Quantum Private Information Retrieval

Klaas Ruben Brokkelkamp

July 19, 2013

Bachelor’s thesis
Mathematics and Computer Science
Supervisor: Prof. Ronald de Wolf

Faculty of Science (FNWI)
University of Amsterdam
Abstract

Locally Decodable Codes are error-correcting codes which make it possible to recover entries from the original string by only looking at a certain number of entries from the codeword.

In Private Information Retrieval a user wants to get data from a database without giving any information about the data she is interested in. The goal is to minimise the amount of communication which takes place between the user and the database. For a database on a single server the trivial protocol of sending the whole database is optimal [9], but when the database is replicated over a number of non-communicating servers much better schemes exist.

There is a connection between Locally Decodable Codes and Private Information Retrieval in the sense that they can be converted into each other. The best known Private Information Retrieval Schemes are derived from Locally Decodable Codes.

Allowing quantum bits as communication yields better Private Information Retrieval Schemes.
Acknowledgements

I am very grateful to my supervisor Ronald de Wolf, for suggesting the subject and his numerous comments and helpful remarks. Also, his course Quantum Computing was informative and inspiring.

The main idea in Theorem 52 came from Ronald.
# Contents

1 Introduction .................................................. 1  
1.1 Locally Decodable Codes .................................... 1  
1.2 Private Information Retrieval ................................ 2  
1.3 Quantum ..................................................... 3  
1.4 Notation ...................................................... 5  

2 Locally Decodable Codes ....................................... 7  
2.1 Definition and First Example .................................. 7  
2.2 Smooth Codes ................................................ 9  
2.3 Efremenko’s Locally Decodable Codes .......................... 10  
   2.3.1 Matching Sets of Vectors and S-decoding Polynomials ... 11  
   2.3.2 The Code ................................................. 12  
   2.3.3 Binary Case .............................................. 15  
2.4 Matching Vector Codes ....................................... 17  

3 Classical Private Information Retrieval .......................... 21  
3.1 Definition and First Example .................................. 21  
3.2 Better Schemes ............................................... 22  
3.3 Connection between Smooth Codes and PIR ...................... 26  

4 Quantum Private Information Retrieval .......................... 29  
4.1 Preliminaries ................................................. 29  
   4.1.1 Qubits and Quantum States .............................. 29  
   4.1.2 Measurements ........................................... 30  
   4.1.3 Evolution ................................................. 31  
   4.1.4 Queries .................................................. 32  
4.2 Locally Decodable Quantum Codes and Quantum Private Information Retrieval .......................................................... 32  
4.3 Classical to Quantum ........................................... 34  
4.4 Multiple-Bit Answers ......................................... 37  
4.5 Quantum Private Information Retrieval Schemes .............. 40  

5 Single-Server Interactive Quantum Private Information Retrieval 41  
5.1 Multi-round Single-Server Quantum Private Information Retrieval ................ 41  
5.2 Generalizing and improving Le Gall’s scheme ................... 42  
5.3 Insecurity when the server does not follow the protocol ........ 46  

6 Populaire Samenvatting ........................................... 49
1 Introduction

An error-correcting code encodes an \( n \)-bit string \( x \) into some \( N \)-bit codeword \( C(x) \) such that the original string \( x \) can be recovered from \( C(x) \) even though part of \( C(x) \) is corrupted or maybe destroyed. This can be very useful when sending information over a channel with a lot of noise. Typically, when the codeword is longer, more redundancy is possible and therefore a larger part of the codeword may be corrupted before it is impossible to retrieve the original string. The trick is to find a good tradeoff between the amount of redundancy and the length of the codeword. What counts as a good tradeoff depends on the applications that it is going to be used for.

Error-correcting codes are closely related to the field of information theory, which is concerned with the quantification of information and how efficiently it can be transmitted. Already in the late 1940s Shannon, the father of information theory, gave a maximum on the number of symbols that can be sent through a channel with random noise [28]. In the early 1950s Hamming presented important error-correcting codes [18] which are now known as Hamming codes. Since those two highly influential publications the field of error-correcting code has grown into a broad field of research [24] and numerous codes have been designed. There are for example the Reed–Solomon codes [27]. A variation on these codes is widely used for encoding information on CDs, DVDs and Blu-ray Discs to protect them against noise. Without error-correcting codes the concept of compact discs would have been completely useless due to unavoidable noise from small scratches and the decoding process.

1.1 Locally Decodable Codes

Traditional error-correcting codes require the decoding procedure to read the entire codeword \( C(x) \) even when one is only interested in a small part of the original string \( x \). You could for example think of a hard disk which is encoded to improve the resistance against noise. If the hard disk is encoded into one codeword and we only need a single file, it is not very efficient to decode the entire hard disk and then look for the file we desire. A possibility is to divide the hard disk into parts and encode each part separately. A disadvantage of this procedure is that if the part of the hard disk where the codeword of the file we need is stored, is
corrupted or destroyed, we are not able to retrieve the file and it is gone forever. We would like an error-correcting code which protects the data against worst-case noise and is able to efficiently recover parts of the original string. A class of error-correcting codes which enables us to efficiently recover parts of the original string while protecting us against worst-case noise is the class of \textit{Locally Decodable Codes}. They encode a string $x$ into a codeword $C(x)$ and allow us to retrieve an entry from the original string by only reading a few of entries from the codeword. Locally Decodable Codes were already used in literature about Probabilistically Checkable Proofs in the 1990s \cite{2}. The first formal definition was given by Katz and Trevisan \cite{20} in 2000. Since then, Locally Decodable Codes has grown into a rather broad field of research.

1.2 Private Information Retrieval

In \cite{9} Chor et al. proposed the idea of \textit{Private Information Retrieval}. In Private Information Retrieval a user wants some information from a database, but she does not want to give any information at all about the thing she is interested in. A situation where this is useful, is, for example, the stock market. Suppose some large trader is interested in buying a stock and she wants to know the price. If someone found out the large trader is interested in some stock, he would buy the stock, which could drive up the price. Thus the large trader likes to keep to herself which information she is interested in. Another situation is when someone likes to access her medical records. For some persons it is not desirable if someone found out they are not completely healthy.

The scarce resource in this setting is communication, where computing power is assumed to be unlimited, although there are variants which also consider the amount of computing power \cite{30}. If the database is stored on a single server you cannot do better than using the trivial protocol of requesting the whole database \cite{9}. In their article Chor et al. considered the situation where the database is replicated over $k$ non-communicating servers. They presented a scheme for $k$ servers with communication complexity $O(n^{\frac{1}{k}})$ for $n$ being the number of bits in the database. For 2 servers they presented a protocol with communication complexity $O(n^{\frac{1}{2}})$ which still has not been improved. There have been a number of improvements for $k \geq 3$ servers starting with Ambainis \cite{1} who presented a scheme with communication complexity $O\left(n^{\frac{1}{k}-\frac{1}{2}}\right)$. Beimel et al. \cite{4} improved this to $n^{O\left(\frac{\log \log k}{k \log k}\right)}$ and the currently best known upper bounds of $n^{O\left(\frac{\sqrt{\log \log n}}{\log n}\right)^{r-1}}$ for $k = 2^r$ servers and $n^{O\left(\frac{\sqrt{\log \log n}}{\log n}\right)}$ for 3 servers are achieved by Efremenko \cite{12}. We will present his construction in this thesis.

Although the assumption of a number non-communicating servers is a strong
one, there are situations in which this applies. We can, for example, reconsider
the example of the stock market. There are a lot of places, for example banks,
which have the latest information on stock prices. They do not share any data
about their customers, since they are competitors.

There are many variations on Private Information Retrieval. In [22] computa-
tional Private Information Retrieval is introduced. In traditional Private Informa-
tion Retrieval, we consider information theoretic privacy whereas computational
Private Information Retrieval only guarantees computationally bounded privacy.
The scheme from [22] gives a 1-server scheme with communication complexity
$O(n^c)$ for every $c > 0$, assuming the hardness of deciding quadratic residuosity.
This was improved in [8] where a scheme with communication complexity polylog-
arithmic in $n$ was presented.

Traditional Private Information Retrieval requires there is no communication at
all between the different servers. The setting where this assumption is dropped
and coalitions up to $t > 1$ servers are allowed was already addressed in [9] and the
currently best known schemes are presented in [32].

Another variation is that of Symmetric Private Information Retrieval, which
gives the extra restriction that the user may only learn the bit in which she is
interested in and nothing else. This notion was introduced by Gertner et al. [15]
and in their article they showed it is impossible in the traditional model of Private
Information Retrieval. To overcome this problem they presented schemes in which
the user and database share a random string.

Locally Decodable Codes and Private Information Retrieval are closely related.
Intuitively one can think of each query to a codeword as a query to a different
server which acts as if the database is encoded using the Locally Decodable Code.
The currently best known Private Information Retrieval schemes from Efremenko
[12] are based on Locally Decodable Codes.

1.3 Quantum

Another variant of Private Information Retrieval is one where we allow quantum
bits as communication. The computers found in every household nowadays are
based on classical physics. For a bit this implies it is either 0 or 1. From quantum
physics we know that if we look at particles at microscopic scales they behave
differently than described in classical physics. A quantum bit (or in short qubit)
can be in a superposition of 0 and 1. Intuitively one can think the qubit is both
0 and 1 at the same time. In the classical world we have two states, but in
the quantum world we have uncountably many states at our disposal. A minor
misfortune is that when we “look” (measure) at such a state it collapses to one of
two basis states and all the information which was contained in the superposition
is gone. It has not been proven that quantum computers are faster than classical computers. Still for some problems the best known quantum algorithms are much faster than the best known classical algorithms. The most famous example is prime factorisation. Using Shor’s algorithm [29] this can be done in polynomial time on a quantum computer while the best known algorithm for a classical computer needs exponential time. In the setting of Private Information Retrieval we will see that in some cases using quantum bits will result in a much lower communication complexity.
1.4 Notation

In this thesis we will use the following notation without further reference.

- \([n] = \{1, 2, \ldots, n\}\).
- \(\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i\), with \(x, y\) vectors of length \(n\). In Sections 2.3 and 2.4 the inner products will be calculated modulo \(m\).
- \(x \cdot y = \sum_{i=1}^{n} x_i y_i\) modulo 2, for bitstrings \(x, y \in \{0, 1\}^n\).
- \(x \oplus y = (x + y) \mod 2\) for \(x, y \in \{0, 1\}\), i.e., XOR of \(x\) and \(y\). If \(x\) and \(y\) are bitstrings, the XOR is taken entry-wise.
- \(d(x, y)\) is the Hamming distance between \(x\) and \(y\), i.e., the number of entries where they differ.
- For a bitstring \(a \in \{0, 1\}^n\), \(|a|\) is the Hamming weight, i.e., the number of entries that are 1.
- \(\delta_{ij}\) is the Kronecker delta, i.e., \(\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}\).
- \(\mathbb{F}_q\) is the finite field consisting of \(q\) elements.
- \(\mathbb{Z}_m\) is the cyclic group consisting of \(m\) elements.
- \(e_i\) is the zero string or zero vector, depending on the context, with a one at position \(i\).
- \(0^t\) or \(1^t\) is used to indicate the string of \(t\) 0s or 1s.
- For a subset \(Q \subseteq S = \{|S|\}\), the indicator string is a string \(q \in \{0, 1\}^{|S|}\) such that \(q_i = 1 \iff i \in Q\).
- \(\exp(x) = 2^x\).
- Logarithms are taken base 2.
2 Locally Decodable Codes

2.1 Definition and First Example

Informally, a \((q, \delta, \epsilon)\)-Locally Decodable Code is an error-correcting code such that an entry from the original string (often referred to as message) can be recovered with probability at least \(\frac{1}{2} + \epsilon\) by looking at \(q\) entries from the codeword even though at most a \(\delta\)-fraction of the codeword is corrupted. Here, an entry is corrupted when it has a different value then the corresponding entry in the correct codeword. The corrupted fraction can be anywhere in the codeword, in our model we want to protect ourselves against the worst-case. More formally:

**Definition 1** (Locally Decodable Code (LDC)). A map \(C : \Gamma^n \rightarrow \Sigma^N\) is called a \((q, \delta, \epsilon)\)-Locally Decodable Code if there exists a classical randomised decoding algorithm \(A\) such that

- \(A\) makes at most \(q\) non-adaptive queries to a string \(y \in \Sigma^N\).
- For all \(x \in \Gamma^n, y \in \Sigma^N\) with Hamming distance \(d(C(x), y) \leq \delta N\) it holds that

\[
P[A^y(i) = x_i] \geq \frac{1}{2} + \epsilon \text{ for all } i \in [n],
\]

where the probability is taken over the randomness of \(A\).

The decoding algorithm \(A\) is called a \((q, \delta, \epsilon)\)-local decoder.

In \(A^y(i)\) we write \(y\) in superscript to denote the difference between input \(i\) and \(y\). For input \(i\) we got full access, while for \(y\) we only have query access.

In our definition we only consider non-adaptive queries. This means that every query is based solely on \(i\). They are all generated in advance and can be executed in parallel. If we consider adaptive queries, the second query may also depend on the answer to the first query and the third query may depend on the answers to the first and the second query and so on. In [20] it is shown that one can convert an adaptive \((q, \delta, \epsilon)\)-local decoder into a non-adaptive \((\frac{|\Sigma|^q - 1}{|\Sigma|-1}, \delta, \epsilon)\)-local decoder and also to a non-adaptive \((q, \delta, \frac{\epsilon}{|\Sigma|^q})\)-local decoder. For a small number of queries and/or a small alphabet, on which we mainly focus, the loss in the parameters is not very significant and therefore we will focus only on non-adaptive queries.
The sets $\Gamma$ and $\Sigma$ are usually finite fields and we are most interested in the binary case, i.e., $\Gamma = \Sigma = \mathbb{F}_2$. Then we can identify $\mathbb{F}_2$ with \{0, 1\} and think in the setting of bits. In this chapter we describe the Locally Decodable Codes from Efremenko, which are first defined for finite fields and later converted into binary LDCs. But we will also see codes where $\Sigma = \{0, 1\}$ for some positive integer $a$, in which case the “answers” to the queries consist not only of one bit, but of blocks of $a$ bits.

It depends on the setting which of the parameters are more important than others. For example in data storage and transmission, one would like to have a large $\delta$ and small length of the codeword whereas the number of queries is less important, as long as it is still much smaller than the size of the encoded string. Also the value of $\epsilon$ does not need to be extremely close to $\frac{1}{2}$, since we can just run the procedure several times, to get a larger successrate. In cryptography (for example Private Information Retrieval) a small number of queries and small codewords is more desirable.

The simplest example of an LDC is the Hadamard code.

**Example 2** (Hadamard Code). For $q = 2$ the first example is the Hadamard code. Given some $x \in \{0, 1\}^n$ the codeword of length $2^n$ consists of the concatenation of bits $x \cdot z \mod 2$, for every $z \in \{0, 1\}^n$. The decoding procedure for bit $x_i$ and some corrupted codeword $w$ consists of picking a random $z \in \{0, 1\}^n$ and querying the bits in $w$ at the positions of $x \cdot z \mod 2$ and $x \cdot z \oplus e_i \mod 2$ in the non-corrupted codeword. If neither one of the bits is corrupted we will find the desired bit

$$w_z \oplus w_{z \oplus e_i} = C(x)_z \oplus C(x)_{z \oplus e_i} = (x \cdot z) \oplus (x \cdot (z \oplus e_i)) = x \cdot e_i = x_i.$$ 

The subscript $z$ is used here as an index where $w_z$ points to the entry at the position of $x \cdot z \mod 2$ in the non-corrupted codeword.

If at most a $\delta$-fraction of the corrupted codeword $w$ is corrupted, by the union bound, the probability that we query a corrupted index is at most

$$\Pr[w_z \text{ or } w_{z \oplus e_i} \text{ is corrupted}] \leq \Pr[w_z \text{ is corrupted}] + \Pr[w_{z \oplus e_i} \text{ is corrupted}] \leq \delta + \delta = 2\delta.$$

Thus the Hadamard code is a $(2, \delta, \frac{1}{2} - 2\delta)$-Locally Decodable Code $C : \{0, 1\}^n \to \{0, 1\}^{2^n}$ for $\delta \in [0, \frac{1}{4}]$.

If we use the Hadamard code and 10 percent of the codeword is corrupted, the probability of recovering a bit is 0.8. One can think that this is not very good, since if we do not use any encoding and 10 percent of the entries are corrupted, we have a probability of 0.9 of reading the correct bit. The subtle difference here is
that in the latter case querying a corrupted index will always give the wrong value, while using the Hadamard code, every bit can be recovered with probability 0.8, no matter which 10 percent of the codeword is corrupted. As mentioned before, we protect ourselves against the worst-case.

2.2 Smooth Codes

There is a close relation between Locally Decodable Codes and so-called Smooth Codes. Smooth Codes are codes where no entry is queried with a probability higher than a certain threshold. Later we will establish a connection between Smooth Codes and Private Information Retrieval.

**Definition 3 (Smooth Code).** A map $C : \Gamma^n \to \Sigma^N$ is called a $(q, c, \epsilon)$-Smooth Code if there exists a probabilistic algorithm $A$ such that for any $x \in \Gamma^n$:

- $A$ makes at most $q$ queries to the codeword $C(x)$.
- For all $j \in [N]$, $C(x)_j$ is queried with probability at most $\frac{c}{N}$.
- For all $i \in [n]$, $\Pr[A^{C(x)}(i) = x_i] \geq \frac{1}{2} + \epsilon$,

where the probability is taken over the randomness of $A$.

The decoding algorithm is called a $(q, c, \epsilon)$-smooth decoder. If $c = q$ we call the decoder perfectly smooth and for $\epsilon = \frac{1}{2}$ we have perfect recovery.

Note that the decoding algorithm does not need to work for corrupted codewords.

To convert a Locally Decodable Code into a Smooth Code we have the following theorem due to [20].

**Theorem 4.** Let $C : \Gamma^n \to \Sigma^N$ be a $(q, \delta, \epsilon)$-Locally Decodable Code. Then $C$ is also a $(q, \frac{q}{\delta}, \epsilon)$-Smooth Code.

**Proof.** Let $L$ be the $(q, \delta, \epsilon)$-local decoder for $C$ and let $x \in \Gamma^n$. For $i \in [n]$ let $Q_i$ be the set of $j \in [N]$ such that $\Pr[L^{C(x)}(i) \text{ queries } j] > \frac{q}{\delta N}$. Note that $|Q_i| < \delta N$, because $L$ queries no more than $q$ indices. Now, we are going to define the smooth decoder $S$. The smooth decoder $S$ will pick the same indices as $L$ except if an index from $Q_i$ is picked we ignore this entry and simply return 0. To see that this is a smooth decoder we define the string $y \in \Sigma^N$ as follows

$$y_k = \begin{cases} 
0 & \text{if } k \in Q_i \\
C(x)_k & \text{if } k \notin Q_i 
\end{cases}$$
Because $|Q| < \delta N$ the string $y$ has Hamming distance $d(C(x), y) < \delta N$. We see that $S^{C(x)} = L^y$ and so we get

$$\mathbb{P}[S^{C(x)}(i) = x_i] = \mathbb{P}[L^y(i) = x_i] \geq \frac{1}{2} + \epsilon$$

Since the indices which are queried with probability greater than $\frac{\epsilon}{2\delta N}$ are ignored, we can conclude that using the smooth decoder $S$ the code $C$ is also a $(q, \frac{\epsilon}{2}, \epsilon)$-Smooth Code.

We can also convert a Smooth Code into a Locally Decodable Code.

**Theorem 5.** Let $C : \Gamma^n \rightarrow \Sigma^N$ be a $(q, c, \epsilon)$-Smooth Code. Then $C$ is also a $(q, \delta, \epsilon - c\delta)$-Locally Decodable Code for any $\delta$.

**Proof.** Let $S$ be the smooth decoder. The probability that an index is queried is at most $\frac{c}{N}$. So, by the union bound, the probability that there is a corrupted index among the queried indices, if at most $\delta N$ of the indices is corrupted, is

$$\mathbb{P}[S \text{ queries a corrupted index}] \leq (\delta N) \frac{c}{N} = c\delta.$$ 

Thus, if $y$ is a codeword of $x$ which is corrupted up to $\delta N$ places, the probability of recovering $x_i$ is

$$\mathbb{P}[S^y(i) = x_i] \geq \mathbb{P}[S^{C(x)}(i) = x_i] - c\delta \geq \frac{1}{2} + \epsilon - c\delta$$

The $(q, c, \epsilon)$-smooth decoder $S$ is also a $(q, \delta, \epsilon - c\delta)$-local decoder for any $\delta$. $\square$

### 2.3 Efremenko’s Locally Decodable Codes

In this section we describe the currently best known construction for Locally Decodable Codes. They are constructed by Efremenko in [12] and are inspired by the work done by Yekhanin in [33]. In his work Yekhanin showed, using the assumption that there are infinitely many Mersenne prime numbers$^1$, the existence of 3-query LDCs with subexponential codeword length $\exp\left(\exp\left(\mathcal{O}\left(\frac{\log n}{\log \log n}\right)\right)\right)$. The assumption of infinitely many prime numbers is not needed in the construction from Efremenko. The length of the codeword he achieves for 3 queries is

$$\exp\left(\exp\left(\mathcal{O}(\sqrt{\log n \log \log n})\right)\right),$$

---

$^1$A Mersenne prime number is a prime which can be written as $2^t - 1$ for some integer $t$.  

10
and for $2^r$ queries a shorter codeword length is achieved, namely
\[ \exp \left( \exp \left( \mathcal{O} \left( \sqrt{\log n \log \log n}^{r-1} \right) \right) \right). \]

In this section we will present the LDCs in the same way Efremenko presented them in [12].

2.3.1 Matching Sets of Vectors and S-decoding Polynomials

All the inner products in the following two sections are done modulo $m$. To construct the LDCs we need a family of vectors with some special properties.

**Definition 6.** For any $S \subseteq \mathbb{Z}_m^n$ a family of vectors $\{u[i]\}_{i=1}^n \subseteq (\mathbb{Z}_m)^h$ is said to be $S$-matching if

- $\langle u[i], u[i] \rangle = 0$ for every $i \in [n]$,
- $\langle u[i], u[j] \rangle \in S$ for every $i, j \in [n]$ such that $i \neq j$.

We will first assume the existence of such families and present the code. When the code is clear, a result from Grolmusz in [17] is given to show the existence of such families. But first some more facts and definitions.

**Fact 7.** For every odd $m$, there exists a finite field $\mathbb{F}_{2^t}$, with $t \leq m$, and an element $\gamma \in \mathbb{F}_{2^t}$ which is a generator of a multiplicative group of size $m$, i.e., $\gamma^m = 1$ and $\gamma^i \neq 1$ for $i \in \{1, 2, \ldots, m-1\}$.

**Proof.** First, note that $2 \in \mathbb{Z}_m^*$, since $m$ is odd. Thus there is some $t < m$ such that $2^t \equiv 1 \mod m$. So $m$ divides $2^t - 1$. For $\mathbb{F}_{2^t}$ we see $|\mathbb{F}_{2^t}^*| = 2^t - 1$. If $g$ is the generator of $\mathbb{F}_{2^t}^*$, we can take $\gamma = g^{2^{t-1}}$.

Fix $m$ odd. From the previous fact there exist $t$, $F = \mathbb{F}_{2^t}$ and $\gamma \in F$, such that $\gamma$ is the generator of a multiplicative subgroup of $F$ of order $m$.

**Definition 8.** A polynomial $P \in \mathbb{F}[x]$ is called an $S$-decoding polynomial if

- $P(\gamma^s) = 0$, $\forall s \in S$,
- $P(\gamma^0) = P(1) = 1$.

An easy way to find such a polynomial is presented in the following claim.

**Claim 9.** For any $S$ such that $0 \notin S$ there exists an $S$-decoding polynomial $P$ with at most $|S| + 1$ monomials

**Proof.** Define $\tilde{P}(x) = \prod_{s \in S}(x - \gamma^s)$. Then $P(x) = \tilde{P}(x)/\tilde{P}(1)$ is an $S$-decoding polynomial. Since the degree of $P$ is $|S|$, this polynomial consists of at most $|S| + 1$ monomials.
2.3.2 The Code

Suppose we have some odd \( m \) and fix \( t \) and \( \gamma \) as above. Suppose we have some set \( S \) with \( S \)-matching vectors \( \{u[i]\}_{i=1}^n, u[i] \in (\mathbb{Z}_m)^h \) and an \( S \)-decoding polynomial \( P \) with \( q \) monomials.

**Encoding:** We can define the encoding \( C : \mathbb{F}^n \rightarrow \mathbb{F}^m^h \) by defining \( C(e_i) \) for \( i \in [n] \) and use the linearity of \( C \) to extend it to \( \mathbb{F}^n \), i.e., for a message \( x = (x_1, x_2, \ldots, x_n) \), \( C(\sum_{i=1}^n x_ie_i) = \sum_{i=1}^n x_iC(e_i) \). The encoding of \( e_i \) is given by

\[
C(e_i) = (\gamma^{(u[i],y)})_{y \in (\mathbb{Z}_m)^h}
\]

(Concatenation of \( \gamma^{(u[i],y)} \) for all possible values of \( y \) into one vector). For a fixed \( x \) one can think of the codeword as a homomorphism from \((\mathbb{Z}_m)^h\) to \( \mathbb{F}^n \) (for a possibly corrupted codeword \( w \) of \( x \) the notation \( w(y) \) is used to indicate the entry of \( w \) at the same position as \( \sum_{i=1}^n x_i\gamma^{(u[i],y)} \) in the non-corrupted codeword). Thus the encoding is given by

\[
C(x) = \left( \sum_{i=1}^n x_i\gamma^{(u[i],y)} \right)_{y \in (\mathbb{Z}_m)^h}.
\]

**The decoder:** First, write \( P(x) = a_0 + a_1x^{b_1} + a_2x^{b_2} + \ldots + a_{q-1}x^{b_{q-1}} \). Suppose \( w \in \mathbb{F}^m^h \) is a corrupted version of the codeword \( x \in \mathbb{F}^n \) which is corrupted up to a \( \delta \)-fraction. If we want to recover entry \( i \) we use the following procedure \( D \).

- Pick \( \nu \in (\mathbb{Z}_m)^h \) at random.
- Query \( w(\nu), w(\nu + b_1u[i]), \ldots, w(\nu + b_{q-1}u[i]) \).
- Output \( x_i = \gamma^{-(u[i],\nu)}(a_0w(\nu) + a_1w(\nu + b_1u[i]) + \ldots + a_{q-1}w(\nu + b_{q-1}u[i])) \).

**Lemma 10.** The decoding procedure \( D \) is a perfectly smooth decoder for \( C \) with perfect recovery.

**Proof.** Since \( \nu \) is chosen uniformly at random, each query \( \nu, \nu + b_1u[i], \ldots, \nu + b_{q-1}u[i] \) is distributed uniformly.

To show that \( D \) has perfect recovery we only have to show that it decodes correctly, i.e., \( D^{C(x)}(i) = x_i \). Because the code is linear it is enough to prove that \( D^{C(e_i)}(j) = \delta_{ij} \).

First \( D^{C(e_i)}(i) \), then

\[
D^{C(e_i)}(i) = \gamma^{-(u[i],\nu)}(a_0\gamma^{(u[i],\nu)} + a_1\gamma^{(u[i],\nu+b_1u[i])} + \ldots + a_{q-1}\gamma^{(u[i],\nu+b_{q-1}u[i])}).
\]
By definition of the matching vectors $\langle u[i], \nu + cu[i] \rangle = \langle u[i], \nu \rangle + c \langle u[i], u[i] \rangle = \langle u[i], \nu \rangle$. And so

$$D^C(e_i)(i) = \gamma^{-\langle u[i], \nu \rangle}(a_0 \gamma^{\langle u[i], \nu \rangle} + a_1 \gamma^{\langle u[i], \nu \rangle + b_1 u[i]} + \ldots + a_{q-1} \gamma^{\langle u[i], \nu + b_{q-1} u[i] \rangle})$$

$$= a_0 + a_1 + \ldots + a_{q-1}$$

$$= P(1)$$

$$= 1.$$

Now the case where $D^C(e_i)(j)$, with $i \neq j$, then

$$D^C(e_i)(j) = \gamma^{-\langle u[i], \nu \rangle}(a_0 \gamma^{\langle u[i], \nu \rangle} + a_1 \gamma^{\langle u[i], \nu + b_1 u[j] \rangle} + \ldots + a_{q-1} \gamma^{\langle u[i], \nu + b_{q-1} u[j] \rangle})$$

$$= a_0 + a_1 \gamma^{h_1(u[i], u[j])} + \ldots + a_{q-1} \gamma^{h_{q-1}(u[i], u[j])}$$

$$= P(\gamma^{\langle u[i], u[j] \rangle})$$

$$= 0$$

Combining Lemma 10 and Theorem 5 gives the following corollary.

**Corollary 11.** For any $S$-matching vectors $\{u[i]\}_{i=1}^n \subseteq (\mathbb{Z}_m)^h$ and $S$-decoding polynomial with $q$ monomials there exists a $(q, \delta, \frac{1}{2} - q\delta)$-Locally Decodable Code $C : \mathbb{F}^n \to \mathbb{F}^{m^h}$.

Until now, we have assumed there is some set $S$ with $S$-matching vectors. But if these do not exist the whole construction is useless. Fortunately there is the following result from Groblemsz [17].

**Lemma 12.** Let $m = p_1 p_2 \cdots p_r$ be a product of $r$ distinct primes. Then there exists $c = c(m) > 0$, such that for every integer $h > 0$, there exists an explicitly constructible set-system $\mathcal{H}$ over the universe of $h$ elements (i.e., $\mathcal{H}$ is a set of subsets of $[h]$) and there is a set $S \subset \mathbb{Z}_m \setminus \{0\}$ such that:

1. $|\mathcal{H}| \geq \exp\left(c \frac{(\log h)^r}{(\log \log h)^{r-1}}\right)$.

2. The size of every set $H$ in set-system $\mathcal{H}$ is divisible by $m$, i.e., $|H| \equiv 0 \mod m$.

3. Let $G, H$ be any two different sets in set-system $\mathcal{H}$. Then the size of intersection of $G, H$ modulo $m$ is restricted to be in $S$, i.e., $\forall G, H \in \mathcal{H}$ such that $G \neq H$ it holds that $|G \cap H| \in S \mod m$.

4. $S$ is a set of size $2^r - 1$. 

13
5. \( \forall s \in S \) it holds that \( s \) (mod \( p_i \)) is 0 or 1 for all \( i \in [r] \).

This result has the following useful corollary.

**Corollary 13.** For every integer \( h, r \) and integer \( m = p_1 p_2 \cdots p_r \), which is a product of distinct primes, there exists a set \( S \) of size \( 2^r - 1 \) and a family of \( S \)-matching vectors \( \{u[i]\}_{i=1}^n \subseteq (\mathbb{Z}_m)^h \) such that \( n \geq \exp \left( c \frac{(\log h)^r}{(\log \log h)^{r-1}} \right) \).

**Proof.** Take the set system \( \mathcal{H} \) as in the previous lemma. For \( H \in \mathcal{H} \) define \( u_H \in (\mathbb{Z}_m)^h \) to be the indicator vector of \( H \in \mathcal{H} \). Then for every \( G, H \in \mathcal{H} \) it holds that \( \langle u_H, u_H \rangle = |H| \equiv 0 \) mod \( m \) and \( \langle u_H, u_G \rangle = |H \cap G| \equiv S \) mod \( m \).

Corollary 13 enables us to prove the existence of LDCs with \( 2^r \) queries.

**Theorem 14.** For any positive integer \( r \) there exists a \((q, \delta, \frac{1}{2} - q\delta)\)-Locally Decodable Code \( C : \mathbb{F}^n \rightarrow \mathbb{F}^N \), with codeword length \( N = \exp(\exp(O(\sqrt{\log n (\log \log n)^{r-1}}))) \) and \( q \leq 2^r \).

**Proof.** Fix \( m = p_1 p_2 \cdots p_r \) with \( p_i \) prime for \( i \in [r] \) and \( p_i \neq p_j \) for \( i \neq j \). And fix some \( h = \exp \left( O(\sqrt{\log n (\log \log n)^{r-1}}) \right) \). From the previous corollary we now know that there exists a set \( S \) of size \( 2^r - 1 \) and \( n = \exp \left( c \frac{(\log h)^r}{(\log \log h)^{r-1}} \right) \) \( S \)-matching vectors. Claim 9 gives us an \( S \)-decoding polynomial with at most \( 2^r \) monomials. Now we know from Theorem 11 that there exists a \((2^r, \delta, 2^r \delta)\)-Locally Decodable Code with codeword length \( m^h \). For \( m \) constant we have \( m^h = \exp(O(h)) \). So,

\[
m^h = \exp(O(h)) = \exp \left( \exp \left( O \left( \sqrt{\log n (\log \log n)^{r-1}} \right) \right) \right)
\]

concludes the proof. 

For a code with 3 queries we need an \( S \)-decoding polynomial consisting of 3 monomials, instead of the \( 2^r \) which are given by Claim 9. Efremenko was able to find an \( S \)-decoding polynomial with 3 monomials by exhaustive search. For more details on the method we refer to [12].

**Example 15.** Let \( m = 511 = 7 \cdot 73 \) and let \( S = \{1, 365, 147\} \). By Corollary 13 there exist \( S \)-matching vectors \( \{u[i]\}_{i=1}^n, u[i] \in (\mathbb{Z}_m)^h \), where \( n \geq \exp(c \frac{(\log h)^2}{(\log \log h)^{r-1}}) \). Set \( \mathbb{F} = \mathbb{F}_{2^9} \cong \mathbb{F}_2[X]/(X^9 + X^4 + 1) \).

Let \( \gamma \) be the generator of \( \mathbb{F}^\ast \). Then it can be verified that the polynomial \( P(x) = \gamma^{423} \cdot X^{65} + \gamma^{257} \cdot X^{12} + \gamma^{342} \) is an \( S \)-decoding polynomial with 3 monomials.

This example together with Corollary 11 proves the main result from [12].

**Theorem 16.** There exists a \((3, \delta, \frac{1}{2} - 3\delta)\)-Locally Decodable Code \( C : \mathbb{F}_2^t \rightarrow \mathbb{F}_2^N \) of length \( N = \exp(\exp(O(\sqrt{\log n \log \log n}))) \) for some \( t \).
2.3.3 Binary Case

The construction in the previous section does not give us binary codes for any number of queries. We can only obtain binary codes with at most 2 queries. In this section we present a binary code for any number of queries, for which there exists an \( S \)-decoding polynomial with the same number of monomials. The previous section gave \( S \)-decoding polynomials with 3 and \( 2^r \) monomials for any positive integer \( r \).

Let \( x = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_2^n \) be the message. We can also view it as a message in \( \mathbb{F}_2^n \). Let \( \tilde{C} \) be the encoding as obtained in the previous section, so \( \tilde{w} = \tilde{C}(x) \in (\mathbb{F}_2)^m \). Let \( P(x) = a_0 + a_1 x^{b_1} + a_2 x^{b_2} + \cdots + a_{q-1} x^{b_{q-1}} \) be the \( S \)-decoding polynomial. Set the codeword to be

\[
w = w_0 \circ w_1 \circ \cdots \circ w_{q-1} = a_0 \tilde{w} \circ a_1 \tilde{w} \circ \cdots \circ a_{q-1} \tilde{w},
\]

the concatenation of \( a_i \tilde{w} \) for \( i \in \{0, 1, \ldots, q-1\} \) where \( a_i \tilde{w} \) is the coordinate-wise scalar multiplication. Here we implicitly define \( w_i \) to be \( a_i \tilde{w} \) for \( i \in \{0, 1, \ldots, q-1\} \). Note that the codeword length is increased by a factor \( q \), but this is negligible in our parameters.

This codeword is still in \( \mathbb{F}_2^t \) and we need one in \( \mathbb{F}_2 \). The previous section showed us we can recover \( x_i \) using

\[
x_i \gamma^{\langle u[i], \nu \rangle} = w_0(\nu) + w_1(\nu + b_1 u[i]) + \cdots + w_{q-1}(\nu + b_{q-1} u[i]).
\]

We can apply some linear functional \( L : \mathbb{F}_2^t \to \mathbb{F}_2 \) on every coordinate of the codeword. Then we still see

\[
L(x_i \gamma^{\langle u[i], \nu \rangle}) = L(w_0(\nu)) + L(w_1(\nu + b_1 u[i])) + \cdots + L(w_{q-1}(\nu + b_{q-1} u[i])).
\]

Of course we want \( L(x_i \gamma^{\langle u[i], \nu \rangle}) = x_i \) and for \( x_i = 0 \) this holds, because then \( L(x_i \gamma^{\langle u[i], \nu \rangle}) = L(0) = 0 \). For \( x_i = 1 \) it is possible that \( L(x_i \gamma^{\langle u[i], \nu \rangle}) = L(\gamma^{\langle u[i], \nu \rangle}) = 0 \). We need to find a \( \nu \) such that \( L(\gamma^{\langle u[i], \nu \rangle}) = 1 \). A restriction on \( \nu \) affects the smoothness of the decoder, since we cannot pick \( \nu \) uniformly at random anymore. To restrain this, we choose \( \nu \) such that \( \mathbb{P}[L(\gamma^{\langle u[i], \nu \rangle}) = 1] \geq \frac{1}{2} \) for all \( i \in [n] \).

**Lemma 17.** There exists a linear functional \( L : \mathbb{F}_2^t \to \mathbb{F}_2 \) such that

\[
\forall i \in [n] \quad \mathbb{P}_{\nu \in (\mathbb{Z}_m)^t}[L(\gamma^{\langle u[i], \nu \rangle}) = 1] \geq \frac{1}{2}
\]

**Proof.** Note that for a random \( \nu, \langle u[i], \nu \rangle \) is a random number in \( \mathbb{Z}_m \), since \( u[i] \neq 0 \) \( \forall i \in [n] \). Therefore it is sufficient to find an \( L \) such that

\[
\mathbb{P}_{j \in \mathbb{Z}_m}[L(\gamma^j) = 1] \geq \frac{1}{2}.
\]
For fixed $j$ and random $L$, $\mathbb{P}[L(\gamma^j) = 1] = \frac{1}{2}$, so
\[
\mathbb{E}_L[\mathbb{P}_{j \in \mathbb{Z}_m}[L(\gamma^j) = 1]] = \frac{1}{2}
\]
But then there exists an $L$ such that
\[
\mathbb{P}_{j \in \mathbb{Z}_m}[L(\gamma^j) = 1] \geq \frac{1}{2}
\]
\[\square\]

We can now give step by step instructions for the code.

**The Code:** Choose a linear functional $L : \mathbb{F}_2^t \to \mathbb{F}_2$, such that $\mathbb{P}_{j \in \mathbb{Z}_m}[L(\gamma^j) = 1] \geq \frac{1}{2}$.

1. Let $x = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_2^n$ be the message.
2. Let $\tilde{C}$ be the code obtained from working in $\mathbb{F}_2$.
3. Define $C(x) = w_0 \circ w_1 \circ \cdots \circ w_{q-1} = L(a_0 \tilde{C}(x) \circ a_1 \tilde{C}(x) \circ \cdots \circ a_{q-1} \tilde{C}(x))$. We apply $L$ to every coordinate of $a_0 \tilde{C}(x) \circ a_1 \tilde{C}(x) \circ \cdots \circ a_{q-1} \tilde{C}(x)$.

**The Decoder $\mathcal{D}$:**

1. Pick $\nu \in (\mathbb{Z}_m)^h$ at random conditioned on $L(\gamma^{(u[i],\nu)}) = 1$.
2. Query $w_0(\nu), w_1(\nu + b_1 u[i]), \ldots, w_{q-1}(\nu + b_{q-1} u[i])$.
3. Output $x_i = w_0(\nu) \oplus w_1(\nu + b_1 u[i]) \oplus \cdots \oplus w_{q-1}(\nu + b_{q-1} u[i])$.

**Theorem 18.** The binary code $C$ defined above is a $(q, \delta, \frac{1}{2} - 2q\delta)$-Locally Decodable Code.

**Proof.** First, we will show the decoder decodes correctly. As before, by linearity, we only have to show $\mathcal{D}^{C(e_i)}(j) = \delta_{ij}$.

We start with the case $\mathcal{D}^{C(e_i)}(i)$,
\[
\mathcal{D}^{C(e_i)}(i) = L(a_0 \gamma^{(u[i],\nu)}) \oplus L(a_1 \gamma^{(u[i],\nu+b_1 u[i])}) \oplus \cdots \oplus L(a_{q-1} \gamma^{(u[i],\nu+b_{q-1} u[i])})
\]
By definition of the matching vectors $\langle u[i], \nu + cu[i] \rangle = \langle u[i], \nu \rangle + c \langle u[i], u[i] \rangle = \langle u[i], \nu \rangle$. And so
\[
\mathcal{D}^{C(e_i)}(i) = L(a_0 \gamma^{(u[i],\nu)}) \oplus L(a_1 \gamma^{(u[i],\nu+b_1 u[i])}) \oplus \cdots \oplus L(a_{q-1} \gamma^{(u[i],\nu+b_{q-1} u[i])})
= L(a_0 \gamma^{(u[i],\nu)}) \oplus a_1 \gamma^{(u[i],\nu+b_1 u[i])} \oplus \cdots \oplus a_{q-1} \gamma^{(u[i],\nu+b_{q-1} u[i])}
= L(P(1) \gamma^{(u[i],\nu)})
= L(\gamma^{(u[i],\nu)})
= 1.
\]
For $\mathcal{D}^{C(e_i)}(j)$ with $i \neq j$ we see

$$\mathcal{D}^{C(e_i)}(i) = L(a_0 \gamma^{(u[j],\nu)}) \oplus L(a_1 \gamma^{(u[j],\nu+b_1 u[i])}) \oplus \cdots \oplus L(a_{q-1} \gamma^{(u[j],\nu+b_{q-1} u[i])})$$

$$= L\left(\gamma^{(u[j],\nu)} \left(a_0 \oplus a_1 \gamma^{b_1 (u[j],u[i])} \oplus \cdots \oplus a_{q-1} \gamma^{b_{q-1} (u[j],u[i])}\right)\right)$$

$$= L\left(\gamma^{(u[i],\nu)} P\left(\gamma^{(u[j],u[i])}\right)\right)$$

$$= L(0)$$

$$= 0.$$

Secondly, we prove that if at most a $\delta$-fraction of a corrupted codeword

$$w = w_0 \circ w_1 \circ \cdots \circ w_{q-1}$$

is corrupted, the probability of querying a corrupted index is at most $2q\delta$. Let $\delta_j$ be the fraction of corrupted bits in $w_j$, then $\frac{1}{q} \sum_{j=0}^{q-1} \delta_j = \delta$. Because $L$ was chosen such that $\nu$ could be picked uniformly from at least half of all possible values, the probability that query $j$ will be corrupted is at most $2\delta_j$. The probability that one of the queries is corrupted is at most $\sum_{j=0}^{q-1} 2\delta_j = 2q\delta$.

For $w = C(x)$ we have perfect recovery. Hence, for $w$ a corrupted codeword which is corrupted up to a $\delta$-fraction, it holds that

$$\mathbb{P}[\mathcal{D}^w(i) = x_i] \geq 1 - 2q\delta.$$

Thus $\mathcal{D}$ is a $(q, \delta, \frac{1}{2} - 2q\delta)$-local decoder. 

\[\square\]

### 2.4 Matching Vector Codes

One could say there are three families of Locally Decodable Codes, separated by the technical ideas that underlie them. The first family of codes is based on Reed–Muller codes [25].

The second family of Locally Decodable Codes are based on Private Information Retrieval Schemes. As already mentioned there is a close connection between Private Information Retrieval and Locally Decodable Codes. It is possible to convert one into the other and vice versa. More details about the conversion can be found in Section 3.3.

The third and youngest family of Locally Decodable Codes is the family of Matching Vector Codes. The framework was initiated by Yekhanin in [33]. The code consists of evaluating a certain polynomial in $F_q[x_1, \ldots, x_n]$ in all points of $F_q^n$ for some finite field $F_q$. The decoding procedure is based on shooting a line in a certain direction and decoding along it. The polynomials used depend on the chosen matching family of vectors. The LDCs of Efremenko [12] as described
in the previous section can also be viewed in the Matching Vectors framework. Since the introduction there have been some improvements in the decoder, such that it can tolerate larger fractions of errors [5, 11]. The currently used matching families of vectors are based on a result by Grolmusz [17] which is given in the previous section, but if matching families of vectors with better parameters are discovered, they can easily be plugged in the Matching Vectors framework to yield better Locally Decodable Codes. Unfortunately, there is a very recent result from Bohmnick et al. [6] which gives an upper bound on the size of matching families. This upper bound is fairly close to the size of the currently used matching families and implies that Matching Vector Codes cannot be improved very much.

In this section we will only briefly touch the basic encoding and decoding procedures for Matching Vector Codes. For more details, improved decoding procedures and the construction of matching families, we refer to [34].

In the rest of this section we will follow the beginning of Chapter 3 in [34]. As mentioned before, the code evolves around matching families of vectors.

**Definition 19.** Let \( S \subseteq \mathbb{Z}_m \setminus \{0\} \). Two families \( U = \{u[i]\}_{i=1}^n \) and \( V = \{v[i]\}_{i=1}^n \) of vectors in \((\mathbb{Z}_m)^h\) form an \( S \)-matching family if

- \( \langle u[i], v[i] \rangle = 0 \quad \forall i \in [n], \)
- \( \langle u[i], v[j] \rangle \in S \quad \forall i, j \in [n], \) such that \( i \neq j. \)

If we have a set \( S \) and two \( S \)-matching families \( U \) and \( V \) in \((\mathbb{Z}_m)^h\) we are able to construct a Locally Decodable Code. To do this we first need some notation.

- Let \( q \) be a prime power such that \( m \) divides \( q - 1 \). Denote the subgroup of \( \mathbb{F}_q^* \) of order \( m \) by \( \mathcal{C}_m. \)
- Let \( \gamma \) be a generator of \( \mathcal{C}_m. \)
- For \( w \in (\mathbb{Z}_m)^h \) and \( \gamma \in \mathbb{Z}_m \) define \( \gamma^w \) by \( (\gamma^{w_1}, \gamma^{w_2}, \ldots, \gamma^{w_h}). \)
- For \( w, v \in (\mathbb{Z}_m)^h \) define the multiplicative line \( M_{w,v} \) by
  \[
  M_{w,v} = \{ \gamma^{w+\lambda v} \mid \lambda \in \mathbb{Z}_m \}. 
  \]
- For \( u \in (\mathbb{Z}_m)^h \) define the monomial \( \text{mon}_u \in \mathbb{F}_q[z_1, z_2, \ldots, z_h] \) by
  \[
  \text{mon}_u(z_1, z_2, \ldots, z_h) = \prod_{i \in [h]} z_i^{u_i}. 
  \]
Note that for \( u, v, w \in (\mathbb{Z}_m)^h \) and \( \lambda \in \mathbb{Z}_m \) it holds that
\[
\text{mon}_u(\gamma^{w+\lambda v}) = \prod_{i \in [h]} \gamma^{(w_i+\lambda v_i)u_i}
\]
\[
= \prod_{i \in [h]} \gamma^{w_iu_i} \gamma^{\lambda v_iu_i}
\]
\[
= \gamma^{\langle w, u \rangle} \left( \gamma^{\langle \lambda, u \rangle} \right).
\]

So evaluating \( \text{mon}_u \) in \( M_{w,v} \) is the same as evaluating
\[
\gamma^{\langle w, u \rangle} \left( \gamma^{\langle \lambda, u \rangle} \right)
\]
for \( z = \gamma^\lambda \in \mathcal{C}_m \). We can use this observation to our advantage as will become clear in the decoding algorithms.

**Theorem 20.** Let the families \( \mathcal{U} = \{u[i]\}_{i=1}^n \) and \( \mathcal{V} = \{v[i]\}_{i=1}^n \) be an \( S \)-matching family in \((\mathbb{Z}_m)^h\), where \( |S| = s \). Then there exists an \((s+1, \delta, \frac{1}{2} - (s+1)\delta)\)-Locally Decodable Code \( C : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^h \).

**Proof.** First we show how the encoding works.

**Encoding:** Let \( x \in \mathbb{F}_q^n \) be a message. We encode \( x \) by
\[
C(x) = \left( \sum_{j=1}^n x_j \cdot \text{mon}_{u[j]}(y) \right)_{y \in (\mathbb{C}_m)^h}
\]
(Concatenation of \( \sum_{j=1}^n x_j \cdot \text{mon}_{u[j]}(y) \) for all possible values of \( y \) into one vector).

Just like in the LDC of Efremenko, we can view the codeword for a fixed \( x \) as a homomorphism which maps \( y \) to \( \sum_{j=1}^n x_j \cdot \text{mon}_{u[j]}(y) \).

Now let \( w \) be a corrupted version of the codeword \( C(x) \) which is corrupted up to a \( \delta \)-fraction. Suppose we want to recover \( x_i \).

**Basic Decoding:**
- Pick a random \( \nu \in (\mathbb{Z}_m)^h \).
- Query \( w(\gamma^{\nu+\lambda v[i]}) \) for all \( \lambda \in \{0, 1, \ldots, s\} \).
- Recover the sparse univariate polynomial \( p(z) \in \mathbb{F}_q[z] \) such that \( \text{supp}(p)^2 \subseteq S \cup \{0\} \) and \( p(\gamma^\lambda) = w(\gamma^{\nu+\lambda v[i]}) \) for \( \lambda \in \{0, 1, \ldots, s\} \).
- Return \( x_i = \frac{p(0)}{\gamma^{(0,0)}} \).

For \( p \in \mathbb{F}_q[z] \), \( \text{supp}(p) \) is the set of powers of the monomials in \( p \) with non-zero coefficients, i.e., for a monomial \( 6z^e \) in \( p \), \( e \in \text{supp}(p) \).
To show that this decoding procedure works, we first assume \( w \) is the non-corrupted codeword \( C(x) \) of \( x \). Then by (2.1),

\[
\sum_{j=1}^{n} x_j \cdot \text{mon}_{u_j}(\gamma^\nu + \lambda v[i]) = \sum_{j=1}^{n} x_j \cdot \gamma^{\langle u[j], v[i] \rangle} z^{\langle u[j], v[i] \rangle}
\]

for \( z = \gamma^\lambda \in C_m \). We can use the properties of the \( S \)-matching family to see that

\[
\sum_{j=1}^{n} x_j \cdot \gamma^{\langle u[j], v[i] \rangle} z^{\langle u[j], v[i] \rangle} = x_i \cdot \gamma^{\langle u[i], v[i] \rangle} + \sum_{j \in [n] \setminus \{i\}} x_j \cdot \gamma^{\langle u[j], v[i] \rangle} z^{\langle u[j], v[i] \rangle}
\]

\[
= x_i \cdot \gamma^{\langle u[i], v[i] \rangle} + \sum_{s \in S} \left( \sum_{j \in [n]: \langle u[j], v[i] \rangle = s} x_j \cdot \gamma^{\langle u[j], v[i] \rangle} \right) z^s.
\]

(2.2)

Denote the function from 2.2 by \( f(z) \), then it is clear that \( \text{supp}(f) \subseteq S \cup \{0\} \). Since \( p \) the sparse univariate polynomial that is chosen such that for every \( \lambda \in \{0, 1, \ldots, s\} \), \( p(\gamma^\lambda) = w(\gamma^\nu + \lambda v[i]) = f(\gamma^\lambda) \) and \( \text{supp}(p) \subseteq S \cup \{0\} \), we see that

\[
p(0) = f(0) = x_i \cdot \gamma^{\langle u[i], v[i] \rangle}.
\]

Note that \( p \) is unique due to standard properties of Vandermonde matrices [23]. The decoding procedure returns the correct \( x_i \) when the codeword is not corrupted.

Suppose a \( \delta \)-fraction of the corrupted codeword \( w \) is corrupted. Since \( \nu \) is picked uniformly at random, each query goes to a random location. The probability that one query picks a corrupted entry is \( \delta \). Applying the union bound yields the desired result.

The decoding procedure above is essentially the same as the one used in the LDC from Efremenko [12]. What happens in the proof above is that we interpolate the polynomial \( p(y) \) to obtain the free coefficient. For \( m \) and \( S \) with special properties, it is sometimes possible to recover the free coefficient with a smaller number of queries. This is what Efremenko did to obtain better parameters for 3 queries.

The decoding procedure can be improved in terms of tolerating a larger fraction of errors. The procedure above can tolerate up to a \( \frac{1}{2s} \)-fraction of errors for \( s \) the number of queries. In [11], using the assumption that the elements of \( S \) are bounded for some positive real, a decoding procedure is given which can tolerate up to a \( \frac{1}{4} \)-fraction of errors, independent from the number of queries.

For more details, better decoding procedures and information about the construction of matching families we refer to [34].
3 Classical Private Information Retrieval

3.1 Definition and First Example

We start this chapter by giving the formal definition of a Private Information Retrieval scheme.

**Definition 21 (Private Information Retrieval scheme).** A one-round \((1 - \delta)\)-secure \(k\)-server Private Information Retrieval scheme with recovery probability \(1/2 + \epsilon\), query size \(t\), answer size \(a\) for an \(n\)-bit database \(x\), consists of a randomized algorithm (the user), and \(k\) deterministic algorithms \(S_1, S_2, \ldots, S_k\) (the servers), such that

- For some input \(i \in [n]\), the user generates \(k\) \(t\)-bit strings \(q_1, q_2, \ldots, q_k\) (called “queries”) and sends these to the respective servers \(S_1, S_2, \ldots, S_k\). Every server \(S_j\) sends back an answer \(a_j = S_j(x, q_j)\).
- The user outputs a guess \(b\) for \(x_i\) based on \(a_1, a_2, \ldots, a_k\) and \(i\). The probability that \(b = x_i\) is at least \(1/2 + \epsilon\) for all \(x, i\).
- For all \(x\) and \(j\), the distributions on \(q_j\) are \(\delta\)-close (in total variation distance\(^1\)) for different \(i\).

In this thesis we only consider schemes with \(\delta = 0\) (The servers get no information at all about \(i\), informally one could say the odds of guessing \(i\) correctly after executing the scheme are the same as before) and \(\epsilon = 1/2\) (perfect recovery).

We will illustrate a PIR scheme with \(\mathcal{O}(\sqrt{n})\) communication complexity for 2 servers.

**Example 22.** First we arrange the \(n\)-bit database in a \(\sqrt{n} \times \sqrt{n}\) square (if \(n\) is not a square just add 0s until it is). Each bit \(x_i\) from the database is uniquely determined by coordinates \((i_1, i_2)\). The user generates some random string \(q_1 \in \{0, 1\}^{\sqrt{n}}\). She sends \(q_1\) to server 1 and \(q_2 = q_1 \oplus e_{i_1}\) to server 2. Let \(C_\ell\) be the

\(^1\)For two probability distributions \(p, q\) the total variation distance is given by \(\frac{1}{2} \sum_x |p(x) - q(x)|\), where \(x\) runs over all possible inputs.
ℓ-th column of the database. For \( i \in \{1, 2\} \) server \( s \) answers with the \( \sqrt{n} \) bits \( q_s \cdot C_1, q_s \cdot C_2, \ldots, q_s \cdot C_{\sqrt{n}} \).

The user selects \( q_1 \cdot C_{i_2} \) and \( q_2 \cdot C_{i_2} \) and calculates

\[
(q_1 \cdot C_{i_2}) \oplus (q_2 \cdot C_{i_2}) = (q_1 \cdot C_{i_2}) \oplus ((q_1 \oplus e_i) \cdot C_{i_2}) = e_i \cdot C_{i_2} = x_i.
\]

### 3.2 Better Schemes

Already when Chor et al. introduced PIR in [9] they presented schemes with lower communication complexity. For the 2-server case they achieved \( O(n^{1/3}) \) bits of communication and for the general case of \( k \) servers \( O(n^{1/k}) \) bits were used.

The scheme with \( O(n^{1/3}) \) bits of communication is nowadays still the best known and we will present it here. To do this we will first give the following scheme which serves as a basis for the best known 2-server scheme.

**Example 23** (Private Information Retrieval scheme with communication complexity \( O(n^{1/d}) \) for \( k = 2^d \) servers.). The \( n \)-bit database \( x \) is replicated over \( k = 2^d \) non-communicating servers \( S_1, S_2, \ldots, S_{2^d} \). First, we assume without loss of generality that \( n = \ell^d \). We embed \( x \) in the \( d \)-dimensional (hyper)cube such that every index \( j \) is associated with a \( d \)-tuple \( (j_1, j_2, \ldots, j_d) \) and so \( x_i \) is uniquely determined by a \( d \)-tuple \( (i_1, i_2, \ldots, i_d) \). The scheme consists of three steps.

1. The user selects \( d \) random subsets \( Q_0^1, Q_0^2, \ldots, Q_0^s \subseteq \ell^d \). We write \( q_0^1, q_0^2, \ldots, q_0^s \in \{0, 1\}^d \) for the corresponding indicator strings.

   She defines \( q_1^1 = q_0^1 \oplus e_{i_1}, q_2^1 = q_0^2 \oplus e_{i_2}, \ldots, q_1^d = q_0^d \oplus e_{i_d} \). From these indicator strings we can construct sets \( Q_1^1, Q_1^2, \ldots, Q_1^d \).

   We can convert every number of a server into binary such that every server is identified by a string \( \sigma_1 \sigma_2 \ldots \sigma_d \in \{0, 1\}^d \). The user sends the strings \( q_1^1, q_2^1, q_3^1, \ldots, q_1^d, q_2^d, \ldots, q_3^d \) to server \( S_{\sigma_1 \sigma_2 \ldots \sigma_d} \).

2. Server \( S_{\sigma_1 \sigma_2 \ldots \sigma_d} \) receives \( q_1^1, q_2^2, \ldots, q_3^d \) and then also knows the corresponding sets \( Q_1^1, Q_2^2, \ldots, Q_3^d \). The server replies with the bit

   \[
   A_{\sigma_1 \sigma_2 \ldots \sigma_d} = \bigoplus_{j_1 \in Q_1^1, j_2 \in Q_2^2, \ldots, j_d \in Q_d^d} x_{j_1, j_2, \ldots, j_d}
   \]

3. The user calculates the exclusive-or from the \( k \) bits she has received.

   \[
   x_i = \bigoplus_{\sigma_1 \sigma_2 \ldots \sigma_d \in \{0, 1\}^d} A_{\sigma_1 \sigma_2 \ldots \sigma_d}
   \]

22
The scheme works since the bit \( x_{i_1, i_2, \ldots, i_d} \) is only added to the exclusive-or once and all other entries are added an even number of times, which can be seen as follows. The bit \( x_{j_1, j_2, \ldots, j_d} \) is added only by one server, while if we consider another bit \( x_{j_1, j_2, \ldots, j_d} \) which differs from \( x_{i_1, i_2, \ldots, i_d} \) by at least coordinate \( j_t \neq i_t \) we see
\[
(j_1, j_2, \ldots, j_d) \in Q_1^{\sigma_1} \times \cdots \times Q_{t-1}^{\sigma_{t-1}} \times Q_t^{\sigma_t} \times Q_{t+1}^{\sigma_{t+1}} \times \cdots \times Q_d^{\sigma_d}
\]
if and only if
\[
(j_1, j_2, \ldots, j_d) \in Q_1^{\sigma_1} \times \cdots \times Q_{t-1}^{\sigma_{t-1}} \times Q_t^{1} \times Q_{t+1}^{\sigma_{t+1}} \times \cdots \times Q_d^{\sigma_d}.
\]

If coordinate \( j_1, j_2, \ldots, j_d \) differs from \( i_1, i_2, \ldots, i_d \) in \( m \) coordinates, the bit \( x_{j_1, j_2, \ldots, j_d} \) is added either 0 or 2\(^m\) times to the total sum.

Now, if we calculate the exclusive-or, the entries which are added an even number of times cancel and the only value that remains is \( x_{i_1, i_2, \ldots, i_d} \).

The privacy is guaranteed since from the point of each server it gets \( d \) random strings and it receives either \( S^0_j \) or \( S^1_j \) for \( j \in [n] \) but never both. So no information about the index \( i \) is leaked.

As far as the communication is concerning, the user sends \( d \) strings of \( \ell \) bits to each server and each server replies with one bit resulting in a total communication of \( k \cdot (d \cdot \ell + 1) = 2^d \cdot (d \cdot n^2 + 1) = O(n^2) \).

To improve the scheme we make use of so-called covering codes.

**Definition 24 (Covering Codes).** Consider the set \( \{0, 1\}^d \). A Covering Code \( C_d \) with radius \( r \) is a collection \( C_d = \{c_1, c_2, \ldots, c_k\} \subseteq \{0, 1\}^d \) such that the balls of radius \( r \), where the distance is given by the Hamming distance, around the elements in the set \( C_d \) cover the whole set \( \{0, 1\}^d \) i.e.,
\[
\{0, 1\}^d \subseteq \cup_{c_j \in C_d} B(c_j, r)
\]
where \( B(c_j, r) \) is the set of all strings \( \{0, 1\}^d \) which differ in at most \( r \) positions from \( c_j \).

An easy example is the following.

**Example 25.** Consider the set \( C_1 = \{000, 111\} \). Then
\[
B(000, 1) = \{000, 001, 010, 100\},
\]
\[
B(111, 1) = \{111, 110, 101, 011\},
\]
and so \( B(000, 1) \cup B(111, 1) = \{0, 1\}^3 \).

We can use this example to give the 2-server PIR scheme with communication complexity \( O(n^2) \).
Example 26. Consider the scheme from Example 23 for 8 = 2^3 servers.

Server $S_{000}$ gets the queries $q_1^0, q_2^0, q_3^0$. Server $S_{001}$ also gets $q_1^0$ and $q_2^0$, but the third query will be of the form $q_3^0 \oplus e_j$ for $j \in [\sqrt{n}]$. We let $S_{000}$ emulate $S_{001}$ by sending the $\sqrt{n}$ bits corresponding to the $\sqrt{n}$ replies which could have been sent by $S_{001}$. In the same way it also emulates $S_{010}$ and $S_{100}$.

Server $S_{111}$ emulates $S_{110}$, $S_{101}$ and $S_{011}$ in the same fashion and so we get a 2-server scheme with a total communication of $2 \cdot (3 \cdot \sqrt{n} + 3 \cdot \sqrt{n} + 1) = \mathcal{O}(n^{\frac{3}{2}})$ bits of communication.

It is an open question if there exist 2-server PIR schemes with lower communication complexity. The best known lower bound for 2-server PIR is $(5 - o(1)) \log n$ [31]. So there is a huge gap between this lower bound and the best known upper bound of $\mathcal{O}(n^{\frac{3}{2}})$.

The covering codes trick does not only work for the 2-server scheme, but for covering codes with radius 1 in general.

Theorem 27. Let $d$ and $k$ be integers such that there exists a $k$-word Covering Code of radius 1 for $\{0,1\}^d$. Then there exists a Private Information Retrieval scheme for $k$ servers holding an $n$-bit database with communication complexity $\mathcal{O}(k + (2^d + (d - 1) \cdot k) \cdot n^{\frac{3}{2}})$.

Proof. We embed the $n$-bit database into the $d$-dimensional (hyper)cube. Suppose the user is interested in $x_i$ which corresponds to position $(i_1, i_2, \ldots, i_d)$. Just like in the scheme in Example 23 she randomly selects $Q_1^d, Q_2^d, \ldots, Q_d^d \subseteq [n^{\frac{3}{2}}]$ with corresponding indicator strings $q_1^d, q_2^d, \ldots, q_d^d \in \{0,1\}^l$. She sets $q_1^d = q_1^0 \oplus e_1$, $q_2^d = q_2^0 \oplus e_2$, ..., $q_d^d = q_d^0 \oplus e_d$ and to the servers $S_c$ with $\sigma_1 \ldots \sigma_d = c \in C_d$ she sends the strings $q_1^{c_1}, \ldots, q_d^{c_d}$. These servers send their own 1-bit answer (see Example 23) and the $n^{\frac{3}{2}}$ possible answers ($n^{\frac{3}{2}}$-bits) from the servers corresponding to the words covered by the codeword $c$. The user receives the information she needs to retrieve $x_i$ and the privacy is guaranteed, since the servers get the same information as in example 23, only their answer functions are different. The users send $d \cdot n^{\frac{3}{2}}$ bits to $k$ servers, for these $k$ servers only 1-bit answers are needed and for the other $2^d - k$ servers (note that it suffices to emulate a server only once^2) $n^{\frac{3}{2}}$ bits are sent as reply. This sums up to $k + (2^d + (d - 1) \cdot k) \cdot n^{\frac{3}{2}}$ bits.

Another case where this result is interesting is the case of $\{0,1\}^4$ where we have the covering code $C_4 = \{0000, 1111, 1000, 0111\}$. Applying the theorem above results in a PIR scheme for 4 servers with communication complexity $\mathcal{O}(n^{\frac{3}{2}})$. For $\{0,1\}^d$ with $d \geq 5$ there do not exist Cover Codes which improve previously known results.

^2Formally, we consider an exact cover of $\{0,1\}^d$ by sets $S_{c_j} \subseteq B(c_j, 1)$ for $j \in [k]$. 

24
The scheme from Chor et al. with $O(n^{\frac{1}{k}})$ communication complexity was first improved by Ambainis [1].

**Theorem 28.** There exists a $k$ server Private Information Retrieval scheme with communication complexity $O(n^{\frac{1}{2k-1}})$.

For more than 5 years this was the best known construction, but in 2002 it was improved to the best known PIR scheme which does not use LDCs and is due to Beimel et al. [4]. In their article they presented a scheme which gave the following theorem.

**Theorem 29.** There exists a Private Information Retrieval scheme for $k \geq 3$ server with communication complexity $n^{O(\log \log k \log k)}$.

Finally, we present a way to balance the number of bits sent by the user and the number of bits sent by the server. Often schemes are unbalanced in the sense that the user sends more bits to the server than the server to the user or vice versa. The following example illustrates a way to use this to our advantage and lower the communication complexity of such schemes.

**Example 30.** Suppose we have a Private Information Retrieval scheme for an $n$-bit database, then we can derive a scheme for an $m \cdot n$-bit database as follows.

1. The user views the $m \cdot n$-bit database as an $m \times n$ matrix.
2. To retrieve the bit at position $(i_1, i_2)$, the user sends the same query it would have sent to retrieve the $i_2$th bit from an $n$ bit database.
3. The servers send $m$ answers each. Each answer is the answer it would have given if it was executing the scheme for an $n$-bit database with one of the $m$ rows (from the matrix) as database.
4. Using the answer to the $i_1$th row, the user can compute the desired bit.

An application of this method is the scheme from Example 22. It is constructed from the following basic 2-server scheme.

**Example 31.**

1. The user selects an $n$-bit string $q_1$ at random.
2. Define $q_2 = q_1 \oplus e_i$ and send $q_1$ to server 1 and $q_2$ to server 2.
3. The servers reply with the inner product modulo 2 between the query $q_j$, $j \in \{1, 2\}$ and the $n$-bit database.
4. The user exclusive-ors the 2 bits she receives.
3.3 Connection between Smooth Codes and PIR

We already saw a link between Smooth Codes and Locally Decodable Codes. This section will give a connection between Smooth Codes and PIR. There has been a time where the best known constructions of LDCs came from PIR schemes. This is due to the following theorem from [19] which is a generalisation of the 1-secure case of Lemma 7.1 in [16].

**Theorem 32.** Assume we have a one-round, 1-secure, $k$-server Private Information Retrieval scheme with recovery probability $\frac{1}{2} + \epsilon$, query size $t$ and answer size $a$ for an $n$-bit database $x$. Then there exists a $(k, k+1, \epsilon)$-Smooth Code $C : \{0, 1\}^n \rightarrow (\{0, 1\}^a)^N$, with $N \leq (k^2 + k)2^t$.

**Proof.** We enumerate all possible answers from each server. The PIR system can be viewed as an encoding of the database $x \in \{0, 1\}^n$ as $\text{PIR}(x) \in (\{0, 1\}^a)^k$. The user can reconstruct $x_i$ using $k$ queries (the ones which are generated during the PIR scheme). These queries all correspond to an $a$-bit block of $\text{PIR}(x)$, namely the answer the corresponding server would give. This is already a correct decoder, except it is not smooth yet.

Let $\text{PIR}(x)_j$ denote the $j$-th block of $a$ bits from $\text{PIR}(x)$ and let $p_j$ denote the probability that $\text{PIR}(x)_j$ is queries while decoding some bit $x_i$. This $p_j$ is independent of $i$ since we have a 1-secure PIR scheme. Note that $\sum_j p_j = k$.

The smooth code $C$ encodes $x$ as $\text{PIR}(x)$ where each entry $\text{PIR}(x)_j$ is copied $\lceil p_j k 2^t \rceil$ times.

The decoding algorithm first selects $k$ queries $q_1, \ldots, q_k$ in the same way the decoding algorithm for $\text{PIR}(x)$ selects the queries. The queries $q_1, \ldots, q_k$ all correspond to an entry of $\text{PIR}(x)$. Additionally one of the $\lceil p_j k 2^t \rceil$ copies is selected uniformly at random. Thus every entry is queried with probability

$$p_j : \frac{1}{\lceil p_j k 2^t \rceil} \leq \frac{p_j}{p_j k 2^t} \leq \frac{1}{k 2^t}.$$  

The code consists of the following number of $a$-bit blocks

$$N = \sum_{j=1}^{k 2^t} \lceil p_j k 2^t \rceil \leq \sum_{j=1}^{k 2^t} 1 + p_j k 2^t = k 2^t + \sum_{j=1}^{k 2^t} p_j k 2^t = (k + 1)k 2^t.$$  

The probability that an entry is queried is at most $\frac{k+1}{N}$.

Nowadays the best PIR schemes come from LDCs. The following theorem is not really needed for this conversion, since the best known LDCs already possess a perfectly smooth decoder with perfect recovery. But suppose we have an LDC which had to be converted into a Smooth Code using Theorem 4, then we can use the following theorem from [7] to make it perfectly smooth.
Theorem 33. Let $C : \{0,1\}^n \rightarrow \{0,1\}^N$ be a $(q, c, \epsilon)$-Smooth Code, then $C$ is also a $(q, q, \frac{\epsilon^2}{2c})$-Smooth Code.

Each query is distributed uniformly, which gives us the tools to create a PIR scheme.

Theorem 34. Let $C : \{0,1\}^n \rightarrow (\{0,1\}^r)^N$ be a $(q, q, \epsilon)$-Smooth Code. Then we can construct a one-round, 1-secure, $q$-server Private Information Retrieval scheme with recovery probability $\frac{1}{2} + \epsilon$, query size $\log(N)$ and answer size $a$.

Proof. Both the user and the servers act like the database $x$ is encoded in $C(x)$. Suppose the user wants to retrieve $x_i$. The user generates queries using the smooth decoder. These are all indices from $C(x)$. The user sends each query to a different server. Each server responds with the bit corresponding to the received index. From these answers the user can retrieve the desired bit using the smooth decoder.

The privacy is guaranteed since every index is queried with probability $\frac{a}{m}$.

The following corollary is immediate from the previous theorem and the Locally Decodable Codes from Efremenko.

Corollary 35. For every positive integer $r$ there exists a one-round, 1-secure, $2^r$-server Private Information Retrieval scheme with recovery probability 1, query size $n^{O\left(\sqrt{\log \log n \log \log n} - 1\right)}$ and answer size 1.

Proof. From Theorems 14 and 18 we obtain a binary $(2^r, 0, \frac{1}{2})$-Locally Decodable Code of length

$$\exp\left(\exp\left(O\left(\sqrt{\log n \log \log n} \right)^{r-1}\right)\right) = \exp\left(n^{O\left(\sqrt{\log \log n \log \log n} - 1\right)}\right).$$

Due to Lemma 10 the code is perfectly smooth with perfect recovery. Using Theorem 34 yields the desired PIR scheme.

Theorem 16 gives us a $(3, 0, \frac{1}{2})$-Locally Decodable Code with codeword length $\exp\left(n^{O\left(\sqrt{\log \log n \log \log n} \right)}\right)$. Converting this into a PIR scheme using Theorem 34 yields the currently best-known PIR scheme for 3 servers.

Corollary 36. There exists a one-round, 1-secure, 3-server Private Information Retrieval scheme with recovery probability 1, query size $n^{O\left(\sqrt{\log \log n \log \log n} \right)}$ and answer size 1.
4 Quantum Private Information Retrieval

4.1 Preliminaries

4.1.1 Qubits and Quantum States

Consider the vector space $\mathbb{C}^2$. We can choose an orthonormal basis $|0\rangle$ and $|1\rangle$ and identify these with the vectors $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This is called Dirac (or Bra–Ket) notation. The conjugate transpose $|\phi\rangle^*$ of vector $|\phi\rangle$ is written as $\langle \phi |$. The inner product of two vectors $|\phi\rangle$ and $|\psi\rangle$ is given by $\langle \phi | \psi \rangle$ or $\langle \phi | \psi \rangle$ and the Euclidean norm of a vector $|\phi\rangle$ is $\| |\phi\rangle \| = \sqrt{\langle \phi | \phi \rangle}$.

A qubit is a unit-length vector in this space, and we can write it as a linear combination of the basis states $\alpha_0 |0\rangle + \alpha_1 |1\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}$, where $|\alpha_0|^2 + |\alpha_1|^2 = 1$. The complex numbers $\alpha_0$ and $\alpha_1$ are called amplitudes. One can think of the qubit as being in both basis states at the same time, the qubit is in a superposition of basis states $|0\rangle$ and $|1\rangle$.

One qubit “lives” in the vector space $\mathbb{C}^2$, but an $n$-qubit system “lives” in the $2^n$-dimensional tensor space $V = \mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2$. For example, a 2-qubit system has 4 basis states: $|0\rangle \otimes |0\rangle$, $|0\rangle \otimes |1\rangle$, $|1\rangle \otimes |0\rangle$ and $|1\rangle \otimes |1\rangle$. If the system is in state $|1\rangle \otimes |0\rangle$, the first qubit is in state $|1\rangle$ and the second qubit is in state $|0\rangle$. This is often abbreviated as $|1\rangle |0\rangle$, $|1,0\rangle$ or even $|10\rangle$.

An $n$-qubit quantum register can be in any state of the form $\sum_{i \in \{0,1\}^n} \alpha_i |i\rangle$ where $\sum_{i \in \{0,1\}^n} |\alpha_i|^2 = 1$.

Note that $i$ is a bitstring which is also used as an index. We will often view bitstrings as decimal numbers. This can also happen with basis states. A basis state is of the form $|b_1 b_2 \ldots b_n\rangle$ with $b_i \in \{0,1\}$. We can also write the basis states
as $|0\rangle, |1\rangle, |2\rangle, \ldots, |2^n - 1\rangle$. The sum above is therefore the same as
\[ \sum_{i=0}^{2^n-1} \alpha_i |i\rangle. \]

A mixed state $\{p_i, |\phi_i\rangle\}$ is a classical probability distribution over pure quantum states. The system is in state $|\phi_i\rangle$ with probability $p_i$. We can represent a mixed quantum state with a density matrix $\rho = \sum_i p_i |\phi_i\rangle \langle \phi_i|$. This $\rho$ is a positive semidefinite operator with trace 1 (the sum of the diagonal entries is 1). In particular the density matrix of a pure state $|\phi\rangle$ is $\rho = |\phi\rangle \langle \phi|$. Sometimes we have a state where Alice has some of the qubits and Bob holds the rest. For this total state we can write down the density matrix, but we would also like to write down the density matrix to describe Alice’s local state. We can do this by tracing out Bob’s part. For a matrix $A \otimes B$, $Tr_B(A \otimes B) = A \cdot Tr(B)$. This can be extended linearly to all matrices that are not of product form. Now, for a bipartite state $\rho_{AB}$ Alice’s local density matrix is given by $\rho_A = Tr_B(\rho_{AB})$.

### 4.1.2 Measurements

One thing we can do with a quantum state is measuring it. Consider an $n$-qubit system. The most basic form of measurement we can do is measuring in the standard basis. For a quantum state $\sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ this means that if we observe the state we will “see” a basis states $|i\rangle$ with probability $|\alpha_i|^2$. If we measure a state the superposition will collapse to some observed state $|j\rangle$. All the information that was contained in the state is gone and only $|j\rangle$ is left.

A little more general type of measurement is a projective measurement. We just measured in the standard basis, but we can measure in any orthogonal basis. A projective measurement is described by projectors $P_1, \ldots, P_m (m \leq 2^n)$ with $\sum_{j=1}^m P_j = I$ and $P_i P_j = 0$ for $i \neq j$. Every projector $P_j$ projects on some subspace $V_j$ of our total space $V$. These spaces are pairwise orthogonal and every state $|\phi\rangle$ can be decomposed in a unique way as sum of orthogonal states $|\phi\rangle = \sum_{j=1}^m |\phi_j\rangle$ where $|\phi_j\rangle = P_j |\phi\rangle \in V_j$. When we apply this measurement to the state $|\phi\rangle$ we will measure $P_j |\phi\rangle$ with probability $\|P_j |\phi\rangle\|^2 = tr(P_j |\phi\rangle \langle \phi|)$. The state will then also collapse to $\frac{P_j |\phi\rangle}{\|P_j |\phi\rangle\|}$ (the normalisation is needed to make sure we end up with a unit-length vector). Note that the measurement in the standard basis is a projective measurement consisting of the $2^n$ matrices $P_j = |j\rangle \langle j|$. 

30
The most general type of measurement we can do, is instead of using pairwise orthogonal projectors, using positive semidefinite matrices which sum up to the identity matrix. For a mixed quantum state given by the density matrix $\rho$ and positive semidefinite operators $E_i = M_i^* M_i$ ($i \leq n$) subject to $\sum_i E_i = I$ this means that the probability of observing the $i$th outcome is given by $p_i = tr(E_i \rho) = \frac{tr(M_i \rho M_i^*)}{tr(M_i \rho M_i^*)}$. If the measurement yields outcome $i$ the state collapses to $M_i \rho M_i^*$.

An important property of qubits or quantum mechanics in general is entanglement. This refers to correlations between different qubits. Multiple qubits are entangled if their state cannot be written as a tensor product of separate qubits. There is, for example, the EPR-pair $\frac{1}{\sqrt{2}} (|00\rangle + |11\rangle)$.

Suppose we measure the first qubit in the standard basis and we end up with a $|0\rangle$ the whole state collapses to $|00\rangle$. We have also collapsed the second qubit without observing it.

### 4.1.3 Evolution

Besides measuring a quantum state we can also apply some operation to it.

$$|\phi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle + \ldots + \alpha_{2^n-1} |2^n - 1\rangle \mapsto \beta_0 |0\rangle + \beta_1 |1\rangle + \ldots + \beta_{2^n-1} |2^n - 1\rangle = |\psi\rangle$$

In quantum mechanics we are only able to apply linear operations. This implies that if we view a state $|\phi\rangle$ as a $2^n$-dimensional vector $(\alpha_1 \alpha_2 \ldots \alpha_n)^T$ an operation corresponds to some $2^n \times 2^n$-dimensional matrix $U$.

$$U \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_{2^n-1} \end{pmatrix}.$$ 

Note that $|\psi\rangle = U|\phi\rangle = U(\sum_{i=0}^{2^n-1} \alpha_i |i\rangle) = \sum_{i=0}^{2^n-1} \alpha_i U |i\rangle$ by linearity.

After applying the operation we need to end up with a unit vector. This means the operation $U$ must preserve the norm of vectors and therefore is a unitary transformation. Since every unitary is invertible (the inverse is the conjugate transpose) every operation has an “undo” operation. This as opposed to a measurement which is non-reversible.

\footnote{Named after Einstein, Podolsky and Rosen who where the first to examine this phenomenon and the paradoxical properties [13].}
An example of such an operation on one qubit is the Hadamard transform, given by the matrix
\[
\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.
\]
On the basis states this results in
\[
H |0\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle
\]
\[
H |1\rangle = \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle.
\]
Something worth mentioning here is that if we apply the Hadamard transform on the state \(\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle\) we get
\[
H \left( \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \right) = \frac{1}{\sqrt{2}} H |0\rangle + \frac{1}{\sqrt{2}} H |1\rangle
\]
\[
= \frac{1}{2} |0\rangle + \frac{1}{2} |1\rangle + \frac{1}{2} |0\rangle - \frac{1}{2} |1\rangle
\]
\[
= |0\rangle.
\]
The positive and negative amplitude of \(|1\rangle\) cancel each other out. This phenomenon is called interference.

4.1.4 Queries

A query to an \(m\)-bit string \(y\) is the following unitary transformation, where \(j \in [m]\) and \(b \in \{0, 1\}\) is the target bit:
\[
|j\rangle |b\rangle \mapsto |j\rangle |b \oplus y_j\rangle.
\]
By setting the target bit to \(\frac{1}{\sqrt{2}}( |0\rangle - |1\rangle)\) we get a so-called phase query
\[
|j\rangle \mapsto (-1)^{y_j} |j\rangle.
\]
It is also possible to add an extra control bit \(c\), which controls whether the phase is added or not.
\[
|c\rangle |j\rangle \mapsto (-1)^{c y_j} |c\rangle |j\rangle.
\]

4.2 Locally Decodable Quantum Codes and Quantum Private Information Retrieval

The definitions of Locally Decodable Quantum Codes (LDQCs) and Quantum Private Information Retrieval (QPIR) are basically the same as in the classical setting. The only difference is that now, qubits are allowed as communication.
Definition 37 (Locally Decodable Quantum Code (LDQC)). A \((q, \delta, \epsilon)\)-Locally Decodable Code is a map \(C : \Gamma^n \to \Sigma^N\) together with a randomised quantum decoding algorithm \(A\) such that

- \(A\) makes at most \(q\) queries to a string \(y \in \Sigma^N\).
- For all \(x \in \Gamma^n\) with Hamming distance \(d(C(x), y) \leq \delta N\) it holds that
  \[
  \mathbb{P}[A^y(i) = x_i] \geq \frac{1}{2} + \epsilon \text{ for all } i \in [n],
  \]
  where the probability is taken over the randomness of \(A\), including the randomness generated by its measurements.

Definition 38 (Quantum Private Information Retrieval scheme). A one-round 1-secure \(k\)-server Quantum Private Information Retrieval scheme with recovery probability \(\frac{1}{2} + \epsilon\), query size \(t\), answer size \(a\), for an \(n\)-bit database \(x\), consists of \(k + 1\) quantum algorithms (the user and \(k\) servers), such that

- For some input \(i \in [n]\), the user creates some quantum state \(|\phi\rangle\) consisting of \(k + 1\) registers. She sends register \(j\) to server \(j\) for all \(j \in [k]\) and keeps register \(k + 1\) to herself. Each server applies some transformation depending on \(x\) to its register and sends it back to the user.
- The user outputs a guess \(b\) for \(x_i\) based on \(|\Phi\rangle\) and \(i\). The probability that \(b = x_i\) is at least \(\frac{1}{2} + \epsilon\) for all \(x, i\).
- The local density matrix of the state received by each server is independent of \(i\).

We can capture any randomness we need in a superposition. For example, if we want to choose whether we use the state \(|\varphi\rangle\) or \(|\psi\rangle\) with equal probability. In classical PIR we have to randomly select one of the two before executing the scheme. In quantum PIR, we can use the register \(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\) by creating the state \(\frac{1}{\sqrt{2}}(|0\rangle |\varphi\rangle + |1\rangle |\psi\rangle)\). We can apply all the transformations on the second register and at the end we can measure the first register to randomly choose between \(|\varphi\rangle\) and \(|\psi\rangle\).

Still, in the remainder of this thesis we present schemes where we do randomly select an option before executing the scheme. This is done to avoid extra notation which could make understanding the schemes unnecessarily difficult.
4.3 Classical to Quantum

In this section we present a way to convert classical LDCs and PIRs into LDQCs and QPIRs. Especially in the PIR setting this results in a great improvement over the classical case.

For the conversion we need the following lemma due to Kerenidis and de Wolf [21].

**Lemma 39.** Let $f : \{0, 1\}^2 \to \{0, 1\}$ and suppose we can make queries to the bits of the input $a = a_0a_1$. Then there exists a quantum algorithm that makes only one query (independent of $f$) and outputs $f(a)$ with probability exactly $\frac{11}{14}$.

**Proof.** Suppose we could construct the state

$$|\psi_a\rangle = \frac{1}{2}((-1)^{a_0+a_1}|00\rangle + |01\rangle + (-1)^{a_0}|10\rangle + (-1)^{a_1}|11\rangle)$$

with one quantum query. Then we would be able to determine $a$ with certainty since the states $|\phi_b\rangle$ with $b \in \{0, 1\}^2$ form an orthonormal basis. Because it is not possible to construct the state above with one query, we will approximate the state by

$$|\phi\rangle = \frac{1}{\sqrt{3}}(|01\rangle + (-1)^{a_0}|10\rangle + (-1)^{a_1}|11\rangle).$$

This is done by querying $a$ using

$$\frac{1}{\sqrt{3}}(|01\rangle + |10\rangle + |11\rangle)$$

where the first bit is the control bit, and the phase $(-1)^{a_j}$ is put in front of the second register $|j\rangle$ if the control bit is 1.

We can measure the resulting state in the basis consisting of the four states $|\psi_b\rangle$. The probability that we measure the correct outcome $a$ is $|\langle \phi | \psi_a \rangle|^2 = \frac{1}{3}$ and the other three outcomes all have probability $\frac{1}{12}$.

Based on $f$ and $b \in \{0, 1\}^2$, which we have obtained from our measurement $|\psi_b\rangle$, the algorithm gives some output. There are four cases for $f$:

1. $|f(a)^{-1}| = 1$ i.e., the number of inputs which are mapped to 1 by $f$ is 1. If $f(b) = 1$ the algorithm outputs 1 with probability 1. If $f(b) = 0$ the algorithm outputs 0 with probability $\frac{6}{7}$ and 1 with probability $\frac{1}{7}$.

Now, if $f(a) = 1$ the probability that the algorithm also outputs 1 is

$$\mathbb{P}[f(b) = 1] \cdot 1 + \mathbb{P}[f(b) = 0] \cdot \frac{1}{7} = \frac{3}{4} + \frac{1}{4} \cdot \frac{1}{7} = \frac{11}{14}.$$
and if \( f(a) = 0 \) the probability that the algorithm outputs 0 is
\[
P[f(b) = 0] \cdot \frac{6}{7} = \frac{11}{12} \cdot \frac{6}{7} = \frac{11}{14}
\]

2. The case \(|f(1)^{-1}| = 3\) is analogous to the case \(|f(1)^{-1}| = 1\) except 0 and 1 are reversed
3. \(|f(1)^{-1}| = 2\). Then \(P[f(a) = f(b)] = \frac{3}{4} + \frac{1}{12} = \frac{5}{6}\). If we let the algorithm output \(f(b)\) with probability \(\frac{13}{14}\) and output \(1 - f(b)\) with probability \(\frac{1}{14}\) then the probability of outputting \(f(a)\) is \(\frac{11}{14}\).

4. \(f\) is constant. In this case the algorithm outputs that value with probability \(\frac{11}{14}\).

Using this algorithm the probability of outputting \(f(a)\) is always equal to \(\frac{11}{14}\). □

We can use this lemma if we know nothing about the function \(f\). In case we know \(f\) is de XOR-function we get a better result.

**Lemma 40.** Let \( f : \{0, 1\}^2 \to \{0, 1\} \) be the function which outputs the XOR of its input \(a = a_0a_1\) i.e., \( f(a_0a_1) = a_0 \oplus a_1 \). Suppose we have query access to the input \(a\), then we can determine \(f(a)\) with certainty making only 1 query to the input \(a = a_0a_1\).

**Proof.** We use the query
\[
\frac{1}{\sqrt{2}} \left( |0\rangle + |1\rangle \right),
\]
which results in
\[
\frac{1}{\sqrt{2}} \left( (-1)^{a_0} |0\rangle + (-1)^{a_1} |1\rangle \right).
\]
Applying a Hadamard transform gives
\[
\frac{1}{2} \left( ((-1)^{a_0} + (-1)^{a_1}) |0\rangle + ((-1)^{a_0} - (-1)^{a_1}) |1\rangle \right).
\]
If we measure in the standard basis, we measure \(|0\rangle\) if \(a_0 = a_1\) and \(|1\rangle\) if \(a_0 \neq a_1\). □

Note that this is the Deutsch-Jozsa algorithm [10] for the case \(n = 2\).
We can use Lemma 39 to prove the following theorem [21].

**Theorem 41.** A \((2, \delta, \epsilon)\)-Locally Decodable Code \(C : \{0, 1\}^n \to \{0, 1\}^N\) is also a \((1, \delta, \frac{\delta}{4} \epsilon)\)-Locally Decodable Quantum Code.
Proof. Consider some \( x, y \) such that the Hamming distance \( d(C(x), y) \leq \delta N \) and fix some \( i \in [n] \). From the classical decoder we obtain 2 indices \( j, k \in [N] \) and some \( f : \{0,1\}^2 \to \{0,1\} \) such that

\[
P[f(y_j, y_k) = x_i] = p \geq \frac{1}{2} + \epsilon.
\]

Here we take the probability over the randomness of the decoder. By Lemma 39 we have a 1-query quantum decoder that outputs some bit \( b \) such that

\[
P[b = f(y_j, y_k)] = \frac{11}{14}.
\]

The probability that our decoder outputs the correct bit is\(^2\)

\[
P[b = x_i] = P[b = f(y_j, y_k)] \cdot P[f(y_j, y_k) = x_i] + P[b \neq f(y_j, y_k)] \cdot P[f(y_j, y_k) \neq x_i]
\]

\[
= \frac{11}{14}p + \frac{3}{14}(1-p)
\]

\[
= \frac{3}{14} + \frac{4}{7}p
\]

\[
\geq \frac{1}{2} + \frac{4}{7} \epsilon.
\]

\( \square \)

In case the decoder of a \( (2, \delta, \epsilon) \text{-LDC} \) outputs the XOR of its two queried bits we can give a better reduction using Lemma 40.

**Theorem 42.** A \( (2, \delta, \epsilon) \text{-Locally Decodable Code} \) where the decoder outputs the XOR of the 2 queried bits is also a \( (1, \delta, \epsilon) \text{-Locally Decodable Quantum Code} \).

For classical PIR schemes where the answers of the servers consist of 1 bit we can use the following theorem [21] to reduce the amount of servers needed.

**Theorem 43.** If there exists a classical one-round 1-secure 2-server PIR scheme with \( t \)-bit queries, 1-bit answers and recovery probability \( \frac{1}{2} + \epsilon \), then there exists a quantum one-round 1-secure 1-server PIR scheme with \( (t+2) \)-qubit queries, \( (t+2) \)-qubit answers, and recovery probability \( \frac{1}{2} + \frac{4}{7} \epsilon \).

\(^2\)Note that it is important that the algorithm from Lemma 39 computes \( f(a) \) with probability ‘exactly’ \( \frac{11}{14} \). Suppose the classical decoder outputs AND\((y_1, y_2) = x_i\) with probability \( \frac{3}{14} \) and XOR\((y_3, y_4) = 1 - x_i\) with probability \( \frac{1}{2} \). Then it outputs \( x_i \) with probability \( \frac{3}{14} \). But if the quantum algorithm computes AND\((y_1, y_2) \) with probability \( \frac{11}{14} \) and XOR\((y_3, y_4) \) with probability 1, then its recovery probability is \( \frac{3}{14} \cdot \frac{11}{14} < \frac{1}{2} \).
Proof. Since the two answers \(a_0\) and \(a_1\) are 1 bit, the user computes \(f(a_0, a_1)\) for some function \(f\) to retrieve the desired bit. Let \(q_0\) and \(q_1\) be the queries sent by the user. The user sets up the \((4 + t)\)-qubit state
\[
\frac{1}{\sqrt{3}} (|01\rangle |00, 0\rangle + |10\rangle |10, q_0\rangle + |11\rangle |11, q_1\rangle).
\]
The users keeps the first 2 bits and sends the rest to the server. Since \(|10, q_0\rangle\) does not contain any information about \(i\) and the same holds for \(|11, q_1\rangle\) the privacy is guaranteed.

Next, the server puts the phase \((-1)^{a_j}\) in front of \(|1_j, q_j\rangle\), \(j \in \{0, 1\}\) and leaves \(|00, 0\rangle\) alone. The server can use the first bit as control bit and from the second bit it knows as which classical server it has to act.

Finally the server sends everything back to the user, who can remove the last \(t + 2\) qubits and compute \(f(a_0, a_1)\) with probability \(\frac{11}{14}\) from
\[
\frac{1}{\sqrt{3}} (|01\rangle + (-1)^{a_0} |10\rangle + (-1)^{a_1} |11\rangle)
\]
just like in Theorem 41, which gives us an overall recovery probability of \(\frac{1}{2} + \frac{\epsilon}{4}\).

Again if the desired bit is the XOR of the 2 queried bits, we can use Theorem 42 to obtain a PIR scheme with better recovery.

**Theorem 44.** If there exists a classical one-round 1-secure 2-server PIR scheme with \(t\)-bit queries, 1-bit answers and recovery probability \(\frac{1}{2} + \epsilon\) where the desired bit is the XOR of the 2 queried bits, then there exists a quantum one-round 1-secure 1-server PIR scheme with \((t + 1)\)-qubit queries, \((t + 1)\)-qubit answers, and recovery probability \(\frac{1}{2} + \epsilon\).

### 4.4 Multiple-Bit Answers

Theorems 43 and 44 only give conversions for classical PIR schemes where the answers consist of 1 bit. If we consider PIR schemes where the answers consist of \(a\) bits, we cannot use a trivial conversion. Wehner and de Wolf [31] show in their article that given a state \(\frac{1}{\sqrt{2}}(|0, a_0\rangle + |1, a_1\rangle)\), with \(a_0\) and \(a_1\) \(a\)-bit strings, we can compute any boolean function \(f(a_0, a_1)\) with success probability \(\frac{1}{2} + \frac{1}{2^{2a+1}}\) and that this is optimal for the XOR function.

To show that we can compute \(f(a_0, a_1)\) with probability \(\frac{1}{2} + \frac{1}{2^{2a+1}}\) for any Boolean function Wehner and de Wolf [31] first prove the following lemma.

**Lemma 45.** For every \(f : \{0, 1\}^{2a} \rightarrow \{0, 1\}\) there exist non-normalized states \(|\phi_b\rangle\) for all \(b \in \{0, 1\}^a\) such that \(U : |b\rangle |0\rangle \rightarrow \frac{1}{2^a} \sum_{w \in \{0, 1\}^a} (-1)^{f(w, b)} |w\rangle |0\rangle + |\phi_b\rangle |1\rangle\) is unitary.
Theorem 46. Let \( f : \{0,1\}^2 \rightarrow \{0,1\} \) be a Boolean function. Then there exists a quantum algorithm to compute \( f(a_0, a_1) \) with success probability exactly \( \frac{1}{2} + \frac{1}{2^{a+1}} \) from \( |\Psi_{a_0a_1}\rangle = \frac{1}{\sqrt{2}}(|0, a_0\rangle + |1, a_1\rangle) \), for \( a_0, a_1 \in \{0,1\}^a \).

Proof. First we add an extra \( |0\rangle \) to \( |\Psi_{a_0a_1}\rangle \) to obtain
\[
\frac{1}{\sqrt{2}}(|0\rangle |a_0\rangle |0\rangle + |1\rangle |a_1\rangle |0\rangle).
\]

Let \( U \) be as in the previous lemma. We can apply the transform \( |0\rangle \langle 0| \otimes I^a_1 + |1\rangle \langle 1| \otimes U \), which yields
\[
\frac{1}{\sqrt{2}} \left( |0\rangle |a_0\rangle |0\rangle + |1\rangle \left( \frac{1}{2^a} \sum_{w \in \{0,1\}^a} (-1)^{f(w,a_1)} |w\rangle |0\rangle + |\phi_{a_1}\rangle |1\rangle \right) \right).
\]

Define the states \( |\Gamma\rangle = |a_0\rangle |0\rangle \) and \( |\Lambda\rangle = \frac{1}{2^a} \sum_{w \in \{0,1\}^a} (-1)^{f(w,a_1)} |w\rangle |0\rangle + |\phi_{a_1}\rangle |1\rangle \). We can rewrite the state from 4.1 as \( \frac{1}{\sqrt{2}}(|0\rangle (|\Gamma\rangle + |\Lambda\rangle) + |1\rangle (|\Gamma\rangle - |\Lambda\rangle)) \). Applying a Hadamard transform to the first qubit results in \( \frac{1}{2}(|0\rangle (|\Gamma\rangle + |\Lambda\rangle) + |1\rangle (|\Gamma\rangle - |\Lambda\rangle)) \).

From
\[
\langle \Gamma | \Lambda \rangle = \frac{1}{2^a} \sum_{w \in \{0,1\}^a} (-1)^{f(w,a_1)} \langle 0 | a_0 | w \rangle |0\rangle \langle 0 | a_0 | \phi_{a_1} \rangle |1\rangle
\]
\[
= \frac{1}{2^a} \sum_{w \in \{0,1\}^a} (-1)^{f(w,a_1)} \langle a_0 | w \rangle |0\rangle \langle 0 | a_0 | \phi_{a_1} \rangle |1\rangle
\]
\[
= \frac{1}{2^a} (-1)^{f(a_0,a_1)},
\]
we see that the probability that the first qubit is a 0 is
\[
\frac{1}{4} \langle \Gamma + \Lambda | \Gamma + \Lambda \rangle = \frac{1}{2} + \frac{1}{2} \langle \Gamma | \Lambda \rangle = \frac{1}{2} + \frac{(-1)^{f(a_0,a_1)}}{2^{a+1}}.
\]
Thus the probability that we measure \( f(a_0, a_1) \) is always \( \frac{1}{2} + \frac{1}{2^{a+1}} \).

We will now show that this probability is optimal for the XOR function. For this proof the following lemma is needed [31]. For a square matrix \( A \), let \( \|A\|_\text{tr} \) denote the trace norm of the matrix \( A \), i.e., the sum of its singular values. The trace norm of a matrix is invariant under unitary transforms.

Lemma 47. Two density matrices \( \rho_0 \) and \( \rho_1 \) cannot be distinguished with probability better than \( \frac{1}{2} + \frac{\|\rho_0 - \rho_1\|_\text{tr}}{4} \).
Theorem 48. Let \( f : \{0,1\}^{2a} \to \{0,1\} \) be the XOR function on \( 2a \) bits. Then any quantum algorithm for computing \( f \) from one copy of \(|\Psi_{a_0a_1}\rangle = \frac{1}{\sqrt{2}} (|0, a_0\rangle + |1, a_1\rangle)\) has success probability at most \( \frac{1}{2} + \frac{\epsilon}{2a} \).

Proof. Let \( \rho_i = \frac{1}{2^{2a}} \sum_{a_0a_1 \in f^{-1}(i)} |\Psi_{a_0a_1}\rangle \langle \Psi_{a_0a_1}| \) for \( i \in \{0,1\} \). If a quantum algorithm can compute the XOR of \( a_0a_1 \) with probability \( \frac{1}{2} + \epsilon \) then it can also be used to distinguish between \( \rho_0 \) and \( \rho_1 \). From the previous lemma we know that \( \epsilon \leq \frac{\|\rho_1 - \rho_0\|_{tr}}{2a} \).

Let \( A = \rho_0 - \rho_1 \). Then
\[
A = \frac{1}{2^{2a}} \sum_{a_0a_1 \in f^{-1}(0)} |0, a_0\rangle \langle 0, a_0| + |1, a_1\rangle \langle 0, a_0| + |0, a_0\rangle \langle 1, a_1| + |1, a_1\rangle \langle 1, a_1| - \\
\frac{1}{2^{2a}} \sum_{a_0a_1 \in f^{-1}(1)} |0, a_0\rangle \langle 0, a_0| + |1, a_1\rangle \langle 0, a_0| + |0, a_0\rangle \langle 1, a_1| + |1, a_1\rangle \langle 1, a_1|
\]

Every \( |0, a_0\rangle \langle 0, a_0|\)-entry is the same for \( \rho_0 \) and \( \rho_1 \), because for some \( a_0 \) exactly half of the possibilities of \( a_1 \) will result in \( f(a_0a_1) = 0 \) and the other half results in \( f(a_0a_1) = 1 \). The same arguments holds for the entries \( |1, a_1\rangle \langle 1, a_1| \). Thus the diagonal of \( A \) consists of 0s. For every off-diagonal entry \( |0, a_0\rangle \langle 1, a_1| \) or \( |1, a_0\rangle \langle 1, a_1| \) we see that if the XOR of \( a_0a_1 \) is 0, the entry has value \( \frac{1}{2^{2a}} \) in \( \rho_0 \) and 0 in \( \rho_1 \). If the XOR of \( a_0a_1 \) is 1, the entry has value 0 in \( \rho_0 \) and value \( \frac{1}{2^{2a}} \) in \( \rho_1 \).

Thus in \( A \) the entry has a value of \( \frac{(-1)^{|a_0|+|a_1|}}{2^{2a}} \).

Let \( |\phi\rangle = \frac{1}{\sqrt{2^a}} \sum_{w \in \{0,1\}^a} (-1)^{|w|} |w\rangle \) then
\[
|\phi\rangle \langle \phi| = \frac{1}{2^a} \sum_{a_0, a_1 \in \{0,1\}^a} (-1)^{|a_0|+|a_1|} |a_0\rangle \langle a_1|
\]

and so we can write \( A = \frac{1}{2^a} (|0\rangle \langle 0| + |1\rangle \langle 1|) |\phi\rangle \langle \phi| + U |0\rangle \langle \phi| + U^* |\phi\rangle \langle 0| \)

Define the unitary transform \( U \) and \( V \) by
\[
U |0\rangle \langle 0| = |0\rangle \langle 0^a|, U |1\rangle \langle 0| = |1\rangle \langle 0^a|, V |0\rangle \langle 0| = |1\rangle \langle 0^a|, V^* |1\rangle \langle 0| = |0\rangle \langle 0^a|.
\]

We see
\[
UAV^* = \frac{1}{2^a} (U |0\rangle \langle \phi| + U |1\rangle \langle \phi|) \langle V^* + U |1\rangle \langle \phi| V^* \rangle = \frac{1}{2^a} (|0\rangle \langle 0^a| + |1\rangle \langle 0^a|) \langle 1\rangle \langle 0^a|).
\]

So,
\[
\|\rho_0 - \rho_1\|_{tr} = \|A\|_{tr} = \|UAV^*\|_{tr} = \frac{2}{2^a},
\]

39
which implies
\[ \epsilon \leq \frac{\| \rho_0 - \rho_1 \|_\text{tr}}{4} = \frac{1}{2^{a+1}}. \]

### 4.5 Quantum Private Information Retrieval Schemes

The Private Information Retrieval schemes we have obtained from the Locally Decodable Codes of Efremenko have answer size 1 and the desired bit is obtained by computing the XOR of the received bits. When allowing quantum bits, Theorem 44 enables us to use only half the number of servers needed in the classical case.

**Corollary 49.** For any positive integer \( r \) there exists a one-round 1-secure 2\(^{r-1}\)-server Quantum Private Information Retrieval scheme with query size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)^{r-1}}
\]
answer size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)^{r-1}}
\]
and recovery probability 1.

**Proof.** Corollary 35 gives us a 2\(^r\)-server classical PIR scheme with recovery probability 1, query size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)^{r-1}}
\]
answer size 1. We make pairs of 2 servers, each pair can be converted into 1 quantum server which gives the XOR of the 2 bits which should have been sent by the 2 classical servers, using Theorem 44. Each server receives a query of size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)^{r-1}}
\]
and sends back an answer of size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)^{r-1}}
\]. Computing the XOR of the 2\(^{r-1}\) received bits, yields the desired bit. Since we only have to calculate the XOR, we can make sure the recovery probability of 1 is maintained.

In Section 3.2 we saw the best known classical 2-server Private Information Retrieval scheme which has communication complexity of \( \mathcal{O}(n^{1.5}) \). There are a lot of schemes which achieve the same communication complexity, but none of them improve the scheme. The case \( r = 2 \) from the previous corollary does achieve a much lower communication complexity for a quantum Private Information Retrieval scheme. This result gives a large separation between the case where we allow classical bits and the case where we allow quantum bits as communication.

**Corollary 50.** There exists a one-round 1-secure 2-server Quantum Private Information Retrieval scheme with query size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)}
\]
answer size
\[
n^{\mathcal{O}\left( \sqrt{\log \frac{n}{\log n}} \right)}
\]
and recovery probability 1.
5 Single-Server Interactive Quantum Private Information Retrieval

5.1 Multi-round Single-Server Quantum Private Information Retrieval

The setting in which there are multiple non-communicating servers is not very ideal. But if we consider just one server, it was already proven in [9] that in classical PIR linear communication is necessary, making the protocol of requesting the whole database essentially optimal. Also in quantum PIR where we allow only one query and one answer the trivial protocol is optimal [26]. We can naturally extend the definition of a one-round 1-server Quantum Private Information Retrieval schemes to a definition for a multiple-round 1-server Quantum Private Information Retrieval Scheme.

Definition 5.1 (Multi-Round 1-Server Weakly 1-Secure Quantum Private Information Retrieval scheme). An $m$-round weakly 1-secure 1-server Quantum Private Information Retrieval scheme with recovery probability $\frac{1}{2} + \epsilon$ for an $n$-bit database $x$, consists of a starting state $|\varphi_0\rangle = |0^u\rangle_U |0^c\rangle_C |0^s\rangle_S$ for non-negative integers $u, c, s$ where registers $U$ and $C$ are in hands of the user and register $S$ is in hands of the server and two quantum algorithms (the user and the server), such that

- For some input $i \in [n]$ the user generates $m$ unitaries $V_1^i, V_2^i, \ldots, V_m^i$ acting on registers $U$ and $C$. Based on $x$, the server generates $m$ unitaries $S_1, S_2, \ldots, S_m$ acting on registers $C$ and $S$.

- First, the user constructs the state $|\varphi_0\rangle$. During round $k \in [m]$ the user applies $V_1^i$ to registers $U$ and $C$. She sends register $C$ to the server which first applies $S_k$ to registers $C$ and $S$ and then sends register $C$ back to the user.

  Note that the total state after round $k$ is $|\varphi_k\rangle = \prod_{j=1}^k (I \otimes S_j)(V_j^i \otimes I) |\varphi_0\rangle$.

- The user outputs a guess $b$ for $x_i$ based on $|\varphi_m\rangle$ and $i$. The probability that $b = x_i$ is at least $\frac{1}{2} + \epsilon$ for all $x, i$. 

41
Let $|\psi_k\rangle = (V_k^i \otimes I) |\varphi_{k-1}\rangle$ for all $k \in [m]$. For every round $k \in [m]$ the local density matrix of the server $\text{Tr}_U (|\psi_k\rangle \langle \psi_k|)$ is independent of $i$.

The total communication complexity of this scheme is $2 \cdot m \cdot c$ qubits.

Here we use “weakly” to indicate that the scheme is only secure when the server exactly follows the protocol. It may be possible for the servers to obtain information about $i$ when deviating from the protocol. In this setting Le Gall [14] presented a weakly 1-secure 1-server QPIR scheme which uses 3 messages and has communication complexity $O(n^{\frac{1}{2}})$. In his article Le Gall also showed how the server can manipulate the protocol to get information about $i$. Very recently Baumeler and Broadbent [3] proved that any multi-round scheme which protects the user against a server deviating from the protocol has communication complexity linear in the size of the database. Thus also in the quantum setting requesting the whole database is optimal.

Still the case where we assume the server acts honestly is interesting because it gives such an improvement in the communication complexity. We will present a way to convert certain classical 2-server PIR schemes into quantum 1-server PIR schemes. If we take the square scheme as in Example 22 this will result in a scheme with a communication complexity of $O(n^{\frac{1}{2}})$ just like Le Gall achieves with his scheme. But if we consider the scheme from Example 26 we improve Le Gall’s result with a scheme that has communication complexity $O(n^{\frac{1}{3}})$.

5.2 Generalizing and improving Le Gall’s scheme

The scheme from Le Gall [14] reminds us of the square scheme as given in Example 22. Since in the case of 2 classical servers there also exists a scheme with $O(n^{\frac{1}{2}})$ bits of communication, it is a natural question to ask whether we can also convert this scheme into a quantum 1-server scheme. The following theorem implies this is indeed the case. The theorem is even more general in the sense that every scheme that has certain properties can be converted into a quantum 1-server scheme. An improvement in the classical 2-server case will, if the theorem applies, improve the quantum 1-server case.

**Theorem 52.** If there exists a 2-server one-round 1-secure classical Private Information Retrieval scheme for an $n$-bit database $x$ with recovery probability $\frac{1}{2} + \epsilon$ and the retrieval of bit $x_i$ that consists of the following steps.

1. The user generates 2 $t$-bit queries $q_1$ and $q_2$.
2. The user sends $q_1$ to server 1 and $q_2$ to server 2.
3. Sever $j$ responds with some a-bit answer $a_{q_j}$ for $j \in \{1, 2\}$.

4. The user retrieves $x_i$ by computing $x_i = f(a_{q_1}) \oplus g(a_{q_2})$ for certain functions $f, g : \{0, 1\}^s \rightarrow \{0, 1\}$.

Then there exists a two-round weakly 1-secure 1-server Quantum Private Information Retrieval scheme with recovery probability $\frac{1}{2} + \epsilon$ which uses $4 \cdot (t + a)$ qubits of communication.

Note that the behaviour of the user is randomised to secure the privacy: the queries $q_1$ and $q_2$ and functions $f$ and $g$ are based on random “coin flips” done by the user.

Proof. Let $x$ be the $n$-bit database. Suppose the user wants to know bit $x_i$. From the classical 2-server scheme the user (randomly) obtains 2 $t$-bit queries $q_1$ and $q_2$.

First, define the state $|\varphi_0\rangle = |0\rangle_U |0^t\rangle_{C_1} |0^a\rangle_{C_2}$ and let $a_z$ be the answer to query $z$ in the classical 2-server scheme for all $z \in \{0, 1\}^t$.

In our quantum scheme the following set of gates will be used. The names of the registers correspond to the names of the registers of the state the gate will be applied to.

$$V_1^i = \frac{1}{\sqrt{2}} \left( |0\rangle_U |q_1\rangle_{C_1} |0\rangle_U \langle 0^t|_{C_1} + |1\rangle_U |q_2\rangle_{C_1} |0\rangle_U \langle 0^t|_{C_1} \right),$$

$$S_1 = \sum_{z \in \{0, 1\}^t} \sum_{w \in \{0, 1\}^n} |z\rangle_{C_1} |a_x \oplus w\rangle_{C_2} \langle z|_{C_1} \langle w|_{C_2},$$

$$V_2^i = a'_{z \in \{0, 1\}^t} \sum_{w, z \in \{0, 1\}^n} (-1)^{f(a') z} |0\rangle_U |q_1\rangle_{C_1} |a'\rangle_{C_2} |0\rangle_U \langle q_1|_{C_1} \langle a'|_{C_2} +$$

$$S_2 = S^{-1}$$

To retrieve the bit, the user uses the following gates

$$R = \sum_{z \in \{0, 1\}^t} |0\rangle_U |z \oplus q_1\rangle_{C_1} |0\rangle_U \langle z|_{C_1} + |1\rangle_U |z \oplus q_2\rangle_{C_1} |1\rangle_U \langle z|_{C_1},$$

$$H = \frac{1}{\sqrt{2}} \sum_{w, z \in \{0, 1\}^n} (-1)^{w \cdot z} |w\rangle_U \langle z|_U.$$
in front of $|0\rangle$ and $|1\rangle$. The gate $S_2$ “removes” the answer. Using gate $R$ register $C_1$ will be put to $|0^i\rangle$. Finally, gate $H$ is the Hadamard transform on one qubit.

We are now able to write down the scheme step-by-step.

**The Scheme:**

1. The user constructs the state $|\varphi_0\rangle$ and applies $V_1^i$. Then she sends registers $C_1$ and $C_2$ to the server.

2. The server applies $S_1$ and sends registers $C_1$ and $C_2$ back to the user.

3. The user applies $V_2^i$ and sends registers $C_1$ and $C_2$ to the server.

4. The server applies $S_2$ and sends register $C_1$ and $C_2$ back to the user.

5. The user first applies $R$ and then $H$. Measuring the first register yields the desired bit.

First, we will show that when the scheme is executed properly, the user retrieves the correct bit $x_i$. We will do this by executing the protocol and giving the joint step after each step.

**Correctness:**

1. The user sets up the following state using gate $V_1^i$

$$\frac{1}{\sqrt{2}} \left( |0\rangle_U |q_1\rangle_{C_1} |0^a\rangle_{C_2} + |1\rangle_U |q_2\rangle_{C_1} |0^a\rangle_{C_2} \right).$$

2. After the server has applied $S_1$ the total state is given by:

$$\frac{1}{\sqrt{2}} \left( |0\rangle_U |q_1\rangle_{C_1} |a_{q_1}\rangle_{C_2} + |1\rangle_U |q_2\rangle_{C_1} |a_{q_2}\rangle_{C_2} \right).$$

3. The user applies $V_2^i$ which results in

$$\frac{1}{\sqrt{2}} \left( (-1)^f(a_{q_1}) |0\rangle_U |q_1\rangle_{C_1} |a_{q_1}\rangle_{C_2} + (-1)^g(a_{q_2}) |1\rangle_U |q_2\rangle_{C_1} |a_{q_2}\rangle_{C_2} \right).$$

4. The server applies $S_2$ to put register $C_2$ to $|0^a\rangle$

$$\frac{1}{\sqrt{2}} \left( (-1)^f(a_{q_1}) |0\rangle_U |q_1\rangle_{C_1} |0^a\rangle_{C_2} + (-1)^g(a_{q_2}) |1\rangle_U |q_2\rangle_{C_1} |0^a\rangle_{C_2} \right).$$
5. The user is now able to learn \( x_i \). First she applies 
\[
R(\cdot - 1) f(aq_1) |0\rangle_U |0^a\rangle_{C_1} |0^a\rangle_{C_2} + (-1)^{g(aq_2)} |1\rangle_U |0^a\rangle_{C_1} |0^a\rangle_{C_2}
\]
\[
= \frac{1}{\sqrt{2}} \left( (-1)^{f(aq_1)} |0\rangle_U + (-1)^{g(aq_2)} |1\rangle_U \right) \left( |0^a\rangle_{C_1} |0^a\rangle_{C_2} \right).
\]

Applying a Hadamard transform on the first qubit leaves the following state
\[
\frac{1}{2} \left( \left( (-1)^{f(aq_1)} + (-1)^{g(aq_2)} \right) |0\rangle_U + \left( (-1)^{f(aq_1)} - (-1)^{g(aq_2)} \right) |1\rangle_U \right) \otimes \left( |0^a\rangle_{C_1} |0^a\rangle_{C_2} \right).
\]

Measuring the first qubit yields \( |0\rangle \) when the XOR of \( f(aq_1) \) and \( g(aq_2) \) is 0 and \( |1\rangle \) if the XOR is 1.

Secondly, we will show that when the scheme is executed properly, the server has no chance of retrieving information about \( i \).

**Privacy:** Queries \( q_1 \) and \( q_2 \) independently do not give away any information about \( i \) since they are generated by the classical 2-server PIR scheme. After step 1 and 3 the server holds registers 2 and 3. After step 1 the server has the following local density matrix
\[
\frac{1}{2} \left( |q_1\rangle_{C_1} |0^a\rangle_{C_2} \langle q_1|_{C_1} \langle 0^a|_{C_2} + |q_1\rangle_{C_1} |0^a\rangle_{C_2} \langle q_2|_{C_1} \langle 0^a|_{C_2} + |q_2\rangle_{C_1} |0^a\rangle_{C_2} \langle q_1|_{C_1} \langle 0^a|_{C_2} + |q_2\rangle_{C_1} |0^a\rangle_{C_2} \langle q_2|_{C_1} \langle 0^a|_{C_2} \right).
\]

and after step 3 it holds
\[
\frac{1}{2} \left( |q_1\rangle_{C_1} |aq_1\rangle_{C_2} \langle q_1|_{C_1} \langle aq_1|_{C_2} + |q_1\rangle_{C_1} |aq_1\rangle_{C_2} \langle q_2|_{C_1} \langle aq_2|_{C_2} + |q_2\rangle_{C_1} |aq_2\rangle_{C_2} \langle q_1|_{C_1} \langle aq_1|_{C_2} + |q_2\rangle_{C_1} |aq_2\rangle_{C_2} \langle q_2|_{C_1} \langle aq_2|_{C_2} \right).
\]

These density matrices do not depend on \( i \) so the server cannot extract information about \( i \).

Using Examples 22 and 26 we can use this theorem to create 1-server QPIR schemes. Example 22 gives us the same communication complexity as Le Gall achieves in his scheme [14], albeit with an extra message.
Corollary 53. There exists a two-round weakly 1-secure 1-server Quantum Private Information Retrieval Scheme with recovery probability 1 for an $n$-bit database with $O(n^{1/2})$ qubits of communication.

The majority of classical PIR schemes is of the form as described in Theorem 52. The conversion, as described in the theorem, is therefore a great tool to construct 1-server quantum PIR schemes from 2-server classical PIR schemes. Example 26 enables us to improve the scheme of Le Gall, and when better schemes are constructed for the classical 2-server case we most probably also improve the quantum 1-server case.

Corollary 54. There exists a two-round weakly 1-secure 1-server Quantum Private Information Retrieval Scheme with recovery probability 1 for an $n$-bit database with $O(n^{1/3})$ qubits of communication.

5.3 Insecurity when the server does not follow the protocol

Due to Baumeler and Broadbent [3] we know that a scheme which cannot be manipulated by the server to obtain information about $i$ uses $\Omega(n)$ qubits of communication. Thus, there is a way for the server to manipulate the scheme given in Theorem 52. To do this, the server should replace gate $S_1$ with some other action, such that after the user applies $V_{i_2}$ in step 3, the server receives some state which is not independent of $i$.

Consider for example the square scheme from Example 22. In this scheme we arrange the database as a $\sqrt{n} \times \sqrt{n}$ matrix and bit $x_i$ is uniquely determined by coordinates $(i_1, i_2)$. If we use the theorem to convert this into a 1-server quantum scheme, the functions $f$ and $g$ are the same, namely selecting the $i_2$th bit from the received answers.

To clarify the argument, assume $\sqrt{n}$ is even. Suppose during step 2 the server replaces the register $C_2$ by the register

$$\frac{1}{\sqrt{2}} \left( |0^{\sqrt{n}}\rangle + \frac{1}{\sqrt{2}} (|0^{\sqrt{n}}\rangle + |1^{\sqrt{n}}\rangle) \right).$$

Now, when the user applies gate $V_{i_2}$ register $C'_2$ is transformed into the following state

$$\frac{1}{\sqrt{2}} \left( |0^{\sqrt{n}}\rangle + (-1)^f (\frac{1}{\sqrt{2}} (|0^{\sqrt{n}}\rangle + |1^{\sqrt{n}}\rangle) \right).$$

The function $f$ selects the $i_2$th bit from the input, thus checking whether the phase has changed tells us that $i_2 \leq \frac{\sqrt{n}}{2}$ if the phase has become negative and $i_2 > \frac{\sqrt{n}}{2}$ otherwise. We have eliminated half of the possibilities for $i$, so learned one bit of information about $i$. 

46
6 Populaire Samenvatting

In de informatica heb je op allerlei momenten te maken met ruis en fouten. Om je hier tegen te wapenen kun je de informatie die overgebracht moet worden, verpakken in een codewoord, zodat als het codewoord een beetje verstoord wordt, je de informatie toch nog kunt terughalen. Soms ben je geïnteresseerd in slechts een deel van de informatie. Bij de standaard coderingstechnieken zal het hele codewoord moeten worden gelezen om de informatie terug te halen en vervolgens moet je nog gaan zoeken naar het deel wat je nodig hebt. Een oplossing zou zijn om de informatie op te delen en ieder deel apart de coderen. Een nadeel hiervan is dat als precies het codewoord, waar het deel van de informatie die jij wilt weten in staat, verstoord is, je de informatie niet meer kunt terughalen. Je kunt dan Locally Decodable Codes gebruiken. Ze staan je toe om alle informatie als één codewoord op te slaan en delen van de informatie terug te halen door slechts naar een paar plekken van het codewoord te bekijken.

In Private Information Retrieval wil een gebruiker iets uit een database weten. Ze wil er alleen voor zorgen dat de database niets te weten komt over wat zij wil weten. Het doel is om dit te doen met zo min mogelijk communicatie. In het geval dat de database op één server staat, is de enige mogelijkheid om de gehele database op te vragen en vervolgens te zoeken daar het ding wat je wil weten. Als de database nu wordt gerepliceerd over meerdere servers die verder niet met elkaar communiceren zijn er protocollen waardoor je het met een stuk minder communicatie kan doen.


We kunnen niet alleen kijken naar het geval waar we klassieke computers gebruiken, maar we kunnen ook kijken wat er gebeurt als we quantum computers toestaan. Een klassieke computer maakt gebruik van bits, deze zijn 0 of 1, aan of uit. Een quantum computer maakt gebruik van quantum bits of qubits, deze kunnen in een superpositie van 0 en 1 zijn, waar je over kunt nadenken als een beetje
0 maar ook een beetje 1. Dit geeft veel meer mogelijkheden qua berekeningen die we kunnen doen. Een nadeel van quantum rekenen is dat we zo'n superpositie niet kunnen “zien”. Op het moment dat we de qubit observeren zal de qubit vervallen in een 0 of een 1. Het is niet bewezen dat quantum computers sneller zijn dan klassieke computers. Toch zijn er toepassingen waar de best bekende quantum algoritmes een stuk efficiënter zijn dan de best bekende klassieke algoritmes. In het geval van Private Information Retrieval zullen we zien dat als we qubits toestaan er minder communicatie nodig is.
Bibliography


