

# Qudit ZH-calculus



Patrick Roy

Linacre College

University of Oxford

A thesis submitted for the degree of

*MSc in Mathematics and Foundations of Computer Science*

Trinity 2022



## Abstract

The use of quantum graphical calculi, such as the ZX-calculus [13], has led to great advances in fields such as quantum circuit optimization. Particularly the ZH-calculus has proven useful for reasoning about higher-level quantum operations, such as Toffoli and Hadamard gates, which form an approximately universal set of gates, for which the qubit ZH-calculus is complete [7]. More recently, quantum computing based on  $d$ -ary basis states has gained traction, with the ZX- and ZW-calculi having been generalized to higher dimensions [26][38]. However, no generalization of the ZH-calculus is known. Such a generalization might lead to new insights about higher-dimensional equivalents of the Toffoli-gate. In this dissertation, we provide this generalization of the qubit ZH-calculus to qudits by generalizing a classification of qubit quantum graphical calculi due to Carette and Jeandel [12] by relaxing a condition that would have excluded a qudit ZH-calculus from being constructed. We then use this generalization to reduce the ruleset of the qubit ZH-calculus by one rule. Additionally, we give a first-of-its-kind algorithm for constructing ZH-diagrams from the matrix representation of linear maps, and use this algorithm to show that, for prime dimension, ZH-calculus is universal.

## Acknowledgements

I would like to thank John van de Wetering, Lia Yeh and my supervisor Aleks Kissinger for many insightful discussions, and for providing me with the L<sup>A</sup>T<sub>E</sub>X templates for typesetting ZH-diagrams. Particularly, John's *ZX-calculus for the working quantum computer scientist* has proven itself an invaluable reference. Additionally I would like to thank Quanlong Wang for incredibly helpful discussions about the connections between ZX and ZH.

I would like to thank Zoe M. for proofreading all my written works since undergrad, including this dissertation.

I would like to thank Melissa M., Oisin M., Joseph D., Lauren H., Charlotte B., Elena G. and Becca R. for being amazing friends.

Most of all I would like to thank Marco C. for being the best thing to happen to me during my time in Oxford.

I would like to thank Baliol and St Hilda's colleges for the food, coffee, and quiet study space.

I would like to thank Taylor Swift for the music to which most of this dissertation was written.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	The mathematical model of quantum computation . . . . .	2
1.2	Quantum graphical calculi . . . . .	3
1.2.1	Properties of quantum graphical calculi . . . . .	4
1.2.2	The ZX-calculus . . . . .	5
1.2.3	The ZH-calculus . . . . .	7
1.2.4	Qudit calculi . . . . .	9
1.3	Related work . . . . .	9
1.4	Structure of this dissertation . . . . .	9
<b>2</b>	<b>A categorical construction of quantum graphical calculi</b>	<b>11</b>
2.1	Categorical foundations of graphical calculi . . . . .	11
2.1.1	Strict monoidal categories . . . . .	11
2.1.2	String diagrams . . . . .	13
2.1.3	Symmetric and compact closed structure . . . . .	14
2.2	Anatomy of quantum graphical calculi . . . . .	16
2.2.1	Monoids and Comonoids . . . . .	16
2.2.2	Frobenius Algebrae and their induced compact structure . . . . .	19
2.3	Compatibility is unnecessary . . . . .	21
2.4	A flexsymmetric qudit ZX-Calculus . . . . .	23
<b>3</b>	<b>Construction of a qudit <math>H</math>-box for <math>d &gt; 2</math></b>	<b>24</b>
3.1	Building blocks of <i>qubit</i> ZH . . . . .	24
3.2	Monoid from generalizing AND . . . . .	25

3.3	Comonoid from guesswork . . . . .	27
3.4	Qudit ZH from scratch . . . . .	29
<b>4</b>	<b>Qudit ZH-calculus</b>	<b>31</b>
4.1	Generators . . . . .	31
4.2	Qudit ZH rules . . . . .	33
4.3	Deriving (ft) and color change . . . . .	35
4.4	Further generalizations . . . . .	36
4.4.1	The ortho rule . . . . .	36
4.4.2	Lemma 2.28 . . . . .	37
4.5	Comparison to the qubit case . . . . .	37
4.6	Derived rules . . . . .	38
<b>5</b>	<b>Universality</b>	<b>44</b>
5.1	Constructing ZH-diagrams for binary matrices in prime dimension . . . . .	44
5.1.1	A model-theoretic description of binary matrices . . . . .	45
5.1.2	An algebraic description of binary matrices . . . . .	46
5.2	Realizing a multiplexer . . . . .	48
5.3	Universality for arbitrary rings . . . . .	49
5.4	Universality for $\mathbb{Z}$ . . . . .	51
5.4.1	The qudit successor gadget . . . . .	51
5.4.2	Negative integers . . . . .	53
5.5	Considerations for $d$ not prime . . . . .	55
5.6	The normalform generalizes the qubit normalform . . . . .	56
<b>6</b>	<b>Conclusion</b>	<b>59</b>
6.1	Novel contributions . . . . .	60
6.2	Future Work . . . . .	60
<b>A</b>	<b>Roots of unity</b>	<b>66</b>
A.1	Introduction . . . . .	66
A.2	Irrelevance of choice of primitive $\zeta$ for quantum graphical calculi . . . . .	67

<b>B</b>	<b>Soundness of qudit ZH rewrite rules</b>	<b>69</b>
<b>C</b>	<b>Gadgets for ZH-diagrams</b>	<b>72</b>
<b>D</b>	<b>Further rewrite rules and completeness considerations</b>	<b>75</b>
D.1	The indexing gadget copies through $Z$ -spiders . . . . .	75
D.2	Multiply, Intro and Average . . . . .	77
D.3	A review of normalform rewriting . . . . .	78
D.4	Ring identities . . . . .	79

# List of Figures

1.1	The controlled NOT ( <b>CNOT</b> ), Hadamard ( <b>H</b> ) and $Z$ -rotation ( <b>Z</b> ( $\alpha$ )) operations	2
1.2	The <b>CNOT</b> , <b>H</b> and <b>Z</b> ( $\alpha$ ) operations in quantum circuit notation . . . . .	2
1.3	The <b>SWAP</b> operation implemented using three <b>CNOTs</b> . . . . .	3
1.4	A spider . . . . .	4
1.5	The flexsymmetry axioms for a spider . . . . .	4
1.6	Generators of the qubit ZX-calculus . . . . .	5
1.7	Translating quantum circuit notation into ZX-diagrams . . . . .	5
1.8	Example of qubit ZX rewrite-rules, where $\approx$ denotes equality up to non-zero scalar	6
1.9	Realization of the Toffoli-gate in qubit ZX . . . . .	7
1.10	Realization of the Toffoli-gate in qubit ZH . . . . .	7
1.11	The core ZH-calculus rewrite rules, as presented by Backens et al [7, Fig. 1] . . .	8
2.1	Examples of simple string diagrams . . . . .	13
2.2	String diagram for the swap map in a strict symmetric category . . . . .	14
2.3	String diagrams of the swap equations in a strict symmetric category . . . . .	15
2.4	String diagrams for the cup and cap in a compact closed category . . . . .	15
2.5	Yanking equations . . . . .	15
2.6	Symmetry equations for cap and cup . . . . .	15
2.7	String diagrams for the commutative monoid equations . . . . .	17
2.8	String diagrams for the cocommutative comonoid equations . . . . .	17
2.9	String diagram for the Frobenius algebra equations . . . . .	19
2.10	The compact structure defined by a commutative Frobenius algebra . . . . .	19
2.11	Redefined spider components . . . . .	21
2.12	Abstract Hopf-algebra rule for interacting Frobenius algebrae . . . . .	22



3.1	AND-gate realization in qubit ZH . . . . .	25
3.2	Multiply box as a generalization of logical AND . . . . .	25
3.3	The multiply monoid . . . . .	25
3.4	The quantum Fourier transform . . . . .	26
3.5	The inverse quantum Fourier transform . . . . .	26
3.6	Two possible definitions of the bottom half of the qudit $H$ -box . . . . .	26
3.7	Realization of the multiply box in qudit ZH . . . . .	26
3.8	Flexsymmetric qudit $H$ -box . . . . .	30
3.9	Harvestman fusion rule for qudit $H$ -boxes . . . . .	30
4.1	Definition of the $X$ -spider as a derived generator . . . . .	32
4.2	Generalized $\neg$ -spiders . . . . .	32
4.3	Realization of Pauli-X in qudit ZH . . . . .	32
4.4	Correspondence between qudit ZH-generators and arithmetic in $\mathbb{Z}/d\mathbb{Z}$ . . . . .	33
4.5	Qubit rewrite rules that need adjustments for the qudit setting . . . . .	37
5.1	Realization of a linear map by specifying its columns as states and feeding them into a multiplexer . . . . .	49
5.2	Labelled $H$ -boxes in qudit $ZH_R$ . . . . .	49
5.3	Qudit normalform of a matrix that has a single $r$ -entry and is filled with 1s otherwise	56
5.4	Realization of a multiplexer in qudit ZH. Here we assume the Pauli-X inputs to be on the left . . . . .	58
D.1	Dit indexing map . . . . .	75

# Chapter 1

## Introduction

Quantum computers provide a powerful alternative to classical computation, often yielding significant computational speedups. The perhaps best known example is Shor’s algorithm for factoring integers [40], of which a large scale implementation would lead to widespread breakage of contemporary encryption mechanisms that rely on the hardness of the factoring problem<sup>1</sup>. Another famous example is Grover’s algorithm for unstructured search [25], a probabilistic algorithm for finding a needle in a haystack in  $\mathcal{O}(\sqrt{N})$  with constant probability independent of the size  $N$  of the haystack<sup>2</sup>. The first quantum algorithm providing a provably exponential speedup over classical computers was the Deutsch-Jozsa algorithm [16]<sup>3</sup>.

Today, quantum computing is still in its infancy, and theoretical research is far ahead of what is practically feasible. Modern quantum computers can only operate on a handful of qubits (the quantum equivalent of a bit) and can only realize operations of limited scale. These limitations arise from the fact that it is very difficult to keep quantum states from degrading over time, meaning the longer a computation keeps going, the more likely it is to result in random noise. This is why we call the current era of quantum computing the *Noisy Intermediate-Scale Quantum* (NISQ) era [37].

In this chapter, we first give a brief introduction to the mathematical model of quantum computation due to Deutsch [17]. Then we introduce *quantum graphical calculi*, which are tools for manipulating and optimizing quantum circuits. Note that we can only give a very brief introduction to quantum computing concepts in this dissertation. We refer to *Picturing Quantum Processes* by Bob Coecke and Aleks Kissinger [14] for an in-depth introduction to quantum computing in general, and to *ZX-calculus for the working quantum computer scientist* by John van de Wetering [48] for an rigorous introduction to the ZX- and ZH-calculi.

We conclude this chapter by describing the goals of this dissertation and discussing some related research.

---

<sup>1</sup>Josza later found an algorithm to solve the *hidden subgroup problem*[31], which subsumes a variety of traditionally intractable problems that underpin modern cryptography

<sup>2</sup>Implications of this algorithm include the decidability of a non-trivial subclass of regular languages in sublinear time, a result due to Aaronson, Schaeffer and Grier [3]

<sup>3</sup>While the existence of exponential speedups might lead one to conjecture that quantum computers can solve NP-complete problems in polynomial time, no such algorithm has been found (although neither has its non-existence been proven) [1]

## 1.1 The mathematical model of quantum computation

The first formal model of a quantum computer was given by David Deutsch in 1985 in the form of a *quantum Turing-machine* [15]. He later refined this model to the simpler *circuit model* [17] that we still use today and which we introduce here. In this section we assume familiarity with working in Hilbert spaces.

A quantum computer's state is stored in *qubits*, whose state is a unit vector (with regards to the standard euclidean scalar product) in the two dimensional vector space  $\mathbb{C}^2$ . We denote the standard basis  $e_1 = (1, 0)^T$ ,  $e_2 = (0, 1)^T$  by  $|0\rangle$  and  $|1\rangle$  and call it the *computational basis*. A  $n$ -qubit system operates on states represented by unit vectors in  $(\mathbb{C}^2)^{\otimes n} \cong \mathbb{C}^{(2^n)}$ . Its computational basis are the vectors  $|b_1\rangle \otimes \dots \otimes |b_n\rangle$  for  $\vec{b} := (b_1, \dots, b_n) \in \{0, 1\}^n$ , which we denote by  $|\vec{b}\rangle$  or  $|b_1 \dots b_n\rangle$  for short.

Since quantum states are represented by unit vectors, the only operations a quantum computer is allowed to execute have to preserve this property, meaning they have to be *unitary*. The most important qubit operations are *controlled NOT*, *Hadamard* and *Z-rotations*, as these are universal for quantum computation.

$$\mathbf{CNOT} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \mathbf{H} := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \mathbf{Z}(\alpha) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

$\alpha \in [0, 2\pi)$

Figure 1.1: The controlled NOT (**CNOT**), Hadamard (**H**) and Z-rotation (**Z**( $\alpha$ )) operations  
Further commonly used operations are *S*- and *T*-gates, which are special cases of Z-rotations,

$$\mathbf{S} := \mathbf{Z}(\pi/2) = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \quad \text{and} \quad \mathbf{T} := \mathbf{Z}(\pi/4) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix},$$

and the *controlled-controlled-NOT* (**CCNOT**, also known as *Toffoli-gate*). The set of operations generated by **CNOT**, **H** and **S** is the *Clifford group*, a set of operations that is efficiently classically simulatable [2], yet still represents many important quantum protocols such as quantum teleportation [48, Sec. 5.4], various cryptography algorithms for secure key exchange [28] and quantum error correction algorithms [24]. If we adjoin the **T**-gate to this group, we get the *Clifford+T group*, which is approximately universal [22], meaning every quantum operation can be approximate within an arbitrarily small margin of error. Another approximately universal set of operations is the Toffoli+H set [39].

Since writing out a quantum operation as its unitary matrix is very cumbersome (the matrices' dimensions are exponential in the number of qubits), quantum operations are often written in *quantum circuit notation*. A quantum circuit is a visual representation of a sequence of operations on a fixed number of qubits. Each qubit is represented by a wire, with operations read left-to-right.

$$\mathbf{CNOT} := \begin{array}{c} \bullet \\ | \\ \oplus \\ | \end{array} \quad \mathbf{H} := \boxed{H} \quad \mathbf{Z}(\alpha) := \boxed{Z_\alpha} \quad (1.1)$$

Figure 1.2: The **CNOT**, **H** and **Z**( $\alpha$ ) operations in quantum circuit notation

However, while quantum circuit notation provides a convenient way of describing quantum operations, it does not allow for any manipulation of quantum circuits, as it does not provide a set of rewrite rules. Proving identities such as

$$\begin{array}{c} \bullet \oplus \bullet \\ | \quad | \quad | \\ \oplus \bullet \oplus \end{array} = \begin{array}{c} \diagup \quad \diagdown \\ \diagdown \quad \diagup \end{array} \quad (1.2)$$

Figure 1.3: The **SWAP** operation implemented using three **CNOT**s

requires to evaluate the underlying matrices and checking for equality.

Yet, equipping quantum circuit notation with an equational theory makes us runs into the problem that we are only allowed to express equations between unitary operations (potentially with post-selections on specific qubits), a very restricted class. Quantum graphical calculi, which we introduce in the next section, work around this limitation by instead providing an equational theory for the entire class of linear maps. The downside of this is that diagrams cannot be readily interpreted as quantum circuits anymore, even if they represent a unitary operation. The process of turning arbitrary diagrams back into quantum circuits is called *circuit extraction* and an active area of research [20].

## 1.2 Quantum graphical calculi

In this section we introduce the concept of quantum graphical calculi as a generalization of quantum circuit notation. While the foundational building blocks of quantum circuit notation are specific unitary operations, quantum graphical calculi break these unitary operations into sequences of linear operations. Thus, quantum graphical calculi provide a way to reason about general linear maps. The second difference to quantum circuit notation is that diagrams are often laid out bottom-to-top instead of left-to-right.

There are three main properties that are always investigated first for quantum graphical calculi (and, in fact, for any equational theory):

- **Soundness:** Given an equality derived using the rules of the calculus, does it still hold when interpreted as an equality of matrices?
- **Universality** for a class  $\mathcal{L}$  of linear maps: Given an arbitrary linear map  $L \in \mathcal{L}$ , can we find a graphical representation of it?
- **Completeness** for a class  $\mathcal{L}$  of linear maps: Given an arbitrary equation involving linear maps in  $\mathcal{L}$  (e.g.  $\mathbf{CNOT} \circ \mathbf{CNOT}^\dagger \circ \mathbf{CNOT} = \mathbf{SWAP}$ ), can we derive it graphically (e.g. can we find a series of rewrite steps transforming the graphical representation of the LHS into the graphical representation of the RHS)?

Clearly, any useful equational theory has to be sound. Universality and completeness on the other hand often require significant work to be proven.

### 1.2.1 Properties of quantum graphical calculi

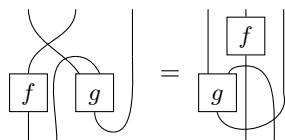
Quantum graphical calculi allow us to reason about the interaction between two or more families of *spiders*, which are special linear maps usually represented by colored dots (the name derives from the wires connecting the dots looking like legs):



Figure 1.4: A spider

There are three fundamental properties quantum graphical calculi generally have in common: The *only connectivity matters* (ocm) paradigm [14, Cor. 3.5], flexsymmetry of spiders [12, Def. 6] and fusion of spiders [13, Thm. 1]. Furthermore, *bialgebra*-style rules are considered a central component [12].

The ocm paradigm is a metarule for reasoning about diagrams. It states that wires can be deformed at will, as long as their connectivity with non-wire objects remains unchanged. Additionally, non-wire objects can be placed at arbitrary locations in the diagram, as long as their connectivity to wires remains unchanged. For example, the following diagrams have the same interpretation, since they only differ in the way wires are laid out:



Flexsymmetry states that any spider is invariant under swapping legs and additionally allows bending them, turning inputs into outputs and vice versa:

$$\begin{array}{c} \dots \\ \dots \end{array} \begin{array}{c} \dots \\ \dots \end{array} = \begin{array}{c} \dots \\ \dots \end{array} \begin{array}{c} \dots \\ \dots \end{array} \quad \text{and} \quad \begin{array}{c} \dots \\ \dots \end{array} \begin{array}{c} \dots \\ \dots \end{array} = \begin{array}{c} \dots \\ \dots \end{array} \begin{array}{c} \dots \\ \dots \end{array} \quad (1.3)$$

Figure 1.5: The flexsymmetry axioms for a spider

Lastly, fusion allows us to merge spiders of the same kind whenever they are connected via their legs. There are two kinds of fusion rules:

- i) Spider fusion, where two spiders are directly connected via one (or more) legs, e.g.

$$\begin{array}{c} \overbrace{\dots}^n \\ \dots \\ \underbrace{\dots}_m \end{array} \begin{array}{c} \overbrace{\dots}^{\tilde{n}} \\ \dots \\ \underbrace{\dots}_{\tilde{m}} \end{array} = \begin{array}{c} \overbrace{\dots}^{n+\tilde{n}} \\ \dots \\ \underbrace{\dots}_{m+\tilde{m}} \end{array}$$

- ii) Harvestman fusion, where the spiders need to be connected via some intermediary node,



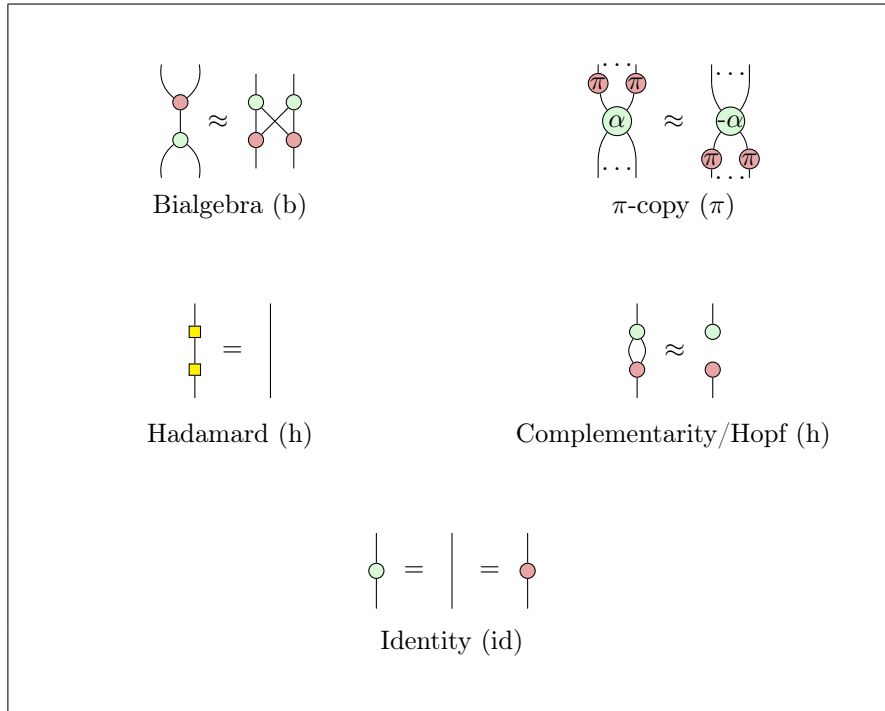
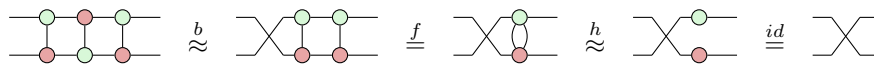


Figure 1.8: Example of qubit ZX rewrite-rules, where  $\approx$  denotes equality up to non-zero scalar

The ruleset for rewriting ZX-diagrams is not generally agreed upon, and different rulesets yield complete calculi for different subclasses of linear maps: Backens first provided a ZX-calculus that is complete for the Clifford fragment [6], Jeandel et al provide a version complete for Clifford+T [29], which they later modify to be complete for quantum mechanics in its entirety [30]. However, beyond the ocm paradigm, flexsymmetry of Z- and X-spiders, and the following variant of the spider fusion rule (which holds unchanged for X-spiders)

$$\begin{array}{c} \overbrace{\dots}^n \\ \circ \\ \underbrace{\dots}_m \end{array} \begin{array}{c} \overbrace{\dots}^{\tilde{n}} \\ \circ \\ \underbrace{\dots}_{\tilde{m}} \end{array} = \begin{array}{c} \overbrace{\dots}^{n+\tilde{n}} \\ \circ \\ \underbrace{\dots}_{m+\tilde{m}} \end{array} \quad \text{where } \delta := \alpha + \beta \pmod{2\pi},$$

most versions of qubit ZX use a ruleset subsuming the rules of Figure 1.8. The rewrite rules of the ZX-calculus make proving the swap identity from 1.2 almost trivial [35]:



Note how in our translation from quantum circuit notation we have failed to give a ZX-diagram for the Toffoli gate. This was with good reason, as a representation of the Toffoli as a ZX-diagram is quite involved [14, p. 694]:

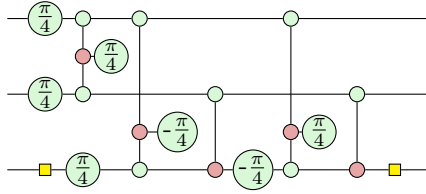


Figure 1.9: Realization of the Toffoli-gate in qubit ZX

The main difficulty arises from the fact that the Toffoli-gate contains logical conjunction of its two control wires. Conjunction, however, is not a reversible map, meaning its implementation as a unitary operation is challenging. Since the ZX-calculus directly generalizes the basic unitary operators of quantum circuit notation, this challenge continues to be an obstacle. The ability to give a more compact representation of intermediary-level gates such as the Toffoli gate was one of the main motivations behind the development of the ZH-calculus, which we introduce next.

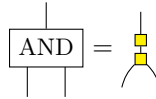
### 1.2.3 The ZH-calculus

While the ZX-calculus uses spiders that generalize specific rotation operations, the ZH-calculus only uses the phaseless version of the Z-spider, and pairs it with a spider that generalizes the hadamard operation:

$$\underbrace{\begin{array}{c} \overbrace{\left( \begin{array}{c} \dots \\ \dots \end{array} \right)}^n \\ \square \\ \underbrace{\left( \begin{array}{c} \dots \\ \dots \end{array} \right)}_m \end{array}} := \frac{1}{\sqrt{2}} \sum_{\substack{i_1, \dots, i_m \in \{0,1\} \\ j_1, \dots, j_n \in \{0,1\}}} (-1)^{i_1 \dots i_m \cdot j_1 \dots j_n} |j_1 \dots j_n\rangle \langle i_1 \dots i_m|.$$

Writing out the matrix for this linear map, we see that it is filled with 1-entries everywhere but the bottom-right corner, where we have a  $(-1)$ -entry.

The  $H$ -box allows for the following simple realization of the AND-gate



and from this the following realization of the Toffoli-gate



Figure 1.10: Realization of the Toffoli-gate in qubit ZH

The ZH-calculus is complete for the approximately universal Toffoli+H set of operations [7][39], which corresponds to matrices with entries in  $\mathbb{Z}[1/\sqrt{2}]$  [4]. We provide its rewrite rules in Figure 1.11.



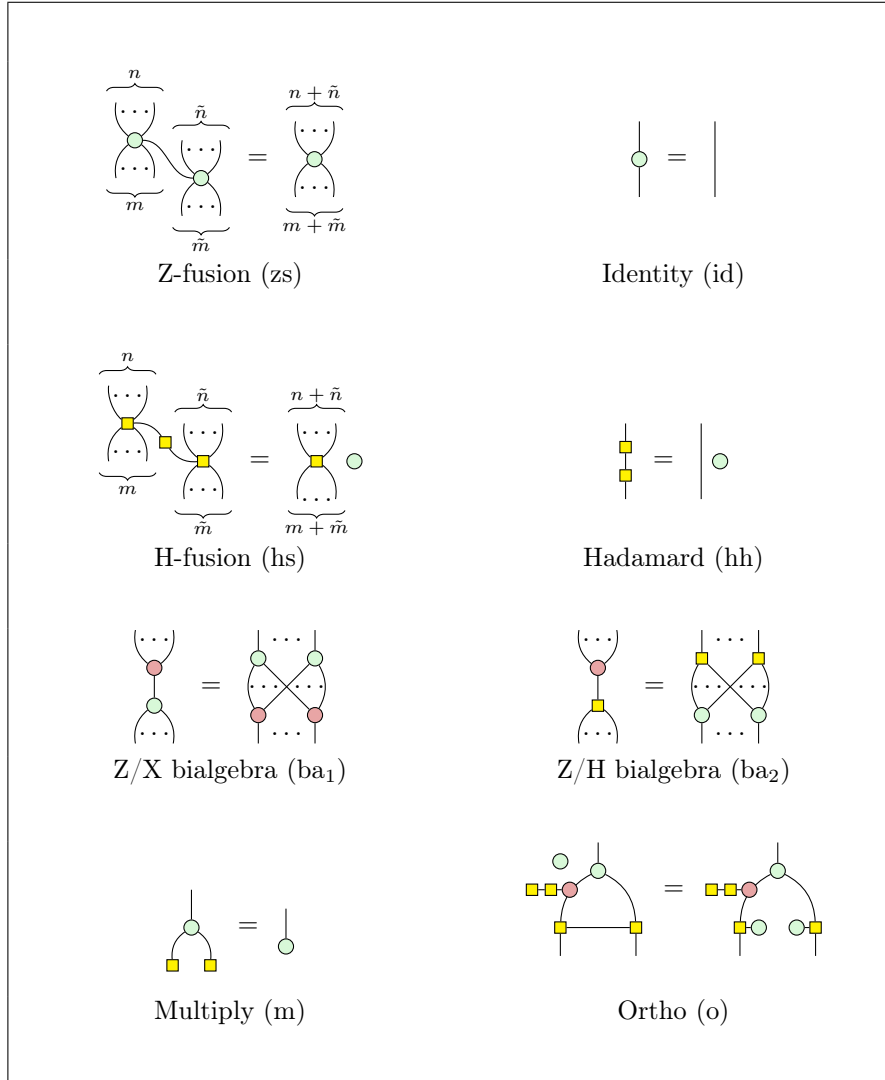


Figure 1.11: The core ZH-calculus rewrite rules, as presented by Backens et al [7, Fig. 1]

### 1.2.4 Qudit calculi

An active area of current research is quantum computing based on *qudits* instead of qubits, meaning quantum computers whose computational basis is the standard basis of the Hilbert space  $\mathbb{C}^d$ . Researchers hope that, by utilizing these higher dimensional states as intermediaries during computation, circuits that would classically require ancilla qubits to store information can be realized without ancillae by instead storing the additional data in qudit states. Additionally, it could give insights about higher-dimensional generalizations of multi-controlled gates such as the Toffoli, although currently there is no characterization of these gatesets in terms of matrix-rings [22].

For this, we seek qudit versions of the mainstream quantum graphical calculi. Ideally, these would behave similarly to their qubit counterparts, e.g. be flexsymmetric, allow some sort of bialgebra rule, obey the ocm paradigm and have either traditional or harvestman fusion of spiders. Carette and Jeandel provide a categorical construction of abstract calculi satisfying these conditions [12], using which they were able to construct qudit versions of the ZX- and ZW-calculi. However, they note that any attempt at generalizing the ZH-calculus to higher dimensions ran into complications [12, Sec. 3.2].

## 1.3 Related work

Research into qudit quantum graphical calculi has been sparse. The first paper venturing beyond qubit mechanics was published in 2014, when Ranchin constructed a non-flexsymmetric qudit ZX-calculus [38], which he proved to be universal. Independently, Wang and Bian developed a (also not flexsymmetric) universal qutrit ZX-calculus [47], which Wang later proved to be complete for stabilizer quantum mechanics (e.g., the Clifford fragment) [46]. Wang also provides a framework for combining qudit ZX-calculi of different dimensions, which he calls the *qufinite ZX-calculus* [45].

The first qudit ZW-calculus was presented by Hadzihasanovic in 2017 as part of his DPhil dissertation, which also introduced qubit ZW-calculus [26]. Wang later simplified this qudit ZW-calculus [42] and provides a qufinite version.

Van de Wetering and Yeh use a flexsymmetric variation of Wang’s qutrit ZX-calculus to investigate qutrit phase gadgets [49]. They also use quantum circuit notation to construct qutrit controlled Clifford+T unitaries in the Clifford+T fragment [50].

To our knowledge, no published work covers a qudit generalization of the ZH-calculus.

## 1.4 Structure of this dissertation

In this dissertation, we construct a qudit ZH-calculus and prove that for  $d$  prime it is universal for matrices with integer entries.

Chapter 2 first provides the necessary background in category theory and briefly recaps Carette and Jeandel’s recipe for constructing quantum graphical calculi. It then generalizes the construction slightly by relaxing a condition that would have excluded a qudit ZH-calculus from being constructed. In Chapter 3 we then construct the main component of our qudit ZH-calculus, the

generalization of the  $H$ -box. After this, we go on to describe a set of rewrite rules in Chapter 4 and derive some useful identities. Then, we prove our universality result in Chapter 5. Chapter 6 concludes and discusses some potential future areas of research.

Appendix A provides some background on the algebraic theory of roots of unity required to understand the qudit constructions of Chapter 3. Appendix B proves the soundness of the rewrite rules of our qudit ZH-calculus. Lastly, Appendix C provides some gadgets useful for constructing ZH-diagrams. Appendix D covers some completeness considerations and additional derived rules.

## Chapter 2

# A categorical construction of quantum graphical calculi

Over time, multiple variations of ZX, ZH and ZW have been proposed, mostly differing in the normalization of the generators. In 2020, Carette and Jeandel [12] provided a classification of flexsymmetric graphical calculi satisfying (a version of) the bialgebra rule as introduced in Section 1.2. Their main result was that, for qubits, ZX, ZH and ZW are essentially the only interesting calculi (a fourth one exists, but it arises from the interaction of the  $Z$ -spider with itself, and is thus a trivial edge case). Their paper provides a recipe for constructing these calculi from monoids and comonoids in the category of finite dimensional Hilbert spaces.

In this chapter, we briefly recap, and slightly generalize, their construction, giving the required background in category theory along the way.

### 2.1 Categorical foundations of graphical calculi

Graphical calculi are a special case of rewriting systems for *string diagrams*, which provide a way to reason about morphisms in a strict monoidal category. Here we give a brief from-scratch introduction to this formalism. This section is mainly based on Carette and Jeandel's classification paper [12] and additional material provided in *Picturing Quantum Processes* [14].

#### 2.1.1 Strict monoidal categories

We start by defining what a category is and which additional constraints we put on it to make it strict monoidal.

**Definition:** A (locally small) *category*  $\mathcal{C}$  consists of

- A collection  $Ob(\mathcal{C})$  of *objects*, and
- For each pair  $A, B$  of objects a set<sup>1</sup> of *morphisms*  $\text{Hom}_{\mathcal{C}}(A, B)$ .

---

<sup>1</sup>Generally, the morphisms do not need to form a set, in which case we drop the *locally small* attribute.

such that

- For each triplet  $A, B, C$  of objects there exists an associative composition map

$$\begin{aligned} \circ : \text{Hom}_{\mathcal{C}}(B, C) \times \text{Hom}_{\mathcal{C}}(A, B) &\rightarrow \text{Hom}_{\mathcal{C}}(A, C) \\ (g, f) &\mapsto g \circ f. \end{aligned}$$

- For each object  $A$  there exists an *identity morphism*  $\text{id}_A \in \text{Hom}_{\mathcal{C}}(A, A)$  such that for every object  $B$  and  $f \in \text{Hom}_{\mathcal{C}}(A, B)$ ,  $g \in \text{Hom}_{\mathcal{C}}(B, A)$  we have

$$\text{id}_A \circ g = g, \quad \text{and} \quad f \circ \text{id}_A = f.$$

Notationally, we write  $A \in \mathcal{C}$  for  $A \in \text{Ob}(\mathcal{C})$ , and  $f : A \rightarrow B$  for  $f \in \text{Hom}_{\mathcal{C}}(A, B)$  whenever the category  $\mathcal{C}$  is obvious from context.

**EXAMPLE 2.1.1 (FINITE-DIMENSIONAL HILBERT SPACES):** The category  $\mathbf{FDHilb}_d$  for fixed  $d \in \mathbb{N}$  consists of the Hilbert spaces  $\mathbb{C}^{(d^n)}$  for  $n \in \mathbb{N}_0$  with linear maps as morphisms. Identity morphisms are the identity maps. We will work almost exclusively in this category.

**EXAMPLE 2.1.2 (PARTIAL ORDERINGS):** It is a common misconception that morphisms always have to be functions. However, for a counter-example, consider the category where the objects are natural numbers and where we define

$$\text{Hom}(n, m) = \begin{cases} \emptyset & n > m \\ \{*\} & n \leq m \end{cases},$$

where  $\{*\}$  denotes an arbitrary singleton set.

Similar to how in abstract algebra we define homomorphisms to be structure preserving maps between groups, rings, fields, etc., we can define something akin to a *category homomorphism*, which we call a *functor*:

**Definition:** A *functor*  $F : \mathcal{C} \rightarrow \mathcal{D}$  between categories  $\mathcal{C}, \mathcal{D}$  is a map  $F : \text{Ob}(\mathcal{C}) \rightarrow \text{Ob}(\mathcal{D})$  and a collection of maps  $F : \text{Hom}_{\mathcal{C}}(A, B) \rightarrow \text{Hom}_{\mathcal{D}}(F(A), F(B))$  such that

$$F(f \circ g) = F(f) \circ F(g), \quad \text{and} \quad F(\text{id}_A) = \text{id}_A$$

for  $A, B \in \mathcal{C}$  and morphisms  $g : A \rightarrow B, f : B \rightarrow C$ .

Using this notion, we can then finally define strict monoidal categories:

---

However, all categories considered here happen to be locally small, meaning we do not need to concern ourselves with foundational issues of set theory

**Definition:** A *strict*<sup>2</sup> *monoidal category* is a category  $\mathcal{C}$  together with a functor  $\otimes : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{C}$ , the *tensor product*, and a special object  $I \in \mathcal{C}$  such that (writing  $\otimes$  as an infix operator)

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C, \quad \text{and} \quad I \otimes A = A = A \otimes I$$

for all  $A, B, C \in \mathcal{C}$ .

Here the *product category*  $\mathcal{C} \times \mathcal{C}$  is defined in the obvious way, meaning objects are just tuples of objects from  $\mathcal{C}$  and morphisms are just tuples of morphisms. Particularly, the requirement that  $\otimes$  is a functor then means that we have

$$(f_1 \circ g_1) \otimes (f_2 \circ g_2) = (f_1 \otimes f_2) \circ (g_1 \otimes g_2). \quad (2.1)$$

EXAMPLE 2.1.3 (FINITE-DIMENSIONAL HILBERT SPACES, AGAIN): The category  $\mathbf{FdHilb}_d$  is a strict monoidal category where  $\otimes$  is the tensor product of vector spaces/linear maps and  $I = \mathbb{C}$ . In fact, this category has the special property that every object apart from  $I$  is just a tensor product of  $\mathbb{C}^d$  with itself.

### 2.1.2 String diagrams

String diagrams are a graphical way to portray and manipulate morphisms in a strict monoidal category. In this dissertation (and in most literature on quantum graphical calculi), string diagrams are read bottom-to-top, with boxes representing morphisms and wires representing in-/outputs. For instance, a morphism  $f : A \otimes B \otimes C \rightarrow D \otimes E$  in some strict monoidal category would be represented by diagram (a) in Figure 2.1, the composite morphism  $(\text{id}_D \otimes g) \circ f$  is represented in diagram (b).

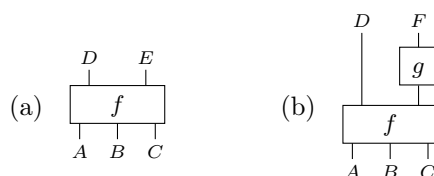
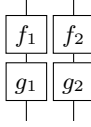


Figure 2.1: Examples of simple string diagrams

We thus see that the tensor product of morphisms (sometimes also called *parallel composition*) is portrayed by juxtaposition, while composition is realized by connecting matching wires. Here the fact that the tensor product is a functor (2.1) becomes central, as it means that the two interpretations of the following diagram are identical:

<sup>2</sup>The attribute *strict* comes from the use of equality below. If we replace it with the concept of *natural isomorphism*, we could drop the “strict”, however all categories we consider in this thesis happen to fulfill the additional requirement of strictness



An identity morphism is represented by just a wire. We further adopt the convention that we do not draw wires of type  $I$ .

Generally, string diagrams provide a much more readable way of writing out identities of strict monoidal categories, as we see in the following subsection.

### 2.1.3 Symmetric and compact closed structure

While string diagrams of a strict monoidal category are a convenient way of portraying morphisms, they do not allow for much manipulation yet. The reason for this is that they are rigid in the sense that we cannot freely move boxes around (a property often called the *only connectivity matters* paradigm [14, Cor. 3.5]). To attain this freedom of movement, we need (strict) symmetric and compact closed structures.

**Definition:** A strict monoidal category  $\mathcal{C}$  is *symmetric* if for every pair  $A, B \in \mathcal{C}$  there exist a *swap* map

$$\sigma_{A,B} : A \otimes B \rightarrow B \otimes A$$

such that

- S1)  $\sigma_{A,I} = \text{id}_A = \sigma_{I,A}$ ,
- S2)  $\sigma_{A,B \otimes C} = (\text{id}_B \otimes \sigma_{A,C}) \circ (\sigma_{A,B} \otimes \text{id}_C)$ ,
- S3)  $\sigma_{A,B} \circ \sigma_{B,A} = \text{id}_{A \otimes B}$ ,
- S4)  $\sigma_{A',B'} \circ (f \otimes g) = (g \otimes f) \circ \sigma_{A,B}$  for  $f : A \rightarrow A', g : B \rightarrow B'$ .

As a string diagram, we portray the swap maps as a crossing of wires:



Figure 2.2: String diagram for the swap map in a strict symmetric category

Then the above conditions S1) to S4) are visualized as follows:





EXAMPLE 2.1.4 (FINITE-DIMENSIONAL HILBERT SPACES, YET AGAIN): We call the orthonormal standard basis  $e_1, \dots, e_d$  of  $\mathbb{C}^d$  the *computational basis* and write its elements  $|0\rangle, \dots, |d-1\rangle$ . By  $\langle 0|, \dots, \langle d-1|$  we denote its dual basis of  $(\mathbb{C}^d)^*$ .

The special structure of  $\mathbf{FdHilb}_d$ , where every element is a tensor product of  $\mathbb{C}^d$  with itself, allows us to define symmetric and compact closed structures on just  $\mathbb{C}^d$ , which then extend to the entire category. The symmetric structure is always induced by the canonical swap

$$\begin{aligned} \sigma : \mathbb{C}^d \otimes \mathbb{C}^d &\rightarrow \mathbb{C}^d \otimes \mathbb{C}^d \\ |i\rangle \otimes |j\rangle &\mapsto |j\rangle \otimes |i\rangle, \end{aligned}$$

which then extends to the entire category via S2).

Regarding the compact closed structure, we can let every element be self-dual and define

$$\varepsilon_{\mathbb{C}^d} := \sum_{i=0}^{d-1} \langle i| \otimes \langle i|, \quad \text{and} \quad \eta_{\mathbb{C}^d} := \sum_{i=0}^{d-1} |i\rangle \otimes |i\rangle$$

which again extends to the entire category. Particularly, this means we only need to consider string diagrams where all wires are  $\mathbb{C}^d$ -labelled and can omit the type labels.

This works not only in  $\mathbf{FdHilb}_d$ , but in any category equivalent to a so-called *PROP* (product and permutation category), which consists of natural numbers as objects and addition as tensor product. Otherwise, we still get a compact structure for the full subcategory induced by tensor products of the element we define the cap and cup on.

This concludes our introduction to the basics of category theory and string diagrams. Next up, we look at how Carette and Jeandel systematically describe the spiders of quantum graphical calculi in purely categorical terms.

## 2.2 Anatomy of quantum graphical calculi

Quantum graphical calculi enrich the structure of a strict symmetric compact closed monoidal category with special *spider*-morphisms which allow for special rewrite rules that depend on the actual category the calculus is defined for (this category is always  $\mathbf{FdHilb}_d$  in our case).

In this section we briefly recap Carette and Jeandel's construction of spiders in terms of interacting monoids and comonoids.

### 2.2.1 Monoids and Comonoids

Let  $(\mathcal{C}, \otimes, I)$  be a strict symmetric monoidal category, e.g. not necessarily compact closed.

**Definition:** A monoid in  $\mathcal{C}$  is a triple  $(M, m, e)$  consisting of an object  $M$ , a *multiplication* morphism  $m : M \otimes M \rightarrow M$  and a *unit*  $e : I \rightarrow M$  such that

$$m \circ (m \otimes \text{id}_M) = m \circ (\text{id}_M \otimes m), \quad \text{and} \quad m \circ (\text{id}_M \otimes e) = \text{id}_M = m \circ (e \otimes \text{id}_M).$$

A monoid  $(M, m, e)$  is *commutative* if

$$m \circ \sigma_{M,M} = m.$$

Pictorially, we depict the multiplication map as a half-circle (because, as we will later see, it is half a spider), and the associativity, unit and symmetry conditions above become:

$$(M1-M3)$$

Figure 2.7: String diagrams for the commutative monoid equations

Let us consider some examples of monoids.

**EXAMPLE 2.2.1 (ALGEBRAIC MONOIDS [12, Ex. 5]):** One might wonder what the connection between a monoid in a monoidal category and an algebraic monoid is. To see this, consider the category **Set** of sets and functions between them. If we consider its *skeleton* by factoring modulo cardinality, we get a strict monoidal category where the tensor product is the cartesian product and the neutral element is the singleton  $\{*\}$ . Then a monoid  $(X, m, e)$  is exactly an algebraic monoid on the chosen representative of sets of cardinality  $|X|$ , where multiplication is given by  $m$  and the neutral element is  $e(*)$ .

**EXAMPLE 2.2.2 (THE CO-COPY MONOID [12, Ex. 6]):** In  $\mathbf{FdHilb}_d$ , define a commutative monoid on  $\mathbb{C}^d$  by setting

$$\begin{aligned} m : \mathbb{C}^d \otimes \mathbb{C}^d &\rightarrow \mathbb{C}^d & e : \mathbb{C} &\rightarrow \mathbb{C}^d \\ |i\rangle \otimes |j\rangle &\mapsto \delta_{ij}|i\rangle & 1 &\mapsto \sum_{i=0}^{d-1} |i\rangle. \end{aligned}$$

This monoid is half of the  $Z$ -spider.

In  $\mathbf{FdHilb}_d$ , monoids on  $\mathbb{C}^d$  correspond to associative,  $d$ -dimensional  $\mathbb{C}$ -algebrae.

Dual to the notion of a monoid is that of a comonoid. Informally, we get a comonoid by turning all the monoid equations upside-down:

$$(C1-C3)$$

Figure 2.8: String diagrams for the cocommutative comonoid equations

Formally, this leads to the following definition.

**Definition:** A *comonoid* in  $\mathcal{C}$  is a triple  $(C, c, \delta)$  consisting of an object  $C$ , a *comultiplication* morphism  $c : C \rightarrow C \otimes C$  and a *counit*  $\delta : C \rightarrow I$  such that

$$(c \otimes \text{id}_C) \circ c = (\text{id}_C \otimes c) \circ c, \quad \text{and} \quad (\text{id}_C \otimes \delta) \circ c = \text{id}_C = (\delta \otimes \text{id}_C) \circ c.$$

A comonoid is *cocommutative* if

$$\sigma_{C,C} \circ c = c.$$

**EXAMPLE 2.2.3 (THE COPY COMONOID [12, EX. 12]):** In  $\mathbf{FdHilb}_d$ , define a cocommutative comonoid on  $\mathbb{C}^d$  by setting

$$\begin{aligned} c : \mathbb{C}^d &\rightarrow \mathbb{C}^d \otimes \mathbb{C}^d & \delta : \mathbb{C}^d &\rightarrow \mathbb{C} \\ |i\rangle &\mapsto |i\rangle \otimes |i\rangle & |i\rangle &\mapsto 1. \end{aligned}$$

This comonoid is the second half of the  $Z$ -spider.

**EXAMPLE 2.2.4 (GROUP (CO-)MONOIDS [12, EX. 7, EX. 13]):** Given a finite group  $(G, \cdot)$  of order  $d$  with neutral element 1, identify the computational basis of  $\mathbb{C}^d$  with the group elements. Then we can define a monoid on  $\mathbb{C}^d$  via

$$\begin{aligned} m : \mathbb{C}^d \otimes \mathbb{C}^d &\rightarrow \mathbb{C}^d & e : \mathbb{C} &\rightarrow \mathbb{C}^d \\ |g\rangle \otimes |h\rangle &\mapsto |g \cdot h\rangle & 1 &\mapsto |1\rangle \end{aligned}$$

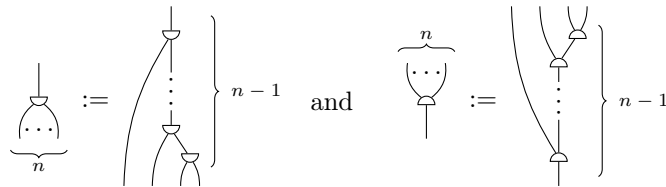
and a comonoid via

$$\begin{aligned} c : \mathbb{C}^d &\rightarrow \mathbb{C}^d \otimes \mathbb{C}^d & \delta : \mathbb{C}^d &\rightarrow \mathbb{C} \\ |g\rangle &\mapsto \sum_{\substack{a,b \in G \\ a \cdot b = g}} |a\rangle \otimes |b\rangle & |g\rangle &\mapsto \delta_{g1}. \end{aligned}$$

These are (co)commutative if  $G$  is commutative.

The group monoid/comonoid pair induced by  $(\mathbb{Z}/d\mathbb{Z}, +)$  forms exactly the two parts of the  $X$ -spider.

Lastly, note how the associativity of (co)multiplication allows us to define semi-circles with arbitrarily many in-/outputs:



This justifies us calling (co)monoids half spiders.

## 2.2.2 Frobenius Algebras and their induced compact structure

In the previous section we mention how specific monoid/comonoid pairs together form a spider. In this section we investigate what additional properties these pairs have to fulfill to actually define a spider that follows a well-defined fusion rule.

**Definition:** A monoid  $(M, m, e)$  and comonoid  $(M, c, \delta)$  on the same object  $M$  in  $\mathcal{C}$  form a *Frobenius algebra* if

$$(\text{id}_M \otimes m) \circ (c \otimes \text{id}_M) = c \circ m = (m \otimes \text{id}_M) \circ (\text{id}_M \otimes c).$$

The Frobenius algebra is commutative if the monoid is commutative and the comonoid is cocommutative.

Graphically, this means:

Figure 2.9: String diagram for the Frobenius algebra equations

It is these equations that allow for well-defined spider fusion. Thus, for a monoid/comonoid pair forming a commutative Frobenius algebra, we can indeed use the term *spider* and write:

The only thing this spider is missing is the ability to bend its legs. For this, we need a compact closed structure on the underlying category (or at least for cups and caps of type  $M$ ). Thankfully, our Frobenius algebra gives us those for free via:

Figure 2.10: The compact structure defined by a commutative Frobenius algebra

This pair of cap and cup satisfies the yanking equation, as:

Note that if we were, for example, in the category  $\mathbf{FdHilb}_d$  and had a commutative Frobenius algebra on  $\mathbb{C}^d$ , this suffices to define a compact closed structure on the entire category.

**Theorem 2.2.5:** *The above-defined spiders are flexsymmetric and fuse.*

PROOF (SKETCH): To see that the spiders fuse correctly, use the Frobenius algebra equations to “sort” all comultiply semicircles to the top of the diagram. Symmetry follows immediately from commutativity. To see that legs bend correctly, observe:

$$\begin{array}{c}
 \begin{array}{c} \text{Diagram 1} \\ \text{Diagram 2} \\ \text{Diagram 3} \\ \text{Diagram 4} \end{array} \\
 \stackrel{2.3}{=} \quad \stackrel{F}{=} \quad \stackrel{M2}{=}
 \end{array}
 \tag{2.4}$$

EXAMPLE 2.2.6 (*Z*-SPIDER): The cocopy monoid and copy comonoid from the previous subsection form a commutative Frobenius algebra, and thus a spider. Their induced compact structure is exactly the one mentioned in Example 2.1.4 and the spider is the well-known *Z*-spider from ZX-calculus:

$$\begin{array}{c}
 \overbrace{\dots}^n \\
 \text{Green Spider} \\
 \underbrace{\dots}_m
 \end{array}
 = \sum_{i=0}^{d-1} |i\rangle^{\otimes n} \langle i|^{\otimes m}$$

EXAMPLE 2.2.7 (*X*-SPIDER): Every pair of group monoid/comonoid forms a Frobenius algebra. The algebra induced by  $(\mathbb{Z}/d\mathbb{Z}, +)$  is commutative and thus yields a spider, namely the *X*-spider from ZX-calculus:

$$\begin{array}{c}
 \overbrace{\dots}^n \\
 \text{Red Spider} \\
 \underbrace{\dots}_m
 \end{array}
 = \sum_{\substack{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}/d\mathbb{Z} \\ i_1 + \dots + i_m = j_1 + \dots + j_n}} |j_1\rangle \dots |j_n\rangle \langle i_1| \dots \langle i_m|.$$

Its compact structure has cap

$$\begin{array}{c}
 \text{Cap Diagram} \\
 \stackrel{2.3}{=} \\
 \text{Cap Diagram} \\
 \stackrel{2.2.4}{=} \langle 0| \sum_{x=0}^{d-1} \sum_{y=0}^{d-1} |x+y\rangle \langle x| \langle y| = \sum_{\substack{x, y \in \mathbb{Z}/d\mathbb{Z} \\ x+y=0}} \langle x| \langle y| = \sum_{x=0}^{d-1} \langle x| \langle -x|
 \end{array}$$

and cup  $\sum_{x=0}^{d-1} |x\rangle \langle -x|$  by analogous computation.

As the previous two examples might have reminded the reader, a quantum graphical calculus generally has two types of spiders. If we construct both spiders from Frobenius algebras, we run into the problem that we get two compact closed structures, which generally will not be identical. This means that bending legs on each type of spider requires a different cup/cap, which is undesirable. The next section thus introduces the machinery to modify one type of spider slightly to allow its legs to bend using the other spider’s compact closed structure.



notion of compatibility comes into play. Carette and Jeandel require this intermediary spider to also be the dualizer, and their notion of compatibility is the requirement that the dualizer is self-inverse, expressed as follows:

$$\begin{array}{c} \bullet \\ \text{---} \\ \text{---} \\ \circ \end{array} = \begin{array}{c} \circ \\ \text{---} \\ \text{---} \\ \bullet \end{array} \tag{2.7}$$

This condition, however, is not necessary if we instead use the RHS of the compatibility condition as our intermediary spider, as it is always the inverse of the dualizer by the snake equations.

**Theorem 2.3.2 (Generalization of [12, p. 9]):** *Any two spiders yield a flexsymmetric quantum graphical calculus, potentially with one spider following harvestman fusion. Additionally, the bialgebra<sup>3</sup> rule [12, p. 9] holds as*

$$\begin{array}{c} \circ^{-1} \quad \circ^{-1} \\ \text{---} \quad \text{---} \\ \text{---} \quad \text{---} \\ \circ \quad \circ \end{array} = \begin{array}{c} \circ^{-1} \\ \text{---} \\ \bullet \end{array} \quad \text{if} \quad \begin{array}{c} \text{---} \quad \text{---} \\ \text{---} \quad \text{---} \\ \circ \quad \circ \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \bullet \end{array}$$

PROOF: Follows immediately from the definitions ■

The compatibility condition arises as a consequence of two interacting Frobenius-algebrae forming a *Hopf*-algebra [18, Cor. 6.9], meaning they satisfy the following condition [18, Thm. 6.8, Def. 6.2]:

$$\begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \bullet \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \bullet \end{array} \tag{2.8}$$

Figure 2.12: Abstract Hopf-algebra rule for interacting Frobenius algebrae

The ZX-Calculus satisfies this rule via the map  $x \mapsto -x$  in  $\mathbb{Z}/d\mathbb{Z}$  (see [36, Eq. 4.3], also H), which is trivial for  $d = 2$ , making qubit ZX particularly pleasant to work with. The underlying Frobenius-algebrae of the ZW-calculus also form a Hopf-algebra [36, p. 133]. In qubit ZH, we do not have a Hopf-algebra (The second condition of [18, Thm. 6.4] is not satisfied), but the dualizer is the Hadamard gate, which is self-adjoint in this special case, which is why we still get complementarity. However, when trying to construct a qudit ZH-calculus, the dualizer is still the quantum Fourier transform, which is no longer self-adjoint in dimension  $d \geq 3$ .

The ability to drop this condition has consequences for Carette and Jeandel’s classification of qubit quantum graphical calculi [12, Thm. 30]. No new graphical calculi arise, as the classification of candidate pairs of spiders is based on a classification of all algebrae in two-dimensional complex vector spaces tracing back to 1890 [41]. The only place where Carette and Jeandel impose the restriction of compatibility is the proof of their Theorem 30 [12, Thm. 30], which classifies valid *phase shifts*. In fact, all this proof entails is a classification of phase shifts yielding compatible

<sup>3</sup>Carette and Jeandel refer to this rule as the *bigebra* rule, as they correctly note that for two Frobenius algebrae to form a bialgebra, they need to fulfill 3 additional axioms [12, Def. 21]. We stick to the name *bialgebra* for this rule, as is established in ZX-literature

calculi. Dropping the requirement of compatibility thus yields that no matter how we phase shift a calculus, it remains valid, except that fusion of one of the involved spiders has to be performed via the inverse of the dualizer, and that the bialgebra rule becomes 2.3.2.

## 2.4 A flexsymmetric qudit ZX-Calculus

As an easy example for how to construct quantum graphical calculi using this recipe, let us consider the case of qudit ZX. Van de Wetering and Yeh describe a flexsymmetric *qutrit* ZX-Calculus in their 2022 paper on qutrit phase gadgets [49], where they notice that only a slight modification to the  $X$ -spider is required, yielding a harvestman fusion rule. As we will see shortly, this modification is necessary because the dualizer of the  $Z$  and  $X$  structures is no longer trivial. Their modification is exactly the last step described in Carette and Jeandel’s paper, and generalizes to arbitrary dimension.

In Examples 2.2.6 and 2.2.7 we already saw how to define the  $Z$ - and  $X$ -spider for arbitrary dimension. Thus the only thing left to do is determine the dualizer and modify the  $X$ -spider to make its legs bend using the cup and cap induced by the  $Z$ -algebra. We have

$$\begin{array}{c} | \\ \circ \end{array} = \begin{array}{c} \text{red circle} \\ \text{cup} \\ \text{green circle} \end{array} = \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} |k\rangle \langle -j|k\rangle \langle j| \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \delta_{-j,k} |k\rangle \langle j| = \sum_{j=0}^{d-1} |j\rangle \langle -j|$$

meaning the dualizer is indeed negation modulo  $d$ . This is in fact self-adjoint, meaning the  $Z$ - and  $X$ -algebrae are even compatible in Carette and Jeandel’s sense. Thus we get a flexsymmetric quantum graphical calculus where the  $Z$ -spider fuses normally, and the  $X$ -spider fuses via the dualizer.

Keeping in line with van de Wetering and Yeh’s notation, we draw the redefined  $X$ -spider in light pink:

$$\begin{array}{c} \overbrace{\text{...}}^n \\ \text{pink spider} \\ \underbrace{\text{...}}_m \end{array} := \begin{array}{c} \overbrace{\text{...}}^n \\ \text{pink spider} \\ \underbrace{\text{...}}_m \end{array} = \sum_{\substack{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}/d\mathbb{Z} \\ i_1 + \dots + i_m = j_1 + \dots + j_n}} | -j_1 \rangle \dots | -j_n \rangle \langle i_1 | \dots \langle i_m | \\ = \sum_{\substack{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}/d\mathbb{Z} \\ i_1 + \dots + i_m + j_1 + \dots + j_n = 0}} | j_1 \rangle \dots | j_n \rangle \langle i_1 | \dots \langle i_m |.$$

This is exactly Equation 4 of van de Wetering and Yeh’s paper [49]. We leave the determination of appropriate rewrite rules for this calculus mostly to other works. Appendix A briefly concerns itself with the color-change rule. Van de Wetering and Yeh describe rewrite rules for the qutrit case [49], while Booth and Carette provide a complete axiomatization of the Clifford fragment in odd prime dimension [10].



## Chapter 3

# Construction of a qudit $H$ -box for $d > 2$

In this chapter we work towards the main goal of this dissertation, the generalization of the qubit ZH-calculus to higher dimensions. We provide the most important building block of ZH-calculus, namely the  $H$ -box, which we construct in a flexsymmetric way (with regard to the compact structure induced by the  $Z$ -spider).

Our approach for constructing the  $H$ -box is based on the following ideas:

- (i) Look at the (co)monoid the qubit ZH-calculus is constructed from. The monoid here turns out to be logical AND, or multiplication in  $\mathbb{F}_2$ .
- (ii) Attempt to use multiplication in  $\mathbb{F}_d$  as monoid and try to find a comonoid with which it forms a Frobenius algebra.
- (iii) Modify the resulting  $H$ -box as in the previous chapter to get a flexsymmetric qudit ZH-calculus.

### 3.1 Building blocks of qubit ZH

Carette and Jeandel's classification of all qubit quantum graphical calculi tells us that the multiply morphism  $m$  of the monoid associated with the qubit ZH-calculus [12, p. 10] acts as logical AND on the computational basis states  $|0\rangle, |1\rangle$ , e.g.

$$\frac{m \quad |0\rangle \quad |1\rangle}{\begin{array}{c|cc} |0\rangle & |0\rangle & |0\rangle \\ |1\rangle & |0\rangle & |1\rangle \end{array}},$$

whereas the comonoid does not allow for any immediately obvious interpretation. We furthermore know of the following relation between logical AND, and  $H$ -boxes in ZH-calculus:

$$\text{AND} = \text{yellow semi-circle with dot}$$
(3.1)

Figure 3.1: AND-gate realization in qubit ZH

In fact, this equation is just the first of 2.6 with the inverse of the dualizer applied to the outputs. These key observations about qubit ZH give us a very good starting point for attempting to construct a qudit ZH-calculus:

- (i) The monoid of qudit ZH should be a generalization of logical AND.
- (ii) The (inverse) dualizer should be a generalization of the Hadamard gate.

We now work out exactly what these two generalizations should be, and how we can construct a whole qudit  $H$ -box from them.

### 3.2 Monoid from generalizing AND

An obvious generalization of logical AND to dits is multiplication, either modulo  $d$  or in finite fields  $\mathbb{F}_d$  (for  $d$  prime these coincide, however multiplication in  $\mathbb{F}_{p^r}$  for  $r \geq 2$  is not multiplication modulo  $p^r$  as  $\mathbb{F}_{p^r} \neq \mathbb{Z}/p^r\mathbb{Z}$ ). As an operation on the computational basis state, we can define such a “multiply box” as follows:

$$\text{multiply box} := \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} |i \cdot j\rangle \langle i| \langle j|$$
(3.2)

Figure 3.2: Multiply box as a generalization of logical AND

Clearly, similarly to logical AND, this multiply box defines a monoid with unit  $|1\rangle$ , so we can portray it as a yellow semi-circle, e.g.

$$\text{yellow semi-circle with dot} := \text{multiply box} \quad \text{yellow semi-circle with dot} := |1\rangle$$
(3.3)

Figure 3.3: The multiply monoid

Intuitively, we now want to replace the AND-box in Equation 3.1 with this multiply box and solve for the  $H$ -box. However, before we can successfully attempt this, we need to determine our generalization of the Hadamard gate. In all generalizations of ZX-calculus to higher dimensions considered so far, the Hadamard has been generalized to the *quantum Fourier transform* [10][45][49], that is, to the map

$$\begin{array}{|} \hline \square \\ \hline \end{array} := \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \zeta^{ij} |i\rangle\langle j|$$

Figure 3.4: The quantum Fourier transform

where  $\zeta$  denotes a primitive  $d$ -th root of unity (see Appendix A for details on roots of unity and why we can pick any primitive root). We keep van de Weterings and Yeh's notation of using a slanted box to denote the quantum Fourier transform [49] to indicate that it is no longer self-adjoint. This loss of self-adjointness, however, introduces a slight problem, as it gives us two possible ways of generalizing the Hadamard: Both the quantum Fourier transform and its adjoint are candidates. Its adjoint, the inverse quantum Fourier transform, is the map

$$\begin{array}{|} \hline \square^{-1} \\ \hline \end{array} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \bar{\zeta}^{ij} |i\rangle\langle j| = \begin{array}{|} \hline \square \\ \hline \end{array}$$

Figure 3.5: The inverse quantum Fourier transform

Thus, we have the following two candidates for the bottom half of our qudit  $H$ -box, based on replacing the AND-box in Equation 3.1 with the multiply box, and the Hadamard with either the quantum Fourier transform or its inverse:

$$\begin{array}{|} \hline \square \\ \hline \end{array} \stackrel{?}{=} \begin{array}{|} \hline \square \\ \hline \end{array} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \bar{\zeta}^{ijk} |k\rangle\langle i|\langle j| \quad \text{and} \quad \begin{array}{|} \hline \square \\ \hline \end{array} \stackrel{?}{=} \begin{array}{|} \hline \square \\ \hline \end{array} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \zeta^{ijk} |k\rangle\langle i|\langle j|$$

Figure 3.6: Two possible definitions of the bottom half of the qudit  $H$ -box

We see that these only differ in the choice of root of unity. However, as we show in Appendix A, we could have picked any primitive root of unity in our definition of the quantum Fourier transform<sup>1</sup>, meaning these two definition of the  $H$ -box are not essentially different. We shall use the second one, as it is ever so slightly simpler.

Note that this choice implies the following generalization of Equation 3.1:

$$\begin{array}{|} \hline \square \\ \hline \end{array} = \begin{array}{|} \hline \square \\ \hline \end{array} \quad (3.4)$$

Figure 3.7: Realization of the multiply box in qudit ZH

We want to note the similarity of this half qudit  $H$ -box with the known qubit  $H$ -box. Recall that in the qubit ZH-calculus we have [7, Sec 2.1]

<sup>1</sup>The conjugate of a primitive root of unity is again primitive, as  $(\bar{\zeta})^k = 1$  implies  $\zeta^k = 1$  by conjugating both sides



which is the identity wire written in bra-ket notation.

The last thing we now need to check is whether this monoid/comonoid pair forms a Frobenius algebra.

**Theorem 3.3.1:** *The multiplication monoid and the comonoid defined above form a commutative Frobenius algebra.*

PROOF: We need to verify that F holds. Essentially, this is just an exercise in rearranging sums and applying A.1.5. The key insight here is that the outputs are always multiples of the inputs. We have

$$\begin{aligned}
& \begin{array}{c} \text{Diagram 1} \\ |a\rangle |b\rangle \end{array} = \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{ajk-\mu j-\nu k} |\mu\rangle |\nu b\rangle = \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} |\mu\rangle |\nu b\rangle \sum_{j=0}^{d-1} \bar{\zeta}^{\mu j} \sum_{k=0}^{d-1} (\zeta^{aj-\nu})^k \\
& \stackrel{A.1.5}{=} d \cdot \sum_{\mu=0}^{d-1} \sum_{j=0}^{d-1} \bar{\zeta}^{\mu j} |\mu\rangle |ajb\rangle \\
& \begin{array}{c} \text{Diagram 2} \\ |a\rangle |b\rangle \end{array} = \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{abjk-\mu j-\nu k} |\mu\rangle |\nu\rangle = \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} |\mu\rangle |\nu\rangle \sum_{j=0}^{d-1} \bar{\zeta}^{\mu j} \sum_{k=0}^{d-1} (\zeta^{abj-\nu})^k \\
& \stackrel{A.1.5}{=} d \cdot \sum_{\mu=0}^{d-1} \sum_{j=0}^{d-1} \bar{\zeta}^{\mu j} |\mu\rangle |ajb\rangle \\
& \begin{array}{c} \text{Diagram 3} \\ |a\rangle |b\rangle \end{array} = \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{abjk-\mu j-\nu k} |\mu\rangle |\nu\rangle = \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} |\mu\rangle |\nu\rangle \sum_{k=0}^{d-1} \bar{\zeta}^{\nu k} \sum_{j=0}^{d-1} (\zeta^{abk-\mu})^j \\
& \stackrel{A.1.5}{=} d \cdot \sum_{\nu=0}^{d-1} \sum_{k=0}^{d-1} \bar{\zeta}^{\nu k} |abk\rangle |\nu\rangle \\
& \begin{array}{c} \text{Diagram 4} \\ |a\rangle |b\rangle \end{array} = \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{bjk-\mu j-\nu k} |\mu a\rangle |\nu\rangle = \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} |\mu a\rangle |\nu\rangle \sum_{k=0}^{d-1} \bar{\zeta}^{\nu k} \sum_{j=0}^{d-1} (\zeta^{bk-\mu})^j \\
& \stackrel{A.1.5}{=} d \cdot \sum_{\nu=0}^{d-1} \sum_{k=0}^{d-1} \bar{\zeta}^{\nu k} |abk\rangle |\nu\rangle.
\end{aligned}$$

The “commutative” part follows as the monoid and comonoid are commutative and cocommutative respectively. ■

Thus our monoid and comonoid form a generalization of the  $H$ -box in arbitrary dimension  $d > 2$ . Next up, we will determine the dualizer and modify these spider parts to arrive at a flexsymmetric ZH-calculus. However, first let us conclude this section with some remarks on the construction so far.

**Remark 3.3.2:** We started our construction of the  $H$ -box by constructing a multiply box as a generalization of the AND-gate. We could just the same have started with an “unmultiply” box that is the transpose of the multiply box, in the sense of 2.2.4. This would have resulted in the same calculus.

However we want to note that multiply and unmultiply together do not form a Frobenius-algebra, as even if  $((\mathbb{Z}/d\mathbb{Z})^*, \cdot)$  is a group,  $(\mathbb{Z}/d\mathbb{Z}, \cdot)$  never is, as 0 does not have an inverse. Thus, we do not actually form a group algebra.

**Remark 3.3.3:** As we mention at the beginning of this chapter, for  $d = p^r$  with  $p$  prime, we have some ambiguity regarding the choice of the monoid. Since we see later in this dissertation that many results based on defining the monoid as multiplication modulo  $d$  do not hold for  $d$  not prime, it is natural to ask if we can recover them for  $d = p^r$  were we to consider arithmetic in  $\mathbb{F}_{p^r}$  instead.

The answer to this question is sadly no, as the structure of  $\mathbb{F}_{p^r}$  is given by considering an irreducible polynomial  $q$  with  $\deg q = r$  over  $F_p$  and then setting <sup>2</sup>

$$\mathbb{F}_{p^r} := \mathbb{F}_p[X]/(q).$$

Particularly, the additive structure of this is no longer isomorphic to any quotient of the integers, making exponentiation of roots of unity impossible.

### 3.4 Qudit ZH from scratch

To build a flexsymmetric qudit ZH-calculus, we first need to determine the ZH dualizer and its inverse. Recall that by Equation 2.5, we need to first compute the compact structure induced by the  $H$ -algebra. For the cap, we have

$$\text{cap} \stackrel{2.3}{=} \text{cap} \stackrel{3.3}{=} \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \zeta^{ij} |i\rangle\langle j|,$$

and for the cup we have

$$\text{cup} \stackrel{2.3}{=} \text{cup} \stackrel{3.3}{=} \frac{1}{\sqrt{d^3}} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{jk-\mu j-\nu k} |\mu\rangle\langle \nu| \stackrel{A.1.5}{=} \frac{1}{\sqrt{d}} \sum_{\mu=0}^{d-1} \sum_{\nu=0}^{d-1} \zeta^{\mu\nu} |\mu\rangle\langle \nu|.$$

<sup>2</sup>As  $q$  is irreducible, it is the minimal polynomial of one of its roots, say  $\alpha$ , e.g.  $\mathbb{F}_p[X]/(q) \cong \mathbb{F}_p(\alpha)$ . As  $q$  is irreducible,  $(q)$  is a maximal ideal and thus  $\mathbb{F}_p(\alpha)$  a field. Since  $q$  has degree  $r$ , we get that the degree  $[\mathbb{F}_p(\alpha) : \mathbb{F}_p]$  of the field extension  $\mathbb{F}_p(\alpha) : \mathbb{F}_p$  is  $r$ . Particularly,  $\mathbb{F}_p(\alpha)$  is a  $r$ -dimensional vector space over  $\mathbb{F}_p$  and thus has  $|\mathbb{F}_p|^r = p^r$  elements. However, finite fields are unique up to isomorphism and therefore we must have  $\mathbb{F}_{p^r} \cong \mathbb{F}_p[X]/(q)$ . We again refer to standard textbooks on abstract algebra such as Lang [33] for an in-depth explanation of these concepts

This means that our dualizer is

and its inverse is

which should come as no big surprise. In fact, from the considerations in the previous two sections, we already know what the resulting  $H$ -box will look like, while Section 2.3 tells us how the harvestman fusion rule will look:

$$\underbrace{\begin{array}{c} \overbrace{\dots}^n \\ \text{---} \\ \underbrace{\dots}_m \end{array}} := \frac{1}{\sqrt{d}} \sum_{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}/d\mathbb{Z}} \zeta^{i_1 \dots i_m j_1 \dots j_n} |j_1 \dots j_n\rangle \langle i_1 \dots i_m| \quad (3.7)$$

Figure 3.8: Flexsymmetric qudit  $H$ -box

Figure 3.9: Harvestman fusion rule for qudit  $H$ -boxes

There is one last thing we need to verify before we can truly claim to have found a qudit ZH-calculus, namely that the  $H$ -box satisfies a form of the bigebra rule [12, p.9]. According to Theorem 2.3.2, it suffices to check that the original  $H$ -monoid satisfies the traditional bigebra rule with the  $Z$ -comonoid. However, this is obvious if we remember that the  $Z$ -comonoid is just the COPY map and the  $H$ -monoid is multiplication:

# Chapter 4

## Qudit ZH-calculus

In Chapter 3 we construct a flexsymmetric  $H$ -box. Now, we pair this box with the  $Z$ -spider to describe the ZH-calculus for qudits, which is the main goal of this dissertation.

We first describe the generators of our qudit ZH-calculus, which are very similar to those of Backens et al's qubit ZH-calculus (up to normalization) [7]. For convenience, we reproduce the qubit version of the rules in Figure 1.11. We then investigate a possible set of rewrite rules and conclude the chapter by generalizing some known qubit identities to the qudit case.

### 4.1 Generators

In this section we briefly describe the (derived) generators of our qudit ZH-calculus. The basic generators are clearly the  $Z$ -spider and the  $H$ -box. Recall from 2.2.6 that

$$\underbrace{\left( \begin{array}{c} \overbrace{\dots}^n \\ \text{---} \\ \underbrace{\dots}_m \end{array} \right)}_{\text{green spider}} := \sum_{i=0}^{d-1} |i\rangle^{\otimes n} \langle i|^{\otimes m}$$

and from the end of Chapter 3 that

$$\underbrace{\left( \begin{array}{c} \overbrace{\dots}^n \\ \text{---} \\ \underbrace{\dots}_m \end{array} \right)}_{\text{yellow spider}} := \frac{1}{\sqrt{d}} \sum_{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}/d\mathbb{Z}} \zeta^{i_1 \dots i_m j_1 \dots j_n} |j_1 \dots j_n\rangle \langle i_1 \dots i_m|.$$

Note that we still write the quantum Fourier transform, e.g. the  $H$ -box with a single in-/output, slanted:

$$\left| \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array} \right| := \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \zeta^{ij} |i\rangle \langle j|$$

Just as in the qubit case, we introduce the  $X$ -spider as a derived generator by applying a quantum Fourier transform to each leg of a  $Z$ -spider:



$$\text{Red Spider} := \text{Green Spider with Yellow Squares} \quad (x)$$

Figure 4.1: Definition of the  $X$ -spider as a derived generator

Note that this red spider does not equal the pink spider introduced for qudit ZX in Section 2.4. As we show in Lemma A.2.1, it is only equal to said spider up to a scalar factor dependant on the number of legs.

Next, let us consider the two additional derived generators Backens et al introduce for the qubit ZH-calculus [7, Eq. 1, Eq. 3]:

$$\text{Green Spider with 1 Output} := \text{Green Spider with 1 Output and Yellow Square} \quad \text{Red Spider with 1 Output} := \text{Red Spider with 1 Output and 3 Yellow Squares} \quad (4.1)$$

Figure 4.2: Generalized  $\neg$ -spiders

One easily verifies that the green negate spider with one in- and output still realizes Pauli-Z, which in the qudit case becomes [49, Eq. 1]

$$Z|i\rangle = \zeta^i|i\rangle.$$

However, the red negate spider with one in- and output no longer realizes Pauli-X. Instead, its operation on a single qudit is best described as pointwise reflection along  $d-1$ . Luckily, we can recover Pauli-X<sup>1</sup> by applying two quantum Fourier-transforms to its output (the scalar factor arises due to the  $X$ -spider only realizing addition up to a scalar factor):

$$\text{Red Spider with Plus} := \text{Green Spider with Plus and Red Spider with Plus} \quad (4.2)$$

Figure 4.3: Realization of Pauli-X in qudit ZH

We pick the notation “+” as qudit Pauli-X realizes the following linear map [49, Eq. 1]:

$$X|i\rangle = |i+1\rangle.$$

Lastly, we introduce generators for some very common scalars:

$$\blacktriangle := \text{Green Spider with 1 Output and Red Spider with 1 Output} \quad \text{and} \quad \star := \text{Green Spider with 1 Output and Red Spider with 1 Output and 2 Yellow Squares} \quad (4.3)$$

Requiring scalars in our rewrite rules is unfortunate, but as de Beaudrap shows, it is impossible to avoid for the ZH-calculus [8, Ap. E]<sup>2</sup>.

We conclude this section by providing an algebraic interpretation of these operators by giving a correspondence with arithmetic operations of  $\mathbb{Z}/d\mathbb{Z}$ , based on how Backens et al interpret the qubit ZH-generators as boolean operations [7, Eq. 5]:

<sup>1</sup>We identify this ability to express Pauli-X using the red  $\neg$ -spider as central, which is why we pick the generalization using the inverse quantum Fourier transform. See C.0.4 for the alternate generalization

<sup>2</sup>He does provide a version with *less* scalars at the cost of less intuitive interpretation of the generators

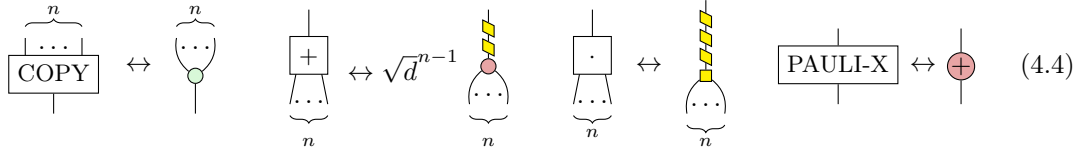


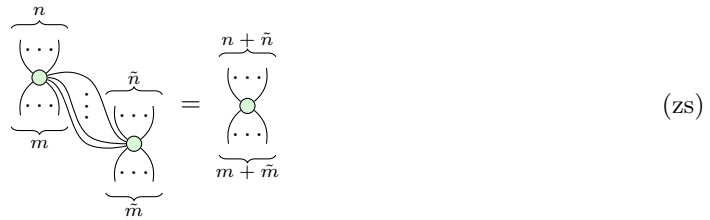
Figure 4.4: Correspondence between qudit ZH-generators and arithmetic in  $\mathbb{Z}/d\mathbb{Z}$

## 4.2 Qudit ZH rules

In this section, we describe a possible ruleset for our qudit ZH-calculus. We use Backens et al's qubit ZH completeness result [7, Fig. 1] as a starting point, and first try to generalize as many of their rules as possible. Note that we defer soundness to Appendix B.

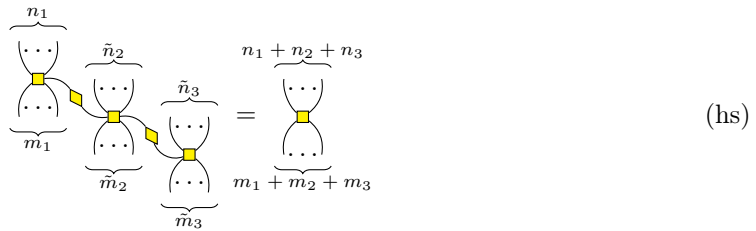
The most obvious rule is the  $Z$ -spider fusion rule, which holds unchanged from qubit ZH:

**Rule 1 ( $Z$ -spider fusion):**



We also already saw the  $H$ -box fusion rule in the previous chapter (Eq. 3.8). We can actually slightly strengthen this into a rule that allows us to contract any odd-length sequence of  $H$ -boxes interspersed by quantum Fourier transforms:

**Rule 2 ( $H$ -box contraction):**



The identity rule also holds unchanged:

**Rule 3 (Identity):**



From here on we deviate slightly from Backens et al’s rule set. This is because their rule (hh) – the self-adjointness of the Hadamard gate – does not hold anymore in the qudit case for  $d > 2$ . We know that its generalization, the quantum Fourier transform, requires fourfold repetition before yielding the identity again. However, we see in Section 4.3 that we can derive this “Fourier transform” rule from the remaining rules we introduce in this section. Therefore, we skip straight to the generalizations of the bialgebra rules.

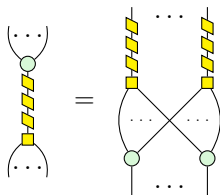
As we choose a normalization of the quantum Fourier transform different from Backens et al [7, Section 2.1] (they pick 1, we pick  $\frac{1}{\sqrt{d}}$ ), our red spider is only equal to the pink spider of qudit ZX introduced in Section 2.4 up to a scalar that depends on the number of legs (see A.2.1). Thus, our version of the bialgebra rule for  $Z$ - and  $X$ -spider introduces some scalar factors:

**Rule 4 (Bialgebra 1):**

$$\text{Diagram (ba1)} \quad \blacktriangle^{(n-1)(m-1)} \tag{ba1}$$

If  $(n - 1)(m - 1) < 0$ , introduce  $\blacktriangle^{-(n-1)(m-1)}$  to the LHS instead.

The second bialgebra rule in Backens et al’s paper [7, Fig. 1, (ba<sub>2</sub>)] concerns the interaction of the  $Z$ -spider and  $H$ -box. We already saw a bialgebra rule for these generators in 2.3.2, which in our specific case becomes:



We can make this rule look more like the one in Backens et al’s paper to arrive at

**Rule 5 (Bialgebra 2):**

$$\text{Diagram (ba2)} \tag{ba2}$$

Our last rule is a generalization of the *multiply* rule (m). It turns out what this rule really captures is the cyclic structure of the additive group of  $\mathbb{Z}/d\mathbb{Z}$ , which is why we call its generalization (c) for *cyclic*:

**Rule 6 (Cyclic):**

$$\blacktriangle^d \text{Diagram (c)} = \text{Diagram (c)} \tag{c}$$

Note how these rules cannot possibly form a complete set of rewrite rules. To see this, consider the following equation, which states that the Frobenius map  $x \mapsto x^d$  is the identity:

In finite fields, such as  $\mathbb{Z}/d\mathbb{Z}$  for  $d$  prime, this equation holds true, as multiplication forms a cyclic group (see C.0.1). However, consider  $d = 4$ . Here we have  $2^2 = 4 = 0$ , meaning  $2^4 = 0 \neq 2$ , meaning the equation is not true in general.

Since none of our rewrite rules actually include any side conditions on the dimension  $d$ , any sequence of rewrite steps that transform the Frobenius map into the identity would work for any  $d$ . Therefore, since the above rules are sound, we cannot derive the Frobenius identity for prime  $d$ .

Beyond simply stating that we are missing rewrite rules, this further implies that any additional rewrite rule must somehow capture the “essence” of the multiplicative group of  $\mathbb{Z}/d\mathbb{Z}$ . While for prime  $d$  this might be simple, the rules required for  $d$  not prime might potentially be quite unwieldy. A general set of rewrite rules that works for all  $d$  simultaneously seems unlikely (or would require a myriad of side conditions).

### 4.3 Deriving (ft) and color change

Note how the above ruleset neither contains a rule stating that repeating Fourier transform four times yields the identity, nor an inverted color change rule. That is because we can derive them from the rules presented in the previous section:

**Lemma 4.3.1 (Quantum Fourier Transform):**

(ft)

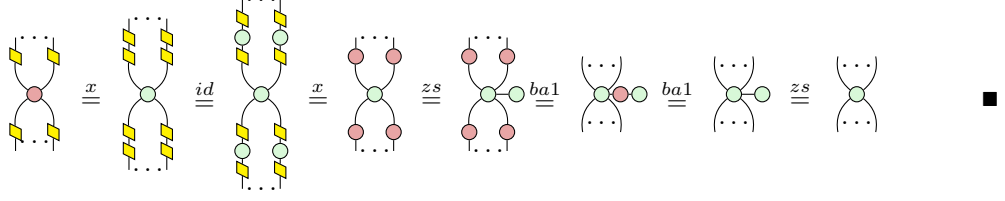
PROOF:

■

**Lemma 4.3.2 (Color Change):**

(h)

PROOF:



Note that in both of these proofs, the application of the bialgebra rule  $ba1$  does not introduce scalars, as the number of inputs or output of the subdiagram we apply the rule to is always 1.

We use the name (h) for the color change rule to keep in line with Yeh and van de Wetering's notation [49, Fig. 1].

Lastly, note how in both of these proofs we only use rules that are also valid in ZX-calculus, meaning these derivations also hold for *phaseless* qudit ZX. If we introduce phases for the Z- and X-spiders, the above proof no longer works. This is because the color change rule h shuffles the phases, as Yeh and van de Wetering demonstrate [49, Fig. 1].

## 4.4 Further generalizations

In this section we talk about generalizing the two further rules proposed by Backens et al [7, Fig. 1, Lem. 2.28]. These rules are instrumental in proving completeness in the qubit case, but two of them become quite complicated in the qudit setting, which is why we are reluctant to include them in our base ruleset. Note that technically Backens et al introduce a third rule [7, Lem. 5.1], but since that rule is simply the Frobenius equation from the previous section, we see no need to cover it again.

### 4.4.1 The ortho rule

The ortho rule (o) is the most complicated qubit rule. It essentially states that

$$\forall x, y, z. xy = z(y + 1) \iff xy = 0 = z(y + 1).$$

This observation is based on “exhausting” all possible values of  $\mathbb{Z}/2\mathbb{Z}$ : No matter what  $y \in \mathbb{Z}/2\mathbb{Z}$  we choose, we have  $\{y, y + 1\} = \mathbb{Z}/2\mathbb{Z}$ . That means either  $y$  or  $y + 1$  is zero, so one of  $xy$  and  $z(y + 1)$  is always zero. Thus, if  $xy = z(y + 1)$ , both products must equal 0.

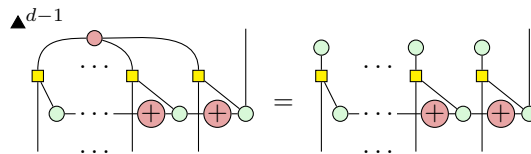
We can generalize this argument to  $\mathbb{Z}/d\mathbb{Z}$  for arbitrary  $d$ :

$$\forall y. \{y, y + 1, \dots, y + d - 1\} = \mathbb{Z}/d\mathbb{Z}$$

and thus

$$\forall x_0, \dots, x_{d-1}, y. x_0 y = \dots = x_{d-1} (y + d - 1) \iff \forall i \in \{0, \dots, d - 1\}. x_i (y + i) = 0.$$

Expressing this as ZH-diagram (assuming that the Pauli-X inputs are on the left) we get



For  $d$  prime this rule actually becomes slightly stronger, as the absence of zero-divisors tells us:

$$\forall x_0, \dots, x_{x-1}, y. x_0 y = \dots = x_{d-1} (y + d - 1) \iff \text{at most one } x_i \neq 0,$$

however, this condition is not easily expressible as a ZH-diagram.

#### 4.4.2 Lemma 2.28

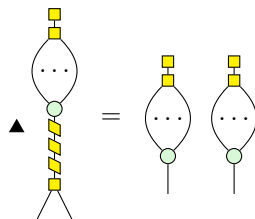
Backens et al.'s Lemma 2.28 [7] states that

$$\forall x, y. xy = 1 \iff x = 1 \wedge y = 1.$$

Clearly, this does not hold for  $d > 2$ , however for prime  $d$  we can generalize it to the following statement:

$$\forall x, y. xy \neq 0 \iff x \neq 0 \wedge y \neq 0$$

Obviously, this does not hold for  $d$  composite, since then  $\mathbb{Z}/d\mathbb{Z}$  admits zero-divisors. Using C.0.1 and C.0.4 we can realize this as the diagrammatic equation



where all “...” represent  $(d - 1)$ -fold repetition.

### 4.5 Comparison to the qubit case

Let us briefly compare our qudit ruleset to the qubit ruleset Backens et al use in their completeness paper [7]. Ignoring differences in scalars introduced by us choosing a different normalization for  $H$ -boxes, only three rules exhibit changes. We denote these qubit versions with superscripts as follows:

$$\begin{array}{c}
 \begin{array}{c} (\dots) \\ \text{[yellow square]} \\ \text{[yellow square]} \\ (\dots) \end{array} \stackrel{(hs^2)}{=} \begin{array}{c} (\dots) \\ \text{[yellow square]} \\ (\dots) \end{array} \quad \left| \quad \begin{array}{c} \text{[yellow square]} \\ \text{[yellow square]} \end{array} \stackrel{(ft^2)}{=} \text{[yellow square]} \quad \left| \quad \begin{array}{c} (\dots) \\ \text{[red circle]} \\ \text{[yellow square]} \\ (\dots) \end{array} \stackrel{(ba2^2)}{=} \begin{array}{c} (\dots) \quad (\dots) \\ \text{[yellow square]} \quad \text{[yellow square]} \\ \text{[yellow square]} \quad \text{[yellow square]} \\ (\dots) \quad (\dots) \end{array}
 \end{array} \tag{4.5}$$

Figure 4.5: Qubit rewrite rules that need adjustments for the qudit setting



**Lemma 4.6.1 ([7, Lem. 2.11]):**

$$\begin{array}{c} | \\ \bullet \\ | \end{array} = \begin{array}{c} | \\ \text{yellow box} \\ | \end{array} \quad \text{and} \quad \begin{array}{c} | \\ \bullet \\ | \\ \bullet \\ | \end{array} = \begin{array}{c} | \\ | \end{array}$$

PROOF:

$$\begin{array}{c} | \\ \bullet \\ | \end{array} \stackrel{x}{=} \begin{array}{c} | \\ \text{green box} \\ | \end{array} \stackrel{id}{=} \begin{array}{c} | \\ \text{yellow box} \\ | \end{array}$$

The second part of the lemma then follows immediately from ft. ■

**Lemma 4.6.2 (Pushthrough rules [7, Lem. 2.15, Lem 2.16]):** *The following rules also hold with all colors swapped:*

$$\begin{array}{c} \dots \\ \text{green box} \\ \dots \end{array} = \begin{array}{c} \dots \\ \text{yellow box} \\ \dots \end{array} \quad \text{and} \quad \begin{array}{c} \dots \\ \text{yellow box} \\ \dots \end{array} = \begin{array}{c} \dots \\ \text{green box} \\ \dots \end{array}$$

PROOF: Follow by alternating application of h, x and then ft. ■

We can prove a similar push through rule for *H*-boxes:

**Lemma 4.6.3 (H-pushthrough):**

$$\begin{array}{c} \text{yellow box} \\ \dots \\ \dots \end{array} = \begin{array}{c} \dots \\ \dots \\ \text{yellow box} \end{array}$$

PROOF:

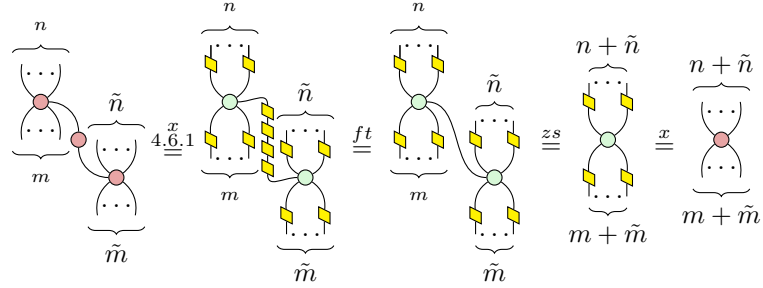
$$\begin{array}{c} \text{yellow box} \\ \dots \\ \dots \end{array} \stackrel{ft}{=} \begin{array}{c} \text{yellow box} \\ \dots \\ \text{yellow box} \end{array} \stackrel{hs}{=} \begin{array}{c} \dots \\ \dots \\ \text{yellow box} \end{array}$$

**Lemma 4.6.4 (X fusion):** *The first of these rules works for an arbitrary, non-zero, number of wires connecting the X-spiders via intermediary spider*

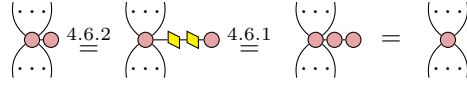
$$\begin{array}{c} \overbrace{\dots}^n \\ \bullet \\ \underbrace{\dots}_m \end{array} \begin{array}{c} \overbrace{\dots}^{\tilde{n}} \\ \bullet \\ \underbrace{\dots}_{\tilde{m}} \end{array} = \begin{array}{c} \overbrace{\dots}^{n+\tilde{n}} \\ \bullet \\ \underbrace{\dots}_{m+\tilde{m}} \end{array} \quad \text{and} \quad \begin{array}{c} \dots \\ \bullet \\ \bullet \\ \dots \end{array} = \begin{array}{c} \dots \\ \bullet \\ \dots \end{array} \quad (\text{xs})$$



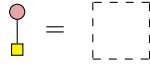
PROOF: For the first equations we have



and for the second one we have

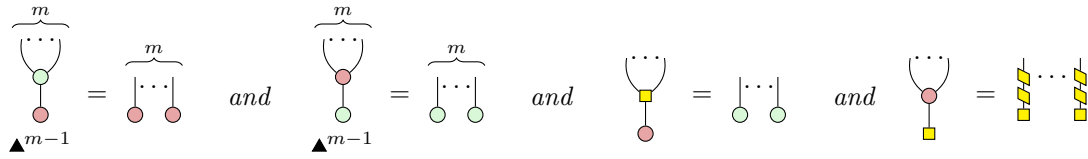


**Lemma 4.6.5 ([7, Lem. 2.5]):**



PROOF: Unchanged from the qubit case via ba2. ■

**Lemma 4.6.6 ([7, Lem 2.21, Lem 2.23, Lem. 2.26]):**



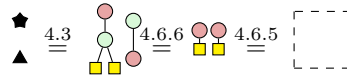
PROOF: Follow immediately from ba1 and ba2. ■

Next up, we investigate some scalar cancellations:

**Lemma 4.6.7 ([7, Lem. 2.3, Lem 2.4]):**



PROOF:



**Remark:** Note that we can also express  $\star$  using just Z- and X-spiders in a slightly more intuitive way via



The proof of this is slightly more complicated, but is quite educational for demonstrating the technique of catalysis. It is based upon the identity

$$\circ \stackrel{zs}{=} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{h}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{x}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{id}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{4.6.6}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{zs}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{4.3}{=} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \triangle \quad (*)$$

We then have

$$\begin{array}{c} \circ \\ | \\ \circ \end{array} \triangle \stackrel{d}{=} \begin{array}{c} \star \\ \star \\ \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \triangle \stackrel{ba1}{=} \begin{array}{c} \star \\ \star \\ \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \\ | \\ \circ \end{array} \stackrel{zs}{=} \begin{array}{c} \star \\ \star \\ \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{(*)}{=} \begin{array}{c} \star \\ \star \\ \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{d}{=} \boxed{\phantom{\circ}} \triangle$$

and adding a  $\triangle$  to both sides yields the desired identity. Furthermore, Booth and Carette [10, Fig 1.] give

$$\star = \begin{array}{c} \circ \\ | \\ \circ \end{array}$$

For this version, we have

$$\triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{id}{=} \triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{zs}{=} \triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{ba1}{=} \triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{d}{=} \triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{4.3}{=} \triangle \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{d}{=} \triangle \boxed{\phantom{\circ}}$$

The following lemma could be seen as a qudit version of the complementarity rule well known in ZX-calculus [7, Lem. 2.30][14, Eq. 9.2]. In fact, if we think of complementarity as the Hopf rule 2.8, then we have found the “true” generalization [23, Lem. 2.8]. However, complementarity also generalizes in a second way (see 4.6.9), which seems to have generally become the accepted generalization [10, Fig. 1]. We thus reserve the name “qudit complementarity” for that version.

**Lemma 4.6.8 (Hopf):**

$$\begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} = \begin{array}{c} \circ \\ | \\ \circ \end{array} \quad (H)$$

PROOF:

$$\begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{4.6.1}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{id}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{zs}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{ba1}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{4.6.6}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{id}{=} \begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \quad \blacksquare$$

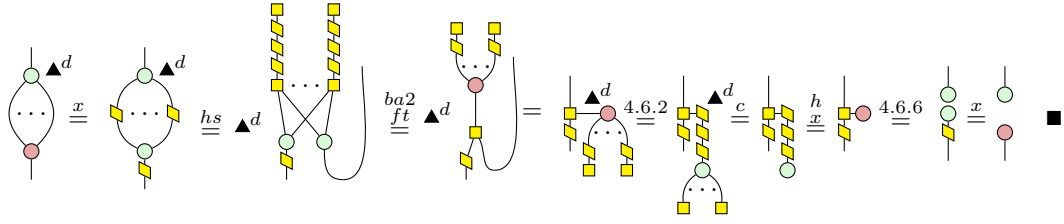
This proof is very similar to the one given by Feng [36, Eq. 4.3] and the abstract one done using the dualizer by Duncan and Dunne [18, Thm. 4.6]<sup>3</sup>. As a rewrite rule, this lemma was already known to Booth and Carette [10, Eq. 21]. In terms of  $\mathbb{Z}/d\mathbb{Z}$ -arithmetic, this lemma simply states “ $x - x = 0$ ”.

**Lemma 4.6.9 (Complementarity [7, Lem 2.30]):**

$$\begin{array}{c} \triangle \\ \triangle \end{array} \begin{array}{c} \circ \\ | \\ \circ \end{array} \stackrel{d}{=} \begin{array}{c} \circ \\ | \\ \circ \end{array}$$

<sup>3</sup>I want to thank John van de Wetering for pointing out these similarities when proof-reading an early draft of this dissertation.

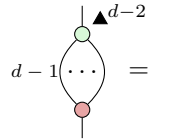
PROOF:



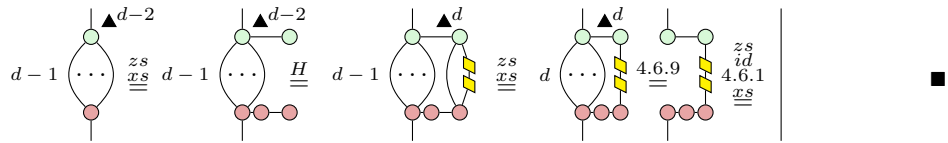
Flipping this rule upside-down, its natural interpretation is that multiplication by  $d$  always results in 0 in  $\mathbb{Z}/d\mathbb{Z}$  (because copying a number  $d$  times and then adding up all those copies is the same as multiplying the number by  $d$ ).

The following lemma is what Yeh and van de Wetering call (SP) for *special* in the qutrit case [49, Fig. 1]:

**Lemma 4.6.10 ([10, Eq. 21]):**



PROOF:

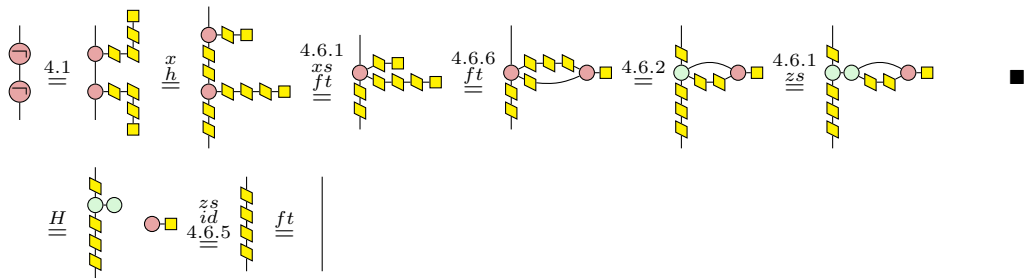


Lastly, we want to document the changed behavior of the red  $\neg$ -spiders:

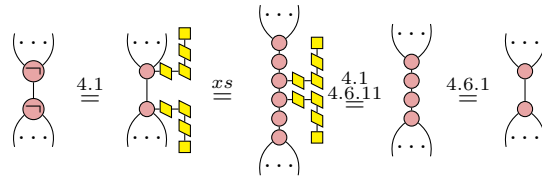
**Lemma 4.6.11 ([7, Lem 2.12, special case]):**



PROOF:



Note that the general form of the lemma, where the red  $\neg$ -spiders have multiple in-/outputs respectively, no longer holds, as

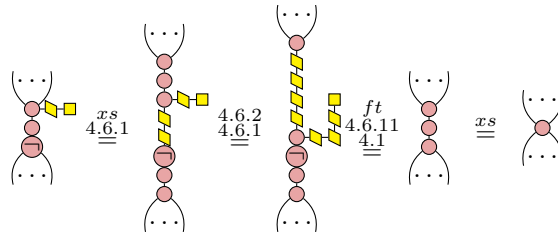


Adding a  $X$ -spider with one in-/output between the two red  $\neg$ -spiders also does not help. However, the following variation does hold:

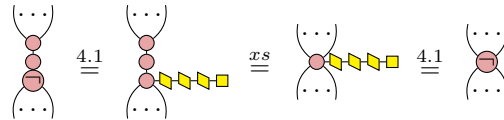
**Lemma 4.6.12:**



PROOF:



and



■

# Chapter 5

## Universality

In this chapter we present two universality results. The first one is for matrices with entries in some arbitrary ring, under the assumption that we have  $H$ -boxes labeled *a priori* with elements  $r \in R$ . The second one is for matrices with integer entries, without the above assumption. Both are valid for dimensions  $d$  prime.

The results of this chapter are based on a novel construction algorithm for ZH-diagrams for binary matrices, which we introduce in the first section. We then use this algorithm to describe normalforms for ZH-diagrams.

### 5.1 Constructing ZH-diagrams for binary matrices in prime dimension

In this section we introduce the most foundational building block of our normalform diagrams, namely an algorithm for constructing ZH-diagrams for linear maps represented in the computational basis by binary matrices. For this, let us first introduce some new jargon:

**Definition 5.1.1:** A matrix  $M = (m_{ij})_{ij}$  is  *$r, s$ -pseudobinary* if  $m_{ij} \in \{r, s\}$  for all  $i, j$ . It is *binary* if it is  $0, 1$ -pseudobinary. Similarly, we call a linear map (pseudo)binary if its presentation matrix regarding the computational basis is (pseudo)binary.

To go from a binary matrix to a ZH-diagram, we perform two intermediary steps:

- 1) Describe the location of the ones in a binary matrix using a logical formula  $\varphi$
- 2) Convert the formula into a polynomial whose roots are exactly the fulfilling assignments of  $\varphi$

Since we know how to express addition and multiplication as ZH-diagrams, turning a polynomial into a diagram is then rather straight-forward.

### 5.1.1 A model-theoretic description of binary matrices

One of the most fundamental facts of linear algebra is that a linear map is uniquely determined by its action on a fixed basis. This property is what allows us to finitely represent a linear map<sup>1</sup>  $L$  of an  $n$ -dimensional vector space as a matrix: For a given basis  $b_1, \dots, b_n$ , write the image  $L(b_i)$  of each  $b_i$  under  $L$  as a linear combination  $\lambda_{1i}b_1 + \dots + \lambda_{ni}b_n$  in the same basis, and then write the coordinates  $\lambda_{1i}, \dots, \lambda_{ni}$  of that linear combination into the  $i$ th column of a matrix. Furthermore, we have

$$\pi_j(L(b_i)) = \lambda_{ji}$$

where  $\pi_j$  is the  $j$ -th coordinate projection.

If the resulting matrix is binary, we have  $\lambda_{ji} \in \{0, 1\}$  for all  $1 \leq i, j \leq n$ . In this case, we can define a relation  $R_L^b \subset \{1, \dots, n\}^2$  as follows:

$$(i, j) \in R_L^b \iff \lambda_{ji} = 1,$$

meaning we have  $(i, j) \in R_L^b$  if the image of  $b_i$  under  $L$  is a linear combination involving  $b_j$ . One could now ask for a propositional formula  $\varphi$  with two free variables  $x, y$  over  $(\{1, \dots, n\}, -, +, \cdot, =)$  such that  $R_L^b$  is the set of all models of  $\varphi$ . Clearly such a formula always exists, e.g.

$$\varphi_L(x, y) = \bigvee_{\substack{i, j=1, \dots, n \\ \lambda_{ji}=1}} (x = i) \wedge (y = j).$$

Let us extend this setting a bit. Instead of an arbitrary basis  $b_1, \dots, b_n$  let us consider the computational basis  $|0\rangle, \dots, |d-1\rangle$  of  $\mathbb{C}^d$ . For linear maps  $L : (\mathbb{C}^d)^{\otimes m} \rightarrow (\mathbb{C}^d)^{\otimes n}$  write

$$L(|\vec{a}\rangle) = \sum_{\vec{b} \in \{0, \dots, d-1\}^n} \lambda_{\vec{b}\vec{a}} |\vec{b}\rangle$$

and assume  $\lambda_{\vec{b}} \in \{0, 1\}$ . In this setting, define  $R_L$  to be a  $(n+m)$ -ary relation over  $\{0, \dots, d-1\}$  such that

$$(\vec{a}, \vec{b}) \in R_L \iff \lambda_{\vec{b}\vec{a}} = 1.$$

Then we look for a formula  $\varphi$  with  $n+m$  free variables such that  $R_L$  is the set of all models of  $\varphi$ . Clearly, we can still very easily construct such a formula for arbitrary  $L$  via

$$\varphi_L(x_1, \dots, x_m, y_1, \dots, y_n) = \bigvee_{\substack{i_1, \dots, i_m \\ j_1, \dots, j_n \\ \in \{0, \dots, d-1\} \\ \lambda_{i_1 \dots i_m j_1 \dots j_n} = 1}} \bigwedge_{k=1}^m (x_k = i_k) \wedge \bigwedge_{\ell=1}^n (y_\ell = j_\ell). \quad (5.1)$$

Sadly, conjunction and disjunction of boolean values are not easily expressible in qudit ZH, meaning this representation of linear maps does not lead to a diagrammatic realization, yet. For that, we rephrase our formula in terms of something we can realize very well as a ZH-diagram: A polynomial over  $\mathbb{Z}/d\mathbb{Z}$ .

---

<sup>1</sup>Here assumed to be an endomorphism

## 5.1.2 An algebraic description of binary matrices

Our construction is based on the following observation:

**Lemma 5.1.2:** *Given a polynomial  $p \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_n]$  in  $n$  variables, we can construct a ZH-diagram that evaluates to 1 on states  $|b_1 \dots b_n\rangle$  such that  $p(b_1, \dots, b_n) = 0$ , and to 0 on all other states.*

PROOF: Let  $p \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_n]$  be a polynomial. We first inductively construct a ZH-diagram realizing the linear map

$$L_p : (\mathbb{C}^d)^{\otimes n} \rightarrow \mathbb{C}^d$$

$$|b_1 \dots b_n\rangle \mapsto |p(b_1, \dots, b_n)\rangle$$

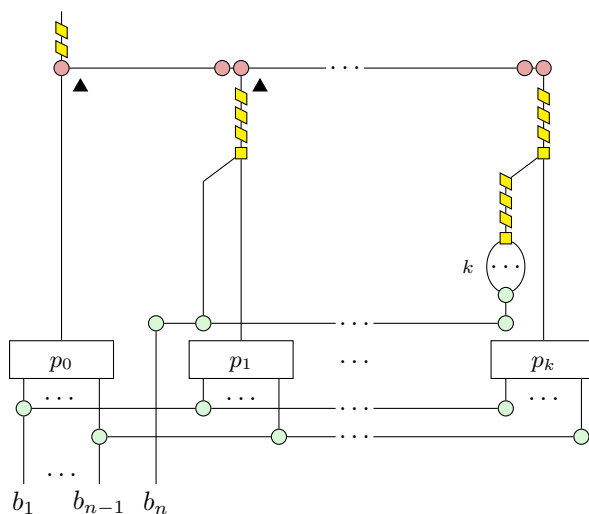
by inducting on the number of variables:

**I.B.:**  $n = 0$ : If  $n = 0$ , then  $p \in \mathbb{Z}/d\mathbb{Z}$  is a constant, which we know how to realize via C.0.2

**I.S.:**  $n - 1 \mapsto n$ : By definition of polynomial rings we can abuse notation slightly to write  $p \in ((\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_{n-1}])[X_n]$ , e.g.

$$p = \sum_{i=0}^k p_i X_n^i \quad \text{for } p_0, \dots, p_k \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_{n-1}].$$

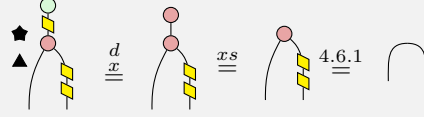
By the inductive hypothesis, we have ZH-diagrams realizing  $p_0, \dots, p_k$ , which we denote by boxes labeled “ $p_0$ ” through “ $p_k$ ”. Then we get the following diagram for  $p$  via 4.4:



To now turn this diagram into an effect that evaluates to 1 on that  $|b_1 \dots b_n\rangle$  such that  $p(b_1, \dots, b_n) = 0$ , and to 0 on all other states, post-select the output by  $|0\rangle$ . ■

Note that often ad-hoc constructions using 4.4 yield simpler diagrams than the inductive algorithm above, particularly as the above diagram is completely unsimplified. Let us look at some simple examples to get a feel for this construction:

EXAMPLE 5.1.3 (IDENTITY MAP): Consider the polynomial  $p(x, y) = x - y$ . It encodes a linear map  $L$  such that  $R_L = \{(x, y) \in (\mathbb{Z}/d\mathbb{Z})^2 \mid x = y\}$ , e.g. the identity. Applying the construction from the above proof, we see that the resulting ZH-diagram indeed represents the identity wire if we bend up one of the inputs:



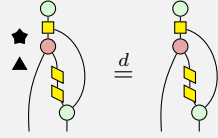
EXAMPLE 5.1.4 (QUDIT ZX TRIANGLE GENERATOR [45, REM. 2.1]): As a slightly more involved example, we construct a ZH-diagram for the triangle generator sometimes defined in qudit ZX-Calculus. Its definition is [45, Rem. 2.1]

$$\text{id} - \sum_{i=0}^{d-1} |0\rangle\langle i|$$

from which we can immediately construct the following formula:

$$\varphi_{\Delta}(x, y) = (x = y) \vee (y = 0)$$

meaning we have to realize the polynomial  $p(x, y) = (x - y)y$ . This is only a very slight adaptation from the previous example and yields



In light of 5.1, this means that if we can find a polynomial  $p \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_m, Y_1, \dots, Y_n]$  such that  $p(x_1, \dots, x_m, y_1, \dots, y_n) = 0 \iff \varphi(x_1, \dots, x_m, y_1, \dots, y_n)$ , we can use 5.1.2 and map-state-duality to construct arbitrary binary linear maps as ZH-diagrams.

**Theorem 5.1.5:** *If  $d$  is prime, then for every propositional formula  $\varphi$  over  $(\{1, \dots, n\}, -, +, \cdot, =)$  in  $n$  free variables there exists a polynomial  $p_{\varphi} \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_n]$  such that*

$$p_{\varphi}(x_1, \dots, x_n) = 0 \iff \varphi(x_1, \dots, x_n).$$

PROOF: Let  $\varphi$  be a formula over  $(\{1, \dots, n\}, -, +, \cdot, =)$  in  $n$  free variables. We describe our polynomial  $p$  inductively. Note that every arithmetic expression in our formula is already a polynomial, since we only allow addition, negation and multiplication in our signature. Thus, we only have to deal with equality, negation, disjunction and conjunction<sup>2</sup>.

<sup>2</sup>Technically we do not need to deal with conjunction, since  $\neg$  and  $\vee$  are functionally complete



1) In the case of  $\varphi = (p_1(x_1, \dots, x_n) = p_2(x_1, \dots, x_n))$  for  $p_1, p_2 \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_n]$ , set

$$p_\varphi = p_1 - p_2$$

2) In the case of  $\varphi = \neg\varphi'$ , set

$$p_\varphi = 1 - (p_{\varphi'})^{d-1}$$

3) In the case of  $\varphi = \varphi_1 \vee \varphi_2$ , set

$$p_\varphi = p_{\varphi_1} \cdot p_{\varphi_2}$$

4) In the case of  $\varphi = \varphi_1 \wedge \varphi_2$  use deMorgan and let

$$\begin{aligned} p_\varphi &= p_{\neg(\neg\varphi_1 \vee \neg\varphi_2)} = 1 - (1 - (p_{\varphi_1})^{d-1})(1 - (p_{\varphi_2})^{d-1}) \\ &= 1 - (1 - (p_{\varphi_1})^{d-1} - (p_{\varphi_2})^{d-1} + (p_{\varphi_1}p_{\varphi_2})^{d-1}) \\ &= (p_{\varphi_1})^{d-1} + (p_{\varphi_2})^{d-1} - (p_{\varphi_1}p_{\varphi_2})^{d-1} \end{aligned}$$

The only non-obvious part of the construction is the construction for negation. For this, recall from C.0.1 that exponentiating with  $d-1$  in  $\mathbb{Z}/d\mathbb{Z}$  maps 0 to 0 and everything else to 1. Lastly, note that the construction in 3) makes use of the absence of zero-divisors in fields.  $\blacksquare$

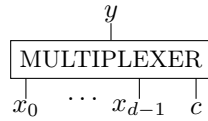
Altogether, this means

**Corollary 5.1.6:** *For prime  $d$ , every binary linear map  $L : (\mathbb{C}^d)^{\otimes m} \rightarrow (\mathbb{C}^d)^{\otimes n}$  has a qudit ZH-diagram realizing  $L$ .*

Furthermore, this gives us two tools to manipulate ZH-diagrams outside of the ZH-Calculus, namely by logical manipulations of a propositional formula, or by algebraic manipulations of a polynomial. These might be useful for determining additional rewrite rules. Conversely, this allows us to reason about polynomials and logical formulae via ZH-Calculus, provided we are able to re-extract those from a modified diagram.

## 5.2 Realizing a multiplexer

In this subsection, our goal is the construction of a *multiplexer*, which is a key part of our normalform diagrams<sup>3</sup>:



A multiplexer is a binary linear map that selects one of the  $d$  first inputs to sent to the output by means of a control input  $|c\rangle$ :

$$\text{MUX}(|x_0 \dots x_{d-1}\rangle \otimes |c\rangle) = |x_c\rangle,$$

or in other words: If  $c = i$ , then  $y = x_i$  for  $i = 0, \dots, d-1$ . This allows us to construct linear maps by means of the fundamental theorem of linear maps:

<sup>3</sup>Multiplexer's have previously been used to describe a variation of the qubit AND-gate in ZW [27, Def. 51]

- 1) Construct some state  $\phi_i$  that represents the  $i$ th column of the matrix of our linear map
- 2) Connect these  $d$  states to the first  $d$  inputs of a multiplexer
- 3) Let  $c$  be the input and  $y$  be the output

Pictorially, this means:

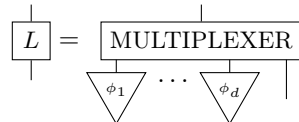


Figure 5.1: Realization of a linear map by specifying its columns as states and feeding them into a multiplexer

Since a multiplexer is a binary linear map, we can describe it by means of the algorithm introduced in the previous section. First, we express it as a propositional formula:

$$\varphi_{mux}(x_0, \dots, x_{d-1}, c, y) = \bigvee_{i=0}^{d-1} (c = i) \wedge (y = x_i).$$

By the construction described in 5.1.5 this yields the polynomial

$$p(x_0, \dots, x_{d-1}, c, y) = \prod_{i=0}^{d-1} ((c - i)^{d-1} + (y - x_i)^{d-1} - (cy - cx_i - iy + ix_i)^{d-1}).$$

This results in the ZH-diagram in Figure 5.4.

### 5.3 Universality for arbitrary rings

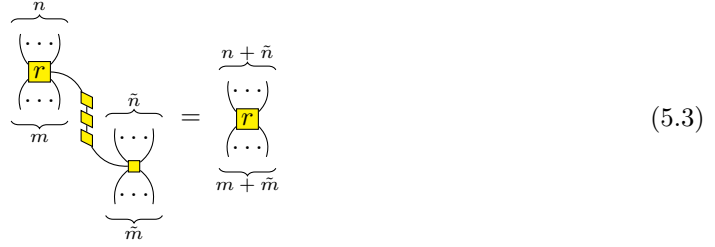
In this section, let  $R \supset \mathbb{Z}[\zeta, \frac{1}{\sqrt{d}}]$  be a commutative unitary ring containing all roots of the polynomial  $X^d - 1$ , as well as  $\sqrt{d}$ . In this chapter we show that by introducing  $H$ -boxes labelled with elements  $r \in R$ , we can realize arbitrary matrices with entries in  $R$  as ZH-diagrams. Keeping in line with Backens et al's notation, we call this calculus  $ZH_R$  [7, Sec. 7].

We formally define these  $H$ -boxes via:

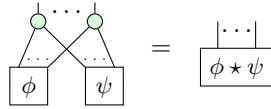
$$\begin{pmatrix} \dots \\ r \\ \dots \end{pmatrix} := \begin{pmatrix} \dots \\ \text{---} \\ \dots \end{pmatrix} \text{---} \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} \text{---} r \quad \text{with} \quad \begin{matrix} | \\ \text{---} \\ | \end{matrix} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} r^i |i\rangle \quad (5.2)$$

Figure 5.2: Labelled  $H$ -boxes in qudit  $ZH_R$

One could interpret the unlabeled  $H$ -box as a  $\zeta$ -labelled one. Furthermore, one easily verifies that the following identity holds:



We now describe a procedure for constructing a normalform for  $ZH_R$ -diagrams by writing them out as *Schur-products*<sup>4</sup>, which we can realize in ZH as follows [7, p. 27]:



For our normalform, we express an arbitrary  $R$ -valued matrix  $M = (m_{ij})$  as a Schur-product of  $r, 1$ -pseudobinary matrices (see 5.1.1), e.g. matrices whose entries are all either 1 or  $r$  for some fixed  $r \in R$ . Namely, for  $r \in R$ , let  $M_r = (m_{ij}^{(r)})$  be the matrix such that

$$m_{ij}^{(r)} = \begin{cases} r & m_{ij} = r \\ 1 & \text{otherwise} \end{cases}. \quad (5.4)$$

Then we have

$$M = \star_{\substack{r \in R \\ M_r \neq \mathbf{1}}} M_r,$$

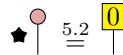
where  $\mathbf{1}$  is the matrix containing only ones ( $M_r = \mathbf{1}$  means that  $M$  has no  $r$ -entries). This Schur product is finite, since clearly a matrix can only have finitely many distinct entries.

Recall from Section 5.1 that we have an algorithm for constructing ZH-diagrams for binary linear matrices  $M$ . The key idea behind this algorithm is the construction of an effect-gadget that evaluates to 1 on inputs  $|x_1 \dots x_m y_1 \dots y_n\rangle$  where  $\pi_{|y_1 \dots y_n\rangle}(M|x_1 \dots x_m\rangle) = 1$ , and to zero otherwise.

We can modify this algorithm to work for  $r, 1$ -pseudobinary matrices as well by making the following slight modification to 5.1.2:

**Corollary 5.3.1:** *Given a polynomial  $p \in (\mathbb{Z}/d\mathbb{Z})[X_1, \dots, X_n]$  in  $n$  variables, we can construct a ZH-diagram that evaluates to 1 on states  $|b_1 \dots b_n\rangle$  such that  $p(b_1, \dots, b_n) = 0$ , and to  $r$  on all other states.*

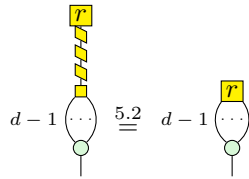
PROOF: In the proof 5.1.2, we post-select the output of the constructed diagram with  $|0\rangle$ . Looking at the definition of labeled  $H$ -boxes, we get



if we use the convention  $0^0 = 1$ . However, just replacing the 0-labelled  $H$ -box with a  $r$ -labelled one does not get the desired result, as we get a gadget that evaluates to  $r^i$  on input  $|i\rangle$ . We thus

<sup>4</sup>entrywise product of matrices of equal dimension

need to somehow first map all  $|i\rangle \neq |0\rangle$  to  $|1\rangle$ . For  $d$  prime, the exponentiator gadget from C.0.1 does exactly this for us, yielding



■

Thus, we can construct  $ZH_R$ -diagrams for each  $M_r$  matrix, and then diagrammatically construct their Schur-product to get a  $ZH_R$ -diagram for  $M$ , proving universality.

## 5.4 Universality for $\mathbb{Z}$

Backens et al [7] show that in the qubit case, ZH is universal for integer-valued matrices even without introducing labeled  $H$ -boxes as new generators. To show this, they construct all integer labeled  $H$ -boxes from realizing that the 0-labelled box is already part of ZX-calculus, and then constructing a successor gadget that increments the label of an arbitrary  $H$ -box by 1. Construction of negative integers is done by finding the inverse of the successor gadget. We follow a similar path of first constructing a successor gadget to increment  $H$ -box labels, but unlike Backens et al we do not provide its inverse, instead realizing that we only need the  $(-1)$ -labeled  $H$ -box to achieve universality.

### 5.4.1 The qudit successor gadget

The qubit successor gadget has a very simple implementation via [7, Eq. 13]:



If we interpret this diagram for qudits,  $d$  prime, instead of qubits, we get the linear map

$$\sum_{\substack{i,j=0 \\ i \neq j}}^{d-1} \zeta^{ij} | -1 - j \rangle \langle i| = \sum_{j=0}^{d-1} |j\rangle \langle 0| + \sum_{i=0}^{d-1} |d-1\rangle \langle i| = \begin{pmatrix} 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & 0 & \dots & 0 \\ 1 & \dots & \dots & 1 \end{pmatrix}.$$

We immediately see that this matrix has rank  $2 < d$  and is thus not invertible. Furthermore, we get

$$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ \vdots \\ 1 \\ d-1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 2d-1 \end{pmatrix} \rightsquigarrow \dots,$$

meaning we do not get anything resembling neither the labeled  $H$ -boxes from the previous section nor any sort of incrementation.

Backens et al note [7, p. 15] that their successor gadget coincides with the qubit triangle generator Wang introduces in his qudit ZX-Calculus [45, Rem. 2.1]. In qudit ZH, we can construct the triangle generator as a ZH-diagram (see 5.1.4, up to a transpose). If we instead use the qudit triangle as our successor, we get

$$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 2 \\ \vdots \\ 2 \end{pmatrix} \rightsquigarrow \dots \rightsquigarrow \begin{pmatrix} 1 \\ a \\ \vdots \\ a \end{pmatrix} \rightsquigarrow \dots$$

which does give us a sensible way of encoding positive integers via an invertible gadget. However, the so-defined  $H$ -boxes do not coincide with those of the previous section.

A successor gadget  $S$  that plays nicely with the previous section's  $H$ -boxes has to satisfy the following equation:

$$S \cdot \begin{pmatrix} 1 \\ a \\ a^2 \\ \vdots \\ a^{d-1} \end{pmatrix} = \begin{pmatrix} 1 \\ a+1 \\ (a+1)^2 \\ \vdots \\ (a+1)^{d-1} \end{pmatrix}.$$

Writing out this linear system of equations, we get

$$\begin{array}{rcccccccc} 1 & = & s_{00} & + & s_{01}a & + & s_{02}a^2 & + & \dots & + & s_{0(d-1)}a^{d-1} \\ a+1 & = & s_{10} & + & s_{11}a & + & s_{12}a^2 & + & \dots & + & s_{1(d-1)}a^{d-1} \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ (a+1)^{d-1} & = & s_{(d-1)0} & + & s_{(d-1)1}a & + & s_{(d-1)2}a^2 & + & \dots & + & s_{(d-1)(d-1)}a^{d-1} \end{array}.$$

Instead of solving this, we can make the following observation:

$$(a+1)^j = \sum_{i=0}^d \binom{j}{i} a^i.$$

Thus, by comparison of coefficients, we get  $s_{ij} = \binom{j}{i}$  with the convention  $\binom{i}{j} = 0$  for  $i > j$ . This means that the matrix  $S$  encodes Pascal's triangle in form of a lower triangular matrix.

Our task is therefore to somehow construct a ZH-diagram for  $S$  using only binary matrices (as these are the only type of matrices we know how to implement using just the zero-labeled  $H$ -box, see Section 5.1). For this, first consider the following matrix

$$R = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ 1 & 1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 & 1 \end{pmatrix}.$$

When repeatedly applying it to  $|0\rangle$  we get

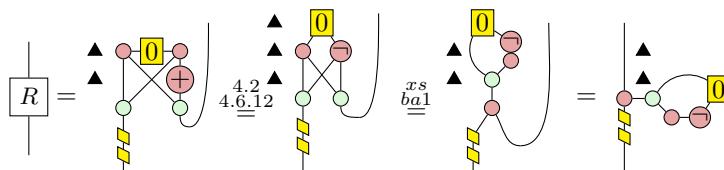
$$\begin{pmatrix} 1 \\ 0 \\ \vdots \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 2 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 3 \\ 3 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \rightsquigarrow \dots$$

meaning  $R^i|0\rangle$  for  $0 \leq i < d$  results in a state representing the  $(i + 1)$ th-row of Pascal's triangle. Since  $R$  is a binary matrix, we know how to construct a ZH-diagram for it. We can simplify  $R$  slightly by adding a 1-entry to the top right corner. This additional entry will only have an effect if we apply the matrix to a vector that has a non-zero  $d$ th entry. However, since we only use  $R$  to compute rows of Pascal's triangle, this is only the case after the  $(d - 1)$ th application. As past the  $(d - 1)$ th iteration a  $d$ -dimensional vector is too small to hold an entire row of Pascal's triangle anyway, we conclude that this inaccuracy is irrelevant.

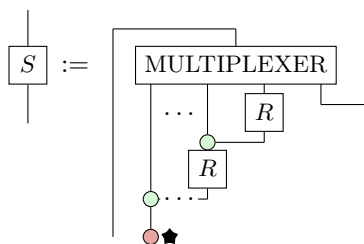
This allows us to encode the location of the 1-entries in  $R$  as

$$\varphi_R(x, y) = (y = x) \vee (y = x + 1),$$

for which we get the following diagram:



We can use the multiplexer from Section 5.2 to write these vectors into the columns of a matrix, yielding a ZH-diagram realizing the matrix that has the  $i$ th row of Pascals triangle in its  $i$ th column. A simple transpose then yields what we want:



Using this we can realize any integer labeled  $H$ -box where the label is non-negative. This already allows us to construct arbitrary  $\mathbb{N}_0$ -valued matrices. To get all integer labeled  $H$ -boxes, we construct  $-1$  in the next subsection.

## 5.4.2 Negative integers

Backens et al construct negative integers by applying a green  $\neg$ -spider to a positively labeled  $H$ -box [7, Eq. 16]. In the qudit setting, this does no longer work, as Pauli-Z is the matrix



Utilizing this gadget to bootstrap our  $(-1)$ -labeled  $H$ -box then allows us to construct  $\text{sg}(M)$  for arbitrary integer-valued matrices  $M$ , and thus yields universality of the qudit ZH-Calculus for integer-valued matrices in prime dimension.

Additionally, we get the following corollary

**Corollary 5.4.4 (Scalar  $-1$ ):**

$$-1 = \text{[Diagram: A vertical chain of yellow squares connected by a vertical line, with a green circle at the bottom and a red circle at the top. The red circle is labeled with an ellipsis and  $d-1$ .]}_{d-1}$$

PROOF: Clearly we have

$$-1 \stackrel{5.4.3}{=} \text{[Diagram: A vertical chain of yellow squares with a red circle at the top containing a plus sign and a triangle above it.]} \stackrel{4.2}{=} \text{[Diagram: A vertical chain of yellow squares with a red circle at the top containing a minus sign and a star above it.]} \stackrel{4.1}{=} \frac{f}{d} \text{[Diagram: A vertical chain of yellow squares with a red circle at the top containing a plus sign and a triangle above it.]} \stackrel{4.6.1}{=} \frac{xs}{ft} \text{[Diagram: A vertical chain of yellow squares with a red circle at the top containing a plus sign and a triangle above it.]}_{d-1}$$

where all “...” represent  $(d - 1)$ -fold repetition. ■

This means that in qudit ZH, we do not need to adjoin  $-1$  as an additional scalar, as is required for qudit ZX [10, Sec. 2.2].

## 5.5 Considerations for $d$ not prime

All of the results of this section so far are only valid for  $d$  prime, meaning for the case where  $\mathbb{Z}/d\mathbb{Z}$  is a field. The obvious question is how much of our machinery translates over to the case of  $d$  not prime. In the finite case, the notions “field” and “integral domain” are equivalent [5, Ch. 11, Ex 7.1], this means that for  $d$  not prime, the ring  $\mathbb{Z}/d\mathbb{Z}$  admits zero-divisors. And indeed, since  $\mathbb{Z}/d\mathbb{Z}$  contains all prime factors of  $d$ , we can simply multiply them all to write 0 as a non-trivial product.

This presence of zero-divisors has profound consequences for polynomial rings over  $\mathbb{Z}/d\mathbb{Z}$ , as we lose the ability to uniquely factor them based on their roots. For example, in  $\mathbb{Z}/6\mathbb{Z}$  consider the polynomial  $f = X^2 + X$ . Clearly, 0 is a root, and so is 5 because we can write  $f = X(X + 1)$ . However, since  $6 = 2 \cdot 3$ , we also get the root 2, and  $12 = 3 \cdot 4$  implies 3 to also be a root. Furthermore, we get a second representation of  $f$  as a product of linear factors via  $f = (X + 2)(X + 3)$ .

This makes polynomial interpolation over  $\mathbb{Z}/d\mathbb{Z}$  impossible in the general case, as above we have shown that any polynomial with roots 1 and 5 also necessarily has roots 2 and 3. Thus we cannot interpolate datasets of the form  $\{(1, 0), (2, 1), (5, 0), \dots\}$ .



Since our construction of ZH-diagrams for (pseudo)binary matrices heavily relies on this interpolatability (via 5.1.5), a different approach for constructing normal forms needs to be found, potentially via a direct translation of formulae to ZH-diagrams without using polynomials.

## 5.6 The normalform generalizes the qubit normalform

In this section we briefly discuss how our qudit ZH normalform is a generalization of the qubit ZH normalform Backens et al use to prove completeness [7, Sec. 4]. Note that for clarity, we omit scalars in this section, since Backens et al use different normalizations for their generators.

Recall from Equation 5.4 that our normalform is based on taking the Schur-product of  $1, r$ -pseudobinary matrices. However, unlike Backens et al, our pseudobinary matrices have more than one non-1 entry, whereas Backens et al break apart our  $M_r$  matrices further into a Schur product of matrices containing only a single  $r$ -entry. For such a matrix there exists only a single pair of computational basis states  $|i_1 \dots i_m\rangle$  and  $|j_1, \dots, j_n\rangle$  such that  $\lambda_{i_1, \dots, i_m, j_1, \dots, j_n} = r$ . Thus, we can simplify Equation 5.1 to

$$\varphi(x_1, \dots, x_m, y_1, \dots, y_n) = \bigvee_{k=1}^m (x_k \neq i_k) \vee \bigvee_{\ell=1}^n (y_\ell \neq j_\ell),$$

as every entry not containing the  $r$  will contain a 1.

Our construction algorithm turns this into the following polynomial

$$p(x_1, \dots, x_m, y_1, \dots, y_n) = \prod_{k=1}^m (1 - (x_k - i_k)^{d-1}) \cdot \prod_{\ell=1}^n (1 - (y_\ell - j_\ell)^{d-1}),$$

which yields the following ZH-diagram (wlog assuming  $n = 0$  as we can always use map-state duality to get row-matrix and thus a diagram with no outputs):

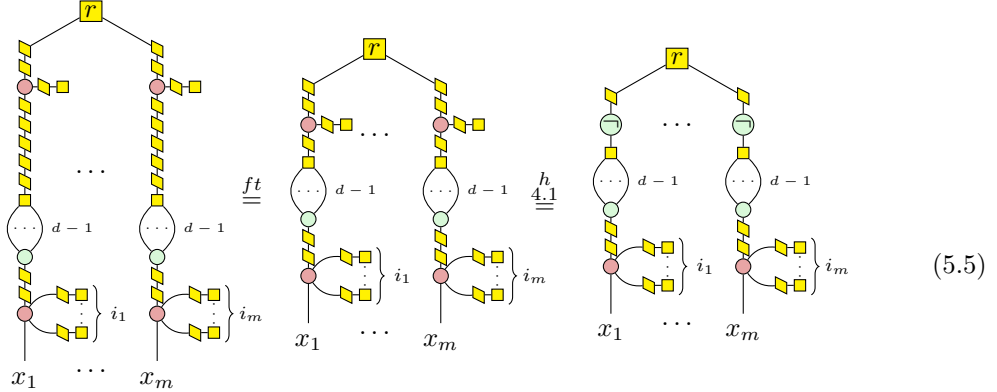
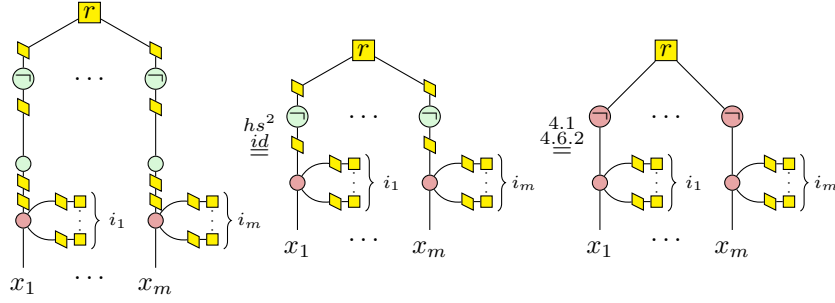


Figure 5.3: Qudit normalform of a matrix that has a single  $r$ -entry and is filled with 1s otherwise

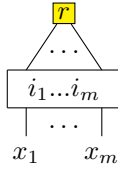
Let us interpret this for the case of  $d = 2$ . Here we have:



Additionally, the  $i_k$  are either 0 or 1, meaning the basis-shift gadgets simplify as follows:

$$\begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \quad \text{and} \quad \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array}$$

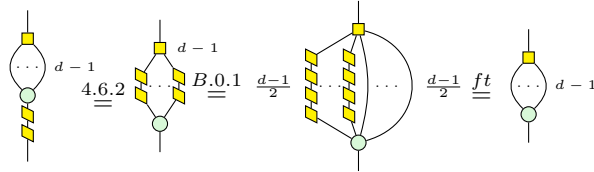
Note how we still have the red negate spiders at the top. Since these cancel in pairs (4.6.11), our diagram becomes



where the “ $i_1 \dots i_m$ ”-labelled box is exactly the indexing gadget Backens et al define [7, Def. 2.38]. Thus, the normalforms coincide up to a transpose.

However, due to the indexing gadget being significantly more complicated, finding a set of rewrites that turns qudit ZH-generators into normalform seems quite difficult. Nevertheless, we wish to make some concluding remarks about this indexing gadget:

- 1) The indexing gadget still copies through  $Z$ -spiders (we provide a proof of this in Appendix D). This seems to imply that the general rewrite strategies for subsuming generators into normalform diagrams Backens et al use are still valid for qudit ZH.
- 2) Since  $d$  is a prime number,  $d$  must be odd if its greater than 2. Thus  $d - 1$  is even and we get



Obviously, for  $d = 2$ , the two hadamards cancel anyway.

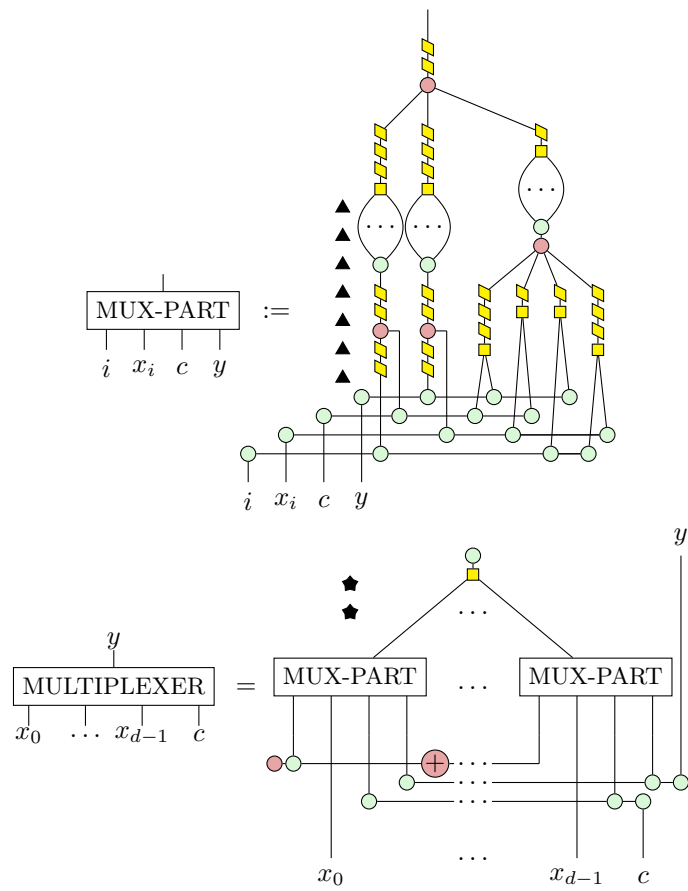


Figure 5.4: Realization of a multiplexer in qudit ZH. Here we assume the Pauli-X inputs to be on the left

# Chapter 6

## Conclusion

In this thesis we spearhead the development of a general qudit ZH-calculus. The main challenge is to generalize the qubit  $H$ -box from the qubit setting. To do this, we generalize a 2020 classification of qubit quantum graphical calculi provided by Carette and Jeandel [12]. This classification enumerates all qubit quantum graphical calculi satisfying some reasonable conditions such as being flex-symmetric and allowing a bialgebra rule [12, Thm. 20]. They achieve this result by describing the spiders of quantum graphical calculi as Frobenius-algebras in a strict symmetric monoidal category, and then classifying these algebras. They also find generalizations of the ZX- and ZW-algebras to the qudit setting, but note that any attempt at generalizing ZH resulting in the resulting Frobenius-algebra violating the *compatibility* condition (2.7), which we show to be unnecessary.

At the end of Chapter 2 we show that the trade-off of dropping the compatibility condition are slightly changed bialgebra and spider fusion rules 2.3.2. This has further implications for Carette and Jeandel's classification [12, Thm. 20], as without compatibility, all phase-shifts of valid quantum graphical calculi again yield quantum graphical calculi, potentially changing the bialgebra and fusion rules to use the inverse dualizer.

After covering these classification results, we set out to actually construct our qudit  $H$ -box in Chapter 3, based on the observation that we can generalize logical AND to a dit operation by considering multiplication modulo  $d$ . In Chapter 4 we then enrich our graphical language with rewrite rules, yielding the titular *qudit ZH-calculus*. These rewrite rules are generalizations of those proposed by Backens et al in their completeness paper for qubit ZH-calculus [7]. We also discuss generalizations of further rewrite rules that are very specific to Backens et al's completeness result [7, Fig. 1, (o), Sec. 8.2], but which we do not consider fundamental enough to include in our preliminary rule set. Furthermore, we specialize our qudit results to the qubit case to show that here, the double hadamard cancellation rule (hh) is subsumed by the other rules. We conclude the chapter by deriving widely known equations, to justify our choice of rewrite rules.

After providing a detailed description of the qudit ZH-calculus, we turn towards proving its universality for quantum systems of prime dimension in Chapter 5. Similarly to Backens et al's result, we first show that by a priori introducing labelled  $H$ -boxes, we get a calculus that, under reasonable assumptions, is universal for the ring  $R$  from which we choose the labels. Then we specifically consider the case  $R = \mathbb{Z}[\zeta, 1/\sqrt{d}]$  of integer-valued matrices and show that here we

do not need to adjoin labelled  $H$ -boxes to our language to achieve universality. The normalform we provide is a generalization of Backens et al’s normalform [7, Sec. 3]. To construct the normal form, we devise a novel construction algorithm for ZH-diagrams for binary matrices.

## 6.1 Novel contributions

Novel contributions of this dissertations are

- the observation that Carette and Jeadels’ compatibility condition [12, Def. 19] is unnecessary for the construction of flexsymmetric quantum graphical calculi, and that phase-shifting a quantum graphical calculus does not break any of its fundamental properties (2.3.2),
- the construction of a flexsymmetric qudit  $H$ -box (Chapter 3) and (to a lesser extend, as we simply generalize what is already known for the qubit case) the surrounding calculus (Section 4.2)
- that for *qubit* ZH, the (hh) rule of Backens et al’s completeness result [7] is redundant (Section 4.3, Corollary 4.5.1),
- the proof that qudit ZH-calculus is universal for integer-valued matrices, and that  $ZH_R$  calculus is universal for  $R$ -valued matrices (Chapter 5),
- the construction algorithm for ZH-diagrams of binary linear maps (Section ??).

## 6.2 Future Work

In this section, we outline some possible directions for future research regarding the qudit ZH-calculus, as well as some ideas that we were not able to explore in the limited scope of this Master’s dissertation.

**Universality** The calculus we present in this dissertation is only universal for prime dimension  $d$ . This is due to our normalform construction relying on polynomial interpolation modulo  $d$  5.1.5, which is impossible if  $\mathbb{Z}/d\mathbb{Z}$  admits zero-divisors (Section ??). Thus, for  $d$  not prime, a different approach is needed to achieve universality. Wang has previously described normalforms for the qudit ZX-calculus via elementary matrices and reverse Gaußian elimination [43]. This approach works without any restrictions on the dimension  $d$ , and even yields universality for arbitrary semi-rings  $R$ . Thus we ask for a similar normalform for ZH-diagrams.

**Completeness** As we discuss at the end of Section 4.2, our ruleset here is provably incomplete. The main issue here is that we do not have any rules for reasoning about the multiplicative group  $(\mathbb{Z}/d\mathbb{Z})^*$ . For  $d = 2$ , all information about the multiplicative group is stored in the hadamard cancellation rule  $\text{ft}^2$ , which encodes the statement “ $1 = -1$ ”. However, Booth and Carette’s completeness result for stabilizer qudit ZX [10] suggests that a rule schema that encodes multiplicative structure even just for prime dimensions (where the multiplicative groups are cyclic) might not be very straightforward.

In Appendix D we briefly discuss a few difficulties that arise when trying to generalize Backens et al’s completeness results, mainly stemming from our significantly more complex indexing gadget used in our normalform diagrams. It would be interesting to see if these could be overcome.

Additionally, qubit ZH allows for a class of hypergraph rewrite rules generalizing *local complementation* and *pivoting* from ZX-calculus [19][34]. Lemonnier et al conjecture that these hypergraph operations could yield an alternate path towards completeness for the ZH-calculus, which should be investigated.

**Relation to qudit ZX** Our qudit ZH-calculus as presented here does not subsume the qudit ZX-calculi presented by Wang [42] or Booth and Carette [11], as we do not allow for phases on our  $Z$ -spiders. Finding translations between the qudit ZX- and ZH-calculi, as was done by Wang for the qubit case [44], might yield new insights and might allow for a completeness proof by translation, as was done for the first completeness proofs for the qubit ZX-calculus (which were translated from the ZW-calculus) [29]. One difficulty in finding such a translation might be that the following equation [21, p. 27, top right]

$$\textcircled{\alpha} = \boxed{e^{i\alpha}}$$

does no longer hold, as qudit ZX phases are vectors, whereas  $H$ -boxes continue to be labelled by scalars.

Furthermore, for the qubit ZX- and ZH-calculi, there exists a Fourier-type relation for translating between diagrams of the two calculi, due to Kuijpers et al [32]. Van de Wetering and Yeh conjecture that this connection could lead to a definition of *qutrit*  $H$ -boxes via phase multipliers [49]. It would be interesting to see whether these correspondences generalize for our categorically constructed  $H$ -boxes, or whether they give rise to an alternate ZH-calculus. A potential approach for generalizing this relation is to replace the boolean Fourier transform Kuijpers et al us with the Fourier transform of a finite abelian group, such as  $\mathbb{Z}/d\mathbb{Z}$ .

Another approach towards translating ZH-diagrams into ZX-diagrams could be to exploit the relation<sup>1</sup>  $(\mathbb{Z}/d\mathbb{Z})^*, \cdot) \cong (\mathbb{Z}/(d-1)\mathbb{Z}, +)$  for prime  $d$  by simulating addition modulo  $d-1$  in  $d$ -dimensional space, and permuting the computational basis according to the isomorphism.

Lastly, the brute-force variant would be to use the universality of Wang’s qudit ZX-calculus to construct the normalform diagram for the  $H$ -box.

**Qudit controlled unitaries** The original rationale for researching the qudit ZH-calculus was to investigate the simulation of qubit gates in higher dimension further. Of specific interest here is the Toffoli-gate (which has a very simple qubit ZH-representation vs. its very complicated ZX-representation) and its higher-dimensional generalizations. Some investigations into this for the qutrit case have been done by Yeh and van de Wetering [50], using quantum circuit notation. We are interested in seeing if qudit ZH could be employed here to simplify and generalize their results. We want to note that a simple reinterpretation of qubit ZH-diagram for the Toffoli-gate (1.4) generalizes the notion of qutrit control introduced by Bocharov et al [9].

Lastly we want to note that the multiplexer we construct in Section 5.2 trivially generalizes to

<sup>1</sup>The multiplicative group of a finite field is cyclic, and cyclic groups of a given order are unique up to isomorphism. Since  $|(\mathbb{Z}/d\mathbb{Z})^*| = d-1 = |\mathbb{Z}/(d-1)\mathbb{Z}|$ , these groups have to be isomorphic

multiple controls, and that multiple multiplexers can be used to multiplex multiple qubits. Then it is trivial to write down ZH-diagrams for controlled unitaries.

# Bibliography

- [1] Scott Aaronson. “The limits of quantum”. In: *Scientific American* 298.3 (2008), pp. 62–69.
- [2] Scott Aaronson and Daniel Gottesman. “Improved simulation of stabilizer circuits”. In: *Phys. Rev. A* 70 (5 Nov. 2004), p. 052328. DOI: 10.1103/PhysRevA.70.052328.
- [3] Scott Aaronson, Daniel Grier, and Luke Schaeffer. “A Quantum Query Complexity Trichotomy for Regular Languages”. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. 2019, pp. 942–965. DOI: 10.1109/FOCS.2019.00061.
- [4] Matthew Amy, Andrew N. Glaudell, and Neil J. Ross. “Number-Theoretic Characterizations of Some Restricted Clifford+T Circuits”. In: (Aug. 2019). DOI: 10.22331/q-2020-04-06-252.
- [5] M. Artin. *Algebra*. Pearson Prentice Hall, 2011. ISBN: 9780132413770.
- [6] Miriam Backens. “The ZX-calculus is complete for stabilizer quantum mechanics”. In: *New Journal of Physics* 16.9 (Sept. 2014), p. 093021. DOI: 10.1088/1367-2630/16/9/093021.
- [7] Miriam Backens et al. *Completeness of the ZH-calculus*. Mar. 2021. DOI: 10.48550/arXiv.2103.06610.
- [8] Niel de Beaudrap. “Well-tempered ZX and ZH Calculi”. In: (June 2020). DOI: 10.4204/EPTCS.340.2.
- [9] Alex Bocharov, Martin Roetteler, and Krysta M. Svore. “Factoring with qutrits: Shor’s algorithm on ternary and metaplectic quantum architectures”. In: *Phys. Rev. A* 96 (1 July 2017), p. 012306. DOI: 10.1103/PhysRevA.96.012306. URL: <https://link.aps.org/doi/10.1103/PhysRevA.96.012306>.
- [10] Robert I. Booth and Titouan Carette. “Complete ZX-calculi for the stabiliser fragment in odd prime dimensions”. In: (Apr. 2022). DOI: 10.48550/arxiv.2204.12531.
- [11] Titouan Carette. “When Only Topology Matters”. In: (Feb. 2021). DOI: 10.48550/arxiv.2102.03178.
- [12] Titouan Carette and Emmanuel Jeandel. “On a recipe for quantum graphical languages”. In: (Aug. 2020). DOI: 10.4230/LIPIcs.ICALP.2020.118.
- [13] Bob Coecke and Ross Duncan. “A graphical calculus for quantum observables”. In: *Preprint* (2007). URL: <http://www.cs.ox.ac.uk/people/bob.coecke/GreenRed.pdf>.
- [14] Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017. DOI: 10.1017/9781316219317.
- [15] David Deutsch. “Quantum theory, the Church–Turing principle and the universal quantum computer”. In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 400.1818 (1985), pp. 97–117. DOI: <https://doi.org/10.1098/rspa.1985.0070>.



- [16] David Deutsch and Richard Jozsa. “Rapid solution of problems by quantum computation”. In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (1992), pp. 553–558. DOI: <https://doi.org/10.1098/rspa.1992.0167>.
- [17] David Elieser Deutsch. “Quantum computational networks”. In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 425.1868 (1989), pp. 73–90. DOI: <https://doi.org/10.1098/rspa.1989.0099>.
- [18] Ross Duncan and Kevin Dunne. “Interacting Frobenius Algebras are Hopf”. In: (Jan. 2016). DOI: [10.48550/arxiv.1601.04964](https://doi.org/10.48550/arxiv.1601.04964).
- [19] Ross Duncan and Simon Perdrix. “Pivoting makes the ZX-calculus complete for real stabilizers”. In: (July 2013). DOI: [10.4204/eptcs.171.5](https://doi.org/10.4204/eptcs.171.5).
- [20] Ross Duncan et al. “Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus”. In: *Quantum* 4 (June 2020), p. 279. ISSN: 2521-327X. DOI: [10.22331/q-2020-06-04-279](https://doi.org/10.22331/q-2020-06-04-279).
- [21] Richard D.P. East et al. “AKLT-States as ZX-Diagrams: Diagrammatic Reasoning for Quantum States”. In: *PRX Quantum* 3 (1 Jan. 2022), p. 010302. DOI: [10.1103/PRXQuantum.3.010302](https://doi.org/10.1103/PRXQuantum.3.010302).
- [22] Brett Giles and Peter Selinger. “Exact synthesis of multiqubit Clifford+T circuits”. In: (Dec. 2012). DOI: [10.1103/physreva.87.032332](https://doi.org/10.1103/physreva.87.032332).
- [23] Xiaoyan Gong and Quanlong Wang. “Equivalence of Local Complementation and Euler Decomposition in the Qutrit ZX-calculus”. In: (Apr. 2017). DOI: [10.48550/arxiv.1704.05955](https://doi.org/10.48550/arxiv.1704.05955).
- [24] Daniel Gottesman. “An introduction to quantum error correction and fault-tolerant quantum computation”. In: *Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics*. Vol. 68. 2010, pp. 13–58. DOI: [10.1090/psapm/068/2762145](https://doi.org/10.1090/psapm/068/2762145).
- [25] Lov K. Grover. “Quantum Mechanics Helps in Searching for a Needle in a Haystack”. In: *Phys. Rev. Lett.* 79 (2 July 1997), pp. 325–328. DOI: [10.1103/PhysRevLett.79.325](https://doi.org/10.1103/PhysRevLett.79.325).
- [26] Amar Hadzihasanovic. *The algebra of entanglement and the geometry of composition*. 2017. DOI: [10.48550/arXiv.1709.08086](https://doi.org/10.48550/arXiv.1709.08086).
- [27] Michael Herrmann. “Models of Multipartite Entanglement”. In: (2010). URL: <http://www.cs.ox.ac.uk/people/bob.coecke/Michael.pdf>.
- [28] Anne Hillebrand. “Quantum Protocols involving Multiparticle Entanglement and their Representations in the zx-calculus”. PhD thesis. University of Oxford, 2011. URL: <http://www.cs.ox.ac.uk/bob.coecke/Anne.pdf>.
- [29] 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. Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 559–568. ISBN: 9781450355834. DOI: [10.1145/3209108.3209131](https://doi.org/10.1145/3209108.3209131).
- [30] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. “Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics”. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS ’18. Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 569–578. ISBN: 9781450355834. DOI: [10.1145/3209108.3209139](https://doi.org/10.1145/3209108.3209139).

- [31] R. Jozsa. “Quantum factoring, discrete logarithms, and the hidden subgroup problem”. In: *Computing in Science & Engineering 3.2* (2001), pp. 34–43. DOI: 10.1109/5992.909000.
- [32] Stach Kuijpers, John van de Wetering, and Aleks Kissinger. “Graphical Fourier Theory and the Cost of Quantum Addition”. In: (Apr. 2019). DOI: 10.48550/arxiv.1904.07551.
- [33] Serge Lang. “Galois Theory”. In: *Algebra*. New York, NY: Springer New York, 2002, pp. 261–332. ISBN: 978-1-4613-0041-0. DOI: 10.1007/978-1-4613-0041-0\_6.
- [34] Louis Lemonnier, John van de Wetering, and Aleks Kissinger. “Hypergraph Simplification: Linking the Path-sum Approach to the ZH-calculus”. In: (Mar. 2020). DOI: 10.4204/eptcs.340.10.
- [35] Hector Miller-Bakewell and John van de Wetering. URL: <http://zxcalculus.com/>.
- [36] Kang Feng Ng. *Completeness of the ZW and ZX calculi*. 2018. URL: <https://ora.ox.ac.uk/objects/uuid:aefeb7a7-275b-4712-b8eb-0727e7ead04e>.
- [37] John Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum 2* (Aug. 2018), p. 79. DOI: 10.22331/q-2018-08-06-79.
- [38] André Ranchin. “Depicting qudit quantum mechanics and mutually unbiased qudit theories”. In: (Apr. 2014). DOI: 10.4204/eptcs.172.6.
- [39] Yaoyun Shi. “Both Toffoli and Controlled-NOT need little help to do universal quantum computation”. In: (May 2002). DOI: 10.48550/arxiv.quant-ph/0205115.
- [40] P.W. Shor. *Algorithms for quantum computation: discrete logarithms and factoring*. 1994. DOI: 10.1109/SFCS.1994.365700.
- [41] E Study. “Über Systeme complexer Zahlen und ihre Anwendung in der Theorie der Transformationsgruppen”. In: *Monatshefte für Mathematik und Physik 1* (1890), pp. 283–354. DOI: <https://doi.org/10.1007/BF01692479>.
- [42] Quanlong Wang. *A non-anyonic qudit ZW-calculus*. 2021. DOI: 10.48550/arXiv.2109.11285.
- [43] Quanlong Wang. “Algebraic complete axiomatisation of ZX-calculus with a normal form via elementary matrix operations”. In: (July 2020). DOI: 10.48550/arxiv.2007.13739.
- [44] Quanlong Wang. “An Algebraic Axiomatisation of ZX-calculus”. In: (Nov. 2019). DOI: 10.4204/eptcs.340.16.
- [45] Quanlong Wang. *Qufinite ZX-calculus: a unified framework of qudit ZX-calculi*. 2022. DOI: 10.48550/arXiv.2104.06429.
- [46] Quanlong Wang. “Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics”. In: (Mar. 2018). DOI: 10.4204/EPTCS.266.3.
- [47] Quanlong Wang and Xiaoning Bian. “Qutrit Dichromatic Calculus and Its Universality”. In: (June 2014). DOI: 10.4204/EPTCS.172.7.
- [48] John van de Wetering. “ZX-calculus for the working quantum computer scientist”. In: (Dec. 2020). DOI: 10.48550/arxiv.2012.13966.
- [49] John van de Wetering and Lia Yeh. “Phase gadget compilation for diagonal qutrit gates”. In: (Apr. 2022). DOI: 10.48550/arxiv.2204.13681.
- [50] Lia Yeh and John van de Wetering. “Constructing all qutrit controlled Clifford+T gates in Clifford+T”. In: (Apr. 2022). DOI: 10.48550/arxiv.2204.00552.

# Appendix A

## Roots of unity

In this chapter we very briefly cover the algebraic theory of roots of unity over some field  $F$ . For an in-depth discussion we refer the reader to any standard textbook on abstract algebra, such as Lang [33].

### A.1 Introduction

Let  $F$  be a field and  $n \in \mathbb{N}$ . We call some  $\zeta$  in the algebraic closure of  $F$  a *n-th root of unity* in  $F$  if it is a root of the polynomial  $f = X^n - 1$ , or in other words: if  $\zeta^n = 1$ . The following basic result about roots of unity can be found in any standard undergraduate course on abstract algebra:

**Lemma A.1.1 ([33, §IV.3]):** *The n-th roots of unity form a cyclic group that has order n if char F does not divide n.*

We call a generator of this cyclic group a *primitive root of unity*. Since we are working exclusively with roots of unity over  $\mathbb{C}$ , let us take a look at their specific properties

EXAMPLE A.1.2: In  $\mathbb{C}$  the  $n$ -th roots of unity are precisely  $\zeta_k = e^{\frac{2\pi ik}{n}}$  for  $k = 0, \dots, n - 1$ . Additionally, we have

$$\zeta_k^{-1} = \bar{\zeta}_k$$

and these are again roots of unity (since if  $x$  is a root of some polynomial  $f$ , then so is its complex conjugate).

**Lemma A.1.3:** *Let  $\zeta \neq 1$  be an n-th root of unity of F. Then*

$$\zeta^{n-1} + \zeta^{n-2} + \dots + \zeta + 1 = 0.$$

PROOF: Factor  $f$  as

$$X^n - 1 = (X^{n-1} + X^{n-2} + \dots + X + 1)(X - 1).$$

Since  $\zeta \neq 1$  is a root of  $f$ , yet it is not a root of  $X - 1$ , it has to be a root of  $X^{n-1} + \dots + X + 1$ , as fields have no zero divisors. Thus, the lemma follows.  $\blacksquare$

In practice, we use this lemma as follows:

**Corollary A.1.4:** *Let  $I \neq \emptyset$  be some finite index set,  $f : I \rightarrow \mathbb{Z}$  some function and  $\zeta \neq 1$  a primitive root of unity. Then*

$$\sum_{i \in I} \sum_{k=0}^{n-1} (\zeta^{f(i)})^k = \sum_{\substack{i \in I \\ f(i) \equiv_n 0}} n.$$

PROOF: Since the  $n$ -th roots of unity form a cyclic group and  $\zeta$  is a generator, we have  $\zeta^{f(i)} = 1$  iff  $n \mid f(i)$ . By A.1.3 the inner summation is 0 whenever  $\zeta^{f(i)} \neq 1$ .  $\blacksquare$

**Corollary A.1.5:** *In the setting of A.1.4, we  $g : \mathbb{Z}/n\mathbb{Z} \rightarrow V$  be some function where  $V$  is some  $\mathbb{F}_d$ -vector space. If  $\text{char } F$  does not divide  $n$ , we have*

$$\sum_{i \in I} \sum_{y=0}^{n-1} \sum_{k=0}^{n-1} (\zeta^{f(i)-y})^k g(y) = n \cdot \sum_{i \in I} g(f(i)).$$

PROOF: Apply A.1.4 to get

$$\sum_{i \in I} \sum_{y=0}^{n-1} \sum_{k=0}^{n-1} (\zeta^{f(i)-y})^k g(y) = \sum_{y=0}^{n-1} g(y) \sum_{i \in I} \sum_{k=0}^{n-1} (\zeta^{f(i)-y})^k \stackrel{\text{A.1.4}}{=} \sum_{y=0}^{n-1} g(y) \sum_{\substack{i \in I \\ f(i)-y \equiv_n 0}} n.$$

Rewriting this slightly using indicator variables we get

$$\sum_{y=0}^{n-1} g(y) \sum_{\substack{i \in I \\ f(i)-y \equiv_n 0}} n = n \cdot \sum_{i \in I} \sum_{y=0}^{n-1} \mathbb{1}_{f(i)-y \equiv_n 0} \cdot g(y).$$

Since  $(\mathbb{Z}/n\mathbb{Z}, +)$  is a group, the equation  $f(i) = y$  has a unique solution modulo  $n$ , namely  $y = f(i) \pmod n$ , and since  $y$  ranges from 0 to  $n - 1$ , only a single term of our inner sum remains. We thus get

$$n \cdot \sum_{i \in I} \sum_{y=0}^{n-1} \mathbb{1}_{f(i)-y \equiv_n 0} \cdot g(y) = n \cdot \sum_{i \in I} g(f(i)). \quad \blacksquare$$

This lemma is highly technical. When we use it,  $\sum_{i \in I}$  will always be some nested summation over integers,  $f(i)$  will be some product of index variables and  $g$  will be a labelling of kets.

## A.2 Irrelevance of choice of primitive $\zeta$ for quantum graphical calculi

Generally, the  $d$ -dimensional quantum Fourier transform is defined with  $\zeta = e^{\frac{2\pi i}{d}}$  [45, p. 4][49, Def. 2.3]. However, for the ZX- and ZH-calculi, we only demand two properties of our yellow box with one input and output:

- i) The *color change rule*, meaning attaching a box to every leg of a green spider turns it red, and vice versa, and
- ii) The fact that repeating the yellow box three times yields its adjoint.

It turns out that, as long as we replace  $\zeta$  with another *primitive* root of unity, these properties still hold.

The second point arises from the fact that, independently of the choice of primitive  $\zeta$ , the quantum Fourier transform applied twice in succession is negation modulo  $d$  on the computational basis. Since negation is involutive (and clearly independent of which root of unity we chose), we get that four times the quantum Fourier transform is the identity, from which our claim follows.

For the first point, observe:

**Lemma A.2.1:** *The color change rule holds independent of the choice of  $\zeta \neq 1$ .*

PROOF:

$$\begin{aligned}
& \begin{array}{c} \overbrace{\phantom{\dots}}^m \\ \vdots \\ \text{[Diagram: A green spider with } m \text{ legs, each ending in a yellow box. The legs are grouped into } n \text{ pairs. The bottom } n \text{ legs are labeled } |x_1\rangle, \dots, |x_n\rangle. \text{]} \\ \vdots \\ |x_1\rangle \quad |x_n\rangle \end{array} = \frac{1}{\sqrt{d}^n} \sum_{i_1=0}^{d-1} \dots \sum_{i_n=0}^{d-1} \left( \prod_{k=1}^n \zeta^{i_k x_k} \right) \begin{array}{c} \overbrace{\phantom{\dots}}^m \\ \vdots \\ \text{[Diagram: A green spider with } m \text{ legs, each ending in a yellow box. The legs are grouped into } n \text{ pairs. The bottom } n \text{ legs are labeled } |i_1\rangle, \dots, |i_n\rangle. \text{]} \\ \vdots \\ |i_1\rangle \quad |i_n\rangle \end{array} = \frac{1}{\sqrt{d}^n} \sum_{j=0}^{d-1} \left( \prod_{k=1}^n \zeta^{j x_k} \right) \overbrace{\phantom{\dots}}^m \begin{array}{c} \text{[Diagram: A yellow box.]} \\ \vdots \\ \text{[Diagram: A yellow box.]} \\ \vdots \\ \text{[Diagram: A yellow box.]} \end{array} \begin{array}{c} |j\rangle \\ \vdots \\ |j\rangle \end{array} \\
& = \frac{1}{\sqrt{d}^{n+m}} \sum_{j=0}^{d-1} \left( \prod_{k=1}^n \zeta^{j x_k} \right) \sum_{y_1=0}^{d-1} \dots \sum_{y_m=0}^{d-1} \left( \prod_{\nu=1}^m \zeta^{y_\nu j} \right) |y_1 \dots y_m\rangle \\
& = \frac{1}{\sqrt{d}^{n+m}} \sum_{j=0}^{d-1} \sum_{y_1=0}^{d-1} \dots \sum_{y_m=0}^{d-1} \zeta^{j(\sum_{k=1}^n x_k + \sum_{\nu=1}^m y_\nu)} |y_1 \dots y_m\rangle \\
& = \frac{1}{\sqrt{d}^{n+m}} \sum_{y_1=0}^{d-1} \dots \sum_{y_m=0}^{d-1} \sum_{j=0}^{d-1} \left( \zeta^{\sum_{k=1}^n x_k + \sum_{\nu=1}^m y_\nu} \right)^j |y_1 \dots y_m\rangle \\
& \stackrel{\text{A.1.4}}{=} \frac{1}{\sqrt{d}^{n+m-2}} \cdot \sum_{\substack{y_1, \dots, y_m \in \mathbb{Z}/d\mathbb{Z} \\ x_1 + \dots + x_n + y_1 + \dots + y_m = 0}} |y_1 \dots y_m\rangle \quad \blacksquare
\end{aligned}$$

# Appendix B

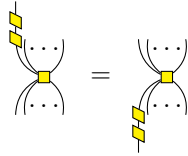
## Soundness of qudit ZH rewrite rules

In this appendix, we argue for the soundness of the rewrite rules introduced in Section 4.2.

The soundness of the spider fusion rule zs follows from the construction of the Z-spider as a commutative Frobenius-algebra.

For the  $H$ -box contraction rule hs, we first introduce the following lemma:

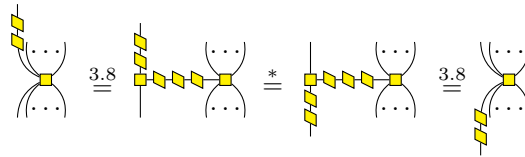
**Lemma B.0.1:** *We can freely transfer a double quantum Fourier transform between the legs of a  $H$ -box, e.g.*



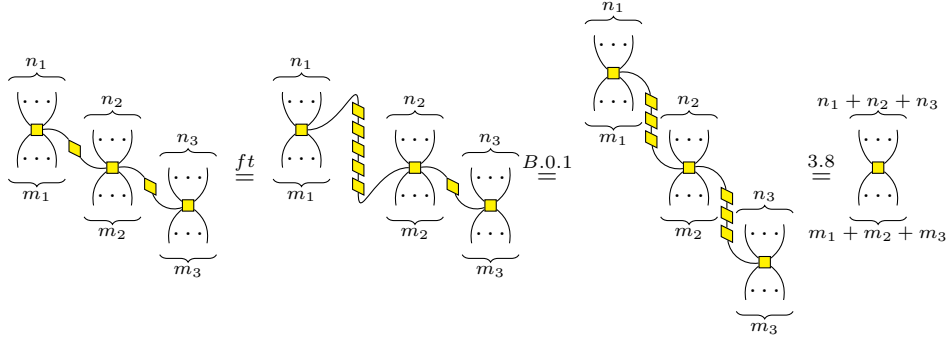
PROOF: We have

$$\begin{aligned}
 \text{Diagram} &= \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \zeta^{ijk} |-k\rangle \langle i| \langle j| = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \bar{\zeta}^{ijk} |k\rangle \langle i| \langle j| = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \zeta^{ijk} |k\rangle \langle i| \langle -j| = \text{Diagram} \quad (*)
 \end{aligned}$$

Using this we then have



Then we get:



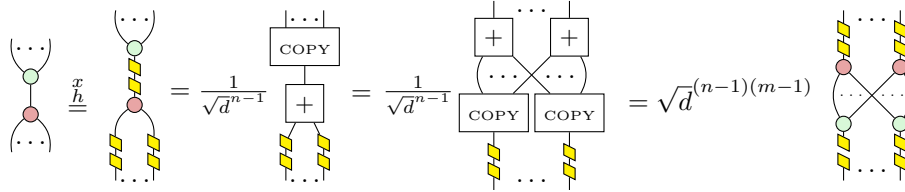
For the identity rule, simply observe that

$$\text{---} \circlearrowleft \text{---} = \sum_{i=0}^{d-1} |i\rangle\langle i| = \text{id}_{\mathbb{C}^d}.$$

To argue soundness of the two bialgebra rules, we adopt the proof strategy of Backens et al [7, Eq. 5], where they reinterpret the qubit ZH-generators as the boolean operators conjunction (and), negation and xor. While our generators no longer correspond to boolean operations, recall from 4.4 that we can interpret them as arithmetic operations in  $\mathbb{Z}/d\mathbb{Z}$ .

Note also that we use *ft* and *h* in these proofs. This might seem like we are going in circles, as we proved *ft* and *h* using the bialgebra rule, but note how in this section we are not conducting proofs *within* ZH-calculus, but rather proofs *about* it. This means that any fact that is provable using only linear algebra and expressible as a diagram is available to use. Thus, without further ado, we present the remaining soundness proofs:

PROOF (BIALGEBRA FOR Z- AND X-SPIDER):



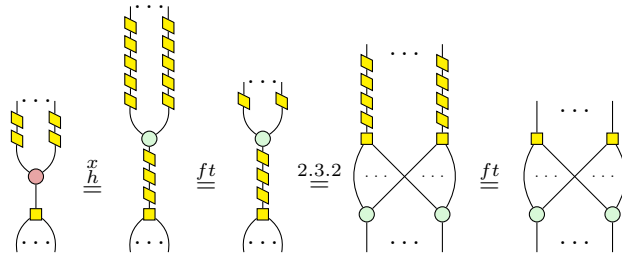
From here on, use (*ft*) on each wire of the bipartite graph in the middle to introduce four quantum Fourier transforms. Then use (*x*) and (*h*) once each on each spider to eliminate all Fourier transforms, yielding the RHS of our rule. Lastly, observe that

$$\blacktriangle \stackrel{4.3}{=} \text{---} \circlearrowleft \text{---} \stackrel{x}{=} \text{---} \circlearrowright \text{---} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \zeta^{ij} \stackrel{A.1.5}{=} \sqrt{d}.$$

■

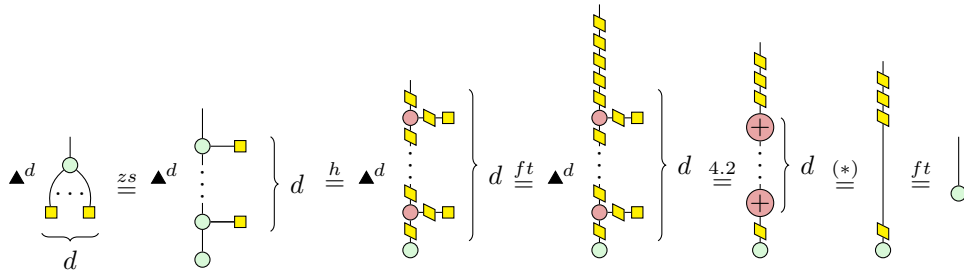
The remaining proofs are straightforward.

PROOF (BIALGEBRA FOR Z-SPIDER AND H-BOX):



■

PROOF (CYCLIC):



Here the step marked (\*) uses the fact that repeating Pauli-X a total of  $d$  times in succession is just the identity. ■

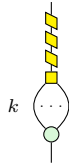


# Appendix C

## Gadgets for ZH-diagrams

In this appendix we present common gadgets useful in the construction of complex ZH-diagrams. We use these ideas mainly for our universality results, but they often prove helpful in constructing ZH-diagrams on-the-fly.

**Theorem C.0.1 (Exponentiator):** *The exponentiator gadget*



realizes the map

$$|i\rangle \mapsto |i^k\rangle.$$

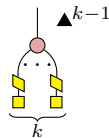
For prime  $d$ , this has the following special cases:

- i) For  $k = d - 2$  we have  $i^k = i^{-1}$  for  $i \neq 0$
- ii) For  $k = d - 1$  we have  $i^k = 1$  for  $i \neq 0$
- iii) For  $k = d$  we have  $i^k = i$

PROOF: Since the multiplicative group of a finite field is cyclic of order  $d-1$  we have  $i^{d-1} = 1$  for all  $i \neq 0$ . The other two special cases then follow from multiplication with  $i^{-1}$  and  $i$  respectively. ■

The next gadget allows us to express arbitrary computational basis states:

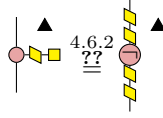
**Theorem C.0.2:** *The gadget*



realizes the state  $|i\rangle$ .



PROOF: We have



which realizes the map

$$|i\rangle \mapsto | - ( - (-i) - 1 ) \rangle = |1 - i\rangle. \quad \blacksquare$$

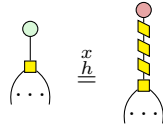
Finally, we introduce a gadget that implements an “is zero” check:

**Theorem C.0.5:** *For  $d$  prime, the effect-gadget*



*evaluates to 1 if at least one of the inputs is  $|0\rangle$  and 0 otherwise.*

PROOF: We have



meaning we post-select the result of a multiplication to be  $\sqrt{d}|0\rangle$ . Since  $d$  is prime,  $\mathbb{Z}/d\mathbb{Z}$  is a field and thus zero-divisor free, the result of a multiplication being zero means at least one of our inputs was  $|0\rangle$ . The scalar factor  $\sqrt{d}$  originates in A.2.1, since we have an  $X$ -spider with one input and no outputs. The proof of the bialgebra rule in the previous chapter shows that  $\blacktriangle = \sqrt{d}$ , meaning by  $d$  it cancels with  $\blackstar$ . ■

# Appendix D

## Further rewrite rules and completeness considerations

In this appendix, we derive further rewrite rules that have no direct application in the main parts of this dissertation, yet might be useful for someone trying to prove completeness. We also briefly discuss some results that might make translating over Backens et al's completeness results [7, Sec. 4.2] very challenging. Many sections of this appendix are open-ended rather than giving positive or negative results. We further ignore scalars.

### D.1 The indexing gadget copies through $Z$ -spiders

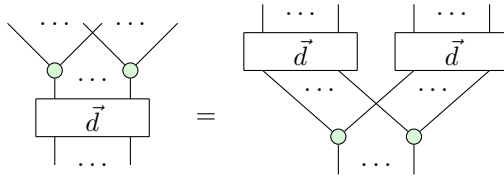
Goal of this section is to provide a qudit proof of Lemma 2.39 in Backens et al's paper [7], which states that for qubits, the indexing map copies through  $Z$ -spiders. For a dit-string  $\vec{d} = d_1 \dots d_m$  define the following qudit generalization of Backens et al's indexing map [7, Def. 2.38] based on 5.5:

$$\text{[Diagram]} \quad (D.1)$$

Figure D.1: Dit indexing map

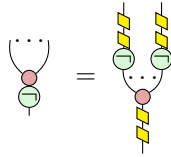
Note that Backens et al's indexing map is invariant under horizontal mirroring, whereas ours is not. Since we chose to define our normalform for effects as opposed to states, our version of Backens et al's Lemma 2.39 is thus upside down:

**Lemma D.1.1 ([7, Lem. 2.39]):**

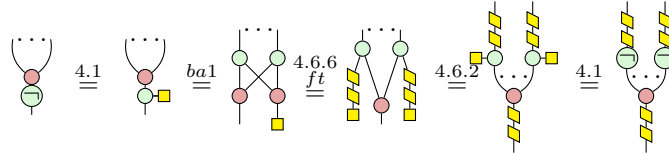


Before we can prove this, we need to generalize a copy-through lemma for green  $\neg$ -spiders

**Lemma D.1.2 ([7, Lem. 2.20]):**

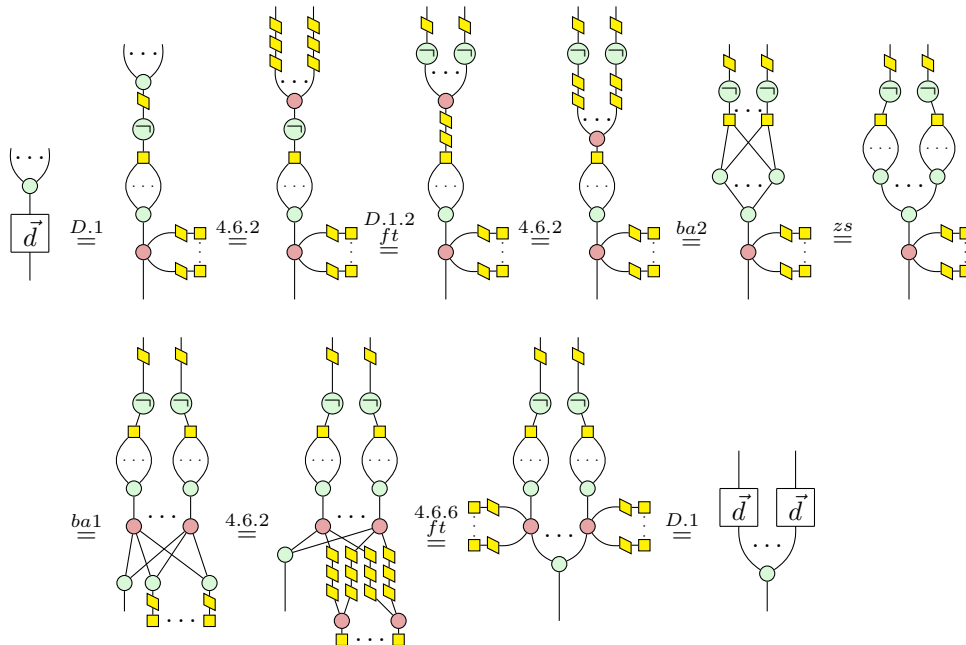


PROOF:



With this we then get

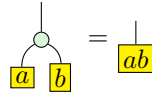
PROOF (OF D.1.1): Clearly it suffices to prove the lemma for  $m = 1$ , e.g.



## D.2 Multiply, Intro and Average

In this section, we briefly discuss the three rules Backens et al [7, Sec. 3] introduce to prove completeness of the qubit ZH-calculus.

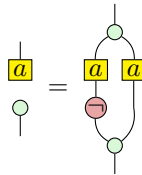
The multiply rule [7, Sec. 3.1]



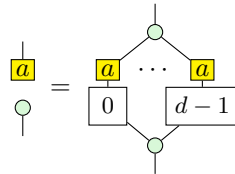
holds unchanged. To see this, recall from Chapter 5 that a  $Z$ -spider with one output realizes the Schur-product of its inputs. Clearly we have

$$(1, a, a^2, \dots, a^{-1}) \star (1, b, b^2, \dots, b^{d-1}) = (1, ab, (ab)^2, \dots, (ab)^{d-1}).$$

The latter two, intro [7, Sec. 3.2] and average [7, Sec. 3.3] are more complicated. Their qubit equations are invalid for the qudit case if  $d > 2$ . For intro, observe that evaluating the LHS of

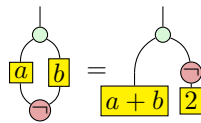


on  $|0\rangle$  yields the  $a$ -labelled  $H$ -box, whereas the RHS yields the  $a^{d-1}$ -labelled  $H$ -box. However, we can generalize the rule as follows using the indexing gadget of Equation D.1:



However, clearly this rule involved a significant number of generators.

Lastly, for the average rule



the LHS becomes

$$\sum_{i=0}^{d-1} a^{d-1-i} b^i,$$

while the RHS becomes

$$\sum_{i=0}^{d-1} (a+b)^i 2^{d-1-i}.$$

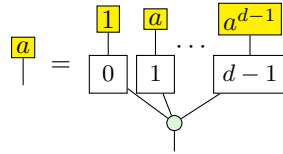
We were not able to find a generalization of the average rule to the qudit setting. Note, however, that Backens et al's Lemma 8.7 holds in a generalized form if we replace  $\frac{1}{2}$  with  $\frac{1}{2^{d-1}}$ .

### D.3 A review of normalform rewriting

In this section, we discuss which parts of Backens et al’s algorithm [7, Sec. 4] for rewriting ZH-diagrams into normalform translates into the qudit setting, and for which steps we have found no generalization.

The most significant obstacle is that we were unable to find a sequence of rewrite steps that transforms the ZH-generators into normalform. These issues mostly arise from the indexing gadget being significantly more complicated than in the qubit case, meaning it is a lot more difficult to create “out of thin air”. This is indeed the main obstacle we faced when trying to bring the identity wire into normal form.

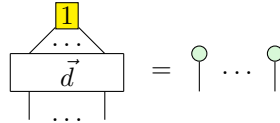
The normalform diagram for an  $a$ -labelled  $H$ -box is



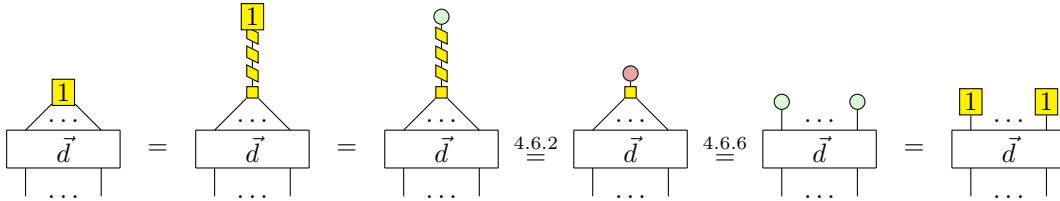
Particularly, this normalform requires  $a^i$ -labelled  $H$ -boxes. Our only idea for introducing these would be a generalization of Lemma 4.6.5 to arbitrary labelled  $H$ -boxes [7, Lem. 2.31].

One observation we can make is that 1-labelled  $H$ -boxes can be omitted from normal-form diagrams, as

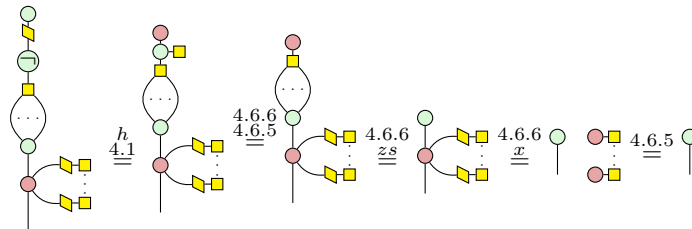
**Proposition D.3.1:**



PROOF: As



it suffices to show the proposition for 1-ary indexing boxes. For these, we get



If one thinks back to how our normal form leverages Schur-products, this makes sense as a matrix filled with 1-entries is the identity.

The next few steps of Backens et al's completeness proof involve either combining normalform diagrams, or absorbing generators into normalform diagrams. Here, we can give some positive results: The proof that a normalform diagram juxtaposed with a  $Z$ -counit can be brought into normalform [7, Prop 4.10] translates over immediately, if we use our generalized intro-rule. The proof that the Schur-product of two normalform diagrams can be brought into normalform [7, Prop 4.11] also works in the qudit case via D.1.1. From this it then also follows that the tensor product of normalform diagrams can be brought into normalform [7, Cor. 4.12].

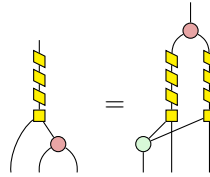
The proof that composing a normalform diagram with a  $Z$ -unit yields a normalform diagram [7, Lem. 4.17] relies on the average rule, and thus has no immediately obvious generalization. Further investigations here could potentially result in a qudit average rule.

We assume that proving the arithmetic rules for the case of integer labelled  $H$ -boxes [7, Sec. 5] is incredibly involved due to the complicated nature of the qudit successor gadget (see Chapter 5).

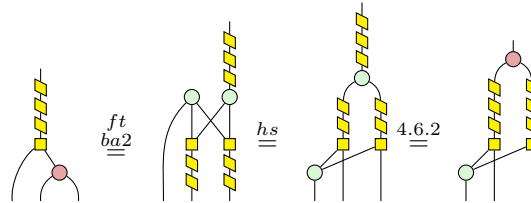
## D.4 Ring identities

In this section, we derive some equations relating to the structure of the unital commutative ring  $(\mathbb{Z}/d\mathbb{Z}, +, \cdot)$  diagrammatically.

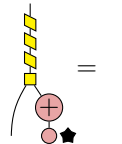
**Lemma D.4.1 (Distributivity [7, Lem. 8.1][44, Lem. 5.34]):** *This lemma expresses the identity  $x \cdot (y + z) = x \cdot y + x \cdot z$ .*



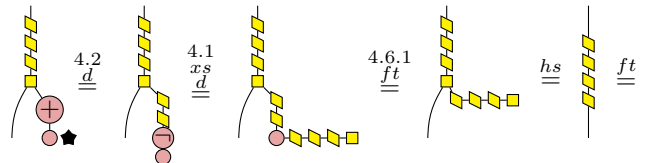
PROOF:



**Lemma D.4.2 (Multiplicative Unit):** *This lemma expresses the identity  $x \cdot 1 = x$ :*



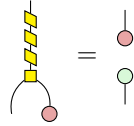
PROOF:





---

**Lemma D.4.3 (Multiplicative annihilation):** *This lemma expresses the identity  $x \cdot 0 = 0$*



PROOF:

