UNIVERSITEIT UTRECHT

MASTER'S THESIS

On secret sharing-based classical and quantum multi-party computation

Author:
Jan Gerrit HÖLTING

Student ID:
6570518

Supervisors:
Thomas ATTEMA (TNO),
Thijs VEUGEN (TNO),
Tristan VAN LEEUWEN (UU),
Ivan KRYVEN (UU)

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

at the

Department of Mathematics

as part of a graduation internship at the

Cyber Security and Robustness Department
TNO

December 29, 2020

Declaration of Authorship

I, Jan Gerrit Hölting, declare that this thesis, titled "On secret sharing-based classical and quantum multi-party computation", and the work presented in it are my own. I confirm that:

- None of this thesis has previously been submitted for a degree or any other qualification at Utrecht University or any other institution.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Jan Gerrit Hölting

Date: December 29, 2020

Abstract

Secure multi-party computation (MPC) is a cryptographic primitive that lets a number N of mutually distrustful parties compute a function $f(x_1, \ldots, x_N)$ on their private inputs x_i such that at the end of the MPC protocol, each honest party obtains the function's correct output and no adversary controlling a certain subset of parties learns anything about the honest parties' inputs beyond what can be inferred from the function's output value. In 2002, [Smi01] and [CGS02] introduced the notion of multi-party quantum computation (MPQC) in which arbitrary quantum circuits can be evaluated in a distributed manner, secure against a quantum adversary. Few protocols for MPQC are known. We present a fundamental study of the building blocks of MPC and MPQC in the information-theoretic setting with a focus on protocols built on top of (quantum) secret sharing schemes. In particular, we point out structural similarities and differences and compile an extensive list of theoretical feasibility results for the maximum number of corrupted parties within classical and quantum secret sharing and MPC protocols.

Acknowledgements

I would like to thank all of my supervisors for their help, guidance, and patience. In particular, I would like to thank Tristan van Leeuwen and Ivan Kryven for taking on the challenge of supervising a graduation internship outside of their usual area of research. I am especially grateful to Thomas Attema and Thijs Veugen, as well as my other former colleagues and fellow thesis interns at the TNO CSR department, for the many hours spent (virtually) on discussing the vast amount of literature that found its way into this thesis, and the enjoyable coffee breaks inbetween.

Equally, if not more importantly, I would like to thank my family and my girlfriend, Anne, for their emotional support and encouragement throughout this year.

Contents

Declaration of Authorship **i**

Abstract **ii**

Acknowledgements **iii**

Introduction **1**

1 Preliminaries **5**
- 1.1 Quantum information and computation 5

2 Secret sharing and error correction **16**
- 2.1 Secret sharing schemes 17
- 2.2 Error correcting codes 21
- 2.4 Constructing secret sharing schemes from ECCs 24
- 2.5 Quantum analogues 27
  - 2.5.1 Quantum error-correcting codes 27
  - 2.5.2 Quantum secret sharing 32
  - 2.5.3 QSS and QEC 34
- 2.6 Feasibility of (quantum) secret sharing 36

3 Multi-party quantum computation **47**
- 3.1 Universal computation 47
- 3.2 Fault-tolerant quantum computation 50
- 3.3 On VQSS-based MPQC protocols 53
- 3.4 Theoretical feasibility of MP(Q)C 68

4 Discussion and open questions **72**

Appendices **74**

A Appendix **75**

List of Figures

3.1 Gate teleportation circuit for the T gate 52
3.2 Transversal CNOT 60
3.3 Two-layered VQSS encoding scheme. 63
3.4 Magic state verification circuit 65

List of Tables

2.1 Threshold parameters for feasibility results 38
2.2 Adversary models for different types of secret sharing schemes 43
2.3 Theoretical feasibility of classical secret sharing schemes 43
2.4 Theoretical feasibility of classical and quantum secret sharing schemes . . 46

3.1 Comparison of ingredients for two different MPQC protocols 56
3.2 Steane's 7 qubit code 59
3.3 Theoretical feasibility of classical MPC 69
3.4 Theoretical feasibility of MPQC 71
3.5 Tolerable active corruptions in existing work on MPQC 71

List of Acronyms

ECC	**E**rror-**C**orrecting **C**ode (Section 2.2)
QEC	**Q**uantum **E**rror **C**orrection (Section 2.5.1)
QECC	**Q**uantum **E**rror-**C**orrecting **C**ode (Definition 2.5.1)
CSS	**C**alderbank-**S**hor-**S**teane (Definition 3.3.3)
LSSS	**L**inear **S**ecret **S**haring **S**cheme (Section 2.1)
VSS	**V**erifiable **S**ecret **S**haring (Section 2.6)
QSS	**Q**uantum **S**ecret **S**haring (Section 2.5.2)
VQSS	**V**erifiable **Q**uantum **S**ecret **S**haring (Section 2.6)
MPC	**M**ulti-**P**arty **C**omputation (Chapter 3)
MPQC	**M**ulti-**P**arty **Q**uantum **C**omputation (Definition 3.3.1)

List of Symbols

$\bar{\alpha}$	complex conjugate of $\alpha \in \mathbb{C}$
$\mathcal{H}$	Hilbert space (over $\mathbb{C}$)
$\|\psi\rangle$	vector
$\langle\psi\|$	complex conjugate transpose of $\|\psi\rangle$
$\langle\psi\|\phi\rangle$	inner product between $\|\psi\rangle$ and $\|\phi\rangle$
$\|\phi\rangle \otimes \|\psi\rangle$	tensor product of $\|\phi\rangle$ and $\|\psi\rangle$
$\|\phi\rangle \|\psi\rangle$	abbreviation for $\|\phi\rangle \otimes \|\psi\rangle$
U^T	transpose of matrix U
$U^\dagger$	complex conjugate transpose of matrix U
$\langle\phi\| U \|\psi\rangle$	inner product between $\|\phi\rangle$ and $U\|\psi\rangle$ or, equivalently, between $\|\psi\rangle$ and $U^\dagger\|\phi\rangle$
$[s]$	vector of shares $(s_1, \ldots, s_n)$ from sharing of secret $s \in \mathbb{F}$
$[n]$	for $n \in \mathbb{N}$ this denotes the set $\{1, \ldots, n\}$
$\mathcal{T}$	set of parties $\{T_1, \ldots, T_n\}$
$\mathcal{P}(\mathcal{T})$	power set of $\mathcal{T}$
Γ	access structure, monotone subset of $\mathcal{P}(\mathcal{T})$
$\mathcal{A}$	adversary structure, antimonotone subset of $\mathcal{P}(\mathcal{T})$

Introduction

Performing analyses of datasets that are distributed over different sources is sought after in many domains. For instance, health care providers as well as patients might profit greatly from models that take into account different kinds of medical records or patient data to predict possible diseases. General privacy concerns and regulations such as the General Data Protection Regulation (GDPR), however, restrict merging or sharing data from different sources and oftentimes, no trusted third party is available that may legally perform such analyses. *Secure multi-party computation (MPC)* is the study of cryptographic protocols with multiple participants that want to jointly compute the output of some function on their private inputs without revealing their inputs to each other. Sometimes called secure function evaluation, multi-party computation was first considered formally by Yao in [Yao82], who illustrated the task at hand by introducing the famous *millionaires' problem*: two millionaires A and B both possess a certain wealth x_A and x_B and want to determine who is wealthier. They do, however, not want to reveal their exact wealth to each other. Formally, their goal is to jointly compute the function $f(x_A, x_B)$ that is 1 whenever $x_A < x_B$, and 0 otherwise, under the restriction that at the end of the protocol, none of the players know more about the other's wealth than what can be inferred from $f(x_A, x_B)$, i.e., whether the other is wealthier or not. More generally, the goal of MPC is to have a number N of players with private inputs $x_1, \dots x_N$ jointly evaluate a function $f(x_1, \dots, x_N)$ without having to reveal their inputs. In a certain sense, this deviates from traditional cryptography which is typically centered around securing point-to-point communications against external adversaries. In the MPC setting, it is usually assumed that a private and authenticated communication channel is given between participants in the protocol and one tries to prevent against attacks from participants *within* the protocol. To achieve the latter, one typically considers the worst case scenario in which a single adversary corrupts the actions of several participants in the protocol, trying to either obtain secret information, or, additionally, manipulate the outcome of the protocol to their advantage. Informally, MPC protocols aim at achieving two goals:

- *Privacy*: apart from the output value of the jointly evaluated function, the adversary learns nothing about the honest parties' inputs.
- *Correctness*: at the end of the protocol, the honest parties obtain the correct output of the function.

Various adversarial models and settings can be considered in the study of secure MPC protocols, such as restrictions on the run time of the adversary's attacks or whether their sole interest is to learn about the honest parties' inputs or whether deviations from the protocol that may change the function's output are considered possible. Two major distinctions on the setting in which MPC protocols are studied are typically made. The first one concerns the abilities of the adversary that are admitted, and one distinguishes between two scenarios:

- *Passive* (or *semi-honest*, or *honest-but-curious*) adversary: the corrupted parties all strictly follow the protocol.
- *Active* (or *malicious*) adversary: the corrupted parties can arbitrarily deviate from the protocol.

Clearly, finding secure MPC solutions against the latter type of adversary is the comparatively harder task. Modern cryptographic protocols for secure communications often tie their security to the assumed computational hardness of problems such as the prime factorisation of large integers [RSA78] or finding a discrete logarithm [DH76]. Perfectly secure solutions such as the one-time pad[1] exist but are hard to implement in the real world since they require a secure exchange of keys. Similarly, one can distinguish between different solutions for the task of multi-party computation. For some protocols, security proofs are based on computational assumptions while others do not impose any restrictions on the computational power of the adversary. We can therefore distinguish between two types of security for MPC protocols:

- *Computational security:* the adversary is limited in the amount of time and computational power.
- *Information-theoretic security:* the adversary has unlimited time and computational resources.

Cryptography in the presence of quantum adversaries

Driven by breakthrough papers in the 90s by Peter Shor, Michael Ben-Or, Dorit Aharonov and others, we can today observe global efforts in building the first large-scale quantum computers. For instance, *Shor's algorithm* ([Sho94]) can be used to efficiently solve the problem of finding the prime factorisation of large integers. A successful realisation of quantum computers would therefore break large parts of (public-key) cryptography that is being widely used at the time of writing. With regard to research in secure multi-party computation, this evokes one central question: what impact does the advent of quantum computers have on MPC solutions? More specifically, some questions that naturally arise are:

- Do quantum computers pose a similar threat to existing MPC solutions as to traditional cryptography?
- Does the use of quantum communications and computations for secure multi-party computation provide any security advantages over that of classical computers?
- Assuming an increasing use of quantum information in the foreseeable future, can we develop MPC protocols that allow for securely processing such information?

This work will be centered around the latter two questions. In particular, we will contrast the theory of *multi-party quantum computation (MPQC)* to that of classical MPC.

[1]The one-time pad is a symmetric-key encryption scheme, i.e., it requires both communicating parties to hold the same private key. The key has to be at least as long as the plaintext, and encryption (and decryption) works by performing a modular addition of plaintext (ciphertext) and the key. For example, a single bit message $m \in \{0,1\}$ is encrypted using the single-bit key k as $c := m \oplus k$. As long as the key is kept completely secret, chosen uniformly at random and is never reused, the scheme is perfectly secure.

While the latter aims at finding ways to implement arbitrary classical functions on classical inputs in the presence of a classical adversary securely and in a distributed manner, MPQC focuses on having a number of parties evaluate arbitrary *quantum* computations on quantum inputs in the presence of a quantum adversary.

Classical MPC protocols have been built on essentially three different kinds of cryptographic primitives. These are called *garbled circuits*, *homomorphic encryption* and *secret sharing*, as for example in [Yao82], [AJL+12], [GMW87], [CDvdG87], [BGW88], and [CCD88]. At the time of writing, essentially two types of approaches for MPQC protocols have been proposed: the work of [CGS02], [BCG+06] and [LRW20] focuses on information-theoretically secure MPQC, while [DNS10], [DNS12] and [DGJ+20] obtain computational security. We focus on both classical and quantum protocols that achieve information-theoretical security. In this regime, the most commonly used MPC protocols are built on top of different kinds of secret sharing schemes in a modular way: a secret sharing scheme is chosen and it is shown that using this scheme, the parties can jointly evaluate a universal set of gates securely, giving them the ability to jointly compute arbitrary functions on their inputs. The work on information-theoretically secure MPQC seems to follow a similar approach in that a *quantum secret sharing scheme* is chosen on top of which the different parties securely evaluate quantum gates. However, recent work on MPQC is still rather sparse and hardly any resources explaining the construction of MPQC protocols on a fundamental level exist.

Contribution

The goal of this work is create an extensive analysis of similarities and differences in the construction of information-theoretically secure MPC and MPQC protocols. In particular, we investigate whether MPQC protocols can be decomposed in a modular way into secret sharing schemes that allow for secure computations, such as it is known for classical MPC. To achieve this task, we will dissect available work on both classical and quantum protocols and study their building blocks on an abstract, fundamental level. We answer the question in the affirmative and argue that we can observe strong similarities in the way that MPC and MPQC protocols are constructed, but also outline several subtle differences between the two. For both classical and quantum multi-party computation we will see that choosing a suitable secret sharing scheme plays a key role in the construction of MP(Q)C protocols, and that both classical and quantum secret sharing schemes are closely related to (quantum) error-correcting codes and that many such codes can be used as secret sharing schemes. Additionally, we collect and compare feasibility results for different types of adversaries for classical and quantum protocols, essentially answering the second of the three questions outlined above. Throughout this work, we keep a focus on information-theoretically secure protocols and therefore on secret sharing schemes and its variants.

Outline

In the first, preliminary chapter, we give a brief introduction into the theory of quantum information and computation with references to more extensive introductory literature. This section gives a reader novel to the field a feeling of the way that quantum information is modelled and processed in contrast to classical information.

The second chapter will then focus on the cryptographic primitive of secret sharing that, as we noted above, is frequently used for designing MPC protocols. We will introduce secret sharing protocols both for classical and quantum information and point out the similarities and differences between the two. In particular, we study the relation between secret sharing schemes and error-correcting codes that turns out to be very fruitful in the design of quantum protocols. Furthermore, feasibility results for classical and quantum secret sharing schemes in different adversarial settings will be collected and compared.

Finally, we study the task of multi-party quantum computation in Chapter 3. After giving a formal definition of the computational model for quantum computations, we study existing work on MPQC protocols and discuss one recent proposal in detail. This chapter combines the work of all previous chapters and contains a detailed analysis of similarities and differences between MPC and MPQC protocols in terms of their design and security guarantees. We conclude with a discussion as well as a list of possible directions for further research.

1. Preliminaries

Today's computers are based on the principles of classical physics. Their basic unit of computation is a *bit* that can be in exactly one out of two possible states, 0 or 1. In a classical computation, local operations called *logical gates* are applied to a system of bits that can only be in one state at a time. The theory of modern quantum physics, however, tells us that nature behaves quite differently: in contrast to a classical system, a quantum system can be in a linear combination of multiple states, in what is also called a *superposition* of classical states. During the course of its evolution, a quantum system can exhibit so-called *interference* effects and may be *entangled* with another, spatially separated quantum system so that operations can cause "non-local" effects on the entangled system.

Quantum computation is the field of research that investigates the properties of computers that are built based on the theory of quantum mechanics. In the following, we will give an introduction into the formalism of quantum information theory in general, and quantum computing in particular, based mainly on [dW11] and [NC16]. This chapter is supposed to give the interested reader a feeling for the mechanisms at play in quantum information theory and introduce the necessary formalism for the following chapters. For an in-depth introduction into the topic, however, we refer in particular to [NC16].

First, *pure quantum states* will be introduced and we will outline the two kinds of operations that such states can be subjected to, namely *measurements* and *unitary evolution*, and discuss some examples. Hereafter, we will show how these concepts generalise to a wider class of quantum systems.

1.1 Quantum information and computation

Superposition

Consider a physical system that can be in a finite number N of mutually exclusive classical states. We start counting at zero and call these classical states $|0\rangle, \ldots, |N-1\rangle$. In our context, a "classical state" denotes a state that the system can be found in if we observe it. A *pure quantum state* $|\psi\rangle$ is a *superposition* of such classical states and we write it as

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle + \ldots + \alpha_{N-1} |N-1\rangle \,. \tag{1.1}$$

The α_i are complex values called *amplitudes* that satisfy $\sum_{i=0}^{N-1} |\alpha_i|^2 = 1$ and correspond to the probability to which the state $|\psi\rangle$ can be found in the classical state $|i\rangle$. We will later see the exact nature of that correspondence. Intuitively, one can think of $|\psi\rangle$ as being in all classical states $|i\rangle$ at the same time with amplitude α_i.

The classical states $|0\rangle, \ldots, |N-1\rangle$ form an orthonormal basis of an N-dimensional Hilbert space $\mathcal{H}$ and each pure quantum state $|\psi\rangle$ is a vector with complex entries in that

Hilbert space,

$$|\psi\rangle = (\alpha_0, \dots, \alpha_{N-1})^t. \tag{1.2}$$

The notation $|\psi\rangle$ that is used to describe a complex-valued vector is often used by physicists for linear algebra and is called *Dirac notation*. Note that we will, especially in this chapter, sometimes use the Dirac notation to describe a complex-valued vector that is not a pure quantum state, and it should be clear from the context when we assume the vector to have unit norm. A vector of amplitudes like the above, $|\psi\rangle$, is called a *ket* while its conjugate transpose, written as $\langle\psi| := |\psi\rangle^\dagger = (\bar{\alpha}_0, \dots, \bar{\alpha}_{N-1})$, is called a *bra*. The inner product of two vectors $|\psi\rangle = (\alpha_0, \dots, \alpha_{N-1})^t$ and $|\phi\rangle = (\beta_0, \dots, \beta_{N-1})^t$ in the Hilbert space spanned by the N classical states is given by their *bra-ket product* $\langle\psi| |\phi\rangle = \sum_{i=0}^{N-1} \bar{\alpha}_i \beta_i$, sometimes abbreviated as $\langle\psi|\phi\rangle$.

Note that we can find different bases for the same Hilbert space and that therefore the amplitudes corresponding to the chosen basis may vary. Consider for example some pure state $|\psi\rangle$ with some amplitudes α_0 and α_1 corresponding to the basis vectors $|0\rangle = (1,0)^t$ and $|1\rangle = (0,1)^t$, $|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$. The space $\mathbb{C}^2$ spanned by $\{|0\rangle, |1\rangle\}$ over $\mathbb{C}$ is also spanned by

$$|+\rangle := (1/\sqrt{2}, 1/\sqrt{2})^t \quad \text{and} \quad |-\rangle := (1/\sqrt{2}, -1/\sqrt{2})^t,$$

so we find that the amplitudes with respect to the different bases vary, $|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle = \beta_0 |+\rangle + \beta_1 |-\rangle$ with $\beta_0 = (\alpha_0 + \alpha_1)/\sqrt{2}$ and $\beta_1 = (\alpha_0 - \alpha_1)/\sqrt{2}$. The latter basis given by $\{|+\rangle, |-\rangle\}$ is sometimes called the *Fourier basis*.

In particular in the context of quantum cryptography with multiple parties we will often consider quantum states that reside in different Hilbert spaces. Let $\mathcal{H}_A$ and $\mathcal{H}_B$ be two Hilbert spaces spanned by the orthonormal bases $|0\rangle_A, \dots, |N-1\rangle_A$ and $|0\rangle_B, \dots, |M-1\rangle_B$, respectively. Then their tensor product $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ is the NM-dimensional space spanned by the set of states $\{|i\rangle \otimes |j\rangle : i \in \{0, \dots, N-1\}, j \in \{0, \dots, M-1\}\}$. An arbitrary state in $\mathcal{H}$ is of the form $\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \alpha_{ij} |i\rangle \otimes |j\rangle$ with some complex amplitudes α_{ij}. Similarly, we can describe quantum states that reside in Hilbert spaces that are comprised of multiple Hilbert spaces. For a formal introduction into the tensor product arithmetic, we refer to [NC16], Section 2.1.7.

Measurement

One of the operations that we can perform on a quantum system is that of a measurement. We can think of a measurement as observing a quantum state. We cannot "see" the superposition of states but only classical states. A measurement of a state $|\psi\rangle$ determines one and only one classical state that we will see. The probability with which we see one of the classical states in the superposition depends on the amplitudes α_i; we see the classical state $|i\rangle$ with probability $|\alpha_i|^2$, which is known as *Born's rule*. Hence, observing a quantum state induces a probability distribution of the classical states $|0\rangle, \dots, |N-1\rangle$ and we have $\sum_{i=0}^{N-1} |\alpha_i|^2 = 1$, i.e., any pure quantum state has Euclidean norm 1. After performing a measurement and obtaining as *outcome* some i, the superposition "collapses". What remains is only the classical state $|i\rangle$; all other information in the form of the other amplitudes α_j vanishes. Note that the probability of any measurement outcomes is determined by the squared absolute value of the corresponding amplitudes. These remain the same when we multiply the overall state $|\psi\rangle$ by some *global phase* $e^{i\theta}$, $e^{i\theta} |\psi\rangle$, which is why we sometimes say that the *global* phase of a state is irrelevant.

The process outlined above describes a *measurement in the computational (standard) basis* and has as its output a classical value $i \in \{1, \dots, N\}$ corresponding to a computational basis state $|i\rangle$. A measurement in the computational basis is a special form of a so-called *projective measurement*: a projective measurement is characterised by some m projectors $P_1, \ldots, P_m$ that sum to identity, $\sum_i P_i = I$. Each of the projectors P_i projects on some subspace $\mathcal{H}_i$ of the total Hilbert space $\mathcal{H}$. The projectors are Hermitian and satisfy the relation $P_iP_j = \delta_{i,j}P_i$ for all i, j, so that the subspaces $\mathcal{H}_i$ are mutually orthogonal as well and $P_i^2 = P_i$ for all i. Every pure quantum state $|\psi\rangle$ can be decomposed uniquely into $|\psi\rangle = \sum_i^m |\psi_i\rangle$ with $|\psi_i\rangle := P_i |\psi\rangle \in \mathcal{H}_i$. When such a projective measurement is applied to a state $|\psi\rangle$ then the outcome will be i with probability $\| |\psi_i\rangle \|^2 = \mathrm{Tr}(P_i |\psi\rangle\langle\psi|) = \langle\psi| P_i |\psi\rangle$. Similar to the above measurement in the computational basis, if the outcome of the measurement is some i then the state collapses to the normalised state $|\psi_i\rangle / \| |\psi_i\rangle \| = P_i |\psi\rangle / \|P_i |\psi\rangle \|$. The measurement in the computational basis is a special case of a projective measurement where the projectors are given by $P_i = |i\rangle\langle i|$ and $m = N$.

It is often convenient to write down a projective measurement given by some projectors $P_1, \ldots, P_m$ and associated *distinct* outcomes $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ in a more succinct way. This can be achieved by writing the projectors as one matrix $M = \sum_{i=0}^m \lambda_i P_i$, which is called an *observable*.

Definition 1.1.1 (Observable). An observable M is a Hermitian operator on the state space $\mathcal{H}$. By the spectral theorem, its spectral decomposition is given by some

$$M = \sum_i \lambda_i P_i,$$

where P_i denotes the projector onto the eigenspace of M with real eigenvalue $\lambda_i \in \mathbb{R}$.

Some important observables that we will often see are given by the 2×2 *Pauli matrices* I, X, Y and Z that we will later introduce in more detail. For example, the Pauli matrices X and Z are given by

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = |0\rangle\langle 0| - |1\rangle\langle 1| \quad \text{and} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = |+\rangle\langle +| - |-\rangle\langle -|,$$

and correspond to the projective measurement of a qubit in the computational basis (Fourier basis) with outcomes $+1$ and -1 for $|0\rangle$ and $|1\rangle$ ($|+\rangle$ and $|-\rangle$), respectively. The most general form of a measurement is described by Postulate 3 of quantum mechanics (Section 2.2.3 in [NC16]). It states that a quantum measurement is described by a collection of *measurement operators* $\{M_i\}$ acting on the state space of the quantum system that is being measured, the index i representing the outcome of the measurement. For some pure quantum state $|\psi\rangle$, the probability of obtaining measurement outcome i is given by $\langle\psi| M_i^\dagger M_i |\psi\rangle$ and the post-measurement state is given by

$$\frac{M_i |\psi\rangle}{\sqrt{\langle\psi| M_i^\dagger M_i |\psi\rangle}}.$$

The measurement operators M_i satisfy the *completeness equation* $\sum_i M_i^\dagger M_i = I$. One can easily see that the projective measurement described above is a special cases of a general quantum measurement. It has its projectors P_i as the measurement operators M_i and the property $P_i^2 = P_i$ ensures that the completeness relation is satisfied. In this work we will often times consider projective measurements instead of general measurements, and in the next section we will see that this in fact does not impose any restrictions.

Unitary evolution of quantum states

The evolution of a closed quantum system is decribed by a change in its amplitudes,

$$|\psi\rangle = \alpha_0 |0\rangle + \ldots + \alpha_{N-1} |N-1\rangle \quad \mapsto \quad \beta_0 |0\rangle + \ldots + \beta_{N-1} |N-1\rangle =: |\phi\rangle$$

Such an evolution of a state is subject to two constraints: first of all, the laws of quantum mechanics only admit linear operations. This means that if we represent a pure state as $|\psi\rangle = (\alpha_0, \ldots, \alpha_{N-1})^t$, evolving the initial state $|\psi\rangle$ to some other state $|\phi\rangle$ corresponds to applying an $N \times N$ complex-valued matrix U to $|\psi\rangle$:

$$U |\psi\rangle = U \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{N-1} \end{pmatrix}.$$

Note that by linearity this implies $|\phi\rangle = U|\psi\rangle = U(\sum_{i=0}^{N-1} \alpha_i |i\rangle) = \sum_{i=0}^{N-1} \alpha_i U |i\rangle$. The second constraint on U is that the resulting vector of amplitudes $|\phi\rangle$ must still induce a probability distribution, $\sum_{i=0}^{N-1} |\beta_i|^2 = 1$. This implies that any such operation U must preserve the norm of vectors, i.e., U must be a *unitary* transformation. A matrix U is unitary if its inverse U^{-1} is equal to its conjugate transpose $U^\dagger$. Hence we see that any unitary transformation is reversible. This is in stark contrast to the application of a measurement that in general "deletes" a lot of information in a quantum state. We denote the set of $N \times N$ unitaries by $\mathcal{U}(N)$. In Appendix A we have listed an overview of the most common unitaries used in this work.

Qubits and quantum entanglement

We have noted earlier that the basic unit of computation in classical computations is a bit which can be either in state 0 or 1. The basic unit of computation in quantum computing is called a *quantum bit* (or *qubit*). A single qubit is in a superposition of the two basis states $|0\rangle$ and $|1\rangle$ which we identify by the two orthogonal unit vectors $(1, 0)^t$ and $(0,\ 1)^t$, respectively. Hence, a single qubit can be written as $\alpha_0 |0\rangle + \alpha_1 |1\rangle$ such that $|\alpha_0|^2 + |\alpha_1|^2 = 1$, and resides in the complex Hilbert space $\mathbb{C}^2$. Similar to the earlier description of bipartite quantum systems, a 2-qubit system is a linear combination of the four basis states $|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle$ and $|1\rangle \otimes |1\rangle$. For instance, the state $|1\rangle \otimes |0\rangle$ characterises a 2-qubit system, the first of which is in basis state $|1\rangle$, the second of which is in basis state $|0\rangle$. Most of the times we will abbreviate these basis states by $|0\rangle|0\rangle, |0\rangle|1\rangle, \ldots, |1\rangle|1\rangle$ or even $|00\rangle, \ldots, |11\rangle$.

More generally, an n-qubit system has 2^n basis states and instead of denoting each of them by its binary representation, $|b_1 b_2 \ldots b_n\rangle$ with $b_i \in \{0, 1\}$ we often identify them with $|i\rangle$ where $i \in \{0, \ldots, 2^n - 1\}$. Note that the i-th of the 2^n basis state in an n-qubit system corresponds to the binary representation of i, i.e., the vector that has a 1 in the i-th coordinate and 0's in all other coordinates, again counting from 0. For example, in a 2-qubit system, $|3\rangle = |11\rangle = |1\rangle \otimes |1\rangle$. This also implies that any two vectors $|i\rangle$ and $|j\rangle$ are orthogonal whenever $i \neq j$. We call a system of n qubits a *quantum register* of n qubits; any such register can be represented as a superposition of these computational basis states,

$$\alpha_0 |0\rangle + \ldots + \alpha_{2^n-1} |2^n - 1\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle \text{ with } \sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1,$$

and a measurement in the computational basis produces outcome i with probability $|\alpha_i|^2$. Similarly, performing a measurement in the computational basis on a single, say, the first, qubit within such higher-dimensional systems corresponds to the projectors $|0\rangle\langle 0| \otimes I_{2^{n-1}}$ and $|1\rangle\langle 1| \otimes I_{2^{n-1}}$. Accordingly, we can also calculate the post-measurement state of such a single-qubit measurement on a larger quantum register.

This leads us to another important property of quantum information called *entanglement*. Entanglement describes the non-local effects that quantum operations might take on different, potentially spatially separated qubits. Consider the 2-qubit system

$$\frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle . \tag{1.3}$$

The above state is one of the so-called *EPR-pairs*, owing their name to Einstein, Podolsky and Rosen who studied these states and their properties [EPR35]. Initially, both qubits in the state are in a superposition of $|0\rangle$ and $|1\rangle$. However, if, say, the first qubit is measured and the outcome is 1, then the total state collapses to $|11\rangle$ and thus immediately fixes the second qubit as well, without measuring it. Such non-local quantum effects hold even if the two qubits in question are far apart which initially led to a lot of disbelief among researchers. What is crucial in the above example is that the two qubits cannot be written as the tensor product of two qubits that reside in two different Hilbert spaces. More generally, a bipartite state $|\psi\rangle$ is called *entangled* if it cannot be written as the tensor product of two pure states, i.e., there exist no $|\psi_A\rangle$ and $|\psi_B\rangle$ such that $|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$.

Elementary gates and quantum interference

We have seen above that we can evolve a pure quantum state by applying a unitary matrix to it. Unitaries are also called (quantum) *gates* in analogy to logic gates such as AND, OR and NOT in classical, binary computations. In the following we will discuss some of the most important quantum gates.

The so-called *bit flip* gate X acts on a single qubit and swaps the two basis states $|0\rangle$ and $|1\rangle$, similar to a classical bit flip. The *phase flip* gate puts a -1 in front of the basis state $|1\rangle$ and leaves $|0\rangle$ unchanged. This *relative phase* is not to be confused with the global phase mentioned earlier, and can in fact lead to observable differences in the measurement statistics of quantum systems as we will see later on. Together with the 2×2 identity matrix I and the Y-gate that corresponds to a phase flip followed by a bit flip, $Y = iXZ$, these four matrices form an important class of single qubit gates:

Definition 1.1.2 (Pauli gates and Pauli group). The *Pauli matrices* are given by

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \text{and} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

The single-qubit *Pauli group*, $\mathcal{P}_1$, is comprised of all Pauli matrices as well as the factors ± 1 and $\pm i$,

$$\mathcal{P}_1 = \langle X, Y, Z \rangle_{\{\pm 1, \pm i\}},$$

while the n-qubit Pauli group, $\mathcal{P}_n$, consists of all n-fold tensor products of Pauli matrices, including the complex factors $\pm 1, \pm i$.

Note that each of the Pauli matrices is self-inverse, that is, $U^{-1} = U$ for all $U \in \{I, X, Y, Z\}$. Another frequently used single-qubit gate is the *Hadamard gate* (also called the *Hadamard* or *Fourier transform*), H, given by

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.4}$$

It acts on the two computational basis states as

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \qquad \text{and} \\ H|1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle, \end{aligned}$$

i.e., it transforms the computational basis to the Fourier basis. Just like the Pauli matrices, the Hadamard gate is self-inverse, as can be seen from the following computation:

$$H(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)) = \frac{1}{\sqrt{2}}H|0\rangle + \frac{1}{\sqrt{2}}H|1\rangle = \frac{1}{2}(|0\rangle + |1\rangle) + \frac{1}{2}(|0\rangle - |1\rangle) = |0\rangle,$$

and similarly for $|1\rangle$. In the above calculation we can observe that the amplitudes on $|1\rangle$ cancelled out in the last equality. This effect is called *interference*, similar to the interference that can be observed between light or sound waves. Moreover, we see that the Hadamard transform can be expressed as a linear combination of single-qubit Pauli matrices, $H = X/\sqrt{2} + Z/\sqrt{2}$. This result in fact holds in more generality and we will use it in Section 2.5.1.

Theorem 1.1.1. *The complex span of single-qubit Pauli matrices generates the space of complex 2×2 matrices, $\mathbb{C}^{2\times 2}$. More generally, the set of tensor products of n Pauli matrices, $\{I, X, Y, Z\}^{\otimes n}$ forms a basis for the complex vector space of $2^n \times 2^n$ matrices, $\mathbb{C}^{2^n \times 2^n}$.*

Proof. This first result follows from the observation that

$$\frac{I+Z}{2} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \frac{I-Z}{2} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad \frac{X+iY}{2} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad \frac{X-iY}{2} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}.$$

Having found a basis for $\mathbb{C}^{2\times 2}$, taking the tensor products then generalises to the n-fold tensor product space $(\mathbb{C}^{2\times 2})^{\otimes n} = \mathbb{C}^{2^n \times 2^n}$. □

In later sections, we will sometimes use an abbreviated notation for the application of a single-qubit Pauli matrix on a larger quantum register: given an n-qubit quantum register, we will sometimes use U_i for a Pauli matrix U to denote the application of U to the i-th qubit, i.e., $U_i := \underbrace{I \otimes \ldots \otimes I}_{i-1} \otimes U \otimes \underbrace{I \ldots \otimes I}_{n-i}$.

Last but not least we introduce a common 2-qubit gate, the *controlled-NOT* or *CNOT gate*. The CNOT gate is characterised by a *control qubit* and a *target qubit* and flips the target qubit based on the value of the control qubit. More specifically, it maps $|b_1\rangle |b_2\rangle \mapsto |b_1\rangle |b_2 \oplus b_1\rangle$ for $b_1, b_2 \in \{0, 1\}$ if b_1 is the control and b_2 is the target, where $b_1 \oplus b_2$ denotes the addition modulo 2. Similarly for the case where the second qubit is the control and the first qubit is the target qubit. In matrix form, the CNOT gate is written as

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{1.5}$$

Having introduced the unitary evolution of quantum states, we recall our earlier claim that considering projective measurements instead of general measurements can be done without loss of generality in many cases. Indeed, Nielsen and Chuang show in [NC16], Section 2.2.8, that projective measurements augmented with the ability to introduce ancilliary quantum registers and unitary evolution can be used to implement an arbitrary general measurement described by some measurement operators $\{M_i\}$. We will use this fact when discussing the computational model for the use in multi-party quantum computation in Section 3.1.

In the above first introduction into the principles of quantum computation we have seen some effects such as superposition, entanglement and interference that are exclusive to quantum information and can seem quite paradoxical at first. However, we want to note that processing quantum information also imposes some restrictions as compared to classical information. One important example is the *quantum no-cloning theorem*:

Theorem 1.1.2 (No-cloning). *There does not exist any unitary U that maps*

$$|\psi\rangle\,|0\rangle \quad \mapsto \quad |\psi\rangle\,|\psi\rangle \tag{1.6}$$

for arbitrary quantum states $|\psi\rangle$.

Proof. Assume such a unitary U existed. Then for any state $|\psi\rangle$ we have $U\,|\psi\rangle\,|0\rangle = |\psi\rangle\,|\psi\rangle$. In particular, U maps $|0\rangle\,|0\rangle \mapsto |0\rangle\,|0\rangle$ and $|1\rangle\,|0\rangle \mapsto |1\rangle\,|1\rangle$. Consider the state $|\psi\rangle = |+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$. By linearity we find

$$U(|+\rangle\,|0\rangle) = \tfrac{1}{\sqrt{2}}(U\,|0\rangle + U\,|1\rangle)\,|0\rangle = \tfrac{1}{\sqrt{2}}(U\,|0\rangle\,|0\rangle + U\,|1\rangle\,|0\rangle) = \tfrac{1}{\sqrt{2}}(|0\rangle\,|0\rangle + |1\rangle\,|1\rangle)$$

but the latter is not equal to $|+\rangle\,|+\rangle$. □

In classical computations, cloning information is frequently used, for instance to protect information from noise, as we will for instance see in Section 2.2. The limitation in form of the above theorem caused by the principles of quantum mechanics cannot be understated and we will frequently see its effects in this work. On the other hand, we also note that the no-cloning theorem concerns cloning of *arbitrary* quantum states. Classical information such as bit strings encoded as computational basis states can be cloned using simple CNOT gates: for any $b \in \{0,1\}$ we have $CNOT(|b\rangle\,|0\rangle) = |b\rangle\,|b\rangle$.

Quantum states in more generality

In the context of classical computations, we sometimes have uncertainty over the state that a system is in. In such cases, the system is expressed as a random variable that has a probability distribution over the different states that it could be in. A similar reasoning can be applied to quantum states. In the previous section, we have focussed solely on pure quantum states that can be expressed as a single vector of amplitudes. More generally, a *mixed quantum state* is a probability distribution over pure quantum states. Let

A be a linear operator on some Hilbert space $\mathcal{H}$, $A : \mathcal{H} \to \mathcal{H}$. We say that A is *positive semi-definite* if A is Hermitian and has only non-negative eigenvalues. Then we define mixed quantum states as follows:

Definition 1.1.3 ((Mixed) quantum states). A *(mixed) quantum state* on a Hilbert space $\mathcal{H}$ is an element of the set of positive semi-definite operators on $\mathcal{H}$ with trace 1. Quantum states are often referred to as *density operators* or *density matrices*.

A pure quantum state $|\psi\rangle$ corresponds to the density matrix $|\psi\rangle\langle\psi|$. A mixed state ρ that is in pure states $|\psi_1\rangle, \dots, |\psi_n\rangle$ with probabilities $p_1, \dots, p_n$, $\sum_{i=1}^n p_i = 1$ corresponds to the density matrix $\rho = \sum_{i=1}^n p_i |\psi_i\rangle\langle\psi_i|$. By the spectral theorem for Hermitian operators, any quantum state can be written as a sum

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \tag{1.7}$$

with eigenvalues p_i and corresponding orthonormal eigenvectors $|\psi_i\rangle$. Since all quantum states are postive semi-definite, all eigenvalues satisfy $p_i \geq 0$. Moreover, since $1 = \mathrm{Tr}(\rho) = \sum_i p_i$ we find that the eigenvalues $(p_i)_i$ do in fact form a probability distribution. We call the number of non-negative eigenvalues the *rank* of the density matrix, such that pure states are exactly the mixed states of rank 1. In the literature on quantum computation, it is assumed that one starts off with a pure quantum state. During the course of a computation, quantum states are then typically depicted as mixed states, which we will see especially in situations where a quantum adversary might have tampered with a quantum state (Section 3).

Similar to the first section, we can define measurements and unitary operations on mixed states. We saw that applying a unitary U to a pure state $|\psi\rangle$ results in the pure state $U|\psi\rangle$. In terms of density matrices, this is equivalent to applying

$$|\psi\rangle\langle\psi| \mapsto U|\psi\rangle\langle\psi| U^\dagger.$$

By linearity of U this generalises to arbitrary mixed states ρ such that U acts on ρ as

$$\rho \mapsto U\rho U^\dagger.$$

Measurements on mixed quantum states are defined as follows. Consider a projective measurement with orthogonal projectors $P_1, \dots, P_m$ that sum to the identity. When applying this measurement to a mixed state ρ, the probability to get outcome i is given by $p_i = \mathrm{Tr}\left(P_i \rho P_i^\dagger\right)$. In case of outcome i, the state then collapses to the (normalised) state $P_i \rho P_i^\dagger / p_i$. To show that this definition is in fact consistent with the measurement we defined earlier, consider a measurement in the computational basis of the n-dimensional space with projectors P_i given by $|0\rangle\langle 0|, \dots, |n\rangle\langle n|$ on some pure state $|\psi\rangle = \sum_{i=0}^n \alpha_i |i\rangle$. By the definition above, the probability to obtain outcome i is given by $p_i = \mathrm{Tr}\left(P_i |\psi\rangle\langle\psi| P_i^\dagger\right) = |\langle i|\psi\rangle|^2 = |\alpha_i|^2$, which corresponds exactly to the probability of obtaining outcome i in our previous definition. Similarly, we see that after the measurement, the state collapses to

$$\frac{P_i |\psi\rangle\langle\psi| P_i^\dagger}{p_i} = \frac{|i\rangle\langle i| \, |\psi\rangle\langle\psi| \, |i\rangle\langle i|}{|\alpha_i|^2} = |i\rangle\langle i| \,.$$

Partial trace and purifications

In quantum cryptography, we are often interested in describing the information that some player holds. For this, we need means to describe the share of a quantum system that is held by some participant, e.g., the adversary. We have already seen in the discussion of entanglement that in general, if, say, Alice and Bob share some pure quantum state $|\psi\rangle$ then this cannot always be described as a tensor product $|\psi_A\rangle \otimes |\psi_B\rangle$ from which we can immediately deduce the state held by Alice respectively Bob. One example for such a state was the EPR pair $(|00\rangle + |11\rangle)/\sqrt(2)$. The generalised definition of quantum states in Def. 1.1.3, however, provides a way to describe the state held by Alice (respectively Bob) locally as a mixed state. This can be accomplished by *tracing out* Bob's part of the global state. For some tensor product of matrices $C \otimes D$ we can define the *partial trace over D* as $\mathrm{Tr}_D(C \otimes D) := C \cdot \mathrm{Tr}(D)$. For example, assume that in the case of the EPR pair the first qubit is held by Alice and the second is held by Bob. Their joint density matrix is given by

$$\begin{aligned}\rho_{AB} &= \frac{1}{2}(|00\rangle + |11\rangle)(\langle 00| + \langle 11|)\\ &= \frac{1}{2}(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)\\ &= \frac{1}{2}(|0\rangle\langle 0| \otimes |0\rangle\langle 0| + |0\rangle\langle 1| \otimes |0\rangle\langle 1| + |1\rangle\langle 0| \otimes |1\rangle\langle 0| + |1\rangle\langle 1| \otimes |1\rangle\langle 1|).\end{aligned}$$

We have that $\mathrm{Tr}(|a\rangle\langle b|) = \delta_{ab}$ for computational basis states (and similarly for arbitrary orthonormal $|a\rangle$ and $|b\rangle$). Hence, by tracing out Bob's part, we find that Alice's density matrix is given by

$$\rho_A = \mathrm{Tr}_B(\rho_{AB}) = \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|).$$

From Alice's point of view, the state she is holding is equivalent to a completely random coin flip. We sometimes call the state that is obtained from removing Bob's share of the state Alice's *reduced density matrix*. By linearity, we can define the partial trace of a quantum state more generally:

Definition 1.1.4 (Partial trace). Let $\mathcal{H}_A$ and $\mathcal{H}_B$ be two Hilbert spaces, and let $(|b\rangle_i)$ be an orthonormal basis of $\mathcal{H}_B$. Let ρ_{AB} be a quantum state in the overall Hilbert space $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$. Then the *partial trace over B* is the linear map defined by

$$\mathrm{Tr}_B(\rho_{AB}) := \sum_i (I_A \otimes \langle b_i|)\rho_{AB}(I_A \otimes |b_i\rangle). \tag{1.8}$$

The partial trace over A is defined analogously. Similarly, we can define the partial trace over multiple subsystems which we will often need when considering protocols with multiple participants. For example, if we consider a global state with three subsystems, ρ_{ABC}, the respective partial traces are given by $\rho_A = \mathrm{Tr}_{BC}(\rho_{ABC}), \rho_{AB} = \mathrm{Tr}_C(\rho_{ABC}), \rho_{AC} = \mathrm{Tr}_B(\rho_{ABC})$, etc.

Quantum channels

The most general formalism to describe the evolution of quantum states, which is a generalisation of all of the above operations, is that of a *quantum channel* or *completely positive, trace-preserving (CPTP) map* which maps density matrices to density matrices. We adopt the definition given by [Pre99]. Quantum channels allow us to describe the evolution of a quantum system S that is part of a larger, closed system, S, together with its environment, E, without having to reference the environment E.

Definition 1.1.5 (Quantum channel). A *quantum channel* is a linear operator $\mathcal{E}$ from the space of density matrices of some Hilbert space $\mathcal{H}_A$ to the space of density matrices of some Hilbert space $\mathcal{H}_B$ that acts on a density matrix ρ on $\mathcal{H}$ as

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger, \tag{1.9}$$

where the $\{E_i\}_i$ are linear operators mapping from $\mathcal{H}_A$ to $\mathcal{H}_B$ such that $\sum_i E_i^\dagger E_i = I_{\mathcal{H}_A}$.

The above set of operators $\{E_i\}$ associated to a quantum channel $\mathcal{E}$ are called the *Kraus operators* and are, in general, not unique. A derivation of the necessary properties of a quantum channel as well as an explanation to the equivalent term of a CPTP map can be found in [Wal20]. For our purposes it suffices to know that we can describe arbitrary quantum operations in form of a quantum channel and that a decomposition in form of Eq. 1.9 exists. This representation is in particular used for modelling noise processes (Section 2.5.1) as well as arbitrary actions on part of an adversary within a quantum multi-party computation protocol (Chapter 3).

Comparing quantum states

We conclude this introduction with a note on ways to compare quantum states. These will be particularly needed when comparing the output states of different quantum protocols such as quantum secret sharing or MPQC protocols such as, e.g., in Section 2.6 and Section 3. Two measures that are used particularly often are the *trace distance* and *fidelity* of two quantum states. Intuitively speaking, the trace distance measures the distance between two quantum states while the fidelity expresses the similarity of two quantum states.

Definition 1.1.6 (Trace norm). Let M be a linear operator between two Hilbert spaces $\mathcal{H}_a$ and $\mathcal{H}_B$ with singular values $s_1, \dots, s_r > 0$. Then the *trace norm* of M is defined as

$$\|M\|_1 := \sum_{i=1}^{r} s_i = \operatorname{Tr} \sqrt{M^\dagger M}. \tag{1.10}$$

Note that on the right-hand side, the trace of the square root of the positive semi-definite operator $M^\dagger M$ is taken. More generally, if an operator A is positive semi-definite, then the *(positive semi-definite) square root* $\sqrt{A}$ is the operator with the same eigenvectors, but whose eigenvalues are the square root of those of A. In particular, if $A = \sum_i \lambda_i |e_i\rangle\langle e_i|$ is the eigendecomposition of A, then $\sqrt{A} = \sum_i \sqrt{\lambda_i} |e_i\rangle\langle e_i|$.

Using the above norm, we can now define metrics to measure the distance between two quantum states. One frequently used measure is the trace distance:

Definition 1.1.7 (Trace distance). For two density matrices ρ, σ on $\mathcal{H}$, the *(normalised) trace distance* between them is defined as

$$\mathrm{T}(\rho, \sigma) := \frac{1}{2} \|\rho - \sigma\|_1 . \tag{1.11}$$

For the trace distance of two quantum states we have for all ρ, σ that $\mathrm{T}(\rho, \sigma) \in [0, 1]$, and that $\mathrm{T}(\rho, \sigma) = 0$ if and only if $\rho = \sigma$, so the trace distance is at its lowest when the states are the same. For pure states $\rho = |\phi\rangle$ and $\sigma = |\psi\rangle$, the trace distance has the following relation to their *overlap* $|\langle\phi|\psi\rangle|$:

$$\mathrm{T}(\rho, \sigma) = \sqrt{1 - |\langle\psi|\phi\rangle|^2}.$$

For more properties of the trace distance, see Section 3.2 in [Wal20]. On the other hand, the *fidelity* can be used to measure the closeness of two quantum states:

Definition 1.1.8 (Fidelity). For two density matrices ρ, σ on $\mathcal{H}$, their *fidelity* is defined as

$$\mathrm{F}(\rho,\sigma) := \mathrm{Tr}\left(\sqrt{\sqrt{\sigma}\rho\sqrt{\sigma}}\right) = \left\|\sqrt{\rho}\sqrt{\sigma}\right\|_1. \tag{1.12}$$

In contrast to the trace distance, the fidelity of two states is large when the two states are similar: we have for all ρ, σ that $\mathrm{F}(\rho,\sigma) \in [0,1]$ and $\mathrm{F}(\rho,\sigma) = 1$ if and only if $\rho = \sigma$. For pure quantum states $\rho = |\psi\rangle\langle\psi|$ and $\sigma = |\phi\rangle\langle\phi|$ we find that $\sqrt{\rho} = |\psi\rangle\langle\psi|$ (similarly for σ), so their fidelity is given by

$$\mathrm{F}(\rho,\sigma) = \sqrt{\langle\phi|\psi\rangle\,\langle\psi|\phi\rangle} = |\langle\phi|\psi\rangle|, \tag{1.13}$$

and therefore generalises the overlap of pure states.

2. Secret sharing and error correction

Secret sharing schemes are an essential ingredient for many secure multi-party computation protocols. Informally, the goal of secret sharing scheme is to distribute a secret among a number of participants by encoding the secret's content into a number of shares. Only certain authorised subsets of parties should be able to reconstruct the secret while unauthorised parties cannot obtain any information at all about the secret by pooling their shares. To illustrate this, imagine that Alice, Bob and Charlie jointly have a bank account which they only want to be able to access if the majority of the three agrees to do so. In order to achieve this, they might distribute the PIN to their bank account using a suitable secret sharing scheme, such that at least two of their three shares are needed in order to reconstruct their PIN and access their joint bank account.

Being able to securely share private inputs in fact constitutes a large step towards realising secure multi-party computation protocols. In fact, encoding into shares and distributing them among multiple participants securely combined with the ability to perform meaningful operations on these shares turns out to be sufficient for performing arbitrary secure multi-party computations. We consider the example of *additive secret sharing*: at the start of the protocol, each of some number n of participants, $P_i, 1 \leq i \leq n$, holds a private input s_i, their *secret*, from some finite field $\mathbb{F}_q$ with $q > n$. To share their secret, each party samples some $n-1$ elements $s_{i,j} \in \mathbb{F}_q, j = 1, \ldots, n-1$, uniformly at random and sets $s_{i,n} := s_i - (s_{i,1} + \cdots + s_{i,n-1})$. Based on this we can observe three things: on one hand, $s_{i,1} + \cdots + s_{i,n} = s_i$ such that if all n parties pool their inputs then they can reconstruct the private input s_i. On the other hand, any smaller number of participants $k < n$ is unable to extract any information about s_i: from the colluding participants' view, the k shares are distributed uniformly at random in $\mathbb{F}_q$ and therefore their sum gives a random element in $\mathbb{F}_q$ as well. These two observations show that the above scheme constitutes a secret sharing scheme. Moreover, we see that if each participant adds together the shares of the other parties' secrets, then the resulting values form a secret sharing of the sum of all of the parties' secrets: party j holds shares $s_{i,j}$ for $i = 1, \ldots, n$. Adding these together, party j now holds a new share $s'_j = s_{1,j} + \cdots + s_{n,j}$ of the secret $s_1 + \cdots + s_n$. Similarly, we can see that multiplying shares with constants is also possible with this basic secret sharing scheme, so we can in fact compute any *linear* function $f(s_1, \ldots, s_n)$ in a distributed, privacy-preserving manner. Achieving multiplicativity, that is, the ability to securely compute the product of multiple shares, requires more work but given a scheme that allows to privately evaluate addition and multiplication of inputs, any *arithmetic circuit* can be evaluated. We will formally introduce the arithmetic circuit model in Section 3.3. For now, we remark that we assume all computations to be additions or multiplications over some finite field $\mathbb{F}$ and that therefore, these two gates are universal in this model of computation. Hence, we see that secret sharing schemes with properties such as additivity and multiplicativity form important building blocks for multi-party computation protocols. In this chapter, we will review some fundamental results on secret sharing schemes and will discuss some constructions. In particular, we will explore their connection to the theory of *error-correcting codes*: we will see that the well-studied area of these codes exhibits close links to that of secret sharing and provides tools to analyse and construct secret sharing schemes efficiently.

Given the overall context of this thesis, we will then discuss the concepts of secret sharing and error correction in the quantum realm. We will see what additional problems may arise when viewing these as means to secret-share or protect quantum information, and what benefit the use of quantum information may offer in solving these cryptographic tasks. In particular, we will see that in particular for quantum protocols, close connections exist and that the more established theory of quantum error correction may be harnessed to construct good quantum secret sharing schemes.

2.1 Secret sharing schemes

Secret sharing schemes aim at distributing a secret value s among some n parties such that certain subsets of parties can recover the secret jointly from their inputs while others cannot obtain any information about the secret. Typically, the party performing the sharing of the secret is called the *dealer* and the input that the individual players are handed are called *shares* of the secret s. The collection of shares obtained from the scheme will sometimes be denoted by $[s]$ and we say that $[s]$ is a *secret sharing* of s. Secret sharing has been invented independently by Adi Shamir [Sha79] and George R. Blakley [Bla79] in 1979. Both authors wanted to tackle the single-point-of-failure risk that arises when storing cryptographic keys. In particular Shamir's scheme remains one of the most widely used secret sharing schemes and we will later discuss it in more detail; it is both an illustrative example of a secret-sharing scheme with nice properties as well as for the close link between classical secret sharing and error correction.

First, we describe some of the elementary notions and basic schemes for secret sharing. Most of the definitions in this section follow [Bei11] and [CDN15], which the reader should consult for a more in-depth introduction of secret sharing schemes. As noted above, a secret sharing scheme should enable certain subsets of players to reconstruct the secret while others cannot. More formally, we can define the notion of an *access structure* as follows:

Definition 2.1.1 (Access structure). We denote the participants in a secret sharing scheme by $\mathcal{T} = \{T_1, \ldots, T_n\}$. A collection of subsets of the participants, $\Gamma \subseteq \mathcal{P}(\mathcal{T})$, is called *monotone* if for every $B \in \Gamma$ and $B \subseteq C \in \mathcal{P}(\mathcal{T})$ we have $C \in \Gamma$. A monotone collection $\Gamma \subseteq \mathcal{P}(\mathcal{T})$ consisting of non-empty subsets of $\mathcal{T}$ is called an *access structure*. The sets in Γ are called *authorised*, and sets outside of Γ are called *unauthorised*.

Similarly, we need to be able to describe the subsets of participants that should not learn anything about the secret from their collective shares. We can do so by means of an *adversary structure*.

Definition 2.1.2 (Adversary structure). A collection of subsets of the participants, $\mathcal{A} \subseteq \mathcal{P}(\mathcal{T})$, is called *antimonotone* if for every $C \in \mathcal{A}$ and $B \subseteq C$ we have $B \in \mathcal{A}$. An antimonotone collection $\mathcal{A} \subseteq \mathcal{P}(\mathcal{T})$ consisting of subsets of $\mathcal{T}$ is called an *adversary structure*.

In the literature, sometimes the term *privacy sets* is used to describe elements in the adversary structure. Note that by definition of an access structure, the collection of unauthorised sets is antimonotone, too. We furthermore need to define the notion of a distribution scheme.

Definition 2.1.3 (Distribution scheme, according to [Bei11]). A *distribution scheme* $\mathsf{Share} = \langle \Pi, \mu \rangle$ with domain of secrets K is a pair consisting of a probability distribution μ on some finite set R called the *set of random strings*, and Π is a mapping from $S \times R$ to the set of n-tuples $S_1 \times \ldots \times S_n$, where S_i is called the *domain* of shares of T_j. A dealer distributes a secret $s \in S$ according to Share by first sampling a random string $r \in R$ according to μ, computing a vector of shares $\Pi(s, r) = (s_1, \ldots, s_n) =: [s]$ and privately communicating share s_j to party T_j. For some set $B \subseteq \mathcal{T}$, let $\Pi(s, r)_B$ denote the restriction of $\Pi(s, r)$ to the entries in B.

In the following work on secret sharing schemes, we will often consider $S = S_i = \mathbb{F}$ for all i, for some finite field $\mathbb{F}$. Formally, we then define a secret sharing scheme as follows.

Definition 2.1.4 (Secret sharing, according to [Bei11]). Let S be a finite set of secrets such that $|S| \geq 2$. A distribution scheme $\mathsf{Share} = \langle \Pi, \mu \rangle$ with domain of secrets S is a *secret sharing scheme* realising an access structure Γ and an adversary structure $\mathcal{A}$ such that $\Gamma \cap \mathcal{A} = \emptyset$, if the following two requirements hold:

(1) *Correctness*: The secret s can be reconstructed by any authorised set of parties, that is, for any $B \in \Gamma$ with $B = \{T_{i_1}, \ldots, T_{i_{|B|}}\}$, there exists a reconstruction function $\mathsf{Rec}_B : S_{i_1} \times \ldots \times S_{i_{|B|}} \to S$ such that for every $s \in S$,

$$\Pr[\mathsf{Rec}_B(\Pi(s, r)_B = s)] = 1. \tag{2.1}$$

(2) *Perfect privacy*: For any set $B \in \mathcal{A}$ and for every two secrets $s, s' \in S$, and for every possible vector of shares $(s_j)_{T_j \in B}$:

$$\Pr\Big[\Pi(s, r)_B = (s_j)_{T_j \in B}\Big] = \Pr\Big[\Pi(s', r)_B = (s_j)_{T_j \in B}\Big]. \tag{2.2}$$

Note that the above definition of perfect privacy guarantees security in the information-theoretic sense: for every two secrets s and s', their distributions of shares are identical. Both of the above requirements can be relaxed such that correctness holds only with high probability and such that the statistical distance between $\Pi(s, r)_B$ and $\Pi(s', r)_B$ is small, in which case we obtain *statistical secret sharing schemes*. A secret sharing scheme for which the collection of unauthorised sets equals the adversary structure, i.e., for which Γ and $\mathcal{A}$ form a partition of $\mathcal{P}(\mathcal{T})$, is referred to as a *perfect secret sharing scheme*. For ease of notation, we will in the following sometimes use Rec to denote the family of reconstruction functions Rec_B associated to a secret sharing scheme Share. One of the most extensively studied kind of access structures is that of a *threshold access structure*:

Definition 2.1.5 (Threshold structure). Let $\mathcal{T} = \{T_1, \ldots, T_n\}$, so $|\mathcal{T}| = n$. Then we call an access structure of the form $\Gamma = \{A \subseteq \mathcal{T} : |A| \geq t\}$ a *(t, n)-threshold (access) structure* with *threshold $t \leq n$*.

Note that in a threshold access structure, any t participants can jointly recover the secret. In the present work, we will mostly focus on *threshold secret sharing schemes* that are perfect secret sharing schemes realising a threshold access structure. Accordingly, an (t, n)-threshold scheme realises a (t, n)-threshold access structure and an adversary structure $\mathcal{A}$ consisting of all subsets of participants of cardinality $\leq t-1$. We say that such a secret sharing scheme has *t-reconstruction*. If any $t-1$ or fewer shares cannot be used to deduce *any* information on the secret, then this is called $(t-1)$-*privacy*. In the present section, we consider secret sharing schemes with t-reconstruction and $(t-1)$-privacy. As noted above, we will consider secret sharing schemes in more generality in Section 2.6 and allow for t reconstruction and p-privacy for some $p < t \leq n$, such that some unauthorised sets may leak partial information about the secret. Such schemes are called *ramp schemes* ([CPT+13]).

From the above definition it is easy to see that any threshold structure is monotone. As an example for a monotone access structure that is not (directly) a threshold structure, consider the following situation: the president holds some secret access code that she wants to secret share among the vice-presidents while being out of office. She does not trust the vice-presidents to each be able to access the secret individually, so she chooses a secret sharing scheme that requires either her share of the secret or at least two vice-presidents' shares to access the secret. Such an access structure would be monotone but not a threshold scheme. Denote the president by P and assume there are two vice-presidents V_1 and V_2. The access structure above would then be given by $\Gamma = \{\{P\}, \{P, V_1\}, \{P, V_2\}, \{P, V_1, V_2\}, \{V_1, V_2\}\}$ and is characterised by different shares having different importance. It should be noted, however, that such an access structure can be realised by a $(4, 2)$ threshold scheme in which the president holds two shares while the vice-presidents hold one each.

For threshold schemes with threshold $t = n$ we already discussed a simple protocol for additive secret sharing based on ideas similar to those of the one-time pad encryption. For the case where $t < n$, slightly more involved ideas are necessary. Shamir's scheme, first proposed in [Sha79], offers a solution that can be adapted for any $t \leq n$:

Definition 2.1.6 (Shamir's secret sharing scheme). Let n denote the number of players and $t \leq n$ be the privacy parameter. Let $\mathbb{F}$ be a finite field containing at least $n+1$ elements and let the secret be $s \in \mathbb{F}$. In order to secret share s, choose some $t-1$ elements $p_1, \ldots, p_{t-1} \in \mathbb{F}$ uniformly at random and define the polynomial

$$f(x) = s + p_1 x + \cdots + p_{t-1} x^{t-1}. \tag{2.3}$$

Fix some distinct, non-zero elements $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ publicly and evaluate f at these n points, $s_i = f(\alpha_i)$ for $i = 1, \ldots, n$. The shares of this scheme are given by the s_i and each player knows which α_i their value corresponds to.

Note that in the above scheme, using Lagrange interpolation we can uniquely determine f from any t shares and therefore recover $s = f(0)$. On the other hand, $t-1$ or less shares reveal no information on s whatsoever: for any $t-1$ shares $s_{k_1}, \ldots, s_{k_{t-1}}$ and for any $z' \in \mathbb{F}$ there exists a unique polynomial $f' \in \mathbb{F}[X]$ of degree $t-1$ such that $f'(\alpha_{k_i}) = s_{k_i}$ for all $i = 1, \ldots, t-1$ and $f'(0) = z'$ (i.e., given only $t-1$ shares, we can reconstruct to any secret). For a full proof of security, see [CDN15].

We have already discussed the linearity of additive secret sharing schemes earlier. Linearity also holds for Shamir's scheme: given two secret sharings $[s] = (s_1, \ldots, s_n)$ and $[s'] = (s'_1, \ldots, s'_n)$ corresponding to secrets s and s', any $\mathbb{F}$-linear combination $a[s] + b[s']$ (i.e., the shares being given by $as_i + bs'_i$) is a secret sharing corresponding to the secret $as + bs'$. Secret sharing schemes satisfying this property are called *linear secret sharing schemes (LSSS)* and allow us to take a first step towards secure MPC, in that linear secret sharing schemes let us perform secure multi-party computations on private inputs for linear functions. MPC protocols designed in this fashion, "on top of a secret sharing scheme", are said to follow the *share-compute-reveal paradigm*: the players secret share their input, perform *local* computations on the shares they received from the other players and finally reconstruct the function value publicly.

We conclude this brief overview on secret sharing schemes with a note on performing multiplications securely using Shamir's secret sharing scheme. For this scheme we have seen that in order to compute a linear function we can simply have each player add shares or multiply them with a (publicly known) constant locally. This turns out to be more difficult for secure multiplication of secrets. Consider an (n, t)-threshold Shamir secret sharing of two secrets s and s' with polynomials f and f' of degree $t-1$. The i-th player holds shares s_i and s'_i. Our goal is to securely compute a secret sharing of the multiplication of the secrets, i.e., compute $[s \cdot s']$. In this setting, having each player multiply the shares they hold, i.e., computing $s_i s'_i$ results in shares of the secret sharing $[ss']$ using the polynomial $f \cdot f'$ of degree $\leq 2t-2$. Contrary to addition and multiplication with a constant, the multiplication of two polynomials increases the degree so that at least $2t-1$ shares are necessary to reconstruct $f \cdot f'$ and recover $ss' = (f \cdot f')(0)$. To circumvent this issue, we convert the shares we have back into a valid shares of a secret sharing of ss' with reconstruction parameter t. Note that the interpolation function L that maps n shares (evaluations) of a polynomial g of degree $< t$ as $L : (g(\alpha_1), \dots, g(\alpha_n)) \mapsto g(0)$, is a linear function and can therefore be securely evaluated locally. To see why, reconsider the Lagrange polynomial:

Definition 2.1.7 (Lagrange polynomial). Given a set of k data points $(x_1, y_1), \dots, (x_k, y_k) \in \mathbb{F}^2$ for some finite field $\mathbb{F}$, the *Lagrange interpolation polynomial* $L \in \mathbb{F}[x]$ is defined as

$$L(x) := \sum_{j=1}^{k} y_j l_j(x).$$

The l_j are called the *Lagrange base polynomials*,

$$l_j(x) = \prod_{\substack{0<m\leq k,\\ m\neq j}} \frac{x - x_m}{x_j - x_m}.$$

Having every party compute the product of their shares $p_i = s_i s'_i$ and secret share this value among the other participants, party i ends up with n shares $p_{i,j}$ for $j = 1, \dots, n$. The participants compute $L(p_1, \dots, p_n) = ss'$ locally and obtain a valid share of the secret ss' with privacy parameter t each. Note that for the local interpolation using the Lagrange polynomials above, the Lagrange base polynomials are available to each participant as the evaluation points $x_i = \alpha_i$ are publicly known. The technique of bringing down the degree of the polynomial that is used in the secret sharing of a multiplication is referred to as *degree-reduction*, as implicitly the degree of the polynomial used in the secret sharing is brought down from at most $2t-2$ to at most $t-1$. More complicated protocols to perform secure multiplication of secrets with Shamir's scheme have been introduced earlier; the above simplified technique was given by Rabin, Rabin and Gennaro in [GRR98]. Note, however, that in order to be able to apply this technique, $2t < n$ has to be satisfied and the degree reduction incurs additional communication cost.

We have seen that achieving multiplicativity comes at the price of additional required communication and that it imposes bounds on the maximum privacy parameter that a scheme can achieve. This observation in fact generalises to many MPC constructions; in Section 2.6, we will analyse the theoretical limitations of secret sharing schemes and in Section 3.4, that of secure MPC protocols.

2.2 Error correcting codes

The development of error-correcting codes (ECCs) was a large step forward for computer science in the 20th century. Nowadays, ECCs are being used to protect information from noise in storage devices or communication channels. Note that the theory of error correction does not physically prevent noise from occurring, but instead protects the underlying information. In the following, we will discuss some of the main concepts of classical error correction and will see that beyond their intended use, they can provide valuable insight and efficient constructions for secret sharing schemes.

To protect information from noise, one adds redundancy to the data in order to make it more "distinguishable". The original data can then still be recovered should some errors occur. Mathematically, data is represented as collections of symbols called *words* over a given finite *alphabet*. For classical information, the alphabet typically is $\{0, 1\}$ and words are represented as binary strings. Codes that process and protect binary information are therefore called *binary codes*. The process of modifying original data in a way that later allows to detect and/ or correct errors is called *encoding*. Data is encoded into words over the same alphabet (this may, in general, also be another target alphabet); the resulting word is referred to as a *codeword*.

One generally distinguishes between two kinds of ECCs, *block codes* and *convolutional codes*. Informally, block codes encode fixed sized blocks of information one by one with fixed output size such that all codewords are of the same length. Convolutional codes on the other hand can process information streams of arbitrary length. Since block codes are more commonly used and all codes that occur in the following fall into that category, we will restrict our attention to block codes and simply refer to them as (error-correcting) codes hereafter. The following general introduction will be based mainly on [Lin98].

Definition 2.2.1 (Code). Let Q be a finite set and let $n \in \mathbb{N}$. Then any subset $\mathcal{C}$ of $Q^n = Q \times \cdots \times Q$ is a *code* of *length n*. The elements $c \in \mathcal{C}$ are called *codewords* over the *alphabet* Q.

In order to distinguish codewords from each other, a distance function is defined on Q^n. Typically, this function is of the following form:

Definition 2.2.2 (Distance of a code). Let Q be an alphabet, $n \in \mathbb{N}$ and $\mathcal{C} \subseteq Q^n$ be an arbitrary code.

(1) For two words $\mathbf{a} = (a_1, \ldots, a_n), \mathbf{b} = (b_1, \ldots, b_n) \in Q^n$,

$$d(\mathbf{a}, \mathbf{b}) = |\{\, i \mid 1 \leq i \leq n,\ a_i \neq b_i\}|$$

is called the *Hamming distance* (usually just referred to as distance) of **a** and **b**.

(2) Accordingly,

$$d(\mathcal{C}) = \min\{\, d(\mathbf{a}, \mathbf{b}) \mid \mathbf{a}, \mathbf{b} \in \mathcal{C}, \mathbf{a} \neq \mathbf{b}\,\}$$

is called the *(minimum) distance* of the code $\mathcal{C}$. We sometimes denote $d(\mathcal{C})$ by d or $d_{\min}$.

One can easily verify that d does indeed define a metric on Q^n: non-negativity, identity of indiscernibles and symmetry follow directly from the definition, and the triangle inequality can be proven by a simple induction over n.

In order to use a code $\mathcal{C} \subseteq Q^n$ for error correction, an injective *encoding procedure* needs to be defined that maps data words to codewords. Recovering a codeword from a received word is called *decoding*. We say that a codeword is affected by an error in one or several positions, that is, coordinates of the codeword, if the symbols in those positions are replaced by other ones. For instance, for binary codes, we can think of an error as some bits being flipped. The goal of error correction is to protect information against noise, so typically it is assumed that errors occur on uniformly random positions, independent of each other, and affect each position in the codeword with some (small) probability. To correct an error, we need to define an *error correction procedure* that maps a received word $\mathbf{x} \in Q^n$ back to the original codeword $\mathbf{c} \in \mathcal{C}$ that can then be decoded to the unique information word it represents. The minimum distance of a code defined above gives us a first mean to determine the error-correcting capabilities of a code, i.e., the amount of changes in different positions of the codeword that can be reverted:

Theorem 2.2.1. *If a code $\mathcal{C} \subseteq Q^n$ has minimum distance d, then it can be used as an error-correcting code that corrects $\lfloor \frac{d-1}{2} \rfloor$ errors.*

Proof. If a code has minimum distance d and $k \leq \lfloor \frac{d-1}{2} \rfloor$ errors occured so that some $\mathbf{a} = (a_1, \ldots, a_n) \in Q^n$ is received, then there exists a unique codeword $\mathbf{c} = (c_1, \ldots, c_n) \in \mathcal{C}$ with distance $d(\mathbf{a}, \mathbf{c}) = k$ to $\mathbf{a}$: if there was a second codeword $\mathbf{b} \in \mathcal{C}$ with distance $d(\mathbf{a}, \mathbf{b}) = k$ then we would find that $d(\mathbf{b}, \mathbf{c}) \leq d(\mathbf{a}, \mathbf{b}) + d(\mathbf{a}, \mathbf{c}) = d - 1$ contradicting the assumption on the minimum distance of $\mathcal{C}$. We can recover $\mathbf{c}$ and therefore correct the errors by computing $\mathbf{c} = \arg\min\{ d(\mathbf{a}, \mathbf{c}') \mid \mathbf{c}' \in \mathcal{C} \}$. □

A similar argument shows that a code $\mathcal{C}$ with minimum distance d can correct $d - 1$ erasure errors. An *erasure error* is an error that erases the information in a *known* coordinate of a codeword. See, e.g., [Iñe20], Theorem 2.2.1 for a full proof.

The (binary) *repetition code* provides a first example for an error-correcting code that is easily understood: in order to protect a bit $b \in \{0, 1\}$ from noise, b is repeated n times, so for $n = 3$, b is mapped to bbb. The code contains exactly two codewords, 0^n and 1^n, with minimum distance n. Hence, up to $\lfloor \frac{n-1}{2} \rfloor$ errors (i.e., bit flips) can be corrected and the error correction and decoding procedure takes a *majority vote*, i.e., it outputs the bit that the majority of positions in the received bit string represents. Alternatively, the repetition code corrects $n - 1$ erasure errors since every coordinate contains the full information on the underlying word. The code length n is chosen based on the likelihood of an error affecting a single bit. Even though the repetition code does not exhibit particularly good error correcting capabilities and uses a lot of redundancy to encode a single bit, it provides a good example for understanding the relation of classical error correction to secret sharing and quantum error correction, as we will see later.

Having covered these basic definitions, we will now shift our attention to codes that exhibit some algebraic structure. In the following, we will consider alphabets Q given by some finite field $\mathbb{F}_q$ where $q = p^r, r \geq 1$ for some prime p. Then Q^n is the n-dimensional vector space $\mathbb{F}_q^n$. In light of this restriction on the underlying alphabet, we can define the *weight* of a word $\mathbf{x} \in \mathbb{F}_q^n$ as the number of positions in which it differs from the all zero vector, $w(\mathbf{x}) = d(\mathbf{x}, \mathbf{0})$. Similar to the minimal distance of a code $\mathcal{C}$, its *minimum weight* is defined as the smallest weight of any non-zero codeword in $\mathcal{C}$. We will later use the weight of a vector to describe the errors that occured on a classical codeword, and similarly in the quantum case.

Definition 2.2.3 (Linear codes). A *q-ary linear code* $\mathcal{C}$ is a linear subspace of $\mathbb{F}_q^n$. If $\mathcal{C}$ has dimension k, then we call $\mathcal{C}$ an $[n, k]_q$ code. If it is clear from the context we omit the subscript and call $\mathcal{C}$ an $[n, k]$ code.

In the following, we will refer to a k-dimensional, linear code of length n with minimum distance d as an $[n, k, d]$ code. For linear codes, it is easy to see that their minimum distance is equal to their minimum weight.

Lemma 2.3. *For an $[n, k, d]$ code, the minimum weight is equal to its minimum distance.*

Proof. Let w denote the minimum weight of $\mathcal{C}$. Since the code's minimum distance is d, there exist two distinct codewords $\mathbf{a}$ and $\mathbf{b}$ such that $d(\mathbf{a}, \mathbf{b}) = d$. Their difference $\mathbf{a} - \mathbf{b}$ will therefore have weight d so that $w \leq d$. Conversely, if a codeword $\mathbf{a}$ has weight w, then $d(\mathbf{0}, \mathbf{a}) = w \geq d$. $\square$

Linear codes can be described in a very compact manner, by means of a *generator matrix*:

Definition 2.3.1 (Generator matrix). A *generator matrix* G for a linear $[n, k]$ code $\mathcal{C}$ is a $k \times n$ matrix for which the rows form a basis of $\mathcal{C}$, i.e., such that

$$\mathcal{C} = \{ \mathbf{v}G \mid \mathbf{v} \in \mathbb{F}_q^k \}. \tag{2.4}$$

Note that the generator matrix of a code simultaneously describes the encoding procedure for an information word $\mathbf{v} \in \mathbb{F}_q^k$. Alternatively, one can describe a linear code $\mathcal{C}$ by means of a *parity-check matrix*: the parity-check matrix describes linear relations that any codeword must satisfy. A word $\mathbf{c} \in \mathbb{F}_q^n$ is in $\mathcal{C}$ if and only if it fulfills the relation $H\mathbf{c}^T = \mathbf{0}$ where H is a $(n-k) \times n$ parity check matrix for $\mathcal{C}$. Neither the generator matrix G, nor the parity check matrix H of a code are unique.

Now that we have seen the encoding procedure of some input data $\mathbf{v} \in \mathbb{F}_q^n$ (apply G from the right), we will give a short overview on some common decoding procedures as well as their relation to error correction. Given a received word $\mathbf{x} \in \mathbb{F}_q^n$, decoding aims at finding the underlying codeword $\mathbf{c} \in \mathbb{F}_q^n$. One general procedure that is often used is called *minimum-distance decoding*, in which one picks the codeword $\mathbf{c}$ that is closest in Hamming distance to the received word $\mathbf{x}$ (see the proof of Theorem 2.2.1) and therefore the most likely to be the original codeword. In case that two or more codewords have the same distance to the received word, one has to settle on a convention on how to break such ties. For linear codes, there exists another, more efficient procedure called *syndrome decoding*.

Definition 2.3.2 (Syndrome). Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be an $[n, k, d]$ linear code with parity check matrix H. Then the *syndrome* of a word $\mathbf{x} \in \mathbb{F}_q^n$ is defined as $H\mathbf{x}$.

Note that the syndrome of any codeword $\mathbf{c} \in \mathcal{C}$ is the all zero vector, $\mathbf{0}$. In the setting of linear codes, an error is modelled as adding an *error pattern* $\mathbf{e} \in \mathbb{F}_q^n$ to the original codeword $\mathbf{c} \in \mathcal{C}$, the resulting received word being $\mathbf{x} = \mathbf{c} + \mathbf{e}$. Hence, upon receiving $\mathbf{x} = \mathbf{c} + \mathbf{e}$ we find for the syndrome that

$$H\mathbf{x} = H(\mathbf{c} + \mathbf{e}) = H\mathbf{c} + H\mathbf{e} = \mathbf{0} + H\mathbf{e} = H\mathbf{e} \quad \in \mathbb{F}_q^{n-k}.$$

We find the corresponding error pattern $\mathbf{e}$ with the lowest weight and recover $\mathbf{c} = \mathbf{x} - \mathbf{e}$. One can most easily see why minimum-distance decoding is rather inefficient compared to syndrome decoding by looking at the size of the lookup tables that are used for both decoding procedures. For simplicity, consider a binary linear code. Under the assumption that at most $t := \lfloor \frac{d-1}{2} \rfloor$ errors occured we find that there are $\sum_{i=0}^{t} \binom{n}{i}$ possible error patterns. To determine which codeword a received word corresponds to, we therefore need to search a table of size $|\mathcal{C}| \sum_{i=0}^{t} \binom{n}{i}$ filled with the words obtained from adding any possible error pattern to any codeword. By contrast, syndrome decoding requires a lookup in a table of size $2 \sum_{i=0}^{t} \binom{n}{i}$, containg each possible error pattern and its syndrome.

To illustrate syndrome coding, consider the binary linear $(7, 4)$ *Hamming-code*, named after its inventor Richard Hamming ([Ham50]), defined by the generator matrix G and corresponding parity check matrix H,

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

The code consists of the codewords $\mathcal{C} = \{0000000, 1110000, 101100, 0111100, 0101010, 1011010, 1100110, 0010110, 1101001, 0011001, 0100101, 1010101, 1000011, 0110011, 0001111, 1111111\}$ as can be verified by computing $\mathbf{v}G$ for all $\mathbf{v} \in \{0,1\}^4$. The code has minimum distance 3, and can therefore correct at most one error. Now assume the word $\mathbf{r} = (0110110)$ is received. By the above consideration, $\mathbf{r} = \mathbf{c} + \mathbf{e}$ for some codeword $\mathbf{c}$ and some error $\mathbf{e}$, and we see that $H\mathbf{r}^T = H\mathbf{e}^T = (010)^T$. Searching the list of error syndromes, we find that the error with matching syndrome and minimum weight is given by $\mathbf{e} = 0100000$. It is therefore most likely to occur, so we correct the error to $\mathbf{c} = \mathbf{r} + \mathbf{e} = 0010110$ and recover the original message 1110. In Section 3.3, we will see that the (7,4) Hamming code can in fact be used for quantum error correction.

To conclude this section, we present a simple upper bound on the minimum distance of a linear code. The upper bound has been proven by Singleton in [Sin64] and is therefore referred to as the *Singleton bound*.

Theorem 2.3.1 (Singleton bound for linear codes). *Let $\mathcal{C}$ be an $[n,k,d]$ linear code over a finite field $\mathbb{F}$ with q elements. Then the Singleton bound implies that $d \leq n - k + 1$.*

Proof. See, e.g., [Sin64]. □

We have already seen one example of a linear code that attains the Singleton bound, namely the binary repetition code. We have seen that it has minimum distance $d = n$ and dimension $k = 1$. The above example, however, does not attain the bound: it has minimum distance $d = 3$ and $n - k = 7 - 4 = 3$. Singleton introduced the name *maximum-distance separable codes* (or *MDS codes*) to denote codes that attain this upper bound, i.e., any $[n, k, n - k + 1]$ code. In the next section, we will see another important family of codes, so-called *Reed-Solomon codes*, that are MDS codes and have a strong connection to Shamir's scheme.

At this point, we have solely focussed on introducing basic concepts of error-correction and have disregarded detailled analysis of time complexity or optimisation of encoding and decoding procedures as well as space efficiency, that is, the tradeoff between storage constraints (the length n of the codewords) and necessary redundancy in the codeword to achieve error-correcting capacities. Spielman's survey on the complexity of error-correcting codes, [Spi97], gives a more detailed account of these issues. The main focus of this section was to introduce the techniques and concepts of classical error correction that will be useful when discussing secret sharing schemes and that we will later see in similar fashion when considering quantum error correction.

2.4 Constructing secret sharing schemes from ECCs

Having covered the basics of error-correcting codes, we will now move on to study the connection between ECCs and secret sharing schemes, in particular the construction of secret sharing schemes from ECCs. We can construct a code that corrects errors on *known* positions, also known as *erasure errors*, from a (t, n) secret sharing scheme as follows:

consider the vector of n shares of a secret s as a codeword; then if less than $n-t$ known positions are affected by an error, the remaining greater or equal to t shares suffice for reconstruction of the initial secret and therefore we can correct up to $n-t-1$ errors on known positions. What is of more interest to us is the opposite direction: how can we use the more established theory and existing constructions of error-correcting codes to construct (efficient) secret sharing schemes?

It can easily be seen that not all ECCs are suitable for secret sharing. Consider for example the binary repetition code $b \mapsto b^n$ and treat each coordinate as a share of a sharing of a secret bit b. This code can correct up to $\lfloor \frac{n-1}{2} \rfloor$ arbitrary errors and $n-1$ erasure errors, and therefore any single share is sufficient to reconstruct and reveal the secret. While such a construction formally satisfies the definition of a secret sharing scheme, it is of little use in practice due to its poor privacy guarantees. Other codes, however, appear to be more suitable. Already in 1981, McEliece and Sarwate were the first to notice the connection between error-correcting codes and secret sharing schemes. In [MS81] they point out the relation between Shamir's scheme, which we outlined in Definition 2.1.6, and Reed-Solomon error-correcting codes and that there are several advantages of discussing the former in context of the latter.

A Reed-Solomon (RS) code $\mathcal{C}$ over a finite field $\mathbb{F}$ with q elements is defined by a code length n and dimension k such that $1 \leq k \leq n \leq q$ and some $n \leq q$ fixed, pairwise distinct values $\alpha_1, \dots, \alpha_n \in \mathbb{F}$. Then the corresponding RS code is given by

$$\mathcal{C} = \{ \, (f(\alpha_1), \dots, f(\alpha_n)) \mid f \in \mathbb{F}[x], \ \deg(f) \leq k-1 \}. \tag{2.5}$$

A word $\mathbf{a} = (a_0, \dots, a_{k-1}), a_i \in \mathbb{F}^k$ is encoded as the evaluations of the polynomial $f_{\mathbf{a}}(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} \in \mathbb{F}[x]$ at the evaluation points α_i,

$$\mathbf{a} \mapsto (f_{\mathbf{a}}(\alpha_1), \dots, f_{\mathbf{a}}(\alpha_n)). \tag{2.6}$$

Note that each such polynomial $f_{\mathbf{a}}$ has at most $k-1$ zeros, so every non-zero codeword in $\mathcal{C}$ has weight at least $n-k+1$. By Lemma 2.3 and the Singleton bound, this implies that $\mathcal{C}$ is an $[n, k, n-k+1]_q$ code and therefore an MDS code.

Now any codeword in such an RS code $\mathcal{C}$ can be viewed as a secret sharing of the secret $s = a_0$. If at least k of the shares are known, the codeword $\mathbf{c} = (f_{\mathbf{a}}(\alpha_1), \dots, f_{\mathbf{a}}(\alpha_n))$ and therefore s can be recovered by using an errors-and-erasures algorithm. Details of the aforementioned algorithm, as well as Reed-Solomon codes can be found for example in [McE02].

The above construction was generalised by Brickell in [Bri89] in terms of vector spaces. We adopt the equivalent formulation proposed by [RD96] for arbitrary linear $[n, k]_q$ error-correcting codes: let G be the generator matrix of an $[n, k]_q$ linear code, i.e., G is an $k \times n$ matrix of rank k over $\mathbb{F}_q$, and denote its columns by $G_1, \dots, G_n$. In order to secret-share $s \in \mathbb{F}$, sample $v_2, \dots, v_k \in \mathbb{F}_q$ uniformly at random. Define $\mathbf{v} := (s, v_2, \dots, v_k)$ and compute the corresponding codeword $(c_1, \dots, c_n) = \mathbf{v} \cdot G$. Then distribute the codeword by giving each participant a share c_i. Any set of participants $\{T_{j_1}, \dots, T_{j_m}\}$ can recover the secret s if the unit vector $\mathbf{e}_1 = (1, 0, \dots, 0)^T$ is a linear combination of $G_{j_1}, \dots, G_{j_m}$: if

$$\mathbf{e}_1 = \sum_{i=1}^{m} x_i G_{j_i}$$

for some $x_i \in \mathbb{F}$, then it follows that

$$s = \mathbf{v}\mathbf{e}_1 = \sum_{i=1}^{m} x_i \mathbf{v} G_{j_i} = \sum_{i=1}^{m} x_i c_{j_i}.$$

In [Bri89], Brickell further shows that the above condition indeed is necessary and sufficient, and that the resulting scheme is a perfect secret sharing scheme realising the access structure defined by that condition. Adding two secrets shared using the above construction as $[s] = (s, v_2, \dots, v_k) \cdot G$ and $[s'] = (s', v_2', \dots, v_k') \cdot G$, we furthermore obtain a secret sharing of $[s + s']$ with uniformly random $v_i + v_i' \in \mathbb{F}$, so that the construction in fact gives a linear secret sharing scheme.

In [RD96], Renvall and Ding give an overview of security guarantees regarding the above construction and show that using another, slightly more involved construction can be used to obtain a (k, n) threshold scheme from any $[n+1, k, n-k+2]$ MDS code. For the special case of Reed-Solomon codes we have seen above that an $[n, k]$ RS code defines a (k, n) threshold scheme. Additionally, Nikova and Nikov note in [NN09] that every (k, n) linear threshold secret sharing scheme is equivalent to some $[n+1, k+1]$ MDS code. A similar equivalence is also shown in [CDN15], Theorem 11.107. In that same book, the authors study various constructions of secret sharing schemes from linear codes. One particular noteworthy result relates the *dual distance* of a linear code to its privacy as a secret sharing scheme. We first define the *dual* of some linear code $\mathcal{C}$.

Definition 2.4.1 (Dual code). The *dual code* of an $[n, k]$ linear code $\mathcal{C}$ is defined as

$$\mathcal{C}^{\perp} = \{\mathbf{x} \in \mathbb{F}_q^n \mid \langle \mathbf{x}, \mathbf{c} \rangle = 0 \ \forall \mathbf{c} \in \mathcal{C}\},$$

where the inner product is given by the dot product, $\langle \mathbf{x}, \mathbf{y} \rangle := \sum_{i=1}^{n} x_i y_i$ for $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{F}_q^n$, $\mathbf{y} = (y_1, \dots, y_n) \in \mathbb{F}_q^n$.

[CDN15] establishes the following relation between secret sharing schemes obtained from linear codes and the code's dual distance:

Theorem 2.4.1 ([CDN15], Theorem 11.91). *Let n, k be positive integers and let $\mathcal{C} \subseteq \mathbb{F}^{n+k}$ be an $\mathbb{F}$-linear code. If*

$$d_{\min}(\mathcal{C}) \geq k+1 \quad \textit{and} \quad d_{\min}(\mathcal{C}^{\perp}) \geq k+1,$$

then $\mathcal{C}$ is a linear secret sharing scheme for $\mathbb{F}^k$ over $\mathbb{F}$ with t-privacy and r-reconstruction, where

$$t = d_{\min}(\mathcal{C}^{\perp}) - k - 1 \quad \textit{and} \quad r = n - d_{\min}(\mathcal{C}) + k + 1.$$

The above result also explains the poor privacy guarantees of the repetition code, when treating the coordinates of a codeword b^{n+1} as shares: the repetition code's dual is given by the *parity check code* that consists of all length $n+1$ binary words of weight 2. Thus, the dual's distance is given by 2 and by the above theorem, we obtain $t = 2 - 1 - 1 = 0$ privacy.

In this section we have seen the close relation of error-correcting codes and linear secret sharing schemes. In particular we observed the equivalence of Shamir's scheme, one of the most widely use secret sharing schemes, to Reed-Solomon codes. The following sections will treat the quantum analogues of the above concepts: we will introduce quantum secret sharing and error correction and discuss the differences and similarities to their classical counterparts and will finally see how, just like in the present section, there is a strong connection between quantum error correction and secret sharing.

2.5 Quantum analogues

Since quantum secret sharing draws heavily on techniques from quantum error correction, we will first introduce the latter before discussing the former. During the introduction of quantum secret sharing schemes we will then see the connection to quantum error correction.

2.5.1 Quantum error-correcting codes

In order to harness the power of quantum computations, it is necessary to be able to protect quantum information from noise that could occur during storage, transmission or processing of the quantum information. In the early stages of research in quantum computation, researchers feared that it might be impossible to protect the fragile physical systems used to implement the quantum computations from too much noise. In modern classical computers, the failure rate is typically kept below one error per 10^{17} operations. Classical computers use error-correcting codes based on the theory outlined in Section 2.2 to preserve information despite noise. For quantum codes, three main obstacles present themselves that gave rise to the doubts that researchers in the field initially had:

- *No-cloning:* due to the quantum no-cloning theorem, simple constructions like the repetition code in the classical world are impossible.
- *Measurement destroys quantum information:* note that even if cloning of quantum states was possible, measuring the received quantum data generally disturbs the state and thereby partially destroys information; this is in sharp contrast to classical ECC, where received data is observed during the error correction process.
- *Continously many errors:* in contrast to classical information encoded in bits, quantum information can be found in continously many states and continously many errors are possible; protecting information against such an infinite range of possible errors appears to require infinite resources.

We will not delve into the details of quantum error correcting codes or too many examples. Instead, we will give an overview of the necessary formalism and present some of the main results that we will need for later sections. The results presented here are standard results from the theory of quantum error correction and are based on [NC16], [Got09] and [Pre99].

Generally, quantum error correction (QEC) resembles its classical counterpart in various ways. In order to protect quantum information from noise, a state is encoded by a unitary operation into a codeword in a larger Hilbert space. The original information is said to be contained in the *logical qubits* while the encoded state is said to consist of *physical qubits*. The distinction between logical and physical qubits will also play an imporant role in the theory of MPQC, Chapter 3. The code is then defined as the set of codewords and it is a linear subspace of the Hilbert space. To start the error correction procedure, a measurement is performed on the received state in order to determine what, if any, error occured. Similarly to Section 2.2, this information is called the *error syndrome*. Based on the outcome of that measurement, a *recovery procedure* is applied that maps back the erroneous word to the initial codeword, which can then be decoded to the underlying information word. Formally, a quantum error-correcting code can be defined as follows, according to [Got06]:

Definition 2.5.1 (Quantum error-correcting code). Let $\mathcal{H}$ be a 2^n-dimensional Hilbert space (consisting of n qubits), and $\mathcal{C}$ be a k-dimensional subspace of $\mathcal{H}$. Then $\mathcal{C}$ is an $((n,k))$ *quantum error-correcting code (QECC)* correcting the set of errors $\mathcal{E} = \{E_a\}$ if and only if there exists a CPTP map $\mathcal{R}$ such that $(\mathcal{R} \circ E_a)(|\psi\rangle) = |\psi\rangle$ for all $E_a \in \mathcal{E}$, $|\psi\rangle \in \mathcal{C}$.

The map $\mathcal{R}$ is also called the *decoding* or *recovery operation*. Note that similarly, quantum error correction can be described over arbitrary sized quantum systems instead of qubits, and that qubit-based quantum error-correcting codes are sometimes called *binary*. Before discussing an example of a simple error correcting code, we will address the problem of the continous spectrum of possible quantum errors.

Modelling quantum errors

Quantum errors are typically assumed to occur on individual qubits independently. This corresponds to the assumption in classical error correction that errors occur independently on individual bits with a certain probability. Common errors that are considered are bit flip and phase flip errors as well as a combination of the two. These correspond to the unitaries X, Z and $Y = iXZ$ that we have already seen in Section 1.1. With the help of the following example we will see that being able to correct these two types of errors turns out to be very powerful.

Shor's 9-qubit code

Shortly after his seminal work on a quantum algorithm for efficient prime factorization [Sho94], Peter Shor addressed the problem of quantum error correction and introduced one of the first quantum error-correcting codes, also known as *Shor's 9-qubit code*. We will discuss this example of a quantum error-correcting code, as it provides insight into the various parts and techniques of QEC and in a sense is similar to the repetition code we have seen earlier. To start with, suppose we are given a state $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$. The encoding procedure is defined on the computational basis states and maps

$$|0\rangle \mapsto |\bar{0}\rangle := \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle), \tag{2.7}$$

$$|1\rangle \mapsto |\bar{1}\rangle := \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle), \tag{2.8}$$

where implicitly, eight ancilla qubits have been added to the initial state. The encoding of $|\phi\rangle$ then is a linear superposition of these two states, $\alpha|\bar{0}\rangle + \beta|\bar{1}\rangle$, and $|\bar{0}\rangle$ and $|\bar{1}\rangle$ form a basis of the code space. Note that this encoding does not violate the no-cloning theorem,

$$\alpha|\bar{0}\rangle + \beta|\bar{0}\rangle \neq [\alpha(|000\rangle + |111\rangle) + \beta(|000\rangle - |111\rangle)]^{\otimes 3}.$$

Shor's code protects against single bit flip and phase flip errors or a combination of the two, as can be seen by the following. A single bit flip error can be thought of as applying a 9-fold tensor product of single-qubit unitaries, eight of which are the identity matrix, the other one being an X-gate. For example, $I \otimes X \otimes I^{\otimes 7}$ is a bit flip on the second qubit and we sometimes abbreviate it as X_2. Assume a bit flip occurred on the second qubit. To detect such an error, we examine the three blocks of qubits block by block. For example, we examine the first block of qubits by comparing two subsequent qubits at a time. We first measure the observable $Z_1Z_2 = (|00\rangle\langle 00| + |11\rangle\langle 11|) - (|01\rangle\langle 01| + |10\rangle\langle 10|)$ to determine whether a bit flip has occurred, and then do the same with the observable Z_2Z_3. Both measurements return the outcome -1 and since we assumed that at most one error occured, we conclude that the second qubit was flipped. We correct the error by flipping the erroneous qubit back to its original state. Similarly, we can correct bit flip errors on any of the nine qubits in a codeword.

For phase flip errors, the three blocks of qubits are compared with each other, as we observe that a phase flip error on any single qubit within a block has the same effect on that block. The corresponding syndrome measurement that determines the block that has been affected by a phase flip error is given by the observables $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$.

The error correction procedures for bit flips and phase flips are independent so that by applying both, we can also correct their product XZ. We can in fact generalise even more and claim that Shor's 9-qubit code protects against arbitrary single-qubit errors. Suppose an arbitrary error affected the second qubit only - the error can have any form, from a small rotation up to replacing the qubit by some generic state. Recall from Chapter 1 that such noise can be described by a quantum channel $\mathcal{E}$. We decompose $\mathcal{E}$ into an operator-sum representation with operation elements $\{E_i\}$ according to Definition 1.1.5; then the encoded state $|\bar{\phi}\rangle = \alpha|\bar{0}\rangle + \beta|\bar{1}\rangle$ after the error occured is given by $\mathcal{E}(|\bar{\phi}\rangle\langle\bar{\phi}|) = \sum_a E_a |\bar{\phi}\rangle\langle\bar{\phi}| E_a^\dagger$. It is easiest to analyse the error correction procedure on a single operation element E_a. We noted in Section 1.1 that the Pauli matrices form a basis of all single-qubit matrices (not only unitaries), so E_a can be decomposed into a linear combination of I, X, Z and Y gates affecting the second qubit only,

$$E_a = e_{a0}I + e_{a1}X_2 + e_{a2}Z_2 + e_{a3}X_2Z_2. \tag{2.9}$$

$E_a|\bar{\phi}\rangle$ therefore is a superposition of $|\bar{\phi}\rangle$, $X_2|\bar{\phi}\rangle$, $Z_2|\bar{\phi}\rangle$ and $X_2Z_2|\bar{\phi}\rangle$. Measuring the error syndrome collapses this superposition into one of these states which can then be corrected by applying the appropriate gate, and similarly for the other operation elements E_i, which proves the claim. The above does not only hold for the Pauli group but any linear combination of correctable errors, as stated in Theorem 2 of [Got09]:

Theorem 2.5.1 (Discretisation of quantum errors). *If a quantum code corrects errors E and F, then it also corrects any linear combination of the two.*

The importance of this result cannot be understated, as it limits the amount of errors that the code needs to be able to correct to a finite set, for example merely the four Pauli gates. For codes that can correct errors in the Pauli group, i.e., tensor products of unitary gates, all of which are single qubit Pauli gates, this implies the following: define the *weight* of a Pauli error as the number of tensors in the tensor product different from the identity, such that, e.g., $I \otimes X \otimes Y \otimes I$ has weight 2. Using Theorem 1.1.1 and Theorem 2.5.1, we obtain the following corollary.

Corollary 2.5.1. *If a quantum error-correcting code corrects any weight-t Pauli errors then it corrects arbitrary t-qubit errors.*

Hence, we see that quantum error correction indeed is very similar to classical error correction in the sense that it is sufficient to protect against a certain finite set of errors. Moreover, we use the error syndrome of a received state to detect what error occured and how to correct that error. It is important to note that the syndrome measurement in QEC only reveals information on the error that occured but not on the encoded information and therefore does not cause any disturbance.

The interested reader is referred to [Got09] or Section 10.3 in [NC16] for a more detailled treatment of general errors as well as errors that occur on multiple qubits. What is more important in our context is a formulation of requirements on error-correcting codes in general, as these will play an important role in the context of quantum secret sharing schemes.

Necessary and sufficient conditions

In [Got09], necessary and sufficient conditions for quantum error-correcting codes are given that simplify the analysis of such codes and in particular help in the analysis of quantum secret sharing schemes, as we will see in the following section. Originally, these are due to [KL97].

Theorem 2.5.2. *Let $\mathcal{E}$ be a linear space of errors acting on a Hilbert space $\mathcal{H}$. Then a subspace $\mathcal{C}$ of $\mathcal{H}$ is a quantum error-correcting code correcting the errors in $\mathcal{E}$, if, and only if, for all errors $E_a, E_b \in \mathcal{E}$ we have*

$$\langle\psi_i| E_a^\dagger E_b |\psi_j\rangle = C_{ab}\delta_{ij}. \tag{2.10}$$

where C_{ab} is a Hermitian matrix and depends only on the errors E_a, E_b, not on the $|\psi_i\rangle, |\psi_j\rangle$, and the $\{|\psi_i\rangle\}$ form a basis of $\mathcal{C}$.

Proof. See, e.g., [Got09] or [Pre99]. □

It is important to note that the above conditions for quantum error correction hold only for *exact* error correction. If we only require the reconstructed state to be close to the original codeword, then these conditions fail. A detailed treatment of *approximate quantum error correction* is out of the scope of this work but can be found in [SW02] and [CvL14]. In Section 2.6 we will see how this distinction can be exploited to improve the privacy guarantees of quantum secret sharing schemes.

For quantum error-correcting codes, several bounds on the codes' parameters have been derived similar to those in the classical case. Here, we outline some of them. One important bound on the parameters of QECC codes concerns a special subset called *non-degenerate* codes. We call a QECC non-degenerate, if any two errors in the set of correctable errors, $\mathcal{E}$, can be perfectly distinguished, that is, if every two errors E_a, E_b in a $\mathcal{E}$ map each basis codeword $|\psi\rangle$ to a mutually orthogonal subspace of $\mathcal{C}$. Otherwise, there exist two error operators that act the same on any codeword, and we call such quantum codes *degenerate*. Formally, it can be shown a quantum error-correcting code is degenerate, if and only if C_{ab} is singular, that is, if it has vanishing eigenvalues ([Gai08], p. 74).

An example of a degenerate code is Shor's 9 qubit code, as, for example, Z errors on a single qubit within any 3-qubit block of a codeword have the same effect on the overall codeword. The concept of degeneracy is in contrast to classical codes, for which every error has to be mapped to an orthogonal subspace: in a sense, degenerate quantum codes allow to "pack more information" ([NC16], Section 10.3.3) into the code space.

For non-degenerate codes, the *quantum Hamming bound* applies: assume that a quantum code encodes k qubits into n qubits and corrects any number less or equal to t of errors. For each $j \leq t$ there are $\binom{n}{j}$ possible locations on which the errors may occur. On each such location there are three possible errors, the Pauli matrices X, Y and Z, that may occur resulting in a total of 3^j possible errors. Overall, this amounts to

$$\sum_{j=0}^{t} \binom{n}{j} 3^j$$

possible errors. Since we assumed the code to be non-degenerate, each of these errors must correspond to a 2^k-dimensional, orthogonal subspace of the 2^n-dimensional total space available. This leads to the inequality

$$2^k \sum_{j=0}^{t} \binom{n}{j} 3^j \leq 2^n, \tag{2.11}$$

known as the quantum Hamming bound. For Shor's 9-qubit code we have $t = k = 1$ and $n = 9$, clearly satisfying the bound. However, we mentioned that the 9-qubit code is an example of a degenerate code, and in fact [DSS98] is an example of a degenerate code that violates the quantum Hamming bound.

Following [Got09], we furthermore define the *distance* d of a quantum error-correcting code that corrects up to t errors as $d = 2t + 1$, as it takes d errors to change one codeword in $\mathcal{C}$ to another. A code that encodes k qubits into n qubits with distance d is called an $[[n, k, d]]$ code, the double brackets emphasizing the use of a *quantum* code. For example, Shor's code is a $[[9, 1, 3]]$ code. Furthermore, [EM96] showed that $[[n, k, d]]$ codes exist whenever the stricter lower bound on n,

$$2^k \sum_{j=0}^{d-1} \binom{n}{j} 3^j \leq 2^n, \tag{2.12}$$

is satisfied. This is known as the *quantum Gilbert-Varshamov* bound. Note the difference to (2.11), where the sum only goes to up to t instead of $2t$. Last but not least, a quantum analogue to the Singleton bound that we presented in Section 2.2 was proven by Knill and Laflamme in [KL97] that applies even to degenerate $[[n, k, d]]$ codes,

$$2d \leq n - k + 2. \tag{2.13}$$

Similar to the classical case, a QECC that satisfies the *Knill-Laflamme-* (or *quantum Singleton*) *bound* is called a *quantum MDS code*. Shor's 9-qubit code, for example, does not attain the quantum Singleton bound, as $6 < 9 - 1 + 3 = 11$, but in Section 3.3 we will encounter a QECC that does. A large amount of work on constructing quantum MDS codes exists (see, e.g., [RGB04]) but it seems like, at the time of writing, no general construction for quantum MDS codes exists.

While a full proof for the Knill-Laflamme bound can be found in [KL97], we note that the proof for codes encoding a single qubit, $k = 1$, provides some interesting insight into employing quantum error-correcting codes as secret sharing schemes. For this partial proof we need a specific kind of error, in particular studied by Grassl et al. in [GBP97]:

Definition 2.5.2 (Quantum erasure error). An *erasure error* is a general error on a *known* position (qubit). While the type of error that occured is unknown, we can think of it as replacing the coordinate with some state $|err\rangle$ orthogonal to the regular Hilbert space.

Using the above conditions for a quantum error-correcting code, Grassl et al. proved the following relation between general errors and erasure errors.

Theorem 2.5.3. *A quantum error-correcting codes corrects t arbitrary single-qubit errors if and only if it corrects* $2t$ *erasure errors.*

Proof. Consider condition (2.10). For an error-correcting quantum code of length n that corrects t arbitrary single-qubits errors, all error operators E_a have at most weight t, i.e., their tensor factors are equal to the identity in at least $n - t$ positions. For a quantum code that corrects t erasure errors, these at most t positions are known, i.e., occur on the same positions, so the product $E_a^\dagger E_b$ also has weight at most t. Since the E_a form a basis for arbitrary weight-t errors, the product $E_a^\dagger E_b$ can again be written as a linear combination of the $\{E_a\}$. Hence, Eq. (2.10) simplifies and we get a necessary and sufficient condition for quantum codes that correct t erasure errors,

$$\langle \psi_i | E_a | \psi_j \rangle = C'_a \delta_{ij}, \tag{2.14}$$

the E_a being exactly the same E_a that form a basis for arbitrary weight-t single-qubit errors. From this relation we can immediately see that Eq. (2.10) for t-error operators E_a implies (2.14) for $2t$-error operators and vice versa. □

Theorem 2.5.4 (Knill-Laflamme bound, [KL97]). *Any $[[n, k, d]]$ QECC (encoding k qubits into n qubits with distance $d = 2t + 1$) must satisfy $2d \leq n - k + 2$.*

Proof. By Theorem 2.5.3, an $[[n, k, d]]$-code corrects $d - 1 = 2t$ erasure errors. Now assume the quantum Singleton bound was violated, which for the case $k = 1$ implies $n \leq 2d - 2$. Then we can split the qubits of a codeword encoding a single qubit into two sets, each containing at most $d - 1$ qubits. Both these sets can be considered as a codeword from which at most $d - 1$ qubits are missing in known positions, i.e., where at most $d - 1$ erasure errors occured. We can correct for these errors in both sets independently, thereby reconstructing two copies of the encoded qubit. This, however, violates the no-cloning theorem. A general proof can be found in [KL97]. □

Note that by the same reasoning, Theorem 2.5.3 implies a general bound on quantum error-correcting codes: no quantum error-correcting code can correct $n/4$ arbitrary errors or more, as this would imply the ability to correct $n/2$ erasure errors or more, again colliding with no-cloning. In the next section we will see that in a similar fashion, quantum no-cloning imposes a lower bound on the threshold in quantum secret sharing schemes.

2.5.2 Quantum secret sharing

Quantum secret sharing (QSS) describes the theory of sharing classical or quantum information using quantum computations. First steps towards a theory of quantum secret sharing have been taken in [HBcvB99], while Cleve, Gottesman and Lo presented the first fundamental results in their seminal paper in 1999, [CGL99]. The following introduction into the topic will be based mainly on [CGL99] as well as a subsequent survey by Gottesman, [Got00]. Before discussing fundamental results and constructions of quantum secret sharing, we have to specify the setting for quantum secret sharing that we will consider.

Similarly to the introduction in Section 2.1, a (quantum) secret will be encoded into shares that the dealer then distributes among a number of participants in the quantum secret sharing protocol. In the theory of quantum secret sharing schemes, it is generally assumed that the secret is a pure quantum state $|\psi\rangle$, such as a single qubit, that is possibly unknown to the dealer. However, better security and efficiency results can be achieved when secret sharing classical information using QSS schemes, as, e.g., Gottesman points out in [Got00]. As in Section 2.1 on classical secret sharing, we assume private point-to-point (quantum) communication channels and focus on the security of the proposed scheme itself. Furthermore, we will consider information theoretic security and do not impose any computational restrictions on the adversary. We adopt the definition given in [CGL99] and say that an unauthorised set of shares does not contain any information about the secret if its partial trace (Def. 1.1.4) is independent of the value of the secret. A quantum information theoretic approach to quantum secret sharing schemes is given in [IMN$^+$05] that confirms the results of [CGL99]. For the purpose of this introduction on QSS, [CGL99] is sufficient.

Similarly to classical secret sharing schemes, we will focus on threshold schemes with n participants, any number $t \leq n$ of which can reconstruct the secret perfectly, while any number $\leq t-1$ cannot deduce any information at all about the secret. We denote such a scheme as a $((t,n))$ threshold scheme, the double brackets indicating the use of quantum information. The secret is an arbitrary pure quantum state and the shares consist of a number of qubits or possibly higher-dimensional states.

A first example of a $((2,3))$ threshold scheme, due to [CGL99], is the following: let the secret state $|\phi\rangle$ be an arbitrary three-dimensional quantum state (sometimes denoted a *qutrit*), $|\phi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle$ with $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$. Then encode the secret into shares of three qutrits each by applying the (unnormalised) mapping

$$\begin{aligned} \alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle \quad \mapsto \quad & \alpha(|000\rangle + |111\rangle + |222\rangle) \\ & +\beta(|012\rangle + |120\rangle + |201\rangle) \\ & +\underbrace{\gamma(|021\rangle + |102\rangle + |210\rangle)}_{:=|\psi\rangle}, \end{aligned} \tag{2.15}$$

where we implicitly appended eight ancilla qutrits to $|\psi\rangle$. We denote the three participants by A, B and C and each receives one of the three qutrits as a share. To see that for this scheme, each individual participant's reduced density matrix is a complete mixture of the states $|0\rangle\,,|1\rangle$ and $|2\rangle$, we examplarily trace out the second and third party. Define the global state of the shares as $|\psi\rangle$. Then the state held by player 1 is given by

$$\begin{aligned} \mathrm{Tr}_{BC}\left[\,|\psi\rangle\langle\psi|\,\right] = \mathrm{Tr}_{BC}\Big[& |\alpha|^2(|000\rangle\langle 000| + |000\rangle\langle 111| + \ldots + |222\rangle\langle 222|) \\ & +\alpha\bar{\beta}\,(|000\rangle\langle 012| + |000\rangle\langle 120| + \ldots + |222\rangle\langle 201|) \\ & +\ldots \\ & +|\beta|^2(|012\rangle\langle 012| + |012\rangle\langle 120| + \ldots + |201\rangle\langle 201|) \\ & +\ldots \\ & +|\gamma|^2(|021\rangle\langle 021| + |021\rangle\langle 102| + \ldots + |210\rangle\langle 210|)\Big] \\ = & \underbrace{(|\alpha|^2 + |\beta|^2 + |\gamma|^2)}_{=1}(|0\rangle\langle 0| + |1\rangle\langle 1| + |2\rangle\langle 2|) \\ = & \,|0\rangle\langle 0| + |1\rangle\langle 1| + |2\rangle\langle 2|\,, \end{aligned}$$

independent of the initial input amplitudes α, β and γ. Note again that, for the sake of readibility, we omitted normalisation factors. On the other hand, any two out of three players can reconstruct the initial state. For instance, if players A and B collaborate, they can (unitarily) add the (computational basis state) value of the first share to the second and afterwards add the value of the second share to the first, both calculations performed modulo 3. For details on how to perform such classical arithmetic operation unitarily, see, e.g., [VBE95]. A simple calculation shows that this results in the global state

$$(\alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle) \otimes (|00\rangle + |12\rangle + |21\rangle),$$

i.e., after these operations, player A holds the initial state. Note that it is impossible for both players to hold a copy of the secret after reconstruction due to no-cloning. Similarly, any other two players can obtain the secret jointly from their shares, showing that the scheme indeed is a $((2,3))$ quantum threshold scheme.

We can construct a $((2,2))$ secret sharing scheme from (2.15) by discarding, i.e., tracing out one share. This leaves the global state of the two shares in a mixed state, the encoding procedure being defined by mapping the pure basis states onto mixed states,

$$\begin{aligned}|0\rangle\langle 0| &\mapsto |00\rangle\langle 00| + |11\rangle\langle 11| + |22\rangle\langle 22| \\ |1\rangle\langle 1| &\mapsto |01\rangle\langle 01| + |12\rangle\langle 12| + |20\rangle\langle 20| \\ |2\rangle\langle 2| &\mapsto |02\rangle\langle 02| + |10\rangle\langle 10| + |21\rangle\langle 21| .\end{aligned}$$

Cleve et al. denote a QSS scheme that encodes pure state secrets into global pure states, such as (2.15), a *pure state schemes* while schemes for which the encoding of a pure state is in a globally mixed state, such as (2.16), are referred to as *mixed state schemes*. They also note that the technique of discarding a share generalises, resulting in the following theorem.

Theorem 2.5.5. *From any $((t, n))$ quantum secret sharing scheme with $t < n$, a $((t, n-1))$ scheme can be derived by discarding one share.*

As we have already hinted when discussing Theorem 2.5.4, processing quantum information imposes limitations on possible thresholds in quantum secret sharing schemes. In Section 2.1 we have shown that, e.g., Shamir's scheme can be used as a (t, n) threshold scheme for *any* $t \leq n$. For quantum schemes, the no-cloning theorem puts a lower bound on the threshold t.

Theorem 2.5.6 ([CGL99]). *No threshold quantum secret sharing scheme exists for $t \leq n/2$.*

Proof. We have already used a similar technique for proving the quantum Singleton bound. If a $((t, n))$ scheme existed for some $t \leq n/2$ then two disjoint subsets of participants could independently reconstruct a copy of an arbitrary secret quantum state. This effectively implements a cloning operation of arbitrary quantum states, violating the no-cloning theorem. □

Hereafter, the authors of [CGL99] proceed to show that beyond this lower bound on the threshold, essentially no restrictions on the parameters exist for $((t, n))$ QSS schemes. Since the constructions they use draw so heavily on quantum error-correcting codes, we will as well move on to discussing the relation between QEC and QSS before concluding this section with the main results of Cleve et al.

2.5.3 QSS and QEC

The relation between quantum error-correcting codes and secret sharing schemes is an asymmetric one: on one hand, every quantum secret sharing scheme in a sense is an error-correcting code, as the global state of shares can be viewed as a codeword that protects the encoded information against erasure errors in certain, known positions, just like we have noted in Section 2.4. What is more interesting for us, however, is the inverse direction. Not all QECCs are useful as secret sharing schemes, as the latter additionally require complete hiding of the secret encoded state. This is illustrated by the following example, due to [GBP97]. Consider the encoding

$$\alpha |0\rangle + \beta |1\rangle \quad \mapsto \quad \alpha(|0000\rangle + |1111\rangle) + \beta(|0011\rangle + |1100\rangle), \tag{2.16}$$

and assume each of the four qubits constitutes one share of the secret $\alpha |0\rangle + \beta |1\rangle$. In [GBP97] it is shown that this code can correct one erasure, thus allowing any three parties to reconstruct the secret. On the other hand, two colluding parties might very well learn *partial* information about the secret. For instance, holding the first and third qubit allows to perfectly distinguish the secrets $|0\rangle$ and $|1\rangle$, and, more generally, to obtain statistical information on the relative size of the amplitudes α and β. Hence, we see that one has to be careful when constructing QSS schemes from quantum codes. Overall, however, the theory of quantum error correction provides useful insight for quantum secret sharing. For example, the theory of QEC tells us that there does not exist a $((2,3))$ threshold scheme that encodes a single qubit into single-qubit shares, as this would constitute an error-correcting, three-qubit code that corrects single qubit erasure errors which is proven to be impossible in [GBP97]. By contrast, we have discussed above that a $((2,3))$ quantum secret sharing scheme exists that encodes qutrits into qutrit-shares. Hence, we see that the minimum dimension the shares of a quantum secret sharing scheme depends on the number of participants in the protocol. Moreover, we note that Gottesman shows in [Got00] that the dimension of each share of a quantum secret sharing scheme must be at least as large as that of the secret. A detailed study of such "efficiency"-properties of quantum secret sharing schemes, however, is outside of the scope of this work.

Cleve et al. ([CGL99]) furthermore make the following observation about the strong relation between QECCs and quantum secret sharing schemes:

Proposition 2.5.1. *Any quantum error-correcting code of length $2k-1$ that corrects $k-1$ erasure errors is a $((k, 2k-1))$ threshold scheme by treating each position in a codeword as a share in the secret sharing scheme.*

Proof. Suppose k shares are given. Then the remaining $k-1$ shares can be treated as erasures which can be corrected for in order to reconstruct the secret. On the other hand, suppose $\leq k-1$ shares are given. By the information-implies-disturbance principle ([BBM92]), any information that is gained from a measurement of these shares would disturb the information in the remaining k qubits and therefore lead to a contradiction. □

Hence, we see that, while for classical error-correcting codes, we need to determine the dual distance of the code to determine its privacy properties (Theorem 2.4.1), the nature of quantum information provides privacy "for free". Nonetheless, a theoretical investigation into a possible quantum analogue of the dual distance of a classical code that might explain this behaviour seems worthwile. Together with Theorem 2.5.3 and Theorem 2.5.5, the above theorem implies the following corollary:

Corollary 2.5.2. *A $((k,n))$ quantum threshold scheme can be constructed from a $[[2k-1,1,k]]$ error-correcting code for any $k > n/2$.*

In Section 2.4 we have discussed the equivalence between Reed-Solomon error-correcting codes and Shamir's scheme and the similarity of MDS codes and threshold schemes in general. For quantum schemes, Rietjens et al. proved equivalence between quantum MDS codes ((2.13)) and quantum threshold schemes in [RST05]:

Theorem 2.5.7. *A $((k, 2k-1))$ quantum secret sharing scheme can be translated into a $[[2k-1,1,k]]$ quantum MDS code and vice versa.*

Finally, Cleve et al. prove the existence of quantum secret sharing schemes for any $k > n/2$ by constructing suitable error-correcting $[[2k-1,1,k]]$ codes and applying the above corollary.

Theorem 2.5.8. *For any $k > n/2$, a $((k,n))$ threshold scheme exists.*

Beyond the relation between QEC and QSS that is exploited in the proof of the above theorem, another relation is notable, namely that of quantum and classical error-correcting codes. So-called *CSS codes*, owing their name to their inventors Calderbank, Steane and Shor, are constructed from linear ECCs and provide a rich and widely studied set of quantum codes. In Section 3.3 we will formally introduce these codes and see how they can be used for constructing secret sharing schemes and, eventually, a protocol for secure multi-party quantum computation.

In this section we have seen that, analogously to the classical case, there is a strong connection between quantum error-correction codes and secret sharing schemes. The use of quantum information in fact seems to simplify the construction of the latter from the former, as Corollary 2.5.2 and Theorem 2.5.7 suggest. This observation combined with the fact that quantum error-correcting codes have been studied far more extensively than QSS schemes explains the frequent use of quantum error-correcting codes when constructing MPQC protocols.

2.6 Feasibility of (quantum) secret sharing

Following the above introduction on classical and quantum secret sharing schemes, we will now discuss some general feasibility results in terms of the number of corrupted parties that a protocol can tolerate. We will see that here, too, the theory of error correction will prove useful in establishing and understanding the bounds on the parameters of the scheme. Besides (threshold) secret sharing, we will also discuss two stronger notions called *robust secret sharing* and *verifiable secret sharing* that are typically used to achieve security against active adversaries that might deviate from the procotol, in particular when constructing MP(Q)C protocols based on (quantum) secret sharing schemes.

The general focus will be kept on security in the information-theoretical model, and we will highlight the differences between quantum and classical protocols, as well as active and passive adversaries. In the information-theoretical model, the availability of secure and authenticated communication channels between each pair of participants in the protocol is assumed. This is often referred to as point-wise secure and authenticated communication. In some situations, the availability of a broadcast channel is additionally assumed, which allows participants to broadcast messages securely to all other parties, such that it is guaranteed that all parties receive the same message. We will mention whenever we make this assumption, as the availability of a broadcast channel is generally considered a strong assumption. We recall from the introduction that we consider a single adversary that corrupts a number t of parties out of the n participants in the protocol. The adversary has complete control over all corrupted parties' in- and outputs, as well as their actions. Throughout this work, and in particular in this section, we consider the adversary to be *static*, that is, to choose which parties to corrupt at the beginning of the protocol. Nonetheless, we remark that many of the results below are conjectured or even proven to hold in the presence of a so-called *adaptive adversary* that can choose which parties to manipulate during the protocol, but proving security in this model is generally considered a harder task. Finally, we assume a *non-rushing* adversary unless explicitly stated otherwise. In an interactive protocol, messages are exchanged round by round, and in each round a party might send and receive messages. A *rushing* adversary is one that is able to choose its messages after seeing the honest parties' messages *within the same round of communication*. In any scenario, the adversary is able to see past messages received by other parties, as long as the no-cloning theorem is not violated for quantum information. In the case of a secret sharing reconstruction protocol, these messages could for example be their shares.

First, results on the feasibility of classical secret sharing protocols will be presented alongside explanations of the adversarial model and the relevant literature. Then we will consider the quantum versions of these tasks and compile in a final table (Table 2.4) an overview of the feasibility of classical and quantum secret sharing schemes in different adversarial settings. As mentioned before, the focus on this work, including the present section, is put on information-theoretical security. For schemes that rely on computational assumptions, better feasibility results might be possible.

Feasibility of classical secret sharing

We recall that the literature on secret sharing that has been discussed in the previous section focussed on threshold schemes in which any number k of n participants can reconstruct the secret jointly from their shares, while any number of up to $k-1$ shares cannot obtain any information on the secret based on their shares. In this setting, we assumed the adversary to be passive, such that the corrupted parties try to derive information on the secret based on the shares in their joint possession, but do not deviate from the protocol and, in particular, do not modify their shares. We also assumed the dealer to honestly perform the share generation and distribution such that the shares correspond to a unique secret. For threshold secret sharing schemes, we have seen that essentially any number of passively corrupted parties smaller than n can be tolerated, since Shamir's scheme achieves this task. Clearly, one cannot expect any secrecy guarantees if all parties in the protocol are corrupted.

Before we start presenting the further feasibility results for classical secret sharing schemes, we will introduce some additional notation that will provide some consistency when discussing the limitations of secret sharing schemes in different adversarial settings. Recall from Section 2.1 that we denoted the set of participants in a secret sharing protocol by $\mathcal{T} = \{T_1, \dots, T_n\}$. We defined the access structure $\Gamma \subseteq \mathcal{P}(\mathcal{T})$ of a protocol to be the set of subsets of parties that are able to reconstruct the secret jointly, based on their shares, and we referred to elements in Γ as authorised sets. Naturally, we required Γ to be monotone such that any superset of an authorised set is authorised as well. We have so far focussed on threshold schemes that are characterised by a single threshold parameter $k \leq n$ such that any subset of parties larger or equal to k is able to reconstruct the secret based on their shares. We also noted that threshold schemes are perfect secret sharing schemes for which any unauthorised set is a privacy set, and we referred to k as the reconstruction parameter and $k-1$ as the privacy parameter. In particular, a passive adversary that corrupts any number of parties of cardinality less than k in a k-threshold scheme cannot obtain any information on the secret. Formally, we characterised an adversary by an antimonotone adversary structure $\mathcal{A} \subseteq \mathcal{P}(\mathcal{T})$ and for the case of (k,n)-threshold secret sharing schemes, we saw that the adversary is given by $\mathcal{A} = \{A \subseteq \mathcal{T} : |A| \leq k-1\}$, while the access structure Γ is given by $\Gamma = \{A \subseteq \mathcal{T} : |A| \geq k\}$.

In this work and in particular in the context of feasibility results, we will restrict ourselves to such threshold access or adversarial structures in which all shares are considered equally important. We are thus interested in how many corruptions a scheme can tolerate against an adversary with certain abilities, regardless of which parties are corrupted. We will use t to denote the maximum number of corruptions by a specified adversary that a scheme can tolerate, that is, with an adversary structure given by $\mathcal{A} = \{A \subseteq \mathcal{T} : |A| \leq t\}$. Furthermore, we will use r to denote the reconstruction parameter. A scheme with reconstruction parameter r has $\Gamma = \{A \subseteq \mathcal{T} : |A| \geq r\}$ as its access structure, meaning that any subset of parties of size larger or equal to r can reconstruct the secret based on their shares alone.

In later parts of this section we will also discuss the number of actively corrupted parties that a secret sharing scheme can tolerate. An active adversary has complete control over the corrupted parties and might deviate from the protocol arbitrarily. For example, an active adversary might have some of the corrupted parties hand in faulty shares for reconstruction or withhold them altogether. In order to clarify the distinctions between

Parameter	Property
t	defines threshold adversary structure
r	defines threshold access structure
p	defines the privacy threshold
d	defines the robustness threshold

Table 2.1: Overview of the different parameters that are used to characterise secret sharing schemes in Section 2.6.

active and passive adversaries, we have to distinguish between the privacy and the robustness of a scheme. In particular, we will use p as the privacy parameter, that is, any subset of parties of size up to and including p does not contain any information on the secret. On the other hand, we will use the parameter d to denote the *robustness*-parameter. A scheme with robustness parameter d maintains correctness during reconstruction (which we will later define more formally) despite the presence of up to d arbitrary, actively corrupted parties. In particular, in the case of a passive adversary, the parameter t that characterises the adversary structure coincides with the privacy parameter p, while in the case of an active adversary, t coincides with $\min(p, d)$. An overview of the parameters used hereafter can be found in Table 2.1.

For instance, for threshold-k schemes for some $k \leq n$, we have that the maximal tolerable adversary structure is given by $t = k - 1$, while the reconstruction parameter r coincides with k. In order to compile feasibility results for different types of adversaries, we are interested in finding the maximum number of tolerable corruptions, i.e., the maximal $t < n$. We have seen earlier that in the case of a passive adversary, threshold schemes such as Shamir's scheme exist that can tolerate up to $n - 1$ corruptions. Thus for secret sharing schemes according to Definition 2.1.4, the theoretical limit of passively corrupted parties is $t < n$ and this bound is tight.

Tolerating active adversaries

In the context of threshold schemes it is assumed that precisely k (arbitrary) shares are available for reconstruction and that these are perfectly intact, in the sense that they have not been tampered with by any adversary and are equal to the shares that have been handed out initially. In the following, we will consider more general types of attacks on the secret sharing protocol. In particular, active security will be considered in which the corrupted parties might modify their shares, thus possibly preventing a correct reconstruction of the secret. We again consider the worst-case-scenario in which arbitrary deviations from the protocol on behalf of the adversary are allowed. For now, we will assume the dealer to honestly perform the share generation and distribution. Additionally, we will later consider protocols that achieve privacy and correctness even in the presence of a corrupted dealer that might try to distribute an inconsistent sharing of a secret.

We will start our discussion with an informal note on the feasibility of actively secure threshold secret sharing schemes, that is, secret sharing schemes that fulfill Definition 2.1.4 in the presence of an active adversary that may perform arbitrary operations on the shares in her possession. For such schemes, the possibility of some of the shares that are used for reconstruction deviating from the ones that have been handed out by the dealer has not been considered thus far. In particular, Lipinska et al. ([LMRW20]) note that what they define as a *strong threshold scheme* is impossible:

Definition 2.6.1 (Strong threshold scheme). A *strong threshold scheme* for some $k \leq n$ is a scheme where, despite the presence of $t \leq n$ actively corrupted parties:

(1) Any set of $k-1$ or less shares does not reveal any information about the secret state.

(2) The secret can be reconstructed from any k shares.

A simple reasoning shows that such strong threshold schemes are possible only if $t = 0$ (Proposition 1 in [LMRW20]):

Proposition 2.6.1. *It is impossible to construct a strong threshold secret sharing scheme according to Definition 2.6.1.*

Proof. Assume that a strong secret sharing scheme according to Definition 2.6.1 exists that tolerates some number of t active cheaters. If k shares are given, then by property (2) of 2.6.1 the secret can be reconstructed from these shares, even if $t < k$ of them are provided by corrupted parties. Since these parties are allowed to perform arbitrary computations on their shares, we can replace these shares by some arbitrary state, implying that the secret can be constructed from the remaining $k-t$ shares. On the other hand, property (1) says that no $k-1$ or less shares reveal any information on the secret. Hence, we obtain $k-t > k-1$ and therefore $t = 0$. □

We note that the above reasoning holds for both classical and quantum secret sharing schemes. In light of this result, we see that in order to allow for active cheaters in the protocol, we will have to consider a specific class of secret sharing schemes that do not satisfy the strict definition of a threshold scheme. Hence, in order to maintain the ability to correctly reconstruct the secret in the presence of an active adversary, we will have to add some redundancy when collecting shares for reconstruction. More formally, this means that whenever active security is considered for a secret sharing scheme, we will necessarily find a gap between the privacy parameter p, and therefore also the adversary structure t, and the reconstruction parameter r, i.e., $t \leq p < r$. In Section 2.1 we have already noted that schemes that exhibit such a gap between privacy and reconstruction parameters are referred to as *ramp schemes*, and have first been introduced as a generalisation of threshold schemes by Blakley and Meadow ([BM84]). Ramp schemes typically consider passive security and possibly leak (partial) information for subsets of shares in the gap between privacy and reconstruction parameter. Even though any secret sharing scheme that can tolerate an active adversary in particular has passive security, we will refrain from using the term in this context to avoid confusion. Since we are interested in finding the maximum number of tolerable corruptions, we will in the following assume that all shares are collected for reconstruction, including potentially faulty ones, thus following, e.g., [CFOR12], [CGS02] and [LMRW20]. In the literature, one often speaks of "reconstruction by any set containing k honest parties", even though means to distinguish honest from dishonest parties (or correct from faulty shares) are not always given. Formally, we thus consider schemes with reconstruction parameter $r = n$ hereafter.

We also remark that typically, the privacy parameter for schemes with active security is higher than the robustness parameter. Note, however, that this property is not trivially satisfied. Consider for example the n-fold repetition code which can tolerate up to $\lfloor (n-1)/2 \rfloor$ corrupted shares using a majority vote as the reconstruction procedure, but has 0-privacy.

Secret sharing schemes that tolerate active adversaries who might provide faulty shares for reconstruction are oftentimes called *robust* or *error-tolerant secret sharing schemes* ([TW88], [CDF01], [CFOR12]) and guarantee privacy and reconstruction despite a number of $t < n$ actively corrupted parties within the protocol. As for secret sharing schemes discussed in the previous sections, robust secret sharing schemes consists of a *sharing* and a *reconstruction* phase, denoted Share and Rec. In the sharing phase, the dealer encodes her secret $s \in S$ into n shares and sends each participant T_i one share $s_i \in S_i$. In the reconstruction phase, *every* participant, including all corrupted parties, sends her shares s_i' to a reconstructor R, who performs the reconstruction and produces a secret s'. Since we consider an active adversary, the corrupted parties shares are considered to be elements of $\{S_i \cup \perp\}$ where $\perp$ describes a party withholding her share from reconstruction. Adopting the definition of [CFOR12], we can quantify the robustness of a secret sharing scheme against an active adversary as follows:

Definition 2.6.2 ((t,δ)-robust secret sharing). We say that an n-player secret sharing scheme Share with reconstruction procedure Rec and secrets in a finite field $\mathbb{F}$ is (t,δ)-*robust*, where $t < n$ and $\delta < 1/2$, if the following properties hold for any $s \in \mathbb{F}$ in the presence of an active adversary corrupting up to t parties.

- *Privacy*: Every subset of shares of size $\leq t$ satisfies perfect privacy in the sense of Definition 2.1.4.
- *Reconstructability*: At the end of Rec, the reconstructor R outputs $s' = s$ with probability $\geq 1 - \delta$.

Here, we require $1 - \delta > 1/2$ since otherwise the reconstruction is no better than a blind guess on part of the reconstructor (remember that we always require that $|\mathbb{F}| \geq 2$). We say that a robust secret sharing scheme has *exact reconstruction* or *zero error-probability* if $\delta = 0$. For robust secret sharing protocols, the dealer is assumed to be honest and, therefore, to hand out consistent shares of some secret state in the sharing phase. Note that, generally, cheaters that withhold their shares from reconstruction are handled more easily than those that provide faulty shares since the withholding cheaters' identities are revealed immediately; compare, e.g., the discussion on erasure errors in Section 2.5.1 or [MS81].

In this model, two bounds on the feasibility of robust secret sharing schemes are well-known ([Cev11]) and we present them in the form of the following two lemmas.

Lemma 2.7. *There does not exist an n-party (t,δ)-robust secret sharing scheme (Share, Rec) for which $t \geq n/2$.*

Proof. Assume a secret sharing scheme with t-privacy that tolerates $t \geq n/2$ actively corrupted parties. Then, if the t cheaters all supply bogus shares (or withhold their shares completely) for reconstruction, then this implies that only $n - t \leq t$ intact shares are available for reconstruction. By t-privacy, these shares do not contain any information on the secret and thus have the same probability distribution for all secrets $s \in \mathbb{F}$. Denote by V_s' the vector of all shares of a secret sharing of a secret s that is sent to the reconstructor, with shares coming from corrupted parties replaced by $\perp$. Then the best that Rec can do is output a blind guess and we find

$$\sum_{s \in \mathbb{F}} \Pr\left[\mathsf{Rec}(V_s') = s\right] \leq 1.$$

Therefore, $\Pr[\mathsf{Rec}(V_s') = s] \leq 1/|\mathbb{F}| \leq 1/2$ for some $s \in \mathbb{F}$, so the scheme is not robust. □

We can furthermore show that if we require the reconstruction to have zero error probability, then at most $t < n/3$ active corruptions can be tolerated.

Lemma 2.8. *There does not exist an n-party* (t, δ) *robust secret sharing scheme* (Share, Rec) *for which* $t \geq n/3$ *and* $\delta = 0$.

Proof. We argue towards contradiction and assume that such a scheme existed. Let $t \geq n/3$ and split the index set of participants, $\{1, 2, \ldots, n\}$, into a partition $I \cup I' \cup J$ such that $|I| = |I'| = t$ and $|J| = n - 2t \leq t$. By the t-privacy property, the probability distribution on the shares indexed by set J is independent of the shared secret, so there exist two secrets s, s' such that the shares generated by the secret sharing scheme agree on the index set J. Now recall that we denote a secret sharing of s by $[s]$, i.e., $\mathsf{Share}(s) = [s]$ and define a vector of "shares" V that has the shares of $[s]$ on the positions indexed by J (equal to those of $[s']$), the positions in I filled with the shares of $[s]$ and the positions in I' filled with the shares of $[s']$. Such a vector differs from the secret sharings $[s]$ and $[s']$ in at most t positions each. Now V can be obtained by two different strategies of an adversary, either corrupting a sharing of s or of s'. In each case, only t active corruptions are required. Hence, we have that either $\mathsf{Rec}(V) \neq s$ or $\mathsf{Rec}(V) \neq s'$ so one of these pairs has non-zero error probability δ, that is, exact reconstruction is impossible for $t \geq n/3$. □

Tompa and Woll ([TW88]) were the first to note that "out-of-the-box" Shamir's scheme with reconstruction procedure described below Definition 2.1.6 cannot tolerate any active corruptions, see also [CFOR12]. On the other hand, for any $t < n/3$ robust secret sharing with exact reconstruction is possible, namely by using Shamir's secret sharing with Reed-Solomon error correction as the reconstruction procedure ([MS81]), showing that this bound is tight, even against a rushing adversary [Che19]. Furthermore, Rabin and Ben-Or give a robust secret sharing scheme that tolerates a strict minority of actively corrupted parties and probability of error during reconstruction that decreases exponentially with the size of the shares in [RB89] (as did [CDF01], [Che19] and [CDD+15]), proving that this bound is tight, too. [CFOR12], [MSV20] and [FY20] even achieve the same result in presence of a rushing adversary. In particular [Che19] and [MSV20] present an overview of robust secret sharing scheme constructions.

Verifiable secret sharing

In particular in the context of secret sharing based MPC, an active adversary might not just corrupt recipients of the shares but also the dealer, causing inconsistent shares to be handed out in the sharing phase that do not reconstruct to a unique secret. This situation is covered by the notion of *verifiable secret sharing (VSS)* and was first introduced in [CGMA85]. Similar to the ordinary definition of secret sharing protocols, a verifiable secret sharing protocol consists of a sharing phase in which the secret is shared by a dealer D, as well as a reconstruction phase in which the secret is recovered by a reconstructor R. Additionally, a *verification phase* is included in which the participants jointly verify their shares as consistent. If at the end of the verification phase the participants conclude that their shares correspond to one unique secret we say that the dealer *passed the verification phase*. Otherwise, the dealer will be detected as dishonest and the protocol typically is aborted. A VSS scheme is therefore defined by a triple of procedures (Share, Rec, Verify). Therefore, verifiable secret sharing schemes are robust secret sharing schemes with an additional verification procedure. The participants' ability to verify the consistency of their shares implies that the dealer is committed to the secret she shared, which is why the sharing phase of a VSS is sometimes referred to as the *commitment phase*. Overall, we require a verifiable secret sharing scheme to satisfy the following requirements despite the malicious actions of t actively corrupted parties:

- *Soundness*: If R is honest and the dealer passes the verification phase successfully, then there is a unique secret that can be recovered by R.

- *Completeness*: As long as the dealer D is honest, she always passes the verification phase. If additionally R is honest, then the secret that R recovers is exactly D's secret.
- *Privacy*: As long as D is honest, no group of $p \geq t$ parties learns any information about the secret before reconstruction.

As a special case of robust secret sharing, the same upper bounds on the maximum number of active corruptions hold for VSS schemes, namely $t < n/3$ without any error probability during reconstruction and $t < n/2$ for non-zero error probability. [BGW88] gives a verifiable secret sharing protocol that satisfies the bound of $t < n/3$ without any further assumptions. In order to surpass the bound of $t < n/3$ for the particular task of VSS, one needs to assume the existence of a so-called *broadcast channel* in addition to the secure point-to-point network already in place.

Definition 2.8.1 (Broadcast, [LSP82]). A broadcast channel allows each participant in an n-party protocol with point-to-point authenticated communication channels to send a message to all other participants with the following guarantees despite the malicious actions of an active adversary corrupting some $t \leq n$ parties:

(1) *Consistency:* Even if the sender is dishonest, the same message is received by all parties.

(2) *Validity:* If the sender is honest then the message that is received by the honest parties is equal to the sender's original message.

Broadcast (in the presence of some t out of n actively corrupted parties) is known to be equivalent to the problem of *Byzantine Agreement (BA)* which was first described and shown to be possible, *if and only if*, $t < n/3$ by works of Pease et al. in [LSP82], [PSL80]. In Byzantine Agreement[1], n parties that are connected via a point-to-point authenticated network each start with some private input v_i and at the end of the protocol, they each output a value c_i. It is then required that on one hand, for each pair of honest parties i, j we have $c_i = c_j$, and that if all honest parties start with the same initial value, that is, $v_i = v_j = v$ for all honest i, j and some value v, then we also have $c_i = c_j = v$. Verifiable secret sharing can be used to implement broadcast by treating the message to be sent as the input to the VSS. Hence, if there was a protocol that implemented VSS for $n/3 \leq t < n/2$, then this would imply a broadcast channel and therefore a Byzantine Agreement protocol secure against $n/3 \leq t < n/2$ active cheaters, leading to a contradiction. The availability of a broadcast channel is therefore assumed for VSS (and, moreover, for MPC solutions) whenever $t \geq n/3$. Note that in the literature, the term broadcast is used ambiguously and does not always allow for a dishonest sender. Sometimes, the term *secure broadcast* or *consensus broadcast* ([CDN15]) is used instead.

We remark that, as we have previously seen, even the weaker task of robust secret sharing requires some probability of error for the reconstruction phase, which thus also extends to the task of VSS. A broadcast channel and therefore a BA protocol that is realised from such a VSS protocol would also exhibit some error probability in the sense that, while consistency is satisfied, validity might only hold with some error probability. But even such a weaker form of BA is shown to be impossible in [LSP82] and, in a simpler form, [FLM86]. Further work can also be found in [GY89], [FM89] and [CPS20]

[1]An interesting fact about Lamport et al.'s seminal work on the Byzantine generals problem, [LSP82], is that in many protocols for secure, distributed computations (e.g., [BGW88], [CDN15]), active security is described as tolerating *Byzantine faults* that are assumed to be strategically placed by corrupted parties in the worst way possible, resembling the actions of malicious Byzantine generals.

who reference unpublished work by Yao and Karlin that supposedly explicitly shows the impossibility of any form of Byzantine Agreement with some non-zero probability of error. Therefore, we see that, while robust secret sharing with some probability of error might be possible for $n/3 \leq t < n/2$ (and, in fact, is [RB89]), the impossibility of Byzantine Agreement for $t \geq n/3$ extends to the task of VSS. In the literature it is therefore commonly seen that a broadcast channel is added as an assumption in the information-theoretical model whenever $t \geq n/3$.

In [RB89], Rabin and Ben-Or give a protocol that achieves VSS with exponentially small probability of error given a strict honest majority and assuming the existence of a broadcast channel, while more recent results by Garay et al. achieve the same task, even secure against a rushing adversary [GGOR13].

Altogether, we discussed the feasibility of (classical) secret sharing for different adversarial settings. An overview of these characterisations can be found in Table 2.2. We remark that various other adversarial settings have been considered in the theory of secret sharing. In this work, we focus on the most prominent adversarial models. A survey and classification of the different settings and existing work on (classical) secret sharing schemes can be found in [Mar08]. Furthermore, Table 2.3 gives a summary of the feasibility results of classical secret sharing protocols in different adversarial settings that we have compiled in this section.

Scheme	**Adversary**	**Dealer**
Threshold secret sharing	passive	honest
Robust secret sharing	active	honest
Verifiable secret sharing	active	dishonest

Table 2.2: Characteristics of the adversary for different types of secret sharing schemes.

Scheme	**Maximal corruptions**
Threshold secret sharing	$t < n$
Robust secret sharing with exact reconstruction	$t < \frac{n}{3}$
Robust secret sharing	$t < \frac{n}{2}$
Verifiable secret sharing with exact reconstruction	$t < \frac{n}{3}$
Verifiable secret sharing with non-zero error probability	$t < \frac{n}{3}$ ($t < \frac{n}{2}$ with broadcast)

Table 2.3: Feasibility of classical secret sharing with unconditional security. All bounds are tight. Here, t denotes the number of tolerable corruptions (depending on the adversary, see Table 2.2)).

Feasibility of quantum secret sharing

In our discussion of quantum secret sharing schemes, too, the focus has so far been on threshold schemes. For such schemes we have seen that by Theorem 2.5.6, the maximum number of passively corrupted parties is limited to be strictly smaller than $n/2$. Furthermore, we have seen that by Theorem 2.5.8, this bound is tight.

QSS in the presence of active adversaries

When considering an active adversary, we find that the theory of quantum error correction discussed in Section 2.5.1 provides a strict upper bound on the number of tolerable corruptions: by Theorem 2.5.3 as well as the quantum no-cloning theorem we know that the number of actively corrupted parties is bounded to be strictly smaller than $n/4$, since otherwise such a quantum secret sharing scheme would constitute a quantum error-correcting code that can correct $\geq n/2$ erasure errors, leading to a contradiction. Since the construction used in the proof of Theorem 2.5.8 by Cleve et al. [CGL99] was based on quantum error-correcting codes with maximal distance in the first place, these automatically provide optimal robustness for any adversary structure characterised by a threshold $t < n/4$. Here, again, we assume that all shares, including potentially corrupted ones, are used for reconstruction, that is, we impose a reconstruction threshold of $r = n$.

We have noted below Theorem 2.5.2 that the necessary and sufficient conditions for quantum error correction, and therefore Theorem 2.5.3, break down when considering approximate quantum error correction. As a result, Crépeau et al. were able to show in [CGS05] that approximate quantum error-correcting codes exist that can correct up to $t = \lfloor (n-1)/2 \rfloor$ errors with the guarantee that the fidelity (Def. 1.1.8) of the reconstructed state to the initial state is exponentially close to 1, depending on the size of the quantum registers of the code. Their construction immediately yields a robust quantum secret sharing scheme under the assumption that the dealer is honest which tolerates up to $t = \lfloor (n-1)/2 \rfloor$ corruptions by an active adversary and for which the reconstructed secret is exponentially close to the initial secret, depending on the size of the shares. This highlights the similarity between classical secret sharing and quantum error correction (as well as quantum secret sharing), namely that relaxing the requirement of exact reconstruction of the secret to approximate reconstruction, or reconstruction with some small probability of error, allows tolerating up to a strict minority of actively corrupted parties, both for robust classical and quantum secret sharing schemes. The analogy is also explicitly pointed out by [CGS05]. Nonetheless, this raises the question whether approximate quantum error-correcting codes can be used to surpass the quantum no-cloning bound of $t < n/2$ that only applies to perfect copies of quantum states. We conjecture that the result of $t < n/2$ active corruptions that is achieved in [CGS05] is optimal if one requires that the fidelity of any reconstructed state can be made arbitrarily close to the secret, as we have seen in previously mentioned constructions. If an approximate QECC, or secret sharing scheme, for that matter, existed that tolerated a number $t \geq n/2$ active corruptions then this would constitute a *universal quantum cloning machine (UQCM)*: similar to what we have seen in the proof of Theorem 2.5.4, we could use such an encoding and split the set of shares into two disjoint sets, each of which could be used to create an approximate copy of the secret. For such UQCMs, constant bounds on the optimal fidelity of the cloned states to the initial state are known that would prevent an arbitrarily close approximation. A review of approximate quantum cloning can be found in [SIGA05] and [RZLZ18] contains many more references on recent results on the topic.

Verifiable quantum secret sharing

Finally, we also discuss the feasibility of *verifiable quantum secret sharing (VQSS)* schemes in which we also consider a potentially dishonest dealer. Similar to the previous discussions, there is a dealer D who shares a quantum state ρ in her possession in the sharing (commitment) phase. A verification phase allows the participants to verify their shares as corresponding to one unique secret or detect whether a sharing is inconsistent. In the latter case, the protocol is typically aborted. Essentially, what is shown in the literature on VQSS schemes (e.g. [CGS02], [LMRW20]) is that after the verification phase, if the protocol verification procedure did not abort, then the sharing is close to a consistent sharing, or the protocol aborts with high probability. For more details, see, e.g., Appendix A of [LMRW20]. As described earlier for classical VSS, we require soundness, completeness and privacy to hold in the presence of a malicious, active adversary corrupting an arbitrary subset of up to t parties. In the reconstruction (recovery) phase, there is a designated reconstructor R that collects all shares and reconstructs the secret state.

As a special instance of the quantum secret sharing schemes we have considered thus far, VQSS also satisfies the known bounds on its parameters, such as the no-cloning theorem for privacy against a passive adversary with up to $p < n/2$ corruptions and robustness (with exact reconstruction) against up to $d < n/4$ actively corrupted parties. Hence, we find that the maximum number of tolerable corruptions for a VQSS protocol with exact reconstruction is given by $t < n/4$.

In [CGS02], Crépeau et al. explicitly prove the upper bound of $t < n/4$ that also holds for verifiable quantum secret sharing schemes with exact reconstruction. Based on ideas of [CCD88], they construct a VQSS scheme that satisfies this bound, with probability of not detecting a dealer handing out inconsistent shares (soundness) exponentially close to 1 in a security parameter that describes the iterations of a subroutine of the verification procedure. They call such a VQSS scheme that includes some probability of error a *statistical VQSS scheme*. They also assume the presence of a broadcast channel since a classical VSS scheme is used as a subprotocol, but note that this is in fact not necessary since $t < n/4 < n/3$ and thus broadcast can be implemented securely. The same result is achieved by Lipinska et al. in [LMRW20]. In fact [CGS02] and [Smi01] remark the construction of a VQSS for $t < n/4$ with zero probability of error as an open problem and to the best of our knowledge, this seems to still be the case at the time of writing.

In [BCG$^+$06], the ideas of [CGS05] are used to devise a VQSS scheme with exponentially low probability of error that is secure against an active adversary corrupting any strict minority of parties, that is, $t < n/2$. Their result holds even in the presence of a rushing adversary. Again, since the number of corrupted parties surpasses $n/3$, the existence of a classical broadcast channel is assumed. Similar to the reasoning above, we conjecture that a strict minority of active corruptions is optimal due to the bounds on approximate quantum cloning.

We have now compiled a list of feasibility results for both classical and quantum secret sharing for different types of adversaries (passive or active, honest or dishonest dealer) and with different requirements on the scheme itself (exact reconstruction or admitting a small probability of error). Table 2.4 displays these results and shows the differences between classical and quantum secret sharing schemes in most adversarial settings.

To summarise, we have observed in this section that classical and quantum secret sharing schemes differ fundamentally with regard to the tolerance of adversaries with different abilities, most importantly because of the quantum no-cloning theorem that, in the case of secret sharing schemes, limits the capacities of quantum protocols. On the other hand, we have also seen that classical and quantum protocols exhibit similar features: for robust classical and quantum schemes as well as the more specific notion of V(Q)SS we observed that allowing for a small probability of error during reconstruction of the secret allows to drastically increase the number of tolerable corrupted parties. In this work, we have focussed on feasibility in the broadest sense, disregarding efficiency which in this case concerns communication complexity as well as share size. To find

Scheme	**Classical**	**Quantum**
Threshold secret sharing	$t < n$	$t < \frac{n}{2}$
Robust secret sharing with exact reconstruction	$t < \frac{n}{3}$	$t < \frac{n}{4}$
Robust secret sharing	$t < \frac{n}{2}$	$t < \frac{n}{2}$
Verifiable secret sharing with exact reconstruction	$t < \frac{n}{3}$	$t < \frac{n}{4}$
Verifiable secret sharing with non-zero error probability	$t < \frac{n}{3}$ ($t < \frac{n}{2}$ with broadcast)	$t < \frac{n}{3}$ ($t < \frac{n}{2}$ with broadcast)

Table 2.4: Feasibility of classical and quantum secret sharing with unconditional security. Except for VQSS with zero error probability, all bounds are tight. t denotes the number of tolerable corruptions (passive for threshold secret sharing, active else, compare also Table 2.2)).

out more about the efficiency of secret sharing in different adversarial models, we recommend consulting, e.g., [Bei11], [HLKB16], [MSV20], [Che19], [FGG$^+$06] for classical schemes or [SS19], [SS20] and [LMRW20] for quantum schemes.

3. Multi-party quantum computation

In this final chapter, the focus will be on multi-party computation, in particular on multi-party quantum computation. Classically, MPC protocols have been built on essentially three different kinds of primitives: garbled circuits, homomorphic encryption and (verifiable) secret sharing (see, e.g., [Yao82], [AJL$^+$12], [GMW87], [CDvdG87], [BGW88]). They can be categorised by the different types of adversaries they can tolerate, and in particular we can distinguish between computational and information-theoretical security. Security for computationally bounded adversaries is based on the assumed computational hardness of some problem, while information-theoretical (or unconditional) security makes no assumption on the computational capacities of the adversary. In this work, we focus on unconditionally secure protocols which are typically constructed from secret sharing schemes. In Section 3.3 we will formalise this notion of security for MPQC protocols.

Recall the discussion in Section 2.1, showing that a linear secret sharing scheme can be used to securely implement addition of secrets by having the parties locally add their shares and that, with some additional work, multiplication of shares can be performed securely as well. MPC protocols of this type are said to follow the *share–compute–reveal* paradigm. All inputs of the participants are shared using a secret sharing scheme and local computations effectively implement the computation on the underlying secrets. Afterwards, shares are gathered and the final output is reconstructed. In the following, we discuss the quantum equivalent of (secret sharing-based) MPC, multi-party quantum computation, and point out conceptual similarities and differences.

MPQC has first been introduced in 2002 in [CGS02], [Smi01] and since then, protocols that essentially follow two types of constructions have been proposed. In [CGS02], [BCG$^+$06] and [LMRW20], information-theoretically secure MPQC protocols have been proposed, while [DNS10], [DNS12] and [DGJ$^+$20] achieve computationally secure MPQC mainly using so-called quantum authentication codes. We focus on the former type of information-theoretically secure protocols. [CGS02], [BCG$^+$06] and [LMRW20] are all based on VQSS schemes and this chapter will be devoted to analysing this approach further. We will therefore introduce the computational model for quantum computations, introduce important techniques for VQSS-based MPQC protocols and discuss the most recent MPQC protocol in [LRW20] in more detail. In the final section, Section 3.4, we discuss and compare feasibility results for MPC and MPQC.

3.1 Universal computation

Before diving into particular MPQC protocols, we need to clarify what computational tasks we try to accomplish and how we model them. The focus of this work and the discussed literature is that of arbitrary (quantum) computation as opposed to specific tasks in distributed quantum computation, for which better results in terms of efficiency or privacy guarantees might be feasible.

The arithmetic circuit model

Classical MPC protocols are typically defined in the *arithmetic circuit model* in which every function can be expressed in terms of addition and multiplication gates over some pre-defined finite field $\mathbb{F}$. In this model, each party $i = 1, \dots, n$ is typically assumed to provide some input $x_i \in \mathbb{F}$ and the goal of the protocol is to compute some function $f : \mathbb{F}^n \to \mathbb{F}^m$ on the inputs that consists of a number of internal addition and multiplication gates as well as additions and multiplications by constant values from $\mathbb{F}$. Hence, addition and multiplication gates are *universal* in the arithmetic circuit model, in the sense that any MPC protocol that achieves to implement both addition and multiplication securely in a distributed manner can be used to compute arbitrary arithmetic circuits. Considering arithmetic circuits comes without loss of generality, as it is known that any computable function can be specified as a polynomial-size Boolean circuit, which in turn can be simulated using an arithmetic circuit. See [CDN15], Section 3.3.1 for a detailed introduction to the arithmetic circuit model.

Universal quantum computation

Similarly, in multi-party quantum computation, it is our goal to perform universal quantum computation in a distributed, secure manner. In this context, a distributed quantum computation has multiple quantum states as inputs that are provided by different parties and some quantum state as its output. The goal of MPQC is to perform the computation *securely*, that is, typically without sacrificing correctness of the computation, or leaking information to some adversary that corrupts a certain subset of parties. In Section 3.3 we specify the notion of security for MPQC protocols further.

We have already discussed a number of operations that can be performed on quantum states, such as unitary evolution or measurement of the state, in Chapter 1. In this section, we specify the model of computation that is often used for quantum computations further. In particular, we will consider the *quantum circuit model of computation* first introduced by David Deutsch ([Deu89]), which at of the time of writing is the most prominent and most widely studied model of quantum computation. The introduction we provide in the following will be based mainly on [NC16], who also give a detailed overview of further references on the topic.

In the quantum circuit model, the basic unit of computation is that of a qubit, i.e., a two-dimensional quantum state. The state space therefore is a 2^n-dimensional complex Hilbert space in which product states of the form $|x_1, \dots, x_n\rangle$ with binary x_i denote the computational basis states of the state space, as discussed in Section 1.1. A quantum circuit $\mathfrak{R}$ is then comprised of a sequence of unitary gates acting on an arbitrary number of qubits and single-qubit measurements in the computational basis. Finally, it is required to be able to introduce quantum states (so-called *ancilla qubits*) in the computational 0 state $|0^n\rangle$, often abbreviated as $|0\rangle$, into the circuit.

What is particularly important to discuss is the ability to perform arbitrary unitary gates on the qubits. There are continuously many such gates and it would be impractical to strive for being able to implement any gate with perfect precision. Instead, one focuses on a certain subset of gates that are *universal for quantum computation*, which says that any unitary operation may be approximated up to arbitrary accuracy by a sequence of gates from that subset. The approximation error of a unitary V approximating another unitary

U acting on the same state space $\mathcal{H}$ is defined as

$$E(U,V) := \max_{|\psi\rangle} \| (U - V) |\psi\rangle \|,$$

where the maximum is taken over all normalised states $|\psi\rangle \in \mathcal{H}$ and $\| \cdot \|$ denotes the norm induced by the inner product, $\| |\phi\rangle \| = \sqrt{\langle\phi|\phi\rangle}$. In [NC16] it is shown that this measure of error can indeed be understood as the difference in the measurement statistics for any measurement and any state $|\psi\rangle$ after U or V are applied.

In order to achieve universal multi-party quantum computation of arbitrary quantum circuits, one therefore needs to show that the designated MPQC protocol is suited to implement the above mentioned functionalities in a secure, distributed manner. In particular, it is typically shown that

(1) a universal set of unitary gates can be implemented,

(2) single-qubit measurement (in the computational basis) is possible, and;

(3) ancilla qubits in the state $|0\rangle$ can be prepared.

If we can perfom the above operations securely in the presence of a specified adversary, then we can do so for arbitrary quantum computations. Note that by the *principle of deferred measurement*[1] we can postpone all measurements to the end of the circuit. Therefore, when working in the quantum circuit model, we typically see that it is implicitly assumed that the circuit itself consists only of gates from the universal set, as well as the introduction of ancilla qubits, potentially followed by single-qubit measurements at the end of the computation. We also note that the quantum circuit model is of sufficient generality: as we have mentioned earlier, using qubits as the smallest unit of computation is rather a matter of convention (and possibly of physical implementation), since qubits can be used to simulate higher-dimensional quantum states. Furthermore, restricting to unitary gates and orthogonal measurements comes without loss of generality, as these, performed on an extended system using ancilla qubits, can be used to implement arbitrary completely-positive trace-preserving maps and the most general forms of measurement. For more information on the generality of the quantum circuit model, see, e.g., [Pre99].

Universal sets of unitaries

Various universal sets of gates are known. One prominent example is the set consisting of *all* single qubit gates combined with the two-qubit CNOT gate discussed in Section 1.1. In fact, any unitary may be expressed *exactly* (as opposed to an approximation) as a combination of these gates. Another example mentioned in [NC16] is that containing the Hadamard gate, phase gate (denoted S or P), CNOT gate and T gate (also denoted $\pi/8$ gate) which demonstrates that a discrete, in fact even a finite set of gates suffices for arbitrarily accurate approximations of any unitary. Note, however, that no finite set of gates is rich enough to implement arbitrary unitaries exactly ([Smi01]). Furthermore, approximating circuits for any given unitary may require large numbers of gates from the universal set (some gates are even known to require exponentially many gates for approximation, see [NC16]). Thus, the choice of MPQC protocol with the accompanying universal set of gates may well influence the efficiency of the computation. This discussion, however, is out of scope for this work, just like the discussion of complexity in

[1]The principle of deferred measurement tells us that any quantum circuit $\mathfrak{R}$ can be represented as $\mathfrak{R} = P \circ \mathcal{U}$, where P is a measurement, and $\mathcal{U}$ can be decomposed into unitary gates ([NC16], Section 4.4).

classical MPC is. We also note that similar results for discrete sets of universal gates hold for higher-dimensional quantum states, called *qudits* or *qupits*, as discussed in [ABO08]. [SI05] gives a succinct overview of universal sets of gates.

In order to achieve universal quantum computation, it would therefore suffice to be able to implement any such universal set of gates (in addition to measurements and ancilla preparation). In the context of MPQC, certain sets of gates are known to be easily realisable in a secure, distributed manner, especially for well-studied constructions based on quantum error-correcting codes. The reasons for this are discussed in Section 3.2. One particular set of gates that often occurs in this context is the set of *Clifford gates*, also known as the *Clifford group*. The Clifford group contains a rich, but not a universal set of gates. Due to its frequent occurrence in the literature of MPQC ([BCG+06], [DNS10], [DNS12], [DGJ+20], [LRW20]) we will discuss some of its properties that will then be used in the next section for an explicit example of an MPQC protocol.

Definition 3.1.1 (Clifford group). The *Clifford group* is defined as the set of unitary operators acting on n qubits which fix the Pauli group $\mathcal{P}_n$ under conjugation. That is, the Clifford group is defined as the normalizer, $N(\mathcal{P}_n) := \{V \in \mathcal{U}(2^n) \mid V\mathcal{P}_n V^\dagger = \mathcal{P}_n\}$, of the Pauli group in the group of n-qubit unitaries $\mathcal{U}(2^n)$.

The Clifford group indeed is a group, as the conjugation operation $N \mapsto UNU^\dagger$ is a multiplicative group homomorphism,

$$MN \mapsto UMNU^\dagger = (UMU^\dagger)(UNU^\dagger). \tag{3.1}$$

The Clifford group, among others, contains the single-qubit Hadamard transform H, the phase gate P, as well as the two-qubit controlled-NOT gate. In fact, the set of tensor products of these three unitaries generates the full Clifford group (Theorem 10.6 in [NC16]). Being able to compute Clifford circuits, however, is in itself not of much value since by the Gottesman–Knill theorem ([Got98a]) we know that any quantum circuit comprised solely of preparation of computational basis states, unitary gates from the Clifford group, as well as measurements in the computational basis, can be efficiently simulated classically.

As noted above, the set $\{H, P, CNOT, T\}$, i.e., the set of Clifford gates combined with the T-gate, forms a universal set of gates, and it is frequently used in the literature on MPQC. This universal set is typically denoted Clifford$+T$ and we will adopt this notation. In fact, adding any non-Clifford gate to the Clifford group constitutes a universal set of gates ([NRS01]), but in our context, simple gates such as the T gate are more well-studied and techniques are known that facilitate the secure distributed implementation of such gates. Without going into too much detail yet, we remark that for many MPQC constructions, especially those based on quantum secret sharing and therefore quantum error-correction techniques, implementing arbitrary Clifford gates as well as measurements in a secure, distributed manner is cheap, and the difficulty lies in implementing a gate that completes a universal set of gates. Thus, for implementing complementary gates such as the T gate, additional techniques are required to overcome this obstacle. The rich theory of *fault-tolerant quantum computation (FTQC)* provides means to understand these difficulties as well as methods to study and implement secure, distributed quantum computations.

3.2 Fault-tolerant quantum computation

While quantum error correction aims at coping with noise that might affect quantum states, the theory of fault-tolerant quantum computation looks at the bigger picture and is concerned with finding methods that limit the effect of noise or faulty hardware on whole quantum circuits, such as noise that might occur due to faulty measurements or quantum gates. For instance, in order to implement a quantum circuit in a noisy environment, one could encode the inputs using a quantum error-correcting code, and, for every quantum gate in the circuit, decode the state, apply the gate, and encode it again afterwards. Such an approach, however, could blow up the complexity of the circuit, might introduce new errors during encoding or decoding, and leaves the decoded state prone to errors. In FTQC, each gate or measurement in the original circuit is replaced by an *encoded gate* or *procedure* that implements the desired operation on the underlying, encoded state directly on the encoding itself. As for the theory of QEC, the quantum state that contains the information, and that we want to perform a computation on is called the *logical state* or *logical qubit*, while the qubits in which the state is encoded into are called *physical qubits*. Hence, an encoded gate acts on the physical qubits and transforms them into an encoding of the logical state on which the desired *logical gate* has acted, and similarly for measurements.

An important property that such an encoded gate should have is that it prevents potential errors from propagating to other qubits through the application of the gate. The simplest example of this phenomenon is a CNOT gate with a control qubit and a target qubit. If the control qubit is hit by an error, applying a CNOT gate to both qubits potentially propagates the single initial error to both qubits, even though no additional noise has occured. In order to prevent this, one tries to perform most computations locally, that is, only within individual blocks of physical qubits. As a consequence, errors remain local, correcting them remains feasible and so does successfully performing the overall quantum computation.

FTQC for MPQC

Recall from the beginning of this chapter that some MPQC protocols follow a quantum version of the share–compute–reveal paradigm, and that quantum error-correcting codes can be used as quantum secret sharing schemes by treating the individual blocks of qubits of the encoding as shares. As a result, we can transfer techniques from FTQC and quantum error correction to the design of secret-sharing based MPQC protocols. The error model in MPQC is quite different to that of FTQC: while in the latter, errors are typically assumed to occur independently and randomly with some fixed probability, in MPQC we have to assume that an active adversary strategically places errors through the corrupted parties in the worst way possible. Nonetheless, techniques from fault-tolerant quantum computation can be used for the design of MPQC protocols in a straightforward manner. The most important technique in our context is that of *transversality*. We say that a logical gate is *transversal* with respect to some quantum error-correcting code, if it can be implemented by local operations on the encoding such that each operation only acts on qubits within the same block. It is important to note that the transversal implementation of a logical gate may not be a simple tensor product of the logical gate, but it may very well be the case that unitaries different from the desired logical gate have to be applied to the individual blocks, and similar for measurements. Accordingly, a logical version of a quantum circuit $\mathfrak{R}$ implements $\mathfrak{R}$ on the logical qubits, but may apply different operations on the physical qubits. By contrast, we have observed that for instance for linear secret sharing schemes, local addition of shares implements a "logical" addition of the secrets.

Given a quantum secret sharing scheme based on an error-correcting code with which the parties in an MPQC protocol shared their inputs, this would mean that in order to implement some quantum gates on their inputs, each party can locally apply some suitable quantum gates on the shares in their possession. This is beneficial for the design of MPQC protocols in two ways: potential errors introduced by the adversary cannot spread to parties outside of the adversary's control, and no additional communication between parties is needed that might leak further information.

In the design of MPQC protocols that are based on quantum secret sharing schemes, we can therefore draw on the theory of both quantum error correction and fault-tolerant computation: many MPQC constructions realise a verifiable quantum secret sharing scheme using a quantum error correcting code. For specific classes of QECCs it is well known that certain sets of gates can be implemented transversally. However, for any quantum error correcting code, it is impossible to perform universal quantum computation using only transversal gates, which is known as the *Eastin–Knill theorem*.

Theorem 3.2.1 (Eastin–Knill, [EK09]). *There does not exist a quantum error-correcting code for which a universal set of gates is transversal.*

The Eastin–Knill theorem therefore also provides an intuition as to why for some quantum error-correcting codes, a large class of gates (such as the Clifford gates) might be applied transversally "out of the box", while additional work or possibly further restrictions are required to complete a universal set of gates (e.g., Clifford + T). This distinction between *easy* and *difficult* gates to implement therefore resembles that in classical MPC between addition and multiplication as discussed, e.g., for Shamir's scheme in Section 2.1.

We will see in the following section that for certain classes of codes, all Clifford gates can be applied transversally. An additional gate such as the T gate completes the universal set and the theory of FTQC provides techniques to apply the T gate using Clifford gates and measurements using so-called *gate teleportation*. The basic idea of gate teleportation is to apply a quantum gate to a quantum state through quantum teleportation ([GC99]). More concretely, in order to effectively implement the T gate on some arbitrary qubit $|\psi\rangle$, one prepares the so-called *magic state* $|m\rangle := \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$, performs a CNOT gate with $|m\rangle$ as control and $|\psi\rangle$ as target, and then measures $|\psi\rangle$. Depending on the classical measurement outcome, a correction operation is applied to the ancilla, resulting in the desired state $T|\psi\rangle$. The corresponding quantum circuit is depicted in Figure 3.1.

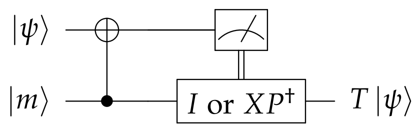

Figure 3.1: Quantum circuit describing gate teleportation for the T gate for some arbitrary initial state $|\psi\rangle$. Single wires describe quantum wires while the double wires are classical and correspond to the classical measurement outcome.

This technique reduces the problem of implementing a T gate on some arbitrary state to that of preparing the magic state $\frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$. Note that other magic states exist that can be used for gate teleportation of gates other than the T gate. In particular in the context of MPQC, this technique is sometimes used to implement the T gate, given that measurements in the computational basis can be performed securely in a distributed manner and (classically controlled) Clifford gates can be performed transversally. The magic state is independent of the initial state $|\psi\rangle$, meaning that preparation of such states can be pushed to an offline phase before executing the MPQC protocol (given that we have sufficient capacities to store them reliably). Choosing an appropriate quantum error-correcting, for which, e.g., the Clifford group is known to be transversal, therefore concentrates most of the remaining difficulty on finding a way to implement an additional gate, such as the T gate, securely.

3.3 On VQSS-based MPQC protocols

As noted earlier, one of the main goals of this work is to dissect and analyse the contruction of information-theoretically secure MPQC protocols from verifiable quantum secret sharing schemes and compare these with classical constructions. In the classical case, multiple constructions for information-theoretically secure MPC exist (e.g. [BGW88], [CCD88], [Bea89]) and they typically follow the same approach: one uses linear secret sharing scheme (in the case of active security, this is chosen to be a verifiable secret sharing scheme) and shows that both addition and multiplication of inputs can be realised through operations on the shares of the scheme, as these gates form a universal set. Afterwards, shares are collected and the function output is reconstructed, typically applying techniques from error-correction. By linearity of the secret sharing scheme, addition typically is the "easy" part as it is inherently satisfied, while multiplication requires some work and, for example, imposes restrictions on the number of corrupted parties (Section 2.1) or requires the successful completion of an offline-phase before the protocol ([Bea91]). We will see that for quantum protocols, similar heuristics apply, and devote this section to discussing a recent proposal for a multi-party quantum protocol based on verifiable quantum secret sharing given by Lipinska, Ribeiro and Wehner [LRW20].

Recall from the introduction that in classical MPC, the goal is to find a protocol that lets some n parties jointly compute some function on their private inputs. Each party is typically assumed to provide some input $x_i \in \mathbb{F}$, and the goal is to compute $f(x_1, \ldots, x_n) = y \in \mathbb{F}^m$ for some m, such that each party receives the output y. More generally, one can require that selected parties obtain parts of the full output $y = (y_1, \ldots, y_m) \in \mathbb{F}^m$, for instance some y_i. Note that the latter approach implies the former. Despite the presence of an adversary with some pre-defined abilities corrupting a certain set of participants, the protocol should remain *correct* and *private*. Correctness implies that despite the actions of the adversary, the honest parties obtain the correct output of the computation. Privacy implies that the adversary does not learn anything on the honest parties' inputs but what can be inferred from the (adversary's) output of the computation. A formal definition of correctness and privacy of MPC protocols can be found in [CDN15]. In the current literature on MPQC, we find a subtle difference in the desired functionality.

[CGS02] and subsequent work define the goal of MPQC protocols as follows: in a network of n parties $1, \ldots, n$, each party starts off with one, possibly unknown quantum state ρ_i. The parties then jointly perform some arbitrary quantum circuit $\mathfrak{R}$ on their inputs, resulting in some global output state ω. Recall that due to the no-cloning theorem, for arbitrary quantum circuits we cannot hope for all parties to hold a copy of the output state at the end of the computation. Instead, after executing the protocol, each party is supposed to hold the i-th part ω_i of the output. At least for the task of multi-party *quantum* computation, we then require the following to hold despite the presence of an *active* adversary controlling some $t < n$ parties:

- **Correctness and Soundness:** The adversary cannot affect the outcome of the computation for the honest parties beyond the ability to choose the corrupted parties' inputs.

- **Privacy:** The adversary does not learn anything about the privat inputs or outputs of the honest parties beyond what can be inferred from the adversary's own input and output.

This is in contrast to classical MPC, where it is commonly assumed that all parties obtain the full output of the computation. However, it is possible to use an MPQC protocol to simulate classical MPC, since any classical computation can be simulated by a quantum circuit, and we can in fact clone computational basis states (representing classical values, e.g., in $\mathbb{F}$) using only CNOTs, as noted below Theorem 1.1.2.

More formally, the authors of [LRW20] base their security definition for MPQC protocols on the work of [Unr10], [Can01], [MR92] and [Bea92], using a simulator-based security definition (Definition 3.3.1). The general idea is that a protocol is considered secure, if one cannot distinguish between a real execution of the MPQC protocol and that of an ideal MPQC protocol. Here, the real protocol could for example correspond to the MPQC protocol outlined at the end of this section, Protocol 4.

In the ideal model, all parties interact with an incorruptible oracle that perfectly implements the MPQC task. More specifically, all honest parties only interact with the oracle in a simple manner: they forward their input of the computation to the oracle, and later output whatever they receive from the oracle. The corrupted parties can apply any joint operation to their inputs before sending them to the oracle and similarly can apply arbitrary operations to their inputs after receiving them back from the oracle and before outputting a state.

As in the previous chapters, we consider a computationally unbounded, static quantum adversary $\mathcal{A}$ that selects up to t parties that she has complete control over. We denote the adversary in the real protocol by $\mathcal{A}_{real}$ and that in the ideal protocol by $\mathcal{A}_{ideal}$. Then we denote an MPQC protocol as ϵ-secure, if it satisfies the following property.

Definition 3.3.1 (ϵ-security of MPQC protocols, according to [LRW20]). We say that an MPQC protocol Π is ϵ-secure if, for any input state ρ and any real adversary $\mathcal{A}_{real}$, there exists an ideal adversary $\mathcal{A}_{ideal}$, such that the output state $\omega_{real} := \Pi_{real}(\rho)$ of the real protocol is ϵ-close with respect to the trace distance (Def. 1.1.7) to the output of the ideal protocol $\omega_{ideal} := \Pi_{ideal}(\rho)$,

$$\tfrac{1}{2}\|\omega_{real} - \omega_{ideal}\|_1 \leq \epsilon.$$

Note that proving security according to Definition 3.3.1 automatically proves correctness, soundness and privacy for MPQC protocols as defined in the beginning of this section. We remark that the security of an MPQC protocol proven according to Definition 3.3.1 follows the paradigm of *sequential composability* as opposed to the stronger notion of *universal composability*. The former guarantees that the protocol remains secure under sequential composition of the protocol, while universal composability guarantees security under *concurrent* execution with arbitrary other protocols (see [Lin17]).

Comparison to previous work

To the best of our knowledge, there exist three protocols that achieve universal multiparty quantum computation based on secret sharing techniques in a setting similar to ours (assuming, for example, a complete and private network for classical and quantum communication, no trusted third party, the possibility to compute arbitrary quantum circuits, etc.).[2] In [CGS02], the task of multi-party quantum computation is first introduced. A verifiable quantum secret sharing scheme based on a quantum error-correcting code is proposed and it is shown how to implement a universal set of gates, ancilla preparation and measurements on the shares held by the different parties securely. In [BCG+06], the authors use quantum authentication codes and approximate quantum error-correcting codes ([CGS05]) to construct a verifiable secret sharing scheme and then show how to securely implement a universal set of gates. Recently, Lipinska et al. proposed an MPQC scheme based on [CGS02] with the aim of improving the efficiency of the earlier protocol.

Due to the fact that the first and the most recent protocol, [CGS02] and [LRW20], are closely related and conceptually slightly differ from [BCG+06], we focus on the former two and start with a high-level discussion of some of their similarities and differences.

In both works, before running the protocol, the parties fix a quantum error-correcting code $\mathcal{C}$ that forms the basis of their VQSS scheme. This will be a so-called *Calderbank–Shor–Steane (CSS) code* $\mathcal{C}$ with n-dimensional codewords as well as certain additional properties. Each party encodes their input state using $\mathcal{C}$, thus creating n shares, and sends one to each of the other parties, keeping one for themselves. The parties then jointly verify the encoding of each input using the verification procedure of the VQSS scheme. To compute some quantum circuit $\mathfrak{R}$ on their inputs, they perform local operations on their shares to evaluate a logical version of the circuit. Finally, each party receives the shares corresponding to their part of the output and locally reconstructs ω_i using the error-recovery and decoding procedures of the code $\mathcal{C}$.

One important difference between the two protocols is the “unit of computation”: Crépeau et al. ([CGS02]) work with p-dimensional quantum systems for prime p, for which the set $\{|a\rangle \mid a \in \mathbb{Z}_p\}$ forms a basis, while Lipinska et al. ([LRW20]) work with single qubits. In this work we have generally focussed on the latter formalism and have for example discussed universal computation in terms of single and two-qubit gates. Note that we can always simulate higher-dimensional systems using qubits. The universal set of gates ([ABO08]) that is used in [CGS02] is defined on p-dimensional computational basis states, while Lipinska et al. use the set Clifford+T that is defined on single and two-qubit states.

In both protocols, the quantum operations necessary for universal quantum computation can be split into transversal and “almost transversal” operations, as the authors informally put it. The transversal operations can be implemented through local operations by the participants that do not require any communication by properties of the quantum error-correcting code $\mathcal{C}$ that is used. The respective operations are listed in Table 3.1. The “almost transversal” operations require some additional effort by the parties. For instance, a logical measurement may be implemented using just classical communication and local computations, or the implementation of a gate that does not have a transversal implementation may be reduced to the preparation and verification of a special ancilla state. In essence, for both protocols the hardness of implementing universal multi-party quantum computation is reduced to finding a secure distributed implementation of a gate that completes a universal set. In [CGS02], this is the (generalized) Toffoli gate while in [LRW20], it is the T gate. On a high level, these can be thought of as the quantum equivalent of a multiplication gate in classical MPC and we see that, in contrast to the classical case, various different approaches and gates exist. Some of the main similarities and differences between the two protocols can be found in Table 3.1.

[2]For example, [SGW20] uses Shamir’s scheme to share classical information and assumes the existence of a trusted server that aids in the sharing and computation phase.

Phase	Techniques	Crépeau, Gottesman, Smith '02	Lipinska, Ribeiro, Wehner '20
Sharing & Verification	Unit of computation	p-dimensional for prime p	2-dimensional (qubit)
	VQSS	CSS code (*Reed–Solomon code*)	CSS code
		3 encoding-levels	2 encoding-levels
Computation	Transversal gates	X^c, $(c-X)$, S_c, Z^c	Clifford group
	"Almost transversal"	$\mathcal{F}_r$, Toffoli, measurement, ancilla verification	T, measurement, ancilla verification
Reconstruction	Collect shares and decode	Error-correction procedure and decoder of CSS code	Error-correction procedure and decoder of CSS code

Table 3.1: Comparison of some key characteristics of the MPQC protocols proposed in [CGS02] and [LRW20].

In the following, we will discuss the protocol proposed by Lipinska et al. in [LRW20] in some more detail and analyse the mechanisms at play. Our choice of this example is motivated by the fact that [LRW20] is the more recent paper and is based on [CGS02]. It uses single qubits instead of higher-dimensional states and claims to achieve better security guarantees. However, we note that we found some issues regarding the distributed implementation of the T gate in [LRW20] that at the time of writing remain unsolved. In personal communication with the authors, the inconsistency has been confirmed and it has been announced that a fix will soon be published.

MPQC with few qubits ([LRW20])

In the following, we will consider each party's input ρ_i to be a single qubit state, as do [LRW20]. Recall from Section 3.1 that a universal set of gates can be used to approximate any quantum circuit arbitrarily close and that by the principle of deferred measurement, all measurements can be moved to the end of the circuit. We therefore assume w.l.o.g. that any quantum circuit $\mathfrak{R}$ on up to n inputs be composed of gates from the universal set (in this case the set Clifford+T) and that any party can locally apply all gates in that universal set. The assumptions that are made on the network are that each pair of participants is connected via both classical and quantum private and authenticated communication channels ([Can04], [BCG$^+$02]) and that all parties have access to an authenticated classical broadcast channel ([CGI$^+$99]) and a public source of randomness. Note that we will at any point in the following assume that the number of corrupted parties is restricted by $t < n/4$ $(< n/3)$ so that the former is feasible (Section 2.6), while the latter can be implemented via a classical MPC protocol (e.g., [RB89]). With respect to the security of classical protocols against quantum adversaries such as the classical MPC used as a subroutine here, [BCG$^+$06] notes that universally composable classical protocols remain secure in the quantum universal composability model, as established by the works of [Can01], [KLR10] and [Unr10][3].

[3]We remark that in a recent work on computationally secure MPQC by Dulek et al. ([DGJ$^+$20]), the instantiation of a post-quantum secure classical MPC was left as an open problem and that only very recently, the first such protocol has been proposed in the setting of computational security in [ABG$^+$20].

In their paper [LRW20], Lipinska et al. additionally assume that each party can locally process $\mathcal{O}(n^2)$ qubits. For the sake of completeness, we therefore add this assumption, as well as the assumption that each party can process and store classical information perfectly, even though our focus in this work is not the efficiency of any particular protocol. As mentioned above, we assume the adversary to corrupt some $t < n$ parties actively, allowing them to perform arbitrary (joint) quantum operations on their joint state, possibly including quantum side information. Thus, no computational assumptions are made on the adversary. Lastly, we assume the adversary to *statically* corrupt some chosen parties, meaning that the corrupted parties are determined at the beginning of the protocol and stay fixed during the execution.

Throughout this section we will see how various results of the previous chapters flow into the construction and help us understand the different techniques. In the following, the different ingredients and subroutines for the protocol will be discussed before giving a final succinct description of the protocol.

High-level description of the protocol

The construction in [LRW20] follows the high-level description outlined at the beginning of this section. The parties agree on a CSS quantum error-correcting code $\mathcal{C}$, and share and verify their inputs using the VQSS proposed in [LMRW20]. They compute a logical version of the circuit $\mathfrak{R}$ before sending their shares to the designated recipient within participants, who then reconstructs her part of the output of $\mathfrak{R}$ using error-recovery techniques of $\mathcal{C}$.

One important detail to be aware of is that throughout the protocol, a public set B is used to accumulate the positions of *apparent cheaters*, that is, indices of players for which erroneous behaviour (such as faulty sharings) has been detected. If B contains more than t apparent cheaters, the protocol aborts. This allows the adversary to force an abort of the protocol by introducing sufficiently many errors. Thus, in any of the following subroutines, the set B can be assumed to contain the indices of apparent cheaters of previous subroutines. B is initiated as an empty set at the beginning of the protocol, and is first used during the sharing phase of the parties' inputs using the VQSS scheme.

We will now introduce the different ingredients and subroutines of the above outline of the protocol. In particular, we will discuss:

I. CSS codes;

II. transversal operations for CSS codes;

III. some aspects of the VQSS;

IV. ancilla preparation;

V. a protocol for distributed gate teleportation, and;

VI. a protocol for the verification of magic states.

I. CSS codes

CSS codes, owing their name to their inventors Robert Calderbank, Peter Shor and Andrew Steane ([Ste96], [CS96]) are quantum error-correcting codes that draw on the theory of classical error-correcting codes. A CSS code $\mathcal{C}$ is defined by two classical binary linear codes V and W for which the dual code $V^\perp$ of V satisfies $V^\perp \subseteq W$. Then $\mathcal{C}$, sometimes also denoted by CSS(V, W), can informally be defined as $\mathcal{C} := V \cap \mathcal{F}W$, the set of n-qubit states that yield a classical codeword in V when measured in the standard basis, and a codeword in W when measured in the Fourier basis[4].

CSS codes are part of the larger class of quantum error-correcting codes called *stabilizer codes*. The stabilizer formalism was developed by Daniel Gottesman [Got96] and we refer the reader to sources such as [Got06] or [NC16] for a more in-depth introduction. Even though stabilizer codes are less general than arbitrary quantum error-correcting codes, their mathematical structure helps studying their properties. Instead of defining the quantum error-correcting code by means of an encoding procedure, stabilizer codes are defined by a stabilizer S that is an abelian subgroup of the Pauli group $\mathcal{P}_n$ acting on n qubits.

Definition 3.3.2 (Stabilizer code). Let $S \subseteq \mathcal{P}_n$ be an abelian subgroup of the Pauli group that does not contain $-I$ or $\pm iI$. Let $\mathcal{C}(S) := \{|\psi\rangle \ : \ M|\psi\rangle = |\psi\rangle \ \forall M \in S\}$. Then $\mathcal{C}(S)$ is a *stabilizer code* and S is its *stablizer*.

By definition, the codewords of a stabilizer code are exactly those states that are in the $+1$–eigenspace of all elements of the stabilizer. An error E acting on a codeword moves the codeword into the -1–eigenspace of any stabilizer element that *anticommutes* with E: two operators U and V are said to anticommute if $UV = -VU$. Therefore, for an error E and a stabilizer element M we have

$$M(E|\psi\rangle) = -EM|\psi\rangle = -E|\psi\rangle .$$

Thus, measuring the eigenvalues of the elements $M \in S$ gives us information about the error that occured. The full set of eigenvalues resulting from such a measurement can be represented as a binary vector and is called the *error syndrome*, similar to what was discussed in Section 2.5.1. Note that the error syndrome does not reveal any information about the codeword but only about the error that occured.

In the stabilizer formalism, CSS codes are defined as follows. Recall from Definition 2.2.3 that for a binary classical $[n, k, d]$ linear code, the parity check matrix H is a $(n - k) \times n$ binary matrix and every classical codeword v must satisfy $Hv = 0$. Now, as noted above, a CSS code is defined by two binary linear codes V and W with parity check matrices H_V and H_W. Then we can define a stabilizer of a quantum code by converting the rows of the two parity matrices into Pauli operators as follows: for each row in each matrix, replace each 0 by an identity matrix. For rows in H_V replace each 1 with a Z operator, while replacing each 1 in H_W by an X operator. Using this construction, the Z generators (obtained from H_V) provide the ability to correct bit flip (X) errors, while the X generators can be used to correct phase flip errors (Z).

Definition 3.3.3 (CSS code). Let V be an $[n, k_1, d_1]$ binary linear code with associated parity check matrix H_1, and let W be an $[n, k_2, d_2]$ binary linear code with parity check matrix H_2 such that $V^\perp \subseteq W$. Then transforming the rows of the parity check matrices H_1 and H_2 as described above defines the stabilizer generators of a $[[n, k_1 + k_2 - n, k]]$ stabilizer code, that is, a quantum error-correcting code that encodes $k_1 + k_2 - n$ qubits into n qubits and minimum distance k, where $k \geq \min(d_1, d_2)$.

[4]The Fourier basis is the basis of n qubit states given by applying the Fourier-transform $\mathcal{F}$ to the standard basis given by $\{\ |j\rangle \mid j = 0, \ldots, n-1\}$,

$$\mathcal{F}|j\rangle := \frac{1}{\sqrt{2n}} \sum_{k=0}^{2^n-1} \omega^{jk} |k\rangle , \qquad \text{where } \omega = \exp^{2\pi i/2^n} .$$

Recall from Section 1 that for $n = 2$ the standard single qubit basis is given by $\{|0\rangle, |1\rangle\}$ and the Fourier basis by $\{|+\rangle, |-\rangle\}$.

In the above definition, the condition $V^\perp \subseteq W$ ensures that the stabilizer elements commute. An example of a CSS code is Shor's 9–qubit code discussed in Section 2.5.1, with its stabilizer generators given be the observables sketched in that section. Another famous example is Steane's 7-qubit code, which, as we will see in this section, has the properties required for Lipinska et al.'s MPQC construction. Its stabilizer is derived from the parity check matrix H of the $[7,4,3]$ Hamming code (Section 2.2) for both V and W, and the resulting stabilizer is depicted in Table 3.2.

$$H := \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Element	Operator
S_1	$XIXIXIX$
S_2	$IXXIIXX$
S_3	$IIIXXXX$
S_4	$ZIZIZIZ$
S_5	$IZZIIZZ$
S_6	$IIIZZZZ$

Table 3.2: Parity check matrix of the $[7,4,3]$ Hamming code and the corresponding generators of the stabilizer of Steane's 7-qubit code.

II. Transversal operations for CSS codes

For any CSS code, certain logical gates can always be implemented transversally. In the following, we list some of the most important ones. The following description of transversal gates is with respect to a single layer of encoding using a fixed CSS code $\mathcal{C}$. However, as noted in Table 3.1, the MPQC protocol in [LRW20] and in particular the underlying VQSS scheme ([LMRW20]) use a two-layer encoding, such that each share of a secret is again secret-shared among the parties using the same code $\mathcal{C}$, as explained in more detail in the next subsection. Regarding the application of transversal gates, however, this does not change much: to implement a certain logical gate U, each party locally applies those gates that correspond to a transversal implementation of U in the first layer of encoding. Hence, in this section it is sufficient to discuss transversal gates for one layer of encoding, since in the VQSS scheme, both layers of encoding are performed using the same code $\mathcal{C}$. For a detailed discussion on what the transversal implementations of the following logical operations are, we refer the interested reader to [Got96], [Got06] or [NC16].

- **Pauli gates:** any operator in the Pauli group can be implemented transversally by locally applying Pauli gates for any *stabilizer code* [Got06]. Note again that the transversal implementation of some Pauli gate U is not necessarily $U^{\otimes n}$, but might be the tensor product of some other single qubit gates.[5]

- **CNOT:** The logical $CNOT$ gate between two logical qubits is transversal for any CSS code ([Got98b]) and can be implemented by applying $CNOT$ gates to the corresponding qubits in the encodings of two different states.

[5]The transversal implementation (if possible) of any specific logical gate often involves some choice, as a stabilizer code is defined only as a code space, instead of defining a mapping that determines the encoded basis vectors $|\bar{0}\rangle$ and $|\bar{1}\rangle$ ([Got98b]). Moreover, any encoded operation on codewords of a stabilizer code is only unique up to multiplication by elements in the stabilizer, since for any operation E, any element $M \in S$ and any codeword $|\psi\rangle$, we have $ME|\psi\rangle = M|\psi\rangle$.

Intuitively, multi-qubit gates such as the two-qubit CNOT gate should be prone to errors propagating "through the gate", as discussed in Section 3.2. The fact that any CSS code allows for a transversal implementation of the CNOT gate is one of the reasons that these codes are frequently used in the context of fault-tolerant computation as well as multi-party quantum computation ([CGS02], [LRW20]). Indeed, it can be shown that CSS codes are the only type of codes from the larger class of stabilizer codes that allow for a transversal CNOT implementation ([Got98b]). Figure 3.2 illustrates what we mean by applying CNOT gates transversally, i.e., qubit-wise, to the physical qubits.

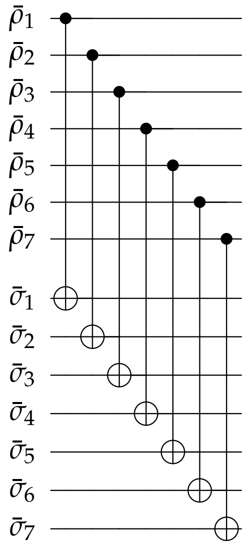

Figure 3.2: Transversal implementation of the logical CNOT gate for Steane's 7-qubit code on two encoded logical qubits $\bar{\rho}$ and $\bar{\sigma}$. Here, $\bar{\rho}_i$ (and $\bar{\sigma}_i$, respectively) denotes the single-qubit share of the encoding of $\bar{\rho}$ (and $\bar{\sigma}$) held by player i. We see that all operations can be performed locally by player i on her shares.

In addition to the code $\mathcal{C}$ being a CSS code encoding single qubits, Lipinska et al. require the following two properties:

(1) $\mathcal{C}$ uses the same classical code to correct X and Z errors, that is, $V = W$.

(2) The weight[6] of the stabilizer generators of $\mathcal{C}$ is a multiple of 4, and the encoded Pauli operators $\bar{X}$ and $\bar{Z}$ have weight 1 mod 4, or 3 mod 4.

Condition (1) ensures that the logical Hadamard gate H can be applied transversally by having the players locally apply H on their single-qubit shares, while condition (2) guarantees that the phase gate P can be applied transversally ([Got98b]).

Given a CSS code with the additional properties outlined above, we therefore see that we can implement the logical gates H, P and $CNOT$ by having all parties apply local operations without requiring any communication. An example of such a code is given by the 7-qubit code. By definition, it fulfills property (1), and we see from Table 3.2 that each stabilizer generator has weight 4. Furthermore, the logical Pauli operators X and Z can be implemented transversally by applying $X^{\otimes 7}$ and $Z^{\otimes 7}$, respectively, to the physical qubits. Both operators have weight 3 mod 4.

As discussed in the previous section, these gates generate the Clifford group, so at this point, any quantum circuit composed of Clifford gates can be computed by applying

[6]Recall from Section 2.5.1 that the weight of a Pauli operator denotes the number of non-identity tensors in the tensor product.

the transversal operations on the shares. In order to complete this set to a universal set of gates, Lipinska et al. give a protocol for distributed T-gate teleportation similar to what we described above, as we will see in part V of this section.

Finally, we already mentioned in Table 3.1 that in [LRW20], measurements in the computational basis are considered "almost transversal". What this means is that for any CSS code, a logical measurement, that is, a measurement of the logical qubit, can be performed by having the parties locally measure their shares, and broadcasting their classical measurement outcome. For a CSS code based on classical codes V and W, the encoding of a computational basis state is an equal superposition of all the words in some particular coset of $W^\perp$, such as $\sum_{y \in W^\perp} |x + y\rangle \,/\, \sqrt{|W^\perp|}$ for some $x \in V$. Measuring all n qubits in such a codeword individually, in the computational basis, therefore gives a classical codeword in form of a binary string from that particular coset of $W^\perp$. Broadcasting the individual measurement results (in form of bits) allows the parties to locally reconstruct the outcome of the logical measurement using the error correction and decoding procedures of the underlying classical codes. Similarly, measuring the encoding of a superposition of computational basis states $\alpha_0 |0\rangle + \alpha_1 |1\rangle$ yields the coset corresponding to $|b\rangle$ with probability $|\alpha_b|^2$.

This operation is not transversal since it requires classical communication of the measurement outcomes. However, note that by properties of the broadcast channel (Def. 2.8.1), corrupted parties cannot communicate differing measurement outcomes to different parties and by properties of the classical error-correcting codes, all honest parties reconstruct the same logical measurement outcome. In the MPQC protocol, including the VQSS protocol, we will see that these "almost transversal" measurements are exclusively used for measuring ancilla qubits that are independent of the parties' inputs or any intermediate computations and therefore neither reveal any information to the adversary other than what she can already deduce from the shares in her possession, nor do they disturb these states. See also [LMRW20], [Smi01] and [CGS02] for further information.

III. The VQSS scheme

In this section, we discuss some details of the VQSS used in [LRW20]. The scheme was first introduced by Crépeau, Gottesman and Smith in [CGS02] and was subsequently modified so as to require less qubits of workspace per participant in [LMRW20]. An outline of the VQSS scheme can be found in Protocol 1.

Recall that a quantum error-correcting code with distance d corrects at most $\lfloor \frac{d-1}{2} \rfloor$ errors, so that the above VQSS scheme tolerates at most $t \leq \lfloor \frac{d-1}{2} \rfloor < \frac{1}{4}$ active corruptions. As described in Protocol 1, the parties create a double-encoding of each input ρ. We denote the double-encoded global state jointly held by the parties as $\bar{\bar{\Psi}}$. We will follow [LRW20] and use the index $i = 1, \ldots, n$ to denote the encoding performed by party i, while $l = 1, \ldots, n$ is used to denote the share held by party l. Thus, $\bar{\bar{\Psi}}_{i_l}$ denotes the share of the encoding done by i that is in possession of party l.

The verification of the encoding of an input ρ is described in detail in [LMRW20] and, for the sake of brevity, we will omit a full description here, but will focus on the essential steps. To verify an encoding, ancilla qubits are encoded with the same CSS code $\mathcal{C}$. Joint operations are performed on certain shares of the encoded input and ancillas that are selected using the public source of randomness. The ancillas are measured and the classical outcomes are broadcasted. Thus, the VQSS is one instance in the MPQC protocol where the assumptions of a public source of randomness and a classical broadcast channel are used.

The measurement outcomes of CSS-encoded states produce codewords in the classical code V when measured in the standard basis (W when measured in the Fourier

Protocol 1 (VQSS in [LRW20], based on [CGS02], [LMRW20])
Input: Single-qubit state ρ of the dealer D, an $[[n, 1, d]]$ CSS code $\mathcal{C}$.
Output: An encoding of ρ, encoded twice with $\mathcal{C}$, each party holding n single-qubit shares of the encoding. A public set B' containing positions of apparent cheaters in the first level of encoding.

1. **Sharing**:
   The dealer D encodes her input ρ using the encoding procedure of $\mathcal{C}$, creating n single-qubit shares, and sends the shares of the encoding to the other parties, keeping one for herself. Then, each party again encodes each share they received using the same code $\mathcal{C}$, and sends the resulting n shares to the other parties. At this point, each party holds n single-qubit shares for each single-qubit input of a party in the protocol. A visualisation of the two-layered encoding for some single input can be seen in Figure 3.3.

2. **Verification**:
   The participants jointly verify that the encoding was performed correctly, that is, that the sharing corresponds to a codeword of $\mathcal{C}$. During the verification procedure, a public set B' is constructed in which a set of *apparent cheaters* that correspond to positions of faulty shares in the first layer of encoding. The verification procedure is iterated $s^2 + 2s$ times, with s denoting the security parameter.

basis), so that applying an error correction procedure of the classical code allows the parties to identify shares in the first level of encoding that carry errors. These positions are recorded in the public set B' of apparent cheaters. The size of B' determines whether the sharing is accepted or rejected. If $|B'| \leq t$, the dealer passes the verification phase. Indeed, note that in the first layer of encoding, there is no way to tell apart any errors introduced by corrupted parties from those introduced by the dealer. If after the verification procedure there are more than t apparent cheaters in the first encoding-layer, that is, $|B'| > t$, the sharing is rejected. For the MPQC protocol, Protocol 4, this means that the protocol aborts.

Now, since we assumed that there are at most t corrupted parties, these can introduce at most t errors in each second-level encoding after the verification phase. Therefore, we know that if the dealer passes the verification, then we can correct errors at both the first and second level of encoding so that at the end of the protocol, there will be a state to reconstruct. Using the verification procedure explained in detail in [LMRW20], preparing and measuring ancilla qubits is repeated $s^2 + 2s$ times, s being the security parameter. The security parameter s determines the probability with which a dishonest dealer is caught (the soundness of the VQSS), which is given by $1 - 2^{-\Omega(s)}$ ([LMRW20], Lemma 1).

In the context of the MPQC protocol, the VQSS scheme is used for the sharing and verification of any inputs such as the initial inputs of the parties as well as for introducing ancilla qubits into the circuit. Thus, if the VQSS protocol is used at any intermediate stage of the circuit, the set of apparent cheaters B' resulting from an execution of Protocol 1 is used to update the global set of apparent cheaters of the MPQC protocol.

As remarked earlier, the adversary can force the protocol to abort by having a corrupted party provide a faulty sharing of their input, or introducing too many errors

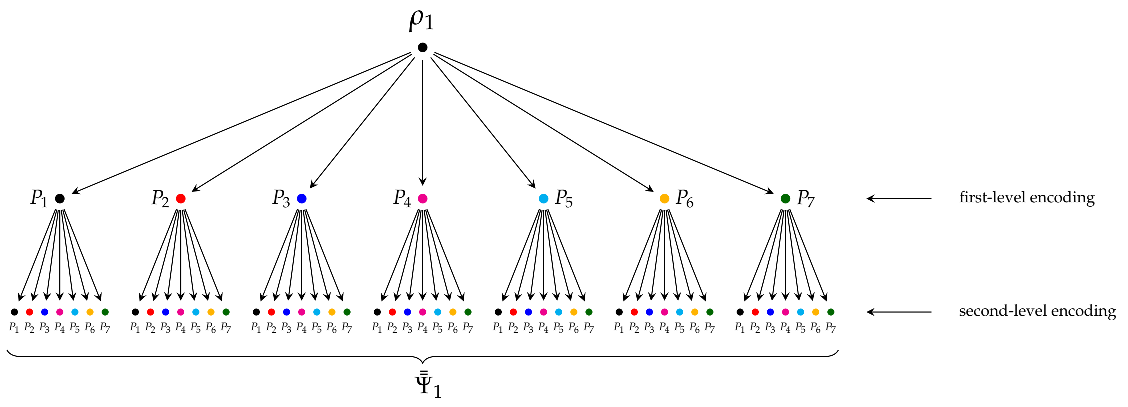

Figure 3.3: Schematic overview of the two-layered encoding of some single-qubit input ρ_1 held by party P_1, as seen in [LMRW20]. The global state after the second layer of encoding is denoted by $\bar{\bar{\Psi}}_1$. The colours represent the player that performs the encoding, or holds the final physical qubit, and each dot represents a single-qubit share.

through the corrupted parties. Lipinska et al. note that introducing the possibility of an abort allowed their construction to drastically reduce the amount of quantum communication needed for the VQSS protocol. In [LRW20], they also discuss an approach that would allow to remove the abort event at the cost of more quantum communication, using ideas of [CGS02].

IV. Verification of logical 0

Part of the quantum circuit model for universal computation that is considered in this work is the ability to introduce ancilla qubits in the logical 0 state, $|\bar{\bar{0}}\rangle$ in our notation, into the circuit. Lipinska et al. note that verifying whether a shared state is the logical 0 can be achieved using the VQSS without much further work. As we will see in the final overview of the MPQC protocol, the idea is that a party that has not been recorded as an apparent cheater is chosen at random using the public source of randomness, who then prepares a sharing of the logical 0. The parties jointly verify that the shared state indeed is $|\bar{\bar{0}}\rangle$ by checking whether the classical measurement outcomes produced by the VQSS all reconstruct to 0, see [CGS02], [Smi01], [LMRW20]. Note that if the measurement outcomes do reconstruct to 0, then the post-measurement encoded state is $|\bar{\bar{0}}\rangle$, too. The modified verification procedure for ancilla qubits will be denoted by VQSS(0).

V. Distributed gate teleportation

We have discussed in the previous section that the set Clifford+T is universal and in Part II of the current section we have seen that Clifford operations are transversal by properties of the chosen CSS code $\mathcal{C}$. In [LRW20], Lipinska et al. give a distributed version of the circuit depicted in Figure 3.1 for implementing the T gate on some single qubit. The underlying idea is that all gates and measurements in 3.1 are transversal for $\mathcal{C}$, thus reducing the problem of performing a T gate to the verification of the encoded magic state $|\bar{\bar{m}}\rangle$ that we will see in Part VI.

More specifically, the protocol for distributed gate teleportation works as follows (see Protocol 2). We assume the double-encoded input states $\bar{\bar{\Psi}}$ and $|\bar{\bar{m}}\rangle$ to be shared by the same dealer D. All parties jointly apply the transversal implementation of the logical CNOT gate with $\bar{\bar{\Psi}}$ as control and $|\bar{\bar{m}}\rangle$ as target. The parties measure their shares of the target qubit in the standard (Z) basis, and announce the measurement outcome using the classical broadcast channel. They publicly check whether the measurement has collapsed the state to an encoded $|\bar{\bar{0}}\rangle$ or $|\bar{\bar{1}}\rangle$ by using the classical decoder associated to the underlying classical code of $\mathcal{C} = CSS(V, V)$ twice, and checking whether the string of measurement outcomes reconstructs to 0 or 1. If the outcome is 0, no correction is necessary, while if the outcome is 1, the Clifford gate $XP^\dagger$ is applied transversally to the control qubit (note that $XP^\dagger \in$ Clifford because of the group structure and $X = HP^2H$).

Protocol 2 (Distributed gate teleportation, denoted *GTele*.)
Input: $\bar{\bar{\Psi}}$, $|\bar{\bar{m}}\rangle$ shared by D and verified by the parties using VQSS (Protocol 1). Set of apparent cheaters B resulting from verification of $\bar{\bar{\Psi}}$ and $|\bar{\bar{m}}\rangle$.
Output: Logical T gate applied to logical input state, $T(\bar{\bar{\Psi}})$.

1. Transversal CNOT: each party l, for shares coming from party i:
   (a) applies CNOT with $|\bar{\bar{m}}\rangle_{i_l}$ as control and $\bar{\bar{\Psi}}_{i_l}$ as target;
   (b) measures the target qubit in the Z basis and broadcasts the classical result.
2. The broadcasted classical values yield words v_i. The parties publicly check at which positions errors occurred, using the decoder of the classical code underlying $\mathcal{C}$ and update B with new apparent cheaters. They decode and obtain a classical value a:
   (a) If $a = 0$, no correction is applied.
   (b) If $a = 1$, the parties transversally apply the logical $XP^\dagger$ to the control qubit.

VI. Magic state verification

We now discuss the final missing ingredient for completing the MPQC protocol: the verification of the magic state that allows the parties to jointly and securely compute T gates. As noted earlier, this is the part in [LRW20] that currently contains some inconsistencies and for which the authors have announced to have found a fix in personal communication. We present the intended verification mechanism as described in the original paper, [LRW20], and point out open questions.

In [LRW20], verifying that the state shared by some dealer D indeed is a sharing of the magic state $|m\rangle$, i.e., that it indeed is $|\bar{\bar{m}}\rangle$ relies on the idea of a stabilizer-measurements in quantum error-correction. Consider the single-qubit gate $XP^\dagger$ mentioned earlier that has $|m\rangle$ as an eigenstate. Then a simple computation shows that $|+\rangle|m\rangle$ is an eigenstate of the controlled $XP^\dagger$ gate, denoted $C - XP^\dagger$, where $|+\rangle$ is the control and $|m\rangle$ is the target qubit. Lipinska et al. claim that the above are both $+1$ eigenstates and thus conclude that if we prepare the control qubit as $|+\rangle = H|0\rangle$, perform $C - XP^\dagger$ and measure the control qubit in the $\{|+\rangle, |-\rangle\}$ basis (or apply H and then measure in the standard basis), this tells us two things: if the target qubit was the magic state in the first place, we will always measure $|+\rangle$. On the other hand, if the target qubit was *not* the magic state to begin with, then if we still measure $|+\rangle$, we know that the target qubit is projected onto $|m\rangle$. Thus, obtaining the measurement outcome associated to $|+\rangle$ we know that the target qubit is in state $|m\rangle$. Figure 3.4 depicts the quantum circuit for magic state verifiction as described in [LRW20], and Protocol 3 describes the distributed version of that circuit.

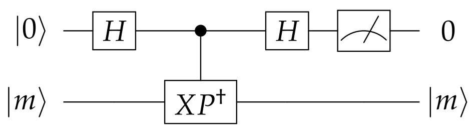

Figure 3.4: The magic state verification circuit as described in [LRW20].

Protocol 3 (Verification of magic states for the T gate, denoted *VMagic*.)
Input: Single-qubit states $|0\rangle$ and $|m\rangle$ prepared by D, CSS code $\mathcal{C}$, set of apparent cheaters B.
Output: Verified encodings of logical states $|\bar{\bar{0}}\rangle$ and $|\bar{\bar{m}}\rangle$, updated global set B.

1. The parties run VQSS(0) with $|0\rangle$ as input, and VQSS with $|m\rangle$ as input. They update B with the sets of apparent cheaters B_0 from VQSS(0), and B_m from verifying $|m\rangle$.
2. For all shares coming from party i, each party l:
   (a) applies H to $|\bar{\bar{0}}\rangle_{i_l}$;
   (b) applies the local operation corresponding to the transversal implementation of $C - XP^\dagger$ to their shares of $|\bar{\bar{0}}\rangle$ and $|\bar{\bar{m}}\rangle$;
   (c) measures the shares of the control qubit in the Z-basis and broadcasts the result using the secure broadcast channel.
3. Broadcasted values yield classical words v_i. The parties publicly check at which positions the errors occurred, using the classical decoder associated to $\mathcal{C}$ and update B with the positions of errors. They decode the classical value a and:
   (a) If $a = 0$, they continue.
   (b) If $a = 1$, they set $B = [n]$ (this will cause the MPQC protocol to abort after the computation phase, see Protocol 4).

Now the authors note that this works, if $C - XP^\dagger$ can be implemented transversally, and that this works for all Clifford-stabilized states such as $|m\rangle$. This is where some clarification is needed: to the best of our knowledge, the state $|m\rangle$ is not a $+1$ eigenstate of $XP^\dagger$ gate but has $e^{i7\pi/4}$ as its eigenvalue, see also Appendix A for some calculations. This difference might already cause problems when applying the controlled version of the gate. Furthermore, we think that the controlled $XP^\dagger$ gate cannot be transversal for the chosen CSS code for the following reason. We already noted that no quantum error-correcting code allows for transversal implementations of a universal set of gates (Theorem 3.2.1). Thus, we must have $C - XP^\dagger \in$ Clifford. But any controlled gate can be decomposed into successive controlled unitaries as $C - XP^\dagger = CNOT \cdot (C - P^\dagger)$, which by the group structure of the Clifford group implies that $C - P \in$ Clifford. The latter gate is known not to be part of the Clifford group which, in our opinion, would lead to a contradiction. At the time of writing, a fix for these potential issues has not been published but is announced.

The full MPQC protocol

Having discussed all subroutines and ingredients, we move on and put together the pieces and describe the MPQC protocol proposed by [LRW20]. Recall that the goal of the protocol is to perform some quantum circuit $\mathfrak{R}$ on private, possibly entangled single-qubit input states $\rho_1, \ldots, \rho_n$ coming from parties $1, \ldots, n$. Following the choice of a universal set of gates, we require the circuit $\mathfrak{R}$ to only include gates from the universal set Clifford+T and, as discussed at the beginning of this section, we assume all measurements to be single-qubit measurements performed on behalf of the parties after reconstruction. As before, we assume the chosen CSS code to have distance d. A complete description of the protocol is given in Protocol 4 and consists of the three phases *Sharing and verification*, *Computation*, and *Reconstruction*, thus resembling the classical share–compute–reveal MPC-paradigm.

Sharing and verification: In the first phase of the protocol, all parties share their inputs ρ_i acting as dealers D_i, and jointly verify for each dealer D_i whether they honestly performed the sharing, i.e., whether there are fewer or equal to $t \leq \lfloor \frac{d-1}{2} \rfloor$ errors in the first level of the encoding. First, for each i, a set containing the positions of apparent cheaters B_i is publicly constructed by the parties, which are then merged to a global set B. If $|B| \leq t$, then the fact that B accumulates all apparent cheaters over all executions of the VQSS scheme implies that each honest party holds shares with at most t errors in the same positions in the first level of encoding of the respective shares, and the protocol continues. If not, then all (honest) parties replace their shares with $|0\rangle$, which is necessary for the security proof of [LRW20].

Computation: In the computation phase, the parties apply the gates in the circuit $\mathfrak{R}$ successively on the shares of the double-encoded and verified inputs, in the way described previously. We remark again that the transversal implementation of a logical gate does not have to be the local application of that same gate on each share. Instead, in order to apply a transversal gate, each party locally applies those gates that translate to a transversal implementation of the desired logical gate on the shares in the first level of encoding. Furthermore, note that the set of apparent cheaters B is cumulative throughout the protocol and accumulates errors from the various subroutines such as VQSS(0), VMagic and GTele. If at any point during the computation we have $|B| > t$, all parties replace their shares with $|0\rangle$. If, after the computation phase, B contains more than t apparent cheaters, the protocol aborts. Otherwise, the parties proceed with the reconstruction.

Reconstruction: After the sharing and verification of inputs, the adversary can still introduce errors on shares held by corrupted parties. During the reconstruction phase, the parties receive all shares corresponding to their output of the circuit. After the computation phase, the parties know that $|B| < t$, because otherwise the protocol would have aborted. Furthermore, in this phase the adversary cannot force an abort anymore, since the best the adversary can do is withhold or provide faulty shares to some recipient. Each party applies the error-correction of $\mathcal{C}$ to identify any further errors in the first level of encoding, of which at most t might occur. In particular, shares that are not received in the first place are identified easily. Each party then randomly selects $n - 2t$ of the shares that have not been recorded in B and reconstructs their output.

For the sake of readability we use the following notation in the reconstruction phase of Protocol 4. Let $\omega_i := \mathrm{Tr}_{[n]\setminus i}(\omega)$, corresponding to the outcome of each party i. Moreover, $\bar{\omega}_i$ denotes the global state in the first level of encoding, while $\bar{\bar{\omega}}_i$ denotes the global state in the second level of encoding. $\bar{\bar{\omega}}_i$ consists of shares $\bar{\bar{\omega}}_{i_{j,l}}$, with $i, j = 1, \ldots, n$, where j

means that the second encoding is performed by party j, and l describes the party that is in possession of share $\bar{\bar{\omega}}_{i_{j,l}}$. Similarly, $\bar{\omega}_i$ consists of shares $\bar{\omega}_{i_j} = \mathrm{Tr}_{[n]\setminus j}(\bar{\omega}_i)$ for $j = 1, \ldots, n$.

Protocol 4 (MPQC Protocol, [LRW20])
Input: Private input ρ_i for every party $i \in [n]$, circuit $\mathfrak{R}$, CSS code $\mathcal{C}$.
Output: Party i receives their output ω_i, or the protocol is aborted.
Sharing and verification

1. Each party i runs the sharing of VQSS (Protocol 1) on their input ρ_i, acting as dealer D_i, creating logical $\bar{\bar{\Psi}}_i$ encoded twice with $\mathcal{C}$.
2. For each input $\bar{\bar{\Psi}}_i$, the parties publicly create sets $B_{i,l}$ containing positions of second-level errors resulting from the execution of the verification of VQSS. If $|B_{i,l}| > t$, then party l is added to the set of apparent cheaters B_i for dealer D_i. After all n executions of VQSS, a global set $B := \bigcup_i B_i$ of apparent cheaters is created. If $|B| > t$ then the parties know that they will abort the computation. They replace all shares in their possession with $|0\rangle$.

Computation

3. For every Clifford gate C of $\mathfrak{R}$, the parties apply C transversally through operations on their local qubits. For every T gate in $\mathfrak{R}$ applied to the input of D_i:
   (a) D_i creates $|0\rangle$ and $|m\rangle$. The parties jointly run VMagic (Protocol 3) and update B with apparent cheaters resulting from VMagic. If $|B| > t$, they replace all their shares with $|0\rangle$.
   (b) The parties run GTele (Protocol 2) on their shares of $\bar{\bar{\Psi}}_i$ and $|\bar{\bar{m}}\rangle$. B is updated with apparent cheaters from GTele. If $|B| > t$, they replace all their shares with $|0\rangle$ and do not apply the correction operation (treating the measurement outcome as 0, see Step 2, Protocol 2).
4. For every ancilla qubit $|0\rangle$ required for $\mathfrak{R}$, a party $i \notin B$ is chosen at random using the public source of randomness. Party i runs VQSS(0) as the dealer, creating a verified $|\bar{\bar{0}}\rangle$. She updates B with the set of apparent cheaters resulting from VQSS(0). If $|B| > t$, the parties replace all their shares with $|0\rangle$.
5. If $|B| > t$, the protocol aborts. Otherwise, continue.

Denote the global outcome of the computation by $\bar{\bar{\omega}}$ and let $\bar{\bar{\omega}}_i := Tr_{[n]\setminus i}(\bar{\bar{\omega}})$, corresponding to the outcome of each party i.
Reconstruction

6. Each party sends all her shares of $\bar{\bar{\omega}}_i$ she holds to D_i.
7. Each D_i:
   (a) For each share $\bar{\bar{\omega}}_{i_{j,l}}$, $l \in [n]$, coming from the encoding of party $j \notin B$, D_i runs the error-correcting circuit of $\mathcal{C}$. She creates a set of errors $\tilde{B}_{i,j}$ such that $B_{i,j} \subseteq \tilde{B}_{i,j}$. For each $k \in [n]$, if $\tilde{B}_{i,j} \leq t$, then the errors are correctable. D_i corrects them and decodes, thus obtaining the k-th share $\bar{\omega}_{i,k}$ of $\bar{\omega}_i$. Otherwise, D_i adds j to the global set B.
   (b) For all $j \notin B$, D_i randomly selects $n - 2t$ shares of $\bar{\omega}_i$, and applies an erasure-recovery circuit, obtaining ω_i.

In light of the unanswered questions regarding the protocol VMagic we omit the security proof of [LRW20].

3.4 Theoretical feasibility of MP(Q)C

We devote the last section of this chapter and of this work to the discussion of feasibility results of both classical MPC and MPQC. For the former task, feasibility results are well-known for various adversarial settings, and we will provide an overview and references to the relevant results. For multi-party *quantum* computation, fewer results are known. We will present these and relate them to existing work on MPQC in general and (verifiable) secret sharing-based protocols in particular, as well as point out some bounds that still remain to be established.

Same as for the earlier sections, we restrict ourselves to information-theoretically secure protocols, and assume private and authenticated communication channels between each two parties. For classical MPC we only assume classical (private, authenticated) channels while for MPQC protocols, both classical and quantum communication channels are available. We assume all communication to be *synchronous* so that the time passed before a message is received is bounded by some constant. This ensures that messages are sent and received within the same round of communication. In Section 2.6, we discussed that assuming the availability of a classical broadcast channel is a strong assumption, as it allows for better feasibility results for interactive protocols, such as verifiable secret sharing schemes. As we will see, the same holds for MP(Q)C protocols, and we will explicitly mention when we add this assumption here. Furthermore, we again discuss different adversarial models such as semi-honest (passive) and malicious (active) adversaries, but will always assume the adversary to be static, that is, that the set of corrupted parties is fixed at the beginning of the protocol and remains the same throughout its execution. Finally, we do not make any assumptions on the computational (or storage) capacities of either a classical or quantum adversary for MPC or MPQC protocols.

Feasibility results for classical MPC

Upper bounds on the maximum number of corrupted parties for various types of adversaries are well-known for classical MPC protocols. These have been established in a series of papers in the late 1980s in, e.g., [GMW87], [CDvdG87], [BGW88] and [CCD88]. Again, note that these results apply to MPC protocols that can implement arbitrary arithmetic circuits. For specific classical functions, better results might be feasible.

Recall from the discussion on the feasibility of secret sharing schemes in Section 2.6 that we distinguish between passive and active adversaries corrupting a number $t < n$ of parties. For passive (semi-honest) adversaries, it is known that general secure multi-party computation is feasibile, if, and only if, at most a strict minority of parties is corrupted. In order to tolerate an active (malicious) adversary, at most $t < n/3$ corruptions can be tolerated without adding any further assumptions, and protocols are known that attain this bound. It has also been shown that relaxing the security requirements, or making additional assumptions on the available resources, can increase these bounds. Recall from Section 3.3 that we require a general MPC protocol to satisfy correctness and privacy. In particular, correctness guarantees that the honest parties receive the correct output from the computation despite any malicious actions on part of the adversary. In the context of MPC, in the presence of active adversaries, this property is also called *guaranteed output delivery* or *robustness*. It can be shown that, if one relaxes the requirements from robustness to *fairness*, that is, guaranteeing that corrupted parties only receive their output if the honest parties do, too, then the upper bound can be lifted from $t < n/3$ to $t < n/2$ ([FGMvR02], [CL17]). Alternatively, one can make additional assumptions to lift the bound. As discussed earlier, assuming the existence of a broadcast channel can be used to increase the number of tolerable corruptions for verifiable secret sharing schemes. Similarly, assuming the availability of broadcast channel increases the feasibility of information-theoretically secure MPC from $t < n/3$ to $t < n/2$ ([Bea89], [RB89]). Lindell provides a succinct overview of techniques, definitions and feasibility results for classical MPC in [Lin20]. Table 3.3 provides an overview of bounds presented above.

Classical MPC	**Passive adversary**	**Active adversary**
Unconditional security	$t < n/2$	$t < n/3$ ($t < n/2$ either without robustness or assuming broadcast)

Table 3.3: Parameter restrictions for different levels of security for MPC protocols for arbitrary arithmetic circuits, as established in [GMW87], [CDvdG87], [BGW88], [CCD88]. All bounds in this table are tight.

Feasibility results for quantum MPQC

For quantum protocols, some general impossibility results are known for the case of two-party quantum computations that extend to general unconditionally secure MPQC protocols. The most important result in this regime concerns the feasibility of unconditionally secure *quantum bit commitment*.

In (classical) bit commitment, two parties Alice and Bob run a protocol that consists of two phases. In the *commit phase*, Alice commits to some chosen bit b, which she can then later open to Bob in the *opening phase*. Such a protocol is called *binding*, if Alice cannot change her mind after the commit phase, and it is called *hiding*, if Bob cannot learn the bit b from the information he is given during the commit phase. An example of a physical protocol for this task is given by Alice writing her secret bit b on a piece of paper which she then locks into a safe. She sends the safe to Bob, keeping the key to complete the commit phase. In the opening phase, Alice simply sends the key to Bob, who unlocks the safe and obtains b. The goal of *quantum* bit commitment is to realise the above protocol (still with a classical bit b as input to one of the players) by using quantum communication and computation, secure in the presence of a quantum adversary. Initial security claims were thought to prove an advantage of quantum protocols over classical protocols (see, e.g., [BCJL93]). Soon after publication of these results, however, Lo, Chau ([LC97], [Lo96]) and Mayers ([May97]) showed that the purported proofs of security were wrong, and that unconditionally secure quantum bit commitment as well as other, more general classes of quantum two-party computations are impossible altogether if one of the parties is actively corrupted.

These results can be used to show the impossibility of a general, unconditionally secure MPQC protocol that realises arbitrary quantum circuits in the presence of an active adversary controlling a dishonest majority, i.e., $t \geq n/2$, as remarked in [CGS02] and [LRW20]. First, consider the case of a two-party protocol, $n = 2$. Suppose there is an unconditionally secure MPQC protocol Π secure against one actively corrupted party. Then Π can be used to implement *quantum 1-out-of-2 oblivious transfer (quantum OT, [LC98])*. Similar to quantum bit commitment, the goal of quantum OT is to solve a problem from classical cryptography using quantum means, secure against a quantum adversary. The desired functionality can be described as follows: Alice starts off with bits a_0 and a_1 and Bob with a single bit b. Bob's bit b indicates his choice of bit of Alice that he would like to receive from Alice without Alice learning which bit Bob requested and received, and without Bob learning both bits in Alice's possession. That is, at the end of the protocol, we want Bob to be in possession of a_b without having Alice learn b and without Bob learning $a_{b\oplus 1}$.

Theorem 3.4.1 (MPQC implies quantum OT). *Let Π be an unconditionally secure MPQC protocol for two parties that is secure against a dishonest majority, that is, against an adversary that actively corrupts one of the two parties. Then Π implies the existence of an unconditionally secure quantum 1-out-of-2 oblivious transfer protocol.*

Idea of the proof, due to [Kar20]. We use Π to implement quantum OT between two parties Alice and Bob. At the beginning of the protocol, Alice holds two bits a_0 and a_1 and Bob holds a bit b indicating his bit of choice in Alice's possession. The two parties jointly compute the classical function $f(a_0, a_1, b, b_1) := a_b \oplus b_1$ where b_1 is an auxiliary input provided by Bob. Since any classical computation can be simulated by a quantum circuit, Π can be used to implement f. Now, if Bob randomises the selection of b_1, then Bob's choice of bit a_{b_0} is completely hidden from Alice by this one-time pad construction, and the two parties can safely reconstruct the outcome of the computation publicly. On the other hand, Bob knows b_1 and can thus recover his bit of choice a_{b_0}. $\square$

Now it is known that in the quantum world, quantum 1-out-of-2 OT implies quantum bit commitment (in fact it can be shown that they are equivalent ([Unr10], [BS16]), which does not hold for their classical counterparts), thus showing the impossibility of general two-party MPQC that tolerates an actively corrupted party. Alternatively, [Lo96] explicitly proves the impossibility of unconditionally secure quantum OT without using the equivalence to quantum bit commitment. This impossibility result for a dishonest majority can be extended to multi-party protocols ($n > 2$) by splitting the full set of parties into two sets of size at least $n/2$, and having each set simulate the action of one participant in a two-party quantum OT protocol.

Note that this impossibility result only applies to general, unconditionally secure MPQC protocols, for which no relaxation of security requirements like fairness or security with abort has been allowed. In a setting where we do allow for such relaxations, better results might be feasible. We are not aware of any theoretical work analysing such relaxations, but do note that [CGS02] obtains general MPQC with robustness tolerating at most $t < n/6$ active corruptions while [LRW20] claims to achieve up to $t < n/4$ active corruptions and allows for an abort event forced by the adversary. The significance of this observation, however, is unclear, as we still await an update on the potential inconsistencies in [LRW20] pointed out in Section 3.3 and, as the authors note themselves, the relaxation to security was chosen to improve efficiency of the protocol. Moreover, if we relax from information-theoretical security to computational security, then protocols for general MPQC exist that allow for up to $n-1$ actively corrupted parties ([DNS12], [DGJ+20]). Note again that for specific multi-party quantum computations, better results might be feasible, but these are outside the scope of this work.

We are not aware of any work on unconditionally secure MPQC protocols that consider a passive adversary. However, any of the previous protocols in [CGS02], [BCG+06] and [LRW20] are in particular secure against a passive instead of an active adversary. Assuming a passive adversary should also allow us to relax from a VQSS to a quantum secret sharing scheme and should therefore in particular allow us to tolerate up to $t < n/2$ corrupted parties (Theorem 2.5.6). In contrast, [CGS02] (and [LRW20]) tolerate up to $t < n/6$ ($t < n/4$ if the open problem pointed out in Section 3.3 can be solved) active corruptions using quantum error-correcting codes for the VQSS, and [BCG+06] tolerates up to $t < n/2$ active corruptions using approximate error-correcting codes. We remark that all three of the above MPQC protocols add the assumption of a classical broadcast channel, but in fact only [BCG+06] requires this assumption, since for the other two protocols, it is assumed that $t < n/6 < n/4 < n/3$ parties are corrupted and therefore broadcast can be implemented, as explained in Section 2.6. Table 3.4 gives an overview of the feasibility results for MPQC which resembles that of Table 3.3. However, possible benefits of further relaxations, such as relaxing the requirement of guaranteed-output-delivery seen in Table 3.3, remain to be established.

MPQC	**Passive adversary**	**Active adversary**
Unconditional security	$t < n/2$	$t < n/3$ ($t < n/2$ assuming broadcast)

Table 3.4: Parameter restrictions for different levels of security for MPQC protocols for arbitrary quantum circuits. All bounds in this table are tight, as achieved by [BCG$^+$06].

Finally, we note that any MPQC protocol based on (verifiable) quantum secret sharing schemes trivially has to satisfy the feasibility bounds outlined in the previous chapter, Table 2.4. A direct reduction from an MPQC protocol that is unconditionally secure against an active adversary to a VQSS scheme (or a passively secure MPQC protocol to a quantum secret sharing scheme) would immediately imply these results, and we expect such a reduction to hold. We conclude with an overview of the tolerable number of corruptions in previous work on unconditionally secure MPQC protocols[7].

Protocol	**Number of active corruptions**
[CGS02], [Smi01]	$t < n/6$
[BCG$^+$06]	$t < n/2$ (assuming broadcast)
[LRW20]*	$t < n/4$

Table 3.5: Existing work on unconditionally secure MPQC against an active adversary. [LRW20] remains to be confirmed, as pointed out in Section 3.3.

[7]For [LRW20], a complete proof of security was not available at the time of writing, see Section 3.3 for a discussion.

4. Discussion and open questions

The goal of this work was to create an extensive analysis of the existing work on multiparty quantum computation and contrast it with classical MPC, with a focus on unconditionally secure protocols and their building blocks. In the first part of this analysis, Chapter 2, we have seen that one of the fundamental primitives used for classical MPC, secret sharing, exists in a very similar form in the quantum world and that by the laws of quantum information, these can be derived from quantum error-correcting codes in a straightforward manner. This view has also been confirmed in the study of MPQC protocols in Chapter 3. Furthermore, we have contrasted feasibility results for classical and quantum secret sharing schemes and its variants (e.g., verifiable secret sharing) and we observed that, for quantum protocols, the no-cloning theorem frequently determined the amount of tolerable corruptions (Table 2.4).

In the context of secure multi-party classical and quantum computations, we observed a general structural similarity, but also some subtle differences. One important difference is that for classical MPC, every party is typically expected to learn the final outcome of the computation, and it is required that no party may learn any more information than what can be inferred from their input *and* the output of the function. More generally, one can require that only dedicated participants receive parts of the final output. Such a protocol would immediately imply the existence of the first approach in which the final function value is known to all parties. The quantum no-cloning theorem generally prevents every party from learning the same quantum state that results from the joint computation, as this would require the ability to copy arbitrary quantum states. Instead, in MPQC, each party is expected to obtain a certain part of the output and may not learn any more about the other parties' in- or outputs other than what can be inferred from their own (or all of the adversary's) in- or output (Section 3.3).

The design of information-theoretically secure MPQC protocols, however, exhibits strong similarities to that of classical MPC protocols (Section 3.3). At the time of writing, all proposed unconditionally secure MPQC protocols follow a quantum share–compute–reveal paradigm that starts off with a verifiable quantum secret sharing scheme for which it is shown that arbitrary quantum circuits can be implemented securely. In particular when comparing the classical gates of addition and multiplication to quantum gates (unitaries), we see that for many current constructions, a large set of operations (e.g., the Clifford gates) can be implemented securely, in a distributed manner, on top of the VQSS scheme, thus resembling the inherent linearity of linear secret sharing schemes. To complete such sets to a universal set (Section 3.1), just like the multiplication complements addition to a universal set of classical gates, we see that oftentimes only one additional gate has to be implemented securely for which more complex constructions are required (Section 3.2). In fact, this gap between "easy" and "hard" gates can also be observed in other MPQC constructions for computationally secure protocols such as [DNS12] and

[DGJ$^+$20] who also use the special commutation rules of the Clifford group to show how the universal set Clifford+T can be implemented securely. Finally, we observed that feasibility results for general MPQC seem to resemble those of classical MPC but still require some investigation into whether security relaxations might lift these bounds (Section 3.4).

Open topics

We conclude with a list of topics that are open for further research, with a focus on and around the design and security of (existing) MPQC protocols. We begin with possible improvements of existing protocols and note that we have discussed in Section 2.6 that current constructions for VQSS all bear some probability of error. We are not aware of any no-go theorem that would prevent an error-free solution. In [CGS02] and [Smi01], the design of an error-free protocol for the task of VQSS is left as an open question, merely noting that techniques used for the design of a classical VSS scheme with zero error in [BGW88] may not apply to the quantum case. As seen in subsequent work in [LMRW20] and [LRW20], this question seems to remain unanswered. In Chapter 3 and in particular in Section 3.3, we have focussed almost exclusively on the secure implementation of the universal set of gates Clifford+T. However, other universal sets of unitaries, or even other models of quantum computation, remain to be studied. The choice of a different class of quantum error-correcting codes and, therefore, of a different universal set of gates might lead to interesting, possibly more efficient MPQC protocols, in particular because different universal sets of gates might be more efficient in approximating arbitrary unitaries than others.

Lipinska et al. note in [LRW20] that the effect of noise in quantum communications, computations and storage needs to be analysed carefully, in particular with respect to the effect on the feasibility and security of MPQC protocols. Recently, [Iñe20] proposed an analysis of VQSS schemes in noisy quantum networks and its effects on the maximum number of tolerable cheaters. The authors of [LRW20] expect these results to generalise to the MPQC setting. Furthermore, assuming noisy or bounded quantum storage on part of the adversary has proven fruitful in the design of other quantum-cryptographic primitives, see, e.g., [STW09], [DFSS05]. Both of the above references study information-theoretically secure protocols in which, e.g., a fixed amount of noise is assumed to occur on any stored qubit. By contrast, one could also study a storage model in which the quality of stored quantum information decays progressively with time, which suggests a shift from unconditional to computational security. We are not aware of any work in this setting, nor of any work on general MPQC in the noisy or bounded quantum storage model.

On a more fundamental level, we have noted in Section 2.5.3 that privacy for quantum secret sharing schemes is more readily obtained from QECCs than for their classical analogues, and that it would be interesting to study whether there is a form of quantum equivalent to the dual distance of ECCs that determines the secret sharing scheme's privacy. Furthermore, we have seen in Section 3.4 that the knowledge of feasibility results for MPQC protocols remains rather limited compared to those for classical MPC. In particular, a rigorous investigation into the effects of relaxing security requirements for the task of MPQC, such as allowing the adversary to force an abort of the protocol, appears to remain open.

Appendices

A. Appendix

Frequently used unitaries

Common single and two-qubit unitary gates used in this work:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$
$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}.$$

Note that in the literature, the P gate can also be found as S gate and that, for historical reasons, the $T = \sqrt{P}$ gate is sometimes referred to as $\pi/8$ gate. According to [NC16], the reason for this is that up to a global phase of $e^{i\pi/8}$, it has $e^{-i\pi/8}$ and $e^{i\pi/8}$ on its diagonal. We also used the CNOT gate,

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Moreover, for any unitary $U = \begin{pmatrix} u_1 & u_2 \\ u_3 & u_4 \end{pmatrix}$, the controlled version, denoted $C - U$, is a two-qubit unitary that is given by

$$C - U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_1 & u_2 \\ 0 & 0 & u_3 & u_4 \end{pmatrix},$$

and applies U to the second qubit if the first qubit is $|1\rangle$ and I else (similarly for reversed control- and target qubits).

Issue in MPQC paper, Section 3 of [LRW20]

We have that

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad P^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix},$$

and therefore

$$XP^\dagger = \begin{pmatrix} 0 & -i \\ 1 & 0 \end{pmatrix}.$$

Moreover,

$$|m\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\frac{\pi}{4}}|1\rangle),$$

so that

$$\begin{aligned} XP^\dagger |m\rangle &= \frac{1}{\sqrt{2}}(|1\rangle - ie^{i\frac{\pi}{4}}|0\rangle) \\ &= -ie^{i\frac{\pi}{4}}\frac{1}{\sqrt{2}}(|0\rangle + e^{i\frac{\pi}{4}}|1\rangle) \\ &= -ie^{i\frac{\pi}{4}}|m\rangle \\ &= e^{i\frac{7\pi}{4}}|m\rangle. \end{aligned}$$

Hence, the outcome of circuit 3.4 is given by:

$$e^{i\frac{7\pi}{4}}|m\rangle,$$

so we see a global phase of $c := e^{i\frac{7\pi}{4}}$. This might cause a problem when applying $(C - XP^\dagger)$ to $|+\rangle|m\rangle$:

$$(C - XP^\dagger)|+\rangle|m\rangle = \frac{1}{\sqrt{2}}((C - XP^\dagger)|0\rangle|m\rangle + (C - XP^\dagger)|1\rangle|m\rangle) = \frac{1}{\sqrt{2}}(|0\rangle|m\rangle + c|1\rangle|m\rangle).$$

Bibliography

[ABG^{+}20] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. Post-quantum multi-party computation in constant rounds. *CoRR*, abs/2005.12904, 2020.

[ABO08] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM Journal on Computing*, 38(4):1207–1282, Jan 2008.

[AJL^{+}12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. 7237:483–501, 2012.

[BBM92] Charles H. Bennett, Gilles Brassard, and N. David Mermin. Quantum cryptography without bell's theorem. *Phys. Rev. Lett.*, 68:557–559, Feb 1992.

[BCG^{+}02] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam D. Smith, and Alain Tapp. Authentication of quantum messages. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 449–458. IEEE Computer Society, 2002.

[BCG^{+}06] Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam D. Smith. Secure multiparty quantum computation with (only) a strict honest majority. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 249–260. IEEE Computer Society, 2006.

[BCJL93] G. Brassard, C. Crepeau, R. Jozsa, and D. Langlois. A quantum bit commitment scheme provably unbreakable by both parties. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 362–371, 1993.

[Bea89] Donald Beaver. Multiparty protocols tolerating half faulty processors. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 560–572. Springer, 1989.

[Bea91] Donald Beaver. Efficient multiparty protocols using circuit randomization. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 420–432. Springer, 1991.

[Bea92] Donald Beaver. Foundations of secure interactive computing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 377–391, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[Bei11] Amos Beimel. Secret-sharing schemes: A survey. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology - Third International Workshop, IWCC 2011, Qingdao, China, May 30-June 3, 2011. Proceedings*, volume 6639 of *Lecture Notes in Computer Science*, pages 11–46. Springer, 2011.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. In *Managing Requirements Knowledge, International Workshop on*, page 313, Los Alamitos, CA, USA, jun 1979. IEEE Computer Society.

[BM84] G. R. Blakley and Catherine A. Meadows. Security of ramp schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 242–268. Springer, 1984.

[Bri89] Ernest F. Brickell. Some ideal secret sharing schemes. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, volume 434 of *Lecture Notes in Computer Science*, pages 468–475. Springer, 1989.

[BS16] Anne Broadbent and Christian Schaffner. Quantum cryptography beyond quantum key distribution. *Des. Codes Cryptogr.*, 78(1):351–382, 2016.

[Can01] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, page 136, USA, 2001. IEEE Computer Society.

[Can04] Ran Canetti. Universally composable signature, certification, and authentication. In *17th IEEE Computer Security Foundations Workshop, (CSFW-17 2004), 28-30 June 2004, Pacific Grove, CA, USA*, page 219. IEEE Computer Society, 2004.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988.

[CDD$^+$15] Ronald Cramer, Ivan Bjerre Damgård, Nico Döttling, Serge Fehr, and Gabriele Spini. Linear secret sharing schemes from error correcting codes and universal hash functions. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International*

Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II, volume 9057 of *Lecture Notes in Computer Science*, pages 313–336. Springer, 2015.

[CDF01] Ronald Cramer, Ivan Damgård, and Serge Fehr. On the cost of reconstructing a secret, or VSS with optimal reconstruction phase. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 503–523. Springer, 2001.

[CDN15] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[CDvdG87] David Chaum, Ivan Damgård, and Jeroen van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In Carl Pomerance, editor, *Advances in Cryptology - CRYPTO '87, A Conference on the Theory and Applications of Cryptographic Techniques, Santa Barbara, California, USA, August 16-20, 1987, Proceedings*, volume 293 of *Lecture Notes in Computer Science*, pages 87–119. Springer, 1987.

[Cev11] Alfonso Cevallos. Reducing the Share Size in Robust Secret Sharing. Master's thesis, Universiteit Leiden, 2011.

[CFOR12] Alfonso Cevallos, Serge Fehr, Rafail Ostrovsky, and Yuval Rabani. Unconditionally-secure robust secret sharing with compact shares. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 195–208. Springer, 2012.

[CGI$^+$99] Ran Canetti, Juan A. Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas. Multicast security: A taxonomy and some efficient constructions. In *Proceedings IEEE INFOCOM '99, The Conference on Computer Communications, Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, The Future Is Now, New York, NY, USA, March 21-25, 1999*, pages 708–716. IEEE Computer Society, 1999.

[CGL99] Richard Cleve, Daniel Gottesman, and Hoi-Kwong Lo. How to share a quantum secret. *Physical Review Letters*, 83(3):648–651, Jul 1999.

[CGMA85] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 383–395. IEEE Computer Society, 1985.

[CGS02] Claude Crépeau, Daniel Gottesman, and Adam D. Smith. Secure multi-party quantum computation. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 643–652. ACM, 2002.

[CGS05] Claude Crépeau, Daniel Gottesman, and Adam D. Smith. Approximate quantum error-correcting codes and secret sharing schemes. In Ronald

Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 285–301. Springer, 2005.

[Che19] Mahdi Cheraghchi. Nearly optimal robust secret sharing. *Des. Codes Cryptogr.*, 87(8):1777–1796, 2019.

[CL17] Ran Cohen and Yehuda Lindell. Fairness versus guaranteed output delivery in secure multiparty computation. *J. Cryptol.*, 30(4):1157–1186, 2017.

[CPS20] T.-H. Hubert Chan, Rafael Pass, and Elaine Shi. Sublinear-round byzantine agreement under corrupt majority. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II*, volume 12111 of *Lecture Notes in Computer Science*, pages 246–265. Springer, 2020.

[CPT$^+$13] Qi Chen, Dingyi Pei, Chunming Tang, Qiang Yue, and Tongkai Ji. A note on ramp secret sharing schemes from error-correcting codes. *Math. Comput. Model.*, 57(11-12):2695–2702, 2013.

[CS96] Calderbank and Shor. Good quantum error-correcting codes exist. *Physical review. A, Atomic, molecular, and optical physics*, 54 2:1098–1105, 1996.

[CvL14] Carlo Cafaro and Peter van Loock. A simple comparative analysis of exact and approximate quantum error correction. *Open Syst. Inf. Dyn.*, 21(3), 2014.

[Deu89] D. Deutsch. Quantum computational networks. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 425(1868):73–90, 1989.

[DFSS05] Ivan Damgård, Serge Fehr, Louis Salvail, and Christian Schaffner. Cryptography in the bounded quantum-storage model. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 449–458. IEEE Computer Society, 2005.

[DGJ$^+$20] Yfke Dulek, Alex B. Grilo, Stacey Jeffery, Christian Majenz, and Christian Schaffner. Secure multi-party quantum computation with a dishonest majority. 12107:729–758, 2020.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory*, 22(6):644–654, 1976.

[DNS10] Frédéric Dupuis, Jesper Buus Nielsen, and Louis Salvail. Secure two-party quantum evaluation of unitaries against specious adversaries. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 685–706. Springer, 2010.

[DNS12] Frédéric Dupuis, Jesper Buus Nielsen, and Louis Salvail. Actively secure two-party evaluation of any quantum operation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 794–811. Springer, 2012.

[DSS98] David P. DiVincenzo, Peter W. Shor, and John A. Smolin. Quantum-channel capacity of very noisy channels. *Phys. Rev. A*, 57:830–839, Feb 1998.

[dW11] Ronald de Wolf. Quantum computing: Lecture notes. *ArXiv*, abs/1907.09415, 2011.

[EK09] Bryan Eastin and Emanuel Knill. Restrictions on transversal encoded quantum gate sets. *Phys. Rev. Lett.*, 102:110502, Mar 2009.

[EM96] Artur Ekert and Chiara Macchiavello. Quantum error correction for communication. *Physical Review Letters*, 77, 10 1996.

[EPR35] A. Einstein, B. Podolsky, and N. Rosen. Can quantum-mechanical description of physical reality be considered complete? *Phys. Rev.*, 47:777–780, May 1935.

[FGG$^+$06] Matthias Fitzi, Juan A. Garay, Shyamnath Gollakota, C. Pandu Rangan, and K. Srinathan. Round-optimal and efficient verifiable secret sharing. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 329–342. Springer, 2006.

[FGMvR02] Matthias Fitzi, Nicolas Gisin, Ueli M. Maurer, and Oliver von Rotz. Unconditional byzantine agreement and multi-party computation secure against dishonest minorities from scratch. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*, pages 482–501. Springer, 2002.

[FLM86] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Comput.*, 1(1):26–39, 1986.

[FM89] Paul Feldman and Silvio Micali. An optimal probabilistic algorithm for synchronous byzantine agreement. In Giorgio Ausiello, Mariangiola Dezani-Ciancaglini, and Simona Ronchi Della Rocca, editors, *Automata, Languages and Programming, 16th International Colloquium, ICALP89, Stresa, Italy, July 11-15, 1989, Proceedings*, volume 372 of *Lecture Notes in Computer Science*, pages 341–378. Springer, 1989.

[FY20] Serge Fehr and Chen Yuan. Robust secret sharing with almost optimal share size and security against rushing adversaries. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 470–498. Springer, 2020.

[Gai08] Frank Gaitan. *Quantum Error Correction and Fault Tolerant Quantum Computing*. CRC Press, 2008.

[GBP97] M. Grassl, Th. Beth, and T. Pellizzari. Codes for the quantum erasure channel. *Phys. Rev. A*, 56:33–38, Jul 1997.

[GC99] Daniel Gottesman and Isaac L. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390–393, 1999.

[GGOR13] Juan A. Garay, Clint Givens, Rafail Ostrovsky, and Pavel Raykov. Broadcast (and round) efficient verifiable secret sharing. In Carles Padró, editor, *Information Theoretic Security - 7th International Conference, ICITS 2013, Singapore, November 28-30, 2013, Proceedings*, volume 8317 of *Lecture Notes in Computer Science*, pages 200–219. Springer, 2013.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987.

[Got96] Daniel Gottesman. Class of quantum error-correcting codes saturating the quantum hamming bound. *Physical Review A*, 54(3):1862–1868, Sep 1996.

[Got98a] Daniel Gottesman. The Heisenberg representation of quantum computers. In *22nd International Colloquium on Group Theoretical Methods in Physics*, pages 32–43, 7 1998.

[Got98b] Daniel Gottesman. Theory of fault-tolerant quantum computation. *Phys. Rev. A*, 57:127–137, Jan 1998.

[Got00] Daniel Gottesman. Theory of quantum secret sharing. *Physical Review A*, 61(4), Mar 2000.

[Got06] D. Gottesman. Quantum error correction and fault tolerance. *Encyclopedia of Mathematical Physics*, page 196–201, 2006.

[Got09] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation, 2009.

[GRR98] Rosario Gennaro, Michael O. Rabin, and Tal Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In Brian A. Coan and Yehuda Afek, editors, *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing, PODC '98, Puerto Vallarta, Mexico, June 28 - July 2, 1998*, pages 101–111. ACM, 1998.

[GY89] R. L. Graham and A. C. Yao. On the improbability of reaching byzantine agreements. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 467–478, New York, NY, USA, 1989. Association for Computing Machinery.

[Ham50] R. W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, 1950.

[HBcvB99] Mark Hillery, Vladimír Bužek, and André Berthiaume. Quantum secret sharing. *Phys. Rev. A*, 59:1829–1834, Mar 1999.

[HLKB16] Wentao Huang, Michael Langberg, Jörg Kliewer, and Jehoshua Bruck. Communication efficient secret sharing. *IEEE Trans. Inf. Theory*, 62(12):7195–7206, 2016.

[IMN$^+$05] Hideki Imai, Jörn Müller-Quade, Anderson C. A. Nascimento, Pim Tuyls, and Andreas J. Winter. An information theoretical model for quantum secret sharing. *Quantum Inf. Comput.*, 5(1):69–80, 2005.

[Iñe20] Álvaro Gómez Iñesta. Verifiable hybrid secret sharing in the presence of noise, 2020. Master Thesis, Delf University of Technology.

[Kar20] Martti Karvonen. Answer on StackExchange Quantum Computing, 2020.

[KL97] Emanuel Knill and Raymond Laflamme. Theory of quantum error-correcting codes. *Phys. Rev. A*, 55:900–911, Feb 1997.

[KLR10] Eyal Kushilevitz, Yehuda Lindell, and Tal Rabin. Information-theoretically secure protocols and security under composition. *SIAM J. Comput.*, 39(5):2090–2112, 2010.

[LC97] Hoi-Kwong Lo and H. F. Chau. Is quantum bit commitment really possible? *Phys. Rev. Lett.*, 78:3410–3413, Apr 1997.

[LC98] Hoi-Kwong Lo and H.F. Chau. Why quantum bit commitment and ideal quantum coin tossing are impossible. *Physica D: Nonlinear Phenomena*, 120(1):177 – 187, 1998. Proceedings of the Fourth Workshop on Physics and Consumption.

[Lin98] J. H. Van Lint. *Introduction to Coding Theory*. Springer-Verlag, Berlin, Heidelberg, 3rd edition, 1998.

[Lin17] Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography*, pages 277–346. Springer International Publishing, 2017.

[Lin20] Yehuda Lindell. Secure multiparty computation (MPC). *IACR Cryptol. ePrint Arch.*, 2020:300, 2020.

[LMRW20] Victoria Lipinska, Gláucia Murta, Jérémy Ribeiro, and Stephanie Wehner. Verifiable hybrid secret sharing with few qubits. *Phys. Rev. A*, 101:032332, Mar 2020.

[Lo96] Hoi-Kwong Lo. Insecurity of quantum secure computations. *CoRR*, quant-ph/9611031, 1996.

[LRW20] Victoria Lipinska, Jérémy Ribeiro, and Stephanie Wehner. Secure multiparty quantum computation with few qubits. *Phys. Rev. A*, 102:022405, Aug 2020.

[LSP82] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[Mar08] Keith M. Martin. Challenging the adversary model in secret sharing schemes. In *In Coding and Cryptography II, Proceedings of the Royal Flemish Academy of Belgium for Science and the Arts*, pages 45–63, 2008.

[May97] Dominic Mayers. Unconditionally secure quantum bit commitment is impossible. *Physical Review Letters*, 78(17):3414–3417, Apr 1997.

[McE02] Robert McEliece. *The Theory of Information and Coding*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2 edition, 2002.

[MR92] Silvio Micali and Phillip Rogaway. Secure computation. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 392–404, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[MS81] Robert J. McEliece and Dilip V. Sarwate. On sharing secrets and reed-solomon codes. *Commun. ACM*, 24(9):583–584, 1981.

[MSV20] Pasin Manurangsi, Akshayaram Srinivasan, and Prashant Nalini Vasudevan. Nearly optimal robust secret sharing against rushing adversaries. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 156–185. Springer, 2020.

[NC16] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information (10th Anniversary edition)*. Cambridge University Press, 2016.

[NN09] Svetla Nikova and Ventzislav Nikov. Secret sharing and error correcting. In Bart Preneel, Stefan M. Dodunekov, Vincent Rijmen, and Svetla Nikova, editors, *Enhancing Cryptographic Primitives with Techniques from Error Correcting Codes*, volume 23 of *NATO Science for Peace and Security Series - D: Information and Communication Security*, pages 28–38. IOS Press, 2009.

[NRS01] Gabriele Nebe, Eric M. Rains, and Neil J. A. Sloane. The invariants of the clifford groups. *Des. Codes Cryptogr.*, 24(1):99–122, 2001.

[Pre99] John Preskill. Lecture Notes for Physics 219/Computer Science 219: Quantum Computation, 1999. `http://www.theory.caltech.edu/~preskill/ph219/index.html#lecture`, accessed 2020-19-03.

[PSL80] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 73–85. ACM, 1989.

[RD96] Ari Renvall and Cunsheng Ding. The access structure of some secret-sharing schemes. In Josef Pieprzyk and Jennifer Seberry, editors, *Information Security and Privacy, First Australasian Conference, ACISP'96, Wollongong, NSW, Australia, June 24-26, 1996, Proceedings*, volume 1172 of *Lecture Notes in Computer Science*, pages 67–78. Springer, 1996.

[RGB04] Martin Rötteler, Markus Grassl, and Thomas Beth. On quantum MDS codes. In *Proceedings of the 2004 IEEE International Symposium on Information Theory, ISIT 2004, Chicago Downtown Marriott, Chicago, Illinois, USA, June 27 - July 2, 2004*, page 355. IEEE, 2004.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[RST05] Karin Rietjens, Berry Schoenmakers, and Pim Tuyls. Quantum information theoretical analysis of various constructions for quantum secret sharing. In *Proceedings of the 2005 IEEE International Symposium on Information Theory, ISIT 2005, Adelaide, South Australia, Australia, 4-9 September 2005*, pages 1598–1602. IEEE, 2005.

[RZLZ18] Pinshu Rui, Wen Zhang, Yanlin Liao, and Ziyun Zhang. Probabilistic quantum cloning of a subset of linearly dependent states. *The European Physical Journal D*, 72(2):26, 2018.

[SGW20] Xiuli Song, Rui Gou, and Aijun Wen. Secure multiparty quantum computation based on lagrange unitary operator. *Scientific Reports*, 10(1):7921, 2020.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[Sho94] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. Comput.*, 26:1484–1509, 1994.

[SI05] A. Steane and Ben Ibinson. Fault-tolerant logical gate networks for calderbank-shor-steane codes. *Physical Review A*, 72:052335, 2005.

[SIGA05] Valerio Scarani, Sofyan Iblisdir, Nicolas Gisin, and Antonio Acín. Quantum cloning. *Rev. Mod. Phys.*, 77:1225–1256, Nov 2005.

[Sin64] Richard C. Singleton. Maximum distance q -nary codes. *IEEE Trans. Inf. Theory*, 10(2):116–118, 1964.

[Smi01] Adam Smith. Multi-party quantum computation, 2001. Master Thesis, Massachusetts Institute of Technology.

[Spi97] Daniel A. Spielman. The complexity of error-correcting codes. In Bogdan S. Chlebus and Ludwik Czaja, editors, *Fundamentals of Computation Theory, 11th International Symposium, FCT '97, Kraków, Poland, September 1-3, 1997, Proceedings*, volume 1279 of *Lecture Notes in Computer Science*, pages 67–84. Springer, 1997.

[SS19] Kaushik Senthoor and Pradeep Kiran Sarvepalli. Communication efficient quantum secret sharing. *Phys. Rev. A*, 100:052313, Nov 2019.

[SS20] Kaushik Senthoor and Pradeep Kiran Sarvepalli. Universal communication efficient quantum threshold secret sharing schemes. *arXiv: Quantum Physics*, 2020.

[Ste96] Andrew M. Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452:2551 – 2577, 1996.

[STW09] Christian Schaffner, Barbara M. Terhal, and Stephanie Wehner. Robust cryptography in the noisy-quantum-storage model. *Quantum Inf. Comput.*, 9(11&12):963–996, 2009.

[SW02] Benjamin Schumacher and Michael D. Westmoreland. Approximate quantum error correction. *Quantum Inf. Process.*, 1(1-2):5–12, 2002.

[TW88] Martin Tompa and Heather Woll. How to share a secret with cheaters. *J. Cryptology*, 1(2):133–138, 1988.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 486–505. Springer, 2010.

[VBE95] Vlatko Vedral, Adriano Barenco, and Artur Ekert. Quantum networks for elementary arithmetic operations. *Physical Review A*, 54, 11 1995.

[Wal20] Michael Walter. Lecture Notes Quantum Information Theory (UvA Mastermath 2020), 2020. [https://staff.fnwi.uva.nl/m.walter/qit20/qit20.pdf; accessed 25-February-2020].

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982.