\equiv\{i_1,i_2,\ldots,i_k\}$, the matrix has the following structure

$$X_{i_l}^{(\hat{m})}=\begin{bmatrix}Z_l & O_{(\frac{k+1}{2}\times(d-k))}\\ O_{((d-k)\times k)} & I_{(d-k)\times(d-k)}\end{bmatrix} \qquad (5.14)$$

The matrices $\{Z_l:\ l\in[k]\}$ each of dimension $\frac{k+1}{2}\times k$ are chosen according to the following Algorithm 1.

The i^{th} node upon receiving query matrix Q_i will return answer string A_i as

$$A_i^{(\hat{m})}=Q_i^{(\hat{m})}Y_i \qquad (5.15)$$

Algorithm 1: Determining matrices $\{Z_l : l \in [k]\}$

Result: Set of Matrices $\{Z_l : l \in [k]\}$

Initialization: Set all rows of Z_l to the unit vector e_1 of length k
with entry 1 in the first position for all $l \in [k]$
$r \leftarrow 1, sh \leftarrow 0, count \leftarrow 0, thres \leftarrow k$;
while $r \leq \frac{k+1}{2}$ **do**
  $l \leftarrow 1$;
  while $l \leq k$ **do**
    Right Shift $Z_l(r)$ cyclically by an amount sh;
    $count \leftarrow count + 1$;
    if $count \geq thres$ **then**
      $sh \leftarrow sh + 1$;
      $count \leftarrow 0$;
      $thres \leftarrow thres - 1$;
    end
    $l \leftarrow l + 1$;
  end
  $r \leftarrow r + 1$;
end

Each row of the matrix $X_i^{(\hat{m})}$ corresponds to one query vector sent to node i and hence is responsible of retrieving (privately) one symbol among the α $(= d)$ coded symbols of the $\hat{m}^{th}$ message stored in the i^{th} node. Hence each such row will be a unit vector of length d and the position of the unit element 1 will denote which of the α coded symbols we are trying to (privately) retrieve from that node in that round of query. To get an intuition of the structure of the $X_i^{(\hat{m})}$ matrices, let us look at the encoded symbols stored in the set of nodes K.

$$\begin{aligned}\begin{bmatrix} Y_{i_1}^{\hat{m}} & Y_{i_2}^{\hat{m}} & \dots & Y_{i_k}^{\hat{m}} \end{bmatrix} &= \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \psi_K^t \\ &= \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \begin{bmatrix} \phi_K^t \\ \Delta_K^t \end{bmatrix} \\ &= \begin{bmatrix} S\phi_K^t + T\Delta_K^t \\ T^t\phi_K^t \end{bmatrix}\end{aligned} \tag{5.16}$$

Observe that, the last $(d-k)$ entries of $Y_{i_l}^{\hat{m}}$ are functions of the T matrix only which contains $k(d-k)$ independent desired message symbols. To recover them, we need $k(d-k)$ linearly independent coded symbols. That is exactly what is being done by the identity matrix at the lower right corner in Equation (5.14) where the linear independence is taken care of by the properties of the ϕ matrix. Now, the rest of the k top entries of each $Y_{i_l}^{\hat{m}}$, are now functions of the matrix S only (since T has already been recovered) and this matrix contains $\frac{k(k+1)}{2}$ independent message symbols. Downloading exactly these many coded symbols should be sufficient to recover the matrix S instead of downloading all k^2 related symbols from all k nodes. That is what the Z-matrices are designed to do. The following claim formally proves this idea. The X-matrices for $k = 3, d = 4$ and $k = 5, d = 8$ are given for a clearer understanding of the structure.

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \quad \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \quad \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$\left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$\left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

Claim 5.2.1: After receiving all the $\{A_i^{(\hat{m})} : i \in [n]\}$, the user will be able to successfully decode the desired message $W_{\hat{m}}$.

Proof: For a subset $D \in [n]$ with $|D| = d$, we had set $X_i^{(\hat{m})}$ to all-zero matrix. For these nodes, Equation (5.4)-(5.6) are valid for similar reasoning as before. Denote $(d - \frac{k-1}{2}) = b$. Consider a node $i \in [n] \setminus D$ and take $r = b$, that is, consider the last row of Q_i. We have

$$\begin{aligned} A_i^{(\hat{m})}(b) = Q_i^{(\hat{m})}(b)Y_i &= U(b)Y_i + V_i^{(\hat{m})}(b)Y_i \\ &= U(b)\overline{M}\psi(i)^t + e_b Y_i^{\hat{m}} \end{aligned} \tag{5.17}$$

After cancelling the interference term $U(b)\overline{M}\psi_i^t$, we are left with $e_b Y_i^{\hat{m}}$ where e_b is the d length unit vector having a 1 at the last coordinate. From k nodes we retrieve k such coded symbols corresponding to the message $W_{\hat{m}}$. Recall the encoding procedure of MBR PM codes

$$\begin{aligned} Y_i^{\hat{m}} &= M_{\hat{m}}\psi(i)^t \\ &= \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \begin{bmatrix} \phi(i) & \Delta(i) \end{bmatrix}^t \\ &= \begin{bmatrix} S\phi(i)^t + T\Delta(i)^t \\ T^t\phi(i)^t \end{bmatrix} \end{aligned} \tag{5.18}$$

Note that S is a $k \times k$ symmetric matrix and T is a $k \times (d-k)$ rectangular matrix with unique message symbols. From the above equation, we have the relationship for the retrieved symbols from the k nodes,

$$Y_K^{\hat{m}}(d) = T^t(d-k)\phi_K^t \tag{5.19}$$

where $K = [n] \setminus D = \{i_1, i_2, \ldots, i_k\}$ of cardinality k. From the properties of the encoding matrix, ϕ_K^t is invertible, and hence, we can recover the $(d-k)^{th}$ row of T^t. Continuing, this process for $r = b-1, \ldots, b-(d-k)+1$, we can completely recover the matrix T.

Among the $(d-k) + \frac{(k+1)}{2}$ query vectors sent to each database, the last $(d-k)$ are used to decode the matrix T. Now we will show that from the rest of the $\frac{(k+1)}{2}$ queries, we are able to recover the matrix S. The construction of the matrix Z_l will come in handy and the proof will follow by induction argument.

$$A_i^{(\hat{m})}(r) = U(r)\overline{M}\psi_i^t + X_i^{(\hat{m})}(r)M_{\hat{m}}\psi_i^t \quad \forall\, i \in K = [n] \setminus D\,, \quad r \in \left[\frac{k+1}{2}\right] \tag{5.20}$$

We can cancel out the first term by using the responses of the nodes in D and hence are left with

$$\begin{aligned} A_{i_l}^{(\hat{m})}(r) - U(r)\overline{M}\psi(i_l)^t &= X_{i_l}^{(\hat{m})}(r)M_{\hat{m}}\psi(i_l)^t \\ &= Z_l(r)[S\phi(i_l)^t + T\Delta(i_l)^t] \quad \forall\, l \in [k],\ r \in \left[\frac{k+1}{2}\right] \end{aligned} \tag{5.21}$$

This is an equation in d variables from which we can cancel the terms of the already known T matrix to get an equation in k variables from the r^{th} row of the S matrix. Now, consider the scenario when all $r'-1$ previous rows of S have already been decoded(starting from the first row) for some $r' \in [k]$. Due to the symmetric structure of S, among the k variables from the r'^{th} row of S forming the equations, $r'-1$ variables are known and hence we are left with equations of $k - r' + 1$ variables. We need this many linearly independent equations to solve this system. But this is exactly the number of downloads Algorithm 1 performs for the r'^{th} row and the linear independence property follows from the properties of the ϕ matrix. So we can decode the r'^{th} row. Also, we download exactly k equations in k variables for row $r' = 1$. This completes the decodability proof. ■

Claim 5.2.2: The rate achieved by this scheme is $1 - \frac{d}{n}$.

Proof: Following the arguments in Equation (5.9) of Claim 5.1.2, we have

$$H(A_i^{(\hat{m})}) = d - \frac{k-1}{2} \tag{5.22}$$

So the rate can be written as

$$\begin{aligned} R &= \frac{H(W_{\hat{m}})}{\sum_{i=1}^{n} H(A_i^{(\hat{m})})} \\ &= \frac{(d-k)k + \binom{k+1}{2}}{n(d - \frac{k-1}{2})} \\ &= \frac{k(d - \frac{k-1}{2})}{n(d - \frac{k-1}{2})} \\ &= 1 - \frac{d}{n} \end{aligned} \tag{5.23}$$

■

Claim 5.2.3: The retrieval scheme described here is private in the sense described in Chapter 4.

Proof: The proof is similar to the one proved in Claim 5.1.3 and hence not repeated here. ■

Remark : Observe that both for the MSR and MBR points, the proposed algorithms have rates that are independent of the number of messages m.

Remark (Case of even k): The implicit assumption in Algorithm 1 is that of k to be an odd number as otherwise the value of $\frac{k+1}{2}$ becomes a fraction. Another alternative way to think of this requirement is that, since we are essentially retrieving the matrix S in equation (5.18) from the $\frac{k+1}{2}$ queries from the set of nodes $K = [n] \setminus D$, the download is optimized since S has exactly $\frac{k(k+1)}{2}$ independent message symbols. The case of even value of k can also be tackled by allowing asymmetrical traffic from the set K that is downloading different number of symbols from different nodes and judiciously choosing the matrices $X_{i_l}^{(\hat{m})}$ which now will have different number of rows to conform with the asymmetrical traffic for $l \in [k]$. This modification does not violate the privacy guarantees as long as the asymmetrical traffic across the k nodes of K are distributed independently of the desired message index.

Example 1: Let us understand the above described scheme through an example. We take a PM MBR code with parameters $[n = 7, k = 3, d = 4, \alpha = 4, \beta = 1, B = 9]$ and three messages $\{W_j : j = 1, 2, 3\}$. Message W_j is represented in the matrix form as M_j and the encoding matrix ψ has the Vandermonde structure over $\mathbb{F}_{11}$.

$$M_j = \begin{bmatrix} S_j & T_j \\ T_j^t & 0 \end{bmatrix} = \begin{bmatrix} u_{j,1} & u_{j,2} & u_{j,3} & u_{j,7} \\ u_{j,2} & u_{j,4} & u_{j,5} & u_{j,8} \\ u_{j,3} & u_{j,5} & u_{j.6} & u_{j.9} \\ u_{j.7} & u_{j,8} & u_{j,9} & 0 \end{bmatrix} \tag{5.24}$$

$$\psi = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 5 \\ 1 & 4 & 5 & 9 \\ 1 & 5 & 3 & 4 \\ 1 & 6 & 3 & 7 \\ 1 & 7 & 5 & 2 \end{bmatrix} \tag{5.25}$$

The contents of the first three nodes are given in Table 1 for better understanding.

Say the user wants to retrieve W_2. The query matrices $\{Q_i\}$s are 3×12 matrices of the form in Equation (5.1) with U chosen to be a random matrix over $\mathbb{F}_{11}$ and $V_i^{(\hat{m})}$ is of the form

$$V_i^{(\hat{m})} = \begin{bmatrix} O_{(3\times 4)} & X_i^{(\hat{m})} & O_{(3\times 4)} \end{bmatrix} \tag{5.26}$$

Without loss of generality , let us choose the set D to be the nodes $\{4, 5, 6, 7\}$ and the set K to be $\{1, 2, 3\}$. So

$$Q_4 = Q_5 = Q_6 = Q_7 = U \tag{5.27}$$

Node 1	Node 2	Node 3
$u_{1,1}+u_{1,2}+u_{1,3}+u_{1,7}$	$u_{1,1}+2u_{1,2}+4u_{1,3}+8u_{1,7}$	$u_{1,1}+3u_{1,2}+9u_{1,3}+5u_{1,7}$
$u_{1,2}+u_{1,4}+u_{1,5}+u_{1,8}$	$u_{1,2}+2u_{1,4}+4u_{1,5}+8u_{1,8}$	$u_{1,2}+3u_{1,4}+9u_{1,5}+5u_{1,8}$
$u_{1,3}+u_{1,5}+u_{1,6}+u_{1,9}$	$u_{1,3}+2u_{1,5}+4u_{1,6}+8u_{1,9}$	$u_{1,3}+3u_{1,5}+9u_{1,6}+5u_{1,9}$
$u_{1,7}+u_{1,8}+u_{1,9}$	$u_{1,7}+2u_{1,8}+4u_{1,9}$	$u_{1,7}+3u_{1,8}+9u_{1,9}$
$u_{2,1}+u_{2,2}+u_{2,3}+u_{2,7}$	$u_{2,1}+2u_{2,2}+4u_{2,3}+8u_{2,7}$	$u_{2,1}+3u_{2,2}+9u_{2,3}+5u_{2,7}$
$u_{2,2}+u_{2,4}+u_{2,5}+u_{2,8}$	$u_{2,2}+2u_{2,4}+4u_{2,5}+8u_{2,8}$	$u_{2,2}+3u_{2,4}+9u_{2,5}+5u_{2,8}$
$u_{2,3}+u_{2,5}+u_{2,6}+u_{2,9}$	$u_{2,3}+2u_{2,5}+4u_{2,6}+8u_{2,9}$	$u_{2,3}+3u_{2,5}+9u_{2,6}+5u_{2,9}$
$u_{2,7}+u_{2,8}+u_{2,9}$	$u_{2,7}+2u_{2,8}+4u_{2,9}$	$u_{2,7}+3u_{2,8}+9u_{2,9}$
$u_{3,1}+u_{3,2}+u_{3,3}+u_{3,7}$	$u_{3,1}+2u_{3,2}+4u_{3,3}+8u_{3,7}$	$u_{3,1}+3u_{3,2}+9u_{3,3}+5u_{3,7}$
$u_{3,2}+u_{3,4}+u_{3,5}+u_{3,8}$	$u_{3,2}+2u_{3,4}+4u_{3,5}+8u_{3,8}$	$u_{3,2}+3u_{3,4}+9u_{3,5}+5u_{3,8}$
$u_{3,3}+u_{3,5}+u_{3,6}+u_{3,9}$	$u_{3,3}+2u_{3,5}+4u_{3,6}+8u_{3,9}$	$u_{3,3}+3u_{3,5}+9u_{3,6}+5u_{3,9}$
$u_{3,7}+u_{3,8}+u_{3,9}$	$u_{3,7}+2u_{3,8}+4u_{3,9}$	$u_{3,7}+3u_{3,8}+9u_{3,9}$

Table 5.1: Content of first three nodes in Example 1

$$
\begin{aligned}
Q_1 &= U + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix}\\
Q_2 &= U + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix}\\
Q_3 &= U + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix}
\end{aligned}
\tag{5.28}
$$

The answer strings from the nodes can be wriiten as

$$A_i = UY_i = U\overline{M}\psi_i^t \quad \forall \quad i \in \{4,5,6,7\} \tag{5.29}$$

where $\overline{M}$ is the matrix $[\ M_1^t \mid M_2^t \mid M_3^t\ ]^t$. Writing this in matrix form:

$$\begin{bmatrix} A_4 & A_5 & A_6 & A_7 \end{bmatrix} = U\overline{M} \begin{bmatrix} 1&1&1&1\\ 4&5&6&7\\ 5&3&3&5\\ 9&4&7&2 \end{bmatrix} \tag{5.30}$$

from which we find the 3×4 matrix $U\overline{M}$. The answer strings from the rest of the nodes are

$$
\begin{aligned}
A_1 &= U\overline{M}\psi_1^t + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix} Y_1\\
&= U\overline{M}\psi_1^t + \begin{bmatrix} Y_1^2(1) & Y_1^2(2) & Y_1^2(4) \end{bmatrix}^t\\
A_2 &= U\overline{M}\psi_2^t + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix} Y_2\\
&= U\overline{M}\psi_2^t + \begin{bmatrix} Y_2^2(1) & Y_2^2(2) & Y_2^2(4) \end{bmatrix}^t\\
A_3 &= U\overline{M}\psi_3^t + \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0 \end{bmatrix} Y_3\\
&= U\overline{M}\psi_3^t + \begin{bmatrix} Y_3^2(1) & Y_3^2(3) & Y_3^2(4) \end{bmatrix}^t
\end{aligned}
\tag{5.31}
$$

Since we already know the matrix $U\overline{M}$, we can cancel the interference of $U\overline{M}\psi_i^t$ from A_i for $i \in \{1,2,3\}$ and get the matrix

$$\begin{bmatrix} Y_1^2(1) & Y_2^2(1) & Y_3^2(1)\\ Y_1^2(2) & Y_2^2(2) & Y_3^2(3)\\ Y_1^2(4) & Y_2^2(4) & Y_3^2(4) \end{bmatrix} \tag{5.32}$$

From the last row we can decode $u_{2,7}, u_{2,8}, u_{2,9}$ which in turn causes the three first row elements to be three linearly independent equations in three unknowns $u_{2,1}, u_{2,2}, u_{2,3}$ and can also be decoded. The first two elements in row 2 are now functions of two unknowns $u_{2,4}, u_{2,5}$ only and can be decoded and the last unknown message symbol $u_{2,6}$ is decoded from the third element of the second row since all other variables are known. Hence the message is decoded correctly. The privacy of the scheme is also apparent.

Remark: We would like to mention here that these schemes are very similar to the ones independently proposed in [27] and [32].

Chapter 6

PIR Capacity of MSR Codes

In this section, we derive the PIR capacity for all MSR codes. It turns out that the capacity expression is the same as that of an $[n, k]$ scalar MDS code as derived in [1]. In this regard, we would like to mention that similar observations were also made in [29] and an achievability scheme identical to the one described in the next chapter and applicable to a much broader class of codes was proposed.

6.1 Converse Proof

Theorem 1. *For an $[n, k, d, \alpha, \beta, B]$ MSR coded distributed storage system, storing m files of equal size $L = B$, the PIR capacity is*

$$C_{MSR} = \frac{1 - (\frac{k}{n})}{1 - (\frac{k}{n})^m} \tag{6.1}$$

i.e., for a fixed number of files, the capacity depends only on the ratio $\frac{k}{n}$ and is independent of the regenerating parameters d, α, β.

Intuitively, this follows simply from the fact that MSR codes are, in fact, vector MDS codes. We formally prove this by first proving, in this section, that the RHS expression of Equation (6.1) is indeed an upper bound on the download rate of any such PIR scheme for MSR coded databases. The achievability scheme described in Chapter 6.2 establishes equality, proving Theorem 1. We proceed by first observing the following property:

Lemma 6.1: For any PIR scheme for an $[n, k, d, \alpha, \beta, B]$ MSR coded database with m independent messages each of size $L = B$, for any set $K = \{i_1, i_2, \ldots, i_k\} \subset [n]$ with $|K| = k$ and any $j \in [m]$, the following property is satisfied

$$H(A_K^{(j)}|Q_K^{(j)}) = \sum_{a=1}^{k} H(A_{i_a}^{(j)}|Q_{i_a}^{(j)}) \tag{6.2}$$

Proof: Following the system model and parameter definitions of Chapter 4, without loss of generality we assume K to be the set of first k nodes $\{1, 2, \ldots, k\}$. We first prove that,

$$H(Y_K) = \sum_{i=1}^{k} H(Y_i) \tag{6.3}$$

Due to the mutual independence of messages (equations (4.3)-(4.5)), we need only focus on the variables $\{Y_i^j : i \in [n]\}$ for a specific $j \in [m]$. Note that, $H(Y_i^j) = \alpha \ \ \forall\, i \in [n]$, i.e., each node stores α coded symbols corresponding to message W_j. From the variables $\{Y_i^j : i \in K\}$, the user should be able to retrieve the message W_j. So

$$\begin{aligned} L = H(W_j) &\leq H(Y_K^j) \\ &\leq \sum_{i=1}^{k} H(Y_i^j) \\ &\leq k\alpha \end{aligned} \tag{6.4}$$

But for MSR codes, $L = k\alpha$. This means all inequalities are in fact satisfied with equality and hence

$$H(Y_K^j) = \sum_{i=1}^{k} H(Y_i^j) \tag{6.5}$$

which gives

$$H(Y_K) = \sum_{i=1}^{k} H(Y_i) \tag{6.6}$$

Since the answer strings $A_i^{(\hat{m})}$ are functions of $(Y_i, Q_i^{(\hat{m})})$, the variables $\{A_i^{(\hat{m})} : i \in K\}$ are conditionally independent conditioned on $\{Q_i^{(\hat{m})} : i \in K\}$.

$$H(A_K^{(\hat{m})}|Q_K^{(\hat{m})}) = \sum_{i=1}^{k} H(A_i^{(\hat{m})}|Q_i^{(\hat{m})}) \tag{6.7}$$

■

With this result, we proceed to the capacity derivation in the same way as [1]. We reproduce the derivation here just for the sake of completeness. We state the two important lemmas that will be used. The proof of these two lemmas is directly from [1].

Lemma 6.2: The interference from undesired messages is upper bounded as

$$L\left(\frac{1}{R} - 1 + \frac{o(L)}{L}\right) \geq I(W_{2:m}; Q_{1:n}^{(1)}, A_{1:n}^{(1)}|W_1)$$

Proof:

$$\begin{aligned}
&I(W_{2:m}; Q_{1:n}^{(1)}, A_{1:n}^{(1)}|W_1) \\
&= I(W_{2:m}; Q_{1:n}^{(1)}, A_{1:n}^{(1)}, W_1) && (6.8)\\
&= I(W_{2:m}; Q_{1:n}^{(1)}, A_{1:n}^{(1)}) + I(W_{2:m}; W_1|Q_{1:n}^{(1)}, A_{1:n}^{(1)}) \\
&= I(W_{2:m}; Q_{1:n}^{(1)}) + I(W_{2:m}; A_{1:n}^{(1)}|Q_{1:n}^{(1)}) + o(L) && (6.9)\\
&= I(W_{2:m}; A_{1:n}^{(1)}|Q_{1:n}^{(1)}) + o(L) && (6.10)\\
&= H(A_{1:n}^{(1)}|Q_{1:n}^{(1)}) - H(A_{1:n}^{(1)}|Q_{1:n}^{(1)}, W_{2:m}) + o(L) \\
&\leq \sum_{i=0}^{n} H(A_i^{(1)}) - H(W_1, A_{1:n}^{(1)}|Q_{1:n}^{(1)}, W_{2:m}) \\
&\quad + H(W_1|Q_{1:n}^{(1)}, W_{2:m}, A_{1:n}^{(1)}) + o(L) && (6.11)\\
&= \frac{L}{R} - H(A_{1:n}^{(1)}|Q_{1:n}^{(1)}, W_{1:m}) \\
&\quad - H(W_1|Q_{1:n}^{(1)}, W_{2:m}) + o(L) && (6.12)\\
&= \frac{L}{R} - L + o(L) && (6.13)\\
&= L\left(\frac{1}{R} - 1 + \frac{o(L)}{L}\right)
\end{aligned}$$

where (6.8) follows from the independence of messages, (6.9) follows from (4.15), (6.10) follows from (4.13), (6.11) follows from chain rule of entropy, (6.12) follows from (4.14) and (6.13) follows from (4.13), (4.14) and message independence. ■

Lemma 6.3:

$$\begin{aligned}
I(W_{\hat{m}:m}; Q_{1:n}^{(\hat{m}-1)}, A_{1:n}^{(\hat{m}-1)}|W_{1:\hat{m}-1}) \;\geq\; &\frac{k}{n} I(W_{\hat{m}+1:m}; Q_{1:n}^{(\hat{m})}, A_{1:n}^{(\hat{m})}|W_{1:\hat{m}}) \\
&+ \frac{kL(1 - \frac{o(L)}{L})}{n}
\end{aligned}$$

Proof:

$$I(W_{\hat{m}:m}; Q_{1:n}^{(\hat{m}-1)}, A_{1:n}^{(\hat{m}-1)}|W_{1:\hat{m}-1})$$

$$
\begin{aligned}
&= \frac{1}{\binom{n}{k}}\binom{n}{k} I(W_{\hat{m}:m}; Q_{1:n}^{(\hat{m}-1)}, A_{1:n}^{(\hat{m}-1)} | W_{1:\hat{m}-1}) \\
&\geq \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} I(W_{\hat{m}:m}; Q_K^{(\hat{m}-1)}, A_K^{(\hat{m}-1)} | W_{1:\hat{m}-1}) && (6.14) \\
&= \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} I(W_{\hat{m}:m}; A_K^{(\hat{m}-1)} | Q_K^{(\hat{m}-1)}, W_{1:\hat{m}-1}) && (6.15) \\
&= \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} \Big[H(A_K^{(\hat{m}-1)} | Q_K^{(\hat{m}-1)}, W_{1:\hat{m}-1}) \\
&\quad - H(A_K^{(\hat{m}-1)} | Q_K^{(\hat{m}-1)}, W_{1:m}) \Big] \\
&= \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} \sum_{i\in K} H(A_i^{(\hat{m}-1)} | Q_i^{(\hat{m}-1)}, W_{1:\hat{m}-1}) && (6.16) \\
&= \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} \sum_{i\in K} H(A_i^{(\hat{m})} | Q_i^{(\hat{m})}, W_{1:\hat{m}-1}) && (6.17) \\
&= \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} [H(A_K^{(\hat{m})} | Q_K^{(\hat{m})}, W_{1:\hat{m}-1}) && (6.18) \\
&\geq \frac{1}{\binom{n}{k}}\binom{n}{k} \sum_{K\subset N:|K|=k} [H(A_K^{(\hat{m})} | Q_{1:n}^{(\hat{m})}, W_{1:\hat{m}-1}) \\
&\geq \frac{k}{n} H(A_{1:n}^{(\hat{m})} | W_{1:\hat{m}-1}, Q_{1:n}^{(\hat{m})}) && (6.19) \\
&= \frac{k}{n} I(W_{\hat{m}:m}; A_{1:n}^{(\hat{m})} | W_{1:\hat{m}-1}, Q_{1:n}^{(\hat{m})}) && (6.20) \\
&= \frac{k}{n} I(W_{\hat{m}:m}; A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}-1}) && (6.21) \\
&= \frac{k}{n} \Big[I(W_{\hat{m}:m}; A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}-1}) \\
&\quad + I(W_{\hat{m}:m}; W_{\hat{m}} | A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})}, W_{1:\hat{m}-1}) - o(L) \Big] && (6.22) \\
&= \frac{k}{n} [I(W_{\hat{m}:m}; W_{\hat{m}}, A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}-1}) - o(L)] \\
&= \frac{k}{n} [I(W_{\hat{m}:m}; W_{\hat{m}} | W_{1:\hat{m}-1}) + I(W_{\hat{m}:m}; A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}}) - o(L)] \\
&= \frac{k}{n} [L + I(W_{\hat{m}+1:m}; A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}}) - o(L)] \\
&= \frac{k}{n} I(W_{\hat{m}+1:m}; A_{1:n}^{(\hat{m})}, Q_{1:n}^{(\hat{m})} | W_{1:\hat{m}}) + \frac{kL(1-\frac{o(L)}{L})}{n}
\end{aligned}
$$

where (6.14) follows from properties of mutual information, (6.15) follows from (4.14), (6.16) follows from Lemma 5.1, (6.17) follows from (4.12) and (4.14), (6.18) again follows from Lemma 5.1, (6.19) follows from Han's inequality, (6.20) follows from (4.14), (6.21) follows from (4.13), (6.22) follows from (4.15) and message independence. ■

Proof of the upper bound on rate: With these lemmas we proceed to prove the upper bound on the PIR download rate. We start with Lemma 6.2 :

$$
\begin{aligned}
L\left(\frac{1}{R} - 1 + \frac{o(L)}{L}\right) &\geq I(W_{2:m}; Q_{1:n}^{(1)}, A_{1:n}^{(1)} | W_1)() && (6.23) \\
&\geq \frac{k}{n} I(W_{3:m}; A_{1:n}^{(2)}, Q_{1:n}^{(2)} | W_{1:2}) + \frac{kL(1-\frac{o(L)}{L})}{n} \\
&\geq \dots \\
&\geq (\frac{k}{n})^{m-2} I(W_{m:m}; A_{1:n}^{(m-1)}, Q_{1:n}^{(m-1)} | W_{1:m-1}) \\
&\quad + (\frac{k}{n} + (\frac{k}{n})^2 + \dots + (\frac{k}{n})^{m-2}) L(1 - \frac{o(L)}{L}) \\
&\geq (\frac{k}{n} + (\frac{k}{n})^2 + \dots + (\frac{k}{n})^{m-1}) L(1 - \frac{o(L)}{L})
\end{aligned}
$$

where (6.23) is from Lemma 6.2 and subsequent steps are by using Lemma 6.3 successively. By rearranging we have

$$\frac{1}{R} \geq (1 + \frac{k}{n} + (\frac{k}{n})^2 + \ldots + (\frac{k}{n})^{m-1})(1 - \frac{o(L)}{L}) \tag{6.24}$$

By taking $L \to \infty$, $\frac{o(L)}{L} \to 0$ and we have

$$\begin{aligned} R &\leq \frac{1}{\sum_{i=0}^{m-1} (\frac{k}{n})^i} \\ &= \frac{1 - \frac{k}{n}}{1 - (\frac{k}{n})^m} \end{aligned} \tag{6.25}$$

□

6.2 Achievability of the PIR Capacity (based on [1])

In this section we propose a PIR scheme that achieves the PIR rate upper bound of MSR codes with equality and hence completes the proof of Theorem 1. Since the rate of this scheme is equal to capacity, this scheme performs strictly better than any algorithms previously suggested for such codes. The algorithm possesses the properties of a PIR scheme described in [3] i.e. symmetry across messages, symmetry across databases and efficient exploitation of side information and are, in intuition, similar to the algorithm for MDS coded databases described by Banawan *et al.* in [1] and to the achievability scheme described in [29] for a much broader class of array codes. To build the intuition, we will first discuss an example of the original scheme of [1] for MDS coded databases and then extend the techniques to MSR codes by using the fact that MSR codes are essentially vector MDS codes. Afterwards, we shall give a similar scheme for MBR PM codes.

6.2.1 PIR Scheme for MDS Codes

For an $[n, k]$ MDS coded database, with m messages each of size $L = kn^m$, each "stripe" of length k is encoded using the $[n, k]$ scalar MDS code and stored across the n nodes. For every such coded stripe, downloading any k out of n coded symbols is sufficient to decode that stripe due to the MDS property. For the undesired messages, such a stripe, once decoded, acts as side information that can be exploited while downloading desired coded symbols from the $n-k$ nodes that did not originally participate in the download of the stripe (of the undesired message). This is the fundamental idea for the efficient use of side information that leads to the capacity-achieving property of the scheme of [1].

Example 1: We present an example with small parameter values: $n = 3, k = 2, m = 3, L = kn^m = 2 \cdot 3^3 = 54, \hat{m} = 1$. The L symbols (each belonging to $\mathbb{F}_q$) of each message is divided into $L' = n^m$ stripes of length k and coded using a $[3, 2]$ MDS code. For each message there is a unique permutation $\pi_j, j \in [m]$, over the L' indices of the stripes which is chosen randomly over all such permutations and is known only at the user end. We denote by $C_r^{(j)}$ as the r^{th} coded stripe (after permutation is applied to the indices) of the j^{th} message for $r \in [L'], j \in [m]$. The i^{th} coded symbol of $C_r^{(j)}$ stored in node $i, i \in [n]$, is denoted by $C_{i,r}^{(j)}$. If the query Q_i for the i^{th} node contains $C_{i,r}^{(j)}$ that means the node is asked to return the $\pi_j^{-1}(r)^{th}$ coded symbol of the j^{th} message. The following Table 6.1 gives the queries to the 3 nodes for this example that achieve the PIR capacity of MDS codes.

Note that, for any message $j \in [m]$, any stripe index $r \in [L']$ appears at most once in a query Q_i, whether as an individual symbol or in a sum of symbols. From a node's perspective, the total number of symbols requested from a message (both as individual symbol and as part of sum) is the same for all messages and this holds for all nodes. The actual stripe indices of a message requested from a node is determined by the random permutation on the user end which is unknown to the node. Finally another private randomly chosen permutation is applied to each query itself to remove any possibility of inference of the desired index from the order of $C_{i,r}^{(j)}$s in the query.

$Q_1^{(1)}$	$Q_2^{(1)}$	$Q_3^{(1)}$
$C_{1,1}^{(1)}$	$C_{2,1}^{(1)}$	$C_{3,2}^{(1)}$
$C_{1,2}^{(1)}$	$C_{2,3}^{(1)}$	$C_{3,3}^{(1)}$
$C_{1,4}^{(1)}$	$C_{2,4}^{(1)}$	$C_{3,5}^{(1)}$
$C_{1,5}^{(1)}$	$C_{2,6}^{(1)}$	$C_{3,6}^{(1)}$
$C_{1,7}^{(1)}$	$C_{2,7}^{(1)}$	$C_{3,8}^{(1)}$
$C_{1,8}^{(1)}$	$C_{2,9}^{(1)}$	$C_{3,9}^{(1)}$
$C_{1,10}^{(1)}$	$C_{2,10}^{(1)}$	$C_{3,11}^{(1)}$
$C_{1,11}^{(1)}$	$C_{2,12}^{(1)}$	$C_{3,12}^{(1)}$
$C_{1,1}^{(2)}$	$C_{2,1}^{(2)}$	$C_{3,2}^{(2)}$
$C_{1,2}^{(2)}$	$C_{2,3}^{(2)}$	$C_{3,3}^{(2)}$
$C_{1,4}^{(2)}$	$C_{2,4}^{(2)}$	$C_{3,5}^{(2)}$
$C_{1,5}^{(2)}$	$C_{2,6}^{(2)}$	$C_{3,6}^{(2)}$
$C_{1,7}^{(2)}$	$C_{2,7}^{(2)}$	$C_{3,8}^{(2)}$
$C_{1,8}^{(2)}$	$C_{2,9}^{(2)}$	$C_{3,9}^{(2)}$
$C_{1,10}^{(2)}$	$C_{2,10}^{(2)}$	$C_{3,11}^{(2)}$
$C_{1,11}^{(2)}$	$C_{2,12}^{(2)}$	$C_{3,12}^{(2)}$
$C_{1,1}^{(3)}$	$C_{2,1}^{(3)}$	$C_{3,2}^{(3)}$
$C_{1,2}^{(3)}$	$C_{2,3}^{(3)}$	$C_{3,3}^{(3)}$
$C_{1,4}^{(3)}$	$C_{2,4}^{(3)}$	$C_{3,5}^{(3)}$
$C_{1,5}^{(3)}$	$C_{2,6}^{(3)}$	$C_{3,6}^{(3)}$
$C_{1,7}^{(3)}$	$C_{2,7}^{(3)}$	$C_{3,8}^{(3)}$
$C_{1,8}^{(3)}$	$C_{2,9}^{(3)}$	$C_{3,9}^{(3)}$
$C_{1,10}^{(3)}$	$C_{2,10}^{(3)}$	$C_{3,11}^{(3)}$
$C_{1,11}^{(3)}$	$C_{2,12}^{(3)}$	$C_{3,12}^{(3)}$
$C_{1,13}^{(1)} + C_{1,3}^{(2)}$	$C_{2,13}^{(1)} + C_{2,2}^{(2)}$	$C_{3,14}^{(1)} + C_{3,1}^{(2)}$
$C_{1,14}^{(1)} + C_{1,6}^{(2)}$	$C_{2,15}^{(1)} + C_{2,5}^{(2)}$	$C_{3,15}^{(1)} + C_{3,4}^{(2)}$
$C_{1,16}^{(1)} + C_{1,9}^{(2)}$	$C_{2,16}^{(1)} + C_{2,8}^{(2)}$	$C_{3,17}^{(1)} + C_{3,7}^{(2)}$
$C_{1,17}^{(1)} + C_{1,12}^{(2)}$	$C_{2,18}^{(1)} + C_{2,11}^{(2)}$	$C_{3,18}^{(1)} + C_{3,10}^{(2)}$
$C_{1,19}^{(1)} + C_{1,3}^{(3)}$	$C_{2,19}^{(1)} + C_{2,2}^{(3)}$	$C_{3,20}^{(1)} + C_{3,1}^{(3)}$
$C_{1,20}^{(1)} + C_{1,6}^{(3)}$	$C_{2,21}^{(1)} + C_{2,5}^{(3)}$	$C_{3,21}^{(1)} + C_{3,4}^{(3)}$
$C_{1,22}^{(1)} + C_{1,9}^{(3)}$	$C_{2,22}^{(1)} + C_{2,8}^{(3)}$	$C_{3,23}^{(1)} + C_{3,7}^{(3)}$
$C_{1,23}^{(1)} + C_{1,12}^{(3)}$	$C_{2,24}^{(1)} + C_{2,11}^{(3)}$	$C_{3,24}^{(1)} + C_{3,10}^{(3)}$
$C_{1,13}^{(2)} + C_{1,13}^{(3)}$	$C_{2,13}^{(2)} + C_{2,13}^{(3)}$	$C_{3,14}^{(2)} + C_{3,14}^{(3)}$
$C_{1,14}^{(2)} + C_{1,14}^{(3)}$	$C_{2,15}^{(2)} + C_{2,15}^{(3)}$	$C_{3,15}^{(2)} + C_{3,15}^{(3)}$
$C_{1,16}^{(2)} + C_{1,16}^{(3)}$	$C_{2,16}^{(2)} + C_{2,16}^{(3)}$	$C_{3,17}^{(2)} + C_{3,17}^{(3)}$
$C_{1,17}^{(2)} + C_{1,17}^{(3)}$	$C_{2,18}^{(2)} + C_{2,18}^{(3)}$	$C_{3,18}^{(2)} + C_{3,18}^{(3)}$
$C_{1,25}^{(1)} + C_{1,15}^{(2)} + C_{1,15}^{(3)}$	$C_{2,25}^{(1)} + C_{2,14}^{(2)} + C_{2,14}^{(3)}$	$C_{1,26}^{(1)} + C_{2,13}^{(2)} + C_{3,13}^{(3)}$
$C_{1,26}^{(1)} + C_{1,18}^{(2)} + C_{1,18}^{(3)}$	$C_{2,27}^{(1)} + C_{2,17}^{(2)} + C_{2,17}^{(3)}$	$C_{1,27}^{(1)} + C_{2,16}^{(2)} + C_{3,16}^{(3)}$

Table 6.1: Queries for MDS coded database with $n = 3.k = 2$

6.2.2 PIR Scheme for MSR codes

With the idea of the MDS PIR scheme in mind, we move to constructing schemes for MSR codes. The simple intuition for this is that MSR codes have the vector MDS property and the PIR scheme discussed in the last section is not restricted to scalar symbols. We shall follow the general MSR code framework rather than work with a more specific setting like the Product-Matrix structure. We take an $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B} = k\tilde{\alpha}]$ MSR code for some feasible parameter values $\tilde{\alpha}, \tilde{\beta}$. Let $\mathbf{x}$ be a $k\tilde{\alpha}$ length message vector : $\mathbf{x} = [x_1 \;\; x_2 \;\; \ldots \;\; x_k]^t$ where each $x_i \in \mathbb{F}_q^{\tilde{\alpha}}, \forall\ i \in [k]$ is a row vector of dimension $\tilde{\alpha}$. The $n\tilde{\alpha} \times k\tilde{\alpha}$ systematic encoding matrix G can be written as

$$G = \begin{bmatrix} I_{\tilde{\alpha}} & & 0 \\ & \ddots & \\ 0 & & I_{\tilde{\alpha}} \\ A_{1,1} & \ldots & A_{1,k} \\ \vdots & \ddots & \vdots \\ A_{n-k,1} & \ldots & A_{n-k,k} \end{bmatrix} \tag{6.26}$$

where $I_{\tilde{\alpha}}$ is the $\tilde{\alpha} \times \tilde{\alpha}$ identity matrix and $A_{u,v}$ for $u \in [n-k], v \in [k]$, are $\tilde{\alpha} \times \tilde{\alpha}$ encoding matrices such that the desired MSR properties are satisfied. The encoded vector $\mathbf{y}$ of length $n\tilde{\alpha}$ is simply $\mathbf{y} = G\mathbf{x} = [y_1 \;\; y_2 \;\; \ldots \;\; y_n]^t$ where $y_i \in \mathbb{F}_q^{\tilde{\alpha}}$ is the row vector of $\tilde{\alpha}$ coded symbols stored in the i^{th} node, i.e., $y_i = G_i\mathbf{x}$ where G_i is the matrix G restricted to the rows $(i-1)\tilde{\alpha}+1$ to $i\tilde{\alpha}$. The assumption of systematic encoding is without loss of generality.

We take the message sizes to be $L = \tilde{\alpha}kn^m$. We divide the long string of message symbols into stripes of $k\tilde{\alpha}$ message symbols. Each stripe is encoded independently using the above MSR code to $n\tilde{\alpha}$ coded symbols and stored across the n nodes as described above. Since, the regeneration property of each such stripe is independently preserved, we can think of this as each message of size L being encoded using an $[n, k, d, \alpha = \tilde{\alpha}n^m, \beta = \tilde{\beta}n^m, B = \tilde{B}n^m = \tilde{\alpha}kn^m]$ MSR code. For message j, we denote the r^{th} stripe of size $k\tilde{\alpha}$ as the vector $B_r^{(j)}$ for $r \in [n^m]$. For this stripe the corresponding $\tilde{\alpha}$ coded symbols stored in node i are $G_iB_r^{(j)}$.
Initial Random Permutation : For each message $j \in [m]$, we choose a random permutation π_j over the stripe indices $[n^m]$ such that

$$G_iB_{\pi_j(r)}^{(j)} = C_{i,r}^{(j)} \quad \forall\ j \in [m],\ i \in [n],\ r \in [n^m] \tag{6.27}$$

This permutation for each message is privately and independently chosen by the user and the database has no information regarding this. Notice the difference from the notation of the previous section. Instead of a single coded symbol in $\mathbb{F}_q$, $C_{i,r}^{(j)}$ now corresponds to a vector (in $\mathbb{F}_q^{\tilde{\alpha}}$) of coded symbols of the $\pi_j(r)^{th}$ stripe of message W_j stored in the i^{th} node. Additionally, we introduce the term 'z-sum', for $1 \leq z \leq m$, to denote a sum of the form $C_{i,r_1}^{(j_1)} + C_{i,r_2}^{(j_2)} + \ldots + C_{i,r_z}^{(j_z)}$ where addition is defined over the vector space $\mathbb{F}_q^{\tilde{\alpha}}$ and the message indices $\{j_1, j_2, \ldots, j_z\}$ are all distinct i.e., each *component* of the sum belongs to a different message. The *type* of a z-sum is described by the z-tuple $(j_1, j_2, \ldots, j_z)$ involved in the z-sum. For a fixed value of $z \in [m]$, there are exactly $\binom{m}{z}$ different *types* of z-sum. A 1-sum is trivially defined as simply $C_{i,r_1}^{(j_1)}$. An *instance* of a $(j_1, j_2, \ldots, j_z)$-type z-sum corresponds to a possible value of the tuple $(r_1, r_2, \ldots, r_z)$ where $r_i \in [n^m], \forall i \in [z]$. To *increment* an instance means to increment each component of it by 1 i.e., incrementing $(r_1, r_2, \ldots, r_z)$ gives us $(r_1+1, r_2+1, \ldots, r_z+1)$. For a type $(j_1, j_2, \ldots, j_z)$ not containing the desired message index, the set of z-tuples $(r_1, r_2, \ldots, r_z)$ corresponding to the instances that are downloaded from node i is denoted by $\mathcal{I}_{i,z}$. It will be clear later that $\mathcal{I}_{i,z}$ depends only on i and z and does not depend on the specific values of $\{j_1, j_2, \ldots, j_z\}$. A query $Q_i^{(\hat{m})}$, for node i, consists of a collection of sub-queries where each sub-query is a request to node i to return a specific instance of a z-sum. With this initial setup and the notation explained, we now give the algorithm for retrieval of message $W_{\hat{m}}$.

1. *Initial download of desired message:* For the desired message (with index $\hat{m}$), starting from node 1, we download the $\tilde{\alpha}$-dimensional vector related to stripe 1 from subsequent k nodes i.e. we download $C_{1,1}^{(\hat{m})}, C_{2,1}^{(\hat{m})}, \ldots, C_{k,1}^{(\hat{m})}$ from nodes $1, 2, \ldots, k$ respectively. The $\tilde{\alpha}$-dimensional vectors corresponding to stripe 2 of the desired message are downloaded starting from node $k+1$ and proceeding to another k nodes in a round-robin fashion i.e., download $C_{(k+1) \mod n,2}^{(\hat{m})}, C_{(k+2) \mod n,2}^{(\hat{m})},$

$\ldots, C^{(\hat{m})}_{2k \mod n,2}$ from the nodes $(k+1) \mod n, (k+2) \mod n, \ldots, 2k \mod n$ respectively. This is continued up to stripe id $n \cdot k^{m-1}$. After completing this procedure, for each stripe id $r \in [n \cdot k^{m-1}]$, we shall have exactly k $\tilde{\alpha}$-dimensional vectors from k different nodes from which we can decode the original r^{th} stripe of length $k\tilde{\alpha}$ using the vector MDS property.

2. *Preserving message symmetry:* To preserve message symmetry property for every node, we download equal number of symbols (equal number of $\tilde{\alpha}$-dimensional vectors) from each of the other messages from each node. The above procedure is repeated for each $j \in [m] \setminus \{\hat{m}\}$. Note that, due to the initial permutation, the original ids of the stripes which are decoded can be different for different messages i.e., the true ids are not necessarily $1, 2, \ldots, n \cdot k^{m-1}$. Further, it can be verified that, for each message, exactly k^m $\tilde{\alpha}$-dimensional vectors have been downloaded from each node up until now.

3. *Exploitation of side information in next round:* From node i, exactly k^m stripe ids for a message index $j \in [m]$ have been queried. But we know that, cumulatively from all the n nodes, the first $n \cdot k^{m-1}$ stripe ids have been queried and decoded for all the messages. Hence, for a node i, $(n-k) \cdot k^{m-1}$ stripe ids for a message index $j \in [m]$ were not queried from that node and by our definition this corresponds to the set $[n \cdot k^{m-1}] \setminus \mathcal{I}_{i,1}$. For each message index $j \in [m] \setminus \hat{m}$, we download the 2-sum of type $(\hat{m}, j)$ from node i with instances $(r_{\hat{m}}, r_j), \forall\ r_j \in [n \cdot k^{m-1}] \setminus \mathcal{I}_{i,1}$ and $r_{\hat{m}}$ is a stripe id that was not queried in the first step. The value of $r_{\hat{m}}$ is gradually *incremented* starting from $n \cdot k^{m-1} + 1$ in such a way that for every value of $r_{\hat{m}}$, an instance of $(r_{\hat{m}}, r_j)$ is downloaded from exactly k nodes, keeping the constraint on r_j as described above for all $j \in [m] \setminus \hat{m}$.
Up until now, we have been only concerned with 2-sums with types that included the desired message index. To preserve message symmetry, we need to download instances of 2-sums of the type $(j_1, j_2), \quad \forall\ j_1, j_2 \in [m] \setminus \hat{m}, j_1 \neq j_2$. For each such type, the instance (r_1, r_2), is gradually *incremented*, starting from $(n \cdot k^{m-1} + 1, n \cdot k^{m-1} + 1)$, such that each such instance is downloaded from exactly k nodes. This goes on until equal number of instance downloads for all types of 2-sums for all nodes is established. Since, each instance of the form $C^{(j_1)}_{i,r_1} + C^{(j_2)}_{i,r_2}$, where $j_1, j_2 \in [m] \setminus \hat{m}$, is downloaded from exactly k nodes, the sum of the r_1^{th} stripe of W_{j_1} and r_2^{th} stripe of W_{j_2} is decodable by linearity of the encoding scheme. These are used as side information in subsequent rounds.

4. *Subsequent rounds:* In the z^{th} round, we download z-sums. For the types of z-sums including the desired message index, we use the side information of the $(z-1)^{th}$ round. To be more specific, for a type $(\hat{m}, j_1, j_2, \ldots, j_{z-1})$, from node i, we download instances $(r_{\hat{m}}, r_1, r_2, \ldots, r_{z-1}), \forall\ (r_1, r_2, \ldots, r_{z-1}) \in [n^{z-2} \cdot k^{m-z+2} + 1 : n^{z-1} \cdot k^{m-z+1}] \setminus \mathcal{I}_{i,z-1}$ and $r_{\hat{m}}$ is chosen to be a new stripe id starting from $n \cdot k^{m-1} + \sum_{a=1}^{z-2} \binom{m-1}{a} \cdot n \cdot k^{m-a-1} \cdot (n-k)^a + 1$ and incremented in a similar fashion as before. For a type $(j_1, j_2, \ldots, j_z)$ not containing the desired message index, the instance $(r_1, r_2, \ldots, r_z)$ is gradually incremented such that each instance is downloaded from exactly k nodes. This goes on till equal number of instance downloads for all types of z-sums, whether containing the desired message index or not, have been established.

5. *Permutation in the order of the queries:* Finally, for each node, a permutation over the sub-queries of the query for that node is randomly chosen by the user and applied without the database's knowledge. The purpose is to preserve the privacy even in the event of complete knowledge of the retrieval scheme by the nodes. If not for this permutation, the nodes could have inferred the index of the desired message by simply observing the first round of sub-queries.

The following Algorithm 2 gives an explicit technique of building the queries for the above scheme. Note that, the initial and final permutations have not been explicitly written in the algorithm. The analysis of the algorithm including the explanation of notation follows.

Analysis of Algorithm 2 : The algorithm takes as input the parameter values of $[n, k, m]$ and the desired index $\hat{m}$ and outputs the queries Q_i ($Q_i^{(\hat{m})}$ to be consistent with our previous notation) for each node $i \in [n]$. As stated before, the queries are basically collection of sub-queries where each sub-query denotes an instance of a type of z-sum being requested from that node. With all the counters initialized to index 1, we proceed as follows:

Algorithm 2: Determining queries $\{Q_i\}$

Result: Set of Queries $\{Q_i : i \in N\}$ for desired message $\hat{m}$

Input: Number of databases n, Number of messages m, Reconstruction parameter k, Desired message $\hat{m}$

Initialization:
For each message in $[m]$ initialize a block index
counter $count_j$ to 1;
For desired message $\hat{m}$ define $S \equiv [m] \setminus \hat{m}$;
// LOOP1
for *node in* $1 : n \cdot k^m$ **do**
  Add $C^{(\hat{m})}_{((node-1) \mod n)+1, count_{\hat{m}}}$ to $Q_{((node-1) \mod n)+1}$
  if $node \mod k == 0$ **then**
    $count_{\hat{m}} \leftarrow count_{\hat{m}} + 1$;
  end
end
// LOOP2
for z *in* $1 : m-1$ **do**
  $y \leftarrow k^{m-z+1} \cdot (n-k)^{z-1}$
  // SUB-LOOP1
  for *Each subset* R *of* S *such that* $|R| = z$ **do**
    // SUB-SUB-LOOP1
    for *node in* $1 : n \cdot y$ **do**
      Add $C^{R}_{((node-1) \mod n)+1, count_R}$ to $Q_{((node-1) \mod n)+1}$;
      if $node \mod k == 0$ **then**
        $count_R \leftarrow count_R + 1$;
      end
    end
    $count_R \leftarrow count_R - \frac{n \cdot y}{k}$;
    $a \leftarrow 1$; $x \leftarrow a \cdot k$;
    // SUB-SUB-LOOP2
    for *node in* $1 : \frac{n \cdot y \cdot (n-k)}{k}$ **do**
      Add $C^{(\hat{m})}_{(x \mod n)+1, count_{\hat{m}}} + C^{R}_{(x \mod n)+1, count_R}$ to $Q_{(x \mod n)+1}$;
      $x \leftarrow x + 1$;
      if $node \mod (n-k) == 0$ **then**
        $count_R \leftarrow count_R + 1$;
        $a \leftarrow a + 1$; $x \leftarrow a \cdot k$;
      end
      if $node \mod k == 0$ **then**
        $count_{\hat{m}} \leftarrow count_{\hat{m}} + 1$;
      end
    end
  end
end

- LOOP1 visits each node, starting from node 1, in a round-robin fashion and increments $count_{\hat{m}}$ every k iterations. This ensures that for each of the first $n \cdot k^{m-1}$ stripe ids of the desired message, exactly k $\tilde{\alpha}$-dimensional vectors are downloaded from k different nodes and hence that corresponding stripe is decodable due to the vector MDS property. After this initial download phase, we enter LOOP2 which progresses in subsequent rounds from 1 to $m-1$.

- Each iteration of LOOP2 operates for a value of $z \in [m-1]$. SUB-LOOP1 iterates, for a fixed value of z, on all possible types of z-sums not containing the desired message index. For each such type, we do the following:

  In SUB-SUB-LOOP1, z-sums of the form $C^R_{i,count_R} \equiv C^{(j_1)}_{i,count_{j_1}} + C^{(j_2)}_{i,count_{j_2}} + \ldots + C^{(j_z)}_{i,count_{j_z}}$ where $R = (j_1, j_2, \ldots, j_z)$ is the type of the current iteration of SUB-LOOP1 and $count_R = (count_{j_1}, count_{j_2}, \ldots, count_{j_z})$ is an instance and each $j_l \in [m] \setminus \hat{m}, l \in [z]$, are downloaded. The logic in Line 16-18 ensures that each such instance is downloaded from exactly k nodes and hence decodable (as a sum, not individually, for $z > 1$ as explained before). For $z = 1$, SUB-SUB-LOOP1 simply replicates the task performed by LOOP1, but for all the undesired messages. In this way, it establishes message symmetry. The task of using the side information to obtain new desired message symbols is performed by SUB-SUB-LOOP2. Firstly, Line 20 resets the increments done in SUB-SUB-LOOP1. For message W_j, with $j \in [m] \setminus \hat{m}$, if the instance $count_j$ was downloaded from nodes $\{i_1, i_2, \ldots, i_k\}$ in SUB-SUB-LOOP1, then the instance $(count_{\hat{m}}, count_j)$ is downloaded from nodes $[n] \setminus \{i_1, i_2, \ldots, i_k\}$. As explained before, $count_{\hat{m}}$ is a new stripe id which was not queried before and is incremented in a k-round-robin fashion. For $z > 1$, the algorithm proceeds in a sort of recursive manner. SUB-SUB-LOOP1 downloads new instances of types containing only undesired messages, ensuring decodability (as a sum) of these instances for use as side information. For each such instance $count_R$, that is downloaded from a set of nodes $\{i_1, i_2, \ldots, i_k\}$, SUB-SUB-LOOP2 downloads $(z+1)$-sum instance $(count_{\hat{m}}, count_R)$ from nodes $[n] \setminus \{i_1, i_2, \ldots, i_k\}$. This proceeds till $z = m-1$, when the SUB-SUB-LOOP2 downloads z-sums with $z = m$.

So in the z^{th} round exactly $\frac{y \cdot (n-k)}{k}$ $\tilde{\alpha}$-dimensional vectors are downloaded due to SUB-SUB-LOOP2 from each node for each $(z+1)$-sum containing the desired message and in the $(z+1)^{th}$ round these many $\tilde{\alpha}$-dimensional vectors are downloaded due to SUB-SUB-LOOP1 from each node for each $(z+1)$-sum not containing the desired message. This ensures message symmetry and database symmetry.

Claim 6.2.1: The rate of this scheme is $\frac{1-(\frac{k}{n})}{1-(\frac{k}{n})^m}$.

Proof: To calculate the rate, first we show that the number of stripes of desired message downloaded by this scheme is n^m. To do that, we simply need to keep track of the variable $count_{\hat{m}}$ which gets initialized to 1. This variable is incremented in Lines 8 and 30 of Algorithm 2. LOOP1 starting at Line 5 runs $n \cdot k^m$ times incrementing $count_{\hat{m}}$ every k iterations. So the number of increments is $n \cdot k^{m-1}$. Next we proceed to LOOP2, having SUB-LOOP1 at Line 13 running for each possible subset of size z which contains SUB-SUB-LOOP2 at Line 22 which contains the increment operation at Line 30. Careful calculation easily gives us the number of increments of $count_{\hat{m}}$ in the z^{th} round, which turns out to be $n \cdot \binom{m-1}{z} \cdot k^{m-z+1} \cdot (n-k)^{z-1} \cdot \frac{(n-k)}{k} \cdot \frac{1}{k} = n \cdot \binom{m-1}{z} \cdot k^{m-z-1} \cdot (n-k)^z$. So the final value of $count_{\hat{m}}$

$$\begin{aligned} count_{\hat{m}} &= n \cdot k^{m-1} + \sum_{z=1}^{m-1} \binom{m-1}{z} \cdot n \cdot k^{m-z-1} \cdot (n-k)^z \\ &= n \cdot \left[\sum_{z=0}^{m-1} \binom{m-1}{z} \cdot k^{m-z-1} \cdot (n-k)^z \right] \\ &= n^m \end{aligned} \tag{6.28}$$

For each value of $count_{\hat{m}}$, exactly k $\tilde{\alpha}$-dimensional vectors, with same stripe id, from k different nodes are downloaded. By the vector MDS property, the corresponding stripe of message is decodable. So, indeed the complete message is decodable. Now in a similar manner the number of undesired stripes downloaded can be calculated by observing SUB-SUB-LOOP1

at Line 14 of the algorithm. The total number of $\tilde{\alpha}$-dimensional vectors downloaded from all nodes having no component from the desired message is

$$\sum_{z=1}^{m-1} \binom{m-1}{z} \cdot n \cdot k^{m-z+1} \cdot (n-k)^{z-1} = \frac{n^m \cdot k^2 - n \cdot k^{m+1}}{n-k} \tag{6.29}$$

Now we can calculate the rate to be

$$\begin{aligned} R &= \frac{Size\ \ of\ \ message}{Total\ \ number\ \ of\ \ symbols\ \ downloaded} \\ &= \frac{\tilde{\alpha} k n^m}{\tilde{\alpha} k n^m + \tilde{\alpha} \frac{n^m \cdot k^2 - n \cdot k^{m+1}}{n-k}} \\ &= \frac{n^m}{n^m + \frac{n^m \cdot k - n \cdot k^m}{n-k}} \\ &= \frac{n^{m+1} - k \cdot n^m}{n^{m+1} - n \cdot k^m} \\ &= \frac{1 - \frac{k}{n}}{1 - (\frac{k}{n})^m} \end{aligned} \tag{6.30}$$

■

Claim 6.2.2: The scheme is private in the sense of Chapter 4.

Proof: It is clear from the algorithm that for any message a particular stripe id appears at most once in query Q_i of a node, whether as a single vector or z-sum of vectors. Symmetry across messages and across databases are also established since the same number of instances are downloaded for each z-sum irrespective of whether it contains the desired message index or not, and each node contributes equally to each z-sum. Due to the initial permutation over block ids, the true realizations of block ids for a query Q_i is completely random and hence independent of the desired message index. The final permutation of each query further ensures that the order of sub-queries in which they arrive at a node i reveals no information about the desired message index. Hence the condition of privacy in Chapter 4 i.e., $I(j; Q_i^{(j)}) = 0$ is indeed satisfied for this scheme for every $i \in [n]$. ■

6.3 A Similar PIR Scheme for PM MBR Codes

The scheme described in the previous section is optimal in terms of rate and works on any MSR code parameters $[n, k, d, \tilde{\alpha}, \tilde{\beta}]$. In this section, we propose a similar scheme for the product-matrix MBR code that achieves the same rate of $\frac{1-\frac{k}{n}}{1-(\frac{k}{n})^m}$. Since the MBR code also allows you to reconstruct the original message by contacting any k out of n nodes, a simple trivial PIR scheme would be to just repeat Algorithm 2 for PM MBR parameters $[n, k, d, \tilde{\alpha} = d, \tilde{\beta} = 1, \tilde{B} = k(d - \frac{k-1}{2})]$ and file sizes $L = \tilde{B} n^m$. But this causes unnecessary downloads since $k\tilde{\alpha} = kd \geq k(d - \frac{k-1}{2})$. Intuitively, this is because the MBR codes, unlike MSR codes, are not storage optimal.

By using the symmetric structure of the message matrix in the PM MBR encoding procedure described by Equation (3.12), we can reduce this overhead. The idea is that for an $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B}]$ PM MBR code, since there are exactly $\tilde{B}$ independent message symbols, we need exactly $\tilde{B}$ coded symbols out of the total kd coded symbols stored in the k nodes. This is not something new as we have already seen use of this property in the suboptimal scheme of Section 5.2. The following claim formally states and proves this property.

Claim 6.3.1: If a message of size $\tilde{B} = k(d - \frac{k-1}{2})$ is encoded using a $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B}]$ PM MBR code and stored across n nodes, there exists a (possibly asymmetric) download strategy that downloads exactly $\tilde{B}$ coded symbols from any k out of n nodes such that, the user is able to decode the original message.

Proof: For an $[n, k, d, \tilde{\alpha}, \tilde{\beta}, \tilde{B}]$ PM MBR code the coded symbols can be represented by a $d \times n$ matrix C

$$C = M\psi^t \tag{6.31}$$

The coded symbols stored at the i^{th} node for $i \in [n]$ can be written as a $d \times 1$ vector C_i

$$\begin{aligned} C_i &= M\psi(i)^t \\ &= \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \begin{bmatrix} \phi(i) & \Delta(i) \end{bmatrix}^t \\ &= \begin{bmatrix} S\phi(i)^t + T\Delta(i)^t \\ T^t\phi(i)^t \end{bmatrix} \end{aligned} \tag{6.32}$$

Observe that the last $(d-k)$ entries of C_i are functions of the T matrix only. Each row of the T matrix contains k independent message symbols. And so any such row of k symbols is encoded into n symbols by the matrix ϕ^t (of dimension $k \times n$). Since any k rows of ϕ are linearly independent by construction, the encoding is MDS and hence to retrieve the k message symbols it is sufficient to download any k out of n symbols. This property holds for all the last $(d-k)$ rows of C. So we need $k(d-k)$ downloads for retrieving the complete matrix T.

Once we have the matrix T, by using symmetry of M, each of the top k rows of C also becomes an $[n,k]$ MDS code. But we can use the symmetry of the matrix S to further reduce the amount of download. Without loss of generality, we start from the first row. We need k coded symbols to retrieve the k corresponding message symbols of the first row of S. Now considering the second row, notice that due to symmetry we already know one message symbol of this row. Hence we need only download $k-1$ coded symbols for this row to completely decode this row of S. For successive rows we can have further reduction in required downloads until finally at the k^{th} row we only need to download a single coded symbol. This way the complete matrix S can be recovered. The total required download for this strategy is $\sum_{i=1}^{k}(k-i+1) + k(d-k) = k(\frac{k+1}{2}) + k(d-k) = \tilde{B}$. ■

To describe a specific download strategy, we use k d-dimensional binary vectors for each node participating in the reconstruction process. The position of '1's in each vector denotes which coded symbols we need to download from that node. We give an algorithm that finds the necessary vectors for an $[n,k,d,\tilde{\alpha},\tilde{\beta},\tilde{B}]$ PM MBR code. This has the same structure and intuition of Algorithm 1 with the difference that it outputs vectors instead of matrices.

Algorithm 3: Determining download strategy

Result: Set of Vectors $\{x_l : l \in [k]\}$ of length d.

Initialization: Set the last $(d-k)$ entries of each vector to 1 and all other entries to 0.

$r \leftarrow 1, l \leftarrow 1, count \leftarrow 1, thres \leftarrow k$;

while $thres > 0$ **do**

  Set the r^{th} entry of x_l to 1;

  $l \leftarrow (l \ mod \ k) + 1$;

  $count \leftarrow count + 1$;

  if $count == thres$ **then**

    $thres \leftarrow thres - 1$;

    $count \leftarrow 1$;

    $r \leftarrow r + 1$;

  end

end

Note that, there is not one unique optimal download strategy. Algorithm 3 just gives one of these. Also, for a fixed n, the download strategy depends on the two parameters k and d. After we have the download strategy for the specific $[n,k,d,\tilde{\alpha}=d,\tilde{\beta}=1,\tilde{B}=k(d-\frac{k-1}{2})]$ PM MBR code, we follow the similar steps as described in the previous section on MSR codes. Each file is now of size $L = \tilde{B}n^m = k(d-\frac{k-1}{2})n^m$ and each block (in the previous section, we used "stripe" to be consistent with the general MSR encoding scheme, but in this section the word "block" is more appropriate with regard to the PM framework) of size $\tilde{B}$ is encoded using the PM MBR code. To generate the queries, the same Algorithm 1 applies but with a slight modification. Recall from the previous section that in the z^{th} round of Algorithm 1, a z-sum of the form $C^{(j_1)}_{i,count_{j_1}} + C^{(j_2)}_{i,count_{j_2}} + \ldots + C^{(j_z)}_{i,count_{j_z}}$ was downloaded where $C^{(j_l)}_{i,count_{j_l}}$ $(l \in [z])$ denoted the $\tilde{\alpha}$-dimensional coded vector stored in node i corresponding

to the block id (after permutation) $count_{j_l}$, '+' denoted element-wise vector sum over $\mathbb{F}_q^{\tilde{\alpha}}$ and $j_1, j_2, \ldots, j_z \in [m]$ are all distinct. For an instance $(count_{j_1}, count_{j_2}, \ldots, count_{j_z})$, the download was carried out from exactly k nodes such that the decodability could be ensured.

The modification is as follows. For an instance, instead of downloading all $\tilde{\alpha}$ symbols of the resulting vector sum from all the k nodes, we download a subset of symbols from each node. That subset is determined by the vectors $\{x_l : l \in [k]\}$. To be more specific, assume that according to Algorithm 1, an instance $(count_{j_1}, count_{j_2}, \ldots, count_{j_z})$ of a type $(j_1, j_2, \ldots, j_z)$ is to be downloaded from a subset $K = \{i_1, i_2, \ldots, i_k\} \subset [n]$. Instead of downloading all the $\tilde{\alpha}$ symbols of the z-sums from some node $i_a \in K$, we only download those symbols that are indexed by having a '1' in the corresponding vector x_a, for $a \in [k]$, as given by Algorithm 2. With this simple modification, we are able to reduce the download requirements without compromising in the decodability.

Claim 6.3.2: The rate achieved by this scheme is also $\frac{1-\frac{k}{n}}{1-(\frac{k}{n})^m}$.

Proof: The rate calculation trivially follows from that of Claim 6.2.1 as the numerator and denominator of Equation (6.14) now are multiples of $k(d - \frac{k-1}{2})$ instead of $k\tilde{\alpha}$. Hence this scheme achieves the same rate of $\frac{1-\frac{k}{n}}{1-(\frac{k}{n})^m}$. ■

Claim 6.3.3: The scheme is private in the sense of Chapter 4.

Proof: The only modification to the PIR scheme of MSR codes is that of following a download strategy instead of downloading all $\tilde{\alpha}$ symbols of a z-sum. Since this modification is applied irrespective of whether the z-sum contains the desired message index or not, the modification does not hurt the privacy arguments. ■

Remark : For odd values of k, all the vectors generated by Algorithm 3 have equal Hamming weights i.e., the number of '1's in each vector is the same. Hence, in this case the number of downloaded symbols across nodes is the same. But, in case of even values of k, we can observe that the Hamming weights of these vectors are different. So, it may seem that an asymmetrical download traffic arises in this case. But, due to the round-robin structure of the algorithm, the overall traffic still remains symmetrical.

Rate comparison with previous schemes : While the rate of $\frac{1-\frac{k}{n}}{1-(\frac{k}{n})^m}$ for MSR codes is proved to be optimal, no such claim can be made for MBR codes. In fact, we would like to point out that the scheme in [28] outperforms our PM MBR PIR scheme when the number of messages in the system becomes large. Figure 6.1 gives a comparison of the rates of the two schemes for fixed parameters of $[n = 6, k = 3, d = 4]$ and varying number of messages.

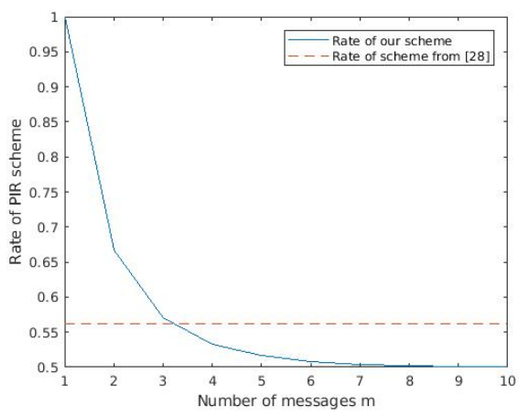


Figure 6.1: Rate Comparison of PIR schemes for PM MBR codes

Chapter 7

GPIR with T-privacy

7.1 A Scheme for GPIR with T-privacy

7.1.1 Preliminaries

In this section, we briefly describe the TPIR scheme construction of Raviv *et al.* [6] for 2-replication graph-based storage systems. The construction achieves the rate of $\frac{1}{n}$ for all applicable cases and preserves the privacy of the desired message against an adversary controlling at most T nodes as long as the T nodes do not induce a cycle. In the next section, we shall propose a technique that circumvents this apparent limitation of the scheme by compromising on the download rate.

We describe the scheme for $L = 1$. The characteristic of the field is taken to be 2. Upon requiring the message $W_{\hat{m}} \in \mathbb{F}_q$, the user prepares the $n \times m$ query matrix Q as follows:

$$Q = \text{diag}(\gamma) I_{\hat{m}} \text{diag}(\upsilon) \tag{7.1}$$

where $\gamma \in \mathbb{F}_q^{*n}$ and $\upsilon \in \mathbb{F}_q^{*m}$ are randomly chosen vectors, diag(.) is a square diagonal matrix containing the given vector in its main diagonal . The matrix $I_{\hat{m}}$ is obtained from the incidence matrix of the graph G by multiplying the upper $+1$ entry of the $\hat{m}^{th}$ column with a random field element $h \in \mathbb{F}_q^* \setminus \{1\}$. After getting back the answers, the user has access to $A = \text{diag}(\gamma) I_{\hat{m}} \text{diag}(\upsilon) \begin{bmatrix} W_1 & W_2 & \dots & W_m \end{bmatrix}^t$ from which it calculates $\mathbb{I} \cdot \text{diag}(\gamma)^{-1} A = \mathbb{I} \cdot I_{\hat{m}} \text{diag}(\upsilon) \begin{bmatrix} W_1 & W_2 & \dots & W_m \end{bmatrix}^t = (h-1)\upsilon_{\hat{m}} W_{\hat{m}}$ where $\mathbb{I}$ is the all ones vector.

For the complete proof of privacy, we refer the reader to [6]. Here we simply restate the two results that describe the graceful degradation of perfect privacy in the case when the adversary controlled nodes induce a cycle.

Lemma 2 from [6]: For any cycle $\mathcal{S} \subset [m]$, the matrix $Q_{\mathcal{S}}$ is full rank if and only if $\hat{m} \in \mathcal{S}$.

Corollary 1 from [6]: An adversary controlling a set $\mathcal{T}$ of servers can narrow down the set of possible values of $\hat{m}$ to

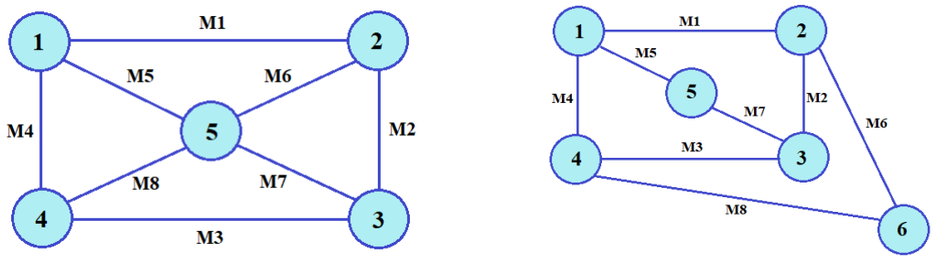
$$\mathcal{S}_{\mathcal{T}} = \left(\cap_{k=1}^{l} \mathcal{C}_k\right) \setminus \left(\cup_{k=1}^{l'} {\mathcal{C}'}_k\right) \tag{7.2}$$

where $\mathcal{C}_1, \dots, \mathcal{C}_l$ are all cycles in $G_{\mathcal{T}}$ that contain $\hat{m}$ and ${\mathcal{C}'}_1, \dots, {\mathcal{C}'}_{l'}$ are all cycles in $G_{\mathcal{T}}$ that do not contain $\hat{m}$.

Clearly, this scheme is restricted in the sense that perfect privacy is achieved only when T is strictly less than the girth of G. The following example demonstrates how we can get around this limitation by incurring some extra downloads.

Example 2: Consider the following graph in Figure 7.1a with $n = 5, m = 8$. The incidence matrix is given by

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



(a) Given graph G (b) Modified graph G'

Figure 7.1: An example demonstrating the technique of Section 3. (a) gives the original graph describing the DSS with girth 3. (b) gives the modified graph of girth 4 with one extra node

and for $\hat{m} = 1$ the query matrix is given by

$$Q = \begin{bmatrix} \gamma_1 & 0 & 0 & 0 & 0 \\ 0 & \gamma_2 & 0 & 0 & 0 \\ 0 & 0 & \gamma_3 & 0 & 0 \\ 0 & 0 & 0 & \gamma_4 & 0 \\ 0 & 0 & 0 & 0 & \gamma_5 \end{bmatrix} \begin{bmatrix} h & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \upsilon_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \upsilon_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \upsilon_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \upsilon_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \upsilon_5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \upsilon_6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \upsilon_7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \upsilon_8 \end{bmatrix}$$

This PIR scheme has a rate of $\frac{1}{5}$ and offers perfect privacy against $T = 2$. The reason it is not perfectly private against $T = 3$ can be deduced by observing the modified incidence matrix $I_{\hat{m}}$. It can be verified that if $\mathcal{T}$ induces a cycle with edges $\mathcal{S}$ then the submatrix of $I_{\hat{m}}$ row-restricted to $\mathcal{T}$ and column-restricted to $\mathcal{S}$ will have full rank if and only if the desired message belongs to $\mathcal{S}$. What we do to remedy this restriction is to virtually transform the original graph $G = (V, E)$ to another graph $G' = (V', E')$ with $|E'| = |E|$ and $|V'| > |V|$ such that there are no 3-cycles in G'. In reality, what we will be doing is to send more than one query to some nodes (in this example, we send two queries instead of one to node 5) such that the supports of these queries do not intersect. The following gives the modified query matrix:

$$Q' = \begin{bmatrix} \gamma_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \gamma_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma_6 \end{bmatrix} \begin{bmatrix} h & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \upsilon_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \upsilon_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \upsilon_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \upsilon_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \upsilon_5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \upsilon_6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \upsilon_7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \upsilon_8 \end{bmatrix}$$

An adversary observing node 5 has access to both the bottom two queries of Q'. So now the observable nodes in the virtual graph G' are $\mathcal{T}' = \mathcal{T} \cup \{6\}$ if previously $\mathcal{T}$ had contained node 5. We redefine the edges according to the modified incidence matrix. The set $\mathcal{S}$ remains unchanged and it can be easily verified that $I'_{\hat{m}}$ row-restricted to $\mathcal{T}'$ and column-restricted to $\mathcal{S}$ now has full rank irrespective of whether the desired message had belonged to the cycle in the original graph or not. The rate of this PIR scheme is $\frac{1}{6}$ (compared to the previous rate of $\frac{1}{5}$) and it is perfectly private against $T = 3$. The transformed graph G' does not contain any 3-cycles but does contain 4-cycles which means it is not perfectly private against $T = 4$. For that, we need to introduce more virtual nodes which further decrease the rate. Hence, there is a trade-off between the collusion resistance capability and the download rate of the PIR scheme.

7.1.2 An Algorithm to Increase the Collusion Resistance of Raviv *et al.*'s PIR Scheme

In this section, we give an algorithm that generalizes the idea discussed in Example 1. The simple idea is to introduce extra virtual nodes in the graph such that the resulting graph has

a girth strictly greater than the desired collusion resistance degree. Every introduction of a virtual node decreases the rate of the PIR scheme and the best possible rate can be found by finding the minimum possible number of virtual node introductions required to make the resulting graph have the desired girth. Note that the following algorithm, although effective in all scenarios, might not always be the optimal way of finding such a graph.

Algorithm 4: Transforming graph G

Result: Modifed graph $G' = (V', E')$
Input: Graph $G = (V, E)$, Collusion Degree T
Step 1: Initialize by setting $V' \equiv V$ and $E' \equiv E$;
Step 2: Identify the smallest length cycles $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_r$ of length σ
Step 3: If $\sigma > T$, STOP;
Step 4: Find a node x with the highest degree among all nodes of $\{\mathcal{C}_i : i \in [r]\}$;
Step 5: Identify all the edges of $\{\mathcal{C}_i : i \in [r]\}$ that contain this node
and split these edges into disjoint sets such that
no two edges of the same cycle belong to the same set;
Step 6: Remove node x from V';
Step 7: For every set of edges found in Step 5, do the following:
$\rightarrow$ Add a new vertex x' to V';
$\rightarrow$ Modify the edges previously incident on x to be incident on x';
Step 8: GO BACK to Step 2 and REPEAT;

Lemma 7.1.1: The PIR scheme described by Raviv *et al.* when applied on the modified graph G' results in perfect privacy against an adversary controlling any T nodes on the original graph G.

Proof: Intuitively, the proof follows because the modified graph G' has girth greater than T. Formally, we follow similar arguments given in the proof of Proposition 1 in [6]. It suffices to prove that $Pr(Q'_{\mathcal{T}'} = M)$ is fixed for any valid $M \in supp(Q'_{\mathcal{T}'})$ irrespective of the index of the desired message. For a given M, fix a node $i_0 \in \mathcal{T}'$ and fix the corresponding value of γ_{i_0}. Every nonzero entry in the i_0^{th} row of $Q'_{\mathcal{T}'}$ is equally likely among $(q-1)$ possibilities and independent due to the randomness in the corresponding multipliers from the α vector. Let j_0 be one column for which the other non-zero column entry also belongs to $Q'_{\mathcal{T}'}$. Conditioned on the value of α_{j_0}, this entry is equally likely among the $(q-1)$ possibilities due to the corresponding entry in the random γ vector. Since, the $|\mathcal{T}'|$ colluding nodes do not induce a cycle anymore in G', proceeding in such a Breadth-First Search fashion does not lead to any discrepancy. It easily follows that

$$Pr(Q'_{\mathcal{T}'} = M) = \frac{1}{(q-1)^{|supp(Q'_{\mathcal{T}'})|}} \tag{7.3}$$

and it is independent of the desired message index. ■

7.2 An Upper Bound for the PIR Rate

7.2.1 Main Result

A cycle cover of a graph G is a set of cycles which are subgraphs of G and contain all vertices of G. If the cycles of the cover have no vertices in common, the cover is called vertex-disjoint. We consider the set of graphs for which there exists a vertex disjoint cycle cover. Let $\mathcal{G}$ be the set of all such graphs. The following theorem gives an upper bound on the linear TPIR rate for a DSS described by any graph belonging to $\mathcal{G}$.

Theorem 2. *For a DSS described by a graph $G = (V, E) \in \mathcal{G}$ with $|V| = n$ nodes storing $|E| = m$ messages, the rate R of any linear TPIR scheme for arbitrary value of T is upper-bounded as follows:*

$$R \leq \begin{cases} \frac{T}{(T-1)n}, & \text{for } 2 \leq T < c_1 \\ \frac{T}{(T-1)n + \sum_{i=1}^{s} c_i}, & \text{for } c_s \leq T < c_{s+1}, 1 \leq s < r \\ \frac{1}{n}, & \text{for } c_r \leq T \leq n \end{cases} \tag{7.4}$$

where $c_1, c_2, \ldots, c_r$ are the cycle-lengths in a vertex disjoint cycle cover of the graph in non-decreasing order.

We shall delegate the proof of this theorem to Section 7.2.2. Here, we shall prove the lemmas that will be useful in proving the bound in the next section.

Lemma 7.2.1: In a valid Q matrix, which satisfies the PIR privacy and decodability requirements for some desired message against an adversary in control of at most T nodes, any set of ρ columns, for $\rho \leq (T-1)$, must have full column rank.

Proof. We prove this by induction on T. Note that the base case for $T = 2$ has already been proved in (4.27). We assume the property to be true for some $(T-1)$ nodes, i.e., when the adversary controls at most $(T-1)$ nodes, any set of $\rho \leq (T-2)$ columns of a valid Q matrix have full column rank. Now consider an adversary controlling any T nodes and a set $\mathcal{S}$ of $T-1$ columns of Q. Each column of Q represents an edge in the graph G describing the DSS. If the subgraph induced by the edges in $\mathcal{S}$ is connected, then the maximum number of nodes $\mathcal{T}$ in this induced subgraph is T. Since, the adversary is able to control at most T nodes, it will be able to observe both the $Q_{i_1,j}, Q_{i_2,j}$ matrices for all $j \in \mathcal{S}, \mathcal{R}_j = \{i_1, i_2\} \subset \mathcal{T}$. To preserve privacy, for every message $j \in \mathcal{S}$, the adversary should be able to find a linear transformation of $Q_{\mathcal{T},\mathcal{S}}$ to a matrix of the following form:

$$Q_{\mathcal{T},\mathcal{S}} \rightarrow \begin{bmatrix} 0 & \overline{Q}_j & 0 \\ X & 0 & Y \end{bmatrix} \tag{7.5}$$

where $\overline{Q}_j$ is full rank $L \times L$ and the number of columns in X and Y sum to $T-2$ (actually $(T-2)L$ over $\mathbb{F}_q$ to be exact). By induction assumption these submatrices are full column rank and hence $Q_{\mathcal{T}|\mathcal{S}}$ is full column rank.

Now consider the other scenario when the subgraph induced by the edges in $\mathcal{S}$ is not connected. Let $\mathcal{S}_1 \subset \mathcal{S}$ be a maximal subset whose edges do induce a connected subgraph. Then the Q matrix restriced to the columns of $\mathcal{S}$ has the following form (subject to column permutation)

$$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \tag{7.6}$$

where X has $|\mathcal{S}_1|$ columns and Y has $|\mathcal{S} \setminus \mathcal{S}_1|$ columns. By induction assumption both these matrices are full column rank and hence $Q_{\mathcal{S}}$ itself has full column rank. This completes the proof. □

Lemma 7.2.2: For a valid Q matrix, if the adversary is able to control any T nodes and the graph G describing the DSS has a cycle of length $c_l \leq T$, then the c_l columns indexed by the edges of the cycle must have full column rank.

Proof. The case $c_l < T$ follows from Lemma 7.2.1. So we need to prove only for $c_l = T$. Let $\mathcal{S}$ denote the edges that constitute the cycle. The subgraph induced by $\mathcal{S}$ has exactly T nodes. Let this set of nodes be $\mathcal{T}$. An adversary observing these T nodes has access to both matrices $Q_{i_1,j}, Q_{i_2,j}$ for all $j \in \mathcal{S}, \mathcal{R}_j = \{i_1, i_2\} \subset \mathcal{T}$. By similar logic of the previous lemma, for any $j \in \mathcal{S}$, the adversary should be able to find a linear transformation of $Q_{\mathcal{T},\mathcal{S}}$ to a matrix of the form of (7.5) where X and Y together have $T-1$ columns and are full column rank by Lemma 7.2.1. Hence, the T columns of Q_{obs} have full column rank. □

7.2.2 Proof of Theorem 2

In this section, we prove Theorem 2. For that we introduce some new notation. Let $G = (V, E)$ be the graph, with $|V| = n, |E| = m$, which describes the DSS and has a vertex disjoint cycle cover. The k^{th} cycle of the cycle cover is denoted by $\mathcal{C}_k = (V_k, E_k)$ and there are r cycles for some $r \in \mathbb{Z}^+$. The length of the k^{th} cycle is $|V_k| = c_k$ and, without loss of generality, we assume $c_1 \leq c_2 \leq \ldots \leq c_r$. Note that, $\sum_{k=1}^{r} c_r = n \leq m$. We will consider three separate cases.

Case I : When $T < c_1$, consider the cycle $\mathcal{C}_k$ of length $c_k, k \in [r]$. Without loss of generality, index the c_k nodes by the set $[c_k]$ and the edge between node i and node $(i \mod c_k) + 1$

as edge i for $i \in [c_k]$. Let Q' be the submatrix of Q restricted to these nodes and edges. Notice that due to the circular structure, any successive $T-1$ columns of Q' indexed by $\{j, j+1, \ldots, (j+T-2 \mod c_k)+1\}$ have nonzero entries only at the rows indexed by $\{j, j+1, \ldots, (j+T-1 \mod c_k)+1\}$. Since every such set of $T-1$ columns of Q' have full column rank (over $\mathbb{F}_q$) by Lemma 7.2.1, we have the following equation:

$$\sum_{p=j}^{(j+T-1 \mod c_k)+1} l_p \geq (T-1)L \quad \forall\, j \in [c_k] \tag{7.7}$$

Summing the c_k equations we have

$$\begin{aligned} T\sum_{p=1}^{c_k} l_p &\geq (T-1)c_k L \\ \implies \sum_{p\in\mathcal{V}_k} l_p &\geq \frac{(T-1)c_k L}{T} \end{aligned} \tag{7.8}$$

Finally summing up over all $k \in [r]$

$$\sum_{k=1}^{r}\sum_{p\in\mathcal{V}_k} l_p \geq \frac{(T-1)L}{T}\ \sum_{k=1}^{r} c_k \tag{7.9}$$

As each node belongs to exactly one cycle $\mathcal{C}_k$, the LHS is just a sum over all nodes of the graph. So,

$$\begin{aligned} \sum_{p=1}^{n} l_p &\geq \frac{(T-1)nL}{T} \\ \implies \frac{L}{\sum_{p=1}^{n} l_p} &\leq \frac{T}{(T-1)n} \end{aligned} \tag{7.10}$$

Case II : When $c_s \leq T < c_{s+1}$ for some $s < r$, for every $k \leq s$, we apply Lemma 7.2.2 to the nodes of the k^{th} cycle. We have

$$\sum_{p\in\mathcal{V}_k} l_p \geq c_k L \quad \forall\, k \leq s \tag{7.11}$$

For all $k > s$, we apply Lemma 7.2.1 as before to get

$$\sum_{p\in\mathcal{V}_k} l_p \geq \frac{(T-1)c_k L}{T} \quad \forall\, k > s \tag{7.12}$$

Summing up we get

$$\begin{aligned} \sum_{p=1}^{n} l_p &\geq \sum_{k\leq s} c_k L + \sum_{k>s} \frac{(T-1)c_k L}{T} \\ &= \frac{(T-1)n + \sum_{k=1}^{s} c_k}{T} L \\ \implies \frac{L}{\sum_{p=1}^{n} l_p} &\leq \frac{T}{(T-1)n + \sum_{k=1}^{s} c_k} \end{aligned} \tag{7.13}$$

Case III : Finally, for the case $T \geq c_r$, we simply apply Lemma 7.2.2 for all the cycles to get

$$\sum_{k=1}^{r}\sum_{p\in\mathcal{V}_k} l_p \geq \sum_{k=1}^{r} c_k L \tag{7.14}$$

As before, the LHS is just a sum over all nodes of the graph and we get

$$\frac{L}{\sum_{p=1}^{n} l_p} \leq \frac{1}{n} \tag{7.15}$$

Remark: Our system model for GPIR described in Chapter 4.2 assumes that for any two $j_1, j_2 \in [m]$ and $j_1 \neq j_2$, $\mathcal{R}_{j_1} \neq \mathcal{R}_{j_2}$ which implies no two nodes share more than one message. This assumption is directly inherited from the system model of [6] which is crucial for the correctness of their proposed PIR scheme. However, we draw the reader's attention to the fact that the correctness of the two lemmas in Section 7.2.1 and the proof of Theorem 2 in Section 7.2.2 do not require this assumption. Hence, the upper bound of Theorem 2 is applicable to not just graphs but also multigraphs that have a vertex disjoint cycle cover.

Chapter 8

Conclusion

In this work, we have tried to explore some open areas of private information retrieval. The work on PIR for MDS codes [8] and the gradual tendency of distributed storage towards the requirement of node repair naturally raised the question of PIR for regenrating codes. Our contributions in Chapter 5 along with recent similar findings of [27] and [28] partly answer this question. The schemes of Chapter 5 although sub-optimal, have the advantage of having download rates that are independent of the number of messages present in the database. Also, these schemes are capable of working on very small message sizes. The question of optimal download rate, i.e., the PIR capacity for MSR codes directly follows from the similar analysis of MDS codes as pointed out in [29] and discussed in detail in Chapter 6. In Section 6.3, we showed that the idea behind the capacity achieving PIR scheme for the class of MSR codes can also be extended to the specific class of PM MBR codes with a slight modification to the former. Hence, it is possible to achieve the same PIR rate for the MBR codes as is optimally possible for the MSR codes. But, as explained at the end of Section 6.3, this can not be the maximum possible rate for MBR codes as PIR schemes giving higher rates have already been discovered using the PM framework of the codes. An explicit capacity expression for the general class of MBR codes and achievability schemes for this capacity seem pertinent questions that need to be answered in this regard.

The work on PIR for graph-based replication systems is motivated by, as pointed out in [6], the contradicting requirements of the simplicity offered by replication based storage and the need for reduction in storage overhead. In such scenarios, partial replication of messages can provide a sweet spot for the system designer. In Section 7.1.1, we discussed the TPIR scheme proposed in [6] which is somewhat limited by the necessity of the girth of the graph being smaller than the collusion degree T. We showed in Section 7.1.2, that application of this scheme is also possible when this requirement is not met provided we are allowed to tolerate some extra download. Our analysis is based on the idea of transforming a given graph to a new graph with more number of nodes such that it does not contain any cycle of length T or smaller. Although, we provide an algorithm to successfully perform this transformation, we do not claim optimality in terms of number of nodes added. Finding out such optimal algorithms, which can be an interesting problem of Graph Theory in itself, might be of interest since it directly relates to the download rate. Finally, in Section 7.2, we gave an upper bound on the linear TPIR rate for a large family of graphs. The bound, interestingly, is only a small multiplicative constant away from the rate obtained by Raviv *et al.*'s scheme in some cases (namely Case I in the proof of Theorem 2) and the constant decreases with increase in the value of T. In other cases, Raviv *et al.*'s scheme is not directly applicable and the use of our algorithm further decreases the rate from $\frac{1}{n}$. Bridging these gaps between the achievable rate and the upper bound might be interesting future directions for the GPIR problem.

Bibliography

[1] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1945–1956, March 2018.

[2] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, vol. 45, no. 6, pp. 965–981, November 1998. [Online]. Available: http://doi.acm.org/10.1145/293347.293350

[3] H. Sun and S. A. Jafar, "The capacity of private information retrieval," *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4075–4088, July 2017.

[4] S. A. Jafar, "Blind interference alignment," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 3, pp. 216–227, June 2012.

[5] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[6] N. Raviv and I. Tamo, "Private information retrieval in graph based replication systems," in *2018 IEEE International Symposium on Information Theory (ISIT)*, June 2018, pp. 1739–1743.

[7] H. Sun and S. A. Jafar, "The capacity of robust private information retrieval with colluding databases," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2361–2370, April 2018.

[8] K. A. Banawan and S. Ulukus, "Private information retrieval from byzantine and colluding databases," *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1091–1098, 2017.

[9] R. Tajeddine, O. W. Gnilke, D. A. Karpuk, R. Freij, and C. Hollanti, "Robust private information retrieval from coded systems with byzantine and colluding servers," *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2451–2455, 2018.

[10] H. Sun and S. A. Jafar, "The capacity of symmetric private information retrieval," in *2016 IEEE Globecom Workshops (GC Wkshps)*, Dec 2016, pp. 1–5.

[11] Q. Wang and M. Skoglund, "Symmetric private information retrieval for MDS coded distributed storage," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[12] ——, "Secure symmetric private information retrieval from colluding databases with adversaries," *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1083–1090, 2017.

[13] S. Kadhe, B. Garcia, A. Heidarzadeh, S. E. Rouayheb, and A. Sprintson, "Private information retrieval with side information: The single server case," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct 2017, pp. 1099–1106.

[14] Z. Chen, Z. Wang, and S. Jafar, "The capacity of private information retrieval with private side information," 2017. [Online]. Available: http://arxiv.org/abs/1709.03022

[15] Y. Wei, K. Banawan, and S. Ulukus, "Cache-aided private information retrieval with partially known uncoded prefetching: Fundamental limits," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1126–1139, June 2018.

[16] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "Capacity of single-server single-message private information retrieval with coded side information," 2018. [Online]. Available: http://arxiv.org/abs/1806.00661

[17] A. Heidarzadeh, B. Garcia, S. Kadhe, S. Y. E. Rouayheb, and A. Sprintson, "On the capacity of single-server multi-message private information retrieval with side information," 2018. [Online]. Available: http://arxiv.org/abs/1807.09908

[18] Y. Wei and S. Ulukus, "The capacity of private information retrieval with private side information under storage constraints," 2018. [Online]. Available: http://arxiv.org/abs/1806.01253

[19] K. A. Banawan and S. Ulukus, "Asymmetry hurts: Private information retrieval under asymmetric traffic constraints," 2018. [Online]. Available: http://arxiv.org/abs/1801.03079

[20] R. Bitar and S. E. Rouayheb, "Staircase-PIR: Universally robust private information retrieval," 2018. [Online]. Available: http://arxiv.org/abs/1806.08825

[21] Y. Wu and A. G. Dimakis, "Reducing repair traffic for erasure coding-based storage via interference alignment," in *2009 IEEE International Symposium on Information Theory*, June 2009, pp. 2276–2280.

[22] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit construction of optimal exact regenerating codes for distributed storage," *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1243–1249, 2009.

[23] Y. Wu, "A construction of systematic MDS codes with minimum repair bandwidth," *IEEE Transactions on Information Theory*, vol. 57, no. 6, pp. 3738–3741, June 2011.

[24] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, Aug 2011.

[25] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Distributed storage codes with repair-by-transfer and nonachievability of interior points on the storage-bandwidth tradeoff," *IEEE Transactions on Information Theory*, vol. 58, no. 3, pp. 1837–1852, 2012.

[26] N. B. Shah, K. V. Rashmi, and K. Ramchandran, "One extra bit of download ensures perfectly private information retrieval," in *2014 IEEE International Symposium on Information Theory*, June 2014, pp. 856–860.

[27] C. Dorkson and S. Ng, "Private information retrieval using product-matrix minimum storage regenerating codes," 2018. [Online]. Available: http://arxiv.org/abs/1805.07190

[28] J. Lavauzelle, R. Tajeddine, R. Freij-Hollanti, and C. Hollanti, "Private information retrieval schemes with regenerating codes," 2018. [Online]. Available: http://arxiv.org/abs/1811.02898

[29] S. Kumar, H. Lin, E. Rosnes, and A. G. i Amat, "Achieving private information retrieval capacity in distributed storage using an arbitrary linear code," *CoRR*, vol. abs/1712.03898, 2017. [Online]. Available: http://arxiv.org/abs/1712.03898

[30] M. Abdul-Wahid, F. Almoualem, D. Kumar, and R. Tandon, "Private information retrieval from storage constrained databases - coded caching meets PIR," *CoRR*, vol. abs/1711.05244, 2017. [Online]. Available: http://arxiv.org/abs/1711.05244

[31] Z. Jia and S. A. Jafar, "On the asymptotic capacity of x-secure t-private information retrieval with graph based replicated storage," *CoRR*, vol. abs/1904.05906, 2019. [Online]. Available: http://arxiv.org/abs/1904.05906

[32] C. Dorkson and S. Ng, "Multi-message private information retrieval using product-matrix MSR and MBR codes," 2018. [Online]. Available: http://arxiv.org/abs/1808.02023