

**A unique normal form for
prime-dimensional qudit Clifford
ZX-calculus**



Boldizsár Poór

Wolfson College

Department of Computer Science

University of Oxford

A thesis presented for the degree of
MSc in Advanced Computer Science

Trinity 2022

Declaration

Before beginning with the thesis, we would like to state that numerous diagrams from Ref. [1] and Ref. [2] or the modified version of such diagrams are used throughout the paper.

The final word count is 13753.

Acknowledgement

First and foremost, I would like to thank John van de Wetering, Lia Yeh, and my supervisor, Aleks Kissinger, for many insightful discussions and the support through the creation of this thesis. I would like to thank Robert Booth for the valuable discussions regarding the odd prime dimensional clifford ZX-calculus. I sincerely appreciate the kindness of Letícia Poór, my sister, for proofreading my thesis. And lastly, I am incredibly grateful for the support of my parents, Titanilla Poór and Csaba Poór, through the year of my Master's degree.

Contents

1	Introduction	1
2	Preliminaries	5
2.1	Quantum computation	5
2.1.1	Basic concepts	6
2.1.2	Clifford gates	13
2.1.3	Quantum computation using qudits	14
2.2	The ZX-calculus	15
2.2.1	ZX-calculus basic concepts	16
2.2.2	Clifford computation	27
2.2.3	Clifford completeness	36
3	Qupit Clifford ZX-calculus	41
3.1	The calculus	42
3.1.1	Generators	42
3.1.2	Interpretation	43
3.1.3	Axioms	44
3.2	Useful derivations	45
3.2.1	Extension of axioms and further rules	45
3.2.2	Multipliers	51
3.3	Scalar completeness	55
3.4	Graph states	58

CONTENTS

3.5	Graph simplifications	59
3.5.1	Weighted local complementation	60
3.5.2	Local complementation	61
3.5.3	Pivoting	63
4	A normal form	69
4.1	AP form	69
4.2	Week simulation	72
4.3	Completeness	73
5	Conclusion	85
5.1	Evaluation of the thesis	85
5.1.1	Limitations	87
5.2	Future work	88

Chapter 1

Introduction

It is predicted that the increase in transistor density will start to slow down around 2025, which means that Moore's law is bound to end [3]. Then once, we will reach the ultimate limitations of physics and engineering which will prevent us from further increasing the performance of classical chips. To overcome this limitation, we are investigating other ways of computation. Among many others, one of the most promising paradigms is quantum computing.

Quantum computing is the use of quantum phenomena such as superposition and entanglement to perform computation. Computers that perform quantum computations are known as quantum computers. Problems related to such devices are in the scope of many studies of today as they could solve specific problems exponentially faster than classical computers. For example, Shor's algorithm presents us with a way to factorise large numbers efficiently, thus, beating many of the cryptographic systems in use today [4]. Furthermore, the simulation of complex molecules is not possible momentarily; however, a quantum computer could tackle such a task, thus, helping researchers design new drugs [5].

Clifford computation is a simple yet particularly important fragment of quantum computation with some differentiating properties. For example, the Gottesman-Knill theorem states that any Clifford state can be efficiently classically

simulated [6]. Although Clifford computation does not give us a computational advantage over classical computation, it is still a widely used part of quantum computing, as many quantum algorithms and protocols rely only on Clifford gates. For example, only Clifford unitaries are required to implement several quantum protocols. Just to name a few, we can implement superdense coding [7], quantum teleportation [8], and quantum key distribution [9] using only Clifford gates.

Most theory of quantum computation has so far focused on qubits; however, in recent years, there has been a surge of interest in studying quantum computation using d -dimensional systems, called *qudits*. For example, the qudit version of several quantum algorithms has been presented in Ref. [10]. We can not only implement the same quantum algorithms using a higher dimensional system but also enhance some, that were originally designed for qubits. For instance, qudits have some advantages in fault-tolerant quantum computing [11]. Also, there is a benefit to quantum communication using higher dimensional systems [12].

While qudit-based quantum computation sounds promising, it would fail to offer an alternative to qubits unless we were able to design and build such systems. In recent years, qudit-based quantum computation has been realised in competing paradigms of quantum computation, such as ion traps [13, 14], photonic devices [15], and superconducting devices [16, 17, 18, 19].

There are many ways to express quantum computation such as linear algebra, the quantum circuit model, or a quantum turning machine. The scope of this thesis is a relatively new language called the *ZX-calculus*, a graphical language for reasoning about quantum computation. It was first introduced by Bob Coecke and Ross Duncan as an extension of categorical quantum mechanics [20]. The *ZX-calculus* is used in many areas of quantum computing, for example, in measurement-based quantum computing [21, 22, 23], error correcting codes [24, 25, 26], quantum circuit optimisation [27, 28, 29], quantum natural language processing [30, 31], and it can also be used in the context of quantum machine

learning [32]. While the ZX-calculus is a rigorous language actively studied by many in academia, it is also an intuitive framework making it an ideal candidate for education.

Unlike the quantum circuit model, the ZX-calculus gives us tools to reason about quantum computation. It has built-in rewrite rules which allow us to transform one diagram into another while preserving the underlying linear map. Although these rules are an essential and powerful part of the calculus, they also give us a challenge to solve, *completeness*: The ZX-calculus is complete if, for two diagrams representing the same linear map, there is a sequence of rewrites that transforms one diagram into the other. The first completeness result was shown for the Clifford fragment in Ref. [33], then it was shown for the Clifford+T fragment in Ref. [34], and finally, an extended version of the calculus was shown to be universally complete [35]. In addition to the completeness results regarding the qubit ZX-calculus, there have been papers dealing with higher-dimensional systems. For example, the completeness of the qutrit Clifford fragment has been shown in Ref. [36] and for odd prime dimensional qudits, completeness has been shown in Ref. [2].

A rewrite rule that some completeness results rely on is called *local complementation*. This rewrite rule allows us to modify Clifford circuits in such a way that we can remove some parts of our diagram, resulting in a denser but smaller one. Another powerful rule that we can use to transform Clifford diagrams is called *pivoting*. Loosely speaking, by applications of these two rewrite rules, one can transform any Clifford circuit into one in a pseudo-normal form. If we transform the resulting diagram a bit further, we can obtain a unique normal form. Then, this unique normal form provides us with an obvious way to prove the completeness of the ZX-calculus. It is worth mentioning that further to normal forms, local complementation and pivoting are also used in other areas such as in optimisation algorithms.

A particular normal form, which we call the *AP-form* in this thesis, that has been proposed in Ref. [37] is the core focus of this thesis. This normal form has some interesting applications. For example, it was shown that this form can be used to efficiently weakly simulate Clifford circuits [38]. Also, this normal form has been used to show the completeness of the Clifford fragment of the ZX-calculus in the Quantum Software course. Lastly, Ref. [23] also presents this form in the ZX-calculus and also proves the completeness of the calculus using it.

Research regarding the ZX-calculus has also mostly focused on two-dimensional systems, e.g. qubits. Therefore, many transformations that are extensively used in the qubit ZX-calculus have not yet been presented for the qudit case, for example, local complementation and pivoting. Since the AP-form is constructed using the two rules mentioned above, its qudit version has not yet been presented either. Moreover, many optimisation algorithms designed in the qubit ZX-calculus cannot be implemented because of the lack of local complementation and pivoting.

In this thesis, we use a slightly modified version of the calculus presented Ref. [2]. This calculus is designed for odd prime dimensional qudits, and the modification is related to the representation of scalars. Using this calculus, we present and prove the correctness of the qudit version of local complementation and pivoting for odd prime dimensions. Moreover, we present the AP-form and its unique variant, the reduced AP-form, and show how to transform any Clifford circuit into these normal forms in the calculus. Furthermore, we present the qudit variant of the efficient weak simulation algorithm of Clifford states presented in Ref. [38]. Lastly, we also show an alternative completeness proof of the qudit ZX-calculus for odd prime dimensions using the reduced AP-form. While the completeness of the above-mentioned calculus has already been shown in Ref. [2], we aim to present a completeness proof that can be more accessible and easier to implement in practice.

Chapter 2

Preliminaries

In this chapter, we provide enough background to make the thesis self-contained. First, we introduce quantum computation, how it can be represented using linear algebra, what the Clifford fragment is, and a higher dimensional model of quantum computation using qudits. Then, we look at a graphical language for expressing and reasoning about quantum computation called the ZX-calculus. Using this graphical language, we explore how to represent the above-mentioned Clifford fragment and qudit quantum computation.

2.1 Quantum computation

Quantum computation is the use of quantum phenomena, such as superposition and entanglement, to perform computation. There are many ways to express quantum computation, be it low- or higher-level. For example, there are low-level languages like pure linear algebra or somewhat higher-level languages like the ZX-calculus. Subsequently, we introduce quantum computation using linear algebra which is a lower-level and more traditional language in the context of quantum computing.

2.1.1 Basic concepts

This section introduces the basic concepts and mathematical formulation of quantum computation, starting with what qubits are, how we can visualise and manipulate them, the quantum circuit model with some basic examples, and lastly, the phenomenon called quantum entanglement.

The qubit

The elementary unit of information in quantum computation is called the *qubit* or quantum bit – the quantum version of the classical binary bit [39]. Unlike a bit that can be in one state, 0, or the other, 1, a qubit can be in any coherent superposition of its basis states. *Superposition* is a fundamental principle of quantum mechanics that quantum states can be added together (‘superposed’) and the result will be another valid quantum state. The *computational basis states* are orthonormal vectors in a 2-dimensional vector space and are denoted $|0\rangle$ and $|1\rangle$. They represent the following values:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We can also combine single qubit basis states to form *product basis states*. For example, two qubits can be represented by the following product basis states:

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

We use the abbreviation $|ij\rangle := |i\rangle \otimes |j\rangle$ where \otimes is the tensor product operator. The consequence of the usage of the tensor product is that n qubit product basis states are represented as a 2^n -dimensional vector.

Any (pure) quantum state $|\psi\rangle$ can be represented as the linear combination

of $|0\rangle$ and $|1\rangle$, that is

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$$

where $\alpha, \beta \in \mathbb{C}$ are the probability amplitudes such that $|\alpha|^2 + |\beta|^2 = 1$. These amplitudes correspond to the likelihood of measurement outcomes. This means that the probability of a measurement outcome $|0\rangle$ is $|\alpha|^2$, and it is $|\beta|^2$ for the $|1\rangle$ outcome.

Bloch sphere

At first, it may seem like there are four degrees of freedom in a quantum state $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$ as both α and β are complex numbers. However, the constraint $|\alpha|^2 + |\beta|^2 = 1$ removes one degree of freedom. Therefore, we can represent qubits using an adequate coordinate system with three degrees of freedom. One option is to use the Hopf coordinates for this matter, that is

$$\alpha = e^{i\delta} \cos \frac{\theta}{2}, \quad \beta = e^{i(\delta+\varphi)} \sin \frac{\theta}{2}.$$

Moreover, the global phase $e^{i\delta}$ is unmeasurable for a single qubit that leaves us with two degrees of freedom that we can represent as

$$\alpha = \cos \frac{\theta}{2}, \quad \beta = e^{i\varphi} \sin \frac{\theta}{2}.$$

The above equation is a parameterization of the surface of a sphere, making it an intuitive way to visualise single qubits. This representation is called the *Bloch sphere*, that can be seen in Figure 2.1. The north and south poles of the Bloch sphere correspond to the computational basis states $|0\rangle$ and $|1\rangle$, respectively. Furthermore, we can represent any single-qubit (pure) quantum state on the surface of the Bloch sphere.

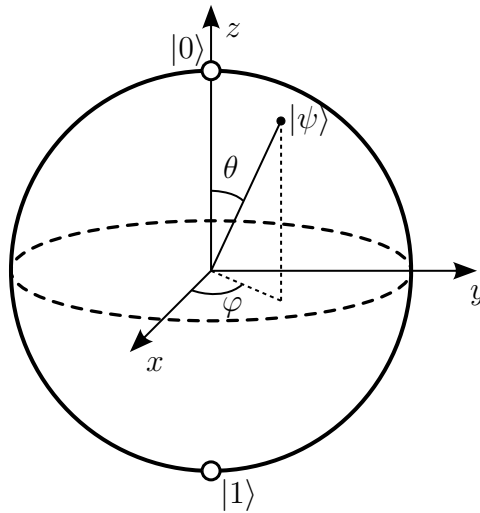


Figure 2.1: The Bloch sphere represents single-qubit pure quantum states.

Quantum logic gates

We can manipulate single qubits using different *gates*, sometimes called *operators*, such as the NOT gate (**X**) or the Hadamard gate (**H**). To describe how these operators evolve quantum states we can provide a mapping that represents how a gate changes each computational basis state. For example, the NOT gate acts on the basis states as follows:

$$|0\rangle \mapsto |1\rangle, \quad |1\rangle \mapsto |0\rangle.$$

This means that the NOT gate, as one would expect, clearly corresponds to the classical NOT gate. The Hadamard gate is a bit more complicated as it maps the two basis states into their uniform superposition, that is:

$$|0\rangle \mapsto \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad |1\rangle \mapsto \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Because these quantum states are used extensively in the field, the states $\frac{|0\rangle + |1\rangle}{\sqrt{2}}$ and $\frac{|0\rangle - |1\rangle}{\sqrt{2}}$ are usually denoted as $|+\rangle$ and $|-\rangle$, respectively. Furthermore, the $|+\rangle$ states correspond to the unit vector on the x-axis of the Bloch sphere and the $|-\rangle$ state to its negative.

Since the computational basis states are represented as vectors, we can conveniently represent operators as matrices. For instance, the above examples correspond to the following matrices:

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \text{and} \quad \mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

We can verify, for example, that the NOT gate indeed acts on the basis states as intended, since

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

There is a particular set of gates, called the *Pauli gates*, which is used in many elementary quantum algorithms. These are the \mathbf{X} , \mathbf{Y} , and \mathbf{Z} gates that correspond to rotations around the x-, y-, and z-axis of the Bloch sphere by π radians, respectively. These gates correspond to the *Pauli matrices* σ_x , σ_y , and σ_z . Another important operation is the identity gate (\mathbf{I}) that corresponds to the identity matrix. Applying an identity gate to a state does not change it, so $\mathbf{I}|\psi\rangle = |\psi\rangle$. A notable feature of the Pauli gates is that consecutive the application of a Pauli gate two times equals the identity gate. The definition of the matrix that the \mathbf{X} gate corresponds to can be found above. The matrix form of the rest of the Pauli gates and the identity gate are as follows:

$$\mathbf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad \mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

We can not only perform rotation around the axes by π radians but also generalise such operations for arbitrary angles. Rotation gates that are essential regarding this thesis are the α rotations around the z- and x-axis, which are,

respectively:

$$\mathbf{R}_Z(\alpha) = \exp\left(-i\frac{\alpha}{2}\mathbf{Z}\right) = \begin{bmatrix} e^{-i\frac{\alpha}{2}} & 0 \\ 0 & e^{i\frac{\alpha}{2}} \end{bmatrix} = e^{i\frac{\alpha}{2}} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

$$\mathbf{R}_X(\alpha) = \exp\left(-i\frac{\alpha}{2}\mathbf{X}\right) = \begin{bmatrix} \cos\left(\frac{\alpha}{2}\right) & -i\sin\left(\frac{\alpha}{2}\right) \\ -i\sin\left(\frac{\alpha}{2}\right) & \cos\left(\frac{\alpha}{2}\right) \end{bmatrix} = e^{-i\frac{\alpha}{2}} \begin{bmatrix} 1 + e^{i\alpha} & 1 - e^{i\alpha} \\ 1 - e^{i\alpha} & e^{i\alpha} \end{bmatrix}$$

Single qubit gates that are also important besides the ones already mentioned are the \mathbf{S} and $\sqrt{\mathbf{X}}$ gates. The rotation gates and matrices corresponding to the above-mentioned gates are as follows:

$$\mathbf{S} = \mathbf{R}_Z\left(\frac{\pi}{2}\right) = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad \sqrt{\mathbf{X}} = \mathbf{R}_X\left(\frac{\pi}{2}\right) = \frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\pi/4} & e^{-i\pi/4} \\ e^{-i\pi/4} & e^{i\pi/4} \end{bmatrix}$$

A useful rule related to single-qubit operations is the Euler decomposition. This rule allows us to express any single-qubit gate as a z-rotation followed by an x-rotation, and once again a z-rotation.

Further to single qubits gates, there are also operators acting on multiple qubits. For example, there is a gate called the CNOT gate (\mathbf{CX}) that applies a NOT gate to the second qubit if the first, control qubit is in the state $|1\rangle$. Therefore, we can describe the CNOT gate as:

$$|00\rangle \mapsto |00\rangle \quad |01\rangle \mapsto |01\rangle \quad |10\rangle \mapsto |11\rangle \quad |11\rangle \mapsto |10\rangle,$$

or more conveniently

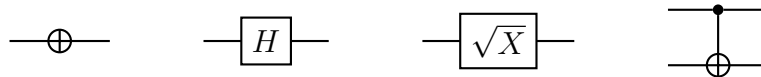
$$|x, y\rangle \mapsto |x, x \oplus y\rangle.$$

Note that the \oplus is the exclusive or operator. The matrix corresponding to the CNOT operator is

$$\mathbf{CX} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The circuit model

A quantum circuit is a widely used model for quantum computation, similar to classical circuits, in which computation is a sequence of quantum gates, measurements, and initializations of qubits to known values. It is a graphical model that depicts qubits as wires and gates as boxes on the qubits. Most of the operators are represented as boxes with their names written into the box. However, there are some exceptions like the NOT gate which is represented the same way the classical NOT gate is. We can also represent controls as black dots on a wire connected to the controlled gate. That is, the NOT, the Hadamard, the \sqrt{X} , and the CNOT gates are, represented respectively as follows:

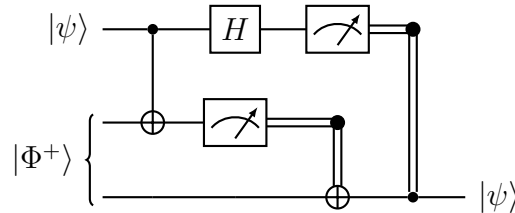


When drawing a full quantum circuit, we have to specify the states of each qubit. We do so by adding the state to the front of the wire. Furthermore, we use a somewhat non-conventional notation: A multi-qubit state can be prepared using braces on each qubit in the given state.

We can also represent a measurement as a box with an analogue measurement display drawn inside it. The outcome of a measurement is stored in bits that we represent in the model as a double, sometimes called classical wire. A classical wire originates from a measurement box and it can be connected to a quantum operation. Such connection translates to a classically controlled quantum operation. Circuits that use classically controlled quantum gates are called feed-forward quantum circuits.

Quantum teleportation Using the model, we can conveniently describe how, for example, the quantum teleportation protocol works. Quantum teleportation is an algorithm for transferring quantum information from one place to another. While one may think about teleportation as a way to transfer an object between two locations, quantum teleportation can only transfer quantum information.

Moreover, to transfer this quantum information, one needs to send classical information to the recipient; therefore, the transfer can not occur faster than the speed of light. We can describe quantum teleportation using the quantum circuit model as follows:



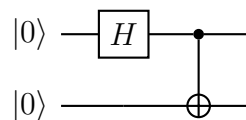
In the above circuit, we begin with an arbitrary quantum state $|\psi\rangle$ on the first qubit; the second and third qubits are in the Bell-state. We apply the given operations as depicted above, and thus the quantum state $|\psi\rangle$ is transferred to the last qubit.

Entanglement

In addition to superposition, another non-classical feature of quantum systems is *entanglement*. Quantum entanglement is a phenomenon that occurs when multiple qubits interact in such a way that the resulting quantum state of each qubit can not be described independently of the others; that is, the qubits do not act as individuals but as an inseparable whole. Therefore, quantum systems can express higher correlation than it is classically possible. The simplest state that demonstrates quantum correlation is called the *Bell state*, that is:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}} =: |\Phi^+\rangle$$

and can be constructed as follows:



The Bell state is in an equal superposition of $|00\rangle$ and $|11\rangle$ meaning that measuring the state yields $|00\rangle$ or $|11\rangle$ with 50% probability since $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$. The

above statement means that if we prepare two qubits in the Bell state, separate them, and measure one of them, we will know with certainty what outcome the other qubit would yield upon measuring. The above result may not seem surprising as a classical system could achieve the same behaviour by setting the bits to the same value when interacting. The difference is that classical systems have definite values for all observables, while quantum systems do not. In a sense, quantum systems acquire a probability distribution for the outcomes of a measurement, but this distribution changes or ‘collapses’ when measuring any qubit that is part of the system. This effect occurs instantaneously such that no information about the measurement outcome can be communicated to the other qubit, assuming that information cannot travel faster than light.

2.1.2 Clifford gates

Recall that the \mathbf{X} , \mathbf{Y} , and \mathbf{Z} gates are called the Pauli gates that correspond to rotations of π radians around the x-, y-, and z-axis, respectively. The *Pauli group* on 1 qubit, \mathcal{P}_1 , is the matrix group consisting of the Identity matrix, the Pauli matrices and the products of these matrices with the factors ± 1 and $\pm i$:

$$\mathcal{P}_1 = \{\pm \mathbf{I}, \pm \mathbf{X}, \pm \mathbf{Y}, \pm \mathbf{Z}, \pm i \mathbf{I}, \pm i \mathbf{X}, \pm i \mathbf{Y}, \pm i \mathbf{Z}\}.$$

The Pauli group on n qubits, \mathcal{P}_n , is the group generated by the elements of \mathcal{P}_1 applied to each of n qubits, or simply $\mathcal{P}_n = \mathcal{P}_1^{\otimes n}$.

The group of automorphisms of the Pauli group is called the *Clifford group* and is denoted as:

$$\mathcal{C}_n = \{U \mid UPU^\dagger = P, \forall P \in \mathcal{P}_n\}.$$

The elements of the Clifford group are known as *Clifford gates*. Alternatively, the Clifford gates are those gates generated by compositions of the CNOT, the Hadamard, and the \mathbf{S} gates. The states we can produce by applying Clifford

gates to the initial state $|00\dots 0\rangle$ are called *Clifford states*.

A differentiating property of Clifford states is that they can be efficiently classically simulated. This result is the content of the Gottesman–Knill theorem [6]. One implication of this theorem is that Clifford states and gates do not allow universal quantum computation (unless $\text{BQP} = \text{BPP}$) as that would indicate that we can efficiently simulate arbitrary quantum systems.

Even though Clifford computation does not give us the computational advantage we may desire, it is still widely used in numerous tasks. For example, we use it in error correcting codes, measurement-based quantum computing, and several protocols such as quantum key distribution, quantum teleportation, or superdense coding.

2.1.3 Quantum computation using qudits

Qudits are a d -level alternative to the unit of quantum information to the conventional 2-level qubits. This means that we have $|0\rangle, |1\rangle, |2\rangle, \dots, |d-1\rangle$ as our computational basis states instead of just $|0\rangle$ and $|1\rangle$. Consequently, qudits provide a larger state space to store and process information. They can also provide a reduction of the circuit complexity and enhancement of the algorithm efficiency.

To visualise a single-qudit, instead of the Bloch sphere, we use a torus on which the basis states are the d roots of unity in a slice. Furthermore, we have states similar to $|+\rangle$ around the torus. An example can be seen in Figure 2.2.

Most theory of quantum computation has so far focused on qubits; however, in recent years there has been a recent surge of interest in studying quantum computation using d -dimensional systems, called *qudits*. For example, the qudit version of several quantum algorithms has been presented in Ref. [10]. We can not only implement the same quantum algorithms using a higher dimensional system but also enhance some, that were originally designed for qubits. For instance, we

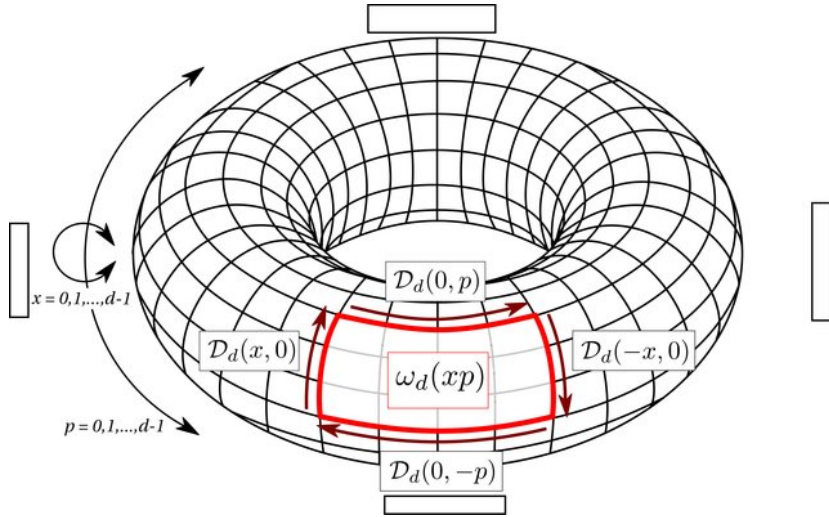


Figure 2.2: Some of the qudit stabilizer states on the unbiased torus.

can enhance fault-tolerant quantum computing using qudits [11]. Also, there is a benefit to quantum communication using higher dimensional systems [12].

While qudit-based quantum computation sounds promising, it would fail to offer an alternative to qubits unless we are able to design and build such computers. In recent years, qudit-based quantum computation has been realised using different paradigms of quantum computation, such as ion traps [13, 14], photonic devices [15], and superconducting devices [16, 17, 18, 19].

2.2 The ZX-calculus

This section introduces the ZX-calculus from scratch; however, an extensive book, *Picturing Quantum Processes* by Coecke and Kissinger, with many examples and exercises can be a good introduction to the topic [40]. Alternatively, a less lengthy review paper, *ZX-calculus for the working quantum computer scientist* by van de Wetering, can be another good source of information regarding the basics of ZX-calculus and a bit further [1].

2.2.1 ZX-calculus basic concepts

The *ZX-calculus* is a graphical language for expressing and reasoning about quantum computation. It was first introduced by Bob Coecke and Ross Duncan as an extension of categorical quantum mechanics [20]. The ZX-calculus consists of ZX-diagrams and a set of graphical rewrite rules that enables us to transform these diagrams without changing the underlying linear map. At its core, ZX-diagrams are built up from spiders that represent specific tensors or linear maps. We can connect these spiders, resulting in a network of spiders that represents a tensor network. One particularly nice feature of ZX-diagrams is that topological deformation does not change the linear map it represents. This means that we can freely move around the spiders in a diagram and the meaning of it remains the same.

Category theory background

The ZX-calculus is often discussed as a particular type of category. A *category* is an abstract mathematical structure consisting of *objects* and *morphisms* such that they satisfy some properties. Generally speaking, a morphism is what goes between objects in a category. Given a morphism $f : A \rightarrow B$ from object A to B and $g : B \rightarrow C$ from object B to C, we can compose these morphisms to get a morphism $g \circ f : A \rightarrow C$. Composition is required to be associative, that is $h \circ (g \circ f) = (h \circ g) \circ f$. We also have something called the *identity morphism* for each object B, $\text{id}_B : B \rightarrow B$, that act as the identity for the composition: $\text{id}_B \circ f = f$. The above two rules are the requirements for a category to be one. As presented in the subsequent sections, we can compose ZX-diagrams, but we can compose them not only sequentially but also in parallel using the tensor product. Those categories that are equipped with some notion of ‘tensor product’ of its objects are called *monoidal categories*. Furthermore, if $A \otimes B$ is also isomorphic to $B \otimes A$, then it is a *symmetric monoidal category*.

scalar $e^{i0} = 1$ and so the structure corresponds to something like a copy: The spider either corresponds to $|0\rangle^{\otimes n} \langle 0|^{\otimes m}$ or $|1\rangle^{\otimes n} \langle 1|^{\otimes m}$ which can be interpreted as inputting and outputting the same computational basis state on each wire. As such spiders are used rather often, we define an empty spider to be a spider with phase 0:

$$\begin{array}{c} m \vdots \\ \diagup \quad \diagdown \\ \textcircled{0} \\ \diagdown \quad \diagup \\ n \vdots \end{array} := \begin{array}{c} m \vdots \\ \diagup \quad \diagdown \\ \textcircled{0} \\ \diagdown \quad \diagup \\ n \vdots \end{array}$$

We call these diagrams ‘spider’ simply because of the way they look. Moreover, the Z in the name corresponds to the eigenbasis of the Z gate, $|0\rangle$ and $|1\rangle$, that we use to define the spider. Similarly, we can define the *X-spider* that is defined with respect to the eigenbasis of the X gate. Its interpretation is similar to the previous spider and is defined as follows:

$$\left[\begin{array}{c} m \vdots \\ \diagup \quad \diagdown \\ \textcircled{\alpha} \\ \diagdown \quad \diagup \\ n \vdots \end{array} \right] = |+\rangle^{\otimes n} \langle +|^{\otimes m} + e^{i\alpha} |-\rangle^{\otimes n} \langle -|^{\otimes m}$$

We define the phases and an empty X-spider similarly to the Z-spider

Subsequently, we examine some specific spiders that play an elemental role in the ZX-calculus. First of all, the 1-input and 1-output Z-spider describes the $\mathbf{R}_Z(\alpha)$ gate up to the phase $e^{-i\frac{\alpha}{2}}$ since:

$$\left[\begin{array}{c} \vdots \\ \diagup \quad \diagdown \\ \textcircled{\alpha} \\ \diagdown \quad \diagup \\ \vdots \end{array} \right] = |0\rangle \langle 0| + e^{i\alpha} |1\rangle \langle 1| = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{bmatrix} = e^{-i\frac{\alpha}{2}} \mathbf{R}_Z(\alpha)$$

And similarly, the 1-input 1-output X-spider corresponds to the $\mathbf{R}_X(\alpha)$ gate up to the phase $e^{i\frac{\alpha}{2}}$ since:

$$\left[\begin{array}{c} \vdots \\ \diagup \quad \diagdown \\ \textcircled{\alpha} \\ \diagdown \quad \diagup \\ \vdots \end{array} \right] = |+\rangle \langle +| + e^{i\alpha} |-\rangle \langle -| = \frac{1}{2} \begin{bmatrix} 1 + e^{i\alpha} & 1 - e^{i\alpha} \\ 1 - e^{i\alpha} & e^{i\alpha} \end{bmatrix} = e^{i\frac{\alpha}{2}} \mathbf{R}_X(\alpha)$$

It is also worth mentioning that if $\alpha = 0$ in the above examples, they correspond

to the identity since $\mathbf{R}_{\mathbf{Z}}(0) = \mathbf{I} = \mathbf{R}_{\mathbf{X}}(0)$, and so:

$$\text{---}\circ\text{---} = \text{---} = \text{---}\bullet\text{---}$$

This is sometimes called the *identity rule* in the ZX-calculus.

Another interesting spider type is a state that has 0-inputs and at least 1 output. Some examples are the following spiders that correspond to the computational basis states, up to the scalar $\sqrt{2}$, since:

$$\begin{aligned} \bullet\text{---} &= |+\rangle + |-\rangle = \sqrt{2}|0\rangle & \circ\text{---} &= |0\rangle + |1\rangle = \sqrt{2}|+\rangle \\ \pi\bullet\text{---} &= |+\rangle - |-\rangle = \sqrt{2}|1\rangle & \pi\circ\text{---} &= |0\rangle - |1\rangle = \sqrt{2}|-\rangle \end{aligned}$$

Note that, even though it may seem counter-intuitive, the X-spiders correspond to the basis states of the Z gate and vice versa for the Z-spiders. Also, note that we get the states with the wrong scalar factors. However, just like with quantum circuits, we can usually ignore global non-zero scalars in ZX-diagrams. A more thorough discussion can be found on scalars in ZX-diagrams in Section 2.2.1. Lastly, it is worth mentioning that throughout the literature one can find other naming conventions as well. Z-spiders are often called green-spiders, and X-spiders can be referenced as red-spiders.

Composition

As it is mentioned above, we can compose spiders that can result in a structure similar to a tensor network. There are two kinds of compositions of ZX-diagrams, parallel and sequential composition. *Parallel composition* corresponds to the tensor product of the underlying linear maps. To parallelly compose two ZX-

diagrams we simply place one diagram on top of the other:

$$\left(\text{---} \circlearrowleft \right) \otimes \left(\text{---} \right) = \text{---} \circlearrowleft$$

The other kind of composition, *sequential composition*, corresponds to the multiplication of the underlying matrices and is denoted with the \circ operator. To sequentially compose two ZX-diagrams we connect the output wires of the first part of the composition with the input wires of the second part. For instance, we can construct the CNOT gate from spiders if we compose the following diagrams:

$$\left[\left(\text{---} \right) \otimes \left(\text{---} \circlearrowright \right) \right] \circ \left[\left(\text{---} \circlearrowleft \right) \otimes \left(\text{---} \right) \right] = \left[\text{---} \circlearrowright \right] \circ \left[\text{---} \circlearrowleft \right]$$

$$= \text{---} \circlearrowleft \circlearrowright$$

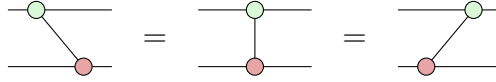
While the above diagram may not resemble the very same structure of the CNOT gate, we introduce a rule subsequently that can help us transform the diagram to more clearly depict it.

Only connectivity matters

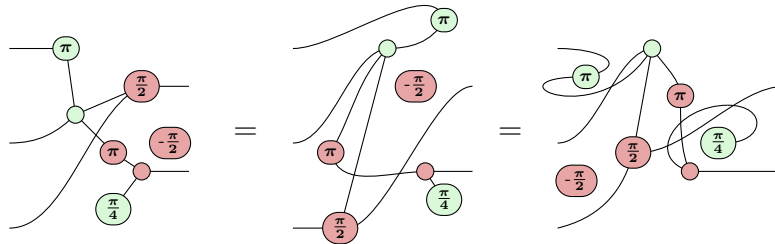
One particularly useful rule in the ZX-calculus is the *Only Connectivity Matters* (*OCM*) rule. This rule allows us to move spiders in a ZX-diagram while preserving interpretation of it. As long as the input and output wires stay in the same place, we can freely rearrange any ZX-diagrams. For example, we can bend or straighten wires as we wish:

$$\text{---} \circlearrowleft \text{---} \circlearrowright = \text{---} \circlearrowleft \text{---} \circlearrowright = \text{---} \circlearrowleft \circlearrowright$$

We can also change the location of spiders in a ZX-diagram; therefore, each of the following diagrams corresponds to the CNOT gate:



We can also do something more radical, for example:

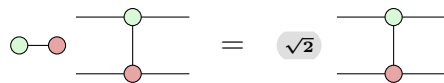


Scalars

Scalars are ZX-diagrams with 0-inputs and 0-outputs. They have a somewhat distinguished role in the calculus as they represent complex numbers. For example, the interpretation of some scalars are:

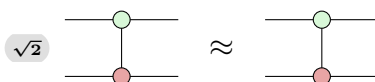
$$\begin{aligned}
 \llbracket \circ \rrbracket &= 2 & \llbracket \alpha - \circ \rrbracket &= \sqrt{2} \\
 \llbracket \pi \rrbracket &= 0 & \llbracket \alpha - \pi \rrbracket &= e^{i\alpha} \\
 \llbracket \alpha \rrbracket &= 1 + e^{i\alpha} & \llbracket \circ - \pi \rrbracket &= \frac{1}{\sqrt{2}}
 \end{aligned} \tag{2.1}$$

Note that we can represent any complex number using only the scalars presented in Eq. (2.1) and their combinations. We can write scalars just as a number in a grey circle that has no legs next to the diagram instead of representing the scalar as a ZX-diagram:



Throughout the literature, it is quite usual to drop the scalar factors from the equations in ZX-diagrams. This is for the same reason why some physicists

calculate with unnormalized values, it is inconvenient and unnecessary to keep track of all the scalars. We annotate this by using the \approx operator, which means that the two diagrams equal up to some non-zero scalar, for instance:



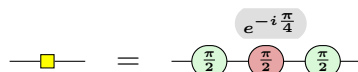
Note that it is only possible to ignore non-zero scalars. One example where scalars can conveniently be dropped is when dealing with ZX-diagrams representing some quantum circuits. This is because for a matrix M representing a ZX-diagram, $MM^\dagger = \lambda\mathbf{I}$ where $\lambda > 0$. Therefore, we can re-normalise M by multiplying it with $\frac{1}{\lambda}$. We can calculate the value λ by composing a diagram with its adjoint and simplifying it until it reduces to the identity. However, we cannot ignore scalars when calculating probability amplitudes, for instance when calculating the probabilities of some particular input and output.

The Hadamard gate

The Hadamard gate has a somewhat distinguished role in quantum computation as it maps Z rotations to X rotations and vice versa. We have a different notation for the Hadamard gate in the ZX-calculus:

$$\llbracket \text{---} \square \text{---} \rrbracket = \mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

This ‘Hadamard box’ can be derived in terms of Z- and X-spiders. To do so, let us first recall the Euler decomposition that allows us to express any single qubit rotation as a rotation around the z-, x-, and once again z-axis of the Bloch sphere. In particular, the Hadamard gate can be decomposed into the following sequence of rotations:



It is also possible to show that we can express the decomposition without scalars which might be desirable on a formal level:

$$\text{---} \square \text{---} = \text{---} \left(\overset{\pi/2}{\circ} \right) \left(\underset{-\pi/2}{\circ} \right) \left(\overset{\pi/2}{\circ} \right) \text{---}$$

Also, note that applying the Hadamard gate two times equals the identity:

$$\text{---} \square \square \text{---} = \text{---} \tag{2.2}$$

Lastly, the property that makes the Hadamard box a particularly useful element of the ZX-calculus is that it maps Z-spiders into X-spiders and vice versa:

$$\begin{array}{c} \square \\ \vdots \\ \square \end{array} \left(\overset{\alpha}{\circ} \right) \begin{array}{c} \square \\ \vdots \\ \square \end{array} = \begin{array}{c} \text{---} \\ \vdots \\ \text{---} \end{array} \left(\underset{\alpha}{\circ} \right) \begin{array}{c} \text{---} \\ \vdots \\ \text{---} \end{array} \quad \begin{array}{c} \square \\ \vdots \\ \square \end{array} \left(\underset{\alpha}{\circ} \right) \begin{array}{c} \square \\ \vdots \\ \square \end{array} = \begin{array}{c} \text{---} \\ \vdots \\ \text{---} \end{array} \left(\overset{\alpha}{\circ} \right) \begin{array}{c} \text{---} \\ \vdots \\ \text{---} \end{array}$$

This rule is called the *colour* rule.

Axioms

As it is mentioned above, the ZX-calculus consists of ZX-diagrams and also a set of graphical rewrite rules. Subsequently, we present a convenient set of graphical rewrite rules as the axioms for the ZX-calculus. Do note that this set is not minimal. We introduce rules to formally deal with scalars as diagrams which is unnecessary for the language, it just makes it simpler. Furthermore, some rules can be derived from the rest. However, our aim is to provide a clear and convenient set of rules in order to make it accessible for most readers.

$$(2.3)$$

The **Z-ELIM**, **X-ELIM**, **EULER**, **COLOUR** rules are mentioned above with some explanation. Further to the above-mentioned rules, the **FUSION** rule allows us to fuse connected Z-spiders. When spiders are fused, their phases are added together. It is possible to show that we can extend this rule to fuse spiders of the same colour if they are connected with one or even more wires. An extended version of the rule would look as follows:

The π rule enables us to copy X-spiders with π phases through a Z-spider with any phase and two outputs. It is also possible to extend this rule to the other colour variant as well with any number of output wires:

$$(2.4)$$

Furthermore, we can extend the Hopf-rule to spiders of the same colour with any phase that are connected with two wires with Hadamard boxes on them. This extension proves to be a useful tool for Clifford computation that is discussed in Section 2.2.2.

Lemma 2. *A modified version of the Hopf-rule, where identically coloured spiders with any phase are connected through Hadamard boxes, is derivable in the ZX-calculus, that is:*

$$\begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array} = \begin{array}{c} \sqrt{2} \\ \alpha \quad \beta \\ \vdots \\ \vdots \end{array} \quad (2.8)$$

Proof.

$$\begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array} \\
 = \begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array} \stackrel{\text{(HOPF)}}{=} \begin{array}{c} \sqrt{2} \\ \alpha \quad \beta \\ \vdots \\ \vdots \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \sqrt{2} \\ \alpha \quad \beta \\ \vdots \\ \vdots \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} \vdots \\ \vdots \end{array} \begin{array}{c} \alpha \\ \beta \end{array} \begin{array}{c} \vdots \\ \vdots \end{array}$$

□

Lastly, the **ZERO** rule allows one to transform X-states to Z-states and vice-versa given that a 0-scalar is present. This has the consequence that a 0-scalar ‘absorbs’ or ‘deletes’ all ZX-diagrams it is composed with.

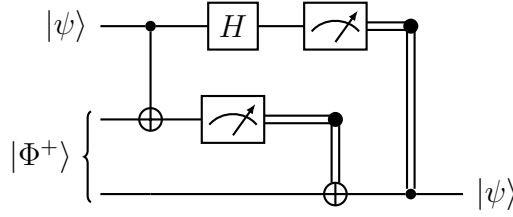
Finally, we also show a useful feature of specific states that we can change their basis. Specifically, we can change the basis of a state with phase $\frac{\pi}{2}$ that follows from the equation below. For some adequate k :

$$\begin{array}{c} \pm \frac{\pi}{2} \\ \vdots \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} \pm \frac{\pi}{2} \\ \vdots \end{array} \begin{array}{c} \square \\ \vdots \end{array} \stackrel{\text{(EULER)}}{\approx} \begin{array}{c} \pm \frac{\pi}{2} \quad -\frac{\pi}{2} \quad -\frac{\pi}{2} \quad -\frac{\pi}{2} \\ \vdots \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} k\pi \quad -\frac{\pi}{2} \quad -\frac{\pi}{2} \\ \vdots \end{array} \quad (2.9) \\
 \stackrel{\text{(COPY)}}{=} \begin{array}{c} k\pi \quad \frac{\pi}{2} \\ \vdots \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \mp \frac{\pi}{2} \\ \vdots \end{array}$$

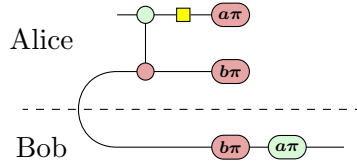
Examples

Using the ZX-calculus, we can prove the correctness of several algorithms. For example, the above-described quantum teleportation is a great example of the

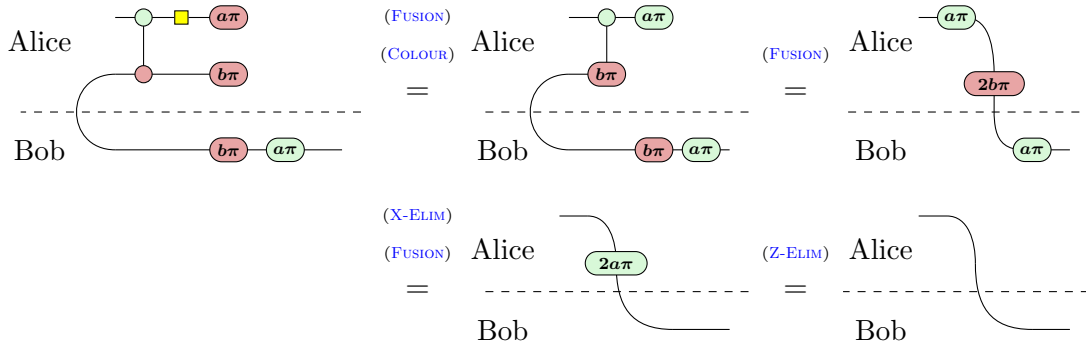
reasoning power of the ZX-calculus. Recall, that using the quantum circuit model the quantum teleportation algorithm is described as follows:



This quantum circuit translates to the following ZX-diagram where we also indicated a space-like separation of Alice and Bob, the two participants in the experiment:



It would be cumbersome to prove the correctness of the algorithm using linear algebra. However, using the axioms of the ZX-calculus presented in Eq. (2.3) we can prove the correctness of the algorithm with ease:



Therefore, it is possible to show that we can reduce the original ZX-diagram to a single wire that represents the identity gate. Since the teleportation equals the identity gate, any state that we input on Alice's side is 'teleported' to Bob's side and so the algorithm is correct.

2.2.2 Clifford computation

Recall that Clifford unitaries are those generated by compositions of the CNOT, the Hadamard, and the **S** gates. Equivalently, the group of automorphisms of

the Pauli group is called the Clifford group and its elements are the Clifford gates. A more throughout discussion is presented in Section 2.1.2. It turns out that Clifford gates are those that can be constructed using such ZX-diagrams that has a phase that is a multiple of $\frac{\pi}{2}$. The *Clifford fragment* of the ZX-calculus contains those diagrams that represent Clifford unitaries. The Clifford fragment seems to be a particularly well-behaved part of the calculus. This subsection examines this fragment in-depth starting with Graph states, through more advanced graphical rewrite rules like local complementation and pivoting, and finally proving completeness.

Graph-like diagrams

There are some tricks that can be used to deal with a more restricted ZX-diagram. For example, we can convert each X-spider into Z-spider using **COLOUR** resulting in only Z-spiders and Hadamard boxes. There are other properties that we can restrict that can help us work with simpler diagrams. A diagram type we use for many algorithms is called a *graph-like diagram* which is a ZX-diagram where:

1. All spiders are Z-spiders;
2. Spiders are only connected via Hadamard edges;
3. There are no parallel Hadamard edges or self-loops;
4. Every input or output is connected to a Z-spider.

We can convert any ZX-diagram into one that is graph-like using the following algorithm:

1. Convert each X-spider into a Z-spider using the **COLOUR** rule;
2. Cancel adjacent Hadamard boxes using Eq. (2.2);
3. Fuse all connected spiders using **FUSION**;

4. Remove all self-loops using Eq. (2.5) and Eq. (2.6);
5. If two spiders are connected by multiple Hadamard edges, remove them using the modified version of the Hopf-rule shown in Eq. (2.8).

In order to make the description of graph-like diagrams easier, we introduce some names for specific spiders in such a diagram. We call spiders that are not connected to an input or an output in a graph-like diagram an *internal spider*. Furthermore, spiders connected to an input or an output are called *boundary spiders*. Note that a spider in a graph-like diagram is either an internal or a boundary spider.

Graph states

A particularly useful subset of graph-like diagrams are the *graph states*. For a simple undirected graph $G = (V, E)$ we define its corresponding graph state $|G\rangle$ as:

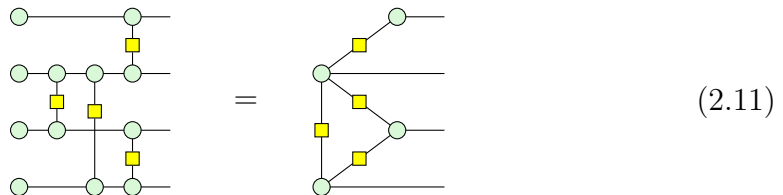
$$|G\rangle = \prod_{(u,v) \in E} \mathbf{CZ}_{u,v} \bigotimes_{u \in V} |+\rangle_u$$

In other words, we prepare each state in $|+\rangle$. Then, for each edge of G we apply a **CZ** gate which entangles the two qubits. Note that since the **CZ** gate commutes, it can be applied in any order.

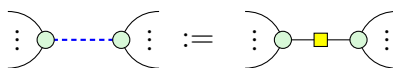
For example, given the following simple graph:



The graph state that corresponds to the graph of Eq. (2.10) can be constructed as follows:



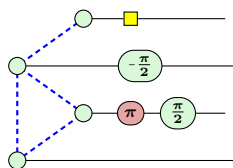
For more dense graphs, writing the Hadamard boxes for each edge can become cumbersome to manage. To overcome this issue, we use a new notation which we call a *Hadamard-edge*. This new type of edge corresponds to an edge with a Hadamard box and is drawn as a blue, dashed line:



Note that it is also possible to define graph-states as a restricted set of graph-like diagrams. A graph-like diagram is a graph state when:

- There are no inputs;
- Each spider is connected to a single output;
- All phases are zero.

We can extend graph states by allowing any Clifford unitaries to be applied to each qubit after the graph state has been initialised. Such a state is called *Graph State with Local Cliffords* (GSLC). It turns out that any Clifford state equals one in GSLC form. We could produce the following example of a GSLC by adding some local Cliffords to the graph state of Eq. (2.11):

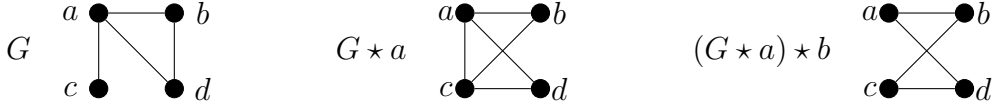


Local complementation

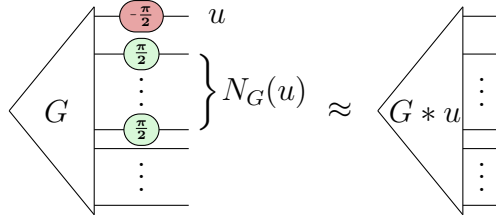
As it is mentioned above, any Clifford state can be represented as a Graph state with local Cliffords. However, this GSLC form is not unique since we can absorb some local Cliffords by changing the graph. This can be done by applying *local complementation* to the graph.

Let $G = (E, V)$ be a simple graph and $u \in V$ a vertex of G . The local

complementation of G about u , written as $G \star u$, is a graph that has the vertices and edges of G but the edges between the neighbours of u are changed. For instance, some examples of a graph and its local complementation could be as follows:



Local complementation is derivable in the ZX-calculus as follows:

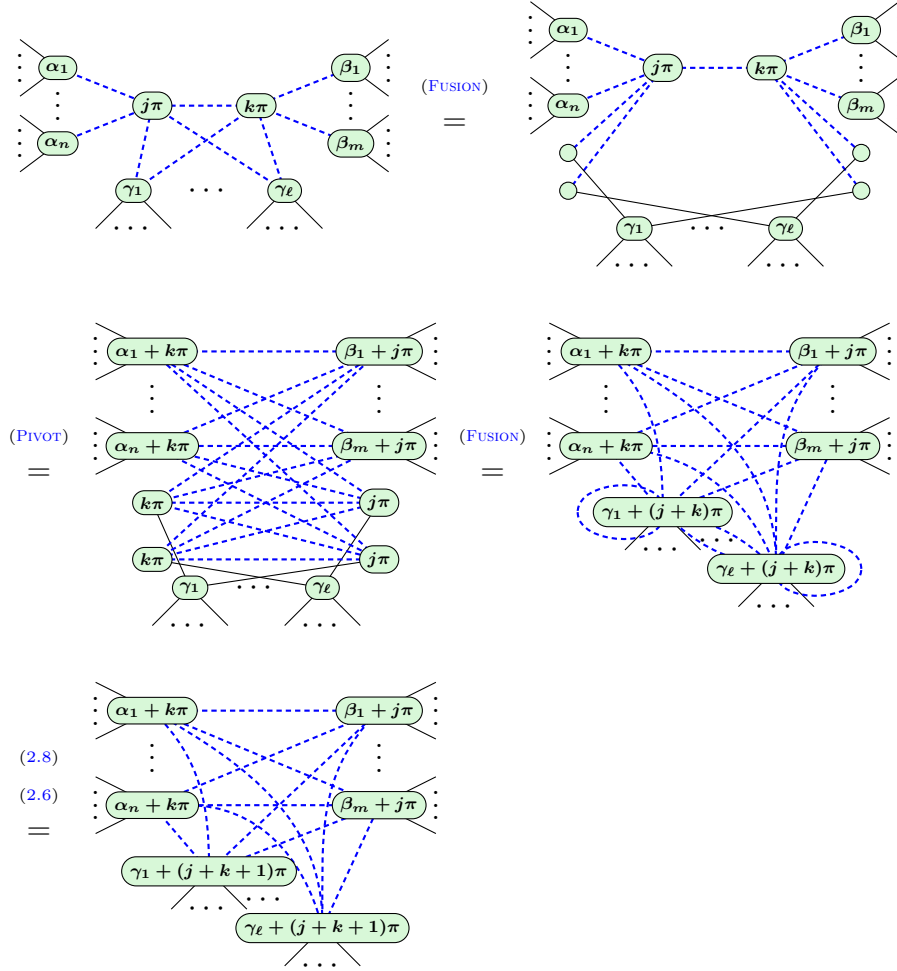


where the triangle on the left represents a graph state corresponding to the graph G and $N_G(u)$ denotes the neighbours of u in G . This equation was originally proved in the ZX-calculus in Ref. [42]. However, a more accessible proof is provided in Ref. [40].

Using this local complementation rewrite rule it is possible to prove the following rule:

We call the simplification presented above *local complementation*, and it removes an internal spider with phase $\pm\frac{\pi}{2}$ by adding $\mp\frac{\pi}{2}$ phases to its neighbours and connecting them. A proof for the above equation can be found in Ref. [28]. Note that if there are two wires between spiders we can remove them using the Hopf-rule. This means that if some neighbouring spiders are connected before local complementation, then the wires get removed. We can interpret this as local

also note that there might be spiders connected to both internal spiders.



A proof of the extended pivot rule can be found in Ref. [28] which naturally implies the correctness of Eq. (2.13).

AP-form

It is clear that by using local complementation and pivoting we can remove a lot of spiders from a Clifford diagram. In fact, we can remove so many spiders that we can reduce any Clifford diagram into a normal form. We say that a graph-like diagram is in *Affine with Phases form* (AP-form) when:

- There are no inputs;
- The internal spiders have only 0 or π phases;

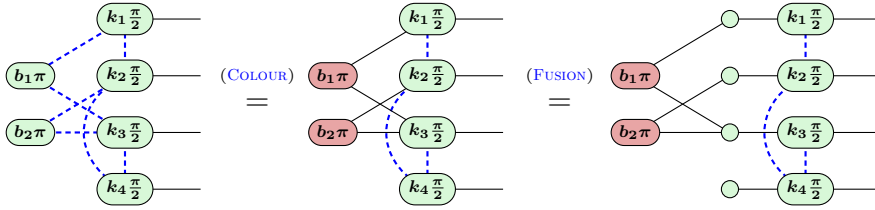
- Internal spiders are only connected to boundary spiders.

It is clear that by applying local complementation we eliminate each inner spider with phase $\pm\frac{\pi}{2}$. Furthermore, any pair of connected inner spiders with phase π or 0 are also removed because of the applications of pivots. Therefore, by applying local complementation and pivoting as much as possible we can indeed transform a Clifford diagram into one in AP-form.

An example of a ZX-diagram in AP-form could be:



We transform the above diagram a bit in order to it easier to make it easier work with:



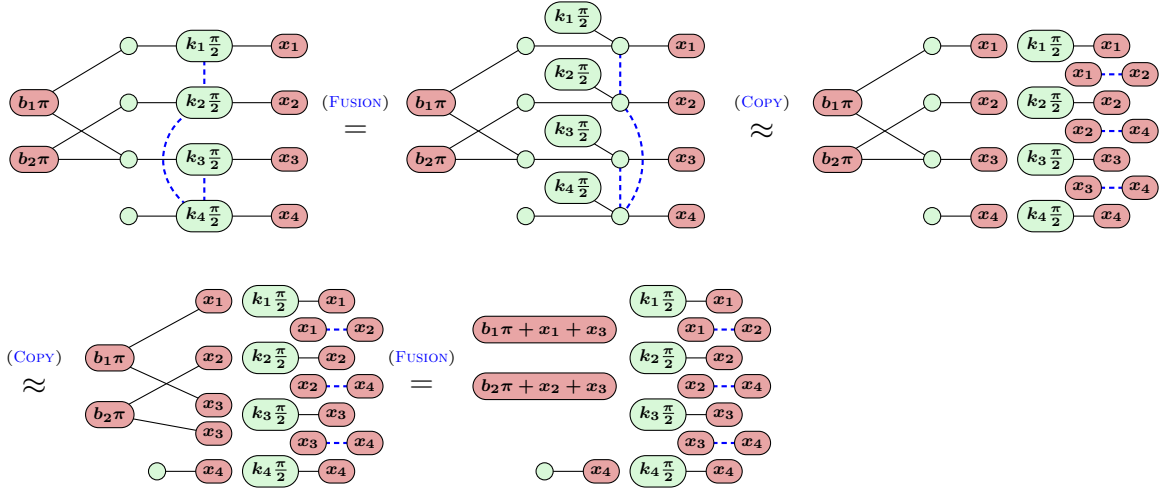
We claim that the above diagram in AP-form, which represents the state $|\psi\rangle$, equals the following state up to some global phase:

$$|\psi\rangle \approx \frac{1}{2^N} \sum_{\substack{\vec{x} \in \mathbb{F}_2^4 \\ A\vec{x} = \vec{b}}} e^{i\pi\phi(\vec{x})} |\vec{x}\rangle \quad (2.15)$$

where $\vec{b} = (b_1, b_2)^T$, A is the parity matrix describing the connectivity of the inner and boundary spiders. Furthermore, ϕ is a real-valued phase function that describes the connectivity and phases of the boundary spiders.

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \quad \phi(\vec{x}) = \frac{k_1}{2} \vec{x}_1 + \vec{x}_1 \vec{x}_2 + \frac{k_2}{2} \vec{x}_2 + \vec{x}_2 \vec{x}_4 + \frac{k_3}{2} \vec{x}_3 + \vec{x}_3 \vec{x}_4 + \frac{k_4}{2} \vec{x}_4$$

This follows from the following transformations:



Note that if a single X-spider has phase π , then it equals the zero scalar which means that it is an impossible scenario to happen. Therefore, the above diagram allows only those vectors, \vec{x} , that satisfy $A\vec{x} = \vec{b}$. Furthermore, the scalars that are copied from the phases part of the diagram equal the $e^{i\pi\phi(\vec{x})}$ component of the equation. We conclude that a diagram in AP-form in Eq. (2.14) indeed equals the equation presented in Eq. (2.15).

Further to diagrams in AP-form, we also need something more restricted. A diagram in AP-form defined by A , \vec{b} , and ϕ is in *reduced AP-form* if it is 0 or it is non-zero and:

- A is in reduced row echelon form (RREF) with no zero rows;
- ϕ only contains free variables from the equation system $A\vec{x} = \vec{b}$;
- All coefficients of ϕ are in the interval $(-1, 1]$.

We work with diagrams in reduced AP-form subsequently and therefore we need to refresh some definitions from linear algebra. Recall that the first non-zero element in a row of A is called a *pivot*. We call the spider that corresponds to the pivot element in the matrix a *pivot spider*. Furthermore, we call the variable x_i a *free variable* if the i -th column of A does not contain a pivot, otherwise we

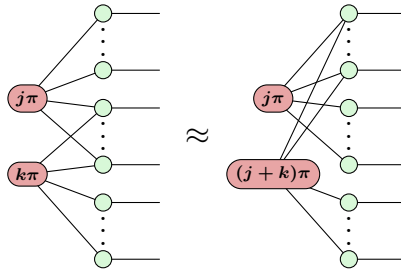
call it a *bound variable*.

2.2.3 Clifford completeness

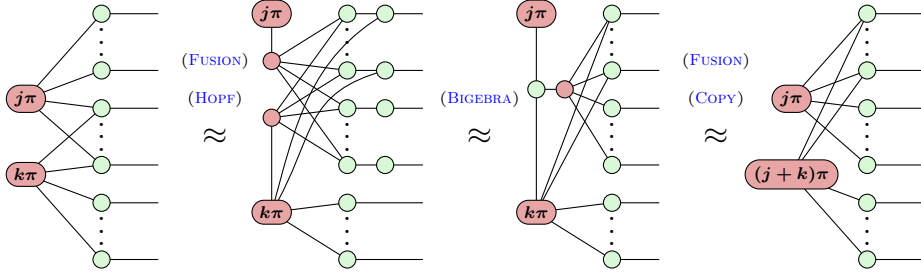
In this section, we examine an issue of many early ZX-calculus papers, completeness. A graphical calculus is complete if its rewrite rules can prove any true equation. More formally, graphical calculus is complete if for any diagram $A, B \in \text{ZX}$ such that $\llbracket A \rrbracket = \llbracket B \rrbracket$, we can provide a sequence of rewrites that transforms A into B . Subsequently, we prove the completeness proof of the Clifford fragment of the ZX-calculus.

We claim that any Clifford diagram equals one in reduced AP-form, and this form is unique. First of all, it is clear that any Clifford diagram can be converted into one in AP-form using local complementation and pivoting. Then, we have to show that the biadjacency matrix A , which corresponds to the connections of the inner and boundary spider of the diagram in AP-form, can be transformed into one in RREF. We note that the X-spiders of a ZX-diagram in AP-form correspond to the rows in the linear system $A\vec{x} = \vec{b}$. If we can show that we can perform primitive row operations in a ZX-diagram in AP-form, then we can simply transform it into one with a biadjacency matrix in RREF using Gaussian elimination.

Lemma 3. *We can perform primitive row operations on a ZX-diagram in AP-form, i.e. we can ‘add’ one inner spider to another, for $j, k \in \{0, 1\}$:*



Proof.



□

Therefore, it is possible to translate a Clifford ZX-diagram into one in AP-form with a biadjacency matrix in RREF.

Subsequently, we prove that we can remove any phase or Hadamard edge connected to a pivot spider in AP-form with biadjacency matrix in REF. To show the above statement, we consider several cases. Firstly, we show that it is possible to remove a 0 or π phase from a pivot spider as follows:

(2.16)

Secondly, we have to show that we can remove $\pm\frac{\pi}{2}$ phases from pivot spiders. Note that we can push a $\pm\frac{\pi}{2}$ Z-spider through an X-spider with no phase, resulting in a fully connected graph with $\pm\frac{\pi}{2}$ Z-spiders on the outputs:

(2.17)

Then, we can push a pivot spider with phase $\pm\frac{\pi}{2}$ through the inner spider to

which it is connected as follows:

(2.18)

Lastly, we have to show that any Hadamard edge connected to a pivot spider can be removed. Firstly, we can remove any Hadamard edge in case the inner spider is connected to the spider the pivot spider is connected through the Hadamard edge as follows:

(2.19)

(2.6)

We can also remove a Hadamard edge that is connected to a boundary spider

that is not connected to the same inner spider as the pivot element as follows:

(2.20)

Therefore, we can remove any phase or Hadamard-edge that is connected to the pivot spider.

In conclusion, we can convert any Clifford diagram into one in AP-form using local complementation and pivoting. Then, such a diagram can be translated into one in AP-form with biadjacency matrix A in RREF using Gaussian elimination which is possible due to the lemma 3. Lastly, we can remove any phase using Eq. (2.16) and Eq. (2.18) or Hadamard-edge using Eq. (2.19) and Eq. (2.20). This enables us to transform a diagram in such a way that its phase function ϕ only contains free variables from the equation system $A\vec{x} = \vec{b}$. Such diagrams are in reduced AP-form and this form is unique. Ultimately, this enables us to prove the completeness of the Clifford fragment of the ZX-calculus.

Theorem 4. *For any pair of ZX-diagrams $A, B \in \text{ZX}$, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, then we can provide a sequence of rewrites that transforms A into B .*

Proof. We only show it for state by map-state duality. If A and B represent the same linear map, i.e. $\llbracket A \rrbracket = \llbracket B \rrbracket$, then their reduced AP-form is identical thanks to its uniqueness. Therefore, we can transform both A and B into diagrams in reduced AP-form. The sequence of transformations from A to A in reduced AP-form composed with the series of rewrites from B in reduced AP-form to B provides us with a sequence of rewrites that transforms A into B \square

Chapter 3

Qupit Clifford ZX-calculus

In this chapter, we first present a family of ZX-calculi, one for each odd prime dimension. Then, we derive some essential rewrite rules that are used in practice. We also define some additional syntactic sugar to represent multi-edges and multi-Hadamard-edges. This is followed by proving the completeness of the scalar fragment of the calculus using the defined rules. Then, we define the qudit version of graph states. Lastly, we present the proof of local complementation and pivoting for odd prime dimensional qudits.

Subsequently, p denotes an arbitrary odd prime, and $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ the ring of integers modulo p . We also need $\omega := e^{i\frac{2\pi}{p}}$ and $\mathbb{Z}_p^* := \mathbb{Z}_p \setminus \{0\}$. It is clear that since p is prime, \mathbb{Z}_p is a field which means that there exists a multiplicative inverse for all non-zero elements of \mathbb{Z}_p . We also need the following definition:

$$\chi_p(x) = \begin{cases} 1 & \text{if } \nexists y \in \mathbb{Z}_p \text{ s.t. } x = y^2; \\ 0 & \text{otherwise;} \end{cases} \quad (3.1)$$

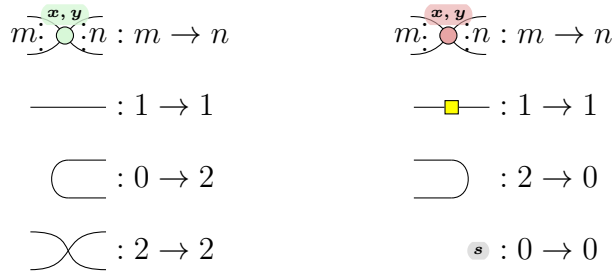
which is the characteristic function of the complement of the set of squares in \mathbb{Z}_p .

3.1 The calculus

In this section, we present a family of ZX-calculi, one for each odd prime dimension. Because of some group-theoretical properties of the qupit Clifford groups, it is possible to present the calculi relatively simply, without the need to explicitly consider p -dimensional rotations. These calculi also satisfy the property of *flexsymmetry*, proposed in [43], which allows one to recover the OCM meta-rule.

3.1.1 Generators

For any odd prime p , consider the symmetric monoidal category $\text{ZX}_p^{\text{Stab}}$ with objects \mathbb{N} and morphisms generated by the following diagrams:



where $x, y \in \mathbb{Z}_p$ and $s \in \mathbb{C}$.

A new generator we introduce to simplify the calculus is the light-grey bubble with a scalar written in the middle that we call an explicit scalar. Note that the empty diagram, $[\] : 0 \rightarrow 0$, is also a part of the language. Composition of morphisms is given by the sequential composition of diagrams, that is we connect output wires to input wires identically to the qubit case. The tensor product is given on morphisms by the parallel composition of diagrams and on the objects by $n \otimes m = n + m$.

It is important to note that there is a key difference between the qupits and the qubits. Among some others, the Hadamard gate does not equal its compositional

inverse. Instead, the application of 4 Hadamard gates equals the identity, that is $\mathbf{H}^4 = \mathbf{I}$. This means that the compositional inverse of the Hadamard gate is \mathbf{H}^3 . In order to keep diagrams relatively clean, we also define the following for shorthand:

$$\text{---} \square \text{---} := \text{---} \square \square \square \text{---} \quad (3.2)$$

which represents the compositional inverse of the Hadamard-box.

3.1.2 Interpretation

The interpretation of a ZX_p^{Stab} -diagram is defined on objects as $\llbracket m \rrbracket := \mathbb{C}^{p \times m}$, and on the generators of the morphisms as:

$$\begin{aligned} \llbracket \overset{x, y}{\circlearrowleft} \rrbracket &= \sum_{k \in \mathbb{Z}_p} \omega^{2^{-1}(xk+yk^2)} |k : Z\rangle^{\otimes n} \langle k : Z|^{\otimes m} \\ \llbracket \overset{x, y}{\circlearrowright} \rrbracket &= \sum_{k \in \mathbb{Z}_p} \omega^{2^{-1}(xk+yk^2)} |-k : X\rangle^{\otimes n} \langle k : X|^{\otimes m} \\ \llbracket \text{---} \rrbracket &= \sum_{k \in \mathbb{Z}_p} |k : Z\rangle \langle k : Z| & \llbracket \text{---} \square \text{---} \rrbracket &= \sum_{k \in \mathbb{Z}_p} |k : X\rangle \langle k : Z| \\ \llbracket \text{---} \square \rrbracket &= \sum_{k \in \mathbb{Z}_p} |kk : Z\rangle & \llbracket \square \text{---} \rrbracket &= \sum_{k \in \mathbb{Z}_p} \langle kk : Z| \\ \llbracket \text{---} \square \text{---} \rrbracket &= \sum_{k, \ell \in \mathbb{Z}_p} |k, \ell : Z\rangle \langle \ell, k : Z| & \llbracket \text{---} s \text{---} \rrbracket &= s \end{aligned}$$

where we denote $|k : Q\rangle$ as the eigenvector of a given Pauli Q associated with eigenvalue ω^k .

Note that the definition of the red, X-spider is differently from the normal conventions. It is defined in such a way that it maps X-eigenstates to their additive inverse. This definition is used in order to satisfy the property of flexsymmetry, which allows us to recover the OCM rule. Also note that spiders are defined with an additional 2^{-1} factor which is necessary for one of the axioms, **GAUSS**, to be sound.

3.2 Useful derivations

This section presents several rewrite rules that are derived from the axioms presented in Eq. (3.3). Furthermore, we introduce some syntactic sugar to represent multi-edges and multi-Hadamard-edges which we use extensively in further calculations.

3.2.1 Extension of axioms and further rules

While the axioms of Eq. (3.3) are enough to prove any equation, it is easier to prove a set of rules that we can use to perform higher-level transformations. Since most of the elementary derivations have been proved in Ref. [2], we only present these equations as proposals. First of all, we present some rules that allow us to move antipodes in some elementary ZX-diagrams.

Proposition 5. *The following equations are derivable in ZX_p^{Stab} ,*

(3.4)

We also present rules related to the Hadamard-box, its inverse, and antipodes.

Proposition 6. *The subsequent equations are derivable in ZX_p^{Stab} , for $a, b \in \mathbb{Z}_p$,*

(3.5)

(3.6)

$$\begin{aligned}
 & \text{---} \begin{array}{c} \square \\ \square \end{array} \text{---} = \text{---} = \text{---} \begin{array}{c} \square \\ \square \end{array} \text{---} \quad (3.7) \\
 & \begin{array}{c} \square \\ \vdots \\ \square \end{array} \begin{array}{c} a, b \\ \circ \\ \vdots \\ \square \end{array} = \begin{array}{c} \square \\ \vdots \\ \square \end{array} \begin{array}{c} a, b \\ \circ \\ \vdots \\ \square \end{array} \begin{array}{c} \square \\ \square \end{array} \quad \begin{array}{c} \square \\ \vdots \\ \square \end{array} \begin{array}{c} a, b \\ \circ \\ \vdots \\ \square \end{array} = \begin{array}{c} \square \\ \vdots \\ \square \end{array} \begin{array}{c} a, b \\ \circ \\ \vdots \\ \square \end{array} \begin{array}{c} \square \\ \square \end{array} \quad (3.8)
 \end{aligned}$$

Note that the Pauli phase of the spiders are negated on the second equation of Eq. (3.8).

Another useful derivation is the general case of the FUSION rule that is defined for both colours and allows any number of edges between spiders.

Proposition 7. *The following equations are derivable in ZX_p^{Stab} , for $a, b, c, d \in \mathbb{Z}_p$,*

$$\begin{array}{c} a, b \\ \vdots \\ \circ \\ \vdots \\ c, d \\ \vdots \\ \circ \\ \vdots \end{array} = \begin{array}{c} \vdots \\ \vdots \\ \circ \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array} \begin{array}{c} a+c, \\ b+d \\ \circ \end{array} \quad \begin{array}{c} a, b \\ \vdots \\ \circ \\ \vdots \\ c, d \\ \vdots \\ \circ \\ \vdots \end{array} = \begin{array}{c} \vdots \\ \vdots \\ \circ \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array} \begin{array}{c} a+c, \\ b+d \\ \circ \end{array} \quad (3.9)$$

Note that we need wires with antipodes in order to fuse red spiders.

Subsequently, we present some general lemmas using the axioms and the equations presented above that are used extensively in later proofs. Firstly, we prove that we can swap the colours of the spiders in a particular set of scalar diagrams.

Lemma 8. *We can change the colours of scalar diagrams of Z- and X-spiders connected with a single wire, for $a, b, c, d \in \mathbb{Z}_p$,*

$$\begin{array}{c} a, b \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} = \begin{array}{c} a, b \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \quad (3.10)$$

Proof.

$$\begin{array}{c} a, b \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} a, b \\ \circ \end{array} \begin{array}{c} \square \\ \square \end{array} \begin{array}{c} c, d \\ \circ \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} a, b \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \quad (3.11)$$

□

Similarly, we can also change the basis of single spider diagrams.

Lemma 9. *We can change $0 \rightarrow 0$ Z-spiders into X-spiders, for any $a, b \in \mathbb{Z}_p$,*

$$\begin{array}{c} a, b \\ \circ \end{array} = \begin{array}{c} a, b \\ \bullet \end{array} \quad (3.12)$$

Proof.

$$\begin{array}{c} a, b \\ \circ \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} a, b \\ \circ \text{---} \circ \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} a, b \\ \bullet \text{---} \square \text{---} \square \text{---} \bullet \end{array} \stackrel{\text{(Eq 3.6)}}{=} \begin{array}{c} a, b \\ \bullet \text{---} \square \text{---} \square \text{---} \bullet \end{array} \stackrel{\text{(Eq 3.5)}}{=} \begin{array}{c} a, b \\ \bullet \text{---} \bullet \text{---} \bullet \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} a, b \\ \bullet \end{array} \quad (3.13)$$

□

Furthermore, we provide a way to translate a set of scalar diagrams into their explicit scalar form.

Lemma 10. *For any $a, b, c \in \mathbb{Z}_p$,*

$$\begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} = \sqrt{p}\omega^{2^{-1}ac+2^{-2}a^2d} = \begin{array}{c} a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \circ \end{array}$$

Proof.

$$\begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} -a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(Z-ELIM)}}{=} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} -a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(LEM 3.4)}}{=} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} -a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(SHEAR)}}{=} \begin{array}{c} c+ad, d \\ \bullet \end{array} \begin{array}{c} -a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \begin{array}{c} c, d \\ \bullet \end{array} \stackrel{\text{(LEM 11)}}{=} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array} = \sqrt{p}\omega^{2^{-1}ac+2^{-2}a^2d} \quad (3.14)$$

The second equality follows from the application of Lemma 8. □

Then, we prove a general version of how Pauli states can absorb single qubit spiders of the other colour.

Lemma 11. *Pauli X-spiders absorb $1 \rightarrow 1$ Clifford Z-spiders and vice versa, for $a, c, d \in \mathbb{Z}_p$,*

$$\begin{array}{c} a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \circ \end{array} = \begin{array}{c} a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \circ \end{array} = \sqrt{p}\omega^{2^{-1}ac+2^{-2}a^2d} \begin{array}{c} a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \circ \end{array} = \begin{array}{c} a, 0 \\ \bullet \end{array} \begin{array}{c} c, d \\ \circ \end{array} = \sqrt{p}\omega^{2^{-1}ac+2^{-2}a^2d} \begin{array}{c} -a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \bullet \end{array}$$

Proof.

$$\begin{array}{c}
 \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \xrightarrow{\text{(FUSION)}} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \xrightarrow{\text{(COPY)}} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \xrightarrow{\text{(LEM 10)}} \begin{array}{c} \omega^{2^{-1}ac+2^{-2}a^2d} \\ \circ \end{array} \begin{array}{c} a, 0 \\ \circ \end{array} \quad (3.15)
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \xrightarrow{\text{(Eq 3.7)}} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \begin{array}{c} \square \\ \square \end{array} \xrightarrow{\text{(Eq 3.8)}} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} c, d \\ \circ \end{array} \begin{array}{c} \square \\ \square \end{array} \xrightarrow{\text{(Eq 3.15)}} \begin{array}{c} \omega^{2^{-1}ac+2^{-2}a^2d} \\ \circ \end{array} \begin{array}{c} a, 0 \\ \circ \end{array} \begin{array}{c} \square \\ \square \end{array} \\
 \xrightarrow{\text{(Eq 3.8)}} \begin{array}{c} \omega^{2^{-1}ac+2^{-2}a^2d} \\ \circ \end{array} \begin{array}{c} -a, 0 \\ \circ \end{array} \quad (3.16)
 \end{array}$$

□

We now show that we can change the basis of strictly Clifford states. Note that this proof is the core element in many proofs we present subsequently and therefore has great importance.

Lemma 12. *We can change the basis of strictly Clifford states as follows, for $a \in \mathbb{Z}_p$ and $z \in \mathbb{Z}_p^*$,*

$$\begin{array}{c}
 \begin{array}{c} a, z \\ \circ \end{array} = \begin{array}{c} \circ \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \begin{array}{c} az^{-1}, -z^{-1} \\ \circ \end{array} \quad \begin{array}{c} a, z \\ \circ \end{array} = \begin{array}{c} \circ \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} -az^{-1}, -z^{-1} \\ \circ \end{array} \quad (3.17)
 \end{array}$$

Proof. First of all,

$$\begin{array}{c}
 \begin{array}{c} az^{-1}, z^{-1} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \xrightarrow{\text{(M-ELIM)}} \begin{array}{c} -z^{-1}(-a), z^{-2}z \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \xrightarrow{\text{(M-ELIM)}} \begin{array}{c} -a, z \\ \circ \end{array} \begin{array}{c} z \\ \vdots \\ \circ \end{array} \begin{array}{c} \sqrt{p^{z-2}} \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \quad (3.18) \\
 \xrightarrow{\text{(X-ELIM)}} \begin{array}{c} -a, z \\ \circ \end{array} \begin{array}{c} z \\ \vdots \\ \circ \end{array} \begin{array}{c} \sqrt{p^{z-2}} \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \xrightarrow{\text{(MULT)}} \begin{array}{c} -a, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \begin{array}{c} 0, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \begin{array}{c} \square \\ \square \end{array} \\
 \xrightarrow{\text{(COLOUR)}} \begin{array}{c} a, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \begin{array}{c} 0, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \xrightarrow{\text{(Z-ELIM)}} \begin{array}{c} a, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array} \begin{array}{c} 0, z \\ \circ \end{array} \begin{array}{c} 0, z^{-1} \\ \circ \end{array}
 \end{array}$$

3.2. USEFUL DERIVATIONS

therefore,

$$\begin{aligned}
 & \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z} \\ \circ \end{array} \stackrel{\text{(X-ELIM)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z} \\ \circ \end{array} \begin{array}{c} \circ \\ \circ \end{array} \begin{array}{c} \circ \\ \circ \end{array} \\
 & \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \stackrel{\text{(Eq 3.18)}}{=} \begin{array}{c} \text{az}^{-1}, \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \quad (3.19) \\
 & \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \stackrel{\text{(Eq 3.4)}}{=} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array}
 \end{aligned}$$

hence,

$$\begin{aligned}
 & \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \stackrel{\text{(Z-ELIM)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \circ \\ \circ \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z} \\ \circ \end{array} \begin{array}{c} \text{0, z} \\ \circ \end{array} \quad (3.20) \\
 & \stackrel{\text{(Eq 3.19)}}{=} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \stackrel{\text{(LEM 11)}}{=} \begin{array}{c} \omega^{-2-2} \text{a}^2 \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \\
 & \stackrel{\text{(Eq 3.19)}}{=} \begin{array}{c} \omega^{-2-2} \text{a}^2 \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{1}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array}
 \end{aligned}$$

so finally,

$$\begin{aligned}
 & \begin{array}{c} \text{a, z} \\ \circ \end{array} \stackrel{\text{(X-ELIM)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \circ \\ \circ \end{array} \begin{array}{c} \circ \\ \circ \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{a, z} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \quad (3.21) \\
 & \stackrel{\text{(Eq 3.20)}}{=} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{0} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{\omega^{-2-2} \text{a}^2 \text{z}^{-1}}{\sqrt{p}} \\ \circ \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{\omega^{-2-2} \text{a}^2 \text{z}^{-1}}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \\
 & \stackrel{\text{(LEM 3.4)}}{=} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{\omega^{-2-2} \text{a}^2 \text{z}^{-1}}{\sqrt{p}} \\ \circ \end{array}
 \end{aligned}$$

The other equation of the lemma simply follows from colour-changing using the Hadamard-box as follows:

$$\begin{aligned}
 & \begin{array}{c} \text{a, z} \\ \circ \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} \text{a, z} \\ \square \end{array} \stackrel{\text{(Eq 3.21)}}{=} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{\omega^{-2-2} \text{a}^2 \text{z}^{-1}}{\sqrt{p}} \\ \circ \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \begin{array}{c} \text{az}^{-1}, \text{z}^{-1} \\ \circ \end{array} \begin{array}{c} \frac{\omega^{-2-2} \text{a}^2 \text{z}^{-1}}{\sqrt{p}} \\ \circ \end{array} \begin{array}{c} \text{0, z}^{-1} \\ \circ \end{array} \quad (3.22)
 \end{aligned}$$

□

Now that we have proved the qudit version of how to do state-change, we present further useful lemmas that rely on the result. For example, we can show how Pauli spiders commute through spiders of the other colour.

Lemma 13. *Z-spiders with any phase copy Pauli X-spiders, and vice versa, for any $a, c, d \in \mathbb{Z}_p$,*

$$\begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \approx \begin{array}{c} \text{green spider } (ad - c, d) \\ \text{red spider } (a, 0) \end{array} \quad \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \approx \begin{array}{c} \text{green spider } (c - ad, d) \\ \text{red spider } (-a, 0) \end{array} \quad (3.23)$$

Proof. First of all,

$$\begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(BIGEBRA)}}{=} \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(Eq 3.4)}}{=} \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \quad (3.24)$$

$$\begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(COPY)}}{=} \begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array}$$

Then, we separate the equation into two cases based on whether the Z-spider is Pauli or not. In case $d = 0$, the Z-spider is Pauli and therefore:

$$\begin{array}{c} \text{green spider } (c, 0) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(LEM 11)}}{=} \begin{array}{c} \text{green spider } (-c, 0) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider } (-c, 0) \\ \text{red spider } (a, 0) \end{array} \quad (3.25)$$

Note that if $d = 0$, then $ad - c = -c$ and so the lemma holds. Otherwise, $d \neq 0$ and therefore d^{-1} exists, so we can apply the state-change lemma:

$$\begin{array}{c} \text{green spider } (c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(LEM 12)}}{\approx} \begin{array}{c} \text{green spider } (cd^{-1}, -d^{-1}) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider } (a - cd^{-1}, -d^{-1}) \\ \text{red spider } (a, 0) \end{array} \quad (3.26)$$

$$\begin{array}{c} \text{green spider } (ad - c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(Eq 3.27)}}{\approx} \begin{array}{c} \text{green spider } (ad - c, d) \\ \text{red spider } (a, 0) \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider } (ad - c, d) \\ \text{red spider } (a, 0) \end{array}$$

Note that the phases after the application of the second state-change follow

from:

$$-(a - cd^{-1})(-d^{-1})^{-1}, -(-d^{-1})^{-1} = -(a - cd^{-1})(-d), d = ad - c, d \quad (3.27)$$

We can prove the second equation of the lemma using Hadamard-boxes as follows:

$$\approx \text{[Diagrammatic proof of equation 3.28]} \quad (3.28)$$

□

3.2.2 Multipliers

Unlike in the qubit case, a Z- and X-spider can be connected by more than just one edge, and such multi-edges cannot be simplified. Therefore, we add some syntactic sugar that represents such multi-edges in order to reduce the size of some recurring diagrams. In particular, their usage results in a nice representation of qupit graph states.

We extend our language by *multipliers*, which are defined recursively by:

$$\text{---}\langle 0 \rangle\text{---} := \text{---}\overset{\frac{1}{\sqrt{p}}}{\text{---}}\text{---} \quad \text{and} \quad \text{---}\langle m+1 \rangle\text{---} := \text{---}\overset{\sqrt{p}}{\text{---}}\text{---} \quad (3.29)$$

It turns out that we can prove the following equation using **CHAR**, for any $m \in \mathbb{N}$,

$$\text{---}\langle m \rangle\text{---} = \text{---}\langle m \bmod p \rangle\text{---} \quad (3.30)$$

We can explicitly express multipliers as, for $x \in \mathbb{Z}_p^*$,

$$\text{---}\langle x \rangle\text{---} = \text{---}\overset{\sqrt{p^{x-1}}}{\text{---}}\text{---} \quad (3.31)$$

We also define inverted multipliers, which is a multiplier pointing in the other direction:

$$\langle x \leftarrow := \leftarrow x \quad (3.32)$$

The following equations hold for multipliers and are proved in Ref. [2]:

Proposition 14.

$$\begin{aligned} \langle x \rangle y &= \langle xy \rangle & \langle z^{-1} \rangle &= \langle z \rangle \\ \bullet &= \langle 1 \rangle & \begin{array}{c} \langle x \rangle \\ \bullet \\ \langle y \rangle \end{array} &= \langle x + y \rangle \\ \text{---} &= \langle 1 \rangle & \langle p \rangle &= \langle 0 \rangle \end{aligned} \quad (3.33)$$

The action of multipliers on spiders is given by, for any $x \in \mathbb{Z}_p^*$,

$$\begin{aligned} \begin{array}{c} \langle x \rangle \\ \vdots \\ \langle x \rangle \end{array} \begin{array}{c} a, b \\ \vdots \\ \langle x \rangle \end{array} &= \begin{array}{c} \langle ax, bx^2 \rangle \\ \vdots \\ \langle x \rangle \end{array} & \begin{array}{c} \langle x \rangle \\ \vdots \\ \langle x \rangle \end{array} \begin{array}{c} a, b \\ \vdots \\ \langle x \rangle \end{array} &= \begin{array}{c} \langle ax, bx^2 \rangle \\ \vdots \\ \langle x \rangle \end{array} \end{aligned} \quad (3.34)$$

It is worth pointing out that multipliers are not flexsymmetric and therefore OCM is technically lost if they are present. This is one of the reasons why multipliers are only a syntactic sugar of the language. Nevertheless, we can recover OCM up to a point by dealing with directed graphs with labelled edges.

Note that by applying a Hadamard-box to the multiplier we construct a symmetric gate:

$$\langle x \rangle \square = \square \langle x \rangle \quad (3.35)$$

Therefore, we can define *H-boxes*:

$$\square x := \langle x \rangle \square \quad (3.36)$$

Explicitly, H-boxes are equivalent to repeated Hadamard wires, for $x \in \mathbb{Z}_p$,

$$\square x = \begin{array}{c} \square \\ \vdots \\ \square \end{array} \begin{array}{c} \sqrt{p^{x-1}} \\ \vdots \\ \square \end{array} \quad (3.37)$$

Note that, unlike multipliers, H-boxes are flexsymmetric, so we can move such boxes freely in a diagram.

We now present some equations related to H-boxes that we use through many proofs. Note that the proofs of these equations can be found in Ref. [2] and are not present in this thesis.

Proposition 15. zx_p proves the following equations:

$$\begin{array}{lcl}
 \text{---} \boxed{x} \text{---} \boxed{y} \text{---} & = & \text{---} \boxed{xy^{-1}} \text{---} \\
 \text{---} \boxed{x} \text{---} \boxed{-x} \text{---} & = & \text{---} \\
 \text{---} \boxed{0} \text{---} & = & \text{---} \circ \text{---} \circ \text{---} \\
 \text{---} \circ \text{---} \circ \text{---} & = & \text{---} \boxed{x+y} \text{---} \\
 \text{---} \boxed{x} \text{---} \boxed{x} \text{---} \boxed{x} \text{---} & = & \text{---} \boxed{-x} \text{---} \\
 \text{---} \boxed{1} \text{---} & = & \text{---} \boxed{} \text{---}
 \end{array} \tag{3.38}$$

Furthermore, Hadamard-loops correspond to pure-Clifford operations, for any $x \in \mathbb{Z}_p$ and $z \in \mathbb{Z}_p^*$,

$$\begin{array}{lcl}
 \text{---} \boxed{x} \text{---} \circ \text{---} & = & \text{---} \overset{0, 2x}{\circ} \text{---} \\
 \text{---} \boxed{-z^{-1}} \text{---} \circ \text{---} & = & \text{---} \overset{0, 2z}{\circ} \text{---}
 \end{array} \tag{3.39}$$

In addition to these rewrite rules, we also prove the subsequent lemmas for convenient calculations.

Lemma 16. H-boxes multiply with multipliers, for any $x, y \in \mathbb{Z}_p$,

$$\text{---} \boxed{y} \text{---} \boxed{x} \text{---} = \text{---} \boxed{xy} \text{---} = \text{---} \boxed{x} \text{---} \boxed{y} \text{---}$$

Proof.

$$\begin{array}{lcl}
 \text{---} \boxed{x} \text{---} \boxed{y} \text{---} & = & \text{---} \boxed{} \text{---} \boxed{x} \text{---} \boxed{y} \text{---} = \text{---} \boxed{} \text{---} \boxed{xy} \text{---} = \text{---} \boxed{xy} \text{---} \\
 \text{---} \boxed{y} \text{---} \boxed{x} \text{---} & = & \text{---} \boxed{y} \text{---} \boxed{x} \text{---} \boxed{} \text{---} = \text{---} \boxed{xy} \text{---} \boxed{} \text{---} = \text{---} \boxed{xy} \text{---}
 \end{array}$$

□

Lemma 17. We can ‘push’ multipliers through spiders as follows, for any $a, b \in \mathbb{Z}_p$ and $x \in \mathbb{Z}_p^*$,

Proof.

The other proofs follow from the above equations while using the multiplicative inverse of the multipliers as presented in Proposition 14. \square

Lemma 18. We can ‘push’ H-boxes through spiders as follows, for any $a, b \in \mathbb{Z}_p$ and $x \in \mathbb{Z}_p^*$,

Proof. First of all,

The other equation can be proved similarly,

| □

Since edges that contain H-boxes are central to most subsequent proofs, we define *H-edges*, similarly to the qubit case, as a blue-, dashed-line with the corresponding numbers written near the line, for $x \in \mathbb{Z}_p$,

$$\begin{array}{c} \vdots \\ \vdots \end{array} \left(\begin{array}{c} \circ \\ \vdots \end{array} \right) \text{---} x \text{---} \left(\begin{array}{c} \circ \\ \vdots \end{array} \right) \begin{array}{c} \vdots \\ \vdots \end{array} := \begin{array}{c} \vdots \\ \vdots \end{array} \left(\begin{array}{c} \circ \\ \vdots \end{array} \right) \text{---} \boxed{x} \text{---} \left(\begin{array}{c} \circ \\ \vdots \end{array} \right) \begin{array}{c} \vdots \\ \vdots \end{array} \quad (3.42)$$

Lastly, we note that scalar diagrams of Pauli X-spiders can be expressed as explicit scalars.

Lemma 19. *Scalar diagrams of Pauli X-spiders connected through a multiplier and a Hadamard-box equal the following explicit scalar, for any $a, b \in \mathbb{Z}_p$ and $x \in \mathbb{Z}_p^*$,*

$$\begin{array}{c} a, 0 \quad b, 0 \\ \circ \quad \circ \\ \vdots \quad \vdots \end{array} \text{---} x \text{---} \begin{array}{c} \circ \\ \vdots \end{array} = \sqrt{p}\omega^{-2^{-1}abx} \quad (3.43)$$

Proof.

$$\begin{array}{c} a, 0 \quad x \quad b, 0 \\ \circ \quad \circ \\ \vdots \quad \vdots \end{array} \stackrel{\text{(Eq 3.36)}}{=} \begin{array}{c} a, 0 \quad \text{---} x \quad b, 0 \\ \circ \quad \circ \\ \vdots \quad \vdots \end{array} \stackrel{\text{(COLOUR)}}{=} \begin{array}{c} a, 0 \quad \text{---} x \quad -b, 0 \\ \circ \quad \circ \\ \vdots \quad \vdots \end{array} \stackrel{\text{(Eq 3.34)}}{=} \begin{array}{c} ax, 0 \quad -b, 0 \\ \circ \quad \circ \\ \vdots \quad \vdots \end{array} \stackrel{\text{(LEM 10)}}{=} \sqrt{p}\omega^{-2^{-1}abx} \quad (3.44)$$

| □

3.3 Scalar completeness

In this section, we prove the completeness of the scalar fragment of ZX_p^{Stab} . To do so we first prove that a state can absorb any single qupit diagram. Then, we show that each of the possible scalar diagrams can be expressed as an explicit scalar; thus we can prove any equation regarding scalars.

The $1 \rightarrow 1$ diagrams in our calculus is generated by the following gates, for

any $x, y \in \mathbb{Z}_p$

$$\begin{array}{cc}
 \begin{array}{c} \text{red } x, y \\ \circ \\ \text{---} \end{array} & \begin{array}{c} \text{green } x, y \\ \circ \\ \text{---} \end{array} \\
 \begin{array}{c} \text{---} \\ \text{red } x \\ \text{---} \end{array} & \begin{array}{c} \text{---} \\ \text{yellow } \square \\ \text{---} \end{array}
 \end{array} \tag{3.45}$$

We call any such diagram a *single-qubit Clifford diagram*.

Lemma 20. *Any scalar diagram can be transformed into an elementary scalar, i.e. one that corresponds to an explicit scalar in our axioms.*

Proof. It is clear that the single-qubit Clifford group is generated by the invertible generators under sequential composition. Therefore, it suffices to show that composing state diagrams with either of the diagrams in Eq. (3.45) can also be transformed into a state diagram with some elementary scalars. We show this using several case distinctions. First of all, any state can absorb a Hadamard-box using the **COLOUR** rule as follows:

$$\begin{array}{cc}
 \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{yellow } \square \end{array} & \stackrel{\text{(COLOUR)}}{=} & \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \end{array} & \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \\ \text{yellow } \square \end{array} & \stackrel{\text{(COLOUR)}}{=} & \begin{array}{c} \text{green } -a, b \\ \circ \\ \text{---} \end{array}
 \end{array} \tag{3.46}$$

Secondly, we can show that multipliers are also absorbed by states, for $z \in \mathbb{Z}_p^*$:

$$\begin{array}{cc}
 \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{grey } \text{Z} \end{array} & \stackrel{\text{(Eq 3.34)}}{=} & \begin{array}{c} \text{green } az^{-1}, bz^{-2} \\ \circ \\ \text{---} \end{array} & \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \\ \text{grey } \text{Z} \end{array} & \stackrel{\text{(Eq 3.34)}}{=} & \begin{array}{c} \text{red } az, bz^2 \\ \circ \\ \text{---} \end{array}
 \end{array} \tag{3.47}$$

In case the multiplier has weight 0, we have the following reduction:

$$\begin{array}{cc}
 \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{grey } \text{0} \end{array} & \stackrel{\text{(Eq 3.29)}}{=} & \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{grey } \frac{1}{\sqrt{p}} \end{array} & \stackrel{\text{(FUSION)}}{=} & \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{red } \frac{1}{\sqrt{p}} \end{array}
 \end{array} \tag{3.48}$$

$$\begin{array}{cc}
 \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \\ \text{grey } \text{0} \end{array} & \stackrel{\text{(Eq 3.29)}}{=} & \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \\ \text{grey } \frac{1}{\sqrt{p}} \end{array} & \stackrel{\text{(FUSION)}}{=} & \begin{array}{c} \text{red } \text{---} \end{array}
 \end{array} \tag{3.49}$$

Another obvious case is when there are spiders of the same colour, which we can reduce as follows:

$$\begin{array}{cc}
 \begin{array}{c} \text{green } a, b \\ \circ \\ \text{---} \\ \text{green } c, d \\ \circ \\ \text{---} \end{array} & \stackrel{\text{(FUSION)}}{=} & \begin{array}{c} \text{green } a+c, b+d \\ \circ \\ \text{---} \end{array} & \begin{array}{c} \text{red } a, b \\ \circ \\ \text{---} \\ \text{red } c, d \\ \circ \\ \text{---} \end{array} & \stackrel{\text{(FUSION)}}{=} & \begin{array}{c} \text{red } -a+c, b+d \\ \circ \\ \text{---} \end{array}
 \end{array} \tag{3.50}$$

Lastly, the case that involves the most complication is when a state has to absorb a spider of the other colour. In case the state is Pauli, we can reduce the diagram as follows:

$$\begin{array}{c} \text{a, 0} \quad \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(LEM 11)}} \begin{array}{c} \omega^{2^{-1}ac+2^{-2}a^2d} \\ \text{-a, 0} \\ \circ \end{array} \quad \begin{array}{c} \text{a, 0} \quad \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(LEM 11)}} \begin{array}{c} \omega^{2^{-1}ac+2^{-2}a^2d} \\ \text{a, 0} \\ \circ \end{array} \quad (3.51)$$

And finally, we can use the state-change lemma to show that a strictly Clifford state can absorb spiders of the other colour as follows,

$$\begin{array}{c} \text{a, z} \quad \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(LEM 12)}} \begin{array}{c} \circ \quad \text{0, z}^{-1} \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} \text{az}^{-1}, -z^{-1} \\ \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(FUSION)}} \begin{array}{c} \circ \quad \text{0, z}^{-1} \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} \text{c - az}^{-1}, \text{d - z}^{-1} \\ \circ \end{array} \quad (3.52)$$

$$\begin{array}{c} \text{a, z} \quad \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(LEM 12)}} \begin{array}{c} \circ \quad \text{0, z}^{-1} \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} \text{-az}^{-1}, -z^{-1} \\ \text{c, d} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(FUSION)}} \begin{array}{c} \circ \quad \text{0, z}^{-1} \\ \omega^{-2^{-2}a^2z^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} \text{c - az}^{-1}, \text{d - z}^{-1} \\ \circ \end{array} \quad (3.53)$$

In conclusion, we have shown that an arbitrary state can absorb any single-qubit Clifford diagram. Therefore, without loss of generality, we can reduce any scalar diagram to a state composed with an effect without a phase. Here we have two cases, either the state and the effect are coloured differently or identically. If they are of different colours, the scalar diagram is an elementary one and corresponds to \sqrt{p} . When they are of the same colour we can fuse them resulting in either a Z- or an X-spider with no legs. As we have shown in Lemma 9, we can convert such X-spiders to Z-spiders so we only have to deal with that case. We can also suppose that neither component of the phase is 0 since that is already an elementary scalar, therefore, for any $s, t \in \mathbb{Z}_p^*$,

$$\begin{array}{c} \text{s, t} \\ \circ \end{array} \xrightarrow{\text{(FUSION)}} \begin{array}{c} \text{s, t} \\ \circ \quad \circ \end{array} \xrightarrow{\text{(LEM 12)}} \begin{array}{c} \circ \quad \text{0, t}^{-1} \\ \omega^{-2^{-2}s^2t^{-1}} \\ \sqrt{p} \end{array} \begin{array}{c} \text{st}^{-1}, -t^{-1} \\ \circ \end{array} \xrightarrow{\text{(FUSION)}} \begin{array}{c} \circ \quad \text{0, t}^{-1} \\ \omega^{-2^{-2}s^2t^{-1}} \\ \sqrt{p} \end{array} \quad (3.54)$$

We conclude that any scalar diagram can be converted into an explicit scalar.

| □

Theorem 21. *For any pair of scalar-diagrams $A, B \in \text{ZX}_p^{\text{Stab}}[0, 0]$, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, we can provide a sequence of rewrites that transform the scalar A into B .*

Proof. It is clear that we can convert both A and B into an explicit scalar form using Lemma 20. Given that $\llbracket A \rrbracket = \llbracket B \rrbracket$, the sequence of transformations from A to A in explicit scalar form composed with the series rewrites from B in explicit scalar to B proves us a sequence of rewrites that transforms A into B . □

3.4 Graph states

As presented in Chapter 2, graph states are a powerful tool in quantum information theory. They are closely linked to Clifford states and have some nice properties. For example, rewrite rules such as local complementation and pivoting can be proved using such graph states. Furthermore, the completeness proof we present also relies extensively on a variant of graph states. The qubit graph states have been generalised to the case of an arbitrary (finite) dimension, and we present them here.

The idea is to associate a state in $\mathbb{Z}_p^{\otimes V}$ to any graph with vertex set V . Unlike in the qubit case, we use \mathbb{Z}_p -edge-weighted graphs that we identify with the adjacency matrix $G \in \mathbb{Z}_p^{V \times V}$. Then, the corresponding graph state is as follows,

$$|G\rangle = \prod_{\substack{(u,v) \in V \\ u \neq v}} E_{u,v}^{G_{u,v}} |0 : X\rangle^{\otimes V}. \quad (3.55)$$

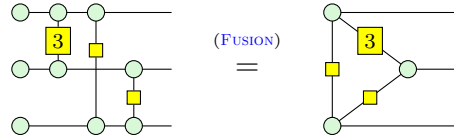
In other words, we can create a graph state by first initialising the qupits in the state $\text{ket}0 : X$. Then we apply the entangling operation $E_{uv}^{G_{uv}}$, an H-box with weight G_{uv} , for each edge in G . Note that since the $E_{u,v}^{G_{u,v}}$ gates commute, they can be applied in any order, just like in the qubit case.

As a simple example, the graph



has adjacency matrix $\begin{pmatrix} 0 & 3 & 1 \\ 3 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ (3.56)

and is associated with the graph state $E_{1,2}^3 E_{1,3} E_{2,3} |0 : X\rangle^{\otimes 3}$. And the graph state corresponding to the graph in Eq. (3.56) is as follows:



(3.57)

It is clear that we can obtain a graph state in $\mathbb{Z}X_p^{\text{Stab}}$ for any given graph $G \in \mathbb{Z}_p^{V \times V}$. This is done by identifying each vertex of the graph with a green spider, and each edge with a correspondingly weighted H-edge. More formally a $\mathbb{Z}X_p^{\text{Stab}}$ -diagram is a graph state diagram if:

1. It contains only green spiders;
2. Each spider is connected to a single output by a plain wire;
3. The spiders are connected only by H-edges.

3.5 Graph simplifications

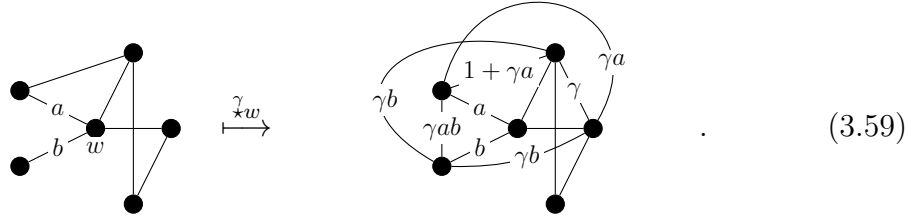
Now that we have seen how to represent graph states in $\mathbb{Z}X_p^{\text{Stab}}$, we give some rules to transform such diagrams.

3.5.1 Weighted local complementation

For any $\gamma \in \mathbb{Z}_d^*$, the γ -weighted local \mathbb{Z}_d -complementation or γ -complementation about a vertex w in a graph $G \in \mathbb{Z}_d^{V \times V}$ is defined as:

$$(G \overset{\gamma}{\star} w)_{uv} := \begin{cases} G_{uv} + \gamma G_{uw} G_{vw} & \text{if } u \neq v; \\ G_{uv} & \text{otherwise.} \end{cases} \quad (3.58)$$

For example, the following graph and its γ -complementation about the vertex w are presented in the following equation:



We now show the γ -weighted local \mathbb{Z}_d -complementation for graphs states in $\mathbb{Z}X_p^{\text{Stab}}$.

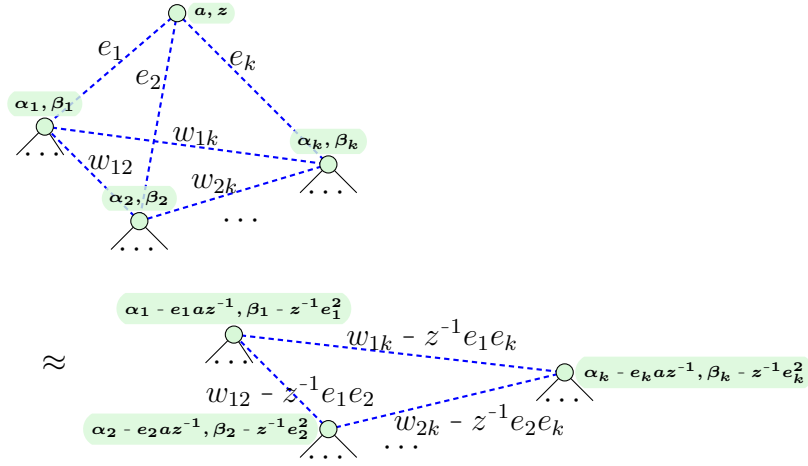
Proposition 22. γ -weighted local \mathbb{Z}_d -complementation is derivable in $\mathbb{Z}X_p$, for any graph $G \in \mathbb{Z}_p^{V \times V}$, $\gamma \in \mathbb{Z}_p$ and $u \in V$,

Note that the proof of this proposition can be found in Ref. [2].

3.5.2 Local complementation

As we have seen in the qubit case, there is another version of local complementation that we used in the qubit case. This allows us to remove a strictly Clifford inner spider by adding phases and wires to the spiders it is connected to. In this section, we show and prove local complementation in $\mathbb{Z}\mathbb{X}_p^{\text{Stab}}$.

Lemma 23. *Local complementation is derivable in $\mathbb{Z}\mathbb{X}_p$, for any $z \in \mathbb{Z}_p^*$ and for all $a, \alpha_i, \beta_i, e_i, w_{i,j} \in \mathbb{Z}_p$ where $i, j \in \{1, \dots, k\}$ such that $i < j$,*



Proof. First, we can prove a simplified version of the lemma without phases of

the boundary spiders and H-edges as follows,

(3.61)

Then, we can use the previous equation to prove the lemma.

(3.62)

□

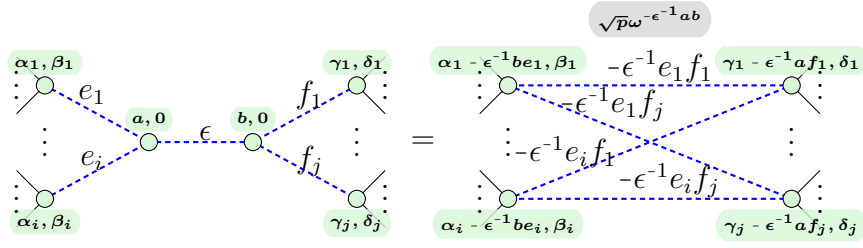
3.5.3 Pivoting

Further to local complementation, there is another important graphical rewrite rule related to graph states, pivoting. Pivoting allows us to remove connected Pauli spiders by adding some phases and connections to the spiders they are connected to. In this section, we show and prove pivoting in $\mathbb{Z}X_p^{\text{Stab}}$.

First, we prove a simplified version of the local complementation.

Lemma 24. *The following version of pivoting is derivable in $\mathbb{Z}X_p$: for any $\epsilon \in \mathbb{Z}_p^*$, $a, b \in \mathbb{Z}_p$; furthermore, for all $\alpha_k, \beta_k, e_k \in \mathbb{Z}_p$ and $\gamma_\ell, \delta_\ell, f_\ell \in \mathbb{Z}_p$ where $k \in$*

$\{1, \dots, i\}$ and $\ell \in \{1, \dots, j\}$,



Proof. First, we can prove a simplified version of the equation that omits the

3.5. GRAPH SIMPLIFICATIONS

phases of the boundary spiders as follows,

(3.63)

Then, we can use the previous equation to prove the lemma as follows,

$$\begin{aligned}
 & \begin{array}{c} \alpha_1, \beta_1 \\ \vdots \\ \alpha_i, \beta_i \end{array} \begin{array}{c} e_1 \\ \vdots \\ e_i \end{array} \begin{array}{c} a, 0 \\ \vdots \\ \epsilon \end{array} \begin{array}{c} b, 0 \\ \vdots \\ \epsilon \end{array} \begin{array}{c} f_1 \\ \vdots \\ f_j \end{array} \begin{array}{c} \gamma_1, \delta_1 \\ \vdots \\ \gamma_j, \delta_j \end{array} \quad (\text{FUSION}) \\
 &= \begin{array}{c} \alpha_1, \beta_1 \\ \vdots \\ \alpha_i, \beta_i \end{array} \begin{array}{c} e_1 \\ \vdots \\ e_i \end{array} \begin{array}{c} a, 0 \\ \vdots \\ \epsilon \end{array} \begin{array}{c} b, 0 \\ \vdots \\ \epsilon \end{array} \begin{array}{c} f_1 \\ \vdots \\ f_j \end{array} \begin{array}{c} \gamma_1, \delta_1 \\ \vdots \\ \gamma_j, \delta_j \end{array} \\
 & \stackrel{(\text{Eq 3.63})}{=} \begin{array}{c} \alpha_1, \beta_1 \\ \vdots \\ \alpha_i, \beta_i \end{array} \begin{array}{c} -\epsilon^{-1} be_1, 0 \\ \vdots \\ -\epsilon^{-1} be_i, 0 \end{array} \begin{array}{c} \sqrt{p}\omega^{-\epsilon^{-1}ab} \\ \vdots \\ \sqrt{p}\omega^{-\epsilon^{-1}ab} \end{array} \begin{array}{c} \gamma_1, \delta_1 \\ \vdots \\ \gamma_j, \delta_j \end{array} \\
 & \stackrel{(\text{FUSION})}{=} \begin{array}{c} \alpha_1 - \epsilon^{-1} be_1, \beta_1 \\ \vdots \\ \alpha_i - \epsilon^{-1} be_i, \beta_i \end{array} \begin{array}{c} -\epsilon^{-1} e_1 f_1 \\ \vdots \\ -\epsilon^{-1} e_i f_i \end{array} \begin{array}{c} \gamma_1 - \epsilon^{-1} af_1, \delta_1 \\ \vdots \\ \gamma_j - \epsilon^{-1} af_j, \delta_j \end{array}
 \end{aligned} \tag{3.64}$$

□

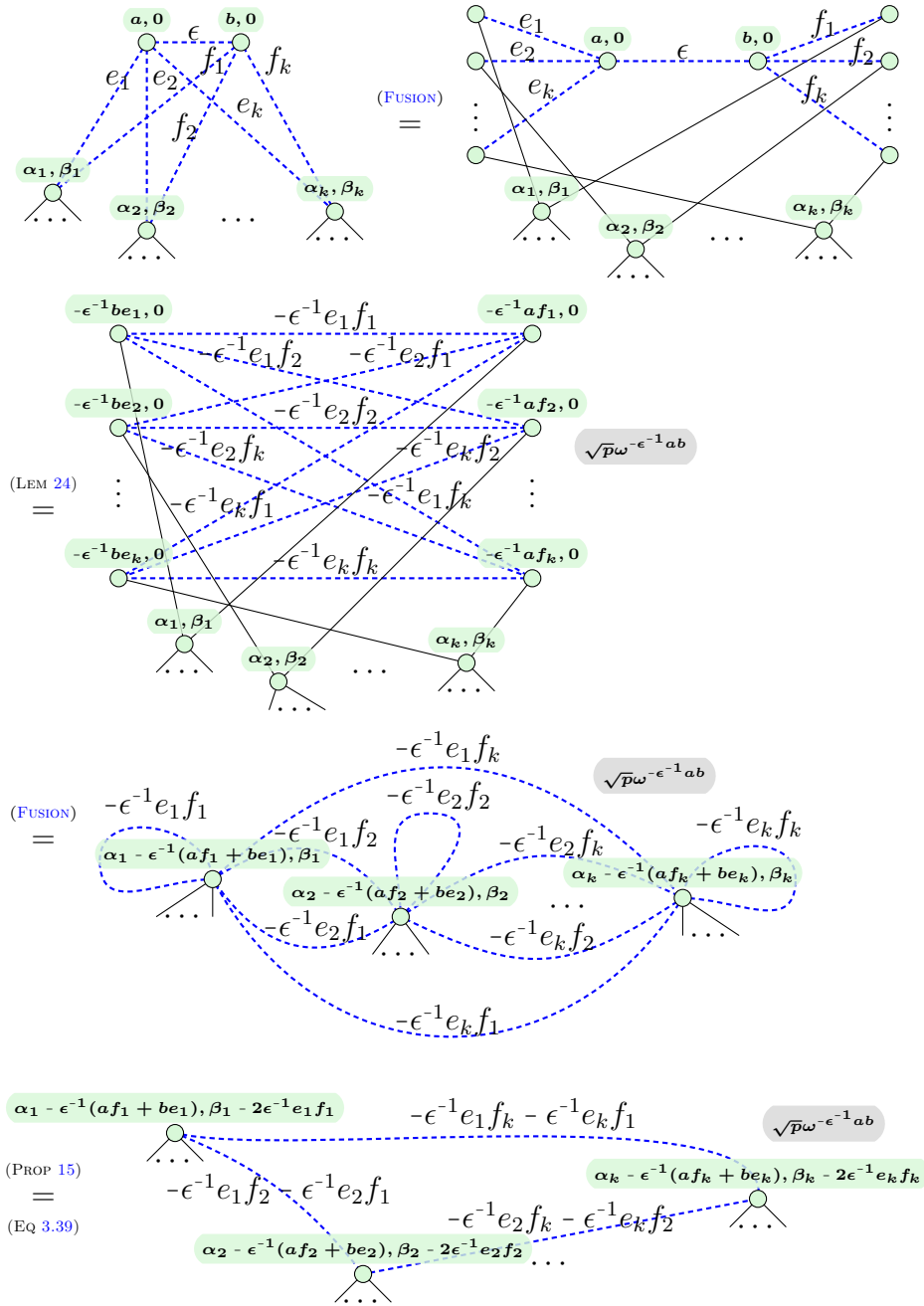
Now, we prove the general version of local complementation.

Lemma 25. *A general version of pivoting is derivable in zx_p : for any $\epsilon \in \mathbb{Z}_p^*$, $a, b \in \mathbb{Z}_p$ and for all $\alpha_i, \beta_i, e_i, f_i \in \mathbb{Z}_p$ where $i \in \{1, \dots, k\}$,*

$$\begin{aligned}
 & \begin{array}{c} a, 0 \\ \vdots \\ \alpha_1, \beta_1 \end{array} \begin{array}{c} e_1 \\ \vdots \\ e_2 \end{array} \begin{array}{c} \epsilon \\ \vdots \\ \epsilon \end{array} \begin{array}{c} b, 0 \\ \vdots \\ \alpha_k, \beta_k \end{array} \begin{array}{c} f_1 \\ \vdots \\ f_k \end{array} \\
 & \stackrel{(\text{FUSION})}{=} \begin{array}{c} \alpha_1 - \epsilon^{-1}(af_1 + be_1), \beta_1 - 2\epsilon^{-1}e_1 f_1 \\ \vdots \\ \alpha_k - \epsilon^{-1}(af_k + be_k), \beta_k - 2\epsilon^{-1}e_k f_k \end{array} \begin{array}{c} -\epsilon^{-1}e_1 f_k - \epsilon^{-1}e_k f_1 \\ \vdots \\ -\epsilon^{-1}e_2 f_k - \epsilon^{-1}e_k f_2 \end{array} \begin{array}{c} \sqrt{p}\omega^{-\epsilon^{-1}ab} \\ \vdots \\ \sqrt{p}\omega^{-\epsilon^{-1}ab} \end{array}
 \end{aligned}$$

|

Proof.



□

Chapter 4

A normal form

In this chapter, we present the qudit version of the AP-form and some of its applications for odd prime dimensions. We first define the AP-form in ZX_p^{Stab} and present some of its properties. Then, we show the qudit version of the weak simulation algorithm presented in Ref. [38]. Lastly, we prove the completeness of the Clifford qudit ZX-calculus for odd prime dimensions.

4.1 AP form

Similarly to the qubit case, we can define the AP-form of diagrams in ZX_p^{Stab} . However, we need to deal with more complicated phases and H-edges between spiders; therefore, the complexity of proofs increases. We say that a graph-like diagram is in *Affine with Phases form* (AP-form) when:

- There are no inputs;
- The internal spiders are Pauli-spiders;
- Internal spiders are only connected to boundary spiders.

Note that we say that a diagram is in Affine with Phases form because it describes an affine subspace with some additional phase function.

We claim that any diagram in ZX_p^{Stab} can be transformed into one in AP-form. This can be done by first transforming the diagram into a graph-like one. Then, applying local complementation and pivoting as much as possible results in a diagram that is in AP-form. It is clear that by applying local complementation we eliminate each strictly Clifford inner spider. Furthermore, pivoting removes any pair of connected inner Pauli spiders. Therefore, we reach a diagram that has neither strictly Clifford inner spiders, nor any connected pair of inner Pauli spiders. That is, the application of local complementation and pivoting transforms a graph-like diagram into one in AP-form.

The general form of a diagram in AP-form can be described and decomposed similarly to the qubit case as follows, for any $a_i, \alpha_i, \beta_i, e_{h,i}, f_{i,j} \in \mathbb{Z}_p$ where $h \in \{1, \dots, k\}$ and $i, j \in \{1, \dots, \ell\}$ such that $i < j$:

(4.1)

We claim that the above diagram, that we describe as the state $|\psi\rangle$, equals the following state up to some global phase:

$$|\psi\rangle \approx \sum_{E\vec{x}=\vec{a}} \omega^{\phi(\vec{x})} |\vec{x}\rangle \quad (4.2)$$

where E is the parity matrix describing the connectivity of the inner and boundary spiders and \vec{a} corresponds to the Pauli phases of the internal spiders as de-

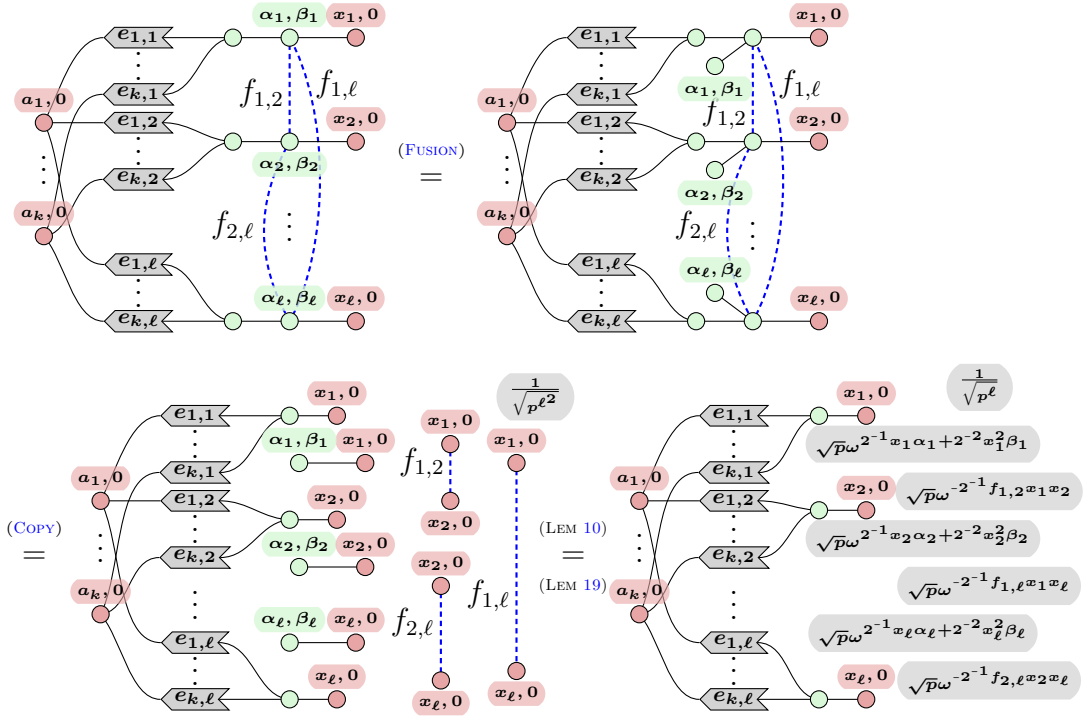
scribed subsequently:

$$E = \begin{bmatrix} e_{1,1} & \cdots & e_{1,\ell} \\ e_{2,1} & \cdots & e_{2,\ell} \\ \vdots & & \vdots \\ e_{k,1} & \cdots & e_{k,\ell} \end{bmatrix}, \quad \vec{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix}. \quad (4.3)$$

Furthermore, ϕ is a phase function that describes the connectivity and phases of the boundary spiders:

$$\phi(\vec{x}) = \sum_{\substack{i,j \in \{1, \dots, \ell\} \\ i < j}} 2^{-1} x_i \alpha_i + 2^{-2} x_i^2 \beta_i - 2^{-1} f_{i,j} x_i x_j \quad (4.4)$$

We can prove this claim purely diagrammatically, by composing the diagram of Eq. (4.1) with an effect that corresponds to the vector $\langle x |$. By rewriting the diagram while keeping track of the scalars, we can prove that the diagram indeed represents the one described in Eq. (4.2). These transformations are as follows:



Our goal is to efficiently simulate $\{|0\rangle, \dots, |d-1\rangle\}$ measurements on the state $|\psi\rangle$. To do so, we argue as follows. The outcome probabilities of a measurement in the computational basis are independent of the phase function. That is, the specific value of ϕ is in this context completely redundant. Therefore, we can set the phase function to its trivial value $\phi \equiv 0$. This leaves us with the following state:

$$\sum_{E\vec{x}=\vec{a}} |\vec{x}\rangle \tag{4.5}$$

Then, sampling from this probability distribution is relatively easy. We simply generate a vector $\vec{u} \in \mathbb{Z}_p^\ell$ uniformly at random. Then, calculating the equation $R\vec{u} + \vec{a}$ results in a sample from the state $|\psi\rangle$. This shows that each Clifford circuit can efficiently be simulated in the weak sense.

4.3 Completeness

In this section, we prove the completeness of $\mathbf{ZX}_p^{\text{Stab}}$. Recall, that we say a graphical calculus is complete if its rewrite rules can prove any true equation. More formally, graphical calculus is complete if, for any diagram $A, B \in \mathbf{ZX}$ such that $\llbracket A \rrbracket = \llbracket B \rrbracket$, we can provide a sequence of rewrites that transforms A into B .

To prove the completeness of the calculus, we use a more restricted version of the AP-form. We say that a diagram in AP-form defined by E, \vec{A} , and ϕ is in *reduced AP-form* if it is 0 or it is non-zero and:

- E is in reduced row echelon form (RREF) with no zero rows;
- ϕ only contains free variables from the equation system $E\vec{x} = \vec{a}$.

We claim that any diagram in $\mathbf{ZX}_p^{\text{Stab}}$ equals one in reduced AP-form, and this form is unique.

First of all, let us prove the uniqueness of the form.

Lemma 26. *For any non-zero state $|\psi\rangle$, there is at most one triple (E, \vec{b}, ϕ) satisfying the conditions of reduced AP-form such that:*

$$|\psi\rangle \approx \sum_{E\vec{x}=\vec{a}} \omega^{\phi(\vec{x})} |\vec{x}\rangle$$

Proof. Since $|\psi\rangle \neq 0$, the set $\mathcal{A} = \{\vec{x} \mid E\vec{x} = \vec{a}\}$ is non-empty. Therefore, there is a unique system of equations in RREF that define \mathcal{A} . This means that E and \vec{a} are uniquely fixed. Now, for any assignment $\{x_{i_1} := c_1, \dots, x_{i_k} := c_k\}$ of free variables, there exists a state $|\vec{x}\rangle \in \mathcal{A}$ such that $x_{i_\mu} = c_\mu$. Therefore, we have $\langle \vec{x} | \psi \rangle = \omega^{\phi(c_1, \dots, c_k)}$ for some fixed constant $\lambda \neq 0$. Using this fact we can determine the value of ϕ at all inputs (c_1, \dots, c_k) which is enough to compute each coefficient of ϕ . We conclude that ϕ is uniquely fixed by $|\psi\rangle$. \square

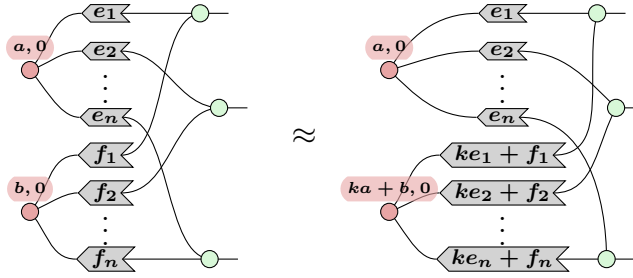
It is clear that any diagram in $\text{ZX}_p^{\text{Stab}}$ can be transformed into one in AP-form using local complementation and pivoting. As the first step, we have to show that we can rewrite a ZX-diagram in AP-form in such a way that its biadjacency matrix E is transformed into one in RREF. Then, we also have to show that we can transform the diagram in such a way to ensure that ϕ only contains free variables from the equation system $E\vec{x} = \vec{a}$. That is, we have to prove that any phase or Hadamard edges connected to boundary spiders can be removed from a pivot spider.

First, we show that we can perform primitive row operations in a ZX-diagram in AP-form; thus, we can transform a diagram in AP-form into one with a biadjacency matrix in RREF using Gaussian elimination.

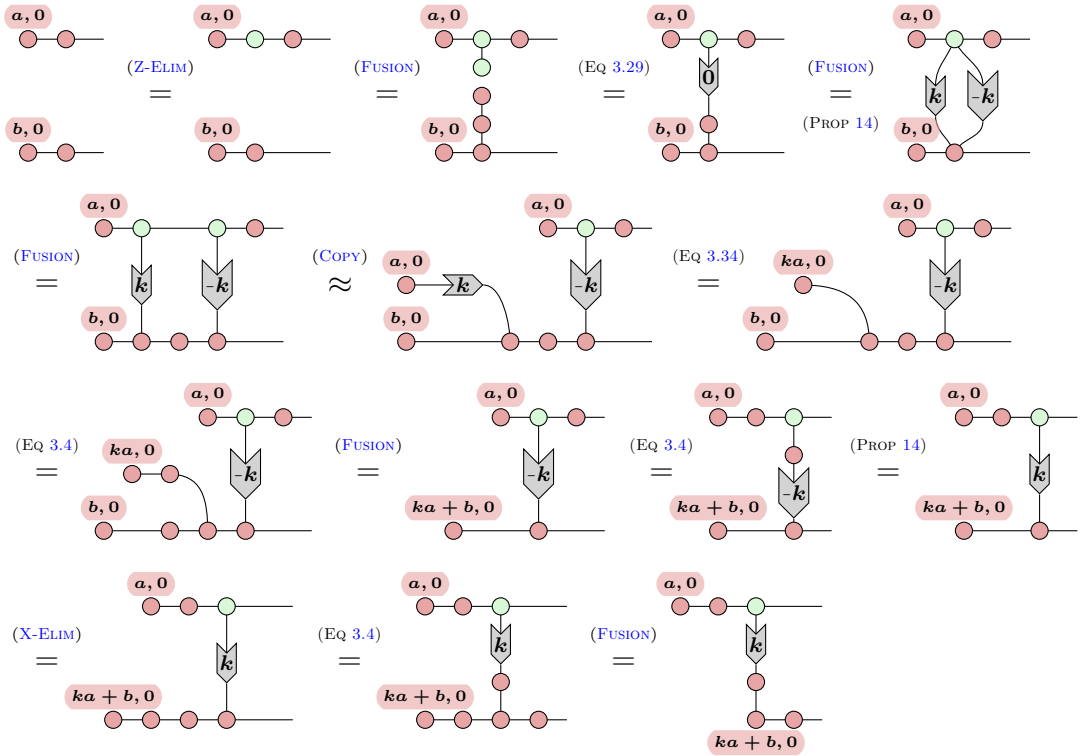
Lemma 27. *We can perform primitive row operations on a ZX-diagram in AP-form, i.e. we can ‘add’ one inner spider to another. For any $k, a, b, e_i, f_j \in \mathbb{Z}_p$*

4.3. COMPLETENESS

where $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\}$:

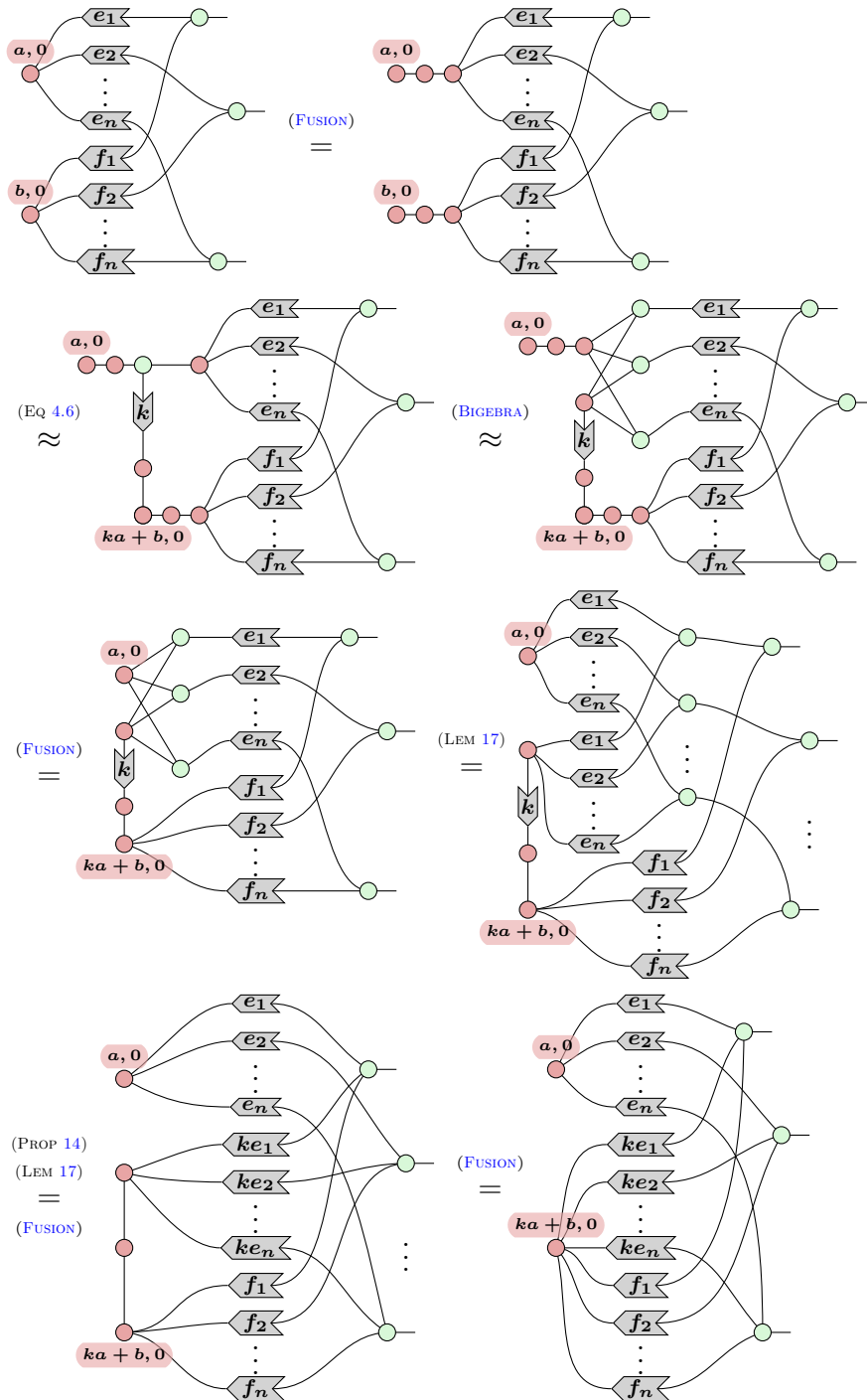


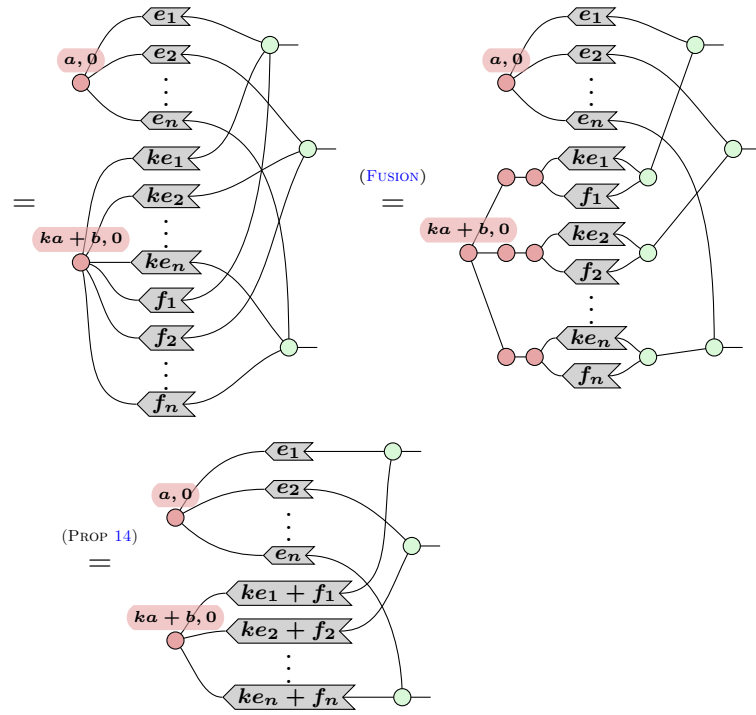
Proof. Firstly, we show that we can transform two disconnected X-states in the following way:



$$(4.6)$$

Then, we can show that we can transform a diagram in AP-form as follows:





□

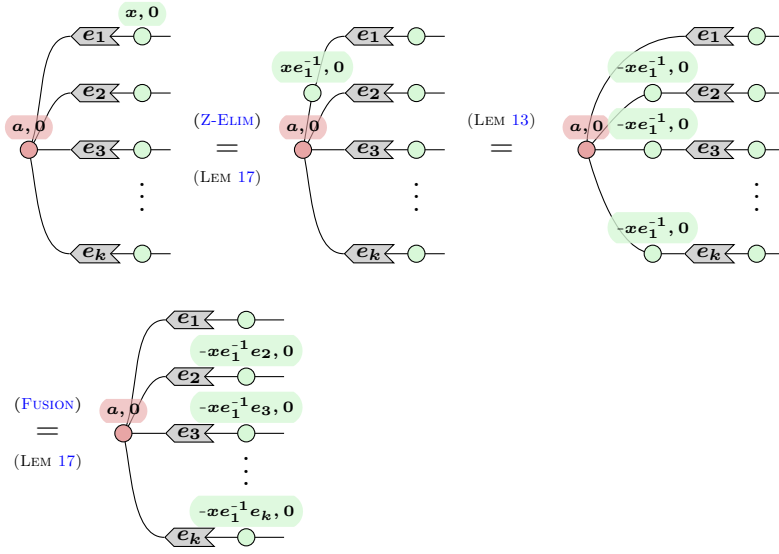
As the above lemma states, we can perform primitive row operations on diagrams in AP-form. This means that we can use Gaussian elimination on diagrams. Thus, we can transform their biadjacency matrix to be in RREF.

Subsequently, we show that we can remove any phase from the pivot spider of a diagram in AP-form. We begin with the case when the phase of the pivot spider is Pauli.

Lemma 28. *We can remove Pauli-phases from the pivot spiders of diagrams in AP-form.*

|

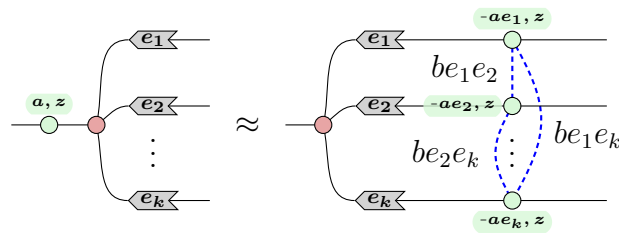
Proof. For any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $e_1 \in \mathbb{Z}_p^*$:



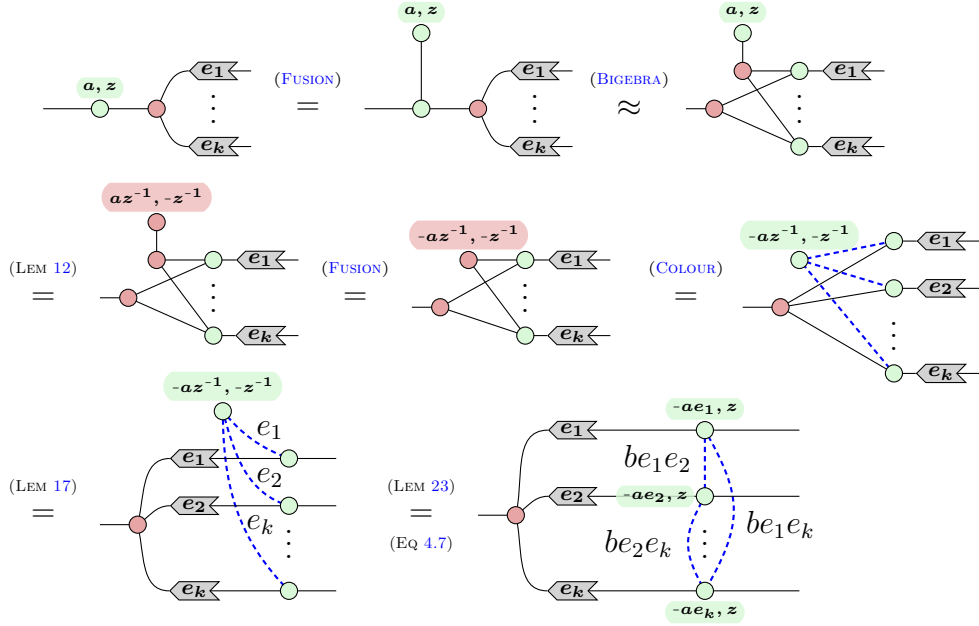
□

We now direct our attention to the elimination of strictly Clifford phases from pivot spiders. To prove this case, we first show that we can push strictly Clifford Z-spider through an X-spider with weighted outputs.

Lemma 29. *We can push a strictly Clifford Z-spider through an X-spider with weighted output legs. That is, for any $a, e_i \in \mathbb{Z}_p$ where $i \in \{1, \dots, k\}$ and $z \in \mathbb{Z}_p^*$:*



Proof.



Note that the phase after the application of the local complementation comes from the following equation:

$$-az^{-1}, -z^{-1} \mapsto -(-az^{-1})(-z^{-1})^{-1}, -(-z^{-1})^{-1} = -az^{-1}z, z = -a, z \quad (4.7)$$

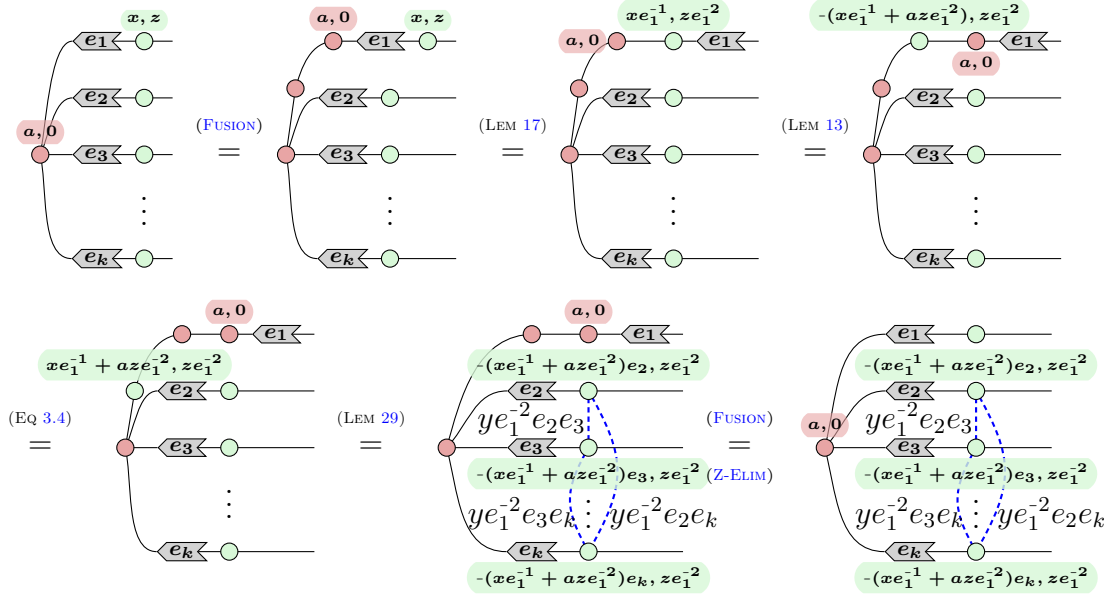
□

Using the lemma proved above, we show that non-Pauli phases from pivots spiders can be removed.

Lemma 30. *We can remove strictly Clifford phases from the pivot spiders of diagrams in AP-form.*

|

Proof. For any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $z, e_1 \in \mathbb{Z}_p^*$:



□

Combining the results of Lemma 28 and Lemma 30, we conclude that any phase can be removed from a pivot spider.

Subsequently, we show how to remove H-edges between the pivot spider and other boundary spiders. We have two case distinctions based on whether the H-box connects to a spider that is connected to the same internal spider as the pivot, or not. However, most parts of the proofs are identical in the two cases. In order to reduce repetition, we prove a lemma that generalises the identical parts of the proofs.

Lemma 31. For any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $e_1 \in \mathbb{Z}_p^*$ the following equation holds:

Proof. Firstly, we focus on transforming a specific part of the diagram of in-

terest.

(4.9)

Then, we can use the previous equation to prove the lemma.

(4.10)

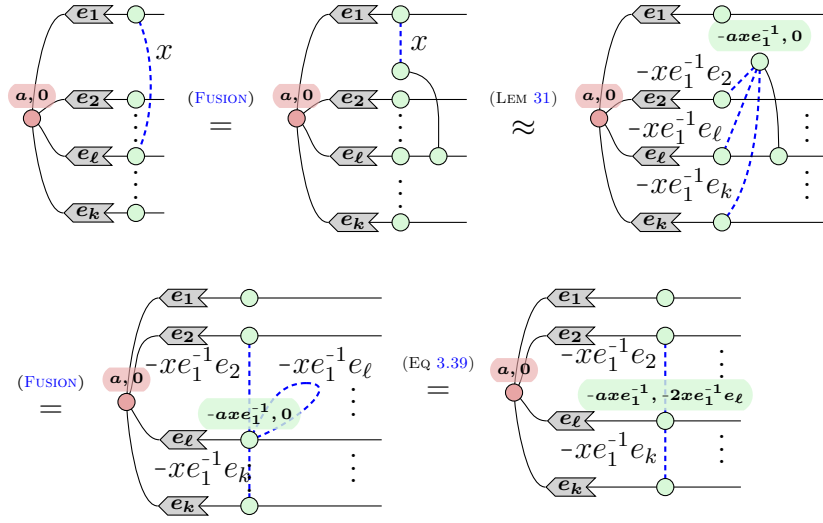
□

Using the lemma proved above, we show that H-edges that are connected to the

pivot spider can be removed.

Lemma 32. *We can remove an H-edge between the pivot spider and a boundary spider that connects to the same internal spider as the pivot.*

Proof. Let us suppose that the pivot spider is connected to the ℓ -th wire with an H-box. Then, for any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $e_1 \in \mathbb{Z}_p^*$:

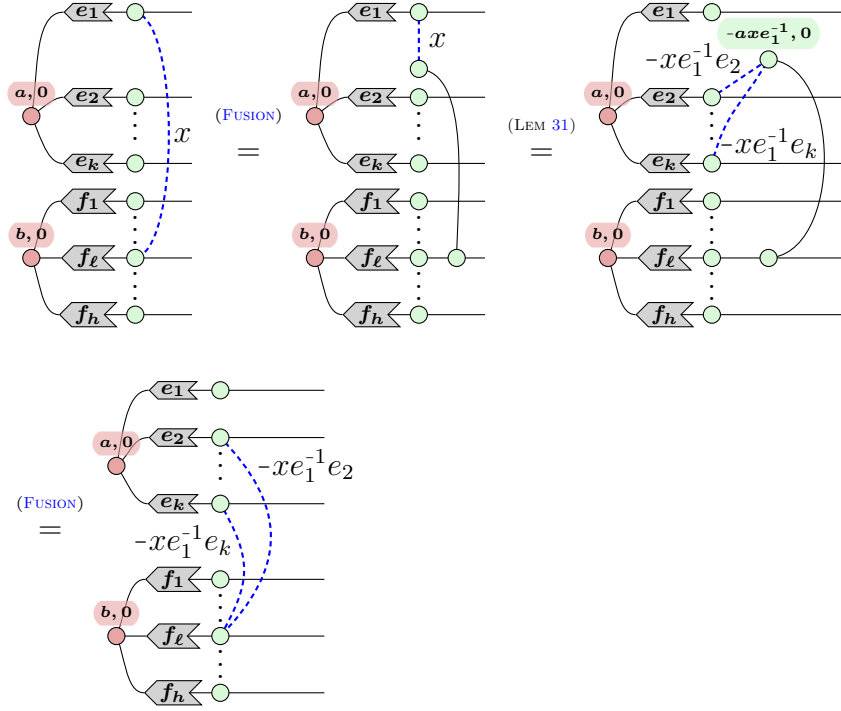


□

Lemma 33. *We can remove an H-edge between the pivot spider and a boundary spider that does not connect to the same internal spider as the pivot.*

Proof. For any $a, b, x, e_i, f_h \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$, $h \in \{1, \dots, j\}$ and

$e_1 \in \mathbb{Z}_p^*$:



□

Therefore, we can remove H-boxes connected to the pivot spider.

We conclude the previous proofs with the following lemma:

Lemma 34. *Any diagram in ZX_p^{Stab} can be converted into one in reduced AP-form.*

Proof. First, we can convert any diagram in ZX_p^{Stab} into one in AP-form using local complementation and pivoting. Then, such a diagram can be translated into one in AP-form with a biadjacency matrix in RREF using Gaussian elimination which can be performed because of Lemma 27. We have also proved that any phase can be removed from the pivot spider thanks to Lemma 28 and Lemma 30. Lastly, we can also remove any H-edge connected to the pivot spider using Lemma 32 and Lemma 33. This enables us to transform a diagram in such a way that its phase function ϕ only contains free variables from the equation system $E\vec{x} = \vec{a}$. In conclusion, we are able to rewrite any diagram in

ZX_p^{Stab} in such a way that it satisfies the properties to be a diagram in reduced AP-form. \square

Ultimately, we can prove the completeness of ZX_p^{Stab} using the proofs discussed above.

Theorem 35. *For any pair of ZX-diagrams $A, B \in ZX_p^{\text{Stab}}$, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, then we can provide a sequence of rewrites that transforms A into B .*

Proof. Without loss of generality, we can assume that A and B are states by map-state duality, that is, we can bend the wires in order to make the diagram form a state. If A and B represent the same linear map, i.e. $\llbracket A \rrbracket = \llbracket B \rrbracket$, then their reduced AP-form is identical thanks to the uniqueness of the form proved in Lemma 26. Therefore, we can transform both A and B into diagrams in reduced AP-form because of Lemma 34. The sequence of transformations from A to A in reduced AP-form composed with the series of rewrites from B in reduced AP-form to B provides us with a sequence of rewrites that transforms A into B . \square

Chapter 5

Conclusion

To sum up, we have presented a modified version of the qudit ZX-calculus for odd prime dimensions of Ref. [2], which includes explicit scalars. Using this calculus, we have shown the qudit version of local complementation and pivoting. We have also presented the qudit version of the AP-form and its unique version, the reduced AP-form. Then, using the AP-form, we have demonstrated how to efficiently weakly simulate Clifford circuits. Lastly, using the reduced AP-form we have also proved that the qudit ZX-calculus for odd prime dimensions is complete.

5.1 Evaluation of the thesis

Our aim was to present a completeness proof that is more accessible and easier to implement compared to the proof of Ref. [2]. Firstly, the completeness proof we present in the thesis is completely constructive and is based on methods like Gaussian elimination. Therefore, we provide clear steps for someone who aims to implement automatic rewrites of Clifford diagrams into one in AP-form or reduced AP-form. Moreover, the applications related to the AP-forms are also implementable as is, without any further development of the theory. On the other hand, it is complicated to reason about the simplicity or understandability of a proof. We may assume that it is easier to grasp the completeness proof of the

thesis for students familiar with the proof presented in the Quantum Software course. However, it might be the case that people familiar with the completeness proof of Ref. [33] would find the proof of Ref. [2] more accessible. Nevertheless, we present each proof in the thesis as simply and intuitively as the author's imagination allowed with many simplifications and sub-lemmas.

The other aspect we may consider with caution is the axiomatisation with explicit scalars. On the one hand, it requires several additional rules in the axioms which means that our calculus is further from one that is minimal. On the other hand, it increases the simplicity of proofs and equations substantially, and can generally improve the comprehensibility of the graphical rewrite rules. It is up to one's preferences what they believe is more important, minimality or comprehensibility.

We also note that while researching the calculus and recalculating the proofs of Ref. [2], several typos in the text and errors in the calculations have been found. These notices have been communicated to the authors of the paper.

One of the main results of the thesis is the general version of the weak simulation algorithm presented in Ref. [38] which is presented for odd prime dimensions in Section 4.2. The other main result is the alternative proof of completeness for the Clifford fragment of the qudit ZX-calculus for odd prime dimensions which is presented in Section 4.3. Since both of our main results rely on local complementation (Lemma 23) and pivoting (Lemma 25), one may suggest that the most valuable contributions of the thesis could be these rewrite rules as we prove our final results using them. The demonstration of the above-mentioned graphical rewrite rules may also enable future researchers to investigate the qudit versions of other proofs and algorithms that rely on them.

5.1.1 Limitations

The first limitation we note is that the number of rules needed in order to represent the scalars explicitly, as part of the calculus, is larger than one may expect. This is because we needed several rules to translate the empty diagram, the zero scalar, and all the elementary scalars in order to make the scalar fragment complete.

Another limitation is the fact that the calculus is only presented for odd prime dimensions. Firstly, this is because the construction does not work for the qubit case, so we have to omit the $d = 2$ case, i.e. the even prime case. More importantly, we need p to be a prime in order for \mathbb{Z}_p to be a field. This allows us to use the multiplicative inverse of every element of \mathbb{Z}_p^* which we use extensively throughout the proofs.

Another limitation is that we only present the completeness of the Clifford fragment of the calculus. However, we can prove that the Clifford+T and also the universal fragment of the qubit ZX-calculus are complete. Nonetheless, the proof of such results would be highly non-trivial even if we know how the proof works for less general cases.

Further to the completeness result, the thesis does not discuss the qudit version of algorithms that rely on local complementation and pivoting. These two rules are used in many papers, and one may expect it not to be difficult to prove and implement such methods for the qudit case. Nevertheless, the aim of this thesis is not to provide the qudit version of algorithms, but to provide an overview of the AP-form and its applications for qudits.

Lastly, it is worth noting that we originally aimed to use a slightly modified interpretation of spiders compared to that of Ref. [2]. We intended to omit the 2^{-1} factor from the power of ω resulting in ω^{xk+yk^2} instead of $\omega^{2^{-1}(xk+yk^2)}$. However, this modification changed our axiomatisation; more specifically, it changed [GAUSS](#), [SHEAR](#), and [MULT](#). Eventually, the changed axioms resulted in more

complicated rules that were undesired for our aims to provide simpler proofs. Also, using a different interpretation would mean that one could not use the results of both papers. Therefore, we decided that using the same interpretation as Ref. [2] is a wise decision.

5.2 Future work

Firstly, the most obvious recommendation for future work would be to define a calculus for not only odd prime dimensions but arbitrary ones. This would probably include the combination of the prime-dimensional calculi in such a way that they represent the given dimension. However, the design of such a calculus is non-trivial, and it may be problematic to provide rules that require the existence of the multiplicative inverse of a given number.

Another reasonable work could be to define the calculus for and prove the completeness of the Clifford+T fragment for odd prime-dimensional qudits. The interpretation of the spiders in such a system would be achievable using a third phase that represents the cubic part in the power of ω . A path to prove completeness could be to examine and use the method of previous papers dealing with the completeness of the Clifford+T fragment in less general cases.

Based on the completeness result for the Clifford+T fragment, a possible path would be to prove that the universal fragment of the qudit ZX-calculus is complete for odd prime dimensions. Finally, by combining the results of the works mentioned above one could prove the completeness of the qudit ZX-calculus for the universal fragment.

Another suggestion for future work would be the further analysis of the local complementation and pivoting for odd prime qudits. One would expect that most algorithms that rely on the rewrite rules mentioned above would be adaptable for the qudit case without much complication. One example could be to design

T-count reducing algorithms for qudits circuits based on the works of Ref. [29].

The last recommendation for future work would be the implementation of a tool to perform automated rewriting for the qudit ZX-calculus, similarly to PyZX [44]. One possibility would be to fork the existing PyZX library, which is a Python tool that implements the theory of ZX-calculus for the creation, visualisation, and automated rewriting of large-scale quantum circuits. This forked library could be modified to implement the qudit version of the ZX-calculus. The drawback of this method is that the fixes and updates to PyZX would not appear in the new library. Alternatively, one could modify PyZX itself to support qudits; however, that would mean that each new contribution would also have to be implemented for qudits, which is undesirable for most. Another option could be to implement some kind of extension that overrides the core of PyZX in such a way that most functions work as originally intended while also implementing the qudit ZX-calculus. Nevertheless, this method seems to involve the most complexity, and it may not be implementable without the modification of PyZX.

Finally, we hope that the contributions of this thesis will be found useful in the community and that they will provide researchers with methods or ideas to prove further and yet more interesting ideas in the ZX-calculus.

Bibliography

- [1] John van de Wetering. *ZX-calculus for the working quantum computer scientist*. 27th Dec. 2020. URL: <http://arxiv.org/abs/2012.13966>.
- [2] Robert I. Booth and Titouan Carette. *Complete ZX-calculi for the stabiliser fragment in odd prime dimensions*. 6th July 2022. DOI: [10.48550/arXiv.2204.12531](https://doi.org/10.48550/arXiv.2204.12531).
- [3] M. Mitchell Waldrop. ‘The chips are down for Moore’s law’. In: *Nature News* 530.7589 (11th Feb. 2016), p. 144. DOI: [10.1038/530144a](https://doi.org/10.1038/530144a).
- [4] P.W. Shor. ‘Algorithms for quantum computation: discrete logarithms and factoring’. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. DOI: [10.1109/SFCS.1994.365700](https://doi.org/10.1109/SFCS.1994.365700).
- [5] Richard P. Feynman. ‘Simulating physics with computers’. In: *International Journal of Theoretical Physics* 21.6 (1st June 1982), pp. 467–488. ISSN: 1572-9575. DOI: [10.1007/BF02650179](https://doi.org/10.1007/BF02650179).
- [6] Daniel Gottesman. ‘The Heisenberg representation of quantum computers’. In: *22nd International Colloquium on Group Theoretical Methods in Physics*. July 1998, pp. 32–43. URL: <https://arxiv.org/abs/quant-ph/9807006>.
- [7] Charles H. Bennett and Stephen J. Wiesner. ‘Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states’. In: *Physical Review Letters* 69.20 (16th Nov. 1992), pp. 2881–2884. DOI: [10.1103/PhysRevLett.69.2881](https://doi.org/10.1103/PhysRevLett.69.2881).
- [8] Charles H. Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres and William K. Wootters. ‘Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels’. In: *Physical Review Letters* 70.13 (29th Mar. 1993), pp. 1895–1899. DOI: [10.1103/PhysRevLett.70.1895](https://doi.org/10.1103/PhysRevLett.70.1895).
- [9] Charles H. Bennett and Gilles Brassard. ‘Quantum cryptography: Public key distribution and coin tossing’. In: *Theoretical Computer Science*. Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84 560 (4th Dec. 2014), pp. 7–11. ISSN: 0304-3975. DOI: [10.1016/j.tcs.2014.05.025](https://doi.org/10.1016/j.tcs.2014.05.025).
- [10] Yuchen Wang, Zixuan Hu, Barry C. Sanders and Sabre Kais. ‘Qudits and High-Dimensional Quantum Computing’. In: *Frontiers in Physics* 8 (2020). ISSN: 2296-424X. DOI: [10.3389/fphy.2020.589504](https://doi.org/10.3389/fphy.2020.589504).

- [11] Earl T. Campbell. ‘Enhanced Fault-Tolerant Quantum Computing in d-Level Systems’. In: *Physical Review Letters* 113.23 (3rd Dec. 2014), p. 230501. DOI: [10.1103/PhysRevLett.113.230501](https://doi.org/10.1103/PhysRevLett.113.230501).
- [12] Daniele Cozzolino, Beatrice Da Lio, Davide Bacco and Leif Katsuo Oxenløwe. ‘High-Dimensional Quantum Communication: Benefits, Progress, and Future Challenges’. In: *Advanced Quantum Technologies* 2.12 (2019), p. 1900038. ISSN: 2511-9044. DOI: [10.1002/qute.201900038](https://doi.org/10.1002/qute.201900038).
- [13] Martin Ringbauer et al. ‘A universal qudit quantum processor with trapped ions’. In: *Nature Physics* (21st July 2022), pp. 1–5. ISSN: 1745-2481. DOI: [10.1038/s41567-022-01658-0](https://doi.org/10.1038/s41567-022-01658-0).
- [14] Pavel Hrmo et al. *Native qudit entanglement in a trapped ion quantum processor*. 8th June 2022. DOI: [10.48550/arXiv.2206.04104](https://doi.org/10.48550/arXiv.2206.04104).
- [15] Yulin Chi et al. ‘A programmable qudit-based quantum processor’. In: *Nature Communications* 13.1 (4th Mar. 2022), p. 1166. ISSN: 2041-1723. DOI: [10.1038/s41467-022-28767-x](https://doi.org/10.1038/s41467-022-28767-x).
- [16] M. S. Blok et al. ‘Quantum Information Scrambling on a Superconducting Qutrit Processor’. In: *Physical Review X* 11.2 (9th Apr. 2021), p. 021010. DOI: [10.1103/PhysRevX.11.021010](https://doi.org/10.1103/PhysRevX.11.021010).
- [17] Biaoliang Ye, Zhen-Fei Zheng, Yu Zhang and Chui-Ping Yang. ‘Circuit QED: single-step realization of a multiqubit controlled phase gate with one microwave photonic qubit simultaneously controlling n - 1 microwave photonic qubits’. In: *Optics Express* 26.23 (12th Nov. 2018). Publisher: OSA, pp. 30689–30702. DOI: [10.1364/OE.26.030689](https://doi.org/10.1364/OE.26.030689).
- [18] M. A. Yurtalan, J. Shi, M. Kononenko, A. Lupascu and S. Ashhab. ‘Implementation of a Walsh-Hadamard Gate in a Superconducting Qutrit’. In: *Physical Review Letters* 125.18 (27th Oct. 2020), p. 180504. DOI: [10.1103/PhysRevLett.125.180504](https://doi.org/10.1103/PhysRevLett.125.180504).
- [19] Alexander D. Hill, Mark J. Hodson, Nicolas Didier and Matthew J. Reagor. *Realization of arbitrary doubly-controlled quantum phase gates*. 3rd Aug. 2021. DOI: [10.48550/arXiv.2108.01652](https://doi.org/10.48550/arXiv.2108.01652).
- [20] Bob Coecke and Ross Duncan. ‘Interacting quantum observables: categorical algebra and diagrammatics’. In: *New Journal of Physics* 13.4 (2011), p. 043016. ISSN: 1367-2630. DOI: [10.1088/1367-2630/13/4/043016](https://doi.org/10.1088/1367-2630/13/4/043016).
- [21] Ross Duncan and Simon Perdrix. ‘Rewriting Measurement-Based Quantum Computations with Generalised Flow’. In: *Automata, Languages and Programming*. Ed. by Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide and Paul G. Spirakis. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 285–296. ISBN: 978-3-642-14162-1. DOI: [10.1007/978-3-642-14162-1_24](https://doi.org/10.1007/978-3-642-14162-1_24).

- [22] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski and John van de Wetering. ‘There and back again: A circuit extraction tale’. In: *Quantum* 5 (25th Mar. 2021), p. 421. DOI: [10.22331/q-2021-03-25-421](https://doi.org/10.22331/q-2021-03-25-421).
- [23] Thomas McElvanney and Miriam Backens. ‘Complete flow-preserving rewrite rules for MBQC patterns with Pauli measurements’. In: *Quantum Physics and Logic* (1st June 2022). DOI: [10.48550/arXiv.2205.02009](https://doi.org/10.48550/arXiv.2205.02009).
- [24] Ross Duncan and Maxime Lucas. ‘Verifying the Steane code with Quantomatic’. In: *Electronic Proceedings in Theoretical Computer Science* 171 (27th Dec. 2014), pp. 33–49. ISSN: 2075-2180. DOI: [10.4204/EPTCS.171.4](https://doi.org/10.4204/EPTCS.171.4).
- [25] Liam Garvie and Ross Duncan. ‘Verifying the Smallest Interesting Colour Code with Quantomatic’. In: *Electronic Proceedings in Theoretical Computer Science* 266 (27th Feb. 2018), pp. 147–163. ISSN: 2075-2180. DOI: [10.4204/EPTCS.266.10](https://doi.org/10.4204/EPTCS.266.10).
- [26] Niel de Beaudrap and Dominic Horsman. ‘The ZX calculus is a language for surface code lattice surgery’. In: *Quantum* 4 (9th Jan. 2020), p. 218. DOI: [10.22331/q-2020-01-09-218](https://doi.org/10.22331/q-2020-01-09-218).
- [27] Niel de Beaudrap, Xiaoning Bian and Quanlong Wang. ‘Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities’. In: *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*. Ed. by Steven T. Flammia. Vol. 158. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 14th Apr. 2020, 11:1–11:23. ISBN: 978-3-95977-146-7. DOI: [10.4230/LIPIcs.TQC.2020.11](https://doi.org/10.4230/LIPIcs.TQC.2020.11).
- [28] Ross Duncan, Aleks Kissinger, Simon Perdrix and John van de Wetering. ‘Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus’. In: *Quantum* 4 (4th June 2020), p. 279. DOI: [10.22331/q-2020-06-04-279](https://doi.org/10.22331/q-2020-06-04-279).
- [29] Aleks Kissinger and John van de Wetering. ‘Reducing T-count with the ZX-calculus’. In: *Physical Review A* 102.2 (11th Aug. 2020), p. 022406. ISSN: 2469-9926, 2469-9934. DOI: [10.1103/PhysRevA.102.022406](https://doi.org/10.1103/PhysRevA.102.022406).
- [30] Bob Coecke, Giovanni de Felice, Konstantinos Meichanetzidis and Alexis Toumi. *Foundations for Near-Term Quantum Natural Language Processing*. 7th Dec. 2020. DOI: [10.48550/arXiv.2012.03755](https://doi.org/10.48550/arXiv.2012.03755).
- [31] Konstantinos Meichanetzidis, Stefano Gogioso, Giovanni de Felice, Nicolò Chiappori, Alexis Toumi and Bob Coecke. ‘Quantum Natural Language Processing on Near-Term Quantum Computers’. In: *Proceedings 17th International Conference on Quantum Physics and Logic*. Ed. by Benoît Valiron, Shane Mansfield, Pablo Arrighi and Prakash Panangaden. Vol. 340. Electronic Proceedings in Theoretical Computer Science. Paris, France: Open Publishing Association, 6th Sept. 2021, pp. 213–229. DOI: [10.4204/EPTCS.340.11](https://doi.org/10.4204/EPTCS.340.11).

- [32] Quanlong Wang and Richie Yeung. *Differentiating and Integrating ZX Diagrams*. 28th Feb. 2022. DOI: [10.48550/arXiv.2201.13250](https://doi.org/10.48550/arXiv.2201.13250).
- [33] Miriam Backens. ‘The ZX-calculus is complete for stabilizer quantum mechanics’. In: *New Journal of Physics* 16.9 (17th Sept. 2014), p. 093021. ISSN: 1367-2630. DOI: [10.1088/1367-2630/16/9/093021](https://doi.org/10.1088/1367-2630/16/9/093021).
- [34] Emmanuel Jeandel, Simon Perdrix and Renaud Vilmart. ‘A Complete Axiomatisation of the ZX-Calculus for Clifford+T Quantum Mechanics’. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS ’18. New York, NY, USA: ACM, 2018, pp. 559–568. ISBN: 978-1-4503-5583-4. DOI: [10.1145/3209108.3209131](https://doi.org/10.1145/3209108.3209131).
- [35] Kang Feng Ng and Quanlong Wang. *A universal completion of the ZX-calculus*. 29th June 2017. DOI: [10.48550/arXiv.1706.09877](https://doi.org/10.48550/arXiv.1706.09877).
- [36] Quanlong Wang. ‘Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics’. In: *Electronic Proceedings in Theoretical Computer Science* 266 (27th Feb. 2018), pp. 58–70. DOI: [10.4204/EPTCS.266.3](https://doi.org/10.4204/EPTCS.266.3).
- [37] Jeroen Dehaene and Bart De Moor. ‘The Clifford group, stabilizer states, and linear and quadratic operations over $\text{GF}(2)$ ’. In: *Physical Review A* 68.4 (20th Oct. 2003), p. 042318. ISSN: 1050-2947, 1094-1622. DOI: [10.1103/PhysRevA.68.042318](https://doi.org/10.1103/PhysRevA.68.042318).
- [38] Maarten Van den Nest. ‘Classical simulation of quantum computation, the Gottesman-Knill theorem, and slightly beyond’. In: *Quantum Information & Computation* 10.3 (1st Mar. 2010), pp. 258–271. ISSN: 1533-7146. URL: <https://arxiv.org/abs/0811.0898>.
- [39] Benjamin Schumacher. ‘Quantum coding’. In: *Physical Review A* 51.4 (1st Apr. 1995), pp. 2738–2747. ISSN: 1050-2947, 1094-1622. DOI: [10.1103/PhysRevA.51.2738](https://doi.org/10.1103/PhysRevA.51.2738).
- [40] Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes*. Cambridge University Press, 16th Mar. 2017. 847 pp. ISBN: 978-1-107-10422-8.
- [41] S. Abramsky and B. Coecke. ‘A categorical semantics of quantum protocols’. In: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004*. ISSN: 1043-6871. July 2004, pp. 415–425. DOI: [10.1109/LICS.2004.1319636](https://doi.org/10.1109/LICS.2004.1319636).
- [42] Ross Duncan and Simon Perdrix. ‘Graph States and the Necessity of Euler Decomposition’. In: *Mathematical Theory and Computational Practice*. Ed. by Klaus Ambos-Spies, Benedikt Löwe and Wolfgang Merkle. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2009, pp. 167–177. ISBN: 978-3-642-03073-4. DOI: [10.1007/978-3-642-03073-4_18](https://doi.org/10.1007/978-3-642-03073-4_18).
- [43] Titouan Carrette. *When Only Topology Matters*. 5th July 2021. DOI: [10.48550/arXiv.2102.03178](https://doi.org/10.48550/arXiv.2102.03178).

- [44] Aleks Kissinger and John van de Wetering. ‘PyZX: Large Scale Automated Diagrammatic Reasoning’. In: *Electronic Proceedings in Theoretical Computer Science* 318 (1st May 2020), pp. 229–241. ISSN: 2075-2180. DOI: [10.4204/EPTCS.318.14](https://doi.org/10.4204/EPTCS.318.14).