

EPTCS 384

Proceedings of the
**Twentieth International Conference on
Quantum Physics and Logic**

Paris, France, 17-21st July 2023

Edited by: Shane Mansfield, Benoît Valiron and Vladimir Zamdzhiev

Published: 30th August 2023
DOI: 10.4204/EPTCS.384
ISSN: 2075-2180
Open Publishing Association

Preface

This volume contains the proceedings of the 20th International Conference on Quantum Physics and Logic (QPL 2023). The conference was held from 17 to 21 July 2023 at Institut Henri Poincaré in Paris, France.

Quantum Physics and Logic is a conference series that brings together academic and industry researchers working on mathematical foundations of quantum computation, quantum physics, and related areas. The main focus is on the use of algebraic and categorical structures, formal languages, type systems, semantic methods, as well as other mathematical and computer scientific techniques applicable to the study of physical systems, physical processes, and their composition. Work applying quantum-inspired techniques and structures to other fields (such as linguistics, artificial intelligence, and causality) is also welcome.

The QPL 2023 conference solicited four different kinds of submissions: proceedings submissions, non-proceedings submissions, poster submissions, and programming tool submissions.

Proceedings submissions were papers that were required to provide sufficient evidence of results of genuine interest. Authors of accepted proceedings submissions were given the opportunity to present their work during a talk at the conference and these papers were included in the proceedings of QPL 2023. No other kinds of submissions were considered for inclusion in the proceedings. Non-proceedings submissions consisted of a three page summary, together with a (link to a) separate published paper or preprint. Authors of accepted non-proceeding submissions were allowed to present their work in the form of a talk during the conference. Poster submissions consisted of a three page abstract of (partial) results or work in progress and authors of accepted poster submissions were invited to present their work during one of the poster sessions of the conference. Programming tool submissions consisted of three page descriptions of programming tools or frameworks. Authors of accepted programming tool submissions were given an opportunity to present their software during a dedicated “Software Session”.

These proceedings contain 14 contributed papers that were selected for publication by the Program Committee. Papers submitted to QPL undergo a review process that is managed by members of the PC. The vast majority of submissions received at least three reviews. The selection of accepted papers was done through the use of the EasyChair conference management system following consideration of the submitted reviews and following (where necessary) discussion among the PC. The review process was single-blind: the identity of the authors is revealed to the reviewers, but not vice-versa. PC members were allowed to invite external experts to serve as subreviewers and to participate in the discussion of submissions which they were invited to review.

A record 152 submissions (excluding withdrawals and retractions) were considered for review by the PC. QPL 2023 had 54 accepted submissions in the non-proceedings track and 14 accepted submissions in the proceedings track. Most of the talks were presented during parallel sessions, but a selection of talks were presented during plenary sessions in the mornings. The program also had several poster sessions and one session dedicated to showcasing accepted programming tool submissions. There was also an industry session where industrial sponsors of QPL 2023 were given an opportunity to present their companies. The industry session consisted of two talks – one by Quandela (Diamond Sponsor) and one by Quantinuum (Gold Sponsor).

The QPL 2023 conference featured an award for Best Student Paper. Papers eligible for the award

were those where all the authors are students at the time of submission. The PC decided to award the Best Student Paper award for QPL 2023 to Cole Comfort (Department of Computer Science, University of Oxford) for his paper “The Algebra for Stabilizer Codes”.

The official website of the conference is <https://qpl2023.github.io/> and it contains a lot of relevant information about QPL 2023.

The Program Committee consisted of 42 members who were: Pablo Arrighi, Miriam Backens, Jonathan Barrett, Alessandro Bisio, Titouan Carette, Ulysse Chabaud, Giulio Chiribella, Bob Coecke, Alejandro Díaz-Caro, Ross Duncan, Pierre-Emmanuel Emeriau, Yuan Feng, Stefano Gogioso, Amar Hadzahasanovic, Emmanuel Jeandel, Martti Karvonen, Kohei Kishida, Aleks Kissinger, Ravi Kunjwal, Shane Mansfield (co-chair), Simon Martiel, Konstantinos Meichanetzidis, Mio Murao, Ognyan Oreshkov, Prakash Panangaden, Simon Perdrix, Neil Ross, Ana Belén Sainz, Carlo Maria Scandolo, John Selby, Peter Selinger, Rui Soares Barbosa, Pawel Sobocinski, Isar Stubbe, Benoît Valiron (co-chair), Augustin Vanrietvelde, Renaud Vilmart, Quanlong Wang, John van de Wetering, Alexander Wilce, Mingsheng Ying, and Vladimir Zamdzhiev (co-chair).

The Organising Committee consisted of eight members who were: Ulysse Chabaud, Pierre-Emmanuel Emeriau, Shane Mansfield, Simon Perdrix, Benoît Valiron, Augustin Vanrietvelde, Renaud Vilmart, and Vladimir Zamdzhiev.

The QPL Steering Committee consisted of Bob Coecke, Prakash Panangaden, and Peter Selinger.

We wish to thank all the members of the PC for their work in selecting the program of QPL 2023. We also thank all external subreviewers for their help and also the authors for their submissions to QPL 2023. Thanks also go to Ulysse Chabaud for making a few decisions on papers for which the PC co-chairs were not available. We are grateful to the EPTCS team for their help in preparing the proceedings of the conference. We also thank the members of the Organising Committee for their help in setting up the conference. We also thank the student helpers who volunteered to assist us: Dogukan Bakircioglu, Marin Costes, Kinnari Dave, Nicolas Heurtel, Paraskevi Kasnetsi, Charlène Laffond, Julien Lamiroy, Louis Lemonnier, Tianyi Li, Quan Long, Fatima-Zahra Merimi, Octave Mestoudjian, Ramdane Mouloua, and Sunheang Ty. Thanks also go to the staff of Institut Henri Poincaré who helped us with the organisation of the conference. Finally, we thank the QPL steering committee for their support and we thank all people who have contributed to the success of QPL 2023.

QPL 2023 received (financial) support from Quandela (Diamond Sponsor), Quantinuum (Gold Sponsor), The University of Chicago (Silver Sponsor), and Inria (Organisational Support and Funding).

August 2023,

Shane Mansfield, Benoît Valiron, and Vladimir Zamdzhiev

Table of Contents

Preface	i
<i>Shane Mansfield, Benoît Valiron and Vladimir Zamdzhiev</i>	
Table of Contents	iii
Graphical CSS Code Transformation Using ZX Calculus	1
<i>Jiaxin Huang, Sarah Meng Li, Lia Yeh, Aleks Kissinger, Michele Mosca and Michael Vasmer</i>	
Light-Matter Interaction in the ZXW Calculus	20
<i>Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang and Bob Coecke</i>	
Locally Tomographic Shadows (Extended Abstract)	47
<i>Howard Barnum, Matthew A. Graydon and Alex Wilce</i>	
Quantum Suplattices	58
<i>Gejza Jenča and Bert Lindenhovius</i>	
Global Synthesis of CNOT Circuits with Holes	75
<i>Ewan Murphy and Aleks Kissinger</i>	
Picturing Counting Reductions with the ZH-Calculus	89
<i>Tuomas Laakkonen, Konstantinos Meichanetzidis and John van de Wetering</i>	
Generators and Relations for 3-Qubit Clifford+CS Operators	114
<i>Xiaoning Bian and Peter Selinger</i>	
Complete Equational Theories for the Sum-Over-Paths with Unbalanced Amplitudes	127
<i>Matthew Amy</i>	
The Qudit ZH-Calculus: Generalised Toffoli+Hadamard and Universality	142
<i>Patrick Roy, John van de Wetering and Lia Yeh</i>	
Moore-Penrose Dagger Categories	171
<i>Robin Cockett and Jean-Simon Pacaud Lemay</i>	
Generalised Winograd Schema and its Contextuality	187
<i>Kin Ian Lo, Mehrnoosh Sadrzadeh and Shane Mansfield</i>	
Flow-preserving ZX-calculus Rewrite Rules for Optimisation and Obfuscation	203
<i>Tommy McElvanney and Miriam Backens</i>	
The Qupit Stabiliser ZX-travaganza: Simplified Axioms, Normal Forms and Graph-Theoretic Simplification	220

Boldizsár Poór, Robert I. Booth, Titouan Carette, John van de Wetering and Lia Yeh

Floquetifying the Colour Code 265
Alex Townsend-Teague, Julio Magdalena de la Fuente and Markus Kesselring

Graphical CSS Code Transformation Using ZX Calculus

Jiixin Huang*

Dept. of Computer Science
Hong Kong University of
Science and Technology
jhuangbo@connect.ust.hk

Sarah Meng Li*

Institute for Quantum Computing (IQC)
Dept. of Combinatorics & Optimization (C&O)
University of Waterloo
sarah.li@uwaterloo.ca

Lia Yeh*

University of Oxford
Quantinuum
17 Beaumont Street
Oxford OX1 2NA, UK
lia.yeh@cs.ox.ac.uk

Aleks Kissinger

University of Oxford
aleks.kissinger@cs.ox.ac.uk

Michele Mosca

Perimeter Institute (PI)
IQC, C&O, University of Waterloo
michele.mosca@uwaterloo.ca

Michael Vasmer

PI, IQC, University of Waterloo
mvasmer@perimeterinstitute.ca

In this work, we present a generic approach to transform CSS codes by building upon their equivalence to phase-free ZX diagrams. Using the ZX calculus, we demonstrate diagrammatic transformations between encoding maps associated with different codes. As a motivating example, we give explicit transformations between the Steane code and the quantum Reed-Muller code, since by switching between these two codes, one can obtain a fault-tolerant universal gate set. To this end, we propose a bidirectional rewrite rule to find a (not necessarily transversal) physical implementation for any logical ZX diagram in any CSS code.

We then focus on two code transformation techniques: *code morphing*, a procedure that transforms a code while retaining its fault-tolerant gates, and *gauge fixing*, where complimentary codes can be obtained from a common subsystem code (e.g., the Steane and the quantum Reed-Muller codes from the $[[15, 1, 3, 3]]$ code). We provide explicit graphical derivations for these techniques and show how ZX and graphical encoder maps relate several equivalent perspectives on these code transforming operations.

1 Introduction

Quantum computation has demonstrated its potential in speeding up large-scale computational tasks [3, 68] and revolutionizing multidisciplinary fields such as drug discovery [11], climate prediction [60], chemistry simulation [47], and the quantum internet [25]. However, in a quantum system, qubits are sensitive to interference and information becomes degraded [50]. To this end, quantum error correction [55, 57] and fault tolerance [33, 41] have been developed to achieve large-scale universal quantum computation [34].

Stabilizer theory [32] is a mathematical framework to describe and analyze properties of quantum error-correcting codes (QECC). It is based on the concept of stabilizer groups, which are groups of Pauli operators whose joint $+1$ eigenspace corresponds to the code space. Stabilizer codes are a specific type of QECC whose encoder can be efficiently simulated [1, 31]. As a family of stabilizer codes, Calderbank-Shor-Steane (CSS) codes permit simple code constructions from classical codes [9, 10, 57, 58].

As a language for rigorous diagrammatic reasoning of quantum computation, the ZX calculus consists of ZX diagrams and a set of rewrite rules [16, 66]. It has been used to relate stabilizer theory to graphical normal forms: notably, efficient axiomatization of the stabilizer fragments for qubits [4, 36, 49],

*equal authorship

qutrits [61, 65], and prime-dimensional qudits [8]. This has enabled various applications, such as measurement-based quantum computation [49, 56], quantum circuit optimization [19, 30] and verification [46], as well as classical simulation [14, 40]. Beyond these, ZX-calculus has been applied to verify QECC [23, 26], represent Clifford encoders [38], as well as study various QECC such as tripartite coherent parity check codes [12, 13] and surface codes [27, 28, 29, 54]. Specific to CSS codes, ZX-calculus has been used to visualize their encoders [39], code maps and code surgeries [22], their correspondence to affine Lagrangian relations [20], and their constructions in high-dimensional quantum systems [21].

In this paper, we seek to answer some overarching questions about QECC constructions and fault-tolerant implementations. We focus on CSS codes and leverage the direct correspondence between phase-free ZX diagrams and CSS code encoders [39]. Given an arbitrary CSS code, based on its normal form, we propose a bidirectional rewrite rule to find a (not necessarily transversal) physical implementation for any logical ZX diagram. Furthermore, we demonstrate diagrammatic transformations between encoding maps associated with different codes. Here, we focus on two code transformation techniques: *code morphing*, a procedure that transforms a code while retaining its fault-tolerant gates [62], and *gauge fixing*, where complimentary codes (such as the Steane and the quantum Reed-Muller codes) can be obtained from a common subsystem code [2, 51, 53, 64]. We provide explicit graphical derivations for these techniques and show how ZX and graphical encoder maps relate several equivalent perspectives on these code transforming operations.

The rest of this paper is organized as follows. In Sec. 2, we introduce notions and techniques used to graphically transform different CSS codes using the ZX calculus. In Sec. 3, we generalize the ZX normal form for CSS stabilizer codes to CSS subsystem codes, and provide generic bidirectional rewrite rules for any CSS encoder. In Sec. 4, we provide explicit graphical derivations for morphing the Steane and the quantum Reed-Muller codes. In Sec. 5, we focus on the switching protocol between these two codes. Through ZX calculus, we provide a graphical interpretation of this protocol as gauge-fixing the $[[15, 1, 3, 3]]$ subsystem code, followed by syndrome-determined recovery operations. We conclude with Sec. 6.

2 Preliminaries

We start with some definitions. The Pauli matrices are 2×2 unitary operators acting on a single qubit. Let i be the imaginary unit.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Y = iXZ = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}.$$

Let \mathcal{P}_1 be the single-qubit Pauli group, $\mathcal{P}_1 = \langle i, X, Z \rangle$, $I, Y \in \mathcal{P}_1$.

Definition 2.1. Let $U \in \mathcal{U}(2)$. In a system over n qubits, $1 \leq i \leq n$,

$$U_i = I \otimes \dots \otimes I \otimes U \otimes I \otimes \dots \otimes I$$

denotes U acting on the i -th qubit, and identity on all other qubits.

Let \mathcal{P}_n be the n -qubit Pauli group. It consists of all tensor products of single-qubit Pauli operators.

$$\mathcal{P}_n = \langle i, X_1, Z_1, \dots, X_n, Z_n \rangle.$$

The stabilizer formalism is a mathematical framework to describe and analyze the properties of certain QECC, called stabilizer codes [32, 33]. Consider n qubits and let $m \leq n$. A stabilizer group $\mathcal{S} = \langle S_1, \dots, S_m \rangle$ is an Abelian subgroup of \mathcal{P}_n that does not contain $-I$. The codespace of the corresponding stabilizer code, \mathcal{C} , is the joint $+1$ eigenspace of \mathcal{S} , i.e.,

$$\mathcal{C} = \{|\psi\rangle \in \mathbb{C}^{2^n}; \mathcal{S}|\psi\rangle = |\psi\rangle, \forall S \in \mathcal{S}\}.$$

The number of encoded qubits in a stabilizer code is $k = n - m$, where m is the number of independent stabilizer generators [32]. Moreover, we can define the *centralizer* of \mathcal{S} as

$$\mathcal{N}(\mathcal{S}) = \{U \in \mathcal{P}_n; [U, S] = 0, \forall S \in \mathcal{S}\}.$$

One can check that $\mathcal{N}(\mathcal{S})$ is a subgroup of \mathcal{P}_n and $\mathcal{S} \subset \mathcal{N}(\mathcal{S})$. We remark that the notions of normalizer and centralizer coincide for any stabilizer group. In what follows, we will use them interchangeably. As we will see later, $\mathcal{N}(\mathcal{S})$ provides an algebraic structure for the subsystem codes. The code distance, d , of a stabilizer code is the minimal weight of operators in $\mathcal{N}(\mathcal{S})/\langle iI \rangle$ that is not in \mathcal{S} . We summarize the properties of a stabilizer code with the shorthand $\llbracket n, k, d \rrbracket$.

Finally, we introduce some notation for subsets of n -qubit Pauli operators, which will prove useful for defining CSS codes.

Definition 2.2. Let M be an $m \times n$ binary matrix and $P \in \mathcal{P}_1/\langle iI \rangle$. In the stabilizer formalism, M is called the stabilizer matrix, and M^P defines m P -type stabilizer generators.

$$M^P := \left\{ \bigotimes_{j=1}^n P^{[M]_{ij}}; 1 \leq i \leq m \right\}.$$

CSS codes are QECC whose stabilizers are defined by two orthogonal binary matrices G and H [9, 57]:

$$\mathcal{S} = \langle G^X, H^Z \rangle, \quad GH^T = \mathbf{0},$$

H^T is the transpose of H . This means that the stabilizer generators of a CSS code can be divided into two types: X-type and Z-type. For example, the $\llbracket 7, 1, 3 \rrbracket$ Steane code [57] in Fig. 1a is specified by

$$G = H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}_{3 \times 7}. \quad (1)$$

Accordingly, the X-type and Z-type stabilizers are defined as

$$S_1^X = X_1 X_3 X_5 X_7, \quad S_2^X = X_2 X_3 X_6 X_7, \quad S_3^X = X_4 X_5 X_6 X_7, \quad S_1^Z = Z_1 Z_3 Z_5 Z_7, \quad S_2^Z = Z_2 Z_3 Z_6 Z_7, \quad S_3^Z = Z_4 Z_5 Z_6 Z_7.$$

The logical operators \bar{X} and \bar{Z} are defined as

$$\bar{X} = X_1 X_4 X_5 \quad \text{and} \quad \bar{Z} = Z_1 Z_4 Z_5. \quad (2)$$

In Sec. 2.1, we define CSS subsystem codes. In Sec. 2.2, we define several CSS codes that will be used in subsequent sections. In Sec. 2.3, we introduce the basics of the ZX calculus and the phase-free ZX normal forms.

2.1 CSS Subsystem Codes

Subsystem codes [43, 52] are QECC where some of the logical qubits are not used for information storage and processing. These logical qubits are called gauge qubits. By fixing gauge qubits to some specific states, the same subsystem code may exhibit different properties, for instance, having different sets of transversal gates [7, 44, 45, 51, 67]. This provides a tool to circumvent restrictions on transversal gates such as the Eastin-Knill theorem [24].

Based on the construction proposed in [52], we describe a subsystem code using the stabilizer formalism.

Definition 2.3. *Given a stabilizer group \mathcal{S} , a gauge group \mathcal{G} is a normal subgroup of $\mathcal{N}(\mathcal{S})$, such that $\mathcal{S} \subset \mathcal{G}$ and that \mathcal{G}/\mathcal{S} contains anticommuting Pauli pairs. In other words, one can write*

$$\mathcal{S} = \langle S_1, \dots, S_m \rangle, \quad \mathcal{G} = \langle S_1, \dots, S_m, g_1^X, g_1^Z, \dots, g_r^X, g_r^Z \rangle, \quad 1 \leq m+r \leq n.$$

$(\mathcal{S}, \mathcal{G})$ defines an $[[n, k, r, d]]$ subsystem code where $n = m + k + r$. The logical operators are elements of the quotient group $\mathcal{L} = \mathcal{N}(\mathcal{S})/\mathcal{G}$.

Under this construction, n physical qubits are used to encode k logical qubits with r gauge qubits. Alternatively, we can think of the gauge group \mathcal{G} as partitioning the code space \mathcal{C} into two subsystems: $\mathcal{C} = \mathcal{A} \otimes \mathcal{B}$. Logical information is encoded in \mathcal{A} and \mathcal{L} serves as the group of logical operations. Gauge operators from \mathcal{G} act trivially on subsystem \mathcal{A} , while operators from \mathcal{L} act trivially on subsystem \mathcal{B} . Therefore, two states $\rho^{\mathcal{A}} \otimes \rho^{\mathcal{B}}$ and $\rho'^{\mathcal{A}} \otimes \rho'^{\mathcal{B}}$ are considered equivalent if $\rho^{\mathcal{A}} = \rho'^{\mathcal{A}}$, regardless of the states $\rho^{\mathcal{B}}$ and $\rho'^{\mathcal{B}}$. When $r = 0$, $\mathcal{G} = \mathcal{S}$. In that case, an $[[n, k, 0, d]]$ subsystem code is essentially an $[[n, k, d]]$ stabilizer code.

CSS subsystem codes are subsystem codes whose stabilizer generators can be divided into X-type and Z-type operators. In what follows, we provide an example to illustrate their construction.

2.2 Some Interesting CSS Codes

We start by defining the stabilizer groups for the $[[7, 1, 3]]$ Steane code, the $[[15, 1, 3]]$ extended Steane code [2], and the $[[15, 1, 3]]$ quantum Reed-Muller code [42]. They are derived from the family of $[[2^m - 1, 1, 3]]$ quantum Reed-Muller codes, with a recursive construction of stabilizer matrices [59]. The Steane code has transversal logical Clifford operators, and the quantum Reed-Muller code has a transversal logical T gate. Together these operators form a universal set of fault-tolerant gates. In Sec. 5, the relations between these codes are studied from a diagrammatic perspective.

For brevity, their corresponding stabilizer groups are denoted as \mathcal{S}_{steane} , \mathcal{S}_{ex} , and \mathcal{S}_{qrm} . As per Def. 2.2, consider three stabilizer matrices F , H , and J . Note that G is defined in Eq. (1). $\mathbf{0}$ and $\mathbf{1}$ denote blocks of 0s' and 1s' respectively. Their dimensions can be inferred from the context.

$$F = \begin{bmatrix} G & \mathbf{0} & G \\ \mathbf{0} & \mathbf{1} & \mathbf{1} \end{bmatrix}_{4 \times 15}, \quad H = [G \quad \mathbf{0}]_{3 \times 15},$$

$$J = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{3 \times 15}.$$

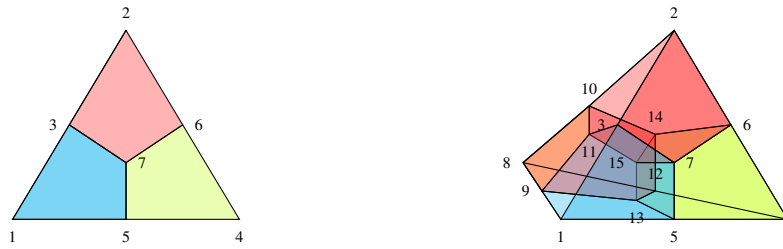
Then, the stabilizer groups are defined as

$$\mathcal{S}_{steane} = \langle G^X, G^Z \rangle, \quad \mathcal{S}_{ex} = \langle F^X, F^Z, H^X, H^Z \rangle, \quad \mathcal{S}_{qrm} = \langle F^X, F^Z, H^Z, J^Z \rangle. \quad (3)$$

Geometrically, one can define \mathcal{S}_{steane} and \mathcal{S}_{qrm} with the aid of Fig. 1. In Fig. 1a, the Steane code is visualized on a 2D lattice. Since the Steane code is self-dual, every coloured face corresponds to an X-type and Z-type stabilizer. In Fig. 1b, the quantum Reed-Muller code is visualized on a 3D lattice. Every coloured face corresponds to a weight-4 Z-type stabilizer. Every coloured cell corresponds to a weight-8 X-type and Z-type stabilizer respectively. For the Steane code, the logical operators defined in Eq. (2) correspond to an edge in the triangle. For the quantum Reed-Muller code, the logical X operator corresponds to a weight-7 triangular face, and the logical Z operator corresponds to a weight-3 edge of the entire tetrahedron. An example is shown below.

$$\bar{X} = X_1X_2X_3X_4X_5X_6X_7 \quad \text{and} \quad \bar{Z} = Z_1Z_4Z_5 \quad (4)$$

Given such representations, the Steane code and the quantum Reed-Muller code are also special cases of colour codes [5, 6, 44].



(a) The Steane code as a 2D colour code. (b) The quantum Reed-Muller code as a 3D colour code.

Figure 1: Each vertex represents a physical qubit. Each edge serves as an aid to the eye. They do not imply any physical interactions or inherent structures.

From Eq. (3), the extended Steane code is self-dual, and its encoded state is characterized by the lemma below. It shows that \mathcal{S}_{ex} and \mathcal{S}_{steane} are equivalent up to some auxiliary state.

Lemma 2.1 ([2]). *Any codeword $|\psi\rangle$ of the extended Steane code can be decomposed into a codeword $|\phi\rangle$ of the Steane code and a fixed state $|\eta\rangle$. That is,*

$$|\psi\rangle = |\phi\rangle \otimes |\eta\rangle,$$

where $|\eta\rangle = \frac{1}{\sqrt{2}}(|0\rangle|\bar{0}\rangle + |1\rangle|\bar{1}\rangle)$, $|\bar{0}\rangle$ and $|\bar{1}\rangle$ are the logical 0 and 1 encoded in the Steane code.

Since the logical information $|\phi\rangle$ encoded in the Steane code is not entangled with $|\eta\rangle$, to switch between the Steane code and the extended Steane code, one may simply add or discard the auxiliary state $|\eta\rangle$. This property will prove useful in Sec. 5.

Next, we define the $[[15, 1, 3, 3]]$ CSS subsystem code [64]. As per Def. 2.3, let \mathcal{S}_{sub} and \mathcal{G} be its stabilizer group and gauge group respectively.

$$\mathcal{S}_{sub} = \langle F^X, F^Z, H^Z \rangle, \quad \mathcal{G} = \langle F^X, F^Z, H^X, H^Z, J^Z \rangle. \quad (5)$$

Let $\mathcal{L}_g = \mathcal{G}/\mathcal{S}$ and $\mathcal{L} = \mathcal{N}(\mathcal{S})/\mathcal{G}$. One can verify that

$$\mathcal{L}_g = \langle H^X, J^Z \rangle, \quad \mathcal{L} = \langle \bar{X}, \bar{Z} \rangle. \quad (6)$$

Thus, the CSS subsystem code has one logical qubit and three gauge qubits, and they are acted on by \mathcal{L} and \mathcal{L}_g respectively. From Sec. 3 onwards, we call operators in \mathcal{L}_g as *gauge operators*.

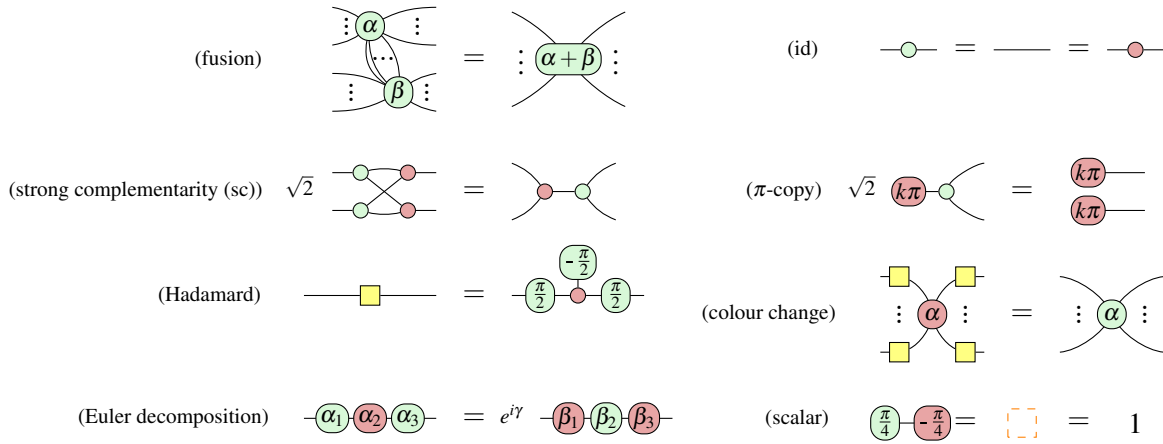


Figure 2: These eight equations suffice to derive all other equalities of linear maps on qubits [63]. $k \in \mathbb{Z}_2$. α_i , β_i and γ are real numbers satisfying the trigonometric relations derived in [18]. Each equation still holds when we replace all spiders with their corresponding spiders of the opposite colour. Whenever there are any two wires with ... between them, the rule holds when replacing this with any number of wires (i.e., 0 or greater).

Moreover, \mathcal{S}_{sub} can be viewed as the stabilizer group of a $[[15, 4, 3]]$ CSS code, with logical operators \mathcal{L}' . This code appears in an intermediary step of the gauge fixing process in Sec. 5.

$$\mathcal{L}' := \mathcal{L}_g \cup \mathcal{L} = \langle H^X, J^Z, \bar{X}, \bar{Z} \rangle. \quad (7)$$

2.3 ZX Calculus

The qubit ZX-calculus [15, 16, 17, 66] is a quantum graphical calculus for diagrammatic reasoning of any qubit quantum computation. Every diagram in the calculus is composed of two types of generators: Z spiders, which sum over the eigenbasis of the Pauli Z operator:

$${}_m \langle \text{Z spider with phase } \alpha \rangle_n := |0\rangle^{\otimes n} \langle 0|^{\otimes m} + e^{i\alpha} |1\rangle^{\otimes n} \langle 1|^{\otimes m}, \quad (8)$$

and X spiders, which sum over the eigenbasis of the Pauli X operator:

$${}_m \langle \text{X spider with phase } \alpha \rangle_n := |+\rangle^{\otimes n} \langle +|^{\otimes m} + e^{i\alpha} |-\rangle^{\otimes n} \langle -|^{\otimes m}. \quad (9)$$

The ZX-calculus is *universal* [16] in the sense that any linear map from m qubits to n qubits corresponds exactly to a ZX diagram, by the construction of Eqs. (8) and (9) and the composition of linear maps.

Furthermore, the ZX-calculus is *complete* [35, 37]: Any equality of linear maps on any number of qubits derivable in the Hilbert space formalism, is derivable using only a finite set of rules in the calculus. The smallest complete rule set to date [63] is shown in Fig. 2. Some additional rules, despite being derivable from this rule set, will be convenient to use in this paper. They are summarized in Fig. 3.

When a spider has phase zero, we omit its phase in the diagram, as shown below. A ZX diagram is *phase-free* if all of its spiders have zero phases. For more discussions on phase-free ZX diagrams, we refer readers to consult [39].

$$\begin{array}{c} \text{Z spider} \\ \text{X spider} \end{array} := \begin{array}{c} \text{Z spider with 0} \\ \text{X spider with 0} \end{array}$$

Due to the universality of the ZX calculus, quantum error-correcting code encoders, as linear isometries, can be drawn as ZX diagrams [38]. Moreover, the encoder for a CSS code corresponds exactly to the phase-free ZX (and XZ) normal form [39].

Definition 2.4. For a CSS stabilizer code defined by \mathcal{S} , let $\{S_i^x; 1 \leq i \leq m\} \subset \mathcal{S}$ be the X-type stabilizer generators and $\{\bar{X}_j; 1 \leq j \leq k\}$ be the logical X operators, $m+k < n$. Its ZX normal form can be found via the following steps:

- (a) For each physical qubit, introduce an X spider.
- (b) For each X-type stabilizer generator S_i^x and logical operator \bar{X}_j , introduce a Z spider and connect it to all X spiders where this operator has support.
- (c) Give each X spider an output wire.
- (d) For each Z spider representing \bar{X}_j , give it an input wire.

As an example, the ZX normal form for the Steane code is drawn in Fig. 4. The XZ normal form can be constructed based on Z-type stabilizer generators and logical Z operators by inverting the roles of X and Z spiders in the above procedure. In [39], Kissinger gave an algorithm to rewrite any phase-free ZX code diagram into both the ZX and XZ normal forms, and pointed out that it is sufficient to represent a CSS code encoder using either one of the forms.

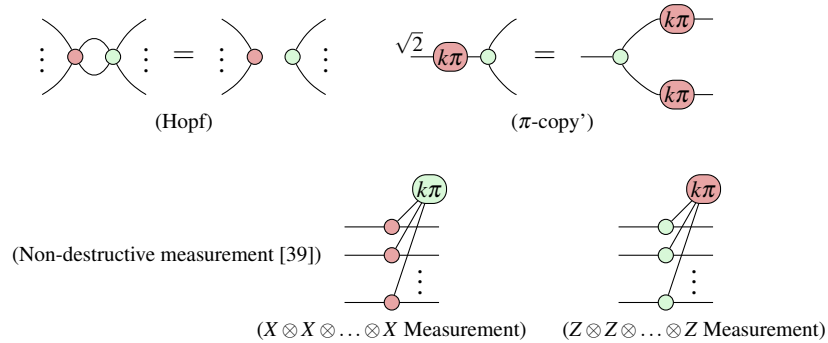


Figure 3: Some other useful rewrite rules, each derivable from the rules in Figure 2. $k \in \mathbb{Z}_2$. Each equation still holds when we interchange X and Z spiders.

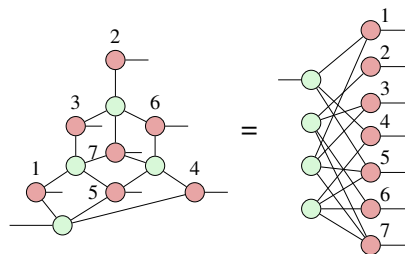


Figure 4: The Steane code encoder in the ZX normal form.

3 Graphical Construction of CSS Encoders

3.1 ZX Normal Forms for CSS Subsystem Codes

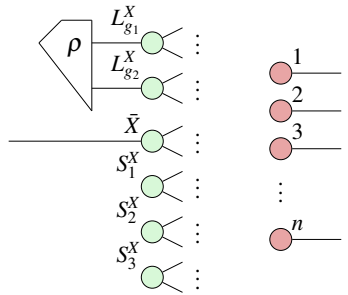
We generalize the ZX normal form for CSS stabilizer codes to CSS subsystem codes as follows.

Definition 3.1. For an $[[n, k, r, d]]$ CSS subsystem code defined by $(\mathcal{S}, \mathcal{G})$, let $\{S_i^x; 1 \leq i \leq m\}$ be the X-type stabilizer generators, $\{L_{g_t}^x; 1 \leq t \leq r\}$ be the X-type gauge operators, and $\{\bar{X}_j; 1 \leq j \leq k\}$ be the logical X operators, $m + k + r < n$. Its ZX normal form can be found via the following steps:

- For each physical qubit, introduce an X spider.
- For each stabilizer generator S_i^x , logical operator \bar{X}_j and gauge operator $L_{g_t}^x$, introduce a Z spider and connect it to all X spiders where this operator has support.
- Give each X spider an output wire.
- For each Z spider representing \bar{X}_j , give it an input wire.
- For all Z spiders representing $L_{g_t}^x$, attach to them a joint arbitrary input state (i.e., a density operator ρ).

Similar to CSS stabilizer codes, CSS subsystem codes also have an equivalent XZ normal form, which can be found by inverting the role of Z and X in the above procedure.

For $n > 3$, below we exemplify the ZX normal form for an $[[n, 1, 2, d]]$ CSS subsystem code with three X-type stabilizers generators $\{S_1^x, S_2^x, S_3^x\}$, two X-type gauge operators $\{L_{g_1}^x, L_{g_2}^x\}$, and one logical operator $\{\bar{X}\}$. For simplicity, we substitute wires connecting Z and X spiders by $\frown \therefore$. The detailed connectivities are omitted here, but they should be clear following step (b) in Def. 3.1. This notation will be used in the remainder of this paper.



3.2 Pushing through the Encoder

For any $[[n, k, d]]$ CSS code, its encoder map E is of the form:

$$k \left\{ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right\} \boxed{E} \left. \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right\} n.$$

Definition 3.2. Let \bar{X}_i and \bar{Z}_i be the X and Z operators acting on the i -th logical qubit. Let $\bar{\mathcal{X}}_i$ and $\bar{\mathcal{Z}}_i$ be the physical implementation of \bar{X}_i and \bar{Z}_i respectively. Diagrammatically, they can be represented as

$$\begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{\bar{X}_1} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{E} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} = \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{E} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{\bar{\mathcal{X}}_1} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \quad \text{and} \quad \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{\bar{Z}_1} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{E} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} = \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{E} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \boxed{\bar{\mathcal{Z}}_1} \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} .$$

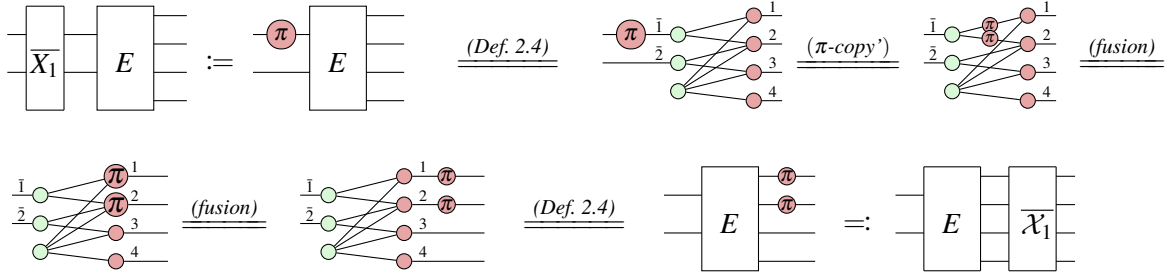
In other words, pushing \bar{X}_i (or \bar{Z}_i) through E yields $\bar{\mathcal{X}}_i$ (or $\bar{\mathcal{Z}}_i$). Using ZX rewrite rules along with the ZX (or XZ) normal form, we can prove the following lemma.

Lemma 3.1. For any CSS code, all \bar{X}_i and \bar{Z}_i are implementable by multiple single-qubit Pauli operators. In other words, all CSS codes have transversal \bar{X}_i and \bar{Z}_i .

Proof. Consider an arbitrary CSS code. Without loss of generality, represent its encoder E in the ZX normal form following Def. 2.4. Then proceed by applying the π -copy' rule on every \bar{X}_i (the X spider with a phase π on the left-hand side of the encoder E). \square

Below we illustrate the proof using the $[[4, 2, 2]]$ code as an example.

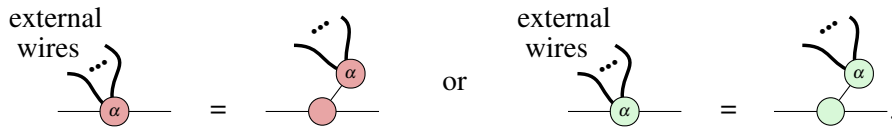
Example 3.1. For the $[[4, 2, 2]]$ code, $\bar{X}_1 = X_1X_2$.



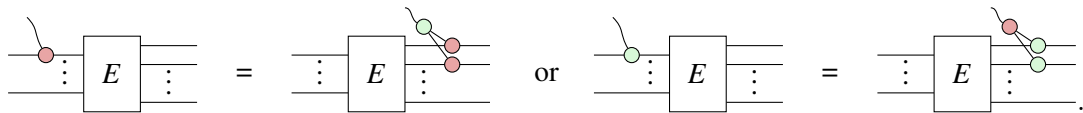
Beyond just X or Z spiders, one can push any ZX diagram acting on the logical qubits through the encoder. Such pushing is bidirectional, and the left-to-right direction is interpreted as finding a physical implementation for a given logical operator.

Proposition 3.1. Let E be the encoder of a CSS code. For any ZX diagram L on the left-hand side of E , one can write down a corresponding ZX diagram P on the right-hand side of E , such that $EL = PE$. In other words, P is a valid physical implementation of L for that CSS code.

Proof. We proceed as follows. First, unfuse all spiders on the logical qubit wires of L , whenever they are not phase-free or have more than one external wire:



For each X (or Z) spider on the logical qubit wire, rewriting E to be in ZX (or XZ) normal form and applying the strong complementarity (sc) rule yields:



On the left-hand side, a phase-free X (or Z) spider acts on the i -th logical qubit; on the right-hand side, phase-free X (or Z) spiders act on all physical qubits wherever \bar{X}_i (or \bar{Z}_i) has support. Therefore, any type of L can be pushed through E , resulting in a diagram P which satisfies $EL = PE$. \square

In [26], it was proved that a physical implementation P of a logical operator L satisfies $L = E^\dagger PE$. This is implied by $EL = PE$ as $E^\dagger E = I$.

4 Graphical Morphing of CSS Codes

One way to transform CSS codes is known as *code morphing*. It provides a systematic framework to construct new codes from an existing code while preserving the number of logical qubits in the morphed code. Here, we present this procedure through the rewrites of the encoder diagram using the ZX calculus. Let us start by revisiting the code morphing definition in [62].

Definition 4.1. Let S be a stabilizer group and \mathcal{C} be its joint $+1$ eigenspace. \mathcal{C} is called the parent code. Let Q denote the set of physical qubits of \mathcal{C} and $R \subseteq Q$. Then $S(R)$ is a subgroup of S generated by all stabilizers of S that are fully supported on R . Let $\mathcal{C}(R)$ be the joint $+1$ eigenspace of $S(R)$, and $\mathcal{C}(R)$ is called the child code. Given the parent code encoder $E_{\mathcal{C}}$, concatenate it with the inverse of the child code encoder $E_{\mathcal{C}(R)}^\dagger$. This gives the morphed code $\mathcal{C}_{\setminus R}$.

Fig. 5 provides two equivalent interpretations for the code morphing process. In Fig. 5a, Def. 4.1 is depicted by the circuit diagram. Since $E_{\mathcal{C}(R)}$ is an isometry, $E_{\mathcal{C}(R)}^\dagger E_{\mathcal{C}(R)} = I$. By construction, the equation shown in Fig. 5a holds [62]. Moreover, the parameters of $\mathcal{C} = \llbracket n, k, d \rrbracket$, $\mathcal{C}(R) = \llbracket n_1, k_1, d_1 \rrbracket$, and $\mathcal{C}_{\setminus R} = \llbracket n_2, k_2, d_2 \rrbracket$ are characterized below. Let m, m_1, m_2 be the number of stabilizer generators for \mathcal{C} , $\mathcal{C}(R)$, and $\mathcal{C}_{\setminus R}$ respectively. Then

$$n_2 = n - n_1 + k_1, \quad k_2 = k, \quad m_2 = (n - k) - (n_1 - k_1) = m - m_1, \quad d_1, d_2 \in \mathbb{N}.$$

Fig. 5b provides a concrete example of applying Def. 4.1 to the $\llbracket 7, 1, 3 \rrbracket$ Steane code, where $S = \{1, 2, 3, 4, 5, 6, 7\}$ and $R = \{2, 3, 6, 7\}$. As a result, the $\llbracket 5, 1, 2 \rrbracket$ code is morphed from the parent code along with the $\llbracket 4, 2, 2 \rrbracket$ child code. This morphed code inherits a fault-tolerant implementation of the Clifford group from the $\llbracket 7, 1, 3 \rrbracket$ code, which has a transversal implementation of the logical Clifford operators. This morphing process is represented in the ZX diagram by cutting the edges labelled by $\bar{1}$ and $\bar{2}$ adjacent to the X spider. This is equivalent to concatenating the ZX diagram of $E_{\llbracket 4, 2, 2 \rrbracket}^\dagger$ in Fig. 5a.

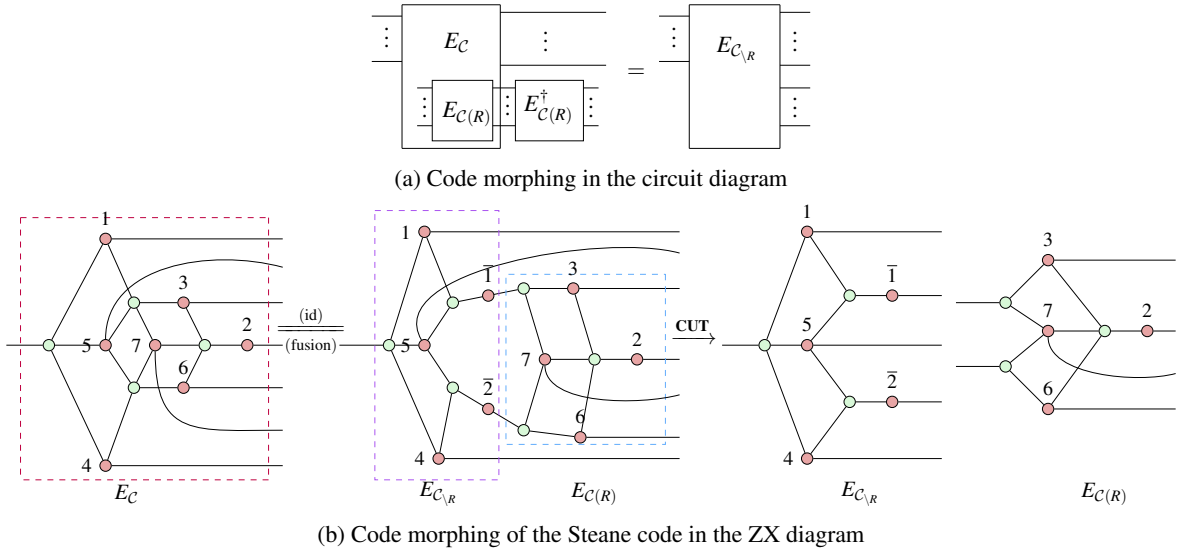


Figure 5: Code morphing can be visualized using both circuit and ZX diagrams. In Fig. 5a, code morphing is viewed as a concatenation of the parent code encoder $E_{\mathcal{C}}$ and the inverse of the child code encoder $E_{\mathcal{C}(R)}^\dagger$. In Fig. 5b, the encoder $E_{\mathcal{C}}$ of the Steane code is represented in the ZX normal form. As described in Proc. 4.1, by applying ZX rules (id) and (fusion) in Fig. 2, we can perform code morphing by bipartitioning it into the encoder $E_{\mathcal{C}_{\setminus R}}$ of the morphed code $\mathcal{C}_{\setminus R} = \llbracket 5, 1, 2 \rrbracket$, and the encoder $E_{\mathcal{C}(R)}$ of the child code $\mathcal{C}(R) = \llbracket 4, 2, 2 \rrbracket$.

Next, we generalize the notion of code morphing and show how ZX calculus could be used to study these relations between the encoders of different CSS codes. More precisely, we provide an algorithm to morph a new CSS code from an existing CSS code.

Procedure 4.1. *Given a parent code C and a child code $C(R)$ satisfying Def. 4.1, construct the encoder of C in the ZX normal form. Then the code morphing proceeds as follows:*

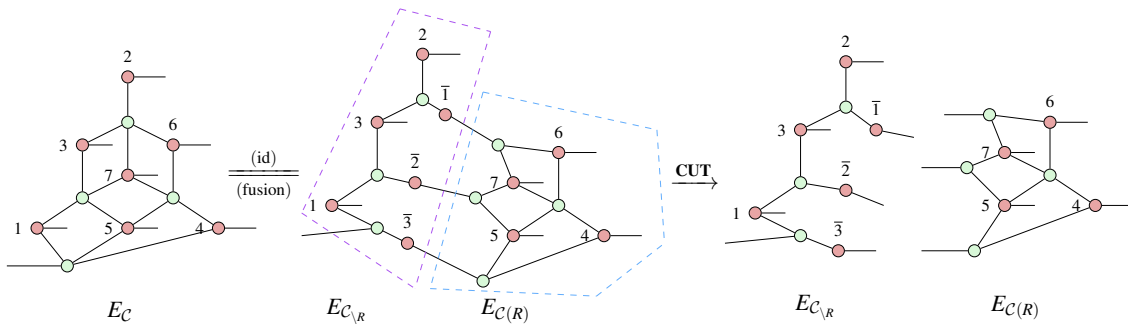
- (a) *Unfuse every Z spider which is supported on c qubits within R and f qubits outside R , $c \neq 0$, $f \neq 0$.*
- (b) *Add an identity X spider between each pair of Z spiders being unfused in step (a).*
- (c) *Cut the edge between every identity X spider and the Z spiders supported on the f qubits in R .*

It follows that the subdiagram containing R corresponds to the ZX normal form of $E_{C(R)}$. It has the same number of X spiders as R , so $n_1 = |R|$. Suppose that there are h Z spiders being unfused. Then h must be bounded by the number of Z spiders in the ZX normal form of E_C . As each spider unfusion introduces a logical qubit to $C(R)$, $k_1 = h$. On the other hand, the complement subdiagram contains $n - n_1 + k_1$ X spiders as each edge cut introduces a new X spider into the complement subdiagram. It also contains k logical qubits as the input edges in the ZX normal form of E_C are invariant throughout the spider-unfusing and edge-cutting process. This gives the ZX normal form for the encoder of the morphed code $C_{\setminus R} = \llbracket n_2, k_2, d_2 \rrbracket$, where $n_2 = n - n_1 + k_1$, $k_2 = k$, $d_2 \in \mathbb{N}$. As a result, the ZX normal form of E_C is decomposed into the ZX normal forms of $E_{C(R)}$ and $E_{C_{\setminus R}}$ respectively.

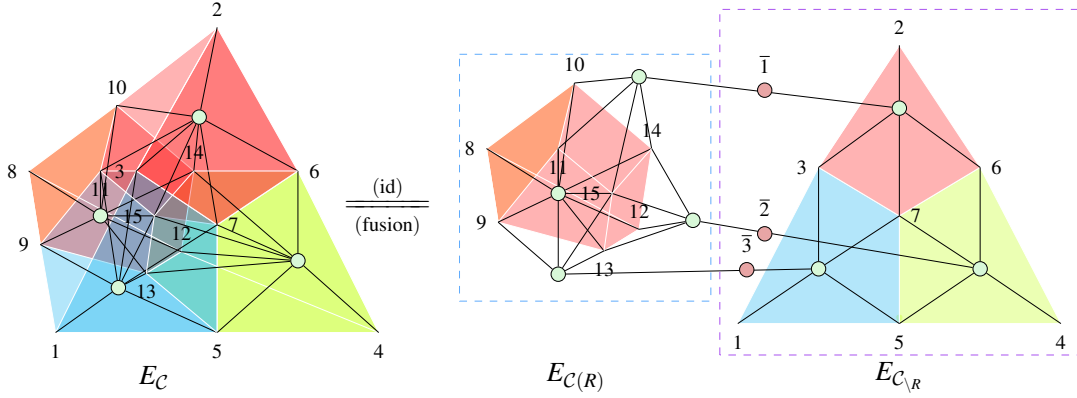
As the XZ and ZX normal forms are equivalent for CSS codes, Proc. 4.1 can be carried out for the XZ normal form by inverting the roles of Z and X at each step.

Here, we exemplify the application of Proc. 4.1 by morphing two simple CSS codes. Unlike Fig. 5b, Ex. 4.1 chooses a different subset of qubits, $R = \{4, 5, 6, 7\}$, to obtain the $\llbracket 6, 1, 1 \rrbracket$ morphed code. In Ex. 4.2, we visualize the $\llbracket 10, 1, 2 \rrbracket$ code morphing from the $\llbracket 15, 1, 3 \rrbracket$ quantum Reed-Muller code. The $\llbracket 10, 1, 2 \rrbracket$ code is interesting because it inherits a fault-tolerant implementation of the logical T gate from its parent code, which has a transversal implementation of the logical T gate.

Example 4.1. *Let the parent code C be the Steane code and the child code be $C(R) = \llbracket 4, 3, 1 \rrbracket$. By Proc. 4.1, we obtain the morphed code $C_{\setminus R} = \llbracket 6, 1, 1 \rrbracket$. Note that for $C(R)$, there is one X-type stabilizer generator and no Z-type stabilizer generator. This means that $C(R)$ cannot detect a single-qubit X error, so it has a distance of 1. In $C_{\setminus R}$, the physical qubit labelled $\bar{3}$ is not protected by any X-type stabilizer. Therefore, $C_{\setminus R}$ is of distance 1.*



Example 4.2. *Let the parent code C be the quantum Reed-Muller and the child code be $C(R) = \llbracket 8, 3, 2 \rrbracket$. By Proc. 4.1, we obtain the morphed code $C_{\setminus R} = \llbracket 10, 1, 2 \rrbracket$. For brevity, the X spiders representing physical qubits and the logical qubit wires inputting to the Z spiders are omitted.*



5 Graphical Code Switching of CSS Codes

Another way to transform CSS codes is known as *code switching*. It is a widely studied technique in quantum error correction. Codes with complementary fault-tolerant gate sets are switched between each other to realize a universal set of logical operations. As a case study, we focus on the code switching protocol between the Steane code and the quantum Reed-Muller code [2, 51, 53]. Since this process is bidirectional, the reasoning for one direction can be simply adjusted for the opposite direction. Recall in Lem. 2.1, we showed that the extended Steane code is equivalent to the Steane code up to some auxiliary state. In what follows, we focus on the *backward switching* from the quantum Reed-Muller code to the extended Steane code.

Using the ZX calculus, we provide a graphical interpretation for the backward code switching. More precisely, it is visualized as gauge-fixing the $[[15, 1, 3, 3]]$ subsystem code, followed by a sequence of syndrome-determined recovery operations.

We first characterize the relations between the quantum Reed-Muller code, the extended Steane code, and the $[[15, 1, 3, 3]]$ subsystem code. For brevity, we denote these codes as C_{qrm} , C_{ex} and C_{sub} , and their respective encoders as E_{qrm} , E_{ex} , and E_{sub} .

Lemma 5.1. *When the three gauge qubits are in the $|+++ \rangle$ state, C_{sub} is equal to C_{ex} , as shown in Fig. 6.*

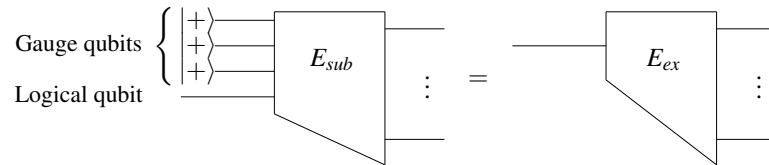
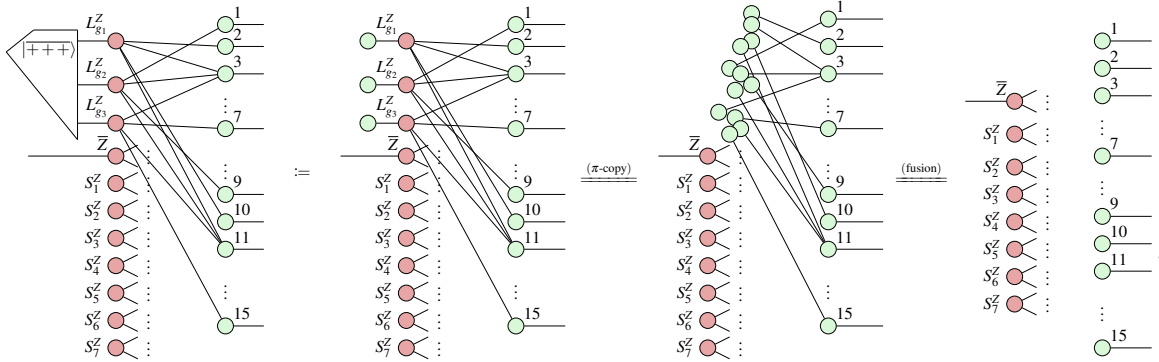


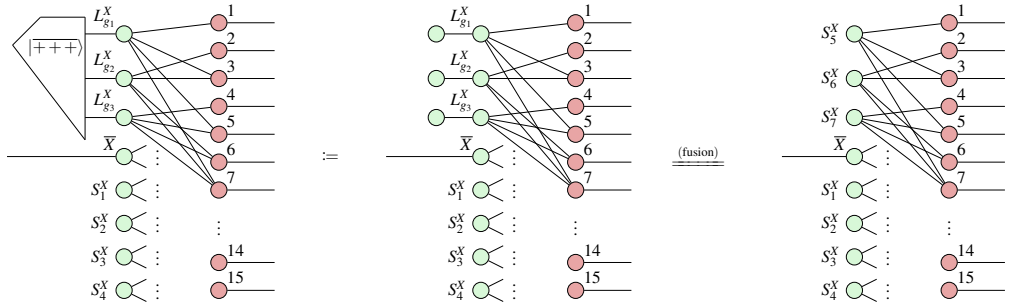
Figure 6: C_{sub} is equivalent to C_{ex} up to a fixed state of gauge qubits.

Proof. According to Def. 2.3, represent E_{sub} in the XZ normal form, with Z-type stabilizer generators S_i^z , Z-type gauge operators $L_{g_j}^z$, and one logical Z operator \bar{Z} , $1 \leq i \leq 7$, $1 \leq j \leq 3$. After applying a

sequence of rewrite rules, we obtain exactly the XZ normal form for E_{ex} .



Alternatively, if one chooses to represent E_{sub} in the ZX normal form, the proof proceeds by applying the (fusion) rule to the Z spiders and identifying the gauge operators $L_{g_1}^X, L_{g_2}^X, L_{g_3}^X$ of C_{sub} as the stabilizers S_5^X, S_6^X, S_7^X of C_{ex} , respectively:



□

Corollary 5.1. When the three gauge qubits are in the $|000\rangle$ state, C_{sub} is equal to C_{qrm} .

In [2, 51], code switching is described as a *gauge fixing* process. Further afield, [64] provides a generic recipe to gauge-fix a CSS subsystem code. Here, we generalize Lem. 5.1 and describe how to gauge-fix C_{sub} to C_{ex} using the ZX calculus.

Proposition 5.2. Gauge-fixing C_{sub} in the following steps results in C_{ex} , as shown in Fig. 7.

- (a) Measure three X-type gauge operators $L_{g_i}^X$ and obtain the corresponding outcomes $k_1, k_2, k_3 \in \mathbb{Z}_2$.
- (b) When $k_i = 1$, the gauge qubit i has collapsed to the wrong state $|-\rangle$. Apply the Z-type recovery operation $L_{g_i}^Z$.

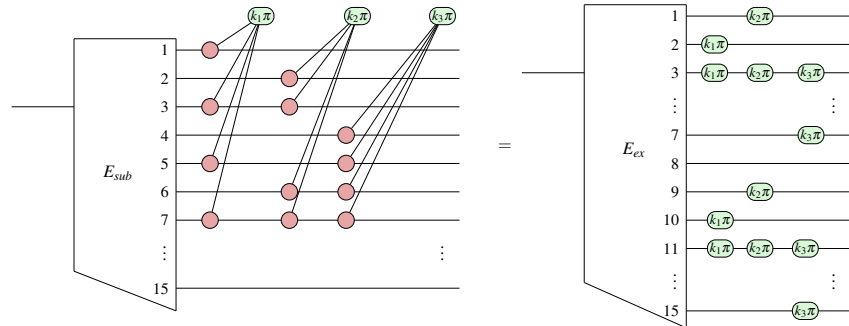
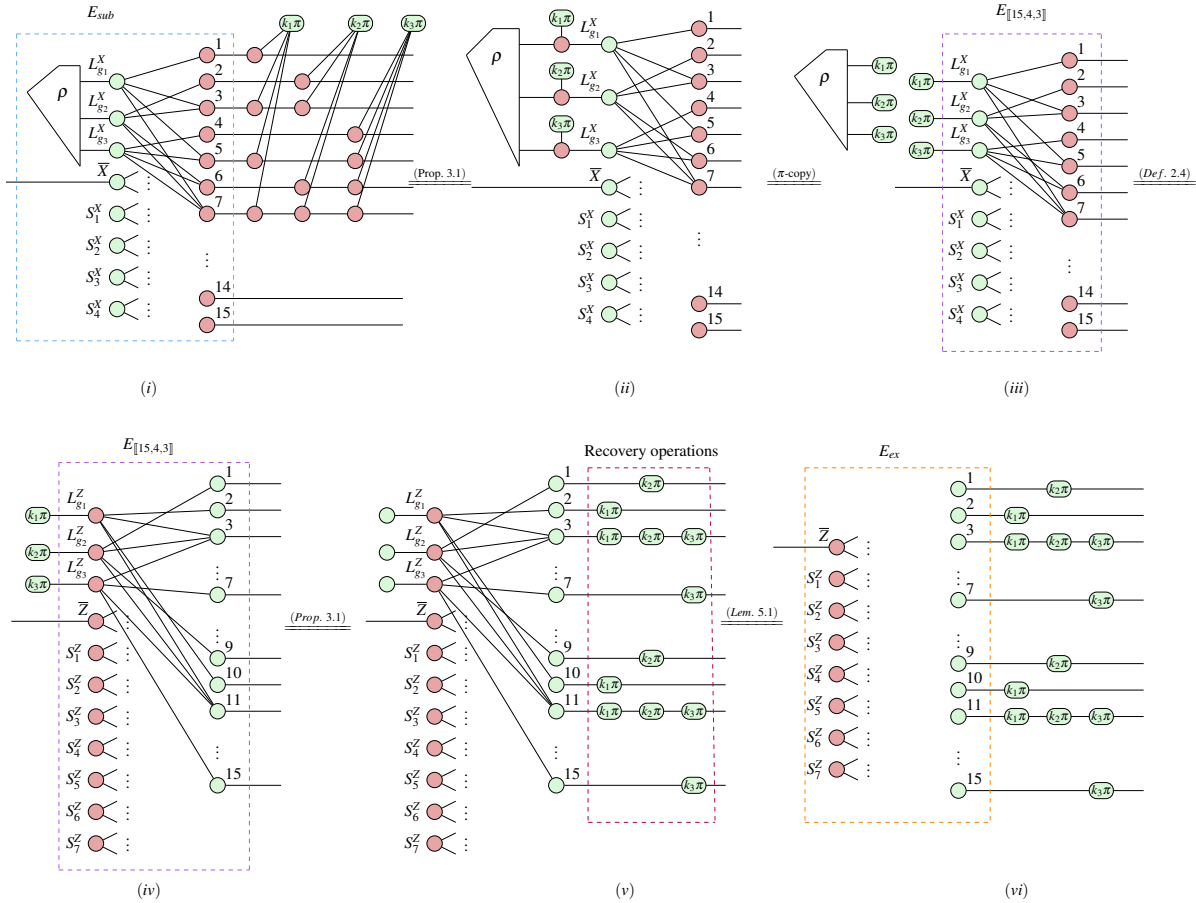


Figure 7: Gauge-fixing C_{sub} to C_{ex} in the circuit diagram.

Proof. By Def. 3.1, construct the ZX normal form of E_{sub} in the blue dashed box of (i). Then the three gauge operators $L_{g_i}^X$ are measured in step (a). The subsequent equalities follow from Figs. 2 and 3. Next, we observe that the purple dashed box in (iii) is exactly the encoder of the $[[15,4,3]]$ stabilizer code. By Lemma 3.2 in [39], it can be equivalently expressed in the XZ normal form, as in (iv). By Prop. 3.1, pushing each Z spider with the phase $k_i\pi$ across $E_{[[15,4,3]]}$ results in (v). In step (b), Pauli Z operators are applied based upon the measurement outcome k_i , which corresponds to the recovery operations in the red dashed box of (v). After that, the gauge qubits of \mathcal{C}_{sub} are set to the $|+++\rangle$ state. By Lem. 5.1, we obtain the XZ normal form for E_{ex} , as shown in the orange dashed box of (vi). Therefore, the equation in Fig. 7 holds. \square



We sum up by explaining how to obtain \mathcal{C}_{ex} and \mathcal{C}_{qrm} by gauge-fixing \mathcal{C}_{sub} . In Prop. 5.2, we showed that measuring the X-type gauge operators $L_{g_i}^X$ followed by the Z-type recovery operations $L_{g_i}^Z$ is equivalent to adding $L_{g_i}^X$ to the stabilizer group \mathcal{S}_{sub} . This results in the formation of \mathcal{C}_{ex} . Analogously, measuring the Z-type gauge operators $L_{g_i}^Z$ followed by the X-type recovery operations $L_{g_i}^X$ is equivalent to adding $L_{g_i}^Z$ to \mathcal{S}_{sub} . Thus, we obtain \mathcal{C}_{qrm} .

Alternatively, gauge-fixing \mathcal{C}_{sub} can be viewed as a way of switching between \mathcal{C}_{ex} and \mathcal{C}_{qrm} [2, 53]. As an example, in Fig. 8, we visualize the measurement of $L_{g_1}^X := X_1X_3X_5X_7$ in order to switch from \mathcal{C}_{qrm} to \mathcal{C}_{ex} . The effect of measuring other X-type gauge operators can be reasoned analogously.

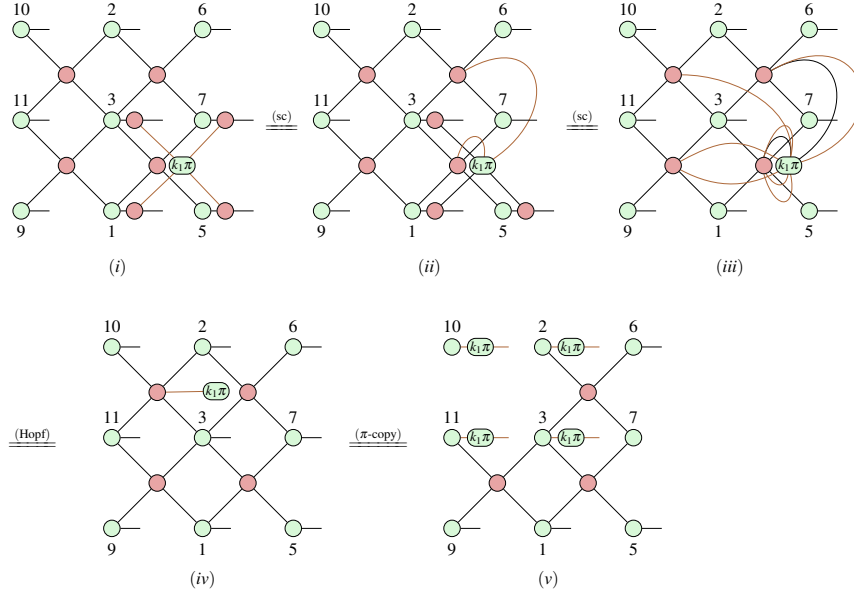


Figure 8: The switching from \mathcal{C}_{qrm} to \mathcal{C}_{ex} provides an alternative interpretation of Prop. 5.2. After measuring $L_{g_1}^X$, $L_{g_1}^Z$ is removed from the stabilizer group \mathcal{S}_{qrm} and the recovery operation is performed based on the measurement syndrome. Note that unrelated X and Z spiders are omitted from the ZX diagrams.

By Def. 3.1, construct the XZ normal form of E_{qrm} in (i). Then measure $L_{g_1}^X$ and apply a sequence of rewrite rules to the ZX diagram. In (v), the stabilizer $L_{g_1}^Z := Z_2Z_3Z_{10}Z_{11}$ is removed from the stabilizer group \mathcal{S}_{qrm} . Meanwhile, the recovery operation can be read off from the graphical derivation: $(Z_2Z_3Z_{10}Z_{11})^{k_1} = (L_{g_1}^Z)^{k_1}$, $k_1 \in \mathbb{Z}_2$.

Overall, ZX visualization provides a deeper understanding of the gauge fixing and code switching protocols. On top of revealing the relations between different CSS codes' encoders, it provides a simple yet rigorous test for various fault-tolerant protocols. Beyond this, it will serve as an intuitive guiding principle for the implementation of various logical operations.

6 Conclusion

In this paper, we generalize the notions in [39] and describe a normal form for CSS subsystem codes. Built upon the equivalence between CSS codes and the phase-free ZX diagrams, we provide a bidirectional rewrite rule to establish a correspondence between a logical ZX diagram and its physical implementation. With these tools in place, we provide a graphical representation of two code transformation techniques: code morphing, a procedure that transforms a code through unfusing spiders for the stabilizer generators, and gauge fixing, where different stabilizer codes can be obtained from a common subsystem code. These explicit graphical derivations show how the ZX calculus and graphical encoder maps relate several equivalent perspectives on these code transforming operations, allowing potential utilities of ZX to simplify fault-tolerant protocols and verify their correctness.

Looking ahead, many questions remain. It is still not clear how to present the general code deformation of CSS codes using phase-free ZX diagrams. Besides, understanding code concatenation through the lens of ZX calculus may help derive new and better codes. In addition, it would be interesting to look at other code modification techniques derived from the classical coding theory [48].

7 Acknowledgement

The authors would like to thank Thomas Scruby for enlightening discussions. SML and MM wish to thank NTT Research for their financial and technical support. This work was supported in part by Canada's NSERC. Research at IQC is supported in part by the Government of Canada through Innovation, Science and Economic Development Canada. Research at Perimeter Institute is supported in part by the Government of Canada through the Department of Innovation, Science and Economic Development Canada and by the Province of Ontario through the Ministry of Colleges and Universities. LY is supported by an Oxford - Basil Reeve Graduate Scholarship at Oriel College with the Clarendon Fund.

References

- [1] Scott Aaronson & Daniel Gottesman (2004): *Improved simulation of stabilizer circuits*. *Physical Review A* 70(5), p. 052328, doi:10.1103/PhysRevA.70.052328.
- [2] Jonas T Anderson, Guillaume Duclos-Cianci & David Poulin (2014): *Fault-tolerant conversion between the steane and reed-muller quantum codes*. *Physical Review Letters* 113(8), p. 080501, doi:10.1103/PhysRevLett.113.080501.
- [3] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell et al. (2019): *Quantum supremacy using a programmable superconducting processor*. *Nature* 574(7779), pp. 505–510, doi:10.1038/s41586-019-1666-5.
- [4] Miriam Backens (2014): *The ZX-calculus is complete for stabilizer quantum mechanics*. *New Journal of Physics* 16(9), p. 093021, doi:10.1088/1367-2630/16/9/093021.
- [5] H. Bombín & M. A. Martin-Delgado (2006): *Topological quantum distillation*. *Physical Review Letters* 97, p. 180501, doi:10.1103/PhysRevLett.97.180501.
- [6] H. Bombín & M. A. Martin-Delgado (2007): *Topological computation without braiding*. *Physical Review Letters* 98, p. 160502, doi:10.1103/PhysRevLett.98.160502.
- [7] Héctor Bombín (2015): *Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes*. *New Journal of Physics* 17(8), p. 083002, doi:10.1088/1367-2630/17/8/083002.
- [8] Robert I. Booth & Titouan Carette: *Complete ZX-calculi for the stabiliser fragment in odd prime dimensions*, doi:10.4230/LIPIcs.MFCS.2022.24. ISSN: 1868-8969.
- [9] A. R. Calderbank & Peter W. Shor (1996): *Good quantum error-correcting codes exist*. *Physical Review A* 54, pp. 1098–1105, doi:10.1103/PhysRevA.54.1098.
- [10] A Robert Calderbank, Eric M Rains, Peter W Shor & Neil JA Sloane (1997): *Quantum error correction and orthogonal geometry*. *Physical Review Letters* 78(3), p. 405, doi:10.1103/PhysRevLett.78.405.
- [11] Yudong Cao, Jhonathan Romero & Alán Aspuru-Guzik (2018): *Potential of quantum computing for drug discovery*. *IBM Journal of Research and Development* 62(6), pp. 6–1, doi:10.1147/JRD.2018.2888987.
- [12] Titouan Carette, Dominic Horsman & Simon Perdrix (2019): *SZX-calculus: scalable graphical quantum reasoning*. In: *44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019), Leibniz International Proceedings in Informatics (LIPIcs)* 138, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 55:1–55:15, doi:10.4230/LIPIcs.MFCS.2019.55. Available at <http://drops.dagstuhl.de/opus/volltexte/2019/10999>.
- [13] Nicholas Chancellor, Aleks Kissinger, Joschka Roffe, Stefan Zohren & Dominic Horsman (2023): *Graphical Structures for Design and Verification of Quantum Error Correction*, doi:10.48550/arXiv.1611.08012. arXiv:1611.08012.
- [14] Julien Codsì (2022): *Cutting-edge graphical stabiliser decompositions for classical simulation of quantum circuits*. Master's thesis, University of Oxford.

- [15] Bob Coecke & Ross Duncan (2008): *Interacting quantum observables*. In: *ICALP, Lecture Notes in Computer Science*, pp. 298–310, doi:10.1007/978-3-540-70583-3_25.
- [16] Bob Coecke & Ross Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016.
- [17] Bob Coecke & Aleks Kissinger (2017): *Picturing quantum processes*. Cambridge University Press, doi:10.1007/978-3-319-91376-6_6.
- [18] Bob Coecke & Quanlong Wang (2018): *ZX-rules for 2-qubit Clifford+T quantum circuits*. In Jarkko Kari & Irek Ulidowski, editors: *Reversible Computation*, Springer International Publishing, Cham, pp. 144–161, doi:10.1007/978-3-319-99498-7_10.
- [19] Oliver Cole (2022): *Quantum circuit optimisation through stabiliser reduction of Pauli exponentials*. Master’s thesis, University of Oxford. Available at <https://www.cs.ox.ac.uk/people/aleks.kissinger/theses/cole-thesis.pdf>.
- [20] Cole Comfort (2023): *The algebra for stabilizer codes*, doi:10.48550/arXiv.2304.10584. arXiv:2304.10584.
- [21] Alexander Cowtan (2022): *Qudit lattice surgery*, doi:10.48550/arXiv.2204.13228. arXiv:2204.13228.
- [22] Alexander Cowtan & Simon Burton (2023): *CSS code surgery as a universal construction*. arXiv preprint arXiv:2301.13738, doi:10.48550/arXiv.2301.13738.
- [23] Ross Duncan & Maxime Lucas (2014): *Verifying the Steane code with Quantomatic*. *Electronic Proceedings in Theoretical Computer Science* 171, pp. 33–49, doi:10.4204/eptcs.171.4. Available at <https://doi.org/10.4204/eptcs.171.4>.
- [24] Bryan Eastin & Emanuel Knill (2009): *Restrictions on transversal encoded quantum gate sets*. *Physical Review Letters* 102(11), p. 110502, doi:10.1103/PhysRevLett.102.110502.
- [25] Kun Fang, Jingtian Zhao, Xiufan Li, Yifei Li & Runyao Duan (2022): *Quantum NETWORK: from theory to practice*. arXiv preprint arXiv:2212.01226, doi:10.48550/arXiv.2212.01226.
- [26] Liam Garvie & Ross Duncan (2017): *Verifying the smallest interesting colour code with Quantomatic*. arXiv preprint arXiv:1706.02717, doi:10.4204/EPTCS.266.10.
- [27] Craig Gidney (2022): *A pair measurement surface code on pentagons*, doi:10.48550/arXiv.2206.12780. arXiv:2206.12780.
- [28] Craig Gidney & Austin G. Fowler (2019): *Efficient magic state factories with a catalyzed $|CCZ\rangle$ to $2|T\rangle$ transformation*. *Quantum* 3, p. 135, doi:10.22331/q-2019-04-30-135.
- [29] Craig Gidney & Austin G. Fowler (2019): *Flexible layout of surface code computations using AutoCCZ states*, doi:10.48550/arXiv.1905.08916. Available at <https://arxiv.org/abs/1905.08916>.
- [30] Jonathan Gorard, Manojna Namuduri & Xerxes D. Arsiwalla (2021): *ZX-calculus and extended Wolfram model systems II: fast diagrammatic reasoning with an application to quantum circuit simplification*. arXiv preprint arXiv:2103.15820, doi:10.48550/arXiv.2103.15820.
- [31] D Gottesman (1999): *The Heisenberg representation of quantum computers*. In: *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, pp. 32–43, doi:10.48550/arXiv.quant-ph/9807006.
- [32] Daniel Gottesman (1997): *Stabilizer codes and quantum error correction*. Ph.D. thesis, California Institute of Technology, doi:10.48550/arXiv.quant-ph/9705052.
- [33] Daniel Gottesman (1998): *Theory of fault-tolerant quantum computation*. *Physical Review A* 57, pp. 127–137, doi:10.1103/PhysRevA.57.127.
- [34] Daniel Gottesman (2022): *Opportunities and challenges in fault-tolerant quantum computation*. arXiv preprint arXiv:2210.15844, doi:10.48550/arXiv.2210.15844.
- [35] Amar Hadzihasanovic, Kang Feng Ng & Quanlong Wang (2018): *Two complete axiomatisations of pure-state qubit quantum computing*. In: *ACM/IEEE Symposium on Logic in Computer Science, LICS ’18*, Association for Computing Machinery, New York, NY, USA, p. 502–511, doi:10.1145/3209108.3209128.

- [36] Alexander Tianlin Hu & Andrey Boris Khesin (2022): *Improved graph formalism for quantum circuit simulation*. *Physical Review A* 105(2), doi:10.1103/physreva.105.022432.
- [37] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2020): *Completeness of the ZX-calculus*. *Logical Methods in Computer Science* Volume 16, Issue 2, doi:10.23638/LMCS-16(2:11)2020.
- [38] Andrey Boris Khesin, Jonathan Z. Lu & Peter W. Shor (2023): *Graphical quantum Clifford-encoder compilers from the ZX calculus*, doi:10.48550/arXiv.2301.02356. Available at <https://arxiv.org/abs/2301.02356>.
- [39] Aleks Kissinger (2022): *Phase-free ZX diagrams are CSS codes (... or how to graphically grok the surface code)*. *arXiv preprint arXiv:2204.14038*, doi:10.48550/arXiv.2204.14038.
- [40] Aleks Kissinger & John van de Wetering (2022): *Simulating quantum circuits with ZX-calculus reduced stabiliser decompositions*. *Quantum Science and Technology* 7(4), p. 044001, doi:10.1088/2058-9565/ac5d20.
- [41] Emanuel Knill & Raymond Laflamme (1997): *Theory of quantum error-correcting codes*. *Physical Review A* 55(2), p. 900, doi:10.1103/PhysRevLett.84.2525.
- [42] Emanuel Knill, Raymond Laflamme & Wojciech Zurek (1996): *Threshold accuracy for quantum computation*. *arXiv preprint quant-ph/9610011*, doi:10.48550/arXiv.quant-ph/9610011.
- [43] David Kribs, Raymond Laflamme & David Poulin (2005): *Unified and generalized approach to quantum error correction*. *Physical Review Letters* 94, p. 180501, doi:10.1103/PhysRevLett.94.180501.
- [44] Aleksander Kubica & Michael E. Beverland (2015): *Universal transversal gates with color codes - a simplified approach*. *Physical Review A* 91(3), p. 032330, doi:10.1103/PhysRevA.91.032330. arXiv:1410.0069.
- [45] Aleksander Kubica & Michael Vasmer (2022): *Single-shot quantum error correction with the three-dimensional subsystem toric code*. *Nature Communications* 13(1), p. 6272, doi:10.1038/s41467-022-33923-4.
- [46] Adrian Lehmann, Ben Caldwell & Robert Rand (2022): *VyZX: a vision for verifying the ZX calculus*. *arXiv preprint arXiv:2205.05781*, doi:10.48550/arXiv.2205.05781.
- [47] Sebastian Leontica, F Tennie & T Farrow (2021): *Simulating molecules on a cloud-based 5-qubit IBM-Q universal quantum computer*. *Communications Physics* 4(1), p. 112, doi:10.1038/s42005-021-00616-1.
- [48] F. J. MacWilliams & N. J. A. Sloane (1977): *The theory of error-correcting codes*. Elsevier.
- [49] Tommy McElvanney & Miriam Backens (2022): *Complete flow-preserving rewrite rules for MBQC patterns with Pauli measurements*, doi:10.48550/arXiv.2205.02009. arXiv:2205.02009.
- [50] Michael A. Nielsen & Isaac L. Chuang (2010): *Quantum computation and quantum information*, 10th anniversary ed edition. Cambridge University Press, doi:10.1017/CBO9780511976667.
- [51] Adam Paetznick & Ben W. Reichardt (2013): *Universal fault-tolerant quantum computation with only transversal gates and error correction*. *Physical Review Letters* 111, p. 090505, doi:10.1103/PhysRevLett.111.090505.
- [52] David Poulin (2005): *Stabilizer formalism for operator quantum error correction*. *Physical Review Letters* 95(23), p. 230504, doi:10.1103/PhysRevLett.95.230504.
- [53] Dong-Xiao Quan, Li-Li Zhu, Chang-Xing Pei & Barry C Sanders (2018): *Fault-tolerant conversion between adjacent Reed–Muller quantum codes based on gauge fixing*. *Journal of Physics A: Mathematical and Theoretical* 51(11), p. 115305, doi:10.1088/1751-8121/aaad13.
- [54] Alexis T. E. Shaw, Michael J. Bremner, Alexandru Paler, Daniel Herr & Simon J. Devitt (2022): *Quantum computation on a 19-qubit wide 2d nearest neighbour qubit array*, doi:10.48550/arXiv.2212.01550. Available at <https://arxiv.org/abs/2212.01550>.
- [55] Peter W. Shor (1995): *Scheme for reducing decoherence in quantum computer memory*. *Physical Review A* 52, pp. R2493–R2496, doi:10.1103/PhysRevA.52.R2493.
- [56] Will Simmons (2021): *Relating measurement patterns to circuits via Pauli flow*. *Electronic Proceedings in Theoretical Computer Science* 343, pp. 50–101, doi:10.4204/eptcs.343.4.

- [57] Andrew Steane (1996): *Multiple-particle interference and quantum error correction*. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452(1954), pp. 2551–2577, doi:10.1098/rspa.1996.0136.
- [58] Andrew M Steane (1996): *Simple quantum error-correcting codes*. *Physical Review A* 54(6), p. 4741, doi:10.1103/PhysRevA.54.4741.
- [59] Andrew M Steane (1999): *Quantum Reed-Muller codes*. *IEEE Transactions on Information Theory* 45(5), pp. 1701–1703, doi:10.1109/18.771249.
- [60] Felix Tennie & Tim Palmer (2022): *Quantum computers for weather and climate prediction: the good, the bad and the noisy*. *arXiv preprint arXiv:2210.17460*, doi:10.48550/arXiv.2210.17460.
- [61] Alex Townsend-Teague & Konstantinos Meichanetzidis (2022): *Classifying complexity with the ZX-calculus: Jones polynomials and Potts partition functions*. In: *Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science (EPTCS 287)*, Oxford, p. 313–344. Available at https://www.qplconference.org/proceedings2022/QPL_2022_paper_4.pdf.
- [62] Michael Vasmer & Aleksander Kubica (2022): *Morphing quantum codes*. *PRX Quantum* 3(3), doi:10.1103/prxquantum.3.030319.
- [63] Renaud Vilmart (2019): *A near-minimal axiomatisation of ZX-calculus for pure qubit quantum mechanics*. In: *LICS 2019*, pp. 1–10, doi:10.1109/LICS.2019.8785765.
- [64] Christophe Vuillot, Lingling Lao, Ben Criger, Carmen García Almudéver, Koen Bertels & Barbara M Terhal (2019): *Code deformation and lattice surgery are gauge fixing*. *New Journal of Physics* 21(3), p. 033028, doi:10.1088/1367-2630/ab0199.
- [65] Quanlong Wang (2018): *Qutrit ZX-calculus is complete for stabilizer quantum mechanics*. *Electronic Proceedings in Theoretical Computer Science* 266, pp. 58–70, doi:10.4204/EPTCS.266.3.
- [66] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*. *arXiv preprint arXiv:2012.13966*, doi:10.48550/arXiv.2012.13966.
- [67] Theodore J. Yoder (2017): *Universal fault-tolerant quantum computation with Bacon-Shor codes*, doi:10.48550/arXiv.1705.01686. *arXiv:1705.01686*.
- [68] Qingling Zhu, Sirui Cao, Fusheng Chen, Ming-Cheng Chen, Xiawei Chen, Tung-Hsun Chung, Hui Deng, Yajie Du, Daojin Fan, Ming Gong et al. (2022): *Quantum computational advantage via 60-qubit 24-cycle random circuit sampling*. *Science bulletin* 67(3), pp. 240–245, doi:10.1016/j.scib.2021.10.017.

Light-Matter Interaction in the ZXW Calculus

Giovanni de Felice¹ Razin A. Shaikh^{1,2} Boldizsár Poór¹
Lia Yeh^{1,2} Quanlong Wang¹ Bob Coecke¹

¹Quantinuum, 17 Beaumont Street, Oxford, OX1 2NA, United Kingdom

²University of Oxford, Oxford, United Kingdom

In this paper we develop a graphical calculus to rewrite photonic circuits involving light-matter interactions and non-linear optical effects. We introduce the infinite ZW calculus, a graphical language for linear operators on the bosonic Fock space which captures both linear and non-linear photonic circuits. This calculus is obtained by combining the QPath calculus, a diagrammatic language for linear optics, and the recently developed qudit ZXW calculus, a complete axiomatisation of linear maps between qudits. It comes with a “lifting” theorem allowing to prove equalities between infinite operators by rewriting in the ZXW calculus. We give a method for representing bosonic and fermionic Hamiltonians in the infinite ZW calculus. This allows us to derive their exponentials by diagrammatic reasoning. Examples include phase shifts and beam splitters, as well as non-linear Kerr media and Jaynes-Cummings light-matter interaction.

1 Introduction

Graphical languages are a powerful tool for understanding, verifying and developing software for quantum computing. The *ZX calculus* [12, 18, 50] is a graphical language for qubit quantum computing which is currently used to solve a range of different quantum computing tasks, including compilation [47], optimization [22, 38], machine learning [48, 54], measurement-based quantum computing [23, 4], error-correction [37], and also education [16]. It was introduced in 2007 by Coecke and Duncan [13, 14, 15] to model the interaction of complementary observables. In 2010, Coecke and Kissinger proposed to introduce the W state [24] in the calculus [17] and showed that rational arithmetic operations had simple graphical representations [19]. Building on their work, Hadzihasanovic gave the first complete axiomatisation for qubit quantum computing [33], known as the *ZW calculus*. The completeness of an equational theory is important as it shows that the language is rich enough to prove all equalities of the underlying linear maps; therefore, there are ‘no missing rules’. The same techniques were subsequently used to prove completeness of the ZX calculus [42, 35, 39, 36]. The rewriting power and the ability of representing sums using W nodes motivated the development of the *ZXW calculus*, allowing the first diagrammatic treatment of Hamiltonians and exponentiation [46], as well as integration and differentiation [54]. A natural generalisation of ZXW to higher dimensions resulted in the first complete axiomatisation for qudit quantum computing [43].

Graphical languages have only recently been applied to photonic quantum computing. The ZX calculus is now used to reason about compilation of linear optical circuits, and particularly its MBQC aspects [56, 21]. It is also used to represent error correction codes for fault-tolerant quantum computing [5]. These approaches all rely on an encoding of qubits or qubit lattices on a state of multiple photons. Graphical languages for reasoning about the underlying physics

of interacting photons have so far been restricted to the study of *linear* optical processes. The language of interferometers built from phases and beam splitters is well understood for single photons [44, 10, 11]. Different methods can be used to compute the amplitudes of linear optical circuits with multiple photons [1]. The *QPath calculus* [21] is a recent graphical approach to this problem which decomposes circuits into simpler primitive operations. However, it has a limited rewrite system where diagrams involving multiple photons are split into sums of terms. Moreover, *non-linear* optical phenomena appear throughout photonic quantum science. Non-linearities are used to construct photon sources by parametric down conversion [32] or by emission from a 2-level atom in a cavity [49, 55]. They include mixed boson-fermion systems such as the ones studied in quantum chemistry [27]. They also include non-linear Kerr media which allow the optical specification of universal quantum gates [3], with recently proposed graphene-based implementations [8]. These non-linear effects rely on different forms of light-matter interaction. Hence, we need a graphical calculus that could represent interaction between bosons and fermions.

The semantics of a photonic graphical calculus is based on infinite dimensional Fock space. However, graphical calculi with infinite dimensional semantics do not have a strong reasoning system. Even simple equalities, such as the snake equation, are not well-defined. To alleviate this problem, we truncate the infinite dimensional vector space to finite dimensions for which we have a complete graphical calculus, i.e. the ZXW calculus. Then we prove a lifting theorem to show that, under mild conditions, the results derived in this truncated graphical calculus also hold in infinite dimensional vector space. This idea is similar to that of Gogioso and Genovese [29, 30, 31] where the transfer theorem from non-standard analysis is used to prove results about infinite dimensional Hilbert spaces. However, by restricting our attention to states with finite support and maps preserving this property, our lifting theorem admits a simpler proof which does not require non-standard analysis.

Our approach of leveraging a finite dimensional calculus has a key advantage over constraining to solely infinite dimensional formalisms. We can now reason about not only bosons, but also other finite dimensional systems such as fermions, altogether in the same framework. Hence, we achieve a unified calculus powerful enough to reason about light-matter interactions.

This paper provides a framework for representing and rewriting quantum optics in the ZXW calculus. We start by introducing the QPath calculus and the ZXW calculus (Section 2). The first is a graphical language for linear optics [21], describing the behaviour of bosonic maps in the category $\mathbf{Vect}_{\mathbb{N}}$ of linear operators on the bosonic Fock space. The second is a complete axiomatisation $[\cdot]_d : \mathbf{ZXW}_d \xrightarrow{\sim} \mathbf{Vect}_d$ of the category of linear maps between qudits [43]. In Section 3 and Section 4 we present our main contributions: (1) we introduce a more expressive calculus \mathbf{ZW}_{∞} for the category of linear operators on the Fock space $\mathbf{Vect}_{\mathbb{N}}$, (2) we give a truncated interpretation $\mathcal{T}_d : \mathbf{ZW}_{\infty} \rightarrow \mathbf{ZXW}_d$ and prove Theorem 4.2 which allows to derive equalities in \mathbf{ZW}_{∞} by lifting them from \mathbf{ZXW}_d for ‘big enough’ d , (3) we give an axiomatisation of \mathbf{ZW}_{∞} for which we prove soundness via the lifting theorem. We end by exploring different applications of this graphical calculus in the field of non-linear optics (Section 5). We give a general method for representing quantum optical Hamiltonians in \mathbf{ZW}_{∞} , including Kerr media and Jaynes-Cummings light-matter interaction. Building on [46], we show how to exponentiate these Hamiltonians using diagrammatic techniques.

2 Preliminaries

2.1 Fock space

Consider the state space of a single bosonic mode $\llbracket 1 \rrbracket = \bigoplus_{n=0}^{\infty} \mathbb{C}$. An element in this space is usually denoted as an infinite sum $|a\rangle = \sum_{n=0}^{\infty} a_n |n\rangle$. Depending on the chosen boundedness condition for these sequences, we get different classes of operators acting on $\llbracket 1 \rrbracket$. In their work on the quantum harmonic oscillator, Vicary [52] imposed a strong normalising condition on sequences, of the form $\sum_n c^n a_n < \infty$ for any $c \in \mathbb{C}$, interpreting \bigoplus as an infinite biproduct. The valid maps are those that preserve this normalising condition. They include phases, coherent states and creation operators. In this paper, we interpret \bigoplus as an infinite direct sum. The valid states are sequences with finitely many non-zero terms. This means we do not allow coherent states in our spaces. The operators that we consider are those that send finite states to finite states; they form a category $\mathbf{Vect}_{\mathbb{N}}$.

For a system of n bosonic particles occupying m possible modes, the state space is given by the Fock space defined as follows.

$$\llbracket m \rrbracket = \bigoplus_{n=0}^{\infty} (\mathbb{C}^m)^{\tilde{\otimes} n} \simeq \llbracket 1 \rrbracket^{\otimes m}$$

where $\tilde{\otimes}$ is the symmetrised tensor product, i.e. the quotient of the tensor product given by identifying $x = x_1 \otimes \dots \otimes x_n$ with $y = y_1 \otimes \dots \otimes y_n$ whenever there is a permutation σ such that $\sigma(x) = y$. The isomorphism $\llbracket m \rrbracket \simeq \llbracket 1 \rrbracket^{\otimes m}$ is given by counting the number of photons occupying each mode [21, Proposition 3.2]. Let \mathbf{Vect} be the symmetric monoidal category of vector spaces and linear maps with tensor product \otimes . Symmetrised spaces of the form $(\mathbb{C}^m)^{\tilde{\otimes} n}$ are objects of this category. We define the infinite direct sum $\bigoplus_{n=0}^{\infty} A_n$ for $A_n \in \mathbf{Vect}$ as the set of sequences (a_0, a_1, \dots) with $a_n \in A_n$ such that $a_n = 0$ except for finitely many n . This means we only allow states $|a\rangle = \sum_{n=0}^k a_n |n\rangle \in \llbracket 1 \rrbracket$ with a finite number k of particles. We can now build our base category $\mathbf{Vect}_{\mathbb{N}}$ of linear maps on the Fock space: objects are finite tensor products of the complex space $\llbracket 1 \rrbracket$, morphisms are linear maps $f : \llbracket m \rrbracket \simeq (\llbracket 1 \rrbracket)^{\otimes m} \rightarrow (\llbracket 1 \rrbracket)^{\otimes k} \simeq \llbracket k \rrbracket$. Note that we allow unbounded linear maps as long as their domain and codomain are in the infinite direct sum. As argued by Vicary [52, Section 6], these are necessary to understand the algebraic structure of the Fock space.

Note that $\mathbf{Vect}_{\mathbb{N}}$ is not compact closed. In fact, generalising the standard cups and caps would give a state without a fixed number of particles $\sum_n |nn\rangle \notin \mathbf{Vect}_{\mathbb{N}}$. $\mathbf{Vect}_{\mathbb{N}}$ admits a dagger structure only on a subclass of morphisms. Indeed, the valid map $\epsilon : \llbracket 1 \rrbracket \rightarrow 1$, with $\epsilon |n\rangle = 1$ for all n , gives an invalid state $\sum_n |n\rangle$ when taking the adjoint. However, the dagger is well-defined for most morphisms of interest, and these are closed under composition. It is also easy to see that $\mathbf{Vect}_{\mathbb{N}}$ is enriched in weighted finite sums over the complex numbers. We now consider an interesting subclass of processes in $\mathbf{Vect}_{\mathbb{N}}$, the ones that may be performed using linear optics and creation/annihilation of single particles.

2.2 Bosonic nodes

Bosonic nodes, or split and merge maps, are linear operators on the Fock space exhibiting a bialgebra structure, equivalent to the binomial bialgebra on polynomials [40, 25]. In categorical quantum mechanics, they were first studied by Vicary and Fiore [52, 26] who characterised the map b that copies coherent states of the form $|\alpha\rangle = \sum_n \frac{\alpha^n}{\sqrt{n!}} |n\rangle$, in the sense that $b|\alpha\rangle =$

$|\alpha\rangle \otimes |\alpha\rangle$. Coherent states are usually defined as eigenvectors of the bosonic creation operator $a^\dagger |\alpha\rangle = \alpha |\alpha\rangle$. They account for Poissonian distributions of photons from coherent light sources such as lasers [28]. The split map was later studied as an anyonic generalisation of the W algebra in [34] and as a generator for Feynman diagrams in [45].

The QPath calculus [21] is graphical calculus for linear optics based on the bosonic split and merge maps. A wire in the QPath calculus corresponds to a bosonic mode and is thus interpreted as the space $\llbracket 1 \rrbracket$ with basis given by occupation numbers $|n\rangle$ for $n \in \mathbb{N}$. The interpretation of m wires is obtained using the tensor product $\llbracket m \rrbracket \simeq \llbracket 1 \rrbracket^{\otimes m}$. The main generator of QPath is the bosonic *split* map, defined as follows:

$$\text{---} \left\langle \begin{array}{c} \text{---} \\ \diagup \quad \diagdown \\ \text{---} \end{array} \right. \xrightarrow{\llbracket 1 \rrbracket} |n\rangle \mapsto \sum_{k=0}^n \binom{n}{k}^{\frac{1}{2}} |k\rangle |n-k\rangle$$

Its adjoint or dagger called *merge*, is also a generator:

$$\left. \begin{array}{c} \text{---} \\ \diagdown \quad \diagup \\ \text{---} \end{array} \right. \xrightarrow{\llbracket 1 \rrbracket} |n, m\rangle \mapsto \binom{n+m}{n}^{\frac{1}{2}} |n+m\rangle$$

We also have *endomorphisms* for any $r \in \mathbb{C}$:

$$\text{---} \boxed{r} \text{---} \xrightarrow{\llbracket 1 \rrbracket} Z(\vec{r}) : |n\rangle \mapsto r^n |n\rangle$$

or phase shifts when $r = e^{i\theta}$, then the symmetry, or *swap*:

$$\text{---} \times \text{---} \xrightarrow{\llbracket 1 \rrbracket} |n, m\rangle \mapsto |m, n\rangle$$

and, n -photon states and effects:

$$\bullet \xrightarrow{\llbracket 1 \rrbracket} |n\rangle \qquad \langle n| \xrightarrow{\llbracket 1 \rrbracket} \bullet$$

All the maps defined above send basis states to finite sums of basis states and are therefore valid morphisms in $\mathbf{Vect}_{\mathbb{N}}$. These generators satisfy the axioms of a bialgebra with endomorphisms, along with additional rules involving occupied photon states [21].

2.3 ZXW calculus

The ZXW calculus is a complete diagrammatic language for qudit quantum computing [43]. Formally, let $\mathbf{Vect}_{\mathbf{d}}$ be the symmetric monoidal category with objects tensor products of qudits \mathbb{C}^d and morphisms given by linear maps between them. Completeness means that the interpretation $\llbracket \cdot \rrbracket_d : \mathbf{ZXW}_{\mathbf{d}} \rightarrow \mathbf{Vect}_{\mathbf{d}}$ is an isomorphism. We describe the generators of the ZXW calculus along with their interpretation. Throughout the section, we fix a natural number d and use the arbitrary complex vectors $\vec{a} = (a_1, \dots, a_{d-1})$ and $\vec{b} = (b_1, \dots, b_{d-1})$. Note that here we made a slight change for the presentation of the qudit ZXW calculus in comparison to the version of [43]: we use the X spider instead of the Hadamard node as a generator now, so the definition of the former becomes a rule and the rule for the latter turns into a definition. Also we removed the (H1) rule of [43] from the rule set since we find it can be derived from other rules now.

2.3.1 Generators of ZXW

The generators of $\mathbf{ZXW}_{\mathbf{d}}$ together with their standard interpretation $\llbracket \cdot \rrbracket_d$ are:

- The Z spider,

$$n \left\{ \begin{array}{c} \vdots \\ \vdots \\ \boxed{\vec{\alpha}} \\ \vdots \\ \vdots \end{array} \right\}_m \xrightarrow{[\cdot]_d} \sum_{j=0}^{d-1} a_j |j\rangle^{\otimes m} \langle j|^{\otimes n}, \quad \text{where } a_0 = 1.$$

- The X spider, with parameter j which can be taken modulo d ,

$$n \left\{ \begin{array}{c} \vdots \\ \vdots \\ \boxed{K_j} \\ \vdots \\ \vdots \end{array} \right\}_m \xrightarrow{[\cdot]_d} \sum_{\substack{0 \leq i_1, \dots, i_m, j_1, \dots, j_n \leq d-1 \\ i_1 + \dots + i_m + j \equiv j_1 + \dots + j_n \pmod{d}}} |i_1, \dots, i_m\rangle \langle j_1, \dots, j_n|,$$

- The W node,

$$\left\langle \begin{array}{c} \vdots \\ \vdots \\ \blacktriangleleft \end{array} \right\rangle \xrightarrow{[\cdot]_d} |0 \dots 0\rangle \langle 0| + \sum_{i=1}^{d-1} (|i0 \dots 00\rangle + \dots + |00 \dots 0i\rangle) \langle i|$$

- The $swap$,

$$\begin{array}{c} \diagup \\ \diagdown \end{array} \xrightarrow{[\cdot]_d} \sum_{i,j=0}^{d-1} |ji\rangle \langle ij|.$$

2.3.2 Notations

For convenience, we introduce the following notation which will be used throughout the paper:

- The phase depicted by a green circle spider can be defined using the Z box:

$$\text{---} \textcircled{\vec{\alpha}} \text{---} := \text{---} \boxed{e^{i\vec{\alpha}}} \text{---}$$

where $\vec{\alpha} = (\alpha_1, \dots, \alpha_{d-1})$, $e^{i\vec{\alpha}} = (e^{i\alpha_1}, \dots, e^{i\alpha_{d-1}})$, and $\alpha_i \in [0, 2\pi)$.

- The cup and cap, i.e. the qudit Bell state and its transpose, are defined as follows:

$$\left(:= \textcircled{\cup} \right) := \textcircled{\cap} \quad (\text{S3})$$

- We use a yellow D box to denote the *dualiser*, with the given interpretation:

$$\text{---} \boxed{D} \text{---} := \begin{array}{c} \textcircled{\cap} \\ \textcircled{\cup} \end{array} \xrightarrow{[\cdot]_d} \sum_{i=0}^{d-1} |i\rangle \langle d-i|. \quad (\text{Du})$$

- It is useful to define the yellow triangle in terms of the W node and green spider:

$$\text{---} \triangleleft \text{---} := \left\langle \begin{array}{c} \vdots \\ \vdots \\ \blacktriangleleft \end{array} \right\rangle \xrightarrow{[\cdot]_d} I_d + \sum_{i=1}^{d-1} |0\rangle \langle i| \quad (\text{YT})$$

Furthermore, some additional notations are presented in Appendix A.1.

We refer to Appendix A.2 for the rules of the calculus.

3 Infinite ZW calculus

We introduce the graphical language \mathbf{ZW}_∞ for reasoning about linear operators on the Fock space. By setting $d = \infty$, we generalise the qudit interpretation of each ZXW generator to obtain novel maps on the Fock space. Combining this with the QPath calculus, we obtain a (partial) axiomatisation of linear operators in $\mathbf{Vect}_{\mathbb{N}}$.

3.1 Generators

The *Z spider* may be interpreted as the following operator in $\mathbf{Vect}_{\mathbb{N}}$:

$$n \left\{ \begin{array}{c} \vdots \\ \vdots \end{array} \right\} \xrightarrow{[\cdot]} \sum_{i=0}^{\infty} a_i |i\rangle^{\otimes m} \langle i|^{\otimes n}, \quad \text{where } n > 0, \vec{a} = (a_1, \dots, a_k, \dots), a_0 := 1. \quad (4)$$

Note that we require the number of inputs to be greater than 0. In fact, the unit Z spider $\sum_{n=0}^{\infty} |n\rangle$ does not have finite support and is thus not allowed in \mathbf{ZW}_{∞} . In particular, we do not have cups because these are defined in terms of unit Z spiders. However, we do have caps and counit Z spiders since these map finite states to finite states. Moreover, Z boxes where $n = 0$ and \vec{b} has a *finite support* are allowed as generators, that is, for some finite $N \in \mathbb{N}$:

$$\left\{ \begin{array}{c} \vdots \\ \vdots \end{array} \right\} \xrightarrow{[\cdot]} \sum_{i=0}^N a_i |i\rangle^{\otimes m}, \quad \text{where } n > 0, \vec{a} = (a_1, \dots, a_N, 0, \dots), a_0 := 1. \quad (6)$$

Note that QPath endomorphisms are Z boxes of the form:

$$\text{---} \boxed{r} \text{---} = \text{---} \boxed{r^{\vec{N}}} \text{---}$$

where $r^{\vec{N}} = (r^1, r^2, \dots, r^k, \dots)$. The *W node* is straightforwardly generalised to infinite dimensions:

$$\left\{ \begin{array}{c} \vdots \\ \vdots \end{array} \right\} \xrightarrow{[\cdot]} |0 \dots 0\rangle \langle 0| + \sum_{i=1}^{\infty} (|i0 \dots 0\rangle + \dots + |0 \dots 0i\rangle) \langle i|$$

We also take the dagger of W as a generator. One may check that both maps preserve finite states. Let us now consider the *X spider* from the ZXW calculus. Its action on the basis in \mathbf{ZXW}_d is given by addition modulo d . We can consider addition in \mathbb{N} as a natural generalisation for the X spider in the Fock space. However, this does not yield a group structure on \mathbb{N} and the Frobenius law fails in infinite dimensions. Moreover, we already have a node that performs addition of natural numbers, the *split map*:

$$\left\{ \begin{array}{c} \vdots \\ \vdots \end{array} \right\} \xrightarrow{[\cdot]} \sum_{n=0}^{\infty} \sum_{k=0}^n \binom{n}{k}^{\frac{1}{2}} |k, n-k\rangle \langle n|$$

We also take *merge* as a generator of \mathbf{ZW}_{∞} . Instead of a Frobenius structure we get a bialgebra between merge and split. The binomial coefficients in fact ensure that the bialgebra law holds. However, this also means that the interaction of the split map with Z spiders is a bialgebra only up to coefficients. We also take the n -particle creation $\bullet \xrightarrow{n}$ and annihilation $\xrightarrow{n} \bullet$ as generators of \mathbf{ZW}_{∞} .

3.2 Axioms

The axioms of \mathbf{ZW}_{∞} are obtained by merging the rules of QPath and those of ZXW governing the Z and W nodes. We only give a partial axiomatisation with the rules that we use in Section 5. The Z spider satisfies the axioms of a non-unital special Frobenius algebra, an algebraic structure which characterises bases in infinite dimensional vector spaces [2]. The rules of the W algebra as well as Axioms (Bsj_∞) and (K0_∞) are directly lifted from qudit ZXW. Bosonic nodes form a bialgebra with a semiring of endomorphisms. All the rules of QPath from [21] hold in \mathbf{ZW}_{∞} , here we only give the ones we use. We moreover give a set of interaction rules relating Z, W and bosonic nodes. Rule (W1_∞) shows the behaviour of the W algebra. (bW1) relates the split

map and W states and replaces the “branching” rule of QPath. The bialgebra law between bosonic nodes and Z spiders only holds up to factorial coefficients (bZBA). It holds on the nose if the binomial coefficients are removed from the definition of bosonic nodes. However then, the bialgebra law between split and merge would fail. We also have a new version of the ZXW trialgebra law (bTA) with bosonic nodes replacing X spiders, proved in Proposition 4.4.

Non-unital Frobenius algebra

$$\text{Diagram 1} = \text{Diagram 2} = \text{Diagram 3} \quad (S1_\infty)$$

Rules generalised from qudit ZW

$$\begin{aligned} \text{Diagram 1} &= \text{Diagram 2} & (BZW_\infty) & \quad \text{Diagram 3} = \text{Diagram 4} & (Sym_\infty) \\ \text{Diagram 5} &= \text{Diagram 6} & (Pcy_\infty) & \quad \text{Diagram 7} = \text{Diagram 8} & (Aso_\infty) \\ \text{Diagram 9} &= \text{Diagram 10} & (AD_\infty) & \quad \text{Diagram 11} = \text{Diagram 12} & (WW_\infty) \end{aligned}$$

Rules from QPath

$$\begin{aligned} \text{Diagram 1} &= \text{Diagram 2} & (bSym) & \quad \text{Diagram 3} = \text{Diagram 4} & (bBA) \\ \text{Diagram 5} &= \text{Diagram 6} & (bAso) & \quad \text{Diagram 7} = \text{Diagram 8} & (bId) \end{aligned}$$

Interaction rules

$$\begin{aligned} \text{Diagram 1} &= \text{Diagram 2} & (bZBA) & \quad \text{Diagram 3} = \text{Diagram 4} & (K0_\infty) \\ \text{Diagram 5} &= \text{Diagram 6} & (bTA) & \quad \text{Diagram 7} = \text{Diagram 8} & (W1_\infty) \\ \text{Diagram 9} &= \text{Diagram 10} & (bW1) & \quad \text{Diagram 11} = \text{Diagram 12} & (Bsj_\infty) \end{aligned}$$

where $e_n = (\underbrace{0, \dots, 1, 0, \dots}_n)$

4 Photonics in ZXW

In this section, we give a method for proving novel equations in \mathbf{ZW}_∞ by rewriting in the ZXW calculus. We give a truncated interpretation $\mathcal{T}_d : \mathbf{ZW}_\infty \rightarrow \mathbf{ZXW}_d$ parametrised by the qudit dimension d . We prove the *lifting* theorem: any equality in the Fock space of the form $\llbracket D \rrbracket = \llbracket D' \rrbracket$ can be proved by showing that there exists some $N \in \mathbb{N}$ such that $\mathcal{T}_d(D)P_N = \mathcal{T}_d(D')P_N$ for any $d > N$, where P_N denotes the projector on the $(d - N)$ -particle sector of the input qudits. We use this theorem to give diagrammatic proofs of soundness for the axioms of the infinite ZW calculus.

4.1 Truncation

Truncation consists in giving a finite dimensional description of operators on an infinite-dimensional space. We define this as a mapping $\mathcal{T}_d : \mathbf{ZW}_\infty \rightarrow \mathbf{ZXW}_d$. We introduce a useful component in \mathbf{ZXW}_d representing the projector on the d -particle sector of a pair of qudits:

$$\begin{array}{c} \bullet \\ | \\ \bullet \end{array} := \begin{array}{c} \xrightarrow{\vec{N}} \\ \boxed{-\frac{N}{d}} \\ \xrightarrow{\vec{1}} \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\llbracket \cdot \rrbracket_d} \sum_{\substack{a_1, a_2=0 \\ a_1+a_2 < d}}^{d-1} |a_1, a_2\rangle \langle a_1, a_2| \quad (25)$$

where $\vec{N} = (1, 2, \dots, d-1)$, $\vec{1} = (-1, \dots, -1)$, and each operation is defined elementwise. We can use this to represent bosonic nodes in ZXW. In fact, the X spider performs addition of basis states modulo d , and this agrees with standard addition whenever the input is in the d -particle sector. We can thus model the truncated split map as an X spider with a projector and Z boxes for the binomial coefficients.

The d -truncation $\mathcal{T}_d : \mathbf{ZW}_\infty \rightarrow \mathbf{ZXW}_d$ is defined on generators as follows:

1. Z spiders and W nodes are mapped to their qudit versions in \mathbf{ZXW}_d .
2. The split map has the following truncation:

$$\begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\mathcal{T}_d} \begin{array}{c} \boxed{\frac{1}{\sqrt{N!}}} \\ \bullet \\ \boxed{\frac{1}{\sqrt{N!}}} \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\llbracket \cdot \rrbracket_d} \sum_{n=0}^{d-1} \sum_{k=0}^n \binom{n}{k}^{\frac{1}{2}} |k, n-k\rangle \langle n| \quad (26)$$

3. The merge map has the following truncation:

$$\begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\mathcal{T}_d} \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \boxed{\frac{1}{\sqrt{N!}}} \\ \bullet \\ \boxed{\frac{1}{\sqrt{N!}}} \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\llbracket \cdot \rrbracket_d} \sum_{n, m=0, n+m < d}^{d-1} \binom{n+m}{n}^{\frac{1}{2}} |n+m\rangle \langle n, m| \quad (27)$$

4. Particle creations and annihilations correspond to unit X spiders.

$$\begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\mathcal{T}_d} \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} = \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} \xrightarrow{\mathcal{T}_d} \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} = \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} \quad (28)$$

This mapping respects all the axioms of the infinite ZW calculus, except for the bialgebra law between split and merge maps (bBA). This rule only holds in \mathbf{ZXW}_d up to a projector which ensures the total number of particles does not exceed d . That is, in the truncated interpretation we have:

$$\begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array} = \begin{array}{c} \bullet \\ \bullet \end{array} \begin{array}{c} \bullet \\ \bullet \end{array}$$

Starting from the projector on 2 modes, we may construct the d -particle projector on m modes recursively:

$$m \left\{ \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \right\} = m-1 \left\{ \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \right\} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array}$$

One may show that the following rewrites are valid in \mathbf{ZXW}_d for any d .

$$\begin{array}{ccc}
 \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} = \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} & \text{(PP)} & \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} = \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} & \text{(PS)} \\
 \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} = \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} & \text{(PZ)} & \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} = \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} & \text{(PK)} \\
 \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} = \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} & \text{(PW)} & \text{where } \vec{1}_j = \underbrace{(1, \dots, 1, 0, \dots, 0)}_{d-j}
 \end{array}$$

In fact projectors commute with any map that preserves the number of photons. Using particle creations and annihilations we may also construct the projector on the n -particle sector for $n < d$ as follows:

$$\begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} := \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \begin{array}{c} \circ \\ \vdots \\ \circ \end{array} \quad (34)$$

4.2 Lifting

We now prove our main result: equalities in the infinite dimensional calculus may be derived by rewriting in the truncated interpretation. In order to relate the infinite and truncated interpretations, we use two ingredients. First, the embedding $\mathcal{E} : \mathbf{Vect}_d \rightarrow \mathbf{Vect}_{\mathbb{N}}$ which views any linear map between qudits as a map on the Fock space, acting on the first d dimensions of each mode. Formally, we have $\mathcal{E}(f) |x_1, \dots, x_m\rangle = f |x_1, \dots, x_m\rangle$ whenever $x_i < d$ for $i = 1, \dots, m$ and $\mathcal{E}(f) |x_1, \dots, x_m\rangle = 0$ otherwise, for any linear map f on m qudits. It is easy to show that this embedding is a faithful monoidal functor. Second, we use the projector on the n -particle sector of the Fock space, i.e. the linear operator $P_n : \llbracket m \rrbracket \rightarrow \llbracket m \rrbracket$ such that $P_n |x_1, \dots, x_m\rangle = |x_1, \dots, x_m\rangle$ when $\sum_{i=1}^m x_i < n$ and $P_n |x_1, \dots, x_m\rangle = 0$ otherwise. The following lemma characterises the infinite interpretation of a diagram in \mathbf{ZW}_{∞} in terms of the truncations.

Lemma 4.1. *For any diagram $D \in \mathbf{ZW}_{\infty}$ and $n \in \mathbb{N}$ there is $d^* \in \mathbb{N}$ such that, whenever $d > d^*$, we have:*

$$\llbracket D \rrbracket P_n = \mathcal{E}(\llbracket \mathcal{T}_d(D) \rrbracket_d) P_n$$

Proof. To prove the statement it is sufficient that, given D and n , we can find a “big enough” dimension d^* , such that $\llbracket D \rrbracket |x_1, \dots, x_m\rangle = \mathcal{E}(\llbracket \mathcal{T}^*(D) \rrbracket_{d^*}) |x_1, \dots, x_m\rangle$ whenever $\sum_{i=1}^m x_i < n$. We prove this by induction over the recursive definition of string diagrams. Indeed, a diagram in \mathbf{ZW}_{∞} may be viewed as a sequence of layers where, in each layer, a single generator appears tensored by identities. Therefore, it is sufficient to find d^* given an n for each generating layer.

Note first that W nodes preserve the total number of photons, similarly for bosonic nodes, as well as for Z spiders $1 \rightarrow 1$ and identities. Thus, for layers involving these generators, we can set $d^* = n$. The Z spiders with $a > 1$ output wires (4) can at most multiply the total number of photons by a , therefore, we can set $d^* = a \cdot n$. Finite sequences, i.e. Z spider states (6) only contribute a finite number of photons t , so we can set $d^* = n + t$ in this case. Finally, in case of a photon preparation we can set $d^* = n + 1$. Since the dimension remains finite after each generator, by induction, we can find a finite d^* for any D and n . \square

Note that there are potentially smaller values of d^* for which the statement above holds, as well as intermediate dimensions that we can assign to the internal wires of D . Finding optimal values for these dimensions would allow more efficient classical simulation of photonic circuits by tensor network contraction.

Theorem 4.2 (Lifting). *For any $D, D' : m \rightarrow m' \in \mathbf{Z}\mathbf{W}_\infty$ the following are equivalent:*

1. In $\mathbf{Vect}_\mathbb{N}$:

$$\llbracket D \rrbracket = \llbracket D' \rrbracket$$

2. For any $n \in \mathbb{N}$ there is a dimension d^* such that, for all $d > d^*$, in \mathbf{ZXW}_d :

$$\mathcal{T}_d(D) = \mathcal{T}_d(D')$$

Proof. First note that $\llbracket D \rrbracket = \llbracket D' \rrbracket$ if and only if $\llbracket D \rrbracket P_n = \llbracket D' \rrbracket P_n$ for any $n \in \mathbb{N}$. By Lemma 4.1, this is equivalent to: for any n there is a d^* such that for all $d > d^*$ we have $\mathcal{E}(\llbracket \mathcal{T}_d(D) \rrbracket_d) P_n = \mathcal{E}(\llbracket \mathcal{T}_d(D') \rrbracket_d) P_n$. Denoting by \tilde{P}_n the projector given in Equation (34), we have that $\mathcal{E}(\llbracket \mathcal{T}_d(D) \tilde{P}_n \rrbracket_d) = \mathcal{E}(\llbracket \mathcal{T}_d(D') \tilde{P}_n \rrbracket_d)$. Finally, using faithfulness of the embedding \mathcal{E} and completeness of the ZXW calculus, we obtain $\mathcal{T}_d(D) \tilde{P}_n = \mathcal{T}_d(D') \tilde{P}_n$. \square

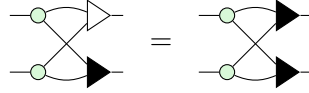
This means in particular that if $\mathcal{T}_d(D) = \mathcal{T}_d(D')$ for any $d \in \mathbb{N}$, then $\llbracket D \rrbracket = \llbracket D' \rrbracket$. However, the theorem is stronger than this. To prove $\llbracket D \rrbracket = \llbracket D' \rrbracket$, it is sufficient to show that there exists $N \in \mathbb{N}$ such that for any $d > N$ we have:

$$\mathcal{T}_d(D) = \mathcal{T}_d(D')$$

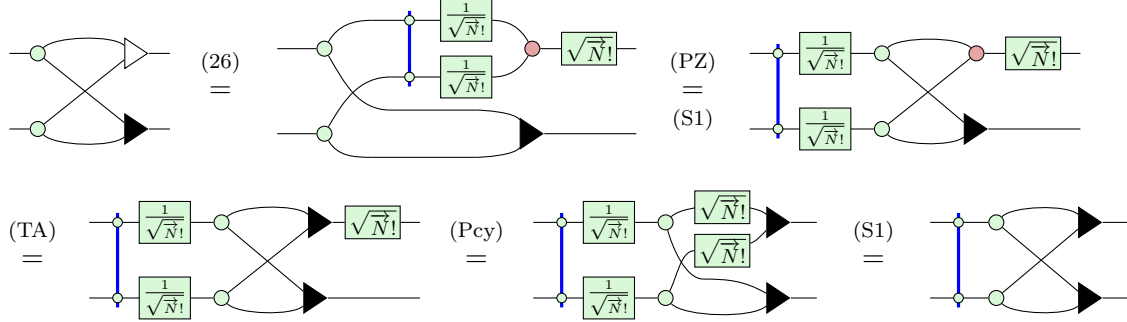
This is proved by setting $d^* = n + N$ given any n , and is still a weaker statement as the difference between d^* and n may not be constant. Most of the rules of ZXW are independent of the dimension d , making it easier to rewrite while preserving this condition.

Remark 4.3. *We say that an equation $D = D'$ holds in $\mathbf{Z}\mathbf{W}_\infty$ whenever $\llbracket D \rrbracket = \llbracket D' \rrbracket$ with the standard interpretation $\llbracket \cdot \rrbracket : \mathbf{Z}\mathbf{W}_\infty \rightarrow \mathbf{Vect}_\mathbb{N}$. We say that an equation holds in \mathbf{ZXW}_d if it holds in the rewriting system given in Section 2.3, and we indicate the rules used. Whenever the split or merge map appears in a statement about \mathbf{ZXW}_d , we use it as syntactic sugar for its truncation as given in Equations (26) and (27).*

Proposition 4.4. *In \mathbf{ZW}_∞ , the following holds:*



Proof. First, in \mathbf{ZXW}_d the following holds:



Precomposing the equation above by projectors gives an equation of the form $\mathcal{T}_d(D)\tilde{P}_d = \mathcal{T}_d(D')\tilde{P}_d$ which holds for any d . Therefore, by Theorem 4.2 we deduce that $\llbracket D \rrbracket = \llbracket D' \rrbracket$. \square

We may apply the same reasoning to the remaining axioms of \mathbf{ZW}_∞ .

Theorem 4.5 (Soundness). *The axioms of the infinite ZW calculus are sound for the infinite interpretation $\llbracket \cdot \rrbracket : \mathbf{ZW}_\infty \rightarrow \mathbf{Vect}_\mathbb{N}$.*

Proof. The axioms for Z spiders and W nodes, as well as (Bsj_∞) and (K0_∞) , are trivially lifted from ZXW since they all lie in the image of the truncation functor and $\mathcal{T}_d(D) = \mathcal{T}_d(D') \forall d \in \mathbb{N} \implies \llbracket D \rrbracket = \llbracket D' \rrbracket$. The rules from QPath can be proved with few ZXW rewrites by using the properties of projectors. The bialgebra axiom for bosonic nodes (bBA) is the hardest one to prove diagrammatically. The proof, obtained by checking on the basis elements, boils down to an application of the Vandermonde identity, as shown explicitly in [34, Section 5.3]. Proving Rule (bW1) diagrammatically requires the definition of projector (25); however, it is easier to verify by computing the interpretations. Similarly, Rule (W1_∞) is proved by considering the interpretation. Rule (bTA) is proved in Proposition 4.4. Rule (bZBA) is proved similarly following from the fact that bialgebra between Z and X holds in the truncation. \square

Even though dualisers and hadamard nodes are not valid maps in the infinite calculus, we may still use them when rewriting in the truncation, as long as they disappear in the resulting diagram. An example is given by Lemma B.8, which is used to prove the commutation relations for bosonic and fermionic operators in the next section.

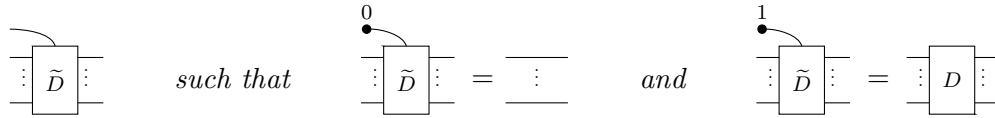
5 Rewriting Hamiltonians

In this section, we use rewriting in ZXW to prove facts about quantum optical Hamiltonians. These are defined as sums and products of creation and annihilation operators, and may always be represented as single ZXW diagrams. We give a range of example applications, from linear optics and non-linear Kerr media to light-matter interaction. We provide the proof of some propositions and lemmas of this section in the appendix.

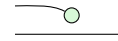
5.1 Sums and products of \mathbf{ZW}_∞ diagrams

Controlled diagrams in \mathbf{ZW}_∞ are defined as follows.

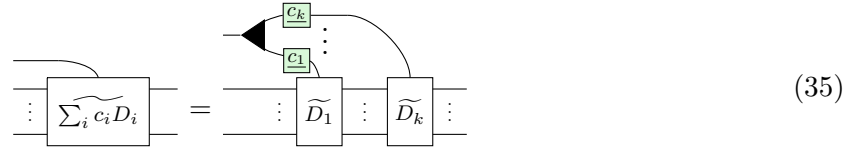
Definition 5.1 (Controlled diagram). *For a diagram D , a controlled diagram \widetilde{D} is*



As an example, a controlled diagram of identity is given by:

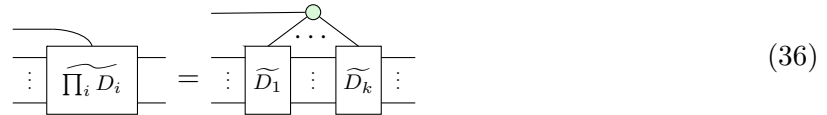


Proposition 5.2 (Controlled sum of diagrams). *Given controlled diagrams $\widetilde{D}_1, \dots, \widetilde{D}_k$ corresponding to diagrams D_1, \dots, D_k and complex numbers c_1, \dots, c_k , a controlled diagram for $\sum_i c_i D_i$ is given by*



where $\underline{c}_i = (c_i, 0, \dots, 0)$.

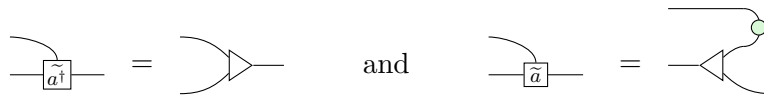
Proposition 5.3 (Controlled product of diagrams). *Given controlled diagrams $\widetilde{D}_1, \dots, \widetilde{D}_k$ corresponding to the diagrams D_1, \dots, D_k , a controlled diagram for $\prod_i D_i$ is given by*



The above propositions can be verified by plugging in $|0\rangle$ and $|1\rangle$. This allows us to write sums and products of diagrams by creating controlled sums and products, and then plugging $|1\rangle$ on the top.

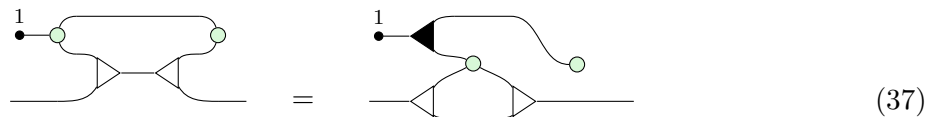
5.2 Creation and annihilation operators

Hamiltonians are usually written in terms of sums and products of creation/annihilation operators. In order to obtain them as diagrams in \mathbf{ZXW} , we need to build the controlled operators. Controlled diagrams for bosonic creation (a^\dagger) and annihilation (a) operators are the following:



One may check that plugging $|0\rangle$ in the control gives the identity in both cases. We can now give a diagrammatic proof of the commutation relations by rewriting in \mathbf{ZXW} .

Proposition 5.4. *In \mathbf{ZW}_∞ , $aa^\dagger = a^\dagger a + id$, that is:*



5.3 Linear optics

The circuits in linear optics are built using two main gates: (1) phase shift and (2) beam splitter. These are often defined by their Hamiltonians. The Hamiltonian of the phase shift is given by

the number operator $H_P = \alpha \hat{n}_1 = \alpha a_1^\dagger a_1$. The phase shift gate is the exponential of this Hamiltonian, i.e. e^{iH_P} .

Proposition 5.5. *In $\mathbf{Z}\mathbf{W}_\infty$, the phase shift gate with phase α is given by:*

$$\exp \left(\begin{array}{c} 1 \\ \bullet \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array} \right) = \text{---} \boxed{e^{i\tilde{N}\alpha}} \text{---}$$

The Hamiltonian of the beam splitter is given as $H_{BS} = \theta \left(e^{i\phi} a_1 a_2^\dagger + e^{-i\phi} a_1^\dagger a_2 \right)$ for some parameters ϕ and θ . The exponential is given by:

$$\exp \left(\begin{array}{c} 1 \\ \bullet \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array} \right) = \begin{array}{c} \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array}$$

where $r = e^{i\phi} \sin \theta$ is the reflectivity, $t = \cos \theta$ is the transmissivity, and $*$ is complex conjugation.

5.4 Non-linear optics with Kerr media

Kerr media are non-linear optical crystals which allow performing entangling operations between bosonic modes, with applications to photonic quantum computing [3]. Here we study the single mode Kerr effect, described by a phase shift with a quadratic term, and the cross-Kerr interaction. We give a representation of the latter as a *phase gadget* which allows for simple rewrite rules that remove non-linearities from circuits.

The Kerr interaction is given by the Hamiltonian $H_K = \kappa \hat{n}_1^2 = \kappa a_1^\dagger a_1 a_1^\dagger a_1$. We can represent this Hamiltonian in ZXW and diagonalise it to get:

Proposition 5.6. *In $\mathbf{Z}\mathbf{W}_\infty$, the Kerr gate with parameter κ is given by:*

$$\exp \left(\begin{array}{c} 1 \\ \bullet \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array} \right) = \text{---} \boxed{e^{i\kappa \tilde{N}^2}} \text{---}$$

The cross-Kerr interaction is given by the Hamiltonian $H_{CK} = \tau \hat{n}_1 \hat{n}_2 = \tau a_1^\dagger a_1 a_2^\dagger a_2$, from which we can derive the cross-Kerr gate $CK(\tau) = \exp(i\tau \hat{n}_1 \hat{n}_2)$. The cross-Kerr gate with $\tau = \pi$ may be used to construct the CZ gate on dual-rail qubits [3]. We find that it has a natural representation in the calculus.

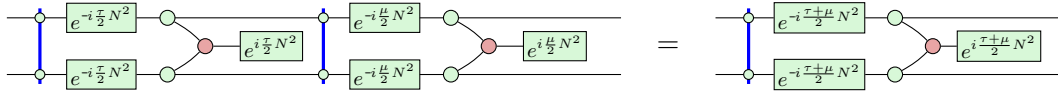
Proposition 5.7. *The cross-Kerr gate with parameter κ is given by:*

$$\exp \left(\begin{array}{c} \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array} \right) = \begin{array}{c} \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array}$$

Utilizing the rewriting rules of our calculi, we can demonstrate different properties of these operators. For instance, we can now prove the following proposition related to the composition of two cross-Kerr operators:

Proposition 5.8. *The composition of two cross-Kerr media with parameters τ and μ results in a cross-Kerr interaction with parameter $\tau + \mu$. Specifically, in $\mathbf{Z}\mathbf{X}\mathbf{W}_d$ the following equation*

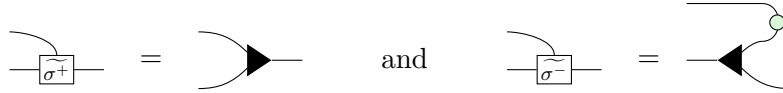
holds:



5.5 Towards light-matter interaction

So far, we have studied only bosonic creation and annihilation operators. We now introduce *fermionic* creation and annihilation. These operators are easily represented in \mathbf{ZW}_∞ using the W node. The W algebra was indeed already shown to have an important role in fermionic quantum computing [41]. We show the anti-commutation relation for these operators and finally, we represent the Jaynes-Cummings Hamiltonian describing the interaction of bosons and fermions.

Controlled diagrams for the fermionic creation (σ^+) and annihilation (σ^-) operators are given by



Note that the fermionic creation and annihilation operators are only defined on the qubit subspace due to the Pauli exclusion principle. These operators acting on the input $|n\rangle$, for $n > 2$, result in a zero diagram.

Now, we use the rules of the infinite ZW calculus to prove the anti-commutation relations.

Proposition 5.9. *In \mathbf{ZW}_∞ , $\sigma^- \sigma^+ + \sigma^+ \sigma^- = id$, that is:*

(38)

where the right-hand side is the identity on the qubit subspace.

The Jaynes-Cummings model describes the interaction between a bosonic mode and a 2-level atom. It is defined by the following Hamiltonian, for some frequency ω :

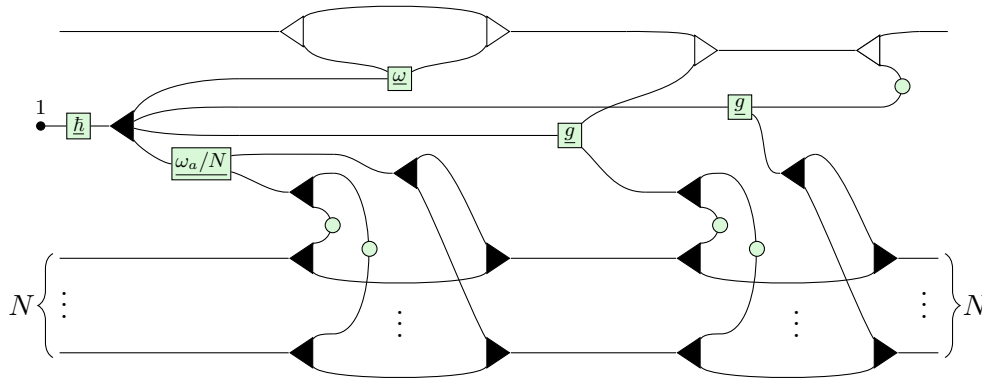
$$H_{JC} = \hbar\omega \left(a_1 \sigma_2^+ + a_1^\dagger \sigma_2^- \right)$$
(39)

The Tavis-Cummings Hamiltonian is a natural generalization of Jaynes-Cummings to a setting in which multiple atoms interact with a bosonic mode:

$$H_{TC} = \hbar \left[\omega_c a_1^\dagger a_1 + \frac{\omega_a}{N} \left(\sum_{n=1}^N \sigma_n^+ \right) \left(\sum_{n=1}^N \sigma_n^- \right) + g a_1^\dagger \sum_{n=1}^N \sigma_n^- + g a_1 \sum_{n=1}^N \sigma_n^+ \right]$$

where N is the number of atoms, ω_a and ω_c are the atomic and cavity resonance frequency, and

g is the atom-photon coupling strength.



Acknowledgements

We are grateful to Amar Hadzihasanovic as well as the anonymous QPL reviewers for their detailed feedback and their numerous suggestions for improvement. We would also like to thank Richie Yeung and Alexis Toumi for insightful discussions related to this paper. RS is supported by the Clarendon Fund Scholarship. LY is supported by the Basil Reeve Graduate Scholarship at Oriel College with the Clarendon Fund.

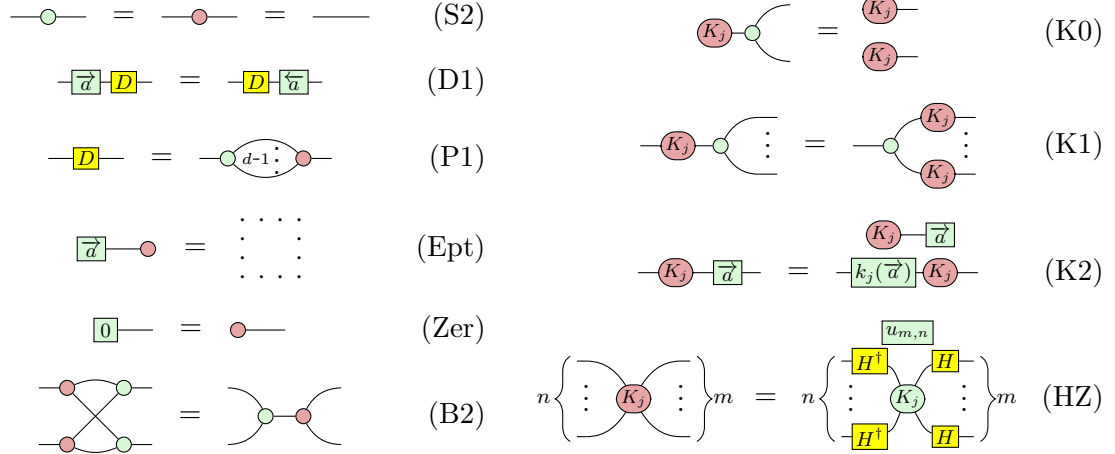
References

- [1] Scott Aaronson & Alex Arkhipov (2011): *The Computational Complexity of Linear Optics*. In: *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing, STOC '11*, Association for Computing Machinery, New York, NY, USA, pp. 333–342, doi:10.1145/1993636.1993682.
- [2] Samson Abramsky & Chris Heunen (2012): *H*-Algebras and Nonunital Frobenius Algebras: First Steps in Infinite Dimensional Categorical Quantum Mechanics*. *AMS Proceedings of Symposia in Applied Mathematics: Mathematical Foundations of Information Flow*, pp. 14–37. arXiv:1011.6123.
- [3] Hiroo Azuma (2007): *Quantum Computation with Kerr-nonlinear Photonic Crystals*. *Journal of Physics D: Applied Physics* 41(2), p. 025102, doi:10.1088/0022-3727/41/2/025102. arXiv:quant-ph/0604086.
- [4] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and Back Again: A Circuit Extraction Tale*. *Quantum* 5, p. 421, doi:10.22331/q-2021-03-25-421.
- [5] Héctor Bombín, Chris Dawson, Ryan V. Mishmash, Naomi Nickerson, Fernando Pastawski & Sam Roberts (2023): *Logical Blocks for Fault-Tolerant Topological Quantum Computation*. *PRX Quantum* 4(2), p. 020303, doi:10.1103/PRXQuantum.4.020303.
- [6] Filippo Bonchi, Paweł Sobociński & Fabio Zanasi (2017): *Interacting Hopf Algebras*. *Journal of Pure and Applied Algebra* 221(1), pp. 144–184, doi:10.1016/j.jpaa.2016.06.002. arXiv:1403.7048.
- [7] Robert I. Booth & Titouan Carette (2022): *Complete ZX-Calculi for the Stabiliser Fragment in Odd Prime Dimensions*. In Stefan Szeider, Robert Ganian & Alexandra Silva, editors: *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 241, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 24:1–24:15, doi:10.4230/LIPIcs.MFCS.2022.24.

- [8] Giuseppe Calajó, Philipp K. Jenke, Lee A. Rozema, Philip Walther, Darrick E. Chang & Joel D. Cox (2023): *Nonlinear Quantum Logic with Colliding Graphene Plasmons*. *Physical Review Research* 5(1), p. 013188, doi:10.1103/PhysRevResearch.5.013188.
- [9] Titouan Carette, Dominic Horsman & Simon Perdrix (2019): *SZX-Calculus: Scalable Graphical Quantum Reasoning*. In Peter Rossmanith, Pinar Heggenes & Joost-Pieter Katoen, editors: *44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019), Leibniz International Proceedings in Informatics (LIPIcs)* 138, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 55:1–55:15, doi:10.4230/LIPIcs.MFCS.2019.55.
- [10] William R. Clements, Peter C. Humphreys, Benjamin J. Metcalf, W. Steven Kolthammer & Ian A. Walmsley (2016): *Optimal Design for Universal Multiport Interferometers*. *Optica* 3(12), pp. 1460–1465, doi:10.1364/OPTICA.3.001460.
- [11] Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix & Benoît Valiron (2022): *LO_v-Calculus: A Graphical Language for Linear Optical Quantum Circuits*. In Stefan Szeider, Robert Ganian & Alexandra Silva, editors: *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022), Leibniz International Proceedings in Informatics (LIPIcs)* 241, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, p. 35:1–35:16, doi:10.4230/LIPIcs.MFCS.2022.35.
- [12] Bob Coecke (2023): *Basic ZX-calculus for Students and Professionals*. arXiv:2303.03163.
- [13] Bob Coecke & Ross Duncan (2007): *Interacting Quantum Observables*. Available at www.cs.ox.ac.uk/people/bob.coecke/GreenRed.pdf.
- [14] Bob Coecke & Ross Duncan (2008): *Interacting Quantum Observables*. In: *Proceedings of the 35th International Colloquium on Automata, Languages and Programming, Part II, ICALP '08*, Springer-Verlag, Berlin, Heidelberg, pp. 298–310, doi:10.1007/978-3-540-70583-3.25.
- [15] Bob Coecke & Ross Duncan (2011): *Interacting Quantum Observables: Categorical Algebra and Diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016.
- [16] Bob Coecke & Stefano Gogioso (2022): *Quantum in Pictures*. Quantinuum.
- [17] Bob Coecke & Aleks Kissinger (2010): *The Compositional Structure of Multipartite Quantum Entanglement*. In: *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming: Part II, ICALP'10*, Springer-Verlag, Berlin, Heidelberg, pp. 297–308, doi:10.1007/978-3-642-14162-1_25. arXiv:1002.2540.
- [18] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes*. Cambridge University Press, doi:10.1017/9781316219317.
- [19] Bob Coecke, Aleks Kissinger, Alex Merry & Shibdas Roy (2011): *The GHZ/W-calculus Contains Rational Arithmetic*. In Farid Ablayev, Bob Coecke & Alexander Vasiliev, editors: *Proceedings CSR 2010 Workshop on High Productivity Computations, Electronic Proceedings in Theoretical Computer Science* 52, Open Publishing Association, Kazan, Russia, pp. 34–48, doi:10.4204/EPTCS.52.4.
- [20] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons & Seyon Sivarajah (2019): *Phase Gadget Synthesis for Shallow Circuits: 16th International Conference on Quantum Physics and Logic 2019*. In Bob Coecke & Matthew Leifer, editors: *Proceedings 16th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 318, Open Publishing Association, Waterloo, Australia, pp. 213–228, doi:10.4204/EPTCS.318.13.
- [21] Giovanni de Felice & Bob Coecke (2022): *Quantum Linear Optics via String Diagrams*. arXiv:2204.12985.
- [22] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.
- [23] Ross Duncan & Simon Perdrix (2010): *Rewriting Measurement-Based Quantum Computations with Generalised Flow*. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf

- der Heide & Paul G. Spirakis, editors: *Automata, Languages and Programming*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 285–296, doi:10.1007/978-3-642-14162-1_24. Available at <http://personal.strath.ac.uk/ross.duncan/papers/gflow.pdf>.
- [24] W. Dür, G. Vidal & J. I. Cirac (2000): *Three Qubits Can Be Entangled in Two Inequivalent Ways*. *Physical Review A* 62(6), p. 062314, doi:10.1103/PhysRevA.62.062314.
- [25] Héctor Figueroa & José M. Gracia-Bondía (2005): *Combinatorial Hopf Algebras in Quantum Field Theory i. Reviews in Mathematical Physics* 17(08), pp. 881–976, doi:10.1142/S0129055X05002467. arXiv:hep-th/0408145.
- [26] Marcelo Fiore (2015): *An Axiomatics and a Combinatorial Model of Creation/Annihilation Operators*. arXiv:1506.06402.
- [27] Nathan Fitzpatrick, Harriet Apel & David Muñoz Ramo (2021): *Evaluating Low-Depth Quantum Algorithms for Time Evolution on Fermion-Boson Systems*. arXiv:2106.03985.
- [28] Kazuyuki Fujii (2002): *Introduction to Coherent States and Quantum Information Theory*. arXiv:quant-ph/0112090.
- [29] Stefano Gogioso & Fabrizio Genovese (2016): *Infinite-Dimensional Categorical Quantum Mechanics*. In Ross Duncan & Chris Heunen, editors: *Proceedings 13th International Conference on Quantum Physics and Logic, EPTCS 236*, Glasgow, Scotland, pp. 51–69, doi:10.4204/EPTCS.236.4.
- [30] Stefano Gogioso & Fabrizio Genovese (2018): *Towards Quantum Field Theory in Categorical Quantum Mechanics*. In Bob Coecke & Aleks Kissinger, editors: *Proceedings 14th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science 266*, Open Publishing Association, Nijmegen, The Netherlands, pp. 349–366, doi:10.4204/EPTCS.266.22.
- [31] Stefano Gogioso & Fabrizio Genovese (2019): *Quantum Field Theory in Categorical Quantum Mechanics*. In Peter Selinger & Giulio Chiribella, editors: *Proceedings of the 15th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science 287*, Open Publishing Association, Halifax, Canada, pp. 163–177, doi:10.4204/EPTCS.287.9.
- [32] Xiang Guo, Chang-ling Zou, Carsten Schuck, Hojoong Jung, Risheng Cheng & Hong X. Tang (2017): *Parametric Down-Conversion Photon-Pair Source on a Nanophotonic Chip*. *Light: Science & Applications* 6(5), pp. e16249–e16249, doi:10.1038/lsa.2016.249.
- [33] Amar Hadzihasanovic (2015): *A Diagrammatic Axiomatisation for Qubit Entanglement*. In: *Proceedings of the 2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '15*, IEEE Computer Society, USA, pp. 573–584, doi:10.1109/LICS.2015.59. arXiv:1501.07082.
- [34] Amar Hadzihasanovic (2017): *The Algebra of Entanglement and the Geometry of Composition*. Ph.D. thesis, University of Oxford. arXiv:1709.08086.
- [35] Amar Hadzihasanovic, Kang Feng Ng & Quanlong Wang (2018): *Two Complete Axiomatisations of Pure-State Qubit Quantum Computing*. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '18*, Association for Computing Machinery, New York, NY, USA, pp. 502–511, doi:10.1145/3209108.3209128.
- [36] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics*. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '18*, Association for Computing Machinery, New York, NY, USA, pp. 569–578, doi:10.1145/3209108.3209139. arXiv:1801.10142.
- [37] Aleks Kissinger (2022): *Phase-free ZX diagrams are CSS codes (...or how to graphically grok the surface code)*. arXiv:2204.14038.
- [38] Aleks Kissinger & John van de Wetering (2020): *Reducing the Number of Non-Clifford Gates in Quantum Circuits*. *Physical Review A* 102(2), p. 022406, doi:10.1103/PhysRevA.102.022406. arXiv:1903.10477.

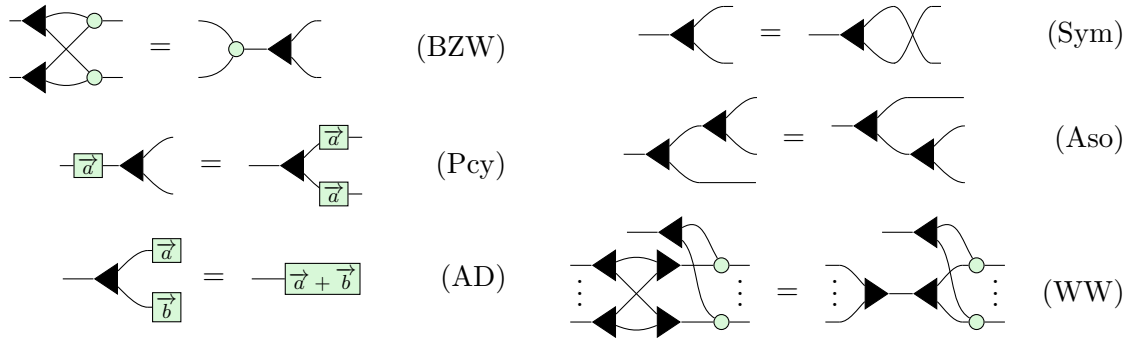
- [39] Aleks Kissinger, John van de Wetering & Renaud Vilmart (2022): *Classical Simulation of Quantum Circuits with Partial and Graphical Stabiliser Decompositions*. In François Le Gall & Tomoyuki Morimae, editors: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 232, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 5:1–5:13, doi:10.4230/LIPIcs.TQC.2022.5.
- [40] Ronald Mullin & Gian-Carlo Rota (1970): *On the Foundations of Combinatorial Theory. III. Theory of Binomial Enumeration*. Institute of Statistics Mimeo Series 600.24, North Carolina State University. Dept. of Statistics. Available at <http://www.lib.ncsu.edu/resolver/1840.4/2578>.
- [41] Kang Feng Ng, Amar Hadzihasanovic & Giovanni de Felice (2019): *A Diagrammatic Calculus of Fermionic Quantum Circuits*. *Logical Methods in Computer Science* Volume 15, Issue 3, doi:10.23638/LMCS-15(3:26)2019.
- [42] Kang Feng Ng & Quanlong Wang (2017): *A Universal Completion of the ZX-calculus*. arXiv:1706.09877.
- [43] Boldizsár Poór, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung & Bob Coecke (2023): *Completeness for Arbitrary Finite Dimensions of ZXW-calculus, a Unifying Calculus*. In: *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, Boston, MA, USA, pp. 1–14, doi:10.1109/LICS56636.2023.10175672. arXiv:2302.12135.
- [44] Michael Reck, Anton Zeilinger, Herbert J. Bernstein & Philip Bertani (1994): *Experimental Realization of Any Discrete Unitary Operator*. *Physical Review Letters* 73(1), pp. 58–61, doi:10.1103/PhysRevLett.73.58.
- [45] Razin A. Shaikh & Stefano Gogioso (2022): *Categorical Semantics for Feynman Diagrams*. arXiv:2205.00466.
- [46] Razin A. Shaikh, Quanlong Wang & Richie Yeung (2022): *How to Sum and Exponentiate Hamiltonians in ZXW Calculus*. arXiv:2212.04462.
- [47] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington & Ross Duncan (2020): *t|ket>: A Retargetable Compiler for NISQ Devices*. *Quantum Science and Technology* 6(1), p. 014003, doi:10.1088/2058-9565/ab8e92. arXiv:2003.10611.
- [48] Alexis Toumi, Richie Yeung & Giovanni de Felice (2021): *Diagrammatic Differentiation for Quantum Machine Learning*. In Chris Heunen & Miriam Backens, editors: *Proceedings 18th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 343, Open Publishing Association, pp. 132–144, doi:10.4204/EPTCS.343.7.
- [49] Rahul Trivedi, Kevin A. Fischer, Jelena Vučković & Kai Müller (2020): *Generation of Non-Classical Light Using Semiconductor Quantum Dots*. *Advanced Quantum Technologies* 3(1), p. 1900007, doi:10.1002/qute.201900007.
- [50] John van de Wetering (2020): *ZX-calculus for the Working Quantum Computer Scientist*. arXiv:2012.13966.
- [51] John van de Wetering & Lia Yeh (2022): *Phase Gadget Compilation for Diagonal Qutrit Gates*. arXiv:2204.13681.
- [52] Jamie Vicary (2008): *A Categorical Framework for the Quantum Harmonic Oscillator*. *International Journal of Theoretical Physics* 47(12), pp. 3408–3447, doi:10.1007/s10773-008-9772-4. arXiv:0706.0711.
- [53] Quanlong Wang (2022): *Qufinite ZX-calculus: A Unified Framework of Qudit ZX-calculi*. arXiv:2104.06429.
- [54] Quanlong Wang, Richie Yeung & Mark Koch (2022): *Differentiating and Integrating ZX Diagrams with Applications to Quantum Machine Learning*. arXiv:2201.13250.
- [55] Stephen C. Wein, Juan C. Loredó, Maria Maffei, Paul Hilaire, Abdelmounaim Harouri, Niccolò Somaschi, Aristide Lemaître, Isabelle Sagnes, Loïc Lanco, Olivier Krebs, Alexia Auffèves,



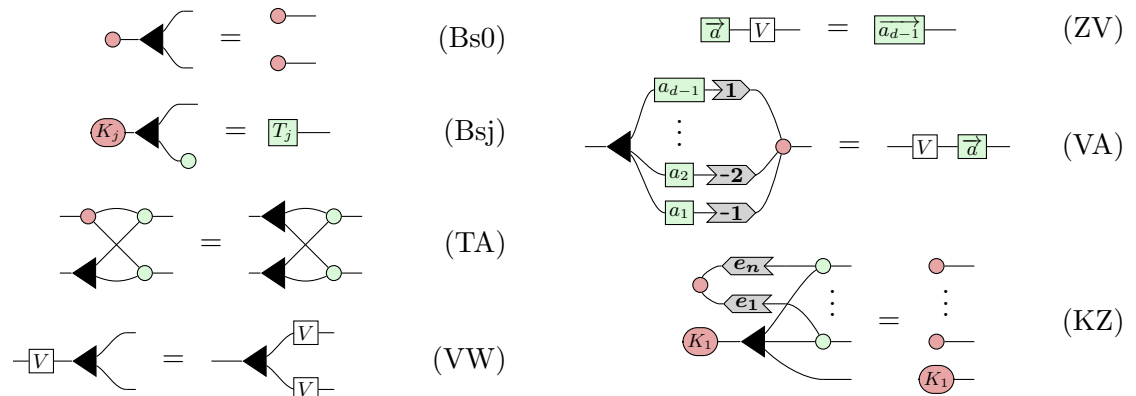
where $\overleftarrow{a} = (a_{d-1}, \dots, a_1)$, $\overrightarrow{ab} = (a_1 b_1, \dots, a_{d-1} b_{d-1})$, and $k_j(\overrightarrow{a}) = \left(\frac{a_{1-j}}{a_{d-j}}, \dots, \frac{a_{d-1-j}}{a_{d-j}} \right)$

In the definition of k_j we take the indices modulo d , that is, $a_j = a_{(j \bmod d)}$.

Qudit ZW-part of the rules



Qudit ZXW-part of the rules



where $T_j = \overbrace{(0, \dots, 1, \dots, 0)}^{d-1}$, $e_1, \dots, e_n \in \{1, \dots, d-1\}$, $\overrightarrow{a_{d-1}} = (a_{d-1}, a_{d-1}, \dots, a_{d-1})$

B Proof of lemmas

Lemma B.1. [53] In \mathbf{ZXW}_d , for any d , we have:

(S4)

Proof. Same as [43, Lemma 13]. □

Lemma B.2. In \mathbf{ZXW}_d , for any d , we have:

Proof. Same as [43, Lemma 25]. □

Lemma B.3. [53] In \mathbf{ZXW}_d , for any d , we have:

Proof. Same as [43, Lemma 35]. □

Lemma B.4. In \mathbf{ZXW}_d , for any d , we have:

Proof.

□

Lemma B.5. In \mathbf{ZW}_∞ , we have:

Proof. The proof is obtained by lifting [43, Lemma 38]. □

Lemma B.6. In \mathbf{ZW}_∞ we have:

$$\bullet \text{---} \boxed{\bar{a}} \text{---} = a_k \bullet \text{---}^k$$

Proof. Firstly, we show that in \mathbf{ZXW}_d we have:

$$\textcircled{K_{-k}} \text{---} \boxed{\bar{a}} \text{---} \stackrel{(S1)}{=} \textcircled{K_{-k}} \text{---} \textcircled{\bar{a}} \text{---} \stackrel{(K0)}{=} \textcircled{K_{-k}} \text{---} \boxed{\bar{a}} \text{---} = \textcircled{K_{-k}} \text{---} a_k \text{---}$$

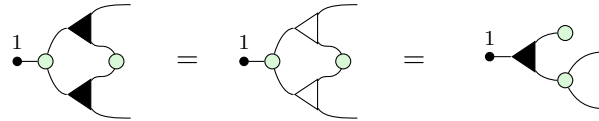
Then, the proof is acquired by lifting. □

Lemma B.7. In \mathbf{ZW}_∞ ,

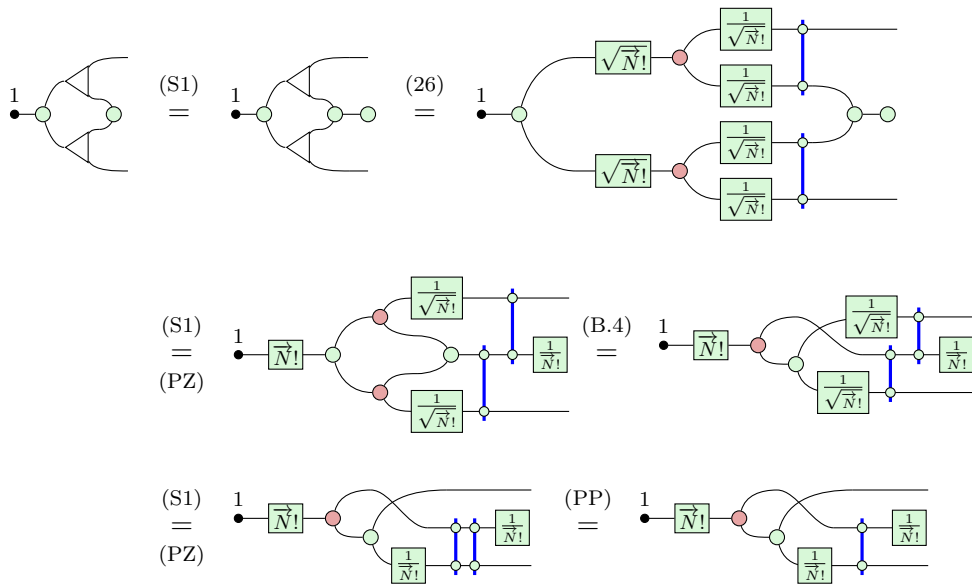


Proof. The statement follows from the lifting theorem as it is a trivial equation in \mathbf{ZXW}_d . □

Lemma B.8. In \mathbf{ZW}_∞ we have:



Proof. The first equation follows from Axioms (K0_∞) and (bW1). For the second equation, we start by truncating and reasoning in \mathbf{ZXW}_d :



$$\begin{array}{l}
\text{(S1)} \\
= \\
\text{(PZ)}
\end{array}
\begin{array}{c}
1 \\
\bullet \\
\sqrt{N!} \\
\sqrt{N!} \\
\bullet \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \text{(26)} \begin{array}{c}
1 \\
\bullet \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}$$

We lift the derivation above using Theorem 4.2 and end by reasoning in \mathbf{ZW}_∞ :

$$\begin{array}{l}
1 \\
\bullet \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \text{(B.6)} \begin{array}{c}
1 \\
\bullet \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \text{(bW1)} \begin{array}{c}
1 \\
\bullet \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}$$

$$= \text{(Pcy}_\infty) \begin{array}{c}
1 \\
\bullet \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \text{(B.6)} \begin{array}{c}
1 \\
\bullet \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}$$

□

Lemma B.9.

$$\begin{array}{c}
1 \\
\bullet \\
i\alpha \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
\stackrel{T_d}{\mapsto}
\begin{array}{c}
i\alpha \\
K_1 \\
(2, \dots, d-1, 0) \\
K_{-1}
\end{array}$$

Proof.

$$\begin{array}{c}
1 \\
\bullet \\
i\alpha \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \text{(K0)} \begin{array}{c}
1 \\
\bullet \\
i\alpha \\
1 \\
1 \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}
= \begin{array}{c}
i\alpha \\
1 \\
1 \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}$$

$$= \text{(26)} \begin{array}{c}
i\alpha \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
K_1 \\
K_{-1} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
\sqrt{N!}
\end{array}$$

$$= \text{(28)} \begin{array}{c}
i\alpha \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
K_1 \\
K_{-1} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
\sqrt{N!}
\end{array}$$

$$= \text{(PK)} \begin{array}{c}
i\alpha \\
\sqrt{N!} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
K_1 \\
(1, \dots, 1, 0) \\
(1, \dots, 1, 0) \\
K_{-1} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
\sqrt{N!}
\end{array}$$

$$= \text{(K0)} \begin{array}{c}
\frac{1}{\sqrt{N!}} \\
K_1 \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet \\
i\alpha \\
\left(\frac{1}{1!}, \dots, \frac{1}{(d-2)!}, 0\right) \\
K_{-1} \\
K_{-1} \\
\frac{1}{\sqrt{N!}} \\
\frac{1}{\sqrt{N!}} \\
\bullet
\end{array}$$

$$= \text{(S4)} \begin{array}{c}
i\alpha \\
\sqrt{N!} \\
K_1 \\
\left(\frac{1}{1!}, \dots, \frac{1}{(d-2)!}, 0\right) \\
K_{-1} \\
\sqrt{N!}
\end{array}$$

$$= \text{(K2)} \begin{array}{c}
i\alpha \\
K_1 \\
\left(\sqrt{2!}, \dots, \sqrt{(d-1)!}, 1\right) \\
\left(\frac{1}{1!}, \dots, \frac{1}{(d-2)!}, 0\right) \\
\left(\sqrt{2!}, \dots, \sqrt{(d-1)!}, 1\right) \\
K_{-1}
\end{array}$$

Proof. First, we diagonalise the truncation of the Hamiltonian H_P using ZXW rewrites:

(See Lemma B.9) Then, we derive the exponential in ZXW:

$$\exp \left(\text{---} \overset{i\alpha}{K_1} \text{---} \overset{(2, \dots, d-1, 0)}{\text{---}} \text{---} \overset{K_{-1}}{\text{---}} \text{---} \right) = \text{---} \overset{e^{i\tilde{N}\alpha}}{\text{---}} \text{---}$$

(See Lemma B.10) Lastly, the result is obtained by lifting. \square

Proposition 5.6. In $\mathbf{Z}\mathbf{W}_\infty$, the Kerr gate with parameter κ is given by:

Proof. In $\mathbf{Z}\mathbf{X}\mathbf{W}_d$

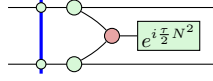
Then, one may exponentiate the diagram in the same way as Lemma B.10. \square

Proposition 5.7. The cross-Kerr gate with parameter κ is given by:

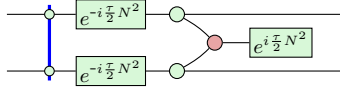
Proof. First, we demonstrate the following relation in $\mathbf{Z}\mathbf{X}\mathbf{W}_d$:

Based on this form, it becomes evident that the diagram corresponds to the operator $i\tau\hat{n}_1\hat{n}_2$, which needs to be exponentiated. We employ the technique described by van de Wetering and Yeh [51] for constructing diagonal qudit gates using *phase gadgets* [20, 38]. Firstly, note that the polynomial equation $(x+y)^2 = x^2 + 2xy + y^2$ implies that $e^{i\tau xy} = e^{i\frac{\tau}{2}((x+y)^2 - x^2 - y^2)} = e^{i\frac{\tau}{2}(x+y)^2} e^{-i\frac{\tau}{2}x^2} e^{-i\frac{\tau}{2}y^2}$. The same equation holds when considering x and y as operators \hat{n}_1

and \hat{n}_2 , respectively, given their commutation. Utilizing these calculations, we can express the truncated cross-Kerr gate as the composition of the number operator on each wire and the diagram corresponding to $e^{i\frac{\tau}{2}(x+y)^2}$. The former is given by Proposition 5.5, while the latter is represented by the diagram in \mathbf{ZXW}_d :

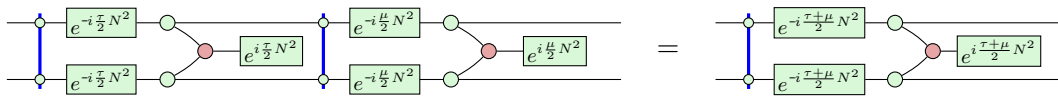


By composing the diagrams and sliding the projector through the green phases according to (PZ), we obtain the cross-Kerr operator in \mathbf{ZXW}_d :

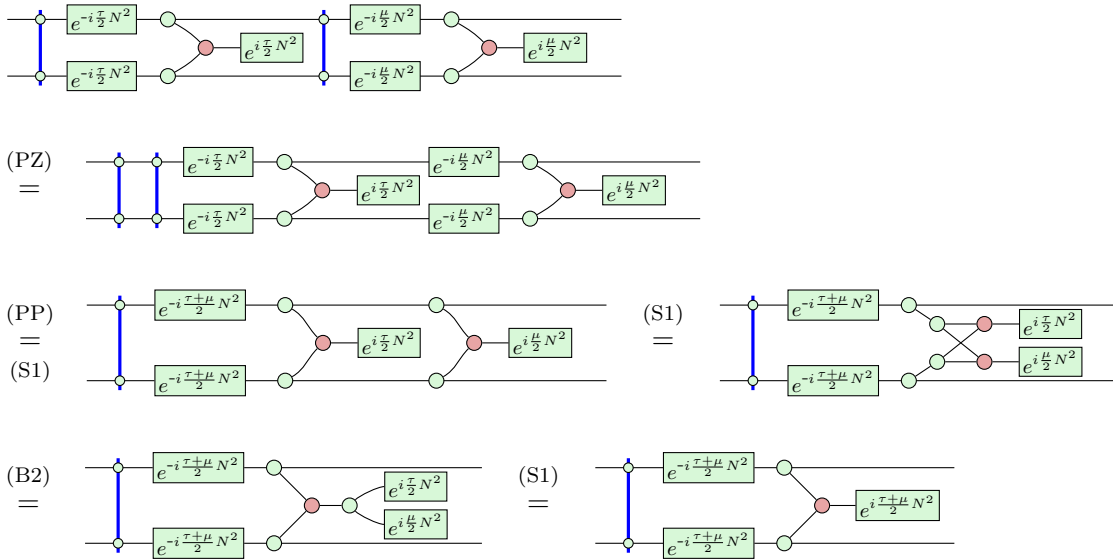


Lifting this diagram to \mathbf{ZW}_∞ completes the proof of the proposition. □

Proposition 5.8. *The composition of two cross-Kerr media with parameters τ and μ results in a cross-Kerr interaction with parameter $\tau + \mu$. Specifically, in \mathbf{ZXW}_d the following equation holds:*



Proof. In \mathbf{ZXW}_d , the following holds:



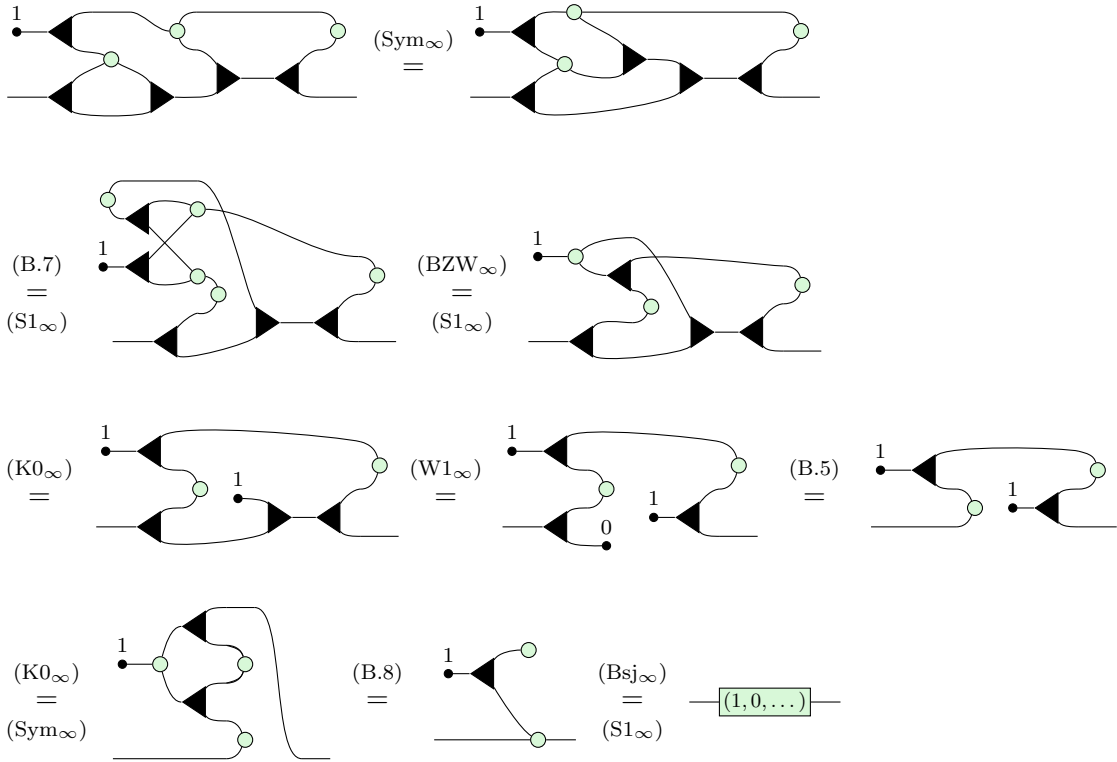
□

Proposition 5.9. *In \mathbf{ZW}_∞ , $\sigma^- \sigma^+ + \sigma^+ \sigma^- = id$, that is:*

(38)

where the right-hand side is the identity on the qubit subspace.

Proof. We reason entirely in \mathbf{ZW}_∞ :



□

Locally Tomographic Shadows (Extended Abstract)

Howard Barnum
hnbarnum@aol.com

Matthew A. Graydon
Institute for Quantum Computing
University of Waterloo
m3graydo@waterloo.ca

Alex Wilce
Susquehanna University
wilce@susqu.edu

Given a monoidal probabilistic theory — a symmetric monoidal category \mathcal{C} of systems and processes, together with a functor \mathbf{V} assigning concrete probabilistic models to objects of \mathcal{C} — we construct a locally tomographic probabilistic theory $\text{LT}(\mathcal{C}, \mathbf{V})$ — the *locally tomographic shadow* of $(\mathcal{C}, \mathbf{V})$ — describing phenomena observable by local agents controlling systems in \mathcal{C} , and able to pool information about joint measurements made on those systems. Some globally distinct states become locally indistinguishable in $\text{LT}(\mathcal{C}, \mathbf{V})$, and we restrict the set of processes to those that respect this indistinguishability. This construction is investigated in some detail for real quantum theory.

1 Introduction

As is well known, complex quantum theory is distinguished from its real counterpart by a property called *tomographic locality* or *local tomography* [4, 10]: states of a bipartite system are entirely determined by the joint probabilities they assign to measurements performed separately on the two component systems. That this fails for finite-dimensional real quantum theory is evident on dimensional grounds, but it's more instructive to note that if \mathbf{H} and \mathbf{K} are real Hilbert spaces, the space $\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$ of bounded self-adjoint operators on $\mathbf{H} \otimes \mathbf{K}$ decomposes as $\mathcal{L}_{ss} \oplus \mathcal{L}_{aa}$, where $\mathcal{L}_{ss} := \mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})$ and $\mathcal{L}_{aa} := \mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K})$, with $\mathcal{L}_a(\mathbf{H})$ the space of bounded anti-self adjoint operators on \mathbf{H} , and similarly for \mathbf{K} . This is an orthogonal decomposition with respect to the trace inner product. Thus, a density operator ρ on $\mathbf{H} \otimes \mathbf{K}$ has a unique decomposition $\rho_{ss} + \rho_{aa}$, with $\rho_{ss} \in \mathcal{L}_{ss}$ and $\rho_{aa} \in \mathcal{L}_{aa}$. Given effects $a \in \mathcal{L}_s(\mathbf{H})$ and $b \in \mathcal{L}_s(\mathbf{K})$, $a \otimes b$ lives in \mathcal{L}_{ss} , so $\text{Tr}((a \otimes b)\rho_{aa}) = 0$, and thus $\text{Tr}((a \otimes b)\rho) = \text{Tr}((a \otimes b)\rho_{ss})$. In other words, states with the same \mathcal{L}_{ss} component but distinct \mathcal{L}_{aa} components are *locally indistinguishable*.¹

This suggests a construction: simply “factor out” the non-locally observable \mathcal{L}_{aa} degrees of freedom to obtain a locally tomographic theory. Here, we not only show that this is feasible, but go much further. By a *monoidal probabilistic theory* we mean a pair $(\mathcal{C}, \mathbf{V})$ where \mathcal{C} is a symmetric monoidal category and $\mathbf{V} : \mathcal{C} \rightarrow \mathbf{Prob}$ is a functor from a symmetric monoidal category \mathcal{C} to a suitable category of concrete probabilistic models, taking monoidal products in \mathcal{C} to (non-signaling) composites in \mathbf{Prob} . We outline this framework in Section 2.² Given a such a theory $(\mathcal{C}, \mathbf{V})$, we construct a new, locally tomographic theory, $\text{LT}(\mathcal{C}, \mathbf{V})$, that describes what the world “looks like” to local agents, at least if one restricts

¹In contrast, in CQM, any anti-selfadjoint operator a has the form ib where b is self-adjoint. Hence, if $a = ib$ and $a' = ib'$ are anti-self adjoint, $a \otimes a'$ (as an element of $\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$) coincides with $-(b \otimes b') \in \mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})$. That is: $\mathcal{L}_{ss} = \mathcal{L}_{aa} = \mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$.

²Because this slightly extends the standard GPT framework as described, e.g., in [5], we go into a bit of detail here

attention to processes that respect local indistinguishability. Taking a cue from Plato, we call this the *locally tomographic shadow* of the original theory. This is described in Section 3. In Section 4, we return to the case of real QM, and show that our construction leads to a non-trivial theory that differs from both real and complex QM, but still allows for entangled states and effects.

If one lifts the restriction on processes mentioned above, it is still possible to construct a “shadow” theory, at the price of accepting an additional layer of non-determinism on top of that built into the probabilistic structure of the original model. This is briefly discussed in the concluding section 5, along with a number of other directions for further work.

This is a preliminary sketch of a longer work in progress.

2 Generalized Probabilistic Theories

For our purposes, a *probabilistic model* [3, 5] is pair (\mathbb{V}, u) where \mathbb{V} is an ordered real vector space and u is a strictly positive linear functional thereon, referred to as the *unit effect* of the model. Elements α of \mathbb{V}_+ with $u(\alpha) = 1$ are the (normalized) *states* of the system. *Effects* (measurement results) are positive functionals a on \mathbb{V} with $a \leq u$; $a(\alpha)$ is the probability of a 's of occurring in state α . Where we wish to keep track of several models, we label them A, B , etc., writing, e.g., $\mathbb{V}(A)$, $\mathbb{V}^*(A)$ for the associated ordered vector spaces and their duals, and u_A , for the unit effects. A *process* from a probabilistic model $(\mathbb{V}(A), u_A)$ to a probabilistic model $(\mathbb{V}(B), u_B)$ is a positive linear mapping $\phi : \mathbb{V}(A) \rightarrow \mathbb{V}(B)$ with $u_B(\phi(\alpha)) \leq u_A(\alpha)$ for every $\alpha \in \mathbb{V}(A)_+$

We write **Prob** for the category of probabilistic models and processes. In the broadest sense, a probabilistic theory is simply a functor \mathbb{V} from some category \mathcal{C} into **Prob**.³ \mathcal{C} can be understood to consist of “actual” physical systems and processes, or of any mathematical proxies for these (classical phase spaces, Hilbert spaces, open regions of spacetime, spin networks, or what-have-you). \mathbb{V} attaches probabilistic apparatus to these systems and processes in such a way as, e.g., to describe the possible experiences of agents.

In what follows, we denote such a probabilistic theory as a pair $(\mathcal{C}, \mathbb{V})$. We assume that \mathbb{V} is *injective on objects*, which makes $\mathbb{V}(\mathcal{C})$ a subcategory of **Prob**. This assumption holds for all of the examples we wish to consider. Where \mathbb{V} is also injective on morphisms, we say that $(\mathcal{C}, \mathbb{V})$ is *process tomographic* [7], in which case, we can simply identify \mathcal{C} with the corresponding subcategory of **Prob**. Note that the latter — more exactly, the probabilistic theory $(\mathbb{V}(\mathcal{C}), I)$, where $I : \mathbb{V}(\mathcal{C}) \rightarrow \mathbf{Prob}$ is the inclusion functor — is automatically process tomographic, regardless of whether or not $(\mathcal{C}, \mathbb{V})$ is so.

The example of primary interest here takes $\mathcal{C} = \mathbf{CPM}_{\mathbb{R}}$, the category of finite-dimensional real Hilbert spaces with morphisms $\mathbf{H} \rightarrow \mathbf{K}$ given by completely positive linear mappings $\mathcal{L}(\mathbf{H}) \rightarrow \mathcal{L}(\mathbf{K})$. In contrast to the complex case, there exist linear mappings $\mathcal{L}_s(\mathbf{H}) \rightarrow \mathcal{L}_s(\mathbf{K})$ that preserve positivity but not adjoints [9]. We reserve the term positive for those linear mappings that preserve both. Equivalently, $\phi : \mathcal{L}(\mathbf{H}) \rightarrow \mathcal{L}(\mathbf{K})$ is positive iff ϕ maps positive operators to positive operators, hence mapping $\mathcal{L}_s(\mathbf{H})$

³Note that such a functor \mathbb{V} comes with a designated unit functional $u_A \in \mathbb{V}(A)^*$ for every object $A \in \mathcal{C}$ with $u_A \circ \mathbb{V}(\phi) \leq u_B$ for all $\phi \in \mathcal{C}(A, B)$.

to $\mathcal{L}_s(\mathbf{K})$, and also maps $\mathcal{L}_a(\mathbf{H})$ to $\mathcal{L}_a(\mathbf{K})$. The probabilistic theory we have in mind when we speak of real QM is then $(\mathbf{CPM}_{\mathbb{R}}, \mathbb{V})$ where $\mathbb{V}(\mathbf{H}) = \mathcal{L}_s(\mathbf{H})$, the latter understood as an order unit space with $u_{\mathbf{H}} = \text{Tr}(\cdot)$. In contrast to the complex case, the restriction of ϕ to the self-adjoint part, $\mathcal{L}_s(\mathbf{H})$, of $\mathcal{L}(\mathbf{H})$ does not determine ϕ , so $\mathbb{R}\mathbf{QM}$ is not process tomographic.

Composite Models A (non-signaling) *composite* of models $\mathbb{V}(A)$ and $\mathbb{V}(B)$ is a model $\mathbb{V}(AB)$, together with a pair of bilinear mappings

$$m : \mathbb{V}(A) \times \mathbb{V}(B) \rightarrow \mathbb{V}(AB) \text{ and } \pi : \mathbb{V}(A)^* \times \mathbb{V}(B)^* \rightarrow \mathbb{V}(AB)^*$$

such that (i) $\omega \circ \pi$ is a joint state on A and B for all $\omega \in \mathbb{V}(AB)$, and (ii) $\pi(a, b)m(\alpha, \beta) = a(\alpha)b(\beta)$ for all $a \in \mathbb{V}^*(A), b \in \mathbb{V}^*(B)$ and all $\alpha \in \mathbb{V}(A), \beta \in \mathbb{V}(B)$.

We think of $\pi(a, b)$ as representing the joint outcome (a, b) of a pair of experiments performed on A and B , and $m(\alpha, \beta)$ as the result of preparing states α and β independently on A and B . We refer to $\pi(a, b)$ and $m(\alpha, \beta)$ as *product effects* and states, respectively.

If \mathbf{H}_A and \mathbf{H}_B are two complex (finite-dimensional) Hilbert spaces, the obvious choice for a composite model is $\mathbb{V}(AB) = \mathbb{V}(\mathbf{H}_A \otimes \mathbf{H}_B) = \mathcal{L}_s(\mathbf{H}_A \otimes \mathbf{H}_B)$, with $\pi(a, b)(\omega) = \text{Tr}((a \otimes b)\omega)$ and $m(\alpha, \beta) = \alpha \otimes \beta$.

If every state $\omega \in \mathbb{V}(AB)$ is determined by the joint probabilities $\omega(\pi(a, b))$, we say that AB is *locally tomographic*. Given arbitrary models $\mathbb{V}(A), \mathbb{V}(B)$, there are two extremal locally tomographic composites, obtained by endowing $\mathbb{V}(A) \otimes \mathbb{V}(B)$ with the *minimal* and *maximal* tensor cones [5, 12, 14]. The former is generated by *separable* states, i.e., convex combinations of product states. The latter consists of all bilinear forms that are positive on products of effects. We write $\mathbb{V}(A) \otimes_{\min} \mathbb{V}(B)$ and $\mathbb{V}(A) \otimes_{\max} \mathbb{V}(B)$ for $\mathbb{V}(A) \otimes \mathbb{V}(B)$ as ordered by these minimal and maximal cones, respectively. A composite $\mathbb{V}(AB)$ is locally tomographic iff $m : \mathbb{V}(A) \otimes \mathbb{V}(B) \rightarrow \mathbb{V}(AB)$ is a linear isomorphism, and in this case it's usual simply to identify the two spaces. One then has

$$(\mathbb{V}(A) \otimes_{\min} \mathbb{V}(B))_+ \leq \mathbb{V}(AB)_+ \leq (\mathbb{V}(A) \otimes_{\max} \mathbb{V}(B))_+.$$

If A and B are quantum models, the inclusions are proper: $(\mathbb{V}(A) \otimes_{\min} \mathbb{V}(B))_+$ contains only separable states, while $(\mathbb{V}(A) \otimes_{\max} \mathbb{V}(B))_+$ contains states corresponding to non-positive operators [11, 12]. More generally, $\mathbb{V}(A) \otimes_{\min} \mathbb{V}(B)$ allows only separable states, but arbitrarily entangled effects; the maximal tensor product $\mathbb{V}(A) \otimes_{\max} \mathbb{V}(B)$ allows the reverse. Both tensor products are naturally associative, and extending straightforwardly to tensor products of more than two factors.

Definition: A *monoidal probabilistic theory* is a structure $(\mathcal{C}, \mathbb{V}, m, \pi)$ where \mathcal{C} is a symmetric monoidal category, \mathbb{V} is a functor $\mathcal{C} \rightarrow \mathbf{Prob}$, and m and π are assignments, to every pair of objects $A, B \in \mathcal{C}$, of bilinear mappings

$$m_{A,B} : \mathbb{V}(A) \times \mathbb{V}(B) \rightarrow \mathbb{V}(AB) \text{ and } \pi_{A,B} : \mathbb{V}^*(A) \times \mathbb{V}^*(B) \rightarrow \mathbb{V}^*(AB)^4$$

such that

- (i) $m_{A,B}, \pi_{A,B}$ make $\mathbb{V}(AB)$ a composite of $\mathbb{V}(A)$ and $\mathbb{V}(B)$,
- (ii) $\mathbb{V}(I) = \mathbb{R}$ with

$$m_{I,A} : \mathbb{V}(I) \times \mathbb{V}(A) \rightarrow \mathbb{V}(A), \quad \pi_{I,A} : \mathbb{V}^*(I) \times \mathbb{V}^*(A) \rightarrow \mathbb{V}^*(A)$$

⁴We will write monoidal products in \mathcal{C} juxtapositively, reserving the symbol \otimes for the tensor product of linear spaces.

- the bilinear mappings uniquely defined by $m(1, \alpha) = \alpha$ and $\pi_{I,A}(1, a) = a$, and similarly for $m_{I,A}$ and $\pi_{A,I}$;
- (iii) $\mathbb{V}(\sigma_{A,B}) \circ \pi_{A,B} = \pi_{B,A}$, and similarly $m_{A,B} \circ \mathbb{V}(\sigma_{A,B}) = m_{B,A}$; and
 - (iv) for all morphisms $\phi : A \rightarrow A'$ and $\psi : B \rightarrow B'$ in \mathcal{C} , the diagram

$$\begin{array}{ccc}
 \mathbb{V}(A) \otimes \mathbb{V}(B) & \xleftarrow{\pi_{A,B}^*} & \mathbb{V}(AB) \\
 \downarrow \mathbb{V}(\phi) \otimes \mathbb{V}(\psi) & & \downarrow \mathbb{V}(\phi \otimes \psi) \\
 \mathbb{V}(A') \otimes \mathbb{V}(B') & \xleftarrow{\pi_{A',B'}^*} & \mathbb{V}(A'B')
 \end{array} \tag{1}$$

commutes.

If $\alpha \in \mathcal{C}(I, A)$, then by condition (ii), $\mathbb{V}(\alpha) : \mathbb{R} \rightarrow \mathbb{V}(A)$ defines an element of $\mathbb{V}(A)$, namely $\mathbb{V}(\alpha)(1)$. We make it a standing assumption in what follows that every normalized state in $\mathbb{V}(A)$ arises in this way.

Remark: On the left side of (1), $\mathbb{V}(\phi) \otimes \mathbb{V}(\psi)$ is the usual tensor product of linear maps, while on the right, $\phi \otimes \psi$ is the monoidal composite of the morphisms ϕ and ψ in \mathcal{C} . An equivalent statement is that, for all states $\omega \in \mathbb{V}(AB)$ and all effects $a' \in \mathbb{V}^*(A')$ and $b' \in \mathbb{V}^*(B')$, we have

$$\omega(\mathbb{V}(\phi)^*(a'), \mathbb{V}(\psi)^*(b')) = \mathbb{V}(\phi \otimes \psi)(\omega)(a', b').$$

More compactly: m and π^* are natural transformations from the functor $\mathbb{V} \otimes_{\min} \mathbb{V}$ to the functor $\mathbb{V} \circ \otimes$, and from $\mathbb{V} \circ \otimes$ to $\mathbb{V} \otimes_{\max} \mathbb{V}$, respectively. When no confusion seems likely, we will henceforth write $(\mathcal{C}, \mathbb{V})$ for a monoidal probabilistic theory $(\mathcal{C}, \mathbb{V}, m, \pi)$.

Evidently, $\mathbb{R}\mathbf{QM}$, with its standard compositional structure, is an example of such a monoidal probabilistic theory.

All of the foregoing applies to composites of more than two models. One can show that both \otimes_{\min} and \otimes_{\max} are both naturally associative. If one has an n -partite composite $A = A_1 \cdots A_n$, built up recursively so that $A = (A_1 \cdots A_{n-1})A_n$, then arguing inductively one has canonical positive mappings $\otimes_{\min} \mathbb{V}(A_i) \xrightarrow{m} \mathbb{V}(A) \xrightarrow{\pi^*} \otimes_{\max} \mathbb{V}(A_i)$.

We call a monoidal probabilistic theory $(\mathcal{C}, \mathbb{V})$ *locally tomographic* iff, for every pair of objects $A, B \in \mathcal{C}$, the composite $\mathbb{V}(AB)$ is locally tomographic.

As pointed out above, as long as \mathbb{V} is injective on objects, as we are assuming here, process-tomography can be enforced by passing to the probabilistic theory. $(\mathbb{V}(\mathcal{C}), I)$ For local tomography, the situation is not so simple. Still, as we show in the next section, it is possible to construct, from a given probabilistic theory, a kind of locally tomographic quotient theory.

3 Locally tomographic Shadows

If a monoidal probabilistic theory $(\mathcal{C}, \mathbb{V})$ is not locally tomographic, one can still ask what the world it describes would “look like” to agents having access only to local measurements. As a first step, we need to assume that objects in \mathcal{C} can be carved up in a preferred way into local pieces. To ensure this, we replace \mathcal{C} with its strictification, the category \mathcal{C}^* having as objects, finite lists $\vec{A} = (A_1, \dots, A_n)$ of non-trivial (non-identity) objects $A_i \in \mathcal{C}$, with the understanding that this represents the composite $\prod_{i=1}^n A_i$ in \mathcal{C} , but with the indicated monoidal decomposition. Morphisms from $\vec{A} = (A_1, \dots, A_n)$ to $\vec{B} = (B_1, \dots, B_k)$ are simply be morphisms $\prod_{i=1}^n A_i \rightarrow \prod_{i=1}^k B_i$ in \mathcal{C} . This is a strict symmetric monoidal category, with $(A_1, \dots, A_n)(B_1, \dots, B_k) = (A_1, \dots, A_n, B_1, \dots, B_k)$, the empty sequence $()$ serving as the monoidal unit. There is a (strong, but not strict) monoidal functor $\Pi : \mathcal{C}^* \rightarrow \mathcal{C}$ taking $\vec{A} = (A_1, \dots, A_n)$ to $\Pi \vec{A} := \prod_{i=1}^n A_i$, with $\Pi() = I_{\mathcal{C}}$, and acting as the identity on morphisms. This is not generally injective on objects (for instance, $\Pi(A, BC) = \Pi(A, B, C)$). Composing Π with \mathbb{V} now gives us a probabilistic theory $(\mathcal{C}^*, \mathbb{V} \circ \Pi)$ in which we have the desired canonical decompositions, but in which we’ve lost the desirable injectivity-on-objects feature. This feature will be recovered when we pass to the locally tomographic shadow, which we will now construct.

3.1 The shadow of a composite

If AB is a composite of probabilistic models A and B , then a state $\omega \in \Omega(AB)$ of the composite system determines a joint state $\tilde{\omega} : \mathbb{V}(A)^* \times \mathbb{V}(B)^* \rightarrow [0, 1]$, given by $\tilde{\omega}(a, b) := \pi_{A,B}(a, b)(\omega)$. This describes the joint probabilities of measurement outcomes carried out on A and B separately. We may call $\tilde{\omega}$ the *local shadow* of ω . More generally, suppose that $\vec{A} := (A_1, \dots, A_n)$ is a sequence in \mathcal{C}^* with composite $\Pi \vec{A} := A$ in \mathcal{C} : as we’ve seen, there is a positive linear mapping

$$\pi_{\vec{A}}^* : \mathbb{V}(A) \rightarrow \mathcal{L}^n(\mathbb{V}^*(A_1), \dots, \mathbb{V}^*(A_n))$$

taking $\omega \in \Omega(A)$ to the corresponding joint state $\tilde{\omega}$ on A_1, \dots, A_n — its local shadow — given by

$$\tilde{\omega}(a_1, \dots, a_n) := \pi_{\vec{A}}^*(\omega)(a_1, \dots, a_n) = (a_1 \otimes \dots \otimes a_n)(\omega)$$

for all $(a_1, \dots, a_n) \in \mathbb{V}^*(A_1) \times \dots \times \mathbb{V}^*(A_n)$.

Definition: For $\vec{A} = (A_1, \dots, A_n) \in \mathcal{C}^*$, let $\tilde{\mathbb{V}}(\vec{A})$ be the space $\otimes_i \mathbb{V}(A_i)$, ordered by the cone $\pi_{\vec{A}}^*(\mathbb{V}(\Pi \vec{A})_+)$ consisting of local shadows of elements of $\mathbb{V}(\Pi \vec{A})_+$, and let $\tilde{u}_{\vec{A}} = u_{A_1} \otimes \dots \otimes u_{A_n}$. We call the probabilistic model $(\tilde{\mathbb{V}}(\Pi \vec{A}), \tilde{u}_{\vec{A}})$ the *locally tomographic shadow* of $A = \Pi \vec{A}$ with respect to the given decomposition (that is, with respect to \vec{A}).

Going forward, it will be convenient to use the abbreviations $\text{LT}_{\vec{A}}$, or even just LT , for the mapping $\pi_{\vec{A}}^* : \mathbb{V}(A) \rightarrow \tilde{\mathbb{V}}(\vec{A})$, whenever context makes it convenient and relatively unambiguous to do so.

We have canonical mappings

$$\pi : \mathbb{V}^*(A_1) \times \dots \times \mathbb{V}^*(A_k) \rightarrow \tilde{\mathbb{V}}(A_1, \dots, A_k)^*, \text{ and } m : \mathbb{V}(A_1) \times \dots \times \mathbb{V}(A_k) \rightarrow \tilde{\mathbb{V}}(A_1, \dots, A_k)$$

given by

$$\pi(a_1, \dots, a_n)(\mu) = \mu(a_1, \dots, a_n) \text{ and } m(\alpha_1, \dots, \alpha_n)(a_1, \dots, a_n) = \prod_{i=1}^n a_i(\alpha_i).$$

It is straightforward that $(\tilde{\mathbb{V}}(A_1, \dots, A_n), \pi, m)$ is a composite of $\mathbb{V}(A_1), \dots, \mathbb{V}(A_n)$.

For any vector spaces $\mathbb{V}_1, \dots, \mathbb{V}_k$, let $\mathcal{L}^k(\mathbb{V}_1, \dots, \mathbb{V}_k)$ denote the space of k -linear forms on $\mathbb{V}_1 \times \dots \times \mathbb{V}_k$. Using the canonical isomorphism $\mathcal{L}^k(\mathbb{V}_1^*, \dots, \mathbb{V}_k^*) \simeq \bigotimes_{i=1}^k \mathbb{V}_i$, we can identify $\tilde{\mathbb{V}}(A_1, \dots, A_k)$, as a vector space, with $\mathbb{V}(A_1) \otimes \dots \otimes \mathbb{V}(A_k)$, now ordered by a cone $\tilde{\mathbb{V}}(A_1, \dots, A_n)_+$ with

$$\left(\bigotimes_{\min} \mathbb{V}(A_i) \right)_+ \subseteq \tilde{\mathbb{V}}(A_1, \dots, A_n)_+ \subseteq \left(\bigotimes_{\max} \mathbb{V}(A_i) \right)_+.$$

We also have an injective linear mapping from $\tilde{\mathbb{V}}(A_1, \dots, A_k) \simeq \bigotimes_i \mathbb{V}(A_i)$ into $\mathbb{V}(A_1, \dots, A_k)$, extending the map m taking $(\alpha_1, \dots, \alpha_n) \in \prod_i \mathbb{V}(A_i)$ to $\alpha_1 \otimes \dots \otimes \alpha_n \in \mathbb{V}(\prod_i A_i)$. However, this mapping is, as a rule, not positive. This will be illustrated below in the case of real quantum theory. This shows that $\tilde{\mathbb{V}}(A_1, \dots, A_n)_+$ is generally larger than the minimal tensor cone. As we'll also see in the next section, it is generally strictly smaller than the maximal tensor cone.

Further Notation: In the case of a bipartite system, we will sometimes find it convenient to use the tensor-like notation $\mathbb{V}(A) \boxtimes \mathbb{V}(B)$ for $\tilde{\mathbb{V}}(A, B)$. Thus, $\mathbb{V}(A) \boxtimes \mathbb{V}(B)$ is the vector space $\mathbb{V}(A) \otimes \mathbb{V}(B)$, but ordered by the cone $\tilde{\mathbb{V}}(A, B)$ generated by local shadows $\tilde{\omega}$ of states $\omega \in \mathbb{V}(AB)$.

The following identifies the effect cone of $\tilde{\mathbb{V}}(A_1, \dots, A_n)$. We omit the straightforward proof.

Lemma 1: $\tilde{\mathbb{V}}(A_1, \dots, A_n)_+^* \simeq \mathbb{V}^*(\prod_i A_i)_+ \cap (\bigotimes_i \mathbb{V}^*(A_i))$. *In the bipartite case:*

$$(\mathbb{V}(A) \boxtimes \mathbb{V}(B))_+^* \simeq \mathbb{V}(AB)^* \cap (\mathbb{V}(A)^* \otimes \mathbb{V}(B)^*).$$

3.2 Shadows of Processes

Given a probabilistic theory $(\mathcal{C}, \mathbb{V})$, we would now like to construct a locally tomographic ‘‘shadow’’ theory by applying the LT construction to the objects of $\mathbb{V}(\mathcal{C})$. In order to do this, we first need to decide what the morphisms should be in this putative ‘‘shadow’’ theory.

In what follows, $A = \Pi \vec{A}$ and $B = \Pi \vec{B}$ are composite systems, with specified decompositions $\vec{A} = (A_1, \dots, A_n)$ and $\vec{B} = (B_1, \dots, B_k)$. The proof of the following is routine.

Lemma 2: *Let $\Phi : \mathbb{V}(A) \rightarrow \mathbb{V}(B)$ be a positive linear mapping. The following are equivalent:*

- (a) Φ maps $\text{Ker}(LT_{\vec{A}})$ into $\text{Ker}(LT_{\vec{B}})$.
- (b) If $\omega, \omega' \in \mathbb{V}(A)$ are locally indistinguishable, so are $\Phi(\omega), \Phi(\omega')$ in $\mathbb{V}(B)$.
- (c) There exists a linear mapping $\phi : \bigotimes_i \mathbb{V}(A_i) \rightarrow \bigotimes_j \mathbb{V}(B_j)$ such that $LT_{\vec{B}} \circ \Phi = \phi \circ LT_{\vec{A}}$

Definition: With notation as above, call a positive linear mapping $\Phi : \mathbb{V}(A) \rightarrow \mathbb{V}(B)$ satisfying any, hence all, of these conditions *locally positive* (with respect to the specified decompositions). The linear mapping ϕ in part (c) is then uniquely determined. We call it the *shadow* of Φ , writing $\phi = \text{LT}(\Phi)$.

As an immediate consequence, we have

Lemma 3: *If $\Phi : \mathbb{V}(A) \rightarrow \mathbb{V}(B)$ is locally positive, then $\phi = \text{LT}(\Phi)$ is positive as a mapping from $\tilde{\mathbb{V}}(A_1, \dots, A_m) \rightarrow \tilde{\mathbb{V}}(B_1, \dots, B_n)$.*

Locally positive maps are reasonably abundant, but do exclude some important morphisms in \mathbb{RQM} . For instance, if σ and α are swap and associator morphisms in \mathcal{C} , $\mathbb{V}(\sigma)$ is locally positive, but $\mathbb{V}(\alpha)$ need not be.

For another example, if α is a state on $A = A_1 \otimes \dots \otimes A_n$, then the corresponding mapping $\alpha : \mathbb{R} = \mathbb{V}(I) \rightarrow \mathbb{V}(A)$ given by $\alpha(1) = \alpha$ is trivially locally positive (the kernel of LT_I is trivial). On the other hand, there is generally no guarantee that an effect $f : \mathbb{V}(A) \rightarrow \mathbb{R}$ will be locally positive. Indeed, if $\omega, \omega' \in \mathbb{V}(A)_+$ are distinct, there will certainly exist some positive linear functional f with $f(\omega) \neq f(\omega')$, and by re-scaling if necessary, we can take f to be an effect. If $\text{LT}(\omega) = \text{LT}(\omega')$, then f is not locally positive. Thus, the passage from $\mathbb{V}(A)$ to $\text{LT}(\mathbb{V}(A))$ not only identifies previously distinct states, but also jettisons some effects.

It follows from part (c) of Lemma 3.3 that if $\Phi : \Pi_i A_i \rightarrow \Pi_j B_j$ and $\Psi : \Pi_j B_j \rightarrow \Pi_k C_k$ are locally positive with shadows $\phi = \text{LT}(\Phi)$ and $\psi = \text{LT}(\Psi)$, then $\Psi \circ \Phi$ is locally positive with shadow $\psi \circ \phi$.

Definition: Let $(\mathcal{C}, \mathbb{V})$ be a probabilistic theory. For objects $\vec{A} = (A_1, \dots, A_n)$ and $\vec{B} = (B_1, \dots, B_k) \in \mathcal{C}^*$, call a morphism $\Pi \vec{A} \xrightarrow{f} \Pi \vec{B}$ *local* iff $\mathbb{V}(f) : \mathbb{V}(\Pi \vec{A}) \rightarrow \mathbb{V}(\Pi \vec{B})$ is locally positive (relative to the preferred factorizations of A and B). We write $\text{Loc}(\mathcal{C}, \mathbb{V})$ for the category having the same objects as \mathcal{C}^* — finite lists of non-identity objects in \mathcal{C} — but only local morphisms.

By the remarks above, $\text{Loc}(\mathcal{C}, \mathbb{V})$ is a monoidal sub-category of \mathcal{C}^* . By construction, the functor $\mathbb{V} \circ \Pi : \mathcal{C}^* \rightarrow \mathbf{Prob}$ descends to a functor $\text{Loc}(\mathcal{C}, \mathbb{V}) \rightarrow \mathbf{Prob}$. However, because Π is not injective on objects, neither will $\mathbb{V} \circ \Pi$ be. To remedy this, we make the following

Definition: For all objects $\vec{A}, \vec{B} \in \mathcal{C}^*$ and local morphisms $f : \vec{A} \rightarrow \vec{B}$, let $\tilde{\mathbb{V}}(\vec{A}) := \text{LT}_{\vec{A}}(\mathbb{V}(\Pi \vec{A}))$ and $\tilde{\mathbb{V}}(f) := \text{LT}(\mathbb{V}(f))$.

Lemma 4: *$\tilde{\mathbb{V}}$ is a functor $\text{Loc}(\mathcal{C}, \mathbb{V}) \rightarrow \mathbf{Prob}$, and is injective on objects.*

Proof: It is straightforward that $\tilde{\mathbb{V}}(f \circ g) = \tilde{\mathbb{V}}(f) \circ \tilde{\mathbb{V}}(g)$ where $f \in \mathcal{C}^*(A, B)$ and $g \in \mathcal{C}^*(B, C)$. That $\tilde{\mathbb{V}}$ is injective on objects follows from the fact that there is a canonical isomorphism between $\bigotimes \mathbb{V}(A_i)$ and $\mathcal{L}^k(\mathbb{V}(A_1)^*, \dots, \mathbb{V}(A_n)^*)$; the latter is literally a space of functions on $\mathbb{V}(A_1)^* \times \dots \times \mathbb{V}(A_n)^*$, from which one can read off the spaces $\mathbb{V}(A_1), \dots, \mathbb{V}(A_n)$. As \mathbb{V} is injective on objects, these in turn determine (A_1, \dots, A_n) . \square

Definition: The *locally tomographic shadow* of a monoidal probabilistic theory $(\mathcal{C}, \mathbb{V})$ is the probabilistic theory $\text{LT}(\mathcal{C}, \mathbb{V}) := (\text{Loc}(\mathcal{C}, \mathbb{V}), \tilde{\mathbb{V}})$.

By construction, $\text{LT}(\mathbb{V}, \mathcal{C}) = (\text{Loc}(\mathcal{C}, \mathbb{V}), \tilde{\mathbb{V}})$ is locally tomographic. We have the following picture: there are two functors (probabilistic models) associated with $\text{Loc}(\mathcal{C}, \mathbb{V})$: one is given by $\mathbb{V} \circ \Pi$, and the other by $\tilde{\mathbb{V}} := \text{LT}(\mathbb{V} \circ \Pi)$, i.e., for each object $\vec{A} = (A_1, \dots, A_n)$ in $\text{Loc}(\mathcal{C}, \mathbb{V})$, we have a positive linear mapping $\text{LT}_{\vec{A}} : \mathbb{V}(\Pi(\vec{A})) \rightarrow \tilde{\mathbb{V}}(\vec{A})$. By construction, LT then defines a natural transformation $\mathbb{V} \circ \Pi \Rightarrow \tilde{\mathbb{V}}$.

4 The Shadow of Real Quantum Theory

We'll now consider the case of finite-dimensional real quantum theory in some detail, concentrating on the bipartite case.

4.1 The LT map

Suppose \mathbf{H} and \mathbf{K} are two finite-dimensional real Hilbert spaces. In what follows, we identify states with the corresponding density operators, so that if W is a density operator on $\mathbf{H} \otimes \mathbf{K}$, $\text{LT}(W)$ is the unique operator in $\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})$ satisfying $\text{Tr}(\text{LT}(W)a \otimes b) = \text{Tr}(Wa \otimes b)$ for all $a \in \mathcal{L}_s(\mathbf{H})$ and $b \in \mathcal{L}_s(\mathbf{K})$.

As discussed in the Introduction, $\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$ has a natural orthogonal direct sum decomposition as

$$\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K}) = (\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})) \oplus (\mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K}))$$

Since $\text{Ker}(\text{LT})$ contains $\mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K})$ and $\text{Ran}(\text{LT})$ equals $\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{H})$, we have $\text{Ker}(\text{LT}) = \mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K})$.

Let $\text{Sym} : \mathcal{L}(\mathbf{H}) \rightarrow \mathcal{L}_s(\mathbf{H})$ be the symmetrization mapping $\text{Sym}(a) = \frac{1}{2}(a + a^t)$. Note that this is the orthogonal projection on $\mathcal{L}(\mathbf{H})$ (with respect to the trace inner product) with range $\mathcal{L}_s(\mathbf{H})$. A straightforward computation gives us

Lemma 5: $\text{LT}(W) = (\text{Sym} \otimes \text{Sym})(W)$ for all $W \in \mathcal{L}(\mathbf{H})$.

4.2 The locally tomographic cone

Using the notation introduced in Section 3, $\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{H})$ stands for $\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{H})$ as ordered by the *locally tomographic cone*

$$(\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+ := \text{LT}(\mathcal{L}_s(\mathbf{H} \otimes \mathbf{H}))_+.$$

Let $(\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K}))_+$ stand for the cone of positive operators belonging to $\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})$. That is,

$$(\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{H}))_+ := (\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{H})) \cap \mathcal{L}_s(\mathbf{H} \otimes \mathbf{H})_+.$$

A priori, we have

$$\begin{aligned} (\mathcal{L}_s(\mathbf{H}) \otimes_{\min} \mathcal{L}_s(\mathbf{K}))_+ &\subseteq (\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K}))_+ \\ &\subseteq (\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+ \subseteq (\mathcal{L}_s(\mathbf{H}) \otimes_{\max} \mathcal{L}_s(\mathbf{K}))_+. \end{aligned}$$

We'll presently see that if \mathbf{H} and \mathbf{K} are of dimension two or greater, all four of these inclusions are proper. If $x, y \in \mathbf{H}$, we use the standard operator-theoretic notation $x \odot y$ (rather than $|x\rangle\langle y|$ as in Dirac notation) for the operator on \mathbf{H} given by $(x \odot y)z = \langle z, y \rangle x$ for all $z \in \mathbf{H}$. If $\|x\| = 1$, then $x \odot x = P_x$, the rank-one projection onto the span of x .

Example 1: Let $\mathbf{H} = \mathbb{R}^2$ and let $\{x, y\}$ be any orthonormal basis. Let z be the real EPR state $z = \frac{1}{\sqrt{2}}(x \otimes y + y \otimes x)$. Direct computation shows that

$$z \odot z = \frac{1}{2}(P_x \otimes P_y + P_y \otimes P_x + (x \odot y) \otimes (y \odot x) + (y \odot x) \otimes (x \odot y))$$

Now, $\text{Sym}(x \odot y) = \frac{1}{2}(x \odot y + y \odot x) = \text{Sym}(y \odot x) =: S$ where $Sx = \frac{1}{2}y$ and $Sy = \frac{1}{2}x$. So $W := \text{LT}(z \odot z) = \frac{1}{2}(P_x \otimes P_y + P_y \otimes P_x) + S \otimes S$. This is not a positive operator. For instance, if $v = x \otimes x - y \otimes y$, then $Wv = -\frac{1}{4}v$.

This shows that the embedding $\mathbb{V}(A \boxtimes B) \rightarrow \mathbb{V}(AB)$ is in general not positive, as mentioned earlier. Consequently, $(\mathcal{L}_s \otimes \mathcal{L}_s)_+$ is strictly smaller than $(\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+$.

Theorem 1: Let $\dim \mathbf{H}$ and $\dim \mathbf{K}$ be 3 or greater. The cone $(\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K}))_+$ is strictly larger than the cone $(\mathcal{L}_s(\mathbf{H}) \otimes_{\min} \mathcal{L}_s(\mathbf{K}))_+$, and the cone $(\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+$ is strictly smaller than the cone $(\mathcal{L}_s(\mathbf{H}) \otimes_{\max} \mathcal{L}_s(\mathbf{K}))_+$.

Proof (sketch): To simplify notation a bit, let $\mathbf{H} = \mathbf{K}$, and write $\mathcal{L}_s(\mathbf{H})$ and $\mathcal{L}_a(\mathbf{H})$ as \mathcal{L}_s and \mathcal{L}_a . Lemma 1 tells us that $(\mathcal{L}_s \boxtimes \mathcal{L}_s)_+^* \simeq (\mathcal{L}_s \otimes \mathcal{L}_s)_+$. There exist well-known examples of entangled states in $\mathcal{L}_s \otimes \mathcal{L}_s$, namely those arising from unextendible product bases; see [6].⁵ Hence, $(\mathcal{L}_s \otimes \mathcal{L}_s) \cap \mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})_+$ is strictly larger than the minimal tensor cone, which contains only unentangled states. Dualizing, we see that $(\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+$ must be strictly smaller than the maximal tensor cone. \square

Remark: The geometry of the locally tomographic cone $(\mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K}))_+$ appears to be quite interesting. Although the mapping $\text{LT} : \mathcal{L}_s(\mathbf{H} \otimes \mathbf{K}) \rightarrow \mathcal{L}_s(\mathbf{H}) \boxtimes \mathcal{L}_s(\mathbf{K})$ is in general many-to-one, remarkably, this is not the case for pure states: as shown by Lemma 17 of [8], any pure state of $\mathbf{H} \otimes \mathbf{K}$ can be distinguished from any other state (pure or mixed) by product effects. Thus, if ω is a pure state in $\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$, it is the *only* state with local shadow $\text{LT}(\omega)$.⁶

4.3 Processes

Let $\Phi : \mathcal{L}_s(\mathbf{H} \otimes \mathbf{K}) \rightarrow \mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})$ be a positive linear mapping. For simplicity of notation, let's write $\mathcal{L}_{s,s}$ for $\mathcal{L}_s(\mathbf{H}) \otimes \mathcal{L}_s(\mathbf{K})$, $\mathcal{L}_{s,a}$ and $\mathcal{L}_{a,s}$ for $\mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K})$, and so on. We then have an orthogonal decomposition

$$\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K}) = \mathcal{L}_{s,s} \oplus \mathcal{L}_{s,a} \oplus \mathcal{L}_{a,s} \oplus \mathcal{L}_{a,a},$$

with respect to which Φ has an operator matrix $\begin{bmatrix} \Phi_{s,s} & \Phi_{s,a} \\ \Phi_{a,s} & \Phi_{a,a} \end{bmatrix}$, where, e.g., $\Phi_{s,s} : \mathcal{L}_{s,s} \rightarrow \mathcal{L}_{s,s}$, $\Phi_{a,s} : \mathcal{L}_{a,s} \rightarrow \mathcal{L}_{a,a}$, etc. A straightforward translation of Lemma 2 gives us

Lemma 6: Let Φ be as above. Then Φ is locally positive iff $\Phi_{s,a} = 0$, and in this case, $\text{LT}(\Phi) = \Phi_{s,s}$.

This provides another way to see that effects on $\mathbf{H} \otimes \mathbf{K}$, as processes

$$\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K}) \rightarrow \mathcal{L}_s(\mathbb{R}),$$

⁵We thank Giulio Chiribella for drawing our attention to these.

⁶If ω is an interior point in the cone $\mathcal{L}_s(\mathbf{H} \otimes \mathbf{K})_+$, then the affine space $\omega + \mathcal{L}_a(\mathbf{H}) \otimes \mathcal{L}_a(\mathbf{K})$, whose elements μ all satisfy $\text{Tr}(\mu(a \otimes b)) = \text{Tr}(\omega(a \otimes b))$ for all product effects $a \otimes b$, will certainly intersect the boundary of the positive cone. However, this intersection will not contain pure states, but only points interior to higher-dimensional faces of the cone.

are generally not locally positive. The following example is particularly noteworthy:

Example 2: Consider the functional $\varepsilon : \mathcal{L}(\mathbb{R}^2 \otimes \mathbb{R}^2) = \mathcal{L}(\mathbb{R}^2) \otimes \mathcal{L}(\mathbb{R}^2) \rightarrow \mathbb{R}$ corresponding to the trace inner product, i.e., the functional uniquely defined on pure tensors by $\varepsilon(a \otimes b) = \text{Tr}(ab^t)$. The subspace $\mathcal{L}_{a,a} \leq \mathcal{L}(\mathbb{R}^2)$ is one-dimensional, spanned by the operator $J \otimes J = \mathbb{J}$, where $J(x, y) = (-y, x)$. Note that $J^2 = -\mathbf{1}$. Hence $\varepsilon(\mathbb{J}) = \text{Tr}(J J^t) = -\text{Tr}(J^2) = \text{Tr}(\mathbf{1}) = 2$. Since ε does not vanish on $\mathcal{L}_{a,a}$, ε is not locally positive.

5 Conclusions and questions

At a minimum, $\text{LT}(\mathbb{R}\mathbf{QM})$ provides us with an interesting “foil” GPT, related to but distinct from both complex and real finite-dimensional real quantum theory, and from their Jordan-algebraic relatives [2] (which, like $\mathbb{R}\mathbf{QM}$, are not locally tomographic). Among the many questions that suggest themselves about this theory, and about the LT construction more generally, the following stand out to us as particularly interesting.

Compact Closure Example 2 shows that $\text{LT}(\mathbb{R}\mathbf{QM})$ does not inherit the usual compact structure from $\mathbb{R}\mathbf{QM}$. Given a monoidal probabilistic theory, theory $(\mathcal{C}, \mathbb{V})$ with \mathcal{C} compact closed, when is $\text{LT}(\mathcal{C}, \mathbb{V})$ compact closed?

LT and Complex QM The functor $R : \mathbb{C}\mathbf{QM} \rightarrow \mathbb{R}\mathbf{QM}$ given by restriction of scalars does not preserve tensor products. It would be of interest to understand the functor $\text{LT} \circ R$ from $\mathbb{C}\mathbf{QM}$ to $\text{LT}(\mathbb{R}\mathbf{QM})$. One can ask a parallel question about complexification.

The Shadow of InvQM In [2], we constructed a non-locally-tomographic theory we called **InvQM**, which contains finite-dimensional real and quaternionic QM as sub-theories, and also a relative of complex QM in which the composite of two complex quantum systems comes with an extra binary superselection rule. As we will discuss elsewhere, much of what was done above for $\mathbb{R}\mathbf{QM}$ generalizes readily to **InvQM**, but the resulting theory — like $\text{LT}(\mathbb{R}\mathbf{QM})$ — remains largely unexplored.

Non-deterministic shadows If local agents Alice and Bob agree that their joint state is ω , this is consistent with the actual, global state being any element $\mu \in \text{LT}_{A,B}^{-1}(\omega)$. If the (unknown) actual state evolves under a (global) process $\phi : \mathbb{V}(AB) \rightarrow \mathbb{V}(CD)$, the result will be one of the states in $\phi(\text{LT}_{A,B}^{-1}(\omega))$. Unless ϕ is local, these will not be confined to a single fibre of $\text{LT}_{C,D}$; parties C and D might observe any of the different states in $\text{LT}_{C,D}(\phi(\text{LT}_{A,B}^{-1}(\omega)))$, giving the impression that ϕ acts indeterministically. How ought this uncertainty be quantified? Note that in this situation, the states of AB act as hidden variables “explaining” this apparent lack of determinism.

References

- [1] Guillaume Aubrun, Ludovico Lami, Carlos Palazuelos & Martin Plávala (2021): *Entangleability of cones*. *Geometric and Functional Analysis* 31(2), pp. 181–205, doi:10.1007/s00039-021-00565-5.

- [2] Howard Barnum, Matthew A. Graydon & Alexander Wilce (2020): *Composites and categories of Euclidean Jordan algebras*. *Quantum* 4, p. 359, doi:10.22331/q-2020-11-08-359.
- [3] Howard Barnum & Alexander Wilce (2011): *Information Processing in Convex Operational Theories*. *Electronic Notes in Theoretical Computer Science* 270(1), pp. 3–15, doi:10.1016/j.entcs.2011.01.002. Proceedings of the Joint 5th International Workshop on Quantum Physics and Logic and 4th Workshop on Developments in Computational Models (QPL/DCM 2008).
- [4] Howard Barnum & Alexander Wilce (2014): *Local tomography and the Jordan structure of quantum theory*. *Foundations of Physics* 44, pp. 192–212, doi:10.1007/s10701-014-9777-1.
- [5] Howard Barnum & Alexander Wilce (2016): *Post-Classical Probability Theory*. In Giulio Chiribella & Robert W. Spekkens, editors: *Quantum Theory: Informational Foundations and Foils, Fundamental Theories of Physics* 181, Springer, Dordrecht, pp. 367–420, doi:10.1007/978-94-017-7303-4_11.
- [6] Charles H. Bennett, David P. DiVincenzo, Tal Mor, Peter W. Shor, John A. Smolin & Barbara M. Terhal (1999): *Unextendible product bases and bound entanglement*. *Physical Review Letters* 82(26), p. 5385, doi:10.1103/PhysRevLett.82.5385.
- [7] Giulio Chiribella (2021): *Process Tomography in General Physical Theories*. *Symmetry* 13(11), p. 1985, doi:10.3390/sym13111985.
- [8] Giulio Chiribella, Giacomo Mauro D’Ariano & Paolo Perinotti (2010): *Probabilistic theories with purification*. *Phys. Rev. A* 81, p. 062348, doi:10.1103/PhysRevA.81.062348.
- [9] Giulio Chiribella, Kenneth R. Davidson, Vern I. Paulsen & Mizanur Rahaman (2023): *Positive maps and entanglement in real Hilbert spaces*. *Annales Henri Poincaré* 2023, pp. 1–30, doi:10.1007/s00023-023-01325-x.
- [10] Lucien Hardy (2001): *Quantum theory from five reasonable axioms*. *arXiv preprint quant-ph/0101012*, doi:10.48550/arXiv.quant-ph/0101012.
- [11] Matthias Kläy, Charles Randall & David Foulis (1987): *Tensor products and probability weights*. *International Journal of Theoretical Physics* 26, pp. 199–219, doi:10.1007/BF00668911.
- [12] Alexander Wilce (1992): *Tensor products in generalized measure theory*. *International Journal of Theoretical Physics* 31, pp. 1915–1928, doi:10.1007/BF00671964.
- [13] Alexander Wilce (2018): *A Shortcut from Categorical Quantum Theory to Convex Operational Theories*. *Electronic Proceedings in Theoretical Computer Science* 266, pp. 222–236, doi:10.4204/eptcs.266.15. Proc. QPL 2017.
- [14] Gerd Wittstock (1974): *Ordered normed tensor products*. In A. Hartkämper & Holger Neumann, editors: *Foundations of Quantum Mechanics and Ordered Linear Spaces: Advanced Study Institute Marburg 1973, Lecture Notes in Physics* 29, Springer, Berlin, Heidelberg, pp. 67–84, doi:10.1007/3-540-06725-6_10.

Quantum Suplattices

Gejza Jenča

Slovak University of Technology,
Bratislava, Slovak Republic
gejza.jenca@stuba.sk

Bert Lindenhovius

Slovak Academy of Sciences,
Bratislava, Slovak Republic
lindenhovius@mat.savba.sk

Building on the theory of quantum posets, we introduce a non-commutative version of suplattices, i.e., complete lattices whose morphisms are supremum-preserving maps, which form a step towards a new notion of quantum topological spaces. We show that the theory of these ‘quantum suplattices’ resembles the classical theory: the opposite quantum poset of a quantum suplattice is again a quantum suplattice, and quantum suplattices arise as algebras of a non-commutative version of the monad of downward-closed subsets of a poset. The existence of this monad is proved by introducing a non-commutative generalization of monotone relations between quantum posets, which form a compact closed category. Moreover, we introduce a non-commutative generalization of Galois connections and we prove that an upper Galois adjoint of a monotone map between quantum suplattices exists if and only if the map is a morphism of quantum suplattices. Finally, we prove a quantum version of the Knaster-Tarski fixpoint theorem: the quantum set of fixpoints of a monotone endomap on a quantum suplattice form a quantum suplattice.

1 Introduction

A poset is called a *complete lattice* if it has all suprema, or equivalently, if it has all infima. However, a function between complete lattices that preserves all suprema does not necessarily preserve all infima. Hence, one can define several categories of complete lattices with different classes of morphisms. For instance, the class consisting of maps that preserve both all suprema and all infima, or the class of maps that only preserve all suprema. If we choose this latter class of morphisms, we typically call the objects of the resulting category **Sup** *suplattices* instead of complete lattices.

In this contribution, we introduce a noncommutative version of suplattices, which we call *quantum suplattices*. One of the reasons why we are interested in these objects is that they might lead to a notion of quantum topological spaces that allows for the quantization of topological spaces that are not locally compact Hausdorff, such as the Scott topology on a dcpo. Any topology of a topological space is in particular a complete lattice; the usual approach to quantum topological spaces are C^* -algebras, which can be regarded as noncommutative locally compact Hausdorff spaces.

The approach we take is the program of *discrete quantization* [11]. Here, *quantizing* some mathematical structure is understood as the operation of finding a noncommutative generalization or version of the structure. This can be done by internalizing the structure in a suitable category of operator algebras. For discrete quantization, this category is called **qRel**, which is equivalent to the category of von Neumann algebras isomorphic to some (possibly infinite) ℓ^∞ -sum of matrix algebras (such von Neumann algebras are also called *hereditarily atomic*) equipped with Weaver’s quantum relations [21]. The category **qRel** shares many properties with **Rel**. For instance, it is also dagger compact [1, 7, 9] and enriched over **Sup**. Since many mathematical structures can be described in terms of the dagger structure and the **Sup**-enrichment of **Rel**, this makes **qRel** a very suitable tool for quantization. Since hereditarily atomic von Neumann algebras have a discrete character, they can be regarded as noncommutative sets, which explains the name ‘discrete quantization’.

Regarding discrete quantization, one could argue that it is a disadvantage that we do not work in full generality with all von Neumann algebras. However, we see this lack of generality as a feature, not as a bug: the category of all von Neumann algebras and quantum relations is not compact, whereas **qRel** is. This is of huge importance for the theory of quantum suplattices, which relies heavily on **qRel** being compact. By definition, any matrix algebra $M_d(\mathbb{C})$, which is often used to represent a qudit, is an example of a hereditarily atomic von Neumann algebra. Since any tensor product of two matrix algebras is a hereditarily atomic von Neumann algebra, systems of multiple qudits can be represented by hereditarily atomic von Neumann algebras. Therefore, hereditarily atomic von Neumann algebras are sufficient for most practical applications in quantum information theory and in quantum computing. Recently, discrete quantization was applied successfully in the denotational semantics of quantum programming languages [13].

1.1 Related work

Discrete quantization can be regarded as a special case of quantization via quantum relations, which we distilled by Weaver [21] from his work with Kuperberg on quantum metric spaces [15]. The category **qRel** whose objects are called *quantum sets* was introduced by Kornell [12], who showed that this category is dagger compact. Moreover, in the same reference, he quantized functions by internalizing them in **qRel**, and showed that resulting category **qSet** of quantum sets and quantized functions is symmetric monoidal closed, complete, cocomplete, and dual to the category of hereditarily atomic von Neumann algebras and normal $*$ -homomorphisms. Furthermore, Kornell showed in [11] that **qRel** is equivalent to the category of hereditarily atomic von Neumann algebras and Weaver’s quantum relations, and introduced a logic with equality for **qRel**.

Quantum posets were already defined by Weaver in [21]. The properties of the category of quantum posets in the framework of discrete quantum mathematics were investigated in [14] by Kornell, Mislove and the second author. The same authors proceeded in [13] by quantizing cpos by means of discrete quantization. Traditionally, cpos are a class of posets that form the essential objects for the denotational semantics of programming languages. One would expect that the denotational semantics of quantum programming languages will require quantized cpos. The current state-of-the art quantum programming language is Proto-Quipper-M, which was introduced by Rios and Selinger [19], subsequently extended with recursive terms in [16] and then with recursive types in [17] by Mislove, Zamdzhiev and the second author. In [13], the quantized cpos were successfully used to construct sound and adequate denotational models for these extensions.

Finally, we mention the quantum graphs [21, Definition 2.6(d)][8], which recently attracted some attention [2, 20, 22, 18, 5, 6]. Quantum graphs can be described in the framework of discrete quantum quantization in a similar way as quantum posets. Just like a poset is a special kind of graph, a quantum poset is a special kind of quantum graph, and many concepts and techniques from quantum graphs carry over to quantum posets.

1.2 Overview of the paper

We start by giving a recap of quantum sets and quantum posets. Then we introduce monotone relations between quantum sets. These are of importance, because the ordinary down-set monad on the category **Pos** of ordinary posets can be obtained from an adjunction between **Pos** and the category **RelPos** of posets and monotone relations. We proceed by introducing the quantum down-set monad **qDwn** and explaining its connection with upper sets and the quantum power set. We then introduce a quantum generalization of

Galois connections, which we use to define quantum suplattices. We show that $\mathbf{qDwn}(\mathcal{X})$ is a quantum suplattice for any quantum poset \mathcal{X} . Together with the characterization of Galois connections between quantum posets (cf. Theorem 7.2), these are the only proofs we include, just to give a flavor of how to work with quantum sets. Furthermore, we sketch why the opposite of a quantum suplattice is also a quantum suplattice. Finally, we discuss enrichment over **Sup** and fixpoints of monotone endomaps between quantum suplattices.

Let us stress that although the quantized theorems we included in the present paper are almost verbatim copies of their classical versions, the proof of a quantized theorem is usually much more complicated than its classical counterpart.

2 Preliminaries

2.1 Quantum sets

The basic reference for this section is [12]. Here, the definition of **qRel** implicitly makes use of categorical constructions, which we choose to highlight. In order to do so, we first introduce the category **FdOS** whose objects are nonzero finite-dimensional Hilbert spaces. A morphism $A : X \rightarrow Y$ in **FdOS** is a *operator space*, that is a subspace of the vector space $L(X, Y)$ of linear operators $X \rightarrow Y$. We define composition of A with a morphism $B : Y \rightarrow Z$ in **FdOS** as the operator space $B \cdot A := \text{span}\{ba : a \in A, b \in B\}$. The identity morphism on X is the operator space $\mathbb{C}1_X := \{\lambda 1_X : \lambda \in \mathbb{C}\}$. Since the space $L(X, Y)$ is actually a finite-dimensional Hilbert space itself via the inner product $(a, b) \mapsto \text{Tr}(a^\dagger b)$, where $a^\dagger : Y \rightarrow X$ denotes the hermitian adjoint of $a \in L(X, Y)$, the homset **FdOS**(X, Y) becomes a complete modular ortholattice; the order on the homset is explicitly given by $A \leq B$ if and only if A is a subspace of B . Since composition in **FdOS**(X, Y) preserves suprema, **FdOS** is enriched over the category **Sup** of complete lattices and suprema-preserving functions; any such category is also called a *quantaloid*.

Products and coproducts in quantaloids coincide and are also called *sums*. Any quantaloid **Q** has a free sum-completion of **Q**, which can be described by the quantaloid $\text{Matr}(\mathbf{Q})$, whose objects are **Set**-indexed families of objects $(X_i)_{i \in I}$ of **Q**, and whose morphisms $R : (X_i)_{i \in I} \rightarrow (Y_j)_{j \in J}$ are ‘matrices’ whose (i, j) -component $R(i, j)$ is a **Q**-morphism $X_i \rightarrow Y_j$. Composition in $\text{Mat}(\mathbf{Q})$ is defined by matrix multiplication: we define $S \circ R$ for $S : (Y_j)_{j \in J} \rightarrow (Z_k)_{k \in K}$ by $(S \circ R)(i, k) = \bigvee_{j \in J} S(j, k) \cdot R(i, j)$ for each $i \in I$ and $k \in K$, where \cdot denotes the composition of morphisms in **Q**. The (i, i') -component of the identity morphism on an object $(X_i)_{i \in I}$ is the identity morphism of X_i if $i = i'$, and 0 otherwise. The order on the homsets of $\text{Matr}(\mathbf{Q})$ is defined componentwise. The matrix-construction as the free sum-completion of quantaloids was introduced in [10], and is a special case of a matrix-construction for more general bicategories as described in [3].

We now define **qRel** as the quantaloid $\text{Matr}(\mathbf{FdOS})$. Any object \mathcal{X} of **qRel** is called a *quantum set*; whose *atoms* are the Hilbert spaces in \mathcal{X} . A quantum set consisting of a single atom is called *atomic*.

For convenience, we denote the elements of the index set $\text{At}(\mathcal{X})$ of \mathcal{X} by the atoms of \mathcal{X} themselves, hence $\text{At}(\mathcal{X})$ can be interpret as the set of atoms of \mathcal{X} . Thus, in some sense \mathcal{X} is indexed by itself, just like ordinary sets can be regarded as indexed families indexed by themselves via the identity function. Since a quantum set \mathcal{X} is formally a indexed family, it does not have elements in the usual sense. We shall use the notation $X \in \mathcal{X}$ to express that X is an atom of \mathcal{X} . Thus, we have $X \in \mathcal{X}$ if and only if $X \in \text{At}(\mathcal{X})$. Conversely, to any ordinary set M consisting of finite-dimensional Hilbert spaces we can associate a unique quantum set $\mathcal{Q}M$ whose set of atoms $\text{At}(\mathcal{Q}M)$ consists of all Hilbert spaces H in M such that $\dim(H) \neq 0$.

Any morphism in **qRel** is called a *binary relation*. We emphasize that binary relations between quantum sets are not binary relations in the usual sense, i.e., subsets of the product of domain and codomain of the relation. However, binary relations between quantum sets turn out to be generalizations of binary relations between ordinary sets. Given our convention that the indices in the index set $\text{At}(\mathcal{X})$ of \mathcal{X} are chosen to be the atoms of \mathcal{X} itself, any binary relation $R : \mathcal{X} \rightarrow \mathcal{Y}$ between quantum sets is an assignment that to each atom $X \in \mathcal{X}$ and each atom Y of \mathcal{Y} assigns a subset $R(X, Y)$ of $L(X, Y)$. Given another binary relation $S : \mathcal{Y} \rightarrow \mathcal{Z}$, the (X, Z) -component of the composition $S \circ R$ is given by $(S \circ R)(X, Z) = \bigvee_{Y \in \mathcal{Y}} S(Y, Z) \cdot R(X, Y)$. The identity morphism on a quantum set \mathcal{X} is denoted by $I_{\mathcal{X}}$ and is given by $I_{\mathcal{X}}(X, X') = \mathbb{C}1_X$ if $X = X'$ and $I_{\mathcal{X}}(X, X') = 0$ otherwise. The quantaloid structure of **qRel** can be described explicitly as follows. For binary relation $R, S : \mathcal{X} \rightarrow \mathcal{Y}$ we have $R \leq S$ if and only if $R(X, Y) \leq S(X, Y)$ for each $X \in \mathcal{X}$ and each $Y \in \mathcal{Y}$. Equipped with this order, any homset of **qRel** becomes a complete lattice. The supremum $\bigvee_{i \in I} R_i$ of a collection $(R_i : i \in I)$ of relations $\mathcal{X} \rightarrow \mathcal{Y}$ is given by $(\bigvee_{i \in I} R_i)(X, Y) = \bigvee_{i \in I} R_i(X, Y)$ for each $X \in \mathcal{X}$ and each $Y \in \mathcal{Y}$, where the supremum in the right-hand side is taken in the complete lattice of subspaces of $L(X, Y)$.

We can generalize the following set-theoretic notions to the quantum setting:

- (1) A quantum set \mathcal{X} is *empty* if $\text{At}(\mathcal{X}) = \emptyset$;
- (2) A quantum set \mathcal{X} is a *subset* of a quantum set \mathcal{Y} if $\text{At}(\mathcal{X}) \subseteq \text{At}(\mathcal{Y})$, in which case we write $\mathcal{X} \subseteq \mathcal{Y}$.
- (3) The *cartesian product* $\mathcal{X} \times \mathcal{Y}$ of two quantum sets \mathcal{X} and \mathcal{Y} is defined by $\text{At}(\mathcal{X} \times \mathcal{Y}) = \{X \otimes Y : X \in \mathcal{X}, Y \in \mathcal{Y}\}$, where $X \otimes Y$ denotes the usual tensor product of the Hilbert spaces X and Y .

We denote the cartesian product of quantum sets by \times , because it is the noncommutative generalization of the usual product. However, it is not a categorical product in any of the categories that we will introduce below. It is not uncommon to use the notation \times for a non-categorical product: for instance, it is also used to denote the monoidal product in the category **Rel** of sets and binary relations.

To each ordinary set S we can assign a quantum set $'S$ whose atoms are one-dimensional Hilbert spaces that are in a one-to-one correspondence with elements of S . This correspondence can be made precise as follows. For each $s \in S$, we define $\mathbb{C}_s := \ell^2(\{s\})$ with the convention that $\mathbb{C}_s \neq \mathbb{C}_t$ if $s \neq t$. Then $\text{At}('S) = \{\mathbb{C}_s : s \in S\}$. Note that $'(S \times T)$ is isomorphic to $'S \times 'T$ as a quantum set.

It is well known that the category **FdHilb** of finite-dimensional Hilbert spaces and linear maps is a dagger compact category, where the dagger of a linear map a is given by taking its Hermitian adjoint a^\dagger .

Also **qRel** is a dagger compact category: for a relation $R : \mathcal{X} \rightarrow \mathcal{Y}$, we define $R^\dagger : \mathcal{Y} \rightarrow \mathcal{X}$ by $R^\dagger(Y, X) = \{a^\dagger : a \in R(X, Y)\}$ for each $X \in \mathcal{X}$ and each $Y \in \mathcal{Y}$. The cartesian product \times of quantum sets extends to a monoidal product that is defined on morphisms $R : \mathcal{X} \rightarrow \mathcal{Y}$ and $S : \mathcal{W} \rightarrow \mathcal{Z}$ by $(R \times S)(X \otimes W, Y \otimes Z) = R(X, Y) \otimes S(W, Z)$ for each $X \otimes W \in \mathcal{X} \times \mathcal{W}$ and each $Y \otimes Z \in \mathcal{Y} \times \mathcal{Z}$. The monoidal unit $\mathbf{1}$ is given by the quantum set consisting of a single one-dimensional atom, typically denoted by \mathbb{C} .

Let H and K be Hilbert spaces. For each linear operator $v \in L(H, K)$, write $v^* \in L(K^*, H^*)$ for the Banach space adjoint of v , defined by $v^*(\varphi) = \varphi \circ v$. For each subspace $V \leq L(H, K)$, write $V^* = \{v^* : v \in V\} \leq L(K^*, H^*)$. The *dual* of a quantum set \mathcal{X} is the quantum set \mathcal{X}^* determined by $\text{At}(\mathcal{X}^*) = \{v^* : v \in \mathcal{X}\}$. The *dual* of a binary relation R from \mathcal{X} to \mathcal{Y} is the binary relation R^* from \mathcal{Y}^* to \mathcal{X}^* defined by $R^*(Y^*, X^*) = R(X, Y)^*$. In **qRel**, the associator A , the unitors L and R , the symmetry S , the unit H and the counit E of the compact structure can be expressed in terms of the associator α , the unitors λ and ρ , the symmetry σ , and the unit η and counit ε of the compact structure of **FdHilb**. For instance, the nonzero components of $S_{\mathcal{X}\mathcal{Y}} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Y} \times \mathcal{X}$ is given by $S_{\mathcal{X}\mathcal{Y}}(X \otimes Y, Y \otimes X) = \mathbb{C}\sigma_{XY}$, and the nonzero components of $E_{\mathcal{X}} : \mathcal{X} \times \mathcal{X}^* \rightarrow \mathbf{1}$ are given by $E_{\mathcal{X}}(X \otimes X^*, \mathbb{C}) = \mathbb{C}\varepsilon_X$.

The assignment $S \mapsto 'S$ extends to a fully faithful functor $'(-): \mathbf{Rel} \rightarrow \mathbf{qRel}$, which is defined on ordinary binary relations $r: S \rightarrow T$ for each $s \in S$ and each $t \in T$ by $'(r)(\mathbb{C}_s, \mathbb{C}_t) = L(\mathbb{C}_s, \mathbb{C}_t)$ if $(s, t) \in r$, and $'(r)(\mathbb{C}_s, \mathbb{C}_t) = 0$ otherwise. Since $L(\mathbb{C}_s, \mathbb{C}_t)$ is one dimensional, it only has two subspaces, whence $'(-)$ is indeed fully faithful. Moreover, it preserves the dagger structure, and the inclusion order on homsets of \mathbf{Rel} .

It is easy to verify that a function $f: X \rightarrow Y$ between ordinary sets is a binary relation such that $f^\dagger \circ f \geq 1_X$ and $f \circ f^\dagger \leq 1_Y$, where f^\dagger is the opposite relation of f . Hence, a relation $F: \mathcal{X} \rightarrow \mathcal{Y}$ between quantum sets is called a *function* if it satisfies $F^\dagger \circ F \geq I_{\mathcal{X}}$ and $F \circ F^\dagger \leq I_{\mathcal{Y}}$. Examples of functions are the associator, unitors, and symmetry of \mathbf{qRel} . Another example of a function is provided by subsets \mathcal{Y} of quantum sets \mathcal{X} , for which there is an *inclusion function* $J_{\mathcal{Y}}^{\mathcal{X}}$ defined for each $Y \in \mathcal{Y}$ and each $X \in \mathcal{X}$ by $J_{\mathcal{Y}}^{\mathcal{X}}(Y, X) = \mathbb{C}1_Y$ if $Y = X$ and $J_{\mathcal{Y}}^{\mathcal{X}}(Y, X) = 0$ otherwise. If it is clear that \mathcal{X} is the ambient quantum set, we often write $J_{\mathcal{Y}}$ instead of $J_{\mathcal{Y}}^{\mathcal{X}}$.

Given a binary relation $R: \mathcal{X} \rightarrow \mathcal{Y}$ and subsets $\mathcal{Z} \subseteq \mathcal{X}$ and $\mathcal{W} \subseteq \mathcal{Y}$, we define the *restriction* $R|_{\mathcal{Z}}$ of R to \mathcal{Z} as the relation $R \circ J_{\mathcal{Z}}^{\mathcal{X}}$. The *corestriction* $R|_{\mathcal{W}}$ of R to \mathcal{W} is defined as the relation $(J_{\mathcal{W}}^{\mathcal{Y}})^\dagger \circ R$. We have $(R|_{\mathcal{Z}})|_{\mathcal{W}} = (R|_{\mathcal{W}})|_{\mathcal{Z}}$, which we denote as $R|_{\mathcal{Z}}^{\mathcal{W}}$.

The wide subcategory of \mathbf{qRel} of functions is denoted by \mathbf{qSet} , which is complete, cocomplete and symmetric monoidal closed with respect to the monoidal product \times . The monoidal unit, associator, unitors and symmetry are the same as for \mathbf{qRel} .

A function $F: \mathcal{X} \rightarrow \mathcal{Y}$ is called *injective* if $F^\dagger \circ F = I_{\mathcal{X}}$ and *surjective* if $F \circ F^\dagger = I_{\mathcal{Y}}$. Any inclusion function is an injective map. The injective and surjective functions are precisely the respective monomorphisms and epimorphisms of \mathbf{qSet} . Functions that are both injective and surjective are called *bijective*, and are precisely the isomorphisms of \mathbf{qSet} . The *range* of a function $F: \mathcal{X} \rightarrow \mathcal{Y}$ is the quantum set $\text{ran } F$ specified by $\text{At}(\text{ran } F) = \{Y \in \mathcal{Y} : F(X, Y) \neq 0 \text{ for some } X \in \mathcal{X}\}$. We have $F = J_{\text{ran } F} \circ \bar{F}$ for some unique surjective function $\bar{F}: \mathcal{X} \rightarrow \text{ran } F$, which is defined by $\bar{F}(X, Y) = F(X, Y)$ for each $X \in \mathcal{X}$ and each $Y \in \text{ran } F$. It follows that F is surjective if and only if $\text{ran } F = \mathcal{Y}$.

The functor $'(-): \mathbf{Rel} \rightarrow \mathbf{qRel}$ restricts and corestricts to a fully faithful functor $'(-): \mathbf{Set} \rightarrow \mathbf{qSet}$. Furthermore, if we denote the category of von Neumann algebras and normal $*$ -homomorphisms by \mathbf{WStar} , then there is a fully faithful functor $\ell^\infty: \mathbf{qSet} \rightarrow \mathbf{WStar}^{\text{op}}$ that on objects is defined by $\mathcal{X} \mapsto \bigoplus_{X \in \mathcal{X}} L(X)$. The essential image of this functor is the category of *hereditarily atomic* von Neumann algebras, i.e., von Neumann algebras that are isomorphic to some (possibly infinite) ℓ^∞ -sum of matrix algebras. Also \mathbf{qRel} can be shown to equivalent to a category of operator algebras [11], namely the category of hereditarily atomic von Neumann algebras and Weaver's quantum relations [21].

2.2 Quantum posets

The basic reference for this section is [14]. Let \mathcal{X} be a quantum set. Then we call a binary relation $R: \mathcal{X} \rightarrow \mathcal{X}$ *reflexive* if $I_{\mathcal{X}} \leq R$, *transitive* if $R \circ R \leq R$, and *antisymmetric* if $R \wedge R^\dagger \leq I_{\mathcal{X}}$. A pair (\mathcal{X}, \preceq) consisting of a quantum set \mathcal{X} and a reflexive, transitive and antisymmetric relation \preceq on \mathcal{X} is called a *quantum poset*. The relation \preceq is called an *order*. In order to improve the readability of expressions and calculations, we sometimes write parentheses around \preceq , so we write (\preceq) .

Example 2.1. Let \mathcal{X} be a quantum set. Then $I_{\mathcal{X}}$ is a quantum order on \mathcal{X} , which we call the *trivial order*.

Example 2.2. Let \mathcal{H} be the quantum set consisting of a single two-dimensional atom H . A 'non-classical' order on \mathcal{H} is given by the relation \preceq on \mathcal{H} specified by $\preceq(H, H) = \mathbb{C} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mathbb{C} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$. Since \mathcal{H} has only one atom H , \preceq is determined by $\preceq(H, H)$. Thus, (\mathcal{H}, \preceq) is a quantum poset.

Let (\mathcal{X}, \preceq) be a quantum poset. The relation \preceq^\dagger is also an order, and is called the *opposite* order on \mathcal{X} . We write $(\mathcal{X}, \preceq)^{\text{op}} = (\mathcal{X}, \preceq^\dagger)$, which is called the *opposite* quantum poset of (\mathcal{X}, \preceq) . Often, we just write \mathcal{X} instead of (\mathcal{X}, \preceq) and \mathcal{X}^{op} instead of $(\mathcal{X}, \preceq^\dagger)$.

Example 2.3. Let (\mathcal{H}, \preceq) be the quantum poset of the previous example. Then \preceq^\dagger is specified by $\preceq^\dagger(H, H) = \mathbb{C} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mathbb{C} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$.

Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets. Then we say that a function $F: \mathcal{X} \rightarrow \mathcal{Y}$ is *monotone* if $F \circ (\preceq_{\mathcal{X}}) \leq (\preceq_{\mathcal{Y}}) \circ F$. Under the composition of functions between quantum sets, quantum posets and monotone functions form a category, which we call **qPos**, which is complete, cocomplete and monoidal closed under the monoidal product that is defined by $(\mathcal{X}, \preceq_{\mathcal{X}}) \times (\mathcal{Y}, \preceq_{\mathcal{Y}}) = (\mathcal{X} \times \mathcal{Y}, \preceq_{\mathcal{X}} \times \preceq_{\mathcal{Y}})$. The monoidal unit is given by $(\mathbf{1}, I_{\mathbf{1}})$. The components of the associator, unitors and symmetry are the components of the respective associator, unitors and symmetry of the underlying quantum sets. We denote the evaluation morphism of **qPos** by $\text{Eval}_{\sqsubseteq}$, and the internal hom by $[\cdot, \cdot]_{\sqsubseteq}$. We call the isomorphisms of **qPos** *order isomorphisms*.

Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets. Then a monotone map $F: \mathcal{X} \rightarrow \mathcal{Y}$ is called an *order embedding* if $(\preceq_{\mathcal{X}}) = F^\dagger \circ (\preceq_{\mathcal{Y}}) \circ F$. The surjective order embeddings are precisely the order isomorphisms. A subset of a quantum poset (\mathcal{Y}, \preceq) consists of a subset \mathcal{X} of \mathcal{Y} equipped with the order $\preceq|_{\mathcal{X}} := J_{\mathcal{X}}^\dagger \circ \preceq \circ J_{\mathcal{X}}$, to which we refer as the *induced* order on \mathcal{X} . It follows that $J_{\mathcal{X}}: (\mathcal{X}, \preceq|_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq)$ is an order embedding.

Given a monotone map $F: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ between quantum posets, if we equip $\text{ran } F \subseteq \mathcal{Y}$ with the relative order, then the unique surjective function $\bar{F}: \mathcal{X} \rightarrow \text{ran } F$ such that $F = J_{\text{ran } F} \circ \bar{F}$ is monotone. Hence, every monotone map can be written as the composition of a monotone surjective map and an order embedding.

If (S, \sqsubseteq) is an ordinary poset, then $(\text{'}S, \text{'}\sqsubseteq)$ is a quantum poset, and vice versa; for example, the trivial order \sqsubseteq on S corresponds to $(\text{'}\sqsubseteq) = I_S$, the trivial order on $\text{'}S$. Moreover, a monotone map f between ordinary posets gives rise to a monotone function $\text{'}f$ between the associated quantum posets, and vice versa. It follows that $\text{'}(-)$ extends to a fully faithful functor **Pos** \rightarrow **qPos** defined on objects by $\text{'}(S, \sqsubseteq) = (\text{'}S, \text{'}\sqsubseteq)$. If $\mathbf{2}$ denotes the two-point set $\{0, 1\}$, we write $\mathbf{2} = \text{'}2$. If we equip $\mathbf{2}$ with the order \sqsubseteq defined by $0 \sqsubseteq 1$, then we write $\preceq_{\mathbf{2}} = \text{'}\sqsubseteq$. Hence, $(\mathbf{2}, \preceq_{\mathbf{2}}) = (\text{'}2, \text{'}\sqsubseteq)$.

Given a quantum set \mathcal{X} and a quantum poset (\mathcal{Y}, \preceq) , and given two functions $F, G: \mathcal{X} \rightarrow \mathcal{Y}$, we define $F \sqsubseteq_{\mathcal{Y}} G$ if $G \leq (\preceq) \circ F$. This defines an order on **qSet** $(\mathcal{X}, \mathcal{Y})$ which is the quantum equivalent of the pointwise order of functions. We sometimes write $F \sqsubseteq G$ instead of $F \sqsubseteq_{\mathcal{Y}} G$.

3 Monotone relations

Let (X, \sqsubseteq) and (Y, \sqsubseteq) be posets. A binary relation $v: X \rightarrow Y$ is called a *monotone relation* [4] if $(x', y') \in v$, $x' \sqsubseteq x$ and $y \sqsubseteq y'$ implies $(x, y) \in v$. Under the usual composition of binary relations, posets and monotone relations form a category **RelPos**, where the identity monotone relation $1_{(X, \sqsubseteq)}$ on the poset (X, \sqsubseteq) is the binary relation \sqsubseteq . One can show that there are bijections between monotone relations $X \rightarrow Y$, monotone functions $X \times Y^{\text{op}} \rightarrow \mathbf{2}$, and monotone functions $X \rightarrow \text{Dwn}(Y)$, where $\mathbf{2}$ is the two-point poset $0, 1$ ordered by $0 \sqsubseteq 1$, and $\text{Dwn}(Y)$ is the poset of down-sets of Y ordered by inclusion. In fact, the assignment $Y \mapsto \text{Dwn}(Y)$ can be made into an endofunctor **Dwn** on **Pos** that is a monad, and whose Kleisli category is isomorphic to **RelPos**, reflected in the previous remark that a monotone relation $X \rightarrow Y$ corresponds to a monotone map $X \rightarrow \text{Dwn}(Y)$, i.e., a Kleisli map. The importance of this monad lies in the fact that its Eilenberg-Moore category is precisely the category **Sup** of suplattices.

A way to see that \mathbf{Dwn} is the underlying endofunctor of a monad is the following. Just like we can embed \mathbf{Set} into \mathbf{Rel} , we have an embedding $(-)_\diamond: \mathbf{Pos} \rightarrow \mathbf{RelPos}$ that is the identity on objects, and that sends any monotone function $f: (X, \sqsubseteq) \rightarrow (Y, \sqsubseteq)$ to the monotone relation $f_\diamond = \{(x, y) : y \sqsubseteq f(x)\}$.

Moreover, just like the covariant power set functor extends to a functor $\mathbf{Rel} \rightarrow \mathbf{Set}$ that is the right adjoint of the embedding $\mathbf{Set} \rightarrow \mathbf{Rel}$, the assignment $X \mapsto \mathbf{Dwn}(X)$ extends to a functor $\mathbf{RelPos} \rightarrow \mathbf{Pos}$ that is the right adjoint of $(-)_\diamond$. The monad induced by this adjunction is precisely the down-set monad on \mathbf{Pos} . Hence, in order to define quantum suplattices, we will have to find the quantum generalization of the down-set monad, and in order to find this quantum down-set monad, we have to find a quantum generalization of monotone relations.

Definition 3.1. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets. A binary relation $V: \mathcal{X} \rightarrow \mathcal{Y}$ is called a monotone relation $(\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ if it satisfies one of the following two equivalent conditions (hence both):

- (1) $(\preceq_{\mathcal{Y}}) \circ V \leq V$ and $V \circ (\preceq_{\mathcal{X}}) \leq V$.
- (2) $(\preceq_{\mathcal{Y}}) \circ V = V = V \circ (\preceq_{\mathcal{X}})$.

The equivalence of both conditions follows from the reflexivity of orders.

Lemma 3.2. Let $V: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ and $W: (\mathcal{Y}, \preceq_{\mathcal{Y}}) \rightarrow (\mathcal{Z}, \preceq_{\mathcal{Z}})$ be monotone relations between quantum posets. Then $W \circ V: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Z}, \preceq_{\mathcal{Z}})$ is a monotone relation.

Lemma 3.3. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ be a quantum poset. Then $\preceq_{\mathcal{X}}: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{X}, \preceq_{\mathcal{X}})$ is a monotone relation such that for each quantum poset $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ and all monotone relations $V: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ and $W: (\mathcal{Y}, \preceq_{\mathcal{Y}}) \rightarrow (\mathcal{X}, \preceq_{\mathcal{X}})$ we have $V \circ (\preceq_{\mathcal{X}}) = V$ and $(\preceq_{\mathcal{X}}) \circ W = W$.

It follows from the previous two lemmas that the following definition is sound:

Definition 3.4. We define the category of quantum posets and monotone relations by $\mathbf{qRelPos}$. The identity monotone relation on a quantum poset $(\mathcal{X}, \preceq_{\mathcal{X}})$ is $\preceq_{\mathcal{X}}$, which we often denote by $I_{(\mathcal{X}, \preceq_{\mathcal{X}})}$.

Lemma 3.5. There is a fully faithful functor $\ulcorner(-): \mathbf{RelPos} \rightarrow \mathbf{qRelPos}$ that sends any poset (S, \sqsubseteq) to $(\ulcorner S, \ulcorner \sqsubseteq)$ and any monotone relation $v: (S, \sqsubseteq) \rightarrow (T, \sqsubseteq)$ to $\ulcorner v$.

Lemma 3.6. There is a faithful functor $(-)_\diamond: \mathbf{qPos} \rightarrow \mathbf{qRelPos}$ which is the identity on objects, and which acts on monotone maps $F: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ by $F_\diamond := (\preceq_{\mathcal{Y}}) \circ F$.

The functor in the previous lemma is an extension of the functor $(-)_\diamond: \mathbf{Pos} \rightarrow \mathbf{RelPos}$ mentioned above, in the sense that $\mathbf{Pos} \xrightarrow{\ulcorner(-)} \mathbf{qPos} \xrightarrow{(-)_\diamond} \mathbf{qRelPos}$ equals $\mathbf{Pos} \xrightarrow{(-)_\diamond} \mathbf{RelPos} \xrightarrow{\ulcorner(-)} \mathbf{qRelPos}$.

Definition 3.7. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ be a quantum poset. Then we define $(\mathcal{X}, \preceq_{\mathcal{X}})^*$ to be the quantum poset $(\mathcal{X}^*, \preceq_{\mathcal{X}^*})$. Sometimes, we write \mathcal{X}^* instead of $(\mathcal{X}, \preceq_{\mathcal{X}})^*$.

Since the operation of taking daggers in dagger compact categories commutes with the operation of taking duals, we obtain the following lemma:

Lemma 3.8. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ be a quantum poset. Then $(\mathcal{X}^*)^{\text{op}} = (\mathcal{X}^{\text{op}})^*$.

Theorem 3.9. The category $\mathbf{qRelPos}$ is compact closed: for each quantum poset $(\mathcal{X}, \preceq_{\mathcal{X}})$, the unit $H_{(\mathcal{X}, \preceq_{\mathcal{X}})}: (\mathbf{1}, I_1) \rightarrow (\mathcal{X}, \preceq_{\mathcal{X}})^* \times (\mathcal{X}, \preceq_{\mathcal{X}})$ and the counit $E_{(\mathcal{X}, \preceq_{\mathcal{X}})}: (\mathcal{X}, \preceq_{\mathcal{X}}) \times (\mathcal{X}, \preceq_{\mathcal{X}})^* \rightarrow (\mathbf{1}, I_1)$ are given by $(\preceq_{\mathcal{X}}^* \times \preceq_{\mathcal{X}}) \circ H_{\mathcal{X}}$ and $E_{\mathcal{X}} \circ (\preceq_{\mathcal{X}} \times \preceq_{\mathcal{X}}^*)$, respectively, where $H_{\mathcal{X}}$ and $E_{\mathcal{X}}$ denote the usual unit and counit of \mathbf{qRel} . The associator, unitors and symmetry are obtained by applying the functor $(-)_\diamond$ to the usual associator, unitors and symmetry in \mathbf{qPos} .

4 The quantum down-set monad

In this section, we introduce the quantum down-set monad. Its construction shares similarities with the construction of the quantum power set in [14]. This construction yields a quantum poset, but the construction of this order seems to be a bit ad hoc. The framework of monotone relations seems to be more appropriate for the construction of an ordered object. We will see that the quantum down-set monad by means of monotone relations is ordered in a natural way. When we apply the monad to a trivially ordered quantum set, then we obtain the quantum power set of this quantum set.

Definition 4.1. We define the quantum poset $\mathbf{qDwn}(\mathcal{X}, \mathfrak{A})$ of down-sets of a quantum poset $(\mathcal{X}, \mathfrak{A})$ to be the internal hom in \mathbf{qPos} from $(\mathcal{X}, \mathfrak{A})^*$ to $(\mathbf{2}, \mathfrak{A}_2)$, i.e., $\mathbf{qDwn}(\mathcal{X}, \mathfrak{A}) := [(\mathcal{X}, \mathfrak{A})^*, (\mathbf{2}, \mathfrak{A}_2)]_{\square}$. We will denote its underlying quantum set by $\mathcal{D}(\mathcal{X}, \mathfrak{A})$. The order on $\mathcal{D}(\mathcal{X}, \mathfrak{A})$ is denoted by $\subseteq_{(\mathcal{X}, \mathfrak{A})}$, so $\mathbf{qDwn}(\mathcal{X}, \mathfrak{A}) = (\mathcal{D}(\mathcal{X}, \mathfrak{A}), \subseteq_{(\mathcal{X}, \mathfrak{A})})$.

Note that the order \subseteq on $\mathbf{qDwn}(\mathcal{X})$ is a boldface symbol to distinguish it from the inclusion order \subseteq between ordinary sets. We will prove that the assignment of objects $(\mathcal{X}, \mathfrak{A}) \mapsto \mathbf{qDwn}(\mathcal{X}, \mathfrak{A})$ extends to a monad on \mathbf{qPos} by showing that the functor $(-)_\diamond: \mathbf{qPos} \rightarrow \mathbf{qRelPos}$ has a right adjoint; the monad is then induced by this adjunction. The right adjoint also sends objects $(\mathcal{X}, \mathfrak{A})$ to $\mathbf{qDwn}(\mathcal{X}, \mathfrak{A})$. Nevertheless, it is useful to make a distinction in the notation between monad and right adjoint, hence we will denote the right adjoint by \mathbf{qDwn}' . The first step of showing the existence of \mathbf{qDwn} is the following lemma, for which we note that we have embeddings $\mathbf{1} \rightarrow \mathbf{2}$ which map $*$ in $\mathbf{1}$ to either $0 \in \mathbf{2}$ or $1 \in \mathbf{2}$. We denote the respective maps by 0 and 1 as well. As a consequence, we have functions $\mathbf{0}, \mathbf{1}: \mathbf{1} \rightarrow \mathbf{2}$.

Lemma 4.2. Let $(\mathcal{X}, \mathfrak{A}_{\mathcal{X}})$ be a quantum poset. Then the bijection $\mathbf{qSet}(\mathcal{X}, \mathbf{2}) \rightarrow \mathbf{qRel}(\mathcal{X}, \mathbf{1})$, $F \mapsto \mathbf{1}^\dagger \circ F$ in [12, Theorem B.8] restricts and corestricts to a bijection

$$\mathbf{qPos}((\mathcal{X}, \mathfrak{A}_{\mathcal{X}}), (\mathbf{2}, \mathfrak{A}_2)) \rightarrow \mathbf{qRelPos}((\mathcal{X}, \mathfrak{A}_{\mathcal{X}}), (\mathbf{1}, I_1)).$$

The counit of the adjunction that yields the ordinary down-set monad is the inverse membership relation \ni . Lemma 4.2 assures the existence of the quantum equivalent of this counit, which we will denote with a boldface symbol \ni .

Lemma 4.3. For any quantum poset $(\mathcal{X}, \mathfrak{A})$ there is a unique monotone relation $\ni_{(\mathcal{X}, \mathfrak{A})}: \mathbf{qDwn}(\mathcal{X}, \mathfrak{A}) \rightarrow (\mathcal{X}, \mathfrak{A})$ such that $\mathbf{1}^\dagger \circ \text{Eval}_{\square} = E_{(\mathcal{X}, \mathfrak{A})} \circ (\ni_{(\mathcal{X}, \mathfrak{A})} \times I_{(\mathcal{X}, \mathfrak{A})^*})$.

Theorem 4.4. There is a functor $\mathbf{qDwn}': \mathbf{qRelPos} \rightarrow \mathbf{qPos}$ whose action on objects is given by $(\mathcal{X}, \mathfrak{A}) \mapsto \mathcal{D}(\mathcal{X}, \mathfrak{A})$, and which is right adjoint to the functor $(-)_\diamond: \mathbf{qPos} \rightarrow \mathbf{qRelPos}$. The $(\mathcal{X}, \mathfrak{A})$ -component of the counit \ni of this adjunction is the monotone relation constructed in Lemma 4.3. The unit of the adjunction is denoted by $\downarrow\{\cdot\}$. Its $(\mathcal{X}, \mathfrak{A})$ -component is an order embedding that is the unique monotone function $(\mathcal{X}, \mathfrak{A}) \rightarrow \mathcal{D}(\mathcal{X}, \mathfrak{A})$ such that $\ni_{(\mathcal{X}, \mathfrak{A})} \circ \downarrow\{\cdot\}_{(\mathcal{X}, \mathfrak{A})} = I_{(\mathcal{X}, \mathfrak{A})}$.

Definition 4.5. We define the quantum down-set monad \mathbf{qDwn} to be the monad induced by the adjunction $(-)_\diamond \dashv \mathbf{qDwn}'$, so $\mathbf{qDwn} = \mathbf{qDwn}' \circ (-)_\diamond$. We denote its multiplication by \mathbf{U} and its unit by $\downarrow\{\cdot\}$.

Note that the multiplication \mathbf{U} is a boldfaced version of the usual union \cup of ordinary sets.

5 Opposite quantum posets and upper sets

Let X be an ordinary poset. Then the complementation operator provides a bijection between the set $D(X)$ of down sets of X and the set $U(X)$ of upper sets of X . Both $D(X)$ and $U(X)$ can be extended

to endofunctors on \mathbf{Pos} , for which we have to order the former by inclusion and the latter by containment. Then, writing $\text{Dwn}(X) = (D(X), \subseteq)$ and $\text{Up}(X) = (U(X), \supseteq)$, the bijection extends to an order isomorphism $\text{Dwn}(X) \rightarrow \text{Up}(X)$. In the quantum world, we can obtain a similar order isomorphism by showing that we can also construct a different right adjoint of $(-)_\diamond$ in terms of upper sets, which, by the uniqueness of right adjoints up to natural isomorphism, should be naturally isomorphic to qDwn' . This natural isomorphism is precisely the operation of taking complements. Before we construct this right adjoint in terms of upper sets, we first have to extend the operation of taking opposite quantum posets to an endofunctor on \mathbf{qPos} .

Lemma 5.1. *There is an endofunctor $(-)^{\text{op}}: \mathbf{qPos} \rightarrow \mathbf{qPos}$, defined on objects by $(\mathcal{X}, \preceq_{\mathcal{X}}) \mapsto (\mathcal{X}, \succ_{\mathcal{X}})$ and which maps any monotone map $F: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \preceq_{\mathcal{Y}})$ to the monotone map $F: (\mathcal{X}, \succ_{\mathcal{X}}) \rightarrow (\mathcal{Y}, \succ_{\mathcal{Y}})$. This functor $(-)^{\text{op}}$ is involutory, i.e., $(-)^{\text{opop}} = 1_{\mathbf{qPos}}$, hence an isomorphism of categories.*

Proposition 5.2. *Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets. Then $[\mathcal{X}^{\text{op}}, \mathcal{Y}^{\text{op}}]_{\square} = [\mathcal{X}, \mathcal{Y}]_{\square}^{\text{op}}$.*

Definition 5.3. *Let (\mathcal{X}, \preceq) be a quantum poset. Then we define the quantum poset of upsets of (\mathcal{X}, R) as the quantum poset $\text{qUp}(\mathcal{X}, \preceq) := [(\mathcal{X}, \preceq)^*, (\mathbf{2}, \succ_{\mathbf{2}})]_{\square}$. We denote its underlying quantum set by $\mathcal{U}(\mathcal{X}, \preceq)$.*

The previous proposition yields $\text{qUp}(\mathcal{X}, \preceq) = [\mathcal{X}^*, \mathbf{2}^{\text{op}}]_{\square} = [(\mathcal{X}^{\text{op}})^*, \mathbf{2}]_{\square}^{\text{op}} = \text{qDwn}(\mathcal{X}, \succ)^{\text{op}}$, whence $\mathcal{U}(\mathcal{X}, \preceq) = \mathcal{D}(\mathcal{X}, \succ)$. The other right adjoint is obtained by constructing a different counit, namely the inverse non-membership relation. This is done by taking $\mathbf{2}^{\text{op}}$ instead of $\mathbf{2}$ and $'0^{\dagger} \circ F$ instead of $'1^{\dagger} \circ F$ in Lemma 4.2: Then given a quantum poset (\mathcal{X}, \preceq) , we have that $\mathfrak{F}_{(\mathcal{X}, \preceq)}$ is the unique monotone relation $\text{qUp}(\mathcal{X}, \preceq) \rightarrow (\mathcal{X}, \preceq)$ such that $'0^{\dagger} \circ \text{Eval}_{\square} = E_{(\mathcal{X}, \preceq)} \circ (\mathfrak{F}_{(\mathcal{X}, \preceq)} \times I_{(\mathcal{X}, \preceq)^*})$.

Theorem 5.4. *There is a functor $\text{qUp}': \mathbf{qRelPos} \rightarrow \mathbf{qPos}$ whose action on objects is given by $(\mathcal{X}, \preceq) \mapsto \text{qUp}(\mathcal{X}, \preceq)$ that is right adjoint to the functor $(-)_\diamond: \mathbf{qPos} \rightarrow \mathbf{qRelPos}$. Its counit is denoted by \mathfrak{F} , and its (\mathcal{X}, \preceq) -component of the counit is the monotone relation $\mathfrak{F}_{(\mathcal{X}, \preceq)}$.*

Definition 5.5. *The monad $\text{qUp}' \circ (-)_\diamond$ on \mathbf{qPos} is denoted by qUp . Note that its action on objects agrees with the assignment $(\mathcal{X}, \preceq) \mapsto \text{qUp}(\mathcal{X}, \preceq)$.*

Corollary 5.6. *There is a natural isomorphism $C: \text{qDwn}' \rightarrow \text{qUp}'$ between functors $\mathbf{qRelPos} \rightarrow \mathbf{qPos}$ that induces a natural isomorphism between the endofunctors $\text{qDwn} \rightarrow \text{qUp}$ on \mathbf{qPos} that we also denote by C . In particular, for a quantum poset (\mathcal{X}, \preceq) , the (\mathcal{X}, \preceq) -component of C is an order isomorphism $C_{(\mathcal{X}, \preceq)}: \text{qDwn}(\mathcal{X}, \preceq) \rightarrow \text{qUp}(\mathcal{X}, \preceq)$ that is natural in (\mathcal{X}, \preceq) such that $\mathfrak{D}_{(\mathcal{X}, \preceq)} \circ C_{(\mathcal{X}, \preceq)} = \mathfrak{F}_{(\mathcal{X}, \preceq)}$.*

6 The quantum power set

In [14], the quantum power set monad \mathcal{P} was introduced. We sketch how to derive this monad from the quantum down-set monad. We have a clear categorical embedding $\mathbf{qSet} \rightarrow \mathbf{qPos}$ acting on objects by $\mathcal{X} \mapsto (\mathcal{X}, I_{\mathcal{X}})$, which is left adjoint to a forgetful functor acting on objects by $(\mathcal{X}, \preceq) \mapsto \mathcal{X}$. We can now define the quantum power set $\mathcal{P}(\mathcal{X})$ of a quantum poset \mathcal{X} as the quantum set $\mathcal{D}(\mathcal{X}, I_{\mathcal{X}})$, which can be made into a functor by composing \mathcal{D} with the embedding of \mathbf{qSet} into \mathbf{qPos} . By playing with the composition of adjunctions, we can equip \mathcal{P} with the structure of a monad such as in [14]. The ordered quantum power set $\text{qPow}(\mathcal{X})$ can be obtained as $\text{qDwn}(\mathcal{X}, I_{\mathcal{X}})$. We note that $\mathcal{P}(\mathcal{X})$ is also the underlying quantum set of $\text{qUp}(\mathcal{X}, I_{\mathcal{X}})$. The natural isomorphism C from Corollary 5.6 now yields a bijection $C_{(\mathcal{X}, I_{\mathcal{X}})}: \mathcal{P}(\mathcal{X}) \rightarrow \mathcal{P}(\mathcal{X})$ that is an order isomorphism $\text{qPow}(\mathcal{X}) \rightarrow \text{qPow}(\mathcal{X})^{\text{op}}$, and which can be regarded as the quantum analog of the complement operator on the power set. For simplicity, we write $C_{\mathcal{X}}$ instead of $C_{(\mathcal{X}, I_{\mathcal{X}})}$.

7 Galois connections

Definition 7.1. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets and let $F: \mathcal{X} \rightarrow \mathcal{Y}$ and $G: \mathcal{Y} \rightarrow \mathcal{X}$ be functions. If $(\preceq_{\mathcal{Y}}) \circ F = G^{\dagger} \circ (\preceq_{\mathcal{X}})$, we say that (F, G) forms a Galois connection, or that F is the lower adjoint of G , or that G is the upper adjoint of F .

This definition is a generalization of the usual definition of a Galois connection between ordinary posets: the $(-)$ functor maps ordinary Galois connections to Galois connections in the sense of the definition above. Similarly as in the classical case, the lower adjoint in a Galois connection determines the upper adjoint and *vice versa*. The next theorem provides alternative characterizations of Galois connections, which might look more familiar to the classical case:

Theorem 7.2. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets and let $F: \mathcal{X} \rightarrow \mathcal{Y}$ and $G: \mathcal{Y} \rightarrow \mathcal{X}$ be monotone functions. Then the following statements are equivalent:

- (1) F is the lower adjoint of $G: \mathcal{Y} \rightarrow \mathcal{X}$.
- (2) $F \circ K \sqsubseteq_{\mathcal{Y}} M \iff K \sqsubseteq_{\mathcal{X}} G \circ M$ for any quantum set \mathcal{Z} and functions $K: \mathcal{Z} \rightarrow \mathcal{X}$ and $M: \mathcal{Z} \rightarrow \mathcal{Y}$.
- (3) We have $I_{\mathcal{X}} \sqsubseteq_{\mathcal{X}} G \circ F$ and $F \circ G \sqsubseteq_{\mathcal{Y}} I_{\mathcal{Y}}$.

Proof. We start by showing that (1) implies (2). So assume that F is the lower adjoint of G , so $(\preceq_{\mathcal{Y}}) \circ F = G^{\dagger} \circ (\preceq_{\mathcal{X}})$. Let \mathcal{Z} be a quantum set and let $K: \mathcal{Z} \rightarrow \mathcal{X}$ and $M: \mathcal{Z} \rightarrow \mathcal{Y}$ be functions. Assume $F \circ K \sqsubseteq_{\mathcal{Y}} M$. By definition of $\sqsubseteq_{\mathcal{Y}}$, we have $M \leq (\preceq_{\mathcal{Y}}) \circ F \circ K$, so $M \leq G^{\dagger} \circ \preceq_{\mathcal{X}} \circ K$, hence $G \circ M \leq G \circ G^{\dagger} \circ \preceq_{\mathcal{X}} \circ K \leq (\preceq_{\mathcal{X}}) \circ K$ since G is a function. Hence $K \sqsubseteq_{\mathcal{X}} G \circ M$.

Conversely if $K \sqsubseteq_{\mathcal{X}} G \circ M$, we have $G \circ M \leq (\preceq_{\mathcal{X}}) \circ K$ by definition of a function, $M \leq G^{\dagger} \circ G \circ M \leq G^{\dagger} \circ \preceq_{\mathcal{X}} \circ K = (\preceq_{\mathcal{Y}}) \circ F \circ K$, so $F \circ K \sqsubseteq_{\mathcal{Y}} M$.

We show that (2) implies (3). Since $F \sqsubseteq_{\mathcal{Y}} F$, condition (2) yields $I_{\mathcal{X}} \sqsubseteq_{\mathcal{X}} G \circ F$ if we choose $\mathcal{Z} = \mathcal{X}$, $K = I_{\mathcal{X}}$ and $M = F$. Since $G \sqsubseteq_{\mathcal{X}} G$, we obtain $F \circ G \sqsubseteq_{\mathcal{Y}} I_{\mathcal{Y}}$ by choosing $\mathcal{Z} = \mathcal{Y}$, $K = G$ and $M = I_{\mathcal{Y}}$.

We proceed by showing that (3) implies (1). Since $I_{\mathcal{X}} \sqsubseteq_{\mathcal{X}} G \circ F$, we have $G \circ F \leq (\preceq_{\mathcal{X}})$. Then $F \leq G^{\dagger} \circ G \circ F \leq G^{\dagger} \circ (\preceq_{\mathcal{X}})$. By Lemma 5.1, $G: \mathcal{Y}^{\text{op}} \rightarrow \mathcal{X}^{\text{op}}$ is also monotone, so $G \circ (\preceq_{\mathcal{Y}}) \leq (\preceq_{\mathcal{X}}) \circ G$, which implies $(\preceq_{\mathcal{Y}}) \circ G^{\dagger} \leq G^{\dagger} \circ (\preceq_{\mathcal{X}})$ after taking daggers. Hence, $(\preceq_{\mathcal{Y}}) \circ F \leq (\preceq_{\mathcal{Y}}) \circ G^{\dagger} \circ \preceq_{\mathcal{X}} \leq G^{\dagger} \circ (\preceq_{\mathcal{X}} \circ \preceq_{\mathcal{X}}) = G^{\dagger} \circ (\preceq_{\mathcal{X}})$.

Since $F \circ G \sqsubseteq_{\mathcal{Y}} I_{\mathcal{Y}}$, we have $I_{\mathcal{Y}} \leq (\preceq_{\mathcal{Y}}) \circ F \circ G$. Hence, $G^{\dagger} \leq (\preceq_{\mathcal{Y}}) \circ F \circ G \circ G^{\dagger} \leq (\preceq_{\mathcal{Y}}) \circ F$. Since F is monotone, we have $F \circ (\preceq_{\mathcal{X}}) \leq (\preceq_{\mathcal{Y}}) \circ F$, whence $G^{\dagger} \circ (\preceq_{\mathcal{X}}) \leq (\preceq_{\mathcal{Y}}) \circ F \circ (\preceq_{\mathcal{X}}) \leq (\preceq_{\mathcal{Y}} \circ \preceq_{\mathcal{Y}}) \circ F = (\preceq_{\mathcal{Y}}) \circ F$. We conclude that $(\preceq_{\mathcal{Y}}) \circ F = G^{\dagger} \circ (\preceq_{\mathcal{X}})$, so F is the lower adjoint of G . \square

The next example is the quantum version of the statement that the direct image and the preimage of a function form a Galois connection:

Example 7.3. $\text{qPow}(F): \text{qPow}(\mathcal{X}) \rightarrow \text{qPow}(\mathcal{Y})$ is the lower Galois adjoint of $\text{qPow}(F^{\dagger})$ for any function $F: \mathcal{X} \rightarrow \mathcal{Y}$.

Also the notion of closure operators can be generalized to the quantum setting:

Definition 7.4. Let (\mathcal{X}, \preceq) be a quantum poset. Then we call a monotone function $C: \mathcal{X} \rightarrow \mathcal{X}$ a closure operator on \mathcal{X} if $I_{\mathcal{X}} \sqsubseteq C$ and $C \circ C = C$.

Just as in the classical case, there is a relation between Galois connections and closure operators:

Theorem 7.5. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum posets and let $F: \mathcal{X} \rightarrow \mathcal{Y}$ be a monotone function that is the lower adjoint of a monotone function $G: \mathcal{Y} \rightarrow \mathcal{X}$. Then $C := G \circ F$ is a closure operator on \mathcal{X} . Conversely, let C be a closure operator on a quantum poset $(\mathcal{X}, \preceq_{\mathcal{X}})$. Let $\mathcal{Y} = \text{ran } C$, and let $\preceq_{\mathcal{Y}}$ be the induced order on \mathcal{Y} , i.e., $\preceq_{\mathcal{Y}} = \preceq_{\mathcal{X}}|_{\mathcal{Y}}$. Then the unique surjective monotone function $\bar{C}: \mathcal{X} \rightarrow \mathcal{Y}$

such that $C = J_{\mathcal{Y}} \circ \bar{C}$ is the lower adjoint of the order embedding $J_{\mathcal{Y}}: \mathcal{Y} \rightarrow \mathcal{X}$. In particular, we have $\bar{C} \circ J_{\mathcal{Y}} = I_{\mathcal{Y}}$.

The quantum version of the operation $A \mapsto \downarrow A$ on a power set $\text{Pow}(X)$ is a closure operator:

Example 7.6. Let \preceq be an order on a quantum set \mathcal{X} . Then $\text{qPow}(\preceq)$ is a closure operator on $\text{qPow}(\mathcal{X})$. Its range equipped with the relative order equals $\text{qDwn}(\mathcal{X}, \preceq)$.

8 Quantum suplattices

Classically, a poset X is a complete lattice if its canonical embedding $X \rightarrow \text{Dwn}(X)$, $x \mapsto \downarrow x$ into its poset of down-sets has a lower Galois adjoint. We will use this fact in order to define quantum suplattices.

Definition 8.1. Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ be a quantum poset, and let $\downarrow\{\cdot\}_{(\mathcal{X}, \preceq_{\mathcal{X}})}: (\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow \text{qDwn}(\mathcal{X}, \preceq_{\mathcal{X}})$ be the order embedding of \mathcal{X} into the quantum poset of down-sets of \mathcal{X} . Then we say that $(\mathcal{X}, \preceq_{\mathcal{X}})$ is a quantum suplattice if $\downarrow\{\cdot\}_{(\mathcal{X}, \preceq_{\mathcal{X}})}$ has a lower adjoint $\mathbf{V}_{(\mathcal{X}, \preceq_{\mathcal{X}})}: \text{qDwn}(\mathcal{X}, \preceq_{\mathcal{X}}) \rightarrow (\mathcal{X}, \preceq_{\mathcal{X}})$. Moreover, if $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ is another quantum suplattice, then we say that a function $K: \mathcal{X} \rightarrow \mathcal{Y}$ is a homomorphism of quantum suplattices if $K \circ \mathbf{V}_{(\mathcal{X}, \preceq_{\mathcal{X}})} = \mathbf{V}_{(\mathcal{Y}, \preceq_{\mathcal{Y}})} \circ \mathcal{D}(K)$. We denote the category of quantum suplattices and homomorphisms of quantum suplattices by **qSup**.

We will show that quantum posets of down-sets form the primary examples of quantum suplattices. In the classical case, this is completely obvious; one just need to observe that a union of down-sets is a down-set. However, in the quantum case, the proof is nontrivial. We need one crucial lemma.

Lemma 8.2. Let (\mathcal{X}, \preceq) be a quantum poset. Then the inverse inclusion order \supseteq on $\mathcal{D}(\mathcal{X})$ is the largest binary relation T on $\mathcal{D}(\mathcal{X})$ such that $(\exists_{(\mathcal{X}, \preceq)}) \circ T \leq (\exists_{(\mathcal{X}, \preceq)})$.

Theorem 8.3. Let (\mathcal{X}, \preceq) be a quantum poset. Then $\text{qDwn}(\mathcal{X}, \preceq)$ is a quantum suplattice. More specifically, the lower Galois adjoint $\mathbf{V}_{\mathcal{D}(\mathcal{X}, \preceq)}$ of the canonical embedding $\downarrow\{\cdot\}_{\text{qDwn}(\mathcal{X}, \preceq)}: \text{qDwn}(\mathcal{X}, \preceq) \rightarrow \text{qDwn}^2(\mathcal{X}, \preceq)$ is given by the (\mathcal{X}, \preceq) -component of $\mathbf{U}_{(\mathcal{X}, \preceq)}$ of the multiplication $\mathbf{U}: \text{qDwn}^2 \rightarrow \text{qDwn}$ of the qDwn -monad on **qPos**.

Proof. For simplicity, we write $\downarrow\{\cdot\}$ instead of $\downarrow\{\cdot\}_{\text{qDwn}(\mathcal{X}, \preceq)}$ and \mathbf{U} instead of $\mathbf{U}_{(\mathcal{X}, \preceq)}$. Consider the following two diagrams:

$$\begin{array}{ccc}
 \text{qDwn}(\mathcal{X}) & & \text{qDwn}(\mathcal{X}) \\
 \downarrow \downarrow\{\cdot\} & \searrow I_{\text{qDwn}(\mathcal{X})} & \downarrow \downarrow\{\cdot\} \\
 \text{qDwn}^2(\mathcal{X}) & \xrightarrow{\exists_{\text{qDwn}(\mathcal{X}, \preceq)}} & \text{qDwn}(\mathcal{X}) \\
 \downarrow \text{qDwn}'(\exists_{(\mathcal{X}, \preceq)}) & & \downarrow \exists_{(\mathcal{X}, \preceq)} \\
 \text{qDwn}(\mathcal{X}) & \xrightarrow{\exists_{(\mathcal{X}, \preceq)}} & (\mathcal{X}, \preceq)
 \end{array}
 \qquad
 \begin{array}{ccc}
 \text{qDwn}(\mathcal{X}) & & \text{qDwn}(\mathcal{X}) \\
 \downarrow \downarrow\{\cdot\} & \searrow \exists_{(\mathcal{X}, \preceq)} & \downarrow \downarrow\{\cdot\} \\
 \text{qDwn}^2(\mathcal{X}) & & \text{qDwn}(\mathcal{X}) \\
 \downarrow \mathbf{U} & & \downarrow \exists_{(\mathcal{X}, \preceq)} \\
 \text{qDwn}(\mathcal{X}) & \xrightarrow{\exists_{(\mathcal{X}, \preceq)}} & (\mathcal{X}, \preceq)
 \end{array}$$

The triangle in the left diagram commutes by Theorem 4.4. The square commutes by naturality of \exists and since qDwn' equals qDwn on objects. The right diagram commutes since $\mathbf{U}_{(\mathcal{X}, \preceq)} = \text{qDwn}(\exists_{(\mathcal{X}, \preceq)})$, so it is the outside of the left diagram. By the universal property of \exists as the counit of the adjunction $(-)_\circ \dashv \text{qDwn}$, the unique monotone function $K: \text{qDwn}(\mathcal{X}) \rightarrow \text{qDwn}(\mathcal{X})$ such that $\exists_{(\mathcal{X}, \preceq)} \circ K = \exists_{(\mathcal{X}, \preceq)}$ is the identity function on $\mathcal{D}(\mathcal{X})$. Hence, we conclude that $\mathbf{U} \circ \downarrow\{\cdot\} = I_{\mathcal{D}(\mathcal{X})}$. Note that the right-hand side is not the same as the identity monotone relation $I_{\text{qDwn}(\mathcal{X})}$, which is \supseteq , even though $\mathcal{D}(\mathcal{X})$ is the underlying quantum set of $\text{qDwn}(\mathcal{X})$.

As a consequence, \mathbf{U} is an epimorphism in \mathbf{qSet} , hence it is surjective, so $\mathbf{U} \circ \mathbf{U}^\dagger = I_{\mathcal{D}(\mathcal{X})}$. Then, using the naturality of \exists , we obtain

$$\exists_{(\mathcal{X}, \preceq)} = \exists_{(\mathcal{X}, \preceq)} \circ \mathbf{U} \circ \mathbf{U}^\dagger = \exists_{(\mathcal{X}, \preceq)} \circ \mathbf{qDwn}'(\exists_{(\mathcal{X}, \preceq)}) \circ \mathbf{U}^\dagger = \exists_{(\mathcal{X}, \preceq)} \circ \exists_{\mathbf{qDwn}(\mathcal{X}, \preceq)} \circ \mathbf{U}^\dagger,$$

which, by Lemma 8.2, implies $\exists_{\mathbf{qDwn}(\mathcal{X})} \circ \mathbf{U}^\dagger \leq (\exists) = I_{\mathbf{qDwn}(\mathcal{X})}$. Since also $I_{\mathbf{qDwn}(\mathcal{X})} = \exists_{\mathcal{D}(\mathcal{X}, \preceq)} \circ \downarrow\{\cdot\}_{\mathcal{D}(\mathcal{X}, \preceq)}$ (cf. Theorem 4.4), we obtain $\exists_{\mathbf{qDwn}(\mathcal{X})} \circ \mathbf{U}^\dagger \leq \exists_{\mathbf{qDwn}(\mathcal{X})} \circ \downarrow\{\cdot\}$. Then, since $\downarrow\{\cdot\}$ is a function, we find $\exists_{\mathbf{qDwn}(\mathcal{X})} \circ \mathbf{U}^\dagger \circ (\downarrow\{\cdot\})^\dagger \leq \exists_{\mathbf{qDwn}(\mathcal{X})} \circ \downarrow\{\cdot\} \circ (\downarrow\{\cdot\})^\dagger \leq \exists_{\mathbf{qDwn}(\mathcal{X})}$. Again applying Lemma 8.2 yields $\mathbf{U}^\dagger \circ (\downarrow\{\cdot\})^\dagger \leq (\exists)$, whence $(\subseteq) \circ \downarrow\{\cdot\} \circ \mathbf{U} \leq (\subseteq) \circ (\exists)^\dagger = (\subseteq) \circ (\subseteq) \leq (\subseteq) = (\subseteq) \circ I_{\mathcal{D}^2(\mathcal{X})}$, which expresses that $I_{\mathcal{D}^2(\mathcal{X})} \leq \mathbf{U} \circ \downarrow\{\cdot\}$. It now follows from Theorem 7.2 that \mathbf{U} is the lower Galois adjoint $\mathbf{V}_{\mathbf{qDwn}(\mathcal{X})}$ of $\downarrow\{\cdot\}$, hence $\mathbf{qDwn}(\mathcal{X})$ is indeed a quantum suplattice. \square

We can now formulate the quantum versions of two classical theorems on suplattices:

Theorem 8.4. *Let $(\mathcal{X}, \preceq_{\mathcal{X}})$ and $(\mathcal{Y}, \preceq_{\mathcal{Y}})$ be quantum suplattices and let $F: \mathcal{X} \rightarrow \mathcal{Y}$ be monotone. Then F is a homomorphism of quantum suplattices if and only if it has an upper Galois adjoint.*

Theorem 8.5. *The Eilenberg-Moore category of \mathbf{qDwn} is equivalent to \mathbf{qSup} .*

9 Quantum inflattices

We can define an inflattice to be the opposite of a suplattice. In the quantum world, we follow the same path to define quantum inflattices:

Definition 9.1. *A quantum poset (\mathcal{X}, \preceq) is called a quantum inflattice if (\mathcal{X}, \succeq) is a quantum suplattice.*

Classically, a poset is a suplattice if and only if it is an inflattice. The same is true in the quantum case. In order to prove this, a first step is showing a different characterization of quantum suplattices, which is analogue to the observation that any ordinary poset X is a suplattice if and only if the embedding $X \rightarrow \mathbf{Pow}(X)$, $x \mapsto \downarrow x$ has a lower Galois adjoint. The analogue embedding $D_{(\mathcal{X}, \preceq)}$ of a quantum poset (\mathcal{X}, \preceq) into $\mathbf{qPow}(\mathcal{X})$ is given by $J \circ \downarrow\{\cdot\}_{\mathcal{X}}$, where $J: \mathbf{qDwn}(\mathcal{X}, \preceq) \rightarrow \mathbf{qPow}(\mathcal{X})$ is the order embedding from Example 7.6.

Lemma 9.2. *A quantum poset (\mathcal{X}, \preceq) is a quantum suplattice if and only if $D_{(\mathcal{X}, \preceq)}: (\mathcal{X}, \preceq) \rightarrow \mathbf{qPow}(\mathcal{X})$ has a lower Galois adjoint $F_{(\mathcal{X}, \preceq)}$.*

Classically, for any ordinary set X and any subset $\mathcal{A} \subseteq \mathbf{Pow}(X)$, the intersection $\bigcap \mathcal{A}$ of all subsets in \mathcal{A} is given by $X \setminus \bigcup_{A \in \mathcal{A}} (X \setminus A)$, so by using the union operator and the complement operator. Since for any quantum set \mathcal{X} its quantum power set $\mathbf{qPow}(\mathcal{X}) = \mathbf{qDwn}(\mathcal{X}, I_{\mathcal{X}})$ is a quantum suplattice (cf. Theorem 8.3), and similar to the complementation operator on ordinary power sets, we have an order isomorphism $C_{\mathcal{X}}: \mathbf{qPow}(\mathcal{X}) \rightarrow \mathbf{qPow}(\mathcal{X})^{\text{op}}$, we can prove:

Lemma 9.3. *The quantum power set $\mathbf{qPow}(\mathcal{X})$ of any quantum set \mathcal{X} is a quantum inflattice.*

Then, given a quantum suplattice (\mathcal{X}, \preceq) with embedding $D_{(\mathcal{X}, \preceq)}: (\mathcal{X}, \preceq) \rightarrow \mathbf{qPow}(\mathcal{X})$ that is the upper adjoint of F , and denoting the lower adjoint of $D_{\mathbf{qPow}(\mathcal{X})^{\text{op}}}: \mathbf{qPow}(\mathcal{X})^{\text{op}} \rightarrow \mathbf{qPow}(\mathcal{P}(\mathcal{X}))$ by N , we can show that $F \circ N \circ \mathbf{qPow}(D_{(\mathcal{X}, \preceq)})$ is the lower Galois adjoint of $D_{(\mathcal{X}, \succeq)}: (\mathcal{X}, \succeq) \rightarrow \mathbf{qPow}(\mathcal{X})$, proving:

Theorem 9.4. *Any quantum suplattice (\mathcal{X}, \preceq) is a quantum inflattice.*

10 Enrichment

It was shown in [14] that the pointwise order of functions between quantum sets induces a **Pos**-enrichment of **qPos**. Similarly, in [13] it was shown that the category **qCPO** of quantum cpos is enriched over the category **CPO** of cpos, i.e., posets for which any monotonically increasing sequence has a supremum. We can enrich **qSup** over **Sup** in a similar way. We first have to quantize the supremum of collections of functions into a quantum suplattice.

Definition 10.1. *Let \mathcal{X} be a quantum set and (\mathcal{Y}, \preceq) a quantum poset. Let \mathfrak{K} be a subset of $\mathbf{qSet}(\mathcal{X}, \mathcal{Y})$. Then a function $F: \mathcal{X} \rightarrow \mathcal{Y}$ is called the limit of \mathfrak{K} , denoted by $F = \lim \mathfrak{K}$ if $(\preceq) \circ F = \bigwedge_{K \in \mathfrak{K}} (\preceq) \circ K$.*

A quantum cpo is a quantum poset (\mathcal{Y}, \preceq) for which the limit of any countable chain in $\mathbf{qSet}(\mathcal{X}, \mathcal{Y})$ exists for any quantum set \mathcal{X} , so the above definition is a generalization of the concept of limits in [14].

Let \mathcal{X} be a quantum set and let (\mathcal{Y}, \preceq) be a quantum poset. Let $\mathfrak{K} \subseteq \mathbf{qSet}(\mathcal{X}, \mathcal{Y})$. If $\lim \mathfrak{K}$ exists, then it is the supremum of \mathfrak{K} in $\mathbf{qSet}(\mathcal{X}, \mathcal{Y})$ ordered by \sqsubseteq . The converse does not always hold. If \mathcal{Y} is a quantum suplattice, then $\lim \mathfrak{K}$ always exists. We can prove this by first showing the case $\mathcal{Y} = \mathbf{2}$, then the case $\mathcal{Y} = \mathbf{qPow}(\mathcal{X})$, which equals $[\mathcal{X}^*, \mathbf{2}]_{\sqsubseteq}$, and finally the general case using Lemma 9.2. Moreover, one can show that $\lim \mathfrak{K}$ is a homomorphism of quantum suplattices if \mathcal{X} is also a quantum suplattice and all functions in \mathfrak{K} are homomorphisms of quantum suplattices. Finally, one can show that composition with homomorphisms of quantum suplattices preserves the operation of taking limits, yielding:

Theorem 10.2. ***qSup** is enriched over **Sup**.*

11 Fixpoints

Let \mathcal{X} be a quantum set and \mathcal{Y} a quantum suplattice. Since $\mathbf{qSet}(\mathcal{X}, \mathcal{Y})$ is a complete lattice, it follows from the Knaster-Tarski fixpoint theorem that:

Proposition 11.1. *Let $F: \mathcal{Y} \rightarrow \mathcal{Y}$ be a monotone map on a quantum suplattice \mathcal{Y} . Then for each quantum set \mathcal{X} , the set of all functions $K: \mathcal{X} \rightarrow \mathcal{Y}$ such that $F \circ K = K$ is a complete lattice.*

Categorically, a generalized fixpoint of an endomorphism $f: Y \rightarrow Y$ in a given category **C** can be defined as a monomorphism $m: X \rightarrow Y$ for some object X such that $f \circ m = m$. This leads to:

Definition 11.2. *Let (\mathcal{Y}, \preceq) be a quantum poset, and let $F: \mathcal{Y} \rightarrow \mathcal{Y}$ be a monotone function. If the largest subset \mathcal{X} of \mathcal{Y} such that $F \circ J_{\mathcal{X}} = J_{\mathcal{X}}$ exists, we call it the quantum set of fixpoints of F , denoted by $\text{Fix}(F)$. Similarly, if the largest subset \mathcal{X} of \mathcal{Y} such that $F \circ J_{\mathcal{X}} \sqsubseteq J_{\mathcal{X}}$ exists, we call it the quantum set of prefixpoints of F , denoted by $\text{Pre}(F)$. Finally, if the largest subset \mathcal{X} of \mathcal{Y} such that $F \circ J_{\mathcal{X}} \sqsupseteq J_{\mathcal{X}}$ exists, we call it the quantum set of postfixpoints of F , denoted by $\text{Post}(F)$.*

Denoting $Q\{X\}$ for the quantum subset of \mathcal{Y} consisting of the atom $X \in \mathcal{Y}$, we can show that the largest subset $\text{Post}(F)$ of postfixpoints of a monotone endofunction F on a quantum poset \mathcal{Y} exists and is determined by $\text{At}(\mathcal{X}) = \{X \in \mathcal{Y} : J_{Q\{X\}} \sqsubseteq F \circ J_{Q\{X\}}\}$. Similarly, one can show that $\text{Pre}(F)$ and $\text{Fix}(F)$ exist.

Classically, the postfixpoints P of a monotone map $f: Y \rightarrow Y$ on a suplattice form a suplattice. This can be seen by a follows. $P = \{y \in Y : y \leq f(y)\}$. Let $S \subseteq P$, and let $x = \sup S$. Then by monotonicity of f , we have $y \leq f(y) \leq f(x)$ for each $y \in S$, hence $x = \bigvee S \leq f(x)$, showing that P is closed under suprema, so a suplattice. In a similar way, we can prove:

Proposition 11.3. *Given a monotone endofunction $F: \mathcal{Y} \rightarrow \mathcal{Y}$ on a quantum suplattice \mathcal{Y} , the quantum set $\text{Post}(F)$ of postfixpoints of F is a quantum suplattice with respect to the relative order.*

Applying Theorem 9.4, it is easy to show that $\text{Pre}(F)$ also is a quantum suplattice with respect to the relative order. Finally, we can show that F restricts and corestricts to a monotone function $F|_{\text{Post}(F)}^{\text{Post}(F)}$ and that $\text{Fix}(F) = \text{Pre}\left(F|_{\text{Post}(F)}^{\text{Post}(F)}\right)$ from which we conclude:

Theorem 11.4 (Quantum Knaster-Tarski Theorem). *Let $F : \mathcal{Y} \rightarrow \mathcal{Y}$ be a monotone endomap on a quantum suplattice \mathcal{Y} . Then the quantum set $\text{Fix}(F)$ of fixpoints of F is a quantum suplattice with respect to the relative order.*

12 The relation between ordinary suplattices and quantum suplattices

Quantum posets are noncommutative generalizations of ordinary posets, since we have a fully faithful functor $'(-) : \mathbf{Pos} \rightarrow \mathbf{qPos}$. This functor restricts and corestricts to a functor $\mathbf{CPO} \rightarrow \mathbf{qCPO}$, hence also quantum cpos are genuine noncommutative generalizations of ordinary cpos. Remarkably, quantum suplattices are a not noncommutative generalizations of ordinary suplattices, but only noncommutative versions of suplattice: the functor $'(-) : \mathbf{Pos} \rightarrow \mathbf{qPos}$ does not restrict and corestrict to a functor $'(-) : \mathbf{Sup} \rightarrow \mathbf{qSup}$. Indeed, if B denotes the four-element Boolean algebra, which is clearly an ordinary suplattice, then $'B$ is not a quantum suplattice. In order to see this, we first recall Section 10, in which it was stated that the limit of any collection of functions with the same domain into a quantum suplattice always exists. We will give a collection of functions into $'B$ that does not have a limit. Write $B = \{0, a, b, 1\}$, where $a^\perp = b$, and denote the order on B by \sqsubseteq , so $0 \sqsubseteq a, b \sqsubseteq 1$. Let $(\mathcal{X}, \mathfrak{K}) = ('B, \sqsubseteq)$. Let \mathcal{H} denote the atomic quantum set whose single atom H is two-dimensional. One can show that any function $K : \mathcal{H} \rightarrow \mathcal{X}$ is of the form $K(H, \mathbb{C}_x) = L(H, \mathbb{C}_x)r_x$ for each $x \in B$, where r_0, r_a, r_b, r_1 are mutually orthogonal projections on H whose sum equals 1_H . Then $(\mathfrak{K} \circ K)(H, \mathbb{C}_x) = L(H, \mathbb{C}_x)\sum_{y \sqsubseteq x} r_y$. So $(\mathfrak{K} \circ K)(H, \mathbb{C}_x)$ is of the form $L(H, \mathbb{C}_x)s_x$ for some projection s_x of H , and clearly s_0, s_a, s_b and s_1 mutually commute, because r_0, r_a, r_b, r_1 are mutually orthogonal.

Let p and q be two distinct nontrivial noncommuting projections on H . For instance, in the standard basis of H , let $p = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ and $q = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$. Then p and q are also not orthogonal, whence, $pq \neq 0$. However, since p and q are both atomic projections, we have $p \wedge q = 0$.

Let $F, G : \mathcal{H} \rightarrow \mathcal{X}$ be functions defined as follows. The nonzero components of F are given by $F(H, \mathbb{C}_0) = L(H, \mathbb{C}_0)p$ and $F(H, \mathbb{C}_a) = L(H, \mathbb{C}_a)p^\perp$, whereas the nonzero components of G are given by $G(H, \mathbb{C}_0) = L(H, \mathbb{C}_0)q$, $G(H, \mathbb{C}_b) = L(H, \mathbb{C}_b)q^\perp$. Then

$$(\mathfrak{K} \circ F)(H, \mathbb{C}_x) = \begin{cases} L(H, \mathbb{C}_x)p, & x = 0, b; \\ L(H, \mathbb{C}_x), & x = a, 1, \end{cases}$$

$$(\mathfrak{K} \circ G)(H, \mathbb{C}_x) = \begin{cases} L(H, \mathbb{C}_x)q, & x = 0, a; \\ L(H, \mathbb{C}_x), & x = b, 1. \end{cases}$$

hence

$$(\mathfrak{K} \circ F \wedge \mathfrak{K} \circ G)(H, \mathbb{C}_x) = \begin{cases} 0, & x = 0 \\ L(H, \mathbb{C}_a)q, & x = a; \\ L(H, \mathbb{C}_b)p, & x = b; \\ L(H, \mathbb{C}_1), & x = 1. \end{cases}$$

Assume that there is a function $K: \mathcal{H} \rightarrow \mathcal{X}$ such that $\aleph \circ R = (\aleph \circ F) \wedge (\aleph \circ G)$. Since $(\aleph \circ K)(H, \mathbb{C}_a) = L(H, \mathbb{C}_a)q$ and $(\aleph \circ K)(H, \mathbb{C}_b) = L(H, \mathbb{C}_b)p$, it follows that p and q should commute, which contradicts our assumptions. We conclude that there is a $\aleph \subseteq \mathbf{qSet}(\mathcal{H}, \mathcal{X})$ such that $\lim \aleph$ does not exist, namely $\aleph = \{F, G\}$. Hence, \mathcal{X} cannot be a quantum suplattice, for which all limits should exist.

The main reason why suplattices are not quantum suplattices lies in the fact that the $'(-)$ functors do not commute with \mathcal{P} and \mathcal{D} . Let S be a set. Then $'P(S)$ does not equal $\mathcal{P}'(S)$, but can only be identified with the subset of one-dimensional atoms of $\mathcal{P}'(S)$. That this subset is proper follows for instance from the proof of [12, Proposition 9.3], which asserts that $(\mathbf{1} \uplus \mathbf{1}) * (\mathbf{1} \uplus \mathbf{1})$, which can be identified with $\mathcal{P}'(2)$, has uncountably many atoms. The same is true for $'D(S)$ if S is a poset: $'D(S)$ can only be identified with the subset of one-dimensional atoms of $\mathcal{D}'(S)$. We plan to investigate whether the quantum power set monad \mathcal{P} is the right Kan extension of $'(-) \circ P: \mathbf{Set} \rightarrow \mathbf{qSet}$ along $'(-): \mathbf{Set} \rightarrow \mathbf{qSet}$, where P denotes the ordinary power set monad. If this is true, we also expect that the quantum down-set monad \mathbf{qDwn} is the right Kan extension of $'(-) \circ \mathbf{Dwn}: \mathbf{Pos} \rightarrow \mathbf{qPos}$ along $'(-): \mathbf{Pos} \rightarrow \mathbf{qPos}$.

It follows from the **Sup**-enrichment of **qSup** that the one-dimensional atoms of a quantum suplattice form an ordinary suplattice. We expect that any ordinary suplattice is the subposet of one-dimensional atoms of some quantum suplattice.

By extension, if quantum suplattices indeed form the right notion that is needed to generalize topologies to the noncommutative setting, then it follows that an ordinary topology on an ordinary set is not a quantum topology, it might only be the classical part of a quantum topology. Moreover, it might be that there are several different quantum topologies on an ordinary set that have the same ordinary topology as classical part. If this is indeed the case, it is the question how to interpret this. Perhaps this would be a quantum feature, in the same spirit as the result of two nonisomorphic graphs that are quantum isomorphic [2].

13 Future work

The classical Knaster-Tarski Theorem implies the Cantor-Schröder-Bernstein Theorem. We conjecture that a quantum version of Cantor-Schröder-Bernstein can be derived from the quantum Knaster-Tarski Theorem.

Conjecture 13.1. *Let $F: \mathcal{X} \rightarrow \mathcal{Y}$ and $G: \mathcal{Y} \rightarrow \mathcal{X}$ be injective functions between quantum sets. Then there exists a bijection between \mathcal{X} and \mathcal{Y} .*

The biggest potential obstacle is that an atom of the quantum power set of a quantum set \mathcal{X} does not directly correspond to a subset of \mathcal{X} , as is the case in the classical case.

A quantum Cantor-Schröder-Bernstein Theorem can be reformulated in terms of operator algebras:

Conjecture 13.2. *Given surjective normal unital $*$ -homomorphisms $\varphi: M \rightarrow N$ and $\psi: N \rightarrow M$ between hereditarily atomic von Neumann algebras, there must exist a $*$ -isomorphism between M and N .*

Furthermore, based on the fact that any quantum suplattice is a quantum inflattice, we expect:

Conjecture 13.3. ***qSup** can be equipped with a monoidal product that makes it $*$ -autonomous.*

Acknowledgements

We thank Andre Kornell for his advice and for inspiring us to further develop the theory of quantum sets. Furthermore, we thank Isar Stubbe for helping us to understand the connection between quantum sets and quantaloids better. Furthermore, we thank the reviewers for their comments. This research is supported by grants VEGA 2/0142/20 and 1/0036/23 and by the Slovak Research and Development Agency under the contracts APVV-18-0052 and APVV-20-0069.

References

- [1] S. Abramsky & B. Coecke (2009): *Categorical Quantum Mechanics*. In K. Engesser, D.M. Gabbay & D. Lehmann, editors: *Handbook of Quantum Logic and Quantum Structures*, Elsevier, Amsterdam, pp. 261–323, doi:10.1016/B978-0-444-52869-8.50010-4. Available at <https://www.sciencedirect.com/science/article/pii/B9780444528698500104>.
- [2] A. Atserias, L. Mančínska, D.E. Roberson, R. Šámal, S. Severini & A. Varvitsiotis (2019): *Quantum and non-signalling graph isomorphisms*. *Journal of Combinatorial Theory, Series B* 136, pp. 289–328, doi:10.1016/j.jctb.2018.11.002. Available at <https://www.sciencedirect.com/science/article/pii/S0095895618301059>.
- [3] R. Betti, A. Carboni, R. Street & R. Walters (1983): *Variation through Enrichment*. *Journal of Pure and Applied Algebra* 29, pp. 109–127, doi:10.1016/0022-4049(83)90100-7.
- [4] M. Bílková, A. Kurz, D. Petrişan & J. Velebil (2011): *Relation Liftings on Preorders and Posets*. In A. Corradini, B. Klin & C. Cîrstea, editors: *Algebra and Coalgebra in Computer Science*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 115–129, doi:10.2168/LMCS-9(4:8)2013.
- [5] M. Brannan, P. Ganesan & S. J. Harris (2022): *The quantum-to-classical graph homomorphism game*. *J. Math. Phys.* 64, p. 112204, doi:10.1063/5.0072288.
- [6] A. Chirvasitu & M. Wasilewski (2022): *Random quantum graphs*. *Trans. Amer. Math. Soc.* 375, pp. 3061–3087, doi:10.1090/tran/8584.
- [7] B. Coecke & A. Kissinger (2017): *Picturing Quantum Processes A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, doi:10.1017/9781316219317.
- [8] R. Duan, S. Severini & A. Winter (2013): *Zero-error communication via quantum channels, noncommutative graphs, and a quantum Lovász number*. *IEEE Trans. Inform. Theory* 59(2), doi:10.1109/TIT.2012.2221677.
- [9] C. Heunen & J. Vicary (2019): *Categories for Quantum Theory*. Oxford University Press, doi:10.1093/oso/9780198739623.001.0001.
- [10] H. Heymans & I. Stubbe (2012): *Grothendieck quantaloids for allegories of enriched categories*. *Bulletin of the Belgian Mathematical Society - Simon Stevin* 19(5), pp. 859 – 888, doi:10.36045/bbms/1354031554.
- [11] A. Kornell (2020): *Discrete quantum structures*. doi:10.48550/arXiv.2004.04377. arXiv:arXiv:2004.04377.
- [12] A. Kornell (2020): *Quantum Sets*. *J. Math. Phys.* 61, doi:10.1063/1.5054128.
- [13] A. Kornell, B. Lindenhovius & M. Mislove: *Quantum CPOs*. In: *Proceedings QPL 2020*, pp. 174–187, doi:10.48550/arXiv.2109.02196.
- [14] A. Kornell, B. Lindenhovius & M. Mislove (2022): *A category of quantum posets*. *Indagationes Mathematicae* 33, pp. 1137–1171, doi:10.1016/j.indag.2022.07.001.
- [15] G. Kuperberg & N. Weaver (2012): *A Von Neumann Algebra Approach to Quantum Metrics: Quantum Relations*. *Memoirs of the American Mathematical Society*, American Mathematical Society, doi:10.1090/S0065-9266-2011-00637-4.
- [16] B. Lindenhovius, M. Mislove & V. Zamdzhiev (2018): *Enriching a Linear/Non-linear Lambda Calculus: A Programming Language for String Diagrams*. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '18*, ACM, New York, NY, USA, pp. 659–668, doi:10.1145/3209108.3209196.
- [17] B. Lindenhovius, M. Mislove & V. Zamdzhiev (2021): *LNL-FPC: The Linear/Non-Linear Fixpoint Calculus*. *Logical Methods in Computer Science* 17, pp. 9:1 – 9:61, doi:10.23638/LMCS-17(2:9)2021.
- [18] B. Musto, D. J. Reutter & D. Verdon (2018): *A compositional approach to quantum functions*. *J. Math. Phys.* 59, doi:10.1063/1.5020566.

- [19] F. Rios & P. Selinger (2017): *A categorical model for a quantum circuit description language*. In B. Coecke & A. Kissinger, editors: *Proceedings 14th International Conference on Quantum Physics and Logic, QPL 2017, Nijmegen, The Netherlands, 3-7 July 2017.*, EPTCS 266, pp. 164–178, doi:10.4204/EPTCS.266.11.
- [20] D. Stahlke (2016): *Quantum Zero-Error Source-Channel Coding and Non-Commutative Graph Theory*. *IEEE Trans. Inform. Theory* **62**(1), doi:10.1109/TIT.2015.2496377.
- [21] N. Weaver (2012): *Quantum relations*. *Mem. Amer. Math. Soc.* 215, doi:10.1090/S0065-9266-2011-00637-4.
- [22] N. Weaver (2021): *Quantum Graphs as Quantum Relations*. *J. Geom. Anal.*, doi:10.1007/s12220-020-00578-w.

Global Synthesis of CNOT Circuits with Holes

Ewan Murphy

University of Oxford

ewan.murphy360@gmail.com

Aleks Kissinger

University of Oxford

aleks.kissinger@cs.ox.ac.uk

A common approach to quantum circuit transformation is to use the properties of a specific gate set to create an efficient representation of a given circuit’s unitary, such as a parity matrix or stabiliser tableau, and then resynthesise an improved circuit, e.g. with fewer gates or respecting connectivity constraints. Since these methods rely on a restricted gate set, generalisation to arbitrary circuits usually involves slicing the circuit into pieces that can be resynthesised and working with these separately. The choices made about what gates should go into each slice can have a major effect on the performance of the resynthesis. In this paper we propose an alternative approach to generalising these resynthesis algorithms to general quantum circuits. Instead of cutting the circuit into slices, we “cut out” the gates we can’t resynthesise leaving holes in our quantum circuit. The result is a second-order process called a quantum comb, which can be resynthesised directly. We apply this idea to the RowCol algorithm, which resynthesises CNOT circuits for topologically constrained hardware, explaining how we were able to extend it to work for quantum combs. We then compare the generalisation of RowCol using our method to the naïve “slice and build” method empirically on a variety of circuit sizes and hardware topologies. Finally, we outline how quantum combs could be used to help generalise other resynthesis algorithms.

1 Introduction

Current quantum computers suffer from severe limitations such as high error rates, low numbers of qubits, and connectivity constraints for multi-qubit operations. Furthermore, without error correction, the poor fidelity of current gate implementations compounds over the execution of the circuit, so it advantageous to find the smallest possible circuit to represent an algorithm.

Using the properties of a specific gate set, an efficient representation of a circuit’s unitary can be resynthesised into an improved circuit. For example, the unitary action of CNOT circuits can be fully described by the associated \mathbb{F}_2 -linear function over basis vectors on n -qubit space, seen as vectors in \mathbb{F}_2^n . In other words, we can represent the action of a CNOT circuit using a matrix over \mathbb{F}_2 , called its *parity matrix*.

Noting that the parity matrix of a single CNOT gate corresponds to an elementary row operation, it is possible to resynthesise a CNOT circuit from its parity matrix by performing Gauss-Jordan elimination to reduce the matrix to identity and introducing one CNOT gate for each corresponding row operation. This method was introduced in [2] and refined in the Patel-Markov-Hayes algorithm [15], which produces asymptotically optimal gate counts for (unconstrained) CNOT synthesis. Similar ideas involving the decomposition of symplectic matrices into basic generators have also been applied for synthesising Clifford circuits from stabiliser tableaux [1, 13, 6, 17].

Some quantum computers, such as some superconducting devices [16], have the additional limitation of restricted connectivity, meaning 2-qubit gates aren’t allowed between arbitrary pairs of qubits. By restricting which row operations are possible when reducing a parity matrix to the identity, circuits that obey specific connectivity constraints can be synthesised from ones that don’t. A variety of techniques

for introducing these constraints based on Steiner trees [14, 11, 18, 10] and integer programming [3] have been introduced in recent years.

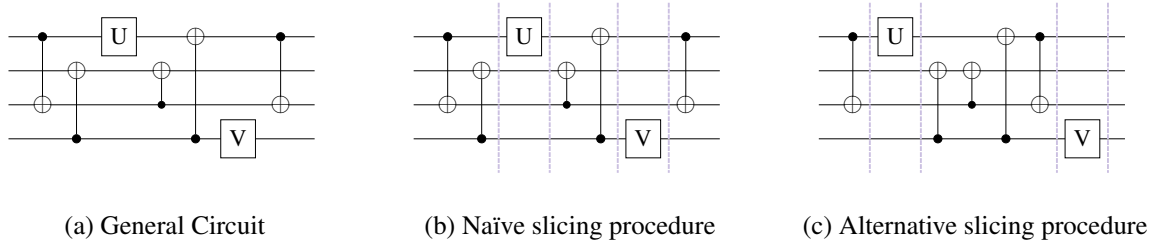


Figure 1: Possible slicing procedures for a synthesis process that can't deal with hadamard gates

These synthesis methods provide a way to optimise circuits that mitigates some of the current limitations of NISQ devices. However, they suffer from relying on the properties of a specific gate set. The conventional generalisation of these methods to arbitrary quantum circuits is to slice the circuit into pieces that can be synthesised, and treat each of them separately[9]. This isn't as straightforward as it might initially sound. Consider for example the circuit in Figure 1a and a CNOT circuit synthesis procedure. The synthesis procedure can't deal with the circuit as it currently is due to the gates U and V, so we need to slice the circuit into sections containing just CNOT gates and synthesise those. As some of the gates in a quantum circuit can be moved past each other, there isn't always a unique way to make a slice. Take the two possible slicing options in Figures 1b and 1c for example. Since the CNOT slices are treated independently during the synthesis procedure, which sides of the slice certain CNOTs end up on can affect the size of the new circuit by allowing/preventing simplifications or cancellations.

In this paper we present a new approach to generalising circuit synthesis that doesn't rely on slicing the circuit into pieces. Instead we remove the gates that can't be synthesised, leaving holes in the circuit and producing what is known as a quantum comb [4, 8]. This quantum comb can be understood as a new circuit with additional qubits, where these new qubits represent the old qubits at different points in time. We then explain how extending the functionality of the CNOT synthesis algorithm RowCol to work quantum combs allows one to route general circuits. Our generalisation is then empirically tested against the naïve slicing process on a variety of circuit sizes and hardware constraints, finding our method has increasingly better performance than the slicing one as circuit sizes increase. Finally, we outline how quantum combs could be used to generalise other circuit synthesis procedures, as well as ways in which our current algorithm could be further optimised.

This paper has the following structure. Section 2.1 explains how the parity matrix representation of a CNOT circuit works. Section 2.2 introduces the idea behind CNOT synthesis algorithms, with a focus on the circuit routing algorithm RowCol. In Section 2.3 we introduce quantum combs, as well as language specific to this paper that will be helpful in explaining our algorithm. Section 3 explains our extension to RowCol that allows working with quantum combs. Empirical results comparing our quantum combs method to the slicing process are presented in Section 4. Finally, Section 5 concludes the paper and outlines possible future works for using quantum combs to generalise other synthesis procedures.

2 Preliminaries

2.1 CNOT Circuit as a Parity Matrix

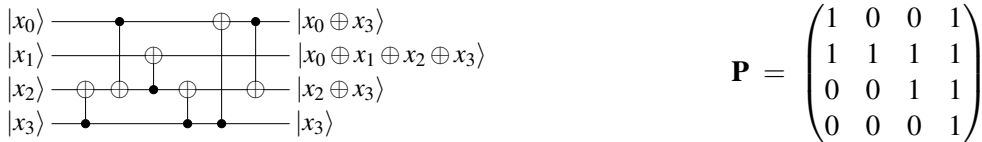
In this paper, we will use the phrases ‘‘CNOT circuit’’ or ‘‘CNOT comb’’ which refer to circuits or quantum combs entirely made of CNOTs. CNOT stands for ‘‘controlled not’’ and is a quantum gate that acts on 2-qubits: the control c and the t , $\text{CNOT}(c, t)$. It acts in such a way that a NOT gate is applied to the target qubit only if the control qubit is in state $|1\rangle$, $\text{CNOT}|0\rangle|0\rangle = |0\rangle|0\rangle$ and $\text{CNOT}|1\rangle|0\rangle = |1\rangle|1\rangle$. When we refer to states in this section we mean computational basis states and the behaviour for general superpositions can be inferred from the linearity of quantum operations. An alternative way of thinking about CNOTs is as modulo 2 addition: the target qubit changes state to the sum of the control and target values modulo 2, $\text{CNOT}|c\rangle|t\rangle = |c\rangle|c \oplus t\rangle$. This idea is represented as a circuit diagram in Figure 2a. A CNOT gate can therefore be written as a list of which qubits are present in the parity equations of the output states: a representation of this is given by the matrix over \mathbb{F}_2 in Figure 2b. By reasoning about CNOT gates in this way we can write an entire CNOT circuit as a list of parity equations, Figure 3a, then represent that circuit by an invertible element of $\mathbb{F}_2^{n \times n}$ (i.e a parity matrix), Figure 3b. One way to construct this matrix is by traversing the circuit and applying row operations for each CNOT, $\text{CNOT}(c, t)$ corresponds to $R(c, t)$, where $R(c, t)$ means setting row t the sum of rows c and t modulo 2. By identifying a set of row operations that reduce a parity matrix to the identity, a CNOT circuit can be generated: this is the core principle behind the CNOT circuit synthesis algorithms discussed in the next section.



(a) CNOT gate as modulo 2 addition of basis states

(b) Parity matrix for a CNOT gate

Figure 2: Parity representations of a CNOT gate



(a) CNOT circuit described as parity equations

(b) Parity matrix corresponding to circuit in 3a

Figure 3: Parity representation of a CNOT circuit

2.2 CNOT Circuit Synthesis Algorithms

Synthesis algorithms provide a means of reducing the size of CNOT circuits[15], or allow the resynthesis of circuits under topological constraints[11, 14, 10]. The ability to perform both of these tasks efficiently is vital for NISQ computing, as it provides a way to best utilise the machines we have available whilst minimising the consequences of their limitations. These CNOT synthesis methods work by converting the circuit into a parity matrix, as described in Section 2.1, then identifying which row operations convert

the matrix back to the identity. This sequence of row operations corresponds to the generated CNOT circuit.

For some quantum computers, superconducting ones, for example [16], CNOT gates may be restricted to only be possible between nearest neighbour qubits. These computers are said to be topologically constrained, with graphs representing the allowed CNOTs called the topology, and the problem of converting a circuit to one that obeys these constraints known as quantum circuit routing. The systematic introduction of SWAPs into the circuit is one way to overcome this problem[5]. However, in this paper we will focus on the circuit synthesis approaches to finding a solution, specifically the ones that use parity matrices, as there are other alternative approaches that synthesise from different representations [3]. By restricting the possible row operations when reducing a parity matrix to the identity, circuit synthesis methods can be also used to produce circuits that obey topological constraints. One of these synthesis algorithms is known as RowCol, and will be the focus of the rest of this section.

Starting with a CNOT circuit \mathcal{C} , and a graph $G(V, E)$ representing connectivity of qubits, RowCol synthesises a new circuit as follows:

Algorithm 1: RowCol

Input : A circuit \mathcal{C} , and topology $G(V, E)$

Output: A circuit \mathcal{C}' , respecting the topological constraints

1. Generate a new empty circuit \mathcal{C}' with the same number of qubits as \mathcal{C} .
 2. Compute the parity matrix P of \mathcal{C} .
 3. Pick a non-cutting vertex of V of G , and get its corresponding qubit q .
 4. Apply elementary row operations, restricted to the edges of $G(V, E)$, to reduce row q and column q to a unit vector.
 5. Remove vertex V from G , and row and column q from P .
 6. Go back to Step 2 and repeat until G has no more vertices.
 7. Return \mathcal{C}'
-

Note the column of q is reduced to a unit vector by using row operations to place a 1 on the diagonal, then adding row q to the other rows to eliminate 1s above and below the diagonal. The row can be made into a unit vector by solving a system of linear equations for other rows to add back on to row q .

In order to respect connectivity constraints, row operations might not be performed directly, but via some intermediate operations computed using Steiner trees. While this is an important aspect of RowCol and related algorithms, we can treat this process essentially as a “black box” for the purposes of our algorithm. We refer readers to the paper that introduced RowCol [18] or other Steiner-tree based algorithms [11, 14, 10] for details.

We are now going to step through the RowCol procedure for the matrix in Figure 3b. We start with qubit 0, meaning we are going to need to eliminate the 0th column, then the 0th row. To eliminate the column we need to perform $R(0, 1)$, and to eliminate the row we need to perform $R(3, 0)$. This has then reduced the 0th column and row to unit vectors, therefore we can ignore these when eliminating the rest of the matrix.

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_0 + R_1} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_0 := R_3 + R_0} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The column for the 1st qubit is already a unit vector, meaning we can move directly onto the row. To eliminate this row we need to perform $R(2,1)$ and $R(3,1)$: although a specific order is shown on the diagram either will work to eliminate the row. As both the column and row are now eliminated, we can ignore these in the subsequent operations.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_2 + R_1} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_3 + R_1} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The column is already a unit vector again here, meaning we move to the row, which can be eliminated by $R(3,2)$. This reduces the matrix to the identity meaning the process is over.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_2 := R_3 + R_2} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The sequence of row operations that converted the matrix to the identity are: $R(0,1)$, $R(3,0)$, $R(2,1)$, $R(3,1)$, $R(3,2)$. This tells us how to construct the resynthesised circuit which is shown in Figure 4.

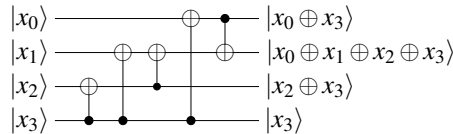


Figure 4: Circuit generated from the step-by-step RowCol procedure

2.3 Quantum Combs

Here we will introduce the concept of a quantum comb, as well as the language and notation used in this paper to discuss specific aspects of them. A quantum comb is a generalisation of a quantum channel that can take other quantum channels as input, rather than states. These can be depicted graphically, as in Figure 5, as circuits which not only have open wires at the top and bottom, but certain “holes” in the middle, where other gates can be inserted. The term “comb” comes from the fact that the entire object should no longer be represented as a box, but as an irregular shape resembling a hair comb, where each the of “teeth” corresponds to a distinct time step. See e.g. [4] for the formal definition and many such pictures.

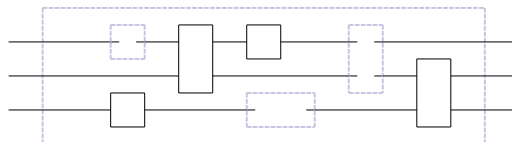
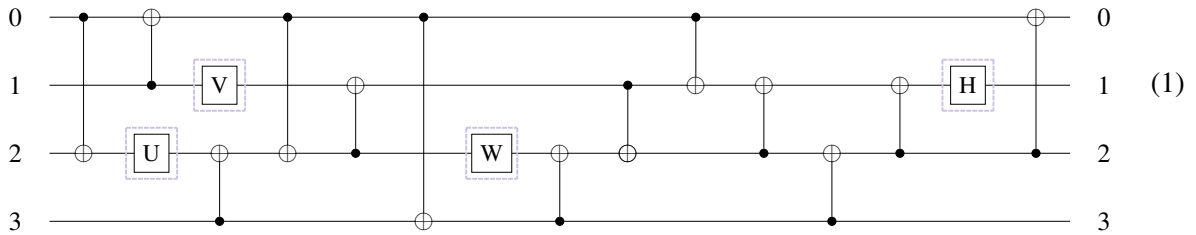
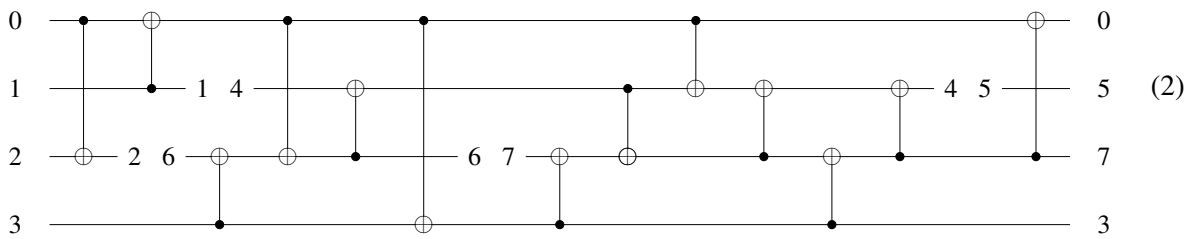


Figure 5: General representation of a quantum comb as quantum circuit with holes

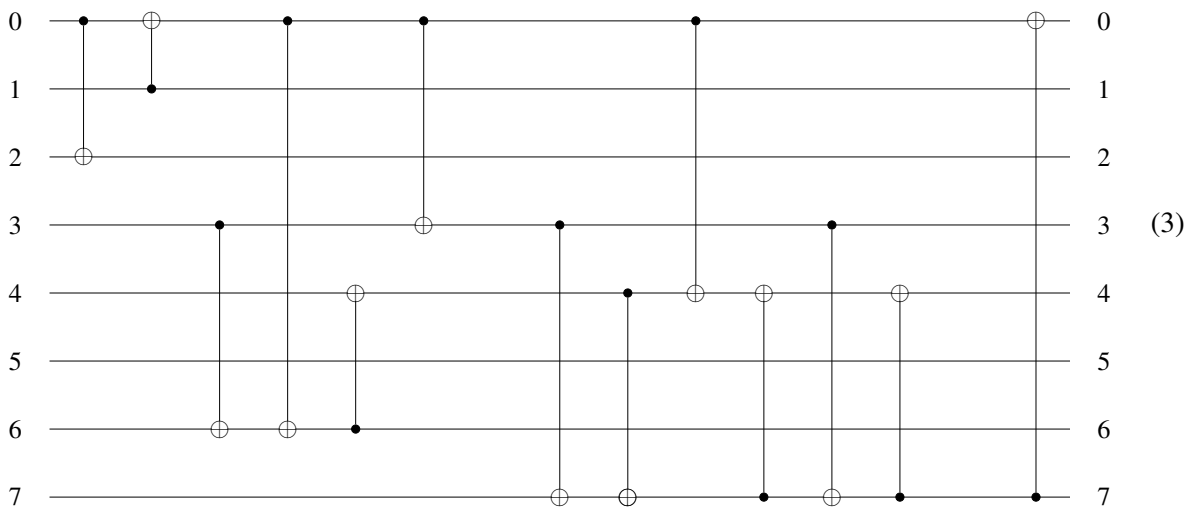
Rather than defining combs in general, we will focus on combs arising from quantum circuits where certain single-qubit gates have been removed. The restriction to single-qubit holes is not essential, but it suffices for our purposes and will make certain aspects of our algorithm simpler. These can be described as normal quantum circuits with some extra information about time ordering of qubits, subject to some constraints. To motivate the definition, we will look at an example circuit, consisting of CNOT gates and several single-qubit gates we wish to remove:



Note that we have labelled the inputs and outputs by the same indices $\{0, 1, 2, 3\}$. We call these labels for the qubits in our original circuit the *logical qubits*. If we remove the gates U, V, W, H from the circuit, we obtain something that looks like this:



We can break the data represented by the picture above into two parts. First, we can see this as just a normal quantum circuit \mathcal{C} , but now acting on more qubits:



Notice how this circuit is over more qubits than just the original logical qubits. The indices above refer to a qubit at a particular point in time, and hence have a many-to-one relationship with the logical qubits. We call these the *temporal qubits*.

There is also a binary relation telling us which temporal qubits come directly before others, which we call the *holes*. We can represent the holes indicated in (2) as the set $\mathcal{H} = \{(1, 4), (2, 6), (6, 7), (4, 5)\}$.

We can also define what it means to plug single-qubit gates into each of these holes. That is, for any set \mathcal{G} of single-qubit gates and a *plugging map* $p : \mathcal{H} \rightarrow \mathcal{G}$, we can unambiguously reconstruct a circuit. For example, we can reconstruct our original circuit using the plugging map

$$p :: \{(1, 4) \mapsto V, (4, 5) \mapsto H, (2, 6) \mapsto U, (6, 7) \mapsto W\}.$$

To define what it means to be a valid comb, we will formalise the process of plugging in gates, and require this to result in a well-defined circuit.

Algorithm 2: Comb composition

Input : A pair $(\mathcal{C}, \mathcal{H})$ of a circuit \mathcal{C} on a set of qubits $Q = \{0, \dots, n-1\}$ and a binary relation $\mathcal{H} \subseteq Q \times Q$, as well as a plugging function $p : \mathcal{H} \rightarrow \mathcal{G}$

Output: A circuit \mathcal{C}' or FAIL

For each $(q_1, q_2) \in \mathcal{H}$:

1. Re-order non-interacting gates in \mathcal{C} such that all gates on qubit q_1 appear before gates on qubit q_2 . If this is not possible, FAIL.
2. Insert $p((q_1, q_2))$ into \mathcal{C} directly before the first gate to refer to qubit q_2 .
3. Remove (q_1, q_2) from \mathcal{H} .
4. Rename $q_2 \mapsto q_1$ in \mathcal{C} and \mathcal{H} and remove q_2 from \mathcal{C} .
5. If this renaming produces a hole of the form $(q, q) \in \mathcal{H}$, then FAIL, otherwise continue to the next hole.

Return $\mathcal{C}' := \mathcal{C}$.

Definition 2.1. A pair $(\mathcal{C}, \mathcal{H})$ is called a *comb* if Algorithm 2 succeeds for any plugging map p .

Note that this Algorithm 2 can fail if the pair $(\mathcal{C}, \mathcal{H})$ introduces cyclic dependencies between temporal qubits. However, when we obtain such pairs by “cutting out” the non-CNOT gates from a circuit, this algorithm will always succeed. We can formalise this “cutting out” process with the following procedure:

3 Algorithm: CombSynth

Our main algorithm proceeds by decomposing a circuit into a comb consisting of just CNOT gates and a plugging map for all the additional single-qubit gates. It then resynthesises the comb using a variation of the RowCol algorithm which preserves the comb structure, then plugs the single-qubit gates back in to give a fully routed circuit.

The RowCol algorithm works by eliminating qubits one by one. To apply this idea to quantum combs we need to know what it means to remove one temporal qubit at a time. As temporal qubits represent sections of the “lifetime” of the original logical qubits, not all of them can exist at the same time. This means that, although we may write the comb as one large circuit, it is not possible to perform CNOTs between all the temporal qubits at any given time. This idea needs to be carried forward into the synthesis by restricting row operations to only be between qubits in the same sections of time.

Algorithm 3: Comb decomposition**Input** : A circuit \mathcal{C}' **Output:** A comb $(\mathcal{C}, \mathcal{H})$ and a plugging map $p : \mathcal{H} \rightarrow \mathcal{G}$.

1. Create an empty comb $(\mathcal{C}, \mathcal{H})$ where \mathcal{C} is an empty circuit with as many qubits as \mathcal{C}' and $\mathcal{H} = \{\}$.
2. Create a mapping t for temporal qubits, where initially $t(q) = q$ for all qubits $q \in \mathcal{C}'$ and p is the empty map $\mathcal{H} \rightarrow \mathcal{G}$.
3. For each gate g in \mathcal{C}' :
 - if g is a CNOT on qubits q_1 and q_2 , add g to \mathcal{C} on qubits $t(q_1), t(q_2)$
 - if g is not a CNOT and acting on qubit q , introduce a fresh qubit q' to \mathcal{C} , add $(t(q), q')$ to \mathcal{H} , set $p((t(q), q')) := g$, and let $t(q) := q'$
4. Return $(\mathcal{C}, \mathcal{H})$ and p .

For a comb $(\mathcal{C}, \mathcal{H})$, a temporal qubit q is called *available* if it does not appear as the first part of a hole. That is, there exists no q' such that $(q, q') \in \mathcal{H}$. It is *extractible* if its row and column can be made into a unit vector using the RowCol algorithm restricted only to row operations between available qubits.

Finally, to track topological constraints, which may be relevant for RowCol, we maintain a connection between logical and available temporal qubits. For each logical qubit q_0 , let $t(q_0)$ be its associated available temporal qubit. That is, let $t(q_0) = q_0$ if q_0 does not appear in any holes, otherwise, let it be q_k for the maximal transitive chain of holes $(q_0, q_1), (q_1, q_2), \dots, (q_{k-1}, q_k)$. Therefore, the mapping t is determined by a collection of holes, meaning as we update \mathcal{H} in the algorithm below we are also updating t .

Our main algorithm, CombSynth, works as follows:

Algorithm 4: CombSynth**Input** : A comb $(\mathcal{C}, \mathcal{H})$ with temporal qubits $Q = \{0, \dots, n-1\}$, topology graph $G(V, E)$ **Output:** A comb $(\mathcal{C}', \mathcal{H}')$, respecting topological constraints for any plugging p

1. Create a new comb $(\mathcal{C}', \mathcal{H}')$ with \mathcal{C}' initially empty and $\mathcal{H}' = \mathcal{H}$
2. Compute the parity matrix P of \mathcal{C} .
3. Identify an extractible temporal qubit e in comb $(\mathcal{C}, \mathcal{H})$.
4. Produce a rectangular sub-matrix P' with columns the same as P and rows labelled by $t(q)$ for each logical qubit q .
5. Run an iteration of RowCol on row $t(e)$ and column e of P' , with topology G , updating \mathcal{C}' .
6. Update the corresponding rows of P using P' .
7. Remove e from the qubits of \mathcal{C} and any hole of the form (e', e) for some e' from \mathcal{H} .
8. Repeat from Step 3 until no qubits remain in \mathcal{C} .
9. Return $(\mathcal{C}', \mathcal{H}')$.

Note that if $(\mathcal{C}, \mathcal{H})$ arose from a circuit by cutting single-qubit gates out, as in Algorithm 3, there will always be at least one extractible temporal qubit. We simply need to choose one corresponding to

the latest gate that has been cut out of the circuit.

The core of the RowCol algorithm is used to reduce the row and column of the chosen temporal qubit to unit vectors. Applying this procedure iteratively, removing the temporal qubits in an allowed order, and updating the sub-matrix as you go, will reduce the overall parity matrix to the identity whilst ensuring that no row operations happen between qubits that exist at different points in time. Therefore we have a procedure for synthesising quantum combs from their parity matrices. This method eliminates rows and columns of a matrix in the same way as RowCol, and the rows of rectangular sub-matrix P' always correspond to the same logical qubits, even though we are swapping the temporal qubits in and out. Hence, our algorithm can be used to route to constrained topologies in the same way as RowCol, but now it can be done for general unitaries by converting to a quantum comb. Comparisons of our generalisation of RowCol to a simple slicing procedure are presented in the next section.

$$\begin{array}{c}
 \begin{array}{cccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
 0 & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} & \color{blue}{1} & \color{blue}{1} & \color{blue}{0} & \color{blue}{1} & \color{blue}{1} \\
 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
 3 & \color{blue}{1} & \color{blue}{1} & \color{blue}{0} & \color{blue}{1} & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} \\
 4 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
 5 & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} & \color{blue}{0} & \color{blue}{1} & \color{blue}{0} & \color{blue}{0} \\
 6 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
 7 & \color{blue}{1} & \color{blue}{1} & \color{blue}{0} & \color{blue}{1} & \color{blue}{1} & \color{blue}{0} & \color{blue}{1} & \color{blue}{1}
 \end{array}
 \end{array}
 \xrightarrow{\text{Generate sub-matrix}}
 \begin{array}{c}
 \begin{array}{cccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
 5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 7 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
 3 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0
 \end{array}
 \end{array}$$

Figure 6: Generation of sub-matrix from full parity matrix

To illustrate how CombSynth works, we'll work through an example, showing each of the steps explicitly, similar to what we did for RowCol. In doing so, we will look at the circuit in (1), which has a quantum comb shown in (3). We will apply this process without topological constraints as these aren't necessary for showing how our generalisation works. The quantum comb is made of 8 temporal qubits, however, our initial circuit is made of 4 logical ones. This means that only 4 of the 8 temporal qubits in our quantum comb can exist at any one time, and a 4×8 rectangular matrix will then be used for the elimination steps. The sub-matrix will initially take the form in Figure 6. Note that the rows are not in the same order as they were in the larger matrix: this is because they are placed in the position of their corresponding logical qubit. A table of the elimination steps is presented in Figure 7, which highlights the row and column that get reduced to unit vectors and lists the row operations needed to do this. The elimination order of the temporal qubits is 5,7,6,3,4,0,1.

To complete the CombSynth algorithm, the comb generated from the row operations in Figure 7 is filled with the gates removed from the original circuit. This circuit, shown in Figure 8, is smaller than the original, though only by 1 gate, but this effect grows as the circuit size increases due to more CNOT cancellations.

4 Results

We have conducted a series of computational experiments to compare our generalisation of RowCol to a circuit slicing procedure. For the slicing we have used a naïve algorithm that cuts the circuit where the gates are found, similar to that shown in Figure 1b. 20 random circuits with CNOT counts of

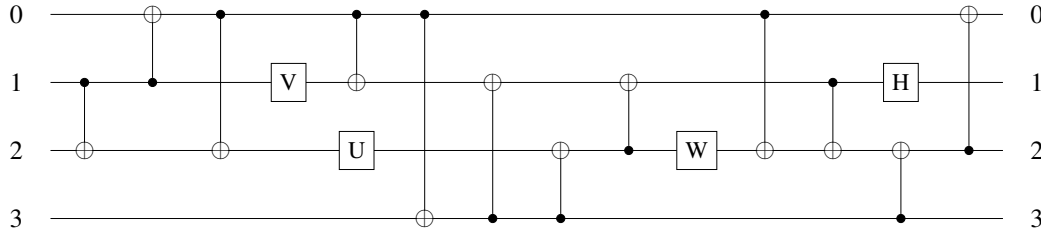


Figure 8: Circuit generated from the CombSynth procedure in Section 3

4, 8, 16, 32, 64, 128, 256, 512 and 1024 were generated, and a set of non-CNOT gates distributed uniformly throughout them. The number of non-CNOT gates was set to be proportional to the number of CNOTs, with the proportionality factor varying from 5% to 50%. The percentages of non-CNOTs in the table and graphs below are therefore the proportion of non-CNOTs to CNOTs, not the proportion of non-CNOTs to the total number of gates in the circuit. A range of architectures, popular for conducting similar computational experiments, was then selected to route our circuits onto. These were the 9-qubit square grid, 16-qubit square grid, Rigetti 16-qubit Aspen, 16-qubit IBM QX5 and 20-qubit IBM Tokyo. For each set of experimental parameters we recorded the proportional change in CNOT gates (CNOT overhead) due to the synthesis algorithms.

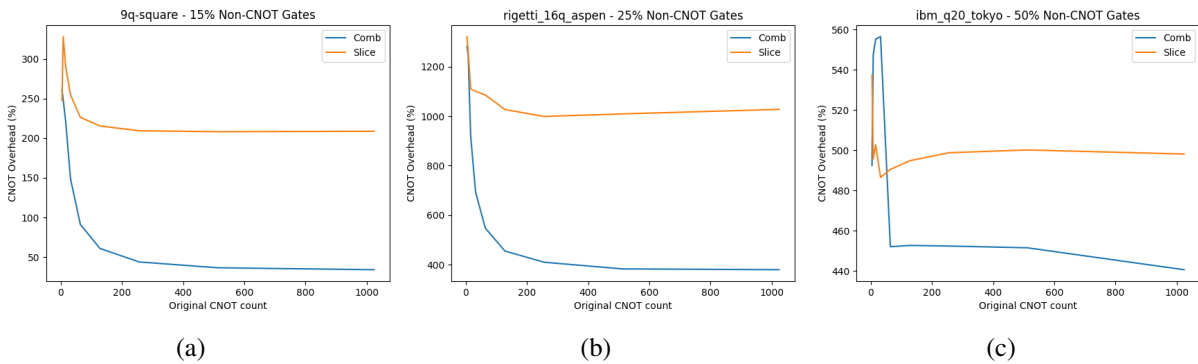


Figure 9: Graphs showing the change in CNOT overhead when increasing the size of the circuit. This selection of graphs was chosen because it covers a wide range of the parameters of the computational experiment, however all the graphs still broadly have the same behaviour. They start off with a larger overhead, fluctuate a bit for the small circuit sizes, then approach some constant value. With this approached value being smaller for the comb routing process then the slicing one.

A set of graphs with different parameters for the computational experiment is shown in Figure 9. These graphs are illustrative of the behaviour of all of the experiments: the overhead is high for small circuits with the slicing procedure sometimes being better, but as the circuit size increases both overheads approach a constant value, with the comb algorithm outperforming the slicing one. This is to be expected as using quantum combs allows for better cancellations than the slicing process, due to each slice being routed independently. The overheads for the largest circuit sizes, 1024 CNOTs, are shown in Table 1, giving a comparison of the asymptotic behaviour of each algorithm. It can be seen that larger proportions of non-CNOT gates reduce the advantage of using the comb algorithm: however it still outperforms the

slicing method for all architectures and gate proportions tested.

Architectures	5%		15%		25%		50%	
	Non-CNOT Gates		Non-CNOT Gates		Non-CNOT Gates		Non-CNOT Gates	
	Comb	Slice	Comb	Slice	Comb	Slice	Comb	Slice
9q-square	-43.1%	80.79%	34.12%	208.7%	91.93%	255.9%	182.1%	306.8%
16q-square	13.11%	344.5%	154.4%	511.1%	263.9%	564.6%	437.0%	606.9%
regetti_16q_aspen	47.31%	555.4%	231.2%	893.1%	379.6%	1027%	614.9%	1119%
bm_qx5	32.27%	461.9%	197.2%	698.8%	322.4%	783.6%	527.8%	837.7%
ibm_q20_tokyo	33.17%	393.1%	183.6%	481.0%	289.3%	500.2%	440.7%	498.2%

Table 1: CNOT overhead when routing to different architectures and proportions of non-CNOT gates. The values above are for the largest circuits tested: 1024 CNOTs

5 Conclusion and Future Work

We have proposed an alternative to slicing the circuit, using quantum combs, for generalising synthesis algorithms to arbitrary circuits. This idea was then concretely outlined for the case of CNOT synthesis by developing a generalisation of RowCol that works for quantum combs, and showing this allows the routing of arbitrary circuits. Finally, through a series of computational experiments, we demonstrated that for large circuits our quantum comb generalisation of RowCol outperforms the slicing procedure on a range of architectures and CNOT/non-CNOT proportions. Work has recently been done on improving the performance of RowCol by allowing the qubits to be permuted by the synthesis algorithm [10], investigating whether the performance of CombSynth could be improved in a similar way would be an interesting research direction. Currently, comb synthesis is designed to work with a subroutine that only works with one qubit (and hence one row/column) at a time, but it would be worth adapting it to work with synthesis algorithms which operate one more than one row or column at once, like the Patel-Markov-Hayes algorithm. Along a similar vein, it appears that our algorithm could be generalised to deal with multi-qubit holes by introducing some extra requirements than certain sets of temporal qubits should be extracted simultaneously.

Throughout this paper, we compared our quantum comb approach to that of slicing for generalising synthesis algorithms, however there have been some recent approaches that don't use slicing to perform circuit optimisation. These are lazy synthesis [12] and ZX circuit extraction [7], our approach differs from these as we focused on the utility of the higher order structure quantum combs for quantum compilation. Although a comparative analysis between our approach and these would be an interesting topic for further research. We focused on using quantum combs to generalise a CNOT synthesis algorithm in this paper: a natural next step would be to try and apply this idea to other synthesis algorithms such as the synthesis of Clifford circuits from stabiliser tableaux or CNOT+phase circuits from phase polynomials. Finally, quantum combs allow for routing of circuits with "black box" operations, where you may not know what operation will eventually be performed at the time of compilation. This may be useful for compilation in the context of parameterised circuits such as those used in variational algorithms, or for classical-quantum algorithms where you want to route around mid-circuit measurements.

References

- [1] Scott Aaronson & Daniel Gottesman (2004): *Improved simulation of stabilizer circuits*. *Physical Review A* 70(5), p. 052328, doi:10.1103/PhysRevA.70.052328.
- [2] Gernot Alber, Thomas Beth, Michał Horodecki, Paweł Horodecki, Ryszard Horodecki, Martin Rötteler, Harald Weinfurter, Reinhard Werner, Anton Zeilinger, Thomas Beth et al. (2001): *Quantum algorithms: Applicable algebra and quantum physics*. *Quantum information: an introduction to basic theoretical concepts and experiments*, pp. 96–150, doi:10.1007/3-540-44678-8_4.
- [3] Timothée Goubault de Brugière, Marc Baboulin, Benoît Valiron, Simon Martiel & Cyril Allouche (2022): *Decoding techniques applied to the compilation of CNOT circuits for NISQ architectures*. *Science of Computer Programming* 214, p. 102726, doi:10.1016/j.scico.2021.102726.
- [4] Giulio Chiribella, G Mauro D’Ariano & Paolo Perinotti (2008): *Quantum circuit architecture*. *Physical review letters* 101(6), p. 060401, doi:10.1103/PhysRevLett.101.060401.
- [5] Alexander Cowtan, Silas Dilkes, Ross Duncan, Alexandre Krajenbrink, Will Simmons & Seyon Sivarajah (2019): *On the Qubit Routing Problem*. In Wim van Dam & Laura Mancinska, editors: *14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019), Leibniz International Proceedings in Informatics (LIPIcs)* 135, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 5:1–5:32, doi:10.4230/LIPIcs.TQC.2019.5.
- [6] Jeroen Dehaene & Bart De Moor (2003): *Clifford group, stabilizer states, and linear and quadratic operations over $GF(2)$* . *Physical Review A* 68(4), p. 042318, doi:10.1103/PhysRevA.68.042318.
- [7] Ross Duncan, Aleks Kissinger, Simon Perdrix & John Van De Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.
- [8] Tilo Eggeling, Dirk Schlingemann & Reinhard F Werner (2002): *Semicausal operations are semilocalizable*. *Europhysics Letters* 57(6), p. 782, doi:10.1209/epl/i2002-00579-4.
- [9] Vlad Gheorghiu, Jiaxin Huang, Sarah Meng Li, Michele Mosca & Priyanka Mukhopadhyay (2022): *Reducing the CNOT count for Clifford+ T circuits on NISQ architectures*. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, doi:10.1109/TCAD.2022.3213210.
- [10] Arianne Meijer-van de Griend & Sarah Meng Li (2022): *dynamic qubit allocation and routing for constrained topologies by cnot circuit re-synthesis*. In: *Quantum Physics and Logic*, Electronic Proceedings in Theoretical Computer Science, doi:10.48550/arXiv.2205.00724.
- [11] Aleks Kissinger & Arianne Meijer (2020): *CNOT circuit extraction for topologically-constrained quantum memories*. *Quantum Information and Computation* 20, pp. 581–596, doi:10.26421/QIC20.7-8-4.
- [12] Simon Martiel & Timothée Goubault de Brugière (2022): *Architecture aware compilation of quantum circuits via lazy synthesis*. *Quantum* 6, p. 729, doi:10.22331/q-2022-06-07-729.
- [13] Dmitri Maslov & Martin Roetteler (2018): *Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations*. *IEEE Transactions on Information Theory* 64(7), pp. 4729–4738, doi:10.1109/TIT.2018.2825602.
- [14] Beatrice Nash, Vlad Gheorghiu & Michele Mosca (2020): *Quantum circuit optimizations for NISQ architectures*. *Quantum Science and Technology* 5(2), p. 025010, doi:10.1088/2058-9565/ab79b1.
- [15] Ketan N. Patel, Igor L. Markov & John P. Hayes (2008): *Optimal Synthesis of Linear Reversible Circuits*. *Quantum Info. Comput.* 8(3), pp. 282–294, doi:10.26421/QIC8.3-4-4.
- [16] Roberto Stassi, Mauro Cirio & Franco Nori (2020): *Scalable quantum computer with superconducting circuits in the ultrastrong coupling regime*. *npj Quantum Information* 6(1), p. 67, doi:10.1038/s41534-020-00294-x.
- [17] Maarten Van Den Nest (2010): *Classical Simulation of Quantum Computation, the Gottesman-Knill Theorem, and Slightly Beyond*. *Quantum Info. Comput.* 10(3), p. 258–271, doi:10.26421/QIC10.3-4-6.

- [18] Bujiao Wu, Xiaoyu He, Shuai Yang, Lifu Shou, Guojing Tian, Jialin Zhang & Xiaoming Sun (2023): *Optimization of CNOT circuits on limited-connectivity architecture*. *Phys. Rev. Res.* 5, p. 013065, doi:10.1103/PhysRevResearch.5.013065.

Picturing Counting Reductions with the ZH-Calculus

Tuomas Laakkonen^{1,a}, Konstantinos Meichanetzidis^{1,b}, John van de Wetering^{2,c}

¹ Quantinuum, 17 Beaumont Street, Oxford OX1 2NA, United Kingdom

² Informatics Institute, University of Amsterdam, 1098 XH Amsterdam, The Netherlands

^{a,b} {tuomas.laakkonen, k.mei}@quantinuum.com, ^c john@vdwetering.name

Counting the solutions to Boolean formulae defines the problem **#SAT**, which is complete for the complexity class **#P**. We use the ZH-calculus, a universal and complete graphical language for linear maps which naturally encodes counting problems in terms of diagrams, to give graphical reductions from **#SAT** to several related counting problems. Some of these graphical reductions, like to **#2SAT**, are substantially simpler than known reductions via the matrix permanent. Additionally, our approach allows us to consider the case of counting solutions modulo an integer on equal footing. Finally, since the ZH-calculus was originally introduced to reason about quantum computing, we show that the problem of evaluating scalar ZH-diagrams in the fragment corresponding to the Clifford+T gate set, is in **FP^{#P}**. Our results show that graphical calculi represent an intuitive and useful framework for reasoning about counting problems.

Graphical calculi like the ZX-calculus [15, 16] are seeing increased usage in reasoning about quantum computations. While earlier work in this area has mostly focused on *representing* existing quantum protocols and quantum algorithms in a graphical way in order to shed light on how these protocols work [17, 18, 26, 27, 33, 38, 39, 57, 66], recent years have seen the development of entirely new results that improve upon the existing state-of-the-art. For instance, there are now new results proved with a graphical calculus in quantum circuit optimization [5, 9, 20, 21, 25, 42], verification [13, 42, 49] and simulation [14, 43, 44, 48, 61], as well as new protocols in measurement-based quantum computing [4, 11, 41], surface codes [6, 30–32, 37] and other fault-tolerant architectures [51, 58].

These results in quantum computing show that diagrammatic reasoning can lead to new insights and algorithms that go beyond what is known or what even can be derived using other methods. However, these graphical languages are in actuality not restricted to just studying quantum computing. In fact, diagrams, the objects of a graphical calculus, can represent arbitrary tensor networks, which can represent arbitrary 2^n -dimensional tensors and so they can be used for a wide variety of problems. Whereas one would in general perform tensor contractions in order to compute with tensor networks, a graphical calculus equips its diagrams with a formal rewrite system, which respects their tensor semantics, and allows for reasoning in terms of two-dimensional algebra.

In this work, we focus on *counting problems* which are of both practical and theoretical importance for a variety of domains, from computing partition functions in statistical mechanics [52], to probabilistic reasoning [56] and planning [10]. The computational complexity of counting problems is of fundamental interest to computer science [55]. Counting problems also have a natural tensor network representation [29], and the complexity of computing with tensor networks has been thoroughly studied [23]. In practice, tensor contraction algorithms for counting problems have been developed, showing competitive performance against the state of the art [34, 45].

Graphical languages like the ZX-calculus, and its close relative, the ZH-calculus, have been used to rederive complexity-theoretic results. Townsend-Teague *et al.* [59] showed that the partition function of a family of Potts models, related to knot theory and quantum computation, is efficiently computable. de Beaudrap *et al.* [24] proved graphically that the decision version of a hard counting problem can be

solved in polynomial time. These proofs are constructive, in that they introduce algorithms in terms of rewriting strategies. Even though this line of work recasts known results in a graphical language, such an approach is arguably more unifying and intuitive, and thus has promising potential for generalization. Recent work by Laakkonen *et al.* [48] actually derived a *novel* complexity-theoretic result in the form of an improved runtime upper bound for counting problems. To obtain this result, reductions to specific counting problems were given a fully graphical treatment, to which then a known algorithm could be applied, after this algorithm was also treated graphically and generalized.

In this work, we continue building on this programme of applying graphical methods to counting. Specifically, we use the ZH-calculus to rederive various counting reductions that appear in the literature, providing a unified, and arguably simpler, presentation. Among others, we give reductions from **#SAT** to **#2SAT**, **#Planar-SAT** and **#Monotone-SAT**. See Table 1 for an overview. Our direct proof that **#2SAT** is **#P**-complete also allows us to considerably simplify the proof that computing the permanent of an integer matrix is **#P**-complete. Our results show that graphical languages can form a useful tool for the study of counting complexity.

In Section 1 we introduce the basics of counting complexity, the ZH-calculus and how to represent **#SAT** in ZH. Then in Section 2 we present our main reductions from **#SAT** by rewriting ZH-diagrams. Section 3 considers the converse problem of reducing ZH-diagram evaluation to **#SAT**. We conclude in Section 4, but note that we also present some additional reductions and proofs in the appendices.

1 Preliminaries

1.1 Counting reductions

Counting complexity is defined in terms of the complexity classes **#P** and **#_MP**, which are the ‘counting analogues’ of **NP**. The class **#P**, first defined by Valiant in 1979 [62], is the class of problems which can be defined as counting the number of accepting paths to a non-deterministic Turing machine (NTM) which halts in polynomial time, whereas **#_MP** is the class of problems which can be defined as counting, modulo M , the number of accepting paths to an NTM (that also halts in polynomial time). Note that the notation $\bigoplus \mathbf{P}$ is also used to indicate **#₂P**. These complexity classes are clearly related to **NP**, which consists of problems that can be defined as deciding whether an NTM has *any* accepting path.

Famously, the Boolean satisfiability problem **SAT** is **NP**-complete [19]. Similarly, there are notions of **#P**-completeness and **#_MP**-completeness [62]. A problem \mathcal{A} is **#P**-hard (**#_MP**-hard) if any problem in **#P** (**#_MP**) can be solved in polynomial time given an oracle for \mathcal{A} (that is, there exists a Cook reduction from any problem in **#P** to \mathcal{A}). A problem \mathcal{A} is **#P**-complete (**#_MP**-complete) if it is both **#P**-hard (**#_MP**-hard) and is in **#P** (**#_MP**).

Definition 1. Suppose $\phi : \mathbb{B}^n \rightarrow \mathbb{B}$ is a Boolean formula in Conjunctive Normal Form (CNF),

$$\phi(x_1, \dots, x_n) = \bigwedge_{i=1}^m (c_{i1} \vee c_{i2} \vee \dots \vee c_{ik_i}) \quad (1)$$

where $c_{ij} = x_l$ or $\neg x_l$ for some l , and let $\#(\phi) = |\{\vec{x} \mid \phi(\vec{x}) = 1\}|$. Each argument to ϕ is called a *variable* and each term $c_{i1} \wedge \dots \wedge c_{ik_i}$ a *clause*. Then, we define the following problems:

1. **SAT**: Decide whether $\#(\phi) > 0$,
2. **#SAT**: Compute the value of $\#(\phi)$,
3. **#_MSAT**: Compute the value of $\#_M(\phi) := \#(\phi) \bmod M$.

We additionally define variants, **kSAT**, **#kSAT**, **#_MkSAT**, which represent the case where ϕ is restricted to contain only clauses of size at most k (note that some sources take this to be size *exactly* k , but we can recover this from our definition by adding dummy variables to each clause). We also take \oplus **SAT** as alternate notation for **#₂SAT**.

To each formula ϕ we associate two graphs: the *incidence graph* is a bipartite graph with one vertex for each variable and one for each clause, and where a variable vertex is connected to a clause vertex if it occurs in that clause. The *primal graph* has one vertex for each variable, which are connected together if the variables occur together in a clause.

The Cook-Levin theorem [19] shows that **kSAT** is **NP**-complete for $k \geq 3$, but in fact also shows that **#kSAT** is **#P**-complete and **#_MkSAT** is **#_MP**-complete for any M , as it maps any NTM into a Boolean formula such that the number of satisfying assignments is exactly equal to the number of accepting paths. We will consider variants on these problems, and specifically the case where the structure of the formula ϕ is restricted in some way. For each of these variants, we will append a prefix to **SAT** to indicate the restriction:

- **PL**: The incidence graph of the formula is planar.
- **MON**: The formula is monotone - it contains either no negated variables or no unnegated variables.
- **BI**: The primal graph is bipartite - the variables can be partitioned into two sets such that each clause contains at most one variable from each set.

1.2 The ZH-calculus

The ZH-calculus is a rigorous graphical language for reasoning about ZH-diagrams in terms of rewriting [2]. We will give here a short introduction, referring the reader to [69, Section 8] for a more in-depth explanation.

ZH-diagrams represent tensor networks [54, Section 4.1] composed of the two generators of the language, the Z-spider and the H-box. The generators and their corresponding tensor interpretations are

$$\begin{aligned}
 \begin{array}{c} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_n \end{array} &= \delta_{\sigma_1 \sigma_2 \dots \sigma_n} = \begin{cases} 1 & \sigma_1 = \sigma_2 = \dots = \sigma_n \\ 0 & \text{otherwise} \end{cases} \\
 \begin{array}{c} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_n \end{array} &= 1 + (a - 1)\delta_{\sigma_1 \sigma_2 \dots \sigma_n} = \begin{cases} a & 1 = \sigma_1 = \sigma_2 = \dots = \sigma_n \\ 1 & \text{otherwise} \end{cases}
 \end{aligned} \tag{2}$$

where the H-box is labeled with a constant $a \in \mathbb{C}$, and we assume $a = -1$ if not given. The tensors corresponding to the generators are composed according to the tensor product and each wire connecting two tensors indicates a contraction, i.e. a summation over a common index [69]. We will also use two derived generators - the Z-spider with a phase, and the X-spider. These are given in terms of the other generators as:

$$\begin{array}{c} \alpha \\ \vdots \end{array} = \begin{array}{c} \boxed{e^{i\alpha}} \\ \vdots \end{array} \quad \begin{array}{c} \alpha \\ \vdots \end{array} = \begin{array}{c} \boxed{\frac{1}{2}} \\ \alpha \\ \vdots \end{array} \tag{3}$$

Note that the tensors are symmetric under permutation of their wires, or indices. This implies that only the connectivity, or the topology, of the tensor network matters. In particular, we will not distinguish

indices of generators as inputs and outputs as in [2]. Any ZH-diagram with n open wires therefore represents a tensor with n indices. In the special case of no open wires this represents a scalar, and we will call such diagrams scalar diagrams.

The rewriting rules of the ZH-calculus are shown in Appendix A. The rules are *sound*, i.e. they respect the tensor semantics, and also *complete* for complex-valued linear maps, i.e. if two ZH-diagrams represent the same tensor, then there exists a sequence of rewrites which transforms one diagram to the other.

1.3 #SAT instances as ZH-diagrams

To embed #SAT instances into ZH-diagrams, we use the translation of de Beaudrap *et al.* [24] where each variable becomes a Z-spider, each clause a zero-labeled H-box, and X-spiders are used for negation. In particular the mapping is as follows

$$\text{Variables} \iff \begin{array}{c} \cdots \\ \circ \end{array} \quad \text{Clauses} \iff \begin{array}{c} \boxed{0} \\ \pi \quad \pi \\ \cdots \end{array} \quad \text{Negation} \iff \begin{array}{c} \pi \\ | \end{array} \quad (4)$$

and to form #SAT instances, we combine these as

$$\#(\phi) = \begin{array}{c} \boxed{0} \quad \cdots \quad \boxed{0} \\ \pi \quad \cdots \quad \pi \quad \pi \quad \cdots \quad \pi \\ \hline G \\ \cdots \quad \cdots \quad \cdots \end{array} \quad (5)$$

where G is a collection of wires and negations, connecting each variable to its corresponding clauses. Due to cancellation of adjacent X-spiders, an instance has an X-spider between a variable and a clause if the variable appears *unnegated* in that clause, and a wire if it appears negated. For example, for the formula $\phi(x_1, x_2, x_3) = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$, we have:

$$\#(\phi) = \begin{array}{c} \boxed{0} \quad \boxed{0} \quad \boxed{0} \\ \pi \quad \pi \quad \pi \\ \swarrow \quad \downarrow \quad \downarrow \\ \circ \quad \circ \quad \circ \\ x_1 \quad x_2 \quad x_3 \end{array} \quad (6)$$

In this representation, a formula that is planar corresponds to a planar ZH-diagram and a monotone one corresponds to a ZH-diagram where there are no X-spiders or where there is an X-spider between every H-box and Z-spider. Instances with maximum clause size k correspond to ZH-diagrams where every H-box has degree at most k .

2 Reductions from #SAT

We will show using the ZH-calculus that the restricted versions of #SAT defined above—planar, monotone or bipartite—are #P- and/or #_MP-complete - Table 1 gives an overview of our reductions. All these results are already known in the literature, as will be discussed in each section, but our main contribution is to provide a simplifying and unifying viewpoint through the use of the ZH-calculus.

<i>Result</i>	<i>Reduction</i>	$\#_k\mathbf{P}$	\mathbf{NP}	$\mathbf{PL-}$	$\#2\mathbf{SAT}$	$\mathbf{MON-}$	$\mathbf{BI-}$	$\mathbf{3DEG-}$
Theorem 1	$\#SAT \rightarrow \#PL-SAT$	✓	✓	✓				
Theorem 2	$\#SAT \rightarrow \#2SAT$	✓		✓	✓		✓	
Theorem 3	$\#SAT \rightarrow \#MON-SAT$	†		✓	✓	✓		
Theorem 6	$\#SAT \rightarrow \#3DEG-SAT$	✓		✓	✓	✓	✓	✓

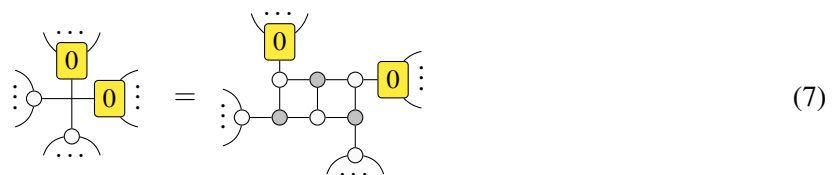
Table 1: An overview of the main reductions presented in this paper. The two leftmost columns give each theorem and the corresponding reduction. The middle columns (marked $\#_k\mathbf{P}$, and \mathbf{NP}) are given a checkmark if the corresponding reduction is valid for that complexity class as well as for $\#P$. A dagger is written for $\#_k\mathbf{P}$ if there are some additional restrictions placed on k . The rightmost columns (marked $\mathbf{PL-}$, etc) show what structure each reduction preserves - a checkmark is given if the corresponding reduction preserves the properties of the given $\#SAT$ variant (here $\#2SAT$ indicates that the maximum clause size is two), i.e. the reduction presented in Theorem 2 sends planar instances to planar instances, but does not send monotone instances to monotone instances. This applies for each complexity class that reduction is valid for (e.g Theorem 2 also implies a reduction $\#_k\mathbf{PL-SAT} \rightarrow \#_k\mathbf{PL-2SAT}$).

2.1 $\#SAT \rightarrow \#PL-SAT$

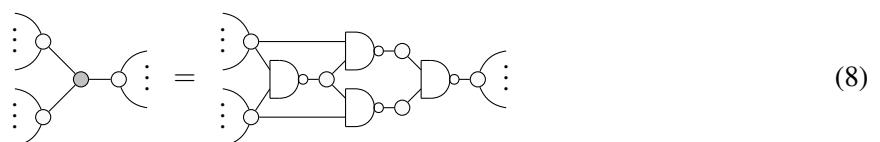
The first, and most commonly taught, proof that $\mathbf{PL-SAT}$ is \mathbf{NP} -complete was published in 1982 by Lichtenstein [50]. This reduction is parsimonious - every satisfying assignment of the original formula corresponds to one satisfying assignment of the planar formula. Hence, this proves also that $\#PL-SAT$ and $\#_M\mathbf{PL-SAT}$ are complete for $\#P$ and $\#_M\mathbf{P}$. Lichtenstein’s construction uses a large gadget to eliminate non-planarity. In the following construction, we derive a similar gadget from first principles, by building on a famous identity from quantum computing.

Lemma 1. For any $\phi \in \#kSAT$ with n variables and m clauses and $k \geq 3$, there is a planar $\phi' \in \#kSAT$ such that $\#(\phi) = \#(\phi')$. Furthermore, ϕ' has $O(n^2m^2)$ variables and clauses, and is computable in $O(\text{poly}(n, m))$ time.

Proof. Any instance $\phi \in \#kSAT$ can be drawn in the plane as a ZH-diagram with some number of crossing wires. By using the famous identity that a SWAP gate can be written as the composition of 3 CNOTs, we have that [53]:



We now need to rewrite the X-spider, which represents a classical XOR function, into CNF for this to be a valid $\#SAT$ instance. Unfortunately, the direct translation via the Tseytin transformation [60] does not preserve planarity. However, we can instead use the following decomposition of an XOR as NAND gates, which is planar:



Finally, NAND gates themselves have the following planar Tseytin transformation [60] into CNF:

$$(9)$$

Therefore, applying this to ϕ gives ϕ' with $n + 12c$ variables and $m + 36c$ clauses where c is the number of crossings. If the ϕ is drawn with straight-line wires only, then since there are at most nm wires in the diagram and each pair can cross at most once, we have $c \leq O(n^2m^2)$. As this rewrite introduces only clauses of size three or less, ϕ' is still a **#kSAT** instance. \square

Theorem 1. We have the following:

1. **#PL-kSAT** and **#PL-SAT** are **#P**-complete for any $k \geq 3$.
2. **#_MPL-kSAT** and **#_MPL-SAT** are **#_MP**-complete for any $M \geq 2$ and $k \geq 3$.
3. **PL-kSAT** and **PL-SAT** are **NP**-complete for any $k \geq 3$.

Proof.

1. This follows immediately from Lemma 1 since the size of the rewriting does not depend on the clause size.
2. This also follows from Lemma 1, since $\#(\phi) = \#(\phi')$ implies $\#_M(\phi) = \#_M(\phi')$ for any M .
3. This follows immediately from Lemma 1 since if $\#(\phi) = \#(\phi')$ implies that ϕ is satisfiable if and only if ϕ' is satisfiable. \square

2.2 #SAT to #2SAT

While it is known that **#2SAT** is **#P**-complete [64], the proof by Valiant relies on a chain of reductions from **#SAT** to the permanent of an integer matrix, to the permanent of a binary matrix, to counting perfect matchings in graphs, to counting all matchings in graphs, and then finally to **#MON-BI-2SAT**. Moreover, this proof does not generalize to the case of **#_MP** - in fact, proof that $\oplus 2SAT$ is $\oplus P$ -complete was only shown 27 years later in 2006 using a completely different method of holographic reductions [65], and then a reduction for any fixed M was given in 2008 by Faben [28]. In this section we give a simple direct reduction from **#SAT** to **#2SAT** that applies both for **#P** and **#_MP**.

Lemma 2 [48, Lemma 3.3]. The following equivalence holds:

$$(10)$$

Lemma 3. For any $M = 2^r + 1$ with $r \in \mathbb{N}$ and $\phi \in \#_M \mathbf{SAT}$ with n variables and m clauses, there is a $\phi' \in \#_M \mathbf{2SAT}$ with $O(n + mr)$ variables such that $\#_M(\phi) = \#_M(\phi')$, and ϕ' can be computed in $O(\text{poly}(n, m, r))$ time.

Proof. By evaluating the tensors, we have $\circ \boxed{0} \pi - = \boxed{2} -$ and therefore:

$$\begin{aligned}
 \circ \boxed{0} &= \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \pi = \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \boxed{-1} \equiv \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \boxed{2^r} \pmod{2^r + 1} \\
 &\stackrel{M}{=} \left. \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \left. \begin{array}{c} \boxed{2} \\ \vdots \\ \boxed{2} \end{array} \right\} r \text{ copies} = \left. \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \left. \begin{array}{c} \pi \boxed{0} \\ \vdots \\ \pi \boxed{0} \end{array} \right\} r \text{ copies}
 \end{aligned} \tag{11}$$

In this way we can rewrite all of the clauses in ϕ to form a suitable ϕ' . □

Lemma 4. For any $M > 2$ and $\phi \in \#_M\text{SAT}$ with n variables and m clauses, there is a $\phi' \in \#_M\text{2SAT}$ with $O(n + mM)$ variables such that $\#_M(\phi) = \#_M(\phi')$, and ϕ' can be computed in $O(\text{poly}(n, m, M))$ time.

Proof. By evaluating the tensors, we have $\boxed{r} \boxed{0} \pi - = \boxed{r+1} -$ for all $r \in \mathbb{C}$. Therefore:

$$\begin{aligned}
 \circ \boxed{0} &= \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \pi = \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \boxed{-1} \equiv \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \boxed{M-1} \pmod{M} \\
 &= \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \underbrace{\begin{array}{c} \pi \boxed{0} \pi \boxed{0} \dots \pi \boxed{0} \boxed{1} \end{array}}_{M-2 \text{ copies}} \\
 &\stackrel{I,U}{=} \begin{array}{c} \vdots \\ \pi \boxed{0} \\ \vdots \end{array} \circ \underbrace{\begin{array}{c} \pi \boxed{0} \circ \pi \boxed{0} \circ \dots \circ \end{array}}_{M-2 \text{ copies}}
 \end{aligned} \tag{12}$$

In this way we can rewrite all of the clauses in ϕ to form a suitable ϕ' . □

Theorem 2. We have the following:

1. $\#_M\text{2SAT}$ is $\#_M\text{P}$ -complete for any $M \geq 2$.
2. $\#2\text{SAT}$ is $\#\text{P}$ -complete.

Proof.

1. If $M = 2$, this follows from Lemma 3 with $r = 0$. If $M > 2$, then since M is fixed, this follows from Lemma 4.
2. For any $\phi \in \#\text{SAT}$ with n variables, note that $0 \leq \#(\phi) \leq 2^n$. Hence $\#(\phi) = \#_{2^n+1}(\phi)$, and so we can apply Lemma 3 with $r = n$ to generate $\phi' \in \#2\text{SAT}$ such that $\#(\phi) = \#_{2^n+1}(\phi') = \#(\phi') \pmod{2^n + 1}$ in polynomial time, giving a polynomial-time counting reduction from $\#\text{SAT}$ to $\#2\text{SAT}$. □

Corollary 1. $\#_M\text{PL-2SAT}$ is $\#_M\text{P}$ -complete for any $M \geq 2$, and $\#\text{PL-2SAT}$ is $\#\text{P}$ -complete.

Proof. Note that the reductions given in Lemmas 3 and 4 preserve the planarity of the input instance. Hence this follows by first applying Lemma 1 and then Theorem 2. □

Corollary 2. $\#_M\text{BI-2SAT}$ is $\#_M\text{P}$ -complete for any $M \geq 2$, and $\#\text{BI-2SAT}$ is $\#\text{P}$ -complete.

Proof. When we apply Lemmas 3 and 4, the $\#2\text{SAT}$ instance obtained will always be bipartite, so this follows from Theorem 2. We can see this as the primal graph has vertices in two groups: the set V of vertices corresponding to variables of the original formula, and the sets C_i of the vertices introduced to decompose clauses. The subgraph for each C_i is clearly bipartite, so let C_i^A and C_i^B be the corresponding partition. Each vertex in V only connects to at most one vertex c_i in each C_i , and assume without loss of generality that $c_i \in C_i^A$. Then the whole graph can be partitioned as $V \cup C_1^B \cup \dots \cup C_m^B$ and $C_1^A \cup \dots \cup C_m^A$, so it is bipartite. \square

2.3 $\#SAT \rightarrow \#MON-SAT$

While in the previous section we showed that $\#2\text{SAT}$ was $\#P$ -complete, other proofs [64] of this fact actually consider the subset $\#MON-BI-2SAT$. In this section we give a reduction from $\#SAT$ to $\#MON-SAT$, allowing us to remove negations from any CNF formula. This shows that our graphical method is not any less powerful than the reduction via the permanent, and we argue that this chain of reductions is more intuitive because it allows us to gradually restrict the formulae, rather than jumping straight to a highly restrictive variant.

Lemma 5. For any $r \geq 0$ and $\phi \in \#_{2^r}\text{SAT}$ with n variables, m clauses, and maximum clause size at least two, there is a monotone $\phi' \in \#_{2^r}\text{SAT}$ with $O(n + nmr)$ variables and $O(m + nmr)$ clauses such that $\#_{2^r}(\phi) = \#_{2^r}(\phi')$. Additionally, ϕ' preserves the maximum clause size of ϕ , and can be computed in $O(\text{poly}(n, m, r))$ time.

Proof. By evaluating the tensors, we have $\boxed{2} = \pi \boxed{0} \pi$ and therefore,

$$\begin{aligned}
 \pi &= \pi \boxed{0} \pi \equiv \pi \boxed{0} \pi \equiv \pi \boxed{2^r} \pi \pmod{2^r} \quad (13) \\
 &\stackrel{M}{=} \pi \boxed{2} \dots \boxed{2} \pi \equiv \pi \boxed{0} \dots \boxed{0} \pi \equiv \pi \boxed{0} \dots \boxed{0} \pi \pmod{2^r}
 \end{aligned}$$

where the first equality follows from the Tseytin transformation of the NOT gate [60]. Thus we can remove every negation in ϕ as follows:

$$\begin{aligned}
 \pi \boxed{0} \pi &\stackrel{SFZ}{=} \pi \boxed{0} \pi \equiv \pi \boxed{0} \pi \pmod{2^r} \quad (14) \\
 &\quad \left. \begin{array}{c} \boxed{0} \pi \boxed{0} \\ \boxed{0} \pi \boxed{0} \\ \vdots \\ \boxed{0} \pi \boxed{0} \end{array} \right\} r \text{ copies}
 \end{aligned}$$

There are at most nm negations in ϕ , and each can be rewritten with $O(r)$ clauses and variables. Note that this rewrite introduces only clauses of size two, so the maximum clause size is preserved. \square

Theorem 3. We have the following:

1. $\#_{2^r}\text{MON-kSAT}$ and $\#_{2^r}\text{MON-SAT}$ are $\#_{2^r}\text{P}$ -complete for any $r \geq 0$ and $k \geq 2$.
2. $\#MON-kSAT$ and $\#MON-SAT$ are $\#P$ -complete for any $k \geq 2$.

Proof.

1. This follows immediately from Lemma 5. Since the transformation preserves maximum clause size, this holds for either bounded or unbounded clause size.
2. For any $\phi \in \#\mathbf{kSAT}$ with n variables and $k \geq 2$, note that $0 \leq \#(\phi) \leq 2^n$. Hence $\#(\phi) = \#_{2^{n+1}}(\phi)$, and so we can apply Lemma 5 with $r = n + 1$ to generate $\phi' \in \#\mathbf{MON-kSAT}$ such that $\#(\phi) = \#_{2^{n+1}}(\phi') = \#(\phi') \bmod 2^{n+1}$ in polynomial time, giving a polynomial-time counting reduction from $\#\mathbf{kSAT}$ to $\#\mathbf{MON-kSAT}$. The same argument also gives a reduction for $\#\mathbf{MON-SAT}$. \square

2.4 Other Reductions

Using similar methods we can also consider other restrictions of $\#\mathbf{SAT}$. For example, in Appendix B, we combine Theorems 1, 2, and 3 with further reductions to show that $\#\mathbf{MON-BI-PL-3DEG-2SAT}$, where $\mathbf{3DEG-}$ indicates each variable participates in at most three clauses, is $\#\mathbf{P}$ -complete. This case is interesting because it is as small as possible - if we instead have each variable participate in only two clauses then this is in \mathbf{P} [22]. Indeed most upper bounds on runtime for $\#\mathbf{2SAT}$ have better special case for this type of formula [22, 67].

As phase-free ZH-diagrams naturally encode $\#\mathbf{SAT}$ instances, the ZH-calculus is mostly suited to treat variations on the $\#\mathbf{SAT}$ problem. To apply the technique of graphical reasoning to other (counting) problems, we hence may need to use other graphical calculi. In particular, in Appendix C, we show how the ZW-calculus [35] is naturally adapted to both the $\#\mathbf{XSAT}$ problem (of which $\#\mathbf{1-in-3SAT}$ is a special case) and the $\#\mathbf{PERFECT-MATCHINGS}$ problem, and use this shared structure to give graphical reductions showing that both are $\#\mathbf{P}$ -complete. This complements the recent result of Carette *et al.* illustrating with the ZW-calculus that $\#\mathbf{PLANAR-PERFECT-MATCHINGS}$ is in \mathbf{P} [12].

While the reductions given above contribute to simplifying the literature in their own right, we can also derive other simplifications from them. For example, the original proof by Valiant [62] (and the simplification by Ben-Dor and Halevi [7]) that computing the permanent of a boolean matrix is $\#\mathbf{P}$ -complete relies on a reduction from $\#\mathbf{3SAT}$. It would be simpler to reduce from $\#\mathbf{MON-2SAT}$, but the original proof that $\#\mathbf{MON-2SAT}$ is $\#\mathbf{P}$ -complete relies on a reduction from the permanent, so this would be circular.

However, by Theorems 2 and 3 we have $\#\mathbf{P}$ -completeness for $\#\mathbf{MON-2SAT}$ independent of the permanent. This then allows us to give an alternate, simpler proof that computing the permanent of an integer matrix is $\#\mathbf{P}$ -complete, which we present in Appendix D. In the original proof of Ben-Dor and Halevi [7], they construct for a given $\#\mathbf{3SAT}$ instance a weighted directed graph with two cycles per variable and a gadget of seven vertices for each clause such that the permanent of the adjacency matrix of the graph equals the value of the $\#\mathbf{3SAT}$ instance. As finding a suitable clause gadget was difficult, they found a suitable one using computer algebra. Our proof adapts theirs, but as we can start from a $\#\mathbf{MON-2SAT}$ instance, our graph can be made simpler, only requiring one cycle per variable, and a symmetric clause gadget of just four vertices. This was found, and can easily be proven correct, by hand.

3 Evaluating Scalar ZH-Diagrams

While we have so far shown that variants of $\#\mathbf{SAT}$ can be embedded into ZH-diagrams, and thus that the problem of evaluating an arbitrary scalar ZH-diagram is $\#\mathbf{P}$ -hard, we haven't yet answered how much harder it might be. I.e whether this problem is in $\#\mathbf{P}$. In this section we will show that evaluating scalar ZH-diagrams comprised of a certain fragment of generators is complete for $\mathbf{FP}^{\#\mathbf{P}}$. \mathbf{FP} is the class of

functions that can be evaluated in polynomial time by a deterministic Turing machine (i.e the function analog of \mathbf{P}), and $\mathbf{FP}^{\#\mathbf{P}}$ is thus the class of functions that can be evaluated in polynomial time by a deterministic Turing machine with access to an oracle for a $\#\mathbf{P}$ -complete problem (in our case we will use $\#\mathbf{SAT}$).

In order to consider the problem of evaluating scalar ZH-diagrams formally, we first define the problem $\text{Eval-}F$ which is the task of finding the complex number corresponding to a scalar diagram that exists in fragment F of a graphical calculus. A fragment is a set of diagrams built from arbitrary combinations of a fixed subset of generators.

Definition 2. For a given fragment F , the problem $\text{Eval-}F$ is defined as follows:

Input A scalar diagram $D \in F$ consisting of n generators and wires in total, where any parameters of the generators of D can be expressed in $O(\text{poly}(n))$ bits.

Output The value $D \in \mathbb{C}$.

The runtime of an algorithm for $\text{Eval-}F$ is defined in terms of the parameter n .

We will examine two fragments $\text{ZH}_{\pi/2^k} \supseteq \text{ZH}_{\pi}$ and show that they can be reduced to $\#\mathbf{SAT}$. Far from being purely academic, ZH_{π} is expressive enough to capture Toffoli-Hadamard quantum circuits, and $\text{ZH}_{\pi/2^k}$ can additionally capture Clifford+T quantum circuits, both of which are approximately universal for quantum computation.

Definition 3. ZH_{π} is the fragment of ZH-calculus given by the following generators:

(15)

Lemma 6. The following diagram equivalence holds:

(16)

This is derived from the Tseytin transformation of the XOR operation [60].

Theorem 4. There is a polynomial-time counting reduction from the problem Eval-ZH_{π} to $\#\mathbf{3SAT}$ and so Eval-ZH_{π} is in $\mathbf{FP}^{\#\mathbf{P}}$. Note that Eval-ZH_{π} is equivalent to the problem $\#\mathbf{SAT}_{\pm}$ as defined in [48].

Proof. In order to rewrite a diagram D from ZH_{π} into $\#\mathbf{3SAT}$, we first rewrite all of the non-scalar H-boxes into zero H-boxes with two legs:

(17)

Where the second equality follows from Lemma A.4 in [48]. Now, we can remove all the X-spiders and π -phase Z-spiders as follows, to rewrite into a valid $\#\mathbf{SAT}$ diagram:

1. Any spiders or H-boxes with no legs should be removed from the diagram. Evaluate them by concrete calculation and multiply their values together to get a scalar multiplier c for the diagram. If there are no such spiders or H-boxes, set $c = 1$.

2. Extract the phases from all π -phase Z-spiders as follows:

$$\begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \dots \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \stackrel{\pi}{=} \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \quad (18)$$

3. Unfuse the phase of every X-spider with at least two legs:

$$\begin{array}{c} a\pi \\ \vdots \\ \dots \end{array} \stackrel{SF}{=} \begin{array}{c} a\pi \\ \vdots \\ \dots \end{array} \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \quad (19)$$

4. For any X-spiders with at least three legs, unfuse them and apply Lemma 6 to each X-spider:

$$\begin{array}{c} \vdots \\ \dots \end{array} = \begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \dots \begin{array}{c} \vdots \\ \dots \end{array} = \begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \dots \begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \quad (20)$$

5. Replace X-spiders with one leg as follows:

$$\begin{array}{c} \vdots \\ \dots \end{array} = \boxed{0} \begin{array}{c} \vdots \\ \dots \end{array} \quad \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} = \boxed{0} \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \quad (21)$$

6. Excepting the single Z-spider with a π -phase, use the SF_Z rule to fuse Z-spiders wherever possible. If there are two two-legged zero H-boxes connected together directly, introduce a Z-spider between them using the I_Z rule.

At this point, this diagram follows the form of a **#3SAT** diagram except for a possible single π -phase Z-spider. To remove the π -phase Z-spider, we can write the diagram as a sum of two diagrams which don't contain this phase:

$$\begin{array}{c} \pi \\ \vdots \\ \dots \end{array} = \begin{array}{c} \vdots \\ \dots \end{array} - \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} = \boxed{0} \begin{array}{c} \vdots \\ \dots \end{array} - \boxed{0} \begin{array}{c} \pi \\ \vdots \\ \dots \end{array} \quad (22)$$

Since these two diagrams are **#3SAT** diagrams associated with some Boolean functions f_1 and f_2 , the value of D is then given by $D = c(\#(f_1) - \#(f_2))$. \square

Note that this method of splitting an instance into positive and negative components is similar to Bernstein and Vazirani's proof that $\mathbf{BQP} \subseteq \mathbf{P}^{\#\mathbf{P}}$ [8, Theorem 8.2.5]. Now we can move on to considering larger fragments fairly easily. We will show that by using gadgets to copy certain "magic states" - that is, one-legged H-boxes with specific labels which we can split as sums - we can introduce phases which are multiples of $\frac{\pi}{2^k}$ for any fixed k .

Definition 4. $ZH_{\pi/2^k}$ for $k \in \mathbb{N}$ is the fragment of ZH-calculus given by the following generators

$$\begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \frac{n\pi}{2^k} \\ \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \quad \begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \frac{n\pi}{2^k} \\ \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \quad \begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} \vdots \\ \dots \end{array} \quad (23)$$

where $n \in \mathbb{Z}$.

Lemma 7. The following diagram equivalence holds:

$$\begin{array}{c} \vdots \\ \dots \end{array} \begin{array}{c} a \\ \vdots \\ \dots \end{array} \begin{array}{c} a^2 \\ \vdots \\ \dots \end{array} = \begin{array}{c} a \\ \vdots \\ \dots \end{array} \begin{array}{c} a \\ \vdots \\ \dots \end{array} \quad (24)$$

Proof. We have that

$$(25)$$

where the first equality follows from [46, Lemma 3.2]. \square

Lemma 8. For all $k > 0$, there is a reduction from $\text{Eval-ZH}_{\pi/2^k}$ to $\text{Eval-ZH}_{\pi/2^{k-1}}$.

Proof. Let D be a diagram in $\text{ZH}_{\pi/2^k}$. In order to rewrite this into a diagram in $\text{ZH}_{\pi/2^{k-1}}$, we need to remove all Z-spiders with phases that are odd multiples n of $\frac{\pi}{2^k}$, since even multiples of $\frac{\pi}{2^k}$ are already valid for $\text{ZH}_{\pi/2^{k-1}}$. Assume that $n = 2m + 1$, then:

$$(26)$$

Let $a = e^{\frac{i\pi}{2^k}}$, then by applying Lemma 7, fold up all of a -labeled H-boxes into one

$$(27)$$

and note that $a^2 = \left(\frac{\pi}{2^{k-1}}\right)$ is in $\text{ZH}_{\pi/2^{k-1}}$. Remove all scalar H-boxes from D and let their product be c . Finally, we can split the remaining a -labeled H-box, giving D as the sum of two diagrams D_1 and D_2 in $\text{ZH}_{\pi/2^{k-1}}$:

$$(28)$$

so then $D = c(D_1 + aD_2)$. \square

Theorem 5. $\text{Eval-ZH}_{\pi/2^k}$ is $\mathbf{FP}^{\#\mathbf{P}}$ -complete for all $k \geq 0$.

Proof. By induction on Lemma 8 with the base case $k = 0$ given by Theorem 4, we have that $\text{Eval-ZH}_{\pi/2^k}$ is in $\mathbf{FP}^{\#\mathbf{P}}$. It is clearly $\mathbf{FP}^{\#\mathbf{P}}$ -hard, as $\text{ZH}_{\pi/2^k}$ contains the diagrams representing $\#\mathbf{SAT}$ instances, and hence is $\mathbf{FP}^{\#\mathbf{P}}$ -complete. \square

The fragment $\text{ZH}_{\pi/2^2}$ captures precisely those diagrams that represent postselected Clifford+T quantum circuits. Our results above hence also lead to a proof that $\mathbf{PostBQP} \subseteq \mathbf{P}^{\#\mathbf{P}}$, although note that this is weaker than Aaronson's result that $\mathbf{PostBQP} = \mathbf{PP}$ [1].

4 Conclusion

In this paper we have used the ZH-calculus to simplify and unify the proofs of several known results in counting complexity. In particular, we examined various variants of $\#\mathbf{SAT}$ and show that they are $\#\mathbf{P}$ -complete, and similarly that the corresponding $\#\mathbf{M}\mathbf{SAT}$ variants are $\#\mathbf{M}\mathbf{P}$ -complete. We for instance produced a simple direct reduction from $\#\mathbf{SAT}$ to $\#\mathbf{2SAT}$, which considerably simplified existing proofs

that proceed via a reduction to the matrix permanent. Our results show that graphical calculi like the ZH-calculus, even though originally meant for the domain of quantum computing, can provide an intuitive framework for working with counting problems, through their interpretation as tensor networks.

We also briefly examined how other graphical calculi can be used to reason about other counting problems, especially the ZW-calculus and its connection to counting perfect matchings in graphs. A natural future direction is to explore which counting problems can be naturally formulated in a graphical calculus. Finally, we also observed how the original domain of quantum computing can be related to #SAT via the ZH-calculus, and show that the computational problem of evaluating scalar ZH-diagrams that represent postselected Clifford+T or Toffoli+Hadamard quantum circuits is in $\mathbf{FP}^{\#\mathbf{P}}$, and hence can be efficiently evaluated with an #SAT oracle - evaluating whether this leads to a more efficient method for simulating quantum circuits is an interesting avenue for future research.

Acknowledgments

Some of this work was done while TL was a student at the University of Oxford, and the results in Section 3 are also presented in his Master’s thesis [47]. We thank Richie Yeung, Matty Hoban, Julien Coudsi, and the anonymous QPL reviewers for helpful feedback.

References

- [1] Scott Aaronson (2005): *Quantum computing, postselection, and probabilistic polynomial-time*. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 461(2063), pp. 3473–3482, doi:10.1098/rspa.2005.1546.
- [2] Miriam Backens & Aleks Kissinger (2019): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity*. *Electronic Proceedings in Theoretical Computer Science* 287, pp. 23–42, doi:10.4204/EPTCS.287.2. arXiv:1805.02175.
- [3] Miriam Backens, Aleks Kissinger, Hector Miller-Bakewell, John van de Wetering & Sal Wolfs (2021): *Completeness of the ZH-calculus*, doi:10.48550/arXiv.2103.06610. arXiv:2103.06610.
- [4] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and back again: A circuit extraction tale*. *Quantum* 5, p. 421, doi:10.22331/q-2021-03-25-421.
- [5] Niel de Beaudrap, Xiaoning Bian & Quanlong Wang (2020): *Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities*. In Steven T. Flammia, editor: *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 158, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 11:1–11:23, doi:10.4230/LIPICs.TQC.2020.11.
- [6] Niel de Beaudrap & Dominic Horsman (2020): *The ZX calculus is a language for surface code lattice surgery*. *Quantum* 4, doi:10.22331/q-2020-01-09-218.
- [7] A. Ben-Dor & S. Halevi (1993): *Zero-One Permanent Is #P-complete, a Simpler Proof*. In: [1993] *The 2nd Israel Symposium on Theory and Computing Systems*, pp. 108–117, doi:10.1109/ISTCS.1993.253457.
- [8] Ethan Bernstein & Umesh Vazirani (1997): *Quantum Complexity Theory*. *SIAM Journal on Computing* 26(5), pp. 1411–1473, doi:10.1137/S0097539796300921.
- [9] Agustín Borgna, Simon Perdrix & Benoît Valiron (2021): *Hybrid quantum-classical circuit simplification with the ZX-calculus*. In Hakjoo Oh, editor: *Programming Languages and Systems*, Springer International Publishing, Cham, pp. 121–139, doi:10.1007/978-3-030-89051-3_8.
- [10] Tom Bylander (1994): *The computational complexity of propositional STRIPS planning*. *Artificial Intelligence* 69(1), pp. 165–204, doi:10.1016/0004-3702(94)90081-7.

- [11] Shuxiang Cao (2022): *Multi-agent blind quantum computation without universal cluster state*. doi:10.48550/arXiv.2206.13330. arXiv:2206.13330.
- [12] Titouan Carette, Etienne Moutot, Thomas Perez & Renaud Vilmart (2023): *Compositionality of Planar Perfect Matchings*, doi:10.48550/arXiv.2302.08767. arXiv:2302.08767.
- [13] Kostia Chardonnet, Benoît Valiron & Renaud Vilmart (2021): *Geometry of Interaction for ZX-Diagrams*. In Filippo Bonchi & Simon J. Puglisi, editors: *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021), Leibniz International Proceedings in Informatics (LIPIcs) 202*, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 30:1–30:16, doi:10.4230/LIPIcs.MFCS.2021.30.
- [14] Julien Coadi & John van de Wetering (2022): *Classically Simulating Quantum Supremacy IQP Circuits through a Random Graph Approach*. doi:10.48550/arXiv.2212.08609. arXiv:2212.08609.
- [15] Bob Coecke & Ross Duncan (2008): *Interacting quantum observables*. In: *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, Lecture Notes in Computer Science, doi:10.1007/978-3-540-70583-3_25.
- [16] Bob Coecke & Ross Duncan (2011): *Interacting Quantum Observables: Categorical Algebra and Diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016.
- [17] Bob Coecke, Bill Edwards & Rob Spekkens (2011): *Phase groups and the origin of non-locality for qubits*. *Electronic Notes in Theoretical Computer Science* 270(2), pp. 15–36, doi:10.1016/j.entcs.2011.01.021.
- [18] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, Cambridge, doi:10.1017/9781316219317.
- [19] Stephen A. Cook (1971): *The Complexity of Theorem-Proving Procedures*. In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing, STOC '71*, Association for Computing Machinery, New York, NY, USA, pp. 151–158, doi:10.1145/800157.805047.
- [20] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons & Seyon Sivarajah (2020): *Phase Gadget Synthesis for Shallow Circuits*. In Bob Coecke & Matthew Leifer, editors: *Proceedings 16th International Conference on Quantum Physics and Logic, Chapman University, Orange, CA, USA., 10-14 June 2019, Electronic Proceedings in Theoretical Computer Science* 318, Open Publishing Association, pp. 213–228, doi:10.4204/EPTCS.318.13.
- [21] Alexander Cowtan, Will Simmons & Ross Duncan (2020): *A Generic Compilation Strategy for the Unitary Coupled Cluster Ansatz*. doi:10.48550/arXiv.2007.10515. arXiv:2007.10515.
- [22] Vilhelm Dahllöf, Peter Jonsson & Magnus Wahlström (2002): *Counting Satisfying Assignments in 2-SAT and 3-SAT*. In Oscar H. Ibarra & Louxin Zhang, editors: *Computing and Combinatorics*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 535–543, doi:10.1007/3-540-45655-4_57.
- [23] Carsten Damm, Markus Holzer & Pierre McKenzie (2002): *The Complexity of Tensor Calculus*. *Computational Complexity* 11(1/2), pp. 54–89, doi:10.1007/s00037-000-0170-4.
- [24] Niel de Beaudrap, Aleks Kissinger & Konstantinos Meichanetzidis (2021): *Tensor Network Rewriting Strategies for Satisfiability and Counting*. *Electronic Proceedings in Theoretical Computer Science* 340, pp. 46–59, doi:10.4204/EPTCS.340.3. arXiv:2004.06455.
- [25] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.
- [26] Ross Duncan & Maxime Lucas (2014): *Verifying the Steane code with Quantomatic*. In Bob Coecke & Matty Hoban, editors: *Proceedings of the 10th International Workshop on Quantum Physics and Logic, Castelldefels (Barcelona), Spain, 17th to 19th July 2013, Electronic Proceedings in Theoretical Computer Science* 171, Open Publishing Association, pp. 33–49, doi:10.4204/EPTCS.171.4.
- [27] Ross Duncan & Simon Perdrix (2010): *Rewriting measurement-based quantum computations with generalised flow*. In: *International Colloquium on Automata, Languages, and Programming*, Springer, pp. 285–296, doi:10.1007/978-3-642-14162-1_24.

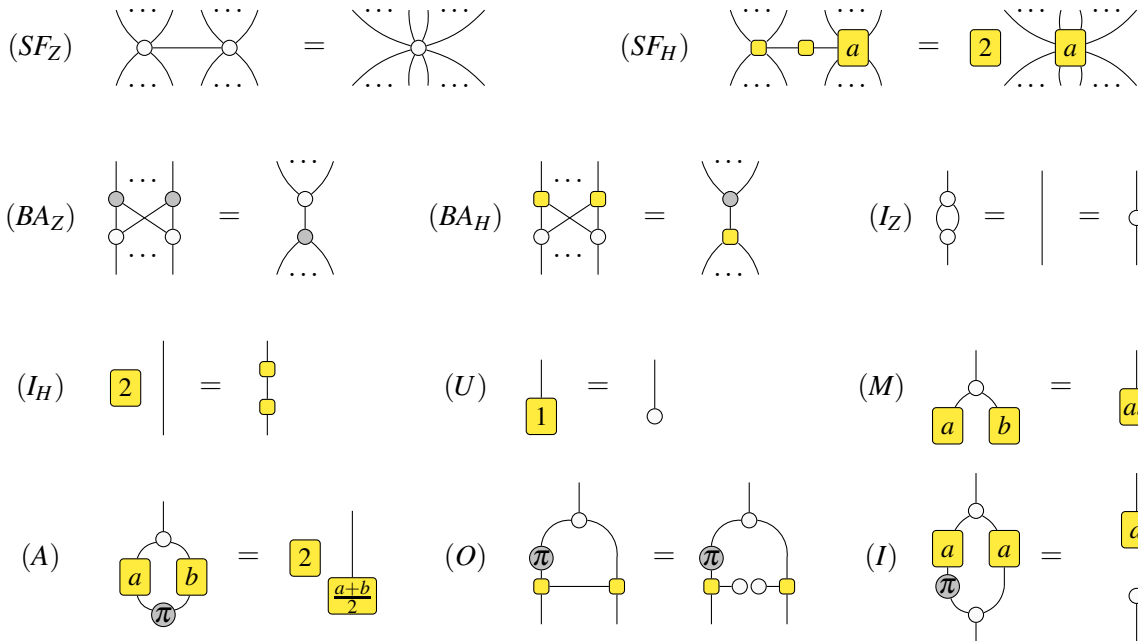
- [28] John Faben (2008): *The Complexity of Counting Solutions to Generalised Satisfiability Problems modulo k* , doi:10.48550/arXiv.0809.1836. arXiv:0809.1836.
- [29] Artur García-Sáez & José I. Latorre (2012): *An Exact Tensor Network for the 3SAT Problem*. *Quantum Info. Comput.* 12(3–4), p. 283–292, doi:10.5555/2230976.2230984.
- [30] Craig Gidney (2022): *A Pair Measurement Surface Code on Pentagons*. doi:10.48550/arXiv.2206.12780. arXiv:2206.12780.
- [31] Craig Gidney & Austin G. Fowler (2019): *Efficient magic state factories with a catalyzed $|CCZ\rangle$ to $2|T\rangle$ transformation*. *Quantum* 3, p. 135, doi:10.22331/q-2019-04-30-135.
- [32] Craig Gidney & Austin G. Fowler (2019): *Flexible layout of surface code computations using AutoCCZ states*. doi:10.48550/arXiv.1905.08916. arXiv:1905.08916.
- [33] Stefano Gogioso & Aleks Kissinger (2017): *Fully graphical treatment of the quantum algorithm for the Hidden Subgroup Problem*, doi:10.48550/arXiv.1701.08669. arXiv:1701.08669.
- [34] Johnnie Gray & Stefanos Kourtis (2021): *Hyper-Optimized Tensor Network Contraction*. *Quantum* 5, p. 410, doi:10.22331/q-2021-03-15-410.
- [35] Amar Hadzihasanovic (2015): *A Diagrammatic Axiomatisation for Qubit Entanglement*, doi:10.48550/arXiv.1501.07082. arXiv:1501.07082.
- [36] John H. Halton (1966): *On the Divisibility Properties of Fibonacci Numbers*. *The Fibonacci Quarterly* 4(3), pp. 217–240. Available at <https://www.fq.math.ca/Scanned/4-3/halton.pdf>.
- [37] Michael Hanks, Marta P. Estarellas, William J. Munro & Kae Nemoto (2020): *Effective Compression of Quantum Braided Circuits Aided by ZX-Calculus*. *Physical Review X* 10, p. 041030, doi:10.1103/PhysRevX.10.041030.
- [38] Anne Hillebrand (2012): *Superdense Coding with GHZ and Quantum Key Distribution with W in the ZX-calculus*. *Electronic Proceedings in Theoretical Computer Science* 95, pp. 103–121, doi:10.4204/EPTCS.95.10.
- [39] Clare Horsman (2011): *Quantum pictorialism for topological cluster-state computing*. *New Journal of Physics* 13(9), p. 095011, doi:10.1088/1367-2630/13/9/095011.
- [40] Mark Jerrum, Alistair Sinclair & Eric Vigoda (2004): *A Polynomial-Time Approximation Algorithm for the Permanent of a Matrix with Nonnegative Entries*. *Journal of the ACM* 51(4), pp. 671–697, doi:10.1145/1008731.1008738.
- [41] Aleks Kissinger & John van de Wetering (2019): *Universal MBQC with generalised parity-phase interactions and Pauli measurements*. *Quantum* 3, doi:10.22331/q-2019-04-26-134.
- [42] Aleks Kissinger & John van de Wetering (2020): *Reducing the number of non-Clifford gates in quantum circuits*. *Physical Review A* 102, p. 022406, doi:10.1103/PhysRevA.102.022406.
- [43] Aleks Kissinger & John van de Wetering (2022): *Simulating quantum circuits with ZX-calculus reduced stabiliser decompositions*. *Quantum Science and Technology* 7(4), p. 044001, doi:10.1088/2058-9565/ac5d20.
- [44] Aleks Kissinger, John van de Wetering & Renaud Vilmart (2022): *Classical Simulation of Quantum Circuits with Partial and Graphical Stabiliser Decompositions*. In François Le Gall & Tomoyuki Morimae, editors: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 232, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 5:1–5:13, doi:10.4230/LIPIcs.TQC.2022.5.
- [45] Stefanos Kourtis, Claudio Chamon, Eduardo Mucciolo & Andrei Ruckenstein (2019): *Fast Counting with Tensor Networks*. *SciPost Physics* 7(5), p. 060, doi:10.21468/SciPostPhys.7.5.060.
- [46] Stach Kuijpers, John van de Wetering & Aleks Kissinger (2019): *Graphical Fourier Theory and the Cost of Quantum Addition*, doi:10.48550/arXiv.1904.07551. arXiv:1904.07551.

- [47] Tuomas Laakkonen (2022): *Graphical Stabilizer Decompositions For Counting Problems*. Master's thesis, University of Oxford. Available at <https://www.cs.ox.ac.uk/people/aleks.kissinger/theses/laakkonen-thesis.pdf>.
- [48] Tuomas Laakkonen, Konstantinos Meichanetzidis & John van de Wetering (2022): *A Graphical #SAT Algorithm for Formulae with Small Clause Density*, doi:10.48550/arXiv.2212.08048. arXiv:2212.08048.
- [49] Adrian Lehmann, Ben Caldwell & Robert Rand (2022): *VyZX : A Vision for Verifying the ZX Calculus*. doi:10.48550/arXiv.2205.05781. arXiv:2205.05781.
- [50] David Lichtenstein (1982): *Planar Formulae and Their Uses*. *SIAM Journal on Computing* 11(2), pp. 329–343, doi:10.1137/0211025.
- [51] Daniel Litinski & Naomi Nickerson (2022): *Active volume: An architecture for efficient fault-tolerant quantum computers with limited non-local connections*. doi:10.48550/arXiv.2211.15465. arXiv:2211.15465.
- [52] Cristopher Moore & Stephan Mertens (2011): *The Nature of Computation*. Oxford University Press, Inc., USA, doi:10.1093/acprof:oso/9780199233212.001.0001.
- [53] Michael A. Nielsen & Isaac L. Chuang (2010): *Quantum Computation and Quantum Information*. Cambridge University Press, doi:10.1017/CB09780511976667.
- [54] Román Orús (2014): *A practical introduction to tensor networks: Matrix product states and projected entangled pair states*. *Annals of Physics* 349, pp. 117–158, doi:10.1016/j.aop.2014.06.013.
- [55] C.H. Papadimitriou (1994): *Computational Complexity*. Theoretical computer science, Addison-Wesley, doi:10.5555/1074100.1074233.
- [56] Dan Roth (1996): *On the hardness of approximate reasoning*. *Artificial Intelligence* 82(1), pp. 273–302, doi:10.1016/0004-3702(94)00092-1.
- [57] Razin A. Shaikh, Quanlong Wang & Richie Yeung (2022): *How to sum and exponentiate Hamiltonians in ZXW calculus*, doi:10.48550/arXiv.2212.04462. arXiv:2212.04462.
- [58] Alexis Shaw, Michael Bremner, Alexandru Paler, Daniel Herr & Simon J. Devitt (2022): *Quantum computation on a 19-qubit wide 2d nearest neighbour qubit array*. doi:10.48550/arXiv.2212.01550. arXiv:2212.01550.
- [59] Alex Townsend-Teague & Konstantinos Meichanetzidis (2021): *Classifying Complexity with the ZX-Calculus: Jones Polynomials and Potts Partition Functions*. doi:10.48550/arXiv.2103.06914. arXiv:2103.06914.
- [60] G. S. Tseitin (1983): *On the Complexity of Derivation in Propositional Calculus*. In Jörg H. Siekmann & Graham Wrightson, editors: *Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970*, Symbolic Computation, Springer, Berlin, Heidelberg, pp. 466–483, doi:10.1007/978-3-642-81955-1_28.
- [61] Christian Ufrecht, Maniraman Periyasamy, Sebastian Rietsch, Daniel D. Scherer, Axel Plinge & Christopher Mutschler (2023): *Cutting multi-control quantum gates with ZX calculus*. doi:10.48550/arXiv.2302.00387. arXiv:2302.00387.
- [62] L. G. Valiant (1979): *The Complexity of Computing the Permanent*. *Theoretical Computer Science* 8(2), pp. 189–201, doi:10.1016/0304-3975(79)90044-6.
- [63] L. G. Valiant & V. V. Vazirani (1986): *NP Is as Easy as Detecting Unique Solutions*. *Theoretical Computer Science* 47, pp. 85–93, doi:10.1016/0304-3975(86)90135-0.
- [64] Leslie G. Valiant (1979): *The Complexity of Enumeration and Reliability Problems*. *SIAM Journal on Computing* 8(3), pp. 410–421, doi:10.1137/0208032.
- [65] Leslie G. Valiant (2006): *Accidental Algorithms*. In: *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pp. 509–517, doi:10.1109/FOCS.2006.7.
- [66] Jamie Vicary (2013): *Topological Structure of Quantum Algorithms*. *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, doi:10.1109/lics.2013.14.

- [67] Magnus Wahlström (2008): *A Tighter Bound for Counting Max-Weight Solutions to 2SAT Instances*. In Martin Grohe & Rolf Niedermeier, editors: *Parameterized and Exact Computation*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 202–213, doi:10.1007/978-3-540-79723-4_19.
- [68] Dominic Welsh & Amy Gale (2011): *The Complexity of Counting Problems*. In: *Aspects of Complexity: Minicourses in Algorithmics, Complexity and Computational Algebra. Mathematics Workshop, Kaikoura, January 7-15, 2000*, De Gruyter, doi:10.1515/9783110889178.
- [69] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*. doi:10.48550/arXiv.2012.13966. arXiv:2012.13966.
- [70] Mingji Xia & Wenbo Zhao (2006): *#3-Regular Bipartite Planar Vertex Cover Is #P-Complete*. In Jin-Yi Cai, S. Barry Cooper & Angsheng Li, editors: *Theory and Applications of Models of Computation*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 356–364, doi:10.1007/11750321_34.

A Rewriting Rules

The ZH-calculus is equipped with the following set of sound and complete rewriting rules [2]:



We will also use the following derived rewriting rules [3, Lemmas 2.10-2.24],

$$(\pi_1) \quad \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \\ \diagdown \quad \diagup \\ \dots \end{array} = \begin{array}{c} \dots \\ \pi \quad \pi \\ \diagup \quad \diagdown \\ \circ \\ \diagdown \quad \diagup \\ \pi \quad \pi \\ \dots \end{array} \quad (\pi_2) \quad \begin{array}{c} \dots \\ \pi \\ \diagdown \quad \diagup \\ \circ \\ \dots \end{array} = \begin{array}{c} \dots \\ \pi \cdots \pi \\ \dots \end{array} \quad (29)$$

$$(SF_X) \quad \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \quad \circ \\ \diagdown \quad \diagup \\ \dots \end{array} = \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \\ \diagdown \quad \diagup \\ \dots \end{array}$$

as well as the generalized (M) rule [2, Lemma 2.3]:

$$(M) \quad \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \quad \circ \\ \diagdown \quad \diagup \\ a \quad b \end{array} = \begin{array}{c} \dots \\ \diagdown \quad \diagup \\ \circ \\ \dots \end{array} \quad (30)$$

B #SAT \rightarrow #MON-BI-PL-3DEG-SAT

The smallest subset of #2SAT that has been considered in the literature is #PL-MON-BI-CUBIC-2SAT, where CUBIC- indicates that the primal graph of the instance is 3-regular [70]. In this section, we show that #3DEG-SAT is #P-complete graphically, by relating the zero-labeled H-box with the Fibonacci numbers. Here, 3DEG- indicates that every variable appears in at most three clauses. Then in Theorem 7 we combine all of our reductions to show #P-completeness for #PL-MON-BI-3DEG-2SAT. This is slightly less restrictive than #PL-MON-BI-CUBIC-2SAT, but retains the interesting property that the maximum degree is the lowest possible (if the maximum degree was two, then the problem can be solved in polynomial time [22]), while avoiding the complicated global construction of the original proof.

We will make use of the following identities concerning the Fibonacci numbers, defined by:

$$F_n = F_{n-1} + F_{n-2} \quad F_1 = 1 \quad F_0 = 0 \quad (31)$$

Lemma 9. $\gcd(F_n, F_{n-1}) = 1$ for all $n > 1$.

Proof. $\gcd(F_n, F_{n-1}) = \gcd(F_{n-1} + F_{n-2}, F_{n-1}) = \gcd(F_{n-2}, F_{n-1})$ thus by induction $\gcd(F_n, F_{n-1}) = \gcd(F_2, F_1) = 1$. \square

Lemma 10 [36]. For every $M \geq 1$ there exists some $0 < n \leq M^2$ such that $F_n \equiv 0 \pmod{M}$.

Lemma 11 [36]. $F_n = \left\lfloor \frac{\phi^n}{\sqrt{5}} + \frac{1}{2} \right\rfloor$ where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio.

Lemma 12. Suppose that $F_n \equiv 0 \pmod{M}$, then the following rewrite holds:

$$\text{---} \equiv \boxed{F_{n-1}^{-1}} \underbrace{\text{---} \boxed{0} \text{---} \boxed{0} \text{---} \cdots \text{---} \boxed{0} \text{---}}_{n \text{ copies}} \text{---} \pmod{M} \quad (32)$$

Proof. Note that the Fibonacci numbers are defined by

$$\begin{pmatrix} F_{n+1} \\ F_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} F_n \\ F_{n-1} \end{pmatrix} \implies \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} \quad (33)$$

and so, supposing that $F_n \equiv 0 \pmod{M}$, we have

$$\begin{aligned} \underbrace{\text{---} \boxed{0} \text{---} \boxed{0} \text{---} \cdots \text{---} \boxed{0} \text{---}}_{n \text{ copies}} &= \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n = \begin{pmatrix} F_n + F_{n-1} & F_n \\ F_n & F_{n-1} \end{pmatrix} \\ &\equiv \begin{pmatrix} F_{n-1} & 0 \\ 0 & F_{n-1} \end{pmatrix} = \boxed{F_{n-1}^{-1}} \text{---} \pmod{M} \end{aligned} \quad (34)$$

but by Lemma 9, F_{n-1} is coprime to F_n , so it must be coprime to M and thus invertible. \square

Lemma 13. Given M and k such that $F_k \equiv 0 \pmod{M}$, for any $\phi \in \#_{\mathbf{M}}\text{SAT}$ with n variables, m clauses, and maximum clause size at least two, there is a $\phi' \in \#_{\mathbf{M}}\text{SAT}$ with $O(nmk)$ variables and $O(nmk)$ clauses such that every variable has degree at most three, and $\#_M(\phi) = c \cdot \#_M(\phi') \pmod{M}$ where c is computable in $O(\text{poly}(n, m, k))$ time. Additionally, ϕ' can be computed in $O(\text{poly}(n, m, k))$ time and preserves the maximum clause size of ϕ .

where the last line follows from the proof of Lemma 3. After this is applied to every clause of size two, every path between two vertices in the incidence graph will have even length (since every edge is replaced by four edges), and hence the incidence graph is bipartite. \square

Theorem 7. We have the following:

1. **#MON-BI-PL-3DEG-2SAT** is **#P**-complete
2. \oplus **MON-BI-PL-3DEG-2SAT** is \oplus **P**-complete

Proof. We can reduce from arbitrary **#SAT** instances to **#MON-BI-PL-3DEG-2SAT** by applying the previously given reductions in the following order:

$$\begin{aligned} \#SAT &\xrightarrow{1} \#PL\text{-SAT} \xrightarrow{2} \#PL\text{-2SAT} \xrightarrow{5} \#MON\text{-PL-2SAT} \\ &\xrightarrow{14} \#MON\text{-BI-PL-2SAT} \xrightarrow{13} \#MON\text{-BI-PL-3DEG-2SAT} \end{aligned}$$

In both cases, we first reduce to **#PL-SAT** using Lemma 1. Then, we continue differently:

1. Apply Theorem 2 and Theorem 3 to reduce to **#MON-PL-2SAT**, as these preserve planarity. Then, given ϕ with n variables, we apply Lemma 14 with $r = n + 1$ to obtain $\phi' \in \#MON\text{-BI-PL-2SAT}$ such that $\#(\phi) = \#_{2^r}(\phi') = \#(\phi') \pmod{2^r}$. Then reduce ϕ' to **#MON-BI-PL-3DEG-2SAT** using Theorem 6, since it preserves planarity, monotonicity and bipartite structure.
2. Similarly, given $\phi \in \#SAT$, apply Lemma 3 with $r = 0$, Lemma 5 with $r = 1$, Lemma 14 with $r = 0$, and Lemma 13 with $M = 2$ and $k = 3$, to obtain $\phi' \in \#MON\text{-BI-PL-3DEG-2SAT}$ such that $\#_2(\phi) = \#_2(\phi')$. \square

C #PERFECT-MATCHINGS and the ZW-Calculus

While so far we have worked exclusively with the ZH-calculus, which naturally represents **#SAT**, we can use other calculi to attack other problems. In this section, we will use the ZW-calculus to examine the connection between the problems **#XSAT** and **#PERFECT-MATCHINGS**, and sketch an argument that they are both **#P**-complete. Like the connection between **#SAT** and **#2SAT** given using the ZH-calculus, with this technique we can circumvent the usual reduction via the permanent.

Definition 5. Let $f : \mathbb{B}^n \rightarrow \mathbb{B}$ be a boolean function defined by

$$f(x) = \bigwedge_{i=0}^m \phi(c_{i1}, \dots, c_{ik_i}) \quad (38)$$

where $c_{ij} = x_k$ or $\neg x_k$ for some k , and $\phi(\vec{x}) = 1$ if and only if $w(\vec{x}) = 1$, where $w(\vec{x})$ is the Hamming weight of \vec{x} . Each ϕ term defines a clause, and so $f(x) = 1$ iff every clause has exactly one true literal. The problem **#XSAT** is to compute $\#(f)$. When $k_i = 3$ for all i , this is also known as **#1-in-3SAT**.

Definition 6. The problem **#PERFECT-MATCHINGS** is as follows: given an undirected simple graph G , compute the number of perfect matchings of G . That is, the number of independent edge sets of G that cover each vertex exactly once. We denote this quantity $\text{PerfMatch}(G)$.

The ZW-calculus is a graphical calculus built from two generators, W-spiders and Z-spiders, which are flexsymmetric [35]. The Z-spider is a close analogue of the Z-spider in ZH-calculus (except with a π phase), whereas the W-spider represents the W-state:

$$\begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \\ \diagdown \quad \diagup \\ \dots \end{array} = |\vec{0}\rangle\langle\vec{0}| - |\vec{1}\rangle\langle\vec{1}| \qquad \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \bullet \\ \diagdown \quad \diagup \\ \dots \end{array} = \sum_{\substack{u,v \\ w(uv)=1}} |u\rangle\langle v| \tag{39}$$

These, along with wires, caps, and cups, are combined with tensor product and tensor contraction in the same way as in the ZH-calculus. Like the ZH-calculus, we will treat diagrams purely as tensor networks rather than formal objects - hence equality of diagrams is just equality of tensors. This calculus is also equipped with a set of sound and complete rewrite rules, including the following spider fusion rules,

$$\begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \circ \\ \diagdown \quad \diagup \\ \dots \end{array} = \begin{array}{c} \dots \\ \diagup \quad \diagdown \quad \diagup \quad \diagdown \\ \circ \quad \circ \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ \dots \end{array} \qquad \begin{array}{c} \dots \\ \diagup \quad \diagdown \\ \bullet \\ \diagdown \quad \diagup \\ \dots \end{array} = \begin{array}{c} \dots \\ \diagup \quad \diagdown \quad \diagup \quad \diagdown \\ \bullet \quad \bullet \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ \dots \end{array} \tag{40}$$

as well as others which we omit for brevity as we don't use them explicitly here. In the same way that ZH-calculus diagrams naturally represent #SAT instances with Z-spiders and clauses, the ZW-calculus naturally represents #XSAT instances with the following mapping:

$$\text{Variable} \iff \begin{array}{c} \circ \\ \diagup \quad \diagdown \\ \dots \end{array} \qquad \text{Clause} \iff \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \dots \end{array} \qquad \text{Negation} \iff \text{---}\bullet\text{---} \tag{41}$$

Lemma 15. #XSAT is #P-complete

Proof. We can translate from #2SAT to #XSAT with the following correspondence:

$$\begin{array}{c} \circ \\ \diagup \quad \diagdown \\ \dots \end{array} = \begin{array}{c} \circ \\ \diagup \quad \diagdown \\ \dots \end{array} \qquad \text{---}\boxed{0}\text{---} = \begin{array}{c} \circ \\ \bullet \\ \dots \end{array} \qquad \text{---}\pi\text{---} = \text{---}\bullet\text{---} \tag{42}$$

In the other direction, we can translate #XSAT to #SAT by first expanding every clause of size more than three as follows:

$$\begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \dots \end{array} = \begin{array}{c} \bullet \quad \bullet \quad \dots \quad \bullet \\ \diagup \quad \diagdown \quad \dots \quad \diagup \quad \diagdown \\ \dots \end{array} = \begin{array}{c} \bullet \quad \circ \quad \dots \quad \circ \\ \diagup \quad \diagdown \quad \dots \quad \diagup \quad \diagdown \\ \dots \end{array} \tag{43}$$

Then, each clause

$$\begin{aligned} \phi(x,y,z) &= (x \wedge \neg y \wedge \neg z) \vee (\neg x \wedge y \wedge \neg z) \vee (\neg x \wedge \neg y \wedge z) \\ \phi(x,y) &= (x \wedge \neg y) \vee (\neg x \wedge y) \\ \phi(x) &= x \end{aligned} \tag{44}$$

can be rewritten as a bounded number of CNF clauses. □

However, as Carette *et al.* [12] note, diagrams of the ZW-calculus can also naturally represent instances of #PERFECT-MATCHINGS by taking each vertex of the graph to be a W-spider and edges of the graph as wires. Therefore, any ZW-diagram containing only W-spiders represents an instance of #PERFECT-MATCHINGS.

Theorem 8. #PERFECT-MATCHINGS is #P-complete.

Proof. Suppose we are given an instance f of **#XSAT** on n variables as a ZW-diagram, then to transform it to an instance of **#PERFECT-MATCHINGS** we need to remove all Z-spiders, which represent variables. First split all the variables so they have degree at most three:

$$\begin{array}{c} \circ \\ \diagup \quad \diagdown \\ \dots \end{array} = \begin{array}{c} \circ \quad \circ \quad \dots \quad \circ \\ | \quad | \quad \dots \quad | \\ \dots \quad \dots \quad \dots \quad \dots \end{array} \quad (45)$$

Then we can use the following rewrites to remove all variables with degree two and three:

$$\begin{array}{c} \circ \\ \diagup \quad \diagdown \\ \dots \end{array} = \frac{1}{2} \begin{array}{c} \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \end{array} \quad \begin{array}{c} \circ \\ | \\ \dots \end{array} = \begin{array}{c} \dots \end{array} \quad (46)$$

We are left with only variables of degree one, some extraneous Z-spiders with degree two, and a constant factor of 2^{-c} . To complete the reduction to **#PERFECT-MATCHINGS** it then remains to show we can get rid of these degree-one variables and the degree-two Z-spiders.

We can remove the Z-spiders by considering the whole diagram modulo $2^{n+c} + 1$: since we started with a **#XSAT** instance with n variables, we have $0 \leq \#(f) \leq 2^n$, so the value of the remaining diagram is at most 2^{n+c} . It is hence sufficient to calculate modulo $2^{n+c} + 1$ for our resulting diagram. In this setting, we have

$$\begin{array}{c} \circ \\ | \\ \dots \end{array} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 \\ 0 & 2^{n+c} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}^{n+c} = \underbrace{\begin{array}{c} \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \end{array}}_{n+c \text{ copies}} \pmod{2^{n+c} + 1} \quad (47)$$

since

$$\begin{array}{c} \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \\ \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \end{array} = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \quad (48)$$

and hence we are left with a diagram containing only variables of degree one, and no other Z-spiders. To remove these variables of degree one, note that

$$\begin{array}{c} \circ \quad \circ \quad \bullet \\ | \quad | \quad | \\ \dots \quad \dots \quad \dots \end{array} = \begin{array}{c} \circ \quad \circ \\ | \quad | \\ \dots \quad \dots \end{array} \quad (49)$$

and thus we can combine all the variables of degree one together:

$$\begin{array}{c} \circ \quad \dots \quad \circ \\ | \quad \dots \quad | \\ \dots \quad \dots \quad \dots \end{array} = \begin{array}{c} \circ \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \dots \quad \bullet \quad \bullet \quad \bullet \\ | \quad | \quad | \quad | \quad | \quad | \quad \dots \quad | \quad | \quad | \\ \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \end{array} \quad (50)$$

Finally, we can remove the last variable by splitting the diagram as a sum of diagrams, neither of which contain any Z-spiders:

$$\begin{array}{c} \circ \quad \circ \\ | \quad | \\ \dots \quad \dots \end{array} = \begin{array}{c} \bullet \quad \bullet \\ | \quad | \\ \dots \quad \dots \end{array} + \begin{array}{c} \bullet \\ | \\ \dots \end{array} \quad (51)$$

Thus, these two diagrams each represent an instance of **#PERFECT-MATCHINGS** - let us denote the graphs of the corresponding instances as G_1 and G_2 . The construction above allows us to obtain G_1 and G_2 in polynomial time, and we have

$$\#(f) = 2^{-c}(\text{PerfMatch}(G_1) + \text{PerfMatch}(G_2)) \pmod{2^{n+c} + 1} \quad (52)$$

hence **#PERFECT-MATCHINGS** is **#P**-hard. We can also see that **#PERFECT-MATCHINGS** \in **#P**, since we can use Equation (46) to rewrite all wires into variables of degree two, and thus transform an instance of **#PERFECT-MATCHINGS** into an instance of **#XSAT**. Therefore, **#PERFECT-MATCHINGS** is **#P**-complete. \square

D #P-Completeness for the Permanent

The proof by Valiant [62] that the permanent of an integer-valued matrix, \mathbb{Z} -Permanent, is #P-complete, and the simplified proof by Ben-Dor and Halevi [7], both rely on a reduction from #3SAT. This reduction could be simplified by using #2SAT instead, but this was unfortunately not possible, as the proof that #2SAT is #P-complete relies itself on a reduction from the permanent. However, since we proved in Theorem 2 that #2SAT is #P-complete independent of the permanent, we can make use of this to simplify the reduction for \mathbb{Z} -Permanent further. In this section, we detail this reduction, which shows that \mathbb{Z} -Permanent is #P-hard. Our construction and proof is essentially identical to that of Ben-Dor and Halevi [7], with the exception that we start with an instance of #MON-2SAT, and can hence use simpler gadgets.

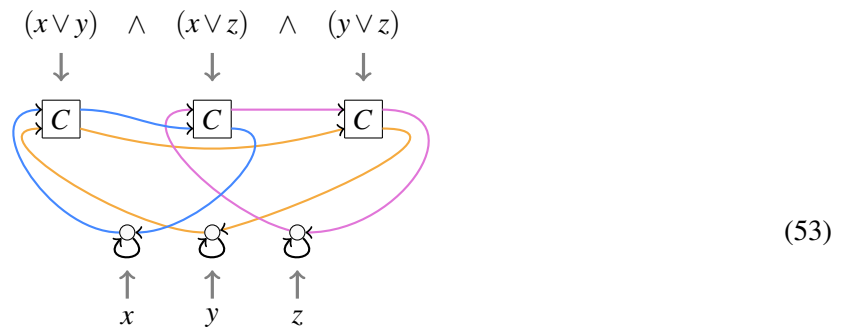
Definition 7. Given a directed edge-weighted graph G with edges E , a *cycle-cover* of G is a set $E' \subseteq E$ of simple cycles that partition the vertices of G . Note that self-loops are permitted in G . The weight of a cycle cover is the product of all the weights of the edges in E' .

Lemma 16. Let G be a directed graph with self-loops and edge weights w_{ij} , and let A be its adjacency matrix, i.e. $A_{ij} = w_{ij}$ if i and j are connected or $A_{ij} = 0$ otherwise. Then the permanent of A is the sum of weights of all cycle-covers of G . We denote this number by $\#(G)$.

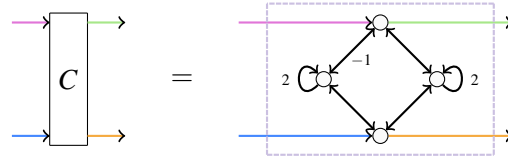
Given this, it is sufficient to reduce #MON-2SAT to the problem of determining the sum of weights of cycle-covers of a graph. We aim to construct a graph G_ϕ from a #MON-2SAT instance ϕ with n variables and m clauses, in polynomial time, such that $\#(G_\phi) = f(\phi) \cdot \#(\phi)$ for some easily computable and suitably bounded $f(\phi)$. We construct G_ϕ as follows:

1. For each variable x_i in ϕ , introduce a vertex v_i to G_ϕ .
2. For each clause in ϕ , introduce a *clause gadget* of four vertices to G_ϕ , the structure of which we will describe momentarily. Two of the vertices of this gadget are designated as the first and second input respectively.
3. For every variable vertex, add a self-loop s_i of weight one to G_ϕ .
4. For every variable vertex v_i , add an edge of weight one from v_i to an unused input c_1 of the first clause in which v appears. Then add an edge of weight one from c_1 to c_2 , an unused input of the second clause in which v appears. Continue similarly until all d_i clauses in which v appears have been processed, then add an edge of weight one from c_{d_i} to v_i . Let these edges be labeled as c_{ij} .

An example of this construction is given below, the loop of edges proceeding from each variable highlighted in a different color:



The clause gadget is given by the following graph



where the top-most vertex is the first input, and the bottom-most vertex is the second. Bidirectional edges represent a pair of edges, one in each direction, with the same weights. Now let E_c be the set of edges in G_ϕ that are internal to clause gadgets, and let $E_r = E \setminus E_c$ be the rest.

Definition 8. Let a *partial cover* of G_ϕ be a subset $E'_r \subseteq E_r$. A *completion* of E'_r is a cycle cover of G_ϕ given by $E'_r \cup E'_c$ where $E'_c \subseteq E_c$. We call the *weight* of E'_r the sum of the weights of all completions of E'_r .

Let us say that a partial cover E'_r is induced by a satisfying assignment \vec{x} of ϕ if, for every variable x_i assigned false in \vec{x} , $s_i \in E'_r$ and $c_{ij} \notin E'_r$ for all $1 \leq j \leq d_i$, and for every variable x_i assigned true in \vec{x} , $c_{ij} \in E'_r$ for all $1 \leq j \leq d_i$ and $s_i \notin E'_r$. We wish to argue that the weight of a partial cover of G_ϕ is non-zero if and only if it is induced by a satisfying assignment.

Lemma 17. Let E'_r be a partial cover of G_ϕ , then the weight of E'_r is 4^m if E'_r is induced by a satisfying assignment, and zero otherwise. Moreover, each such E'_r is induced by a unique satisfying assignment.

Proof. Suppose E'_r is induced by a satisfying assignment. Then for each clause gadget, the ingoing and outgoing edges are included in E'_r for either one or both inputs (otherwise there is an unsatisfied clause). The possible completions of E'_r are as follows for each clause gadget:

$$\begin{array}{c}
 \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} \rightarrow \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} + \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} = 4 \\
 \text{(54)} \\
 \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} \rightarrow \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} = 4
 \end{array}$$

The dotted edges represent edges not included in the cycle-cover. Then the total weight of each clause gadget over the completions is four in either case, so the overall weight of E'_r is 4^m . Now suppose that E'_r is not induced by a satisfying assignment. Note that if the number of incoming and outgoing edges of each clause gadget in E'_r is not equal, then the weight of E'_r is zero, as there is no valid completion of E'_r (because there can be no such cycle-cover). Therefore, the only remaining case is that there is at least one clause gadget which has no incoming and outgoing edges, or has one incoming edge and one outgoing edge on the opposing input (otherwise E'_r would be induced by a satisfying assignment). In either case, we can see the total weight of the gadget over the completions is zero:

$$\begin{array}{c}
 \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} \rightarrow \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} + \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} = 0 \\
 \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} \rightarrow \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} + \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} + \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} + \begin{array}{c} \xrightarrow{\quad} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \circ \xrightarrow{-1} \circ \xrightarrow{\quad} \\ \xrightarrow{\quad} \circ \xrightarrow{\quad} \end{array} = 0 \\
 \text{(55)}
 \end{array}$$

Hence, the weight of E'_r must also be zero. Note that each E'_r that is induced by a satisfying assignment must be induced by a unique assignment, since you can recover the assignment from E'_r . \square

Clearly, the sum of weights of all partial covers of G_ϕ is the same as the sum of weights of all cycle-covers of G_ϕ . But by Lemma 17, this is $4^m \#(\phi)$, so $\#(\phi) = 4^{-m} \#(G_\phi)$, and thus \mathbb{Z} -**Permanent** is $\#\mathbf{P}$ -hard, as G_ϕ can be computed in polynomial time from ϕ .

It is interesting to note that the constructions of Ben-Dor and Halevi, and Valiant, both make use of negative-weight edges and have $f(\phi) = k^m$ for some even integer k (12 for Ben-Dor and Halevi, and 4^5 for Valiant). In order to further simplify the next steps of the reduction to \mathbb{B} -**Permanent**, it would be desirable to have no negative weights, or $k = 1$. However, as Valiant points out [62], neither of these is likely to be possible:

- If k is odd, then $\#(\phi) \equiv \#(G_\phi) \pmod{2}$, but $\#(G_\phi) \pmod{2}$ is easy to compute [7] (as the parity of the permanent is equal to the parity of the determinant), so then $\mathbf{P} = \bigoplus \mathbf{P}$ and $\mathbf{NP} = \mathbf{RP}$ by the Valiant-Vazirani theorem [63].
- Suppose G_ϕ is constructed by reduction from **3SAT**. If there are no negative-weighted edges, then the existence of any cycle-cover of G_ϕ indicates the existence of a satisfying assignment to ϕ . But determining if a cycle-cover exists is easy for general directed graphs, so then $\mathbf{P} = \mathbf{NP}$.

This last argument does not hold up for our construction, since we start from **#MON-2SAT**, for which it is trivial to determine if a satisfying assignment exists (indeed, one always exists by setting every variable true). However, we can still rule out the possibility of a reduction without negative-weighted edges: it is known that \mathbb{Z} -**Permanent** with non-negative weights has an FPRAS [40], whereas **#MON-2SAT** does not, unless $\mathbf{NP} = \mathbf{RP}$ [68, Theorem 57].

Generators and Relations for 3-Qubit Clifford+CS Operators

Xiaoning Bian and Peter Selinger

Dalhousie University

We give a presentation by generators and relations of the group of 3-qubit Clifford+CS operators. The proof roughly consists of two parts: (1) applying the Reidemeister-Schreier theorem recursively to an earlier result of ours; and (2) the simplification of thousands of relations into 17 relations. Both (1) and (2) have been formally verified in the proof assistant Agda. The Reidemeister-Schreier theorem gives a constructive method for computing a presentation of a sub-monoid given a presentation of the super-monoid. To achieve (2), we devise an almost-normal form for Clifford+CS operators. Along the way, we also identify several interesting structures within the Clifford+CS group. Specifically, we identify three different finite subgroups for whose elements we can give unique normal forms. We show that the 3-qubit Clifford+CS group, which is of course infinite, is the amalgamated product of these three finite subgroups. This result is analogous to the fact that the 1-qubit Clifford+T group is an amalgamated product of two finite subgroups.

1 Introduction

Just like Clifford+T circuits, the class of Clifford+CS circuits is universal for quantum computing [2]. Here, CS denotes the controlled-S gate. Amy, Glaudell, and Ross gave a characterization of the group of n -qubit Clifford+CS operators, showing that, up to a trivial condition on the determinant, a matrix is in this group if and only if it is unitary and its matrix entries belong to the ring $\mathbb{Z}[\frac{1}{2}, i]$ [2]. As a consequence of this, or alternatively since the CS gate is representable as a Clifford+T circuit with T-count 3, the Clifford+CS group is a subgroup of Clifford+T; see also [3]. Glaudell, Ross, and Taylor gave a normal form for 2-qubit Clifford+CS circuits [8]. In [9], Haah and Hastings showed how to construct a fault-tolerant CS-gate via magic state distillation. In [7], Garion and Cross described a CS- and CX-optimal canonical form for the 2-qubit group generated by the gates $\{X, T, CX, CS\}$.

This paper is motivated by the problem of optimizing Clifford+CS circuits. Like the T-gate, the CS-gate is a non-Clifford gate that is relatively expensive to perform in a fault-tolerant regime, requiring a magic state to be distilled [7]. It therefore makes sense to try to minimize the number of CS-gates. For example, one of the relations we found,

$$\text{Circuit 1} = \text{Circuit 2},$$

can sometimes be used to reduce the CS-count. Although we do not provide a method for minimizing the CS-count, we solve the important sub-problem of finding a complete set of relations for 3-qubit Clifford+CS circuits. This guarantees that any 3-qubit Clifford+CS circuit can be transformed into any other equivalent Clifford+CS circuit by the repeated application of a finite known set of relations.

Apart from giving a presentation of the group of 3-qubit Clifford+CS circuits by generators and relations, we also identify several interesting structures within this group along the way. Specifically, we identify three different finite subgroups for whose elements we can give unique normal forms. We show that the 3-qubit Clifford+CS group, which is of course infinite, is the amalgamated product of these three finite subgroups.

The CZ gate is usually denoted



but since it is symmetric with respect to its two qubits, we prefer the more symmetric notation shown above. We also use a similar notation for the CS gate, except that we label it with an “ i ”.

We number the qubits from top to bottom, and we write the circuits in the same order as matrix multiplication, i.e., from right to left. For example,

$$CS_{01}CZ_{12} = \text{---} \begin{array}{c} \text{---} \text{---} \\ | \quad | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \text{---} , \quad K_0S_1 = \begin{array}{c} \text{---} \text{---} \\ | \quad | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \text{---} = \begin{array}{c} \text{---} \text{---} \\ | \quad | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \text{---} .$$

Note that the X -gate and the controlled X -gate are definable as follows:

$$X = KSSKi, \quad \text{---} \begin{array}{c} \text{---} \\ | \\ \oplus \end{array} \text{---} = \text{---} \begin{array}{c} \text{---} \text{---} \\ | \quad | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} \text{---} \cdot i.$$

When we use the X - and controlled X -gates, for example in Figure 2, they are to be understood as abbreviations for these definitions.

2.2 A presentation of $U_n(\mathbb{Z}[\frac{1}{2}, i])$

We briefly recall a result from our earlier work [4]. As usual, \mathbb{Z} is the ring of integers. Let $\mathbb{Z}[\frac{1}{2}, i]$ be the smallest subring of the complex numbers containing $\frac{1}{2}$ and i . Let $U_n(\mathbb{Z}[\frac{1}{2}, i])$ be the group of unitary $n \times n$ -matrices with entries in $\mathbb{Z}[\frac{1}{2}, i]$.

In [4], we proved that the following is a presentation of $U_n(\mathbb{Z}[\frac{1}{2}, i])$ by generators and relations. The generators are $i_{[j]}$, $X_{[j,k]}$, and $K_{[j,k]}$, where $j, k \in \{0, \dots, n-1\}$ and $j < k$. The relations are shown in Figure 1. These relations are between words in the generators, and we write ε for the empty word (corresponding to the identity element of the group). The intended interpretation of the generators is as 1- and 2-level matrices; specifically, $i_{[j]}$ is like the identity matrix, except with i in the j th row and column, and $X_{[j,k]}$ and $K_{[j,k]}$ are like identity matrices, except with the entries of X , respectively K , in the j th and k th rows and columns, like this:

$$i_{[j]} = \begin{array}{c} \vdots \\ \vdots \\ j \\ \vdots \\ \vdots \end{array} \begin{array}{c} \dots & j & \dots \\ \left[\begin{array}{ccc} I & 0 & 0 \\ 0 & i & 0 \\ 0 & 0 & I \end{array} \right] \end{array}, \quad X_{[j,k]} = \begin{array}{c} \vdots \\ j \\ \vdots \\ k \\ \vdots \end{array} \begin{array}{c} \dots & j & \dots & k & \dots \\ \left[\begin{array}{ccccc} I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I \end{array} \right] \end{array}, \quad K_{[j,k]} = \begin{array}{c} \vdots \\ j \\ \vdots \\ k \\ \vdots \end{array} \begin{array}{c} \dots & j & \dots & k & \dots \\ \left[\begin{array}{ccccc} I & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{1+i} & 0 & \frac{1}{1+i} & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & \frac{1}{1+i} & 0 & \frac{-1}{1+i} & 0 \\ 0 & 0 & 0 & 0 & I \end{array} \right] \end{array}.$$

Theorem 2.1 ([4]). *Let \mathcal{G} be the set of one- and two-level matrices $i_{[j]}$, $X_{[j,k]}$, and $K_{[j,k]}$, where $j, k \in \{0, \dots, n-1\}$ and $j < k$. Let Δ be the set of relations shown in Figure 1. Then (\mathcal{G}, Δ) is a presentation of $U_n(\mathbb{Z}[\frac{1}{2}, i])$. In other words, the relations in Figure 1 are sound and complete for $U_n(\mathbb{Z}[\frac{1}{2}, i])$.*

2.3 The Reidemeister-Schreier theorem

We will also make use of a result known as the Reidemeister-Schreier theorem for monoids [10, 11, 5]. In a nutshell, if G is a monoid and H is a submonoid of G , the Reidemeister-Schreier theorem, under suitable assumptions, gives a method for deriving generators and relations for H from generators and relations for G . Giving a complete account of the Reidemeister-Schreier theorem is beyond the scope of this paper, but we refer the interested reader to Section 4.2 of [5] for a detailed explanation.

$$\begin{aligned}
i_{[j]}^4 &\sim \varepsilon & (1) & & i_{[k]}X_{[j,k]} &\sim X_{[j,k]}i_{[j]} & (10) \\
X_{[j,k]}^2 &\sim \varepsilon & (2) & & X_{[k,\ell]}X_{[j,k]} &\sim X_{[j,k]}X_{[j,\ell]} & (11) \\
K_{[j,k]}^8 &\sim \varepsilon & (3) & & X_{[j,\ell]}X_{[k,\ell]} &\sim X_{[k,\ell]}X_{[j,k]} & (12) \\
& & & & K_{[k,\ell]}X_{[j,k]} &\sim X_{[j,k]}K_{[j,\ell]} & (13) \\
& & & & K_{[j,\ell]}X_{[k,\ell]} &\sim X_{[k,\ell]}K_{[j,k]} & (14) \\
i_{[j]}i_{[k]} &\sim i_{[k]}i_{[j]} & (4) & & K_{[j,k]}i_{[k]}^2 &\sim X_{[j,k]}K_{[j,k]} & (15) \\
i_{[j]}X_{[k,\ell]} &\sim X_{[k,\ell]}i_{[j]} & (5) & & K_{[j,k]}i_{[k]}^3 &\sim i_{[k]}K_{[j,k]}i_{[k]}K_{[j,k]} & (16) \\
i_{[j]}K_{[k,\ell]} &\sim K_{[k,\ell]}i_{[j]} & (6) & & K_{[j,k]}i_{[j]}i_{[k]} &\sim i_{[j]}i_{[k]}K_{[j,k]} & (17) \\
X_{[j,k]}X_{[\ell,m]} &\sim X_{[\ell,m]}X_{[j,k]} & (7) & & K_{[j,k]}^2i_{[j]}i_{[k]} &\sim \varepsilon & (18) \\
X_{[j,k]}K_{[\ell,m]} &\sim K_{[\ell,m]}X_{[j,k]} & (8) & & K_{[j,k]}K_{[\ell,m]}K_{[j,\ell]}K_{[k,m]} &\sim K_{[j,\ell]}K_{[k,m]}K_{[j,k]}K_{[\ell,m]} & (19) \\
K_{[j,k]}K_{[\ell,m]} &\sim K_{[\ell,m]}K_{[j,k]} & (9) & & & &
\end{aligned}$$

Figure 1: A sound and complete set of relations for $U_n(\mathbb{Z}[\frac{1}{2}, i])$. In each relation, the indices are assumed to be distinct; moreover, whenever a generator $X_{[a,b]}$ or $K_{[a,b]}$ is mentioned, we assume $a < b$.

3 A presentation of Clifford+CS operators

In this section, we state our main result and give an outline of the proof. The full proof can be found in the accompanying Agda code [6].

Theorem 3.1. *The 3-qubit Clifford+CS group is presented by (\mathcal{X}, Γ_X) , where the set of generators is*

$$\mathcal{X} = \{i, K_0, K_1, K_2, S_0, S_1, S_2, CS_{01}, CS_{12}\},$$

and the set of relations Γ_X is shown in Figure 2.

One interesting feature of the axioms in Figure 2 is that the upside-down version of each relation is also a relation, except for (C15). The upside-down version of (C15) is provable, so we do not require it as an axiom.

3.1 Proof outline

Our proof follows a similar general outline as the corresponding proof for 2-qubit Clifford+ T operators in [5]. Let $G = U_8(\mathbb{Z}[\frac{1}{2}, i])$ be the group of unitary 8×8 -matrices with entries in $\mathbb{Z}[\frac{1}{2}, i]$. An exact synthesis algorithm for G was given by Amy, Glaudell, and Ross [2]. Based on this, we gave a presentation of G by generators and relations in [4]. It is clear that $\mathcal{CS}(3)$ is a subgroup of G , because all of its generators belong to G . Moreover, by a result of Amy et al. [2], we know that $\mathcal{CS}(3)$ is precisely the subgroup of G consisting of matrices whose determinant is ± 1 . The only other possible values for the determinant are $\pm i$, and therefore $\mathcal{CS}(3)$ is a subgroup of G of index 2. We can therefore apply the Reidemeister-Schreier procedure [10, 11] to find generators and relations for $\mathcal{CS}(3)$, given the known generators and relations for G . Applying this procedure yields a complete set of relations for $\mathcal{CS}(3)$.

The application of the Reidemeister-Schreier method produces thousands of relations, compared to the 17 cleaned-up relations in Figure 2. Moreover, these relations are very large. In our code, which

(a) Relations for $n \geq 0$:

$$i^4 = \varepsilon \tag{C1}$$

(b) Relations for $n \geq 1$:

$$K^2 = i^3 \tag{C2}$$

$$S^4 = \varepsilon \tag{C3}$$

$$SKSKSK = i^3 \tag{C4}$$

(c) Relations for $n \geq 2$:



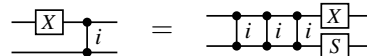
$$\tag{C5}$$



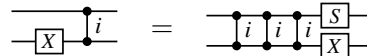
$$\tag{C6}$$



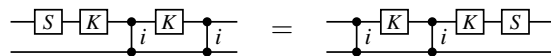
$$\tag{C7}$$



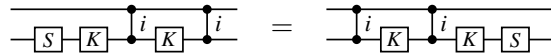
$$\tag{C8}$$



$$\tag{C9}$$

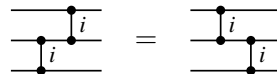


$$\tag{C10}$$

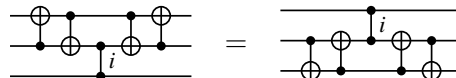


$$\tag{C11}$$

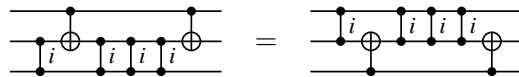
(d) Relations for $n = 3$:



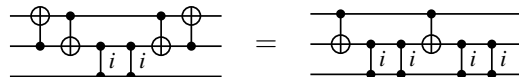
$$\tag{C12}$$



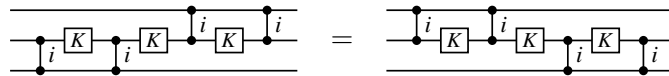
$$\tag{C13}$$



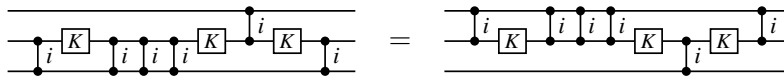
$$\tag{C14}$$



$$\tag{C15}$$



$$\tag{C16}$$



$$\tag{C17}$$

(e) Monoidal relations: the scalar i commutes with everything, and non-overlapping gates commute.

Figure 2: Complete relations for $\mathcal{CS}(3)$. Each relation in (b) denotes three relations (one for each qubit), and each relation in (c) denotes two relations (one for each pair of adjacent qubits).

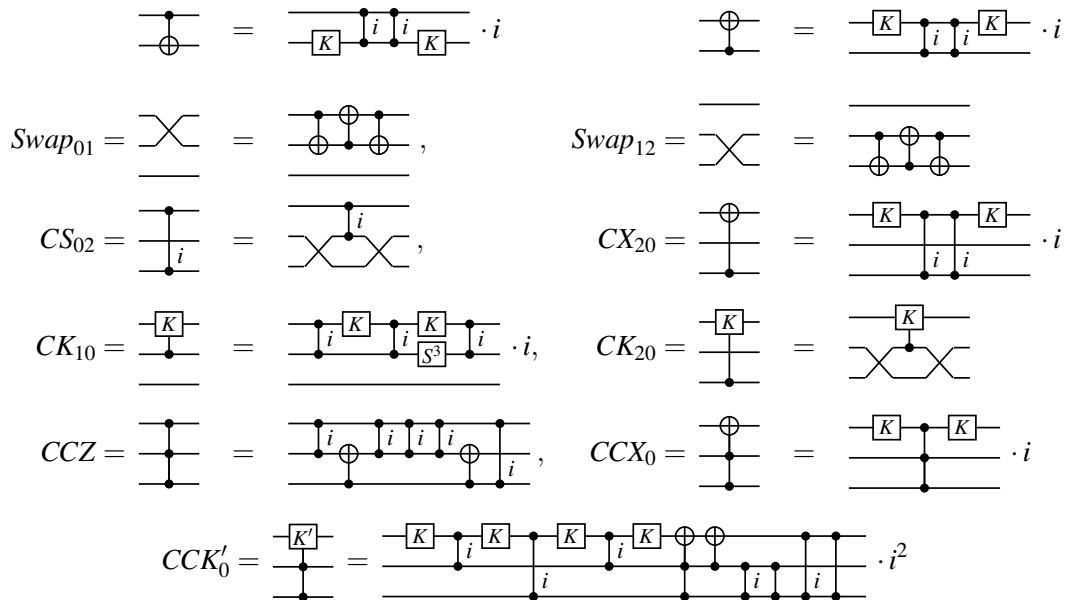
actually uses a sequence of multiple applications of the Reidemeister-Schreier theorem passing through a number of intermediate representations, some of the longest relations involve more than 50,000 generators. Our main contribution is the simplification of these relations. Due to the sheer magnitude of this task, we must rely on a computer to expedite the computation. However, we also require the simplification process to be trustworthy, as it is very easy in a computer program to accidentally use a relation that has not yet been proved. To this end, we have formalized Theorem 3.1 and its proof in the proof assistant Agda. This allows the proof to be verified independently and with a high degree of confidence in its correctness, despite the magnitude of the proof.

The main idea of the simplification is to use the 17 relations from Figure 2, along with some of their easy consequences, to rewrite the thousands of relations until they are all eliminated. We define several rewrite systems for this task. Some of these rewrite systems are confluent and terminating, and others are just heuristics. All of these rewrite systems are implemented in Agda and the computations are verified within Agda.

4 Normal forms and an almost-normal form

4.1 Notations

For convenience, we will use the following notations:



The first two notations extend to 3-qubit circuits, giving us the definitions of, for example, CX_{01} , and CX_{21} . The definitions for a Toffoli gate with target on the second, respectively first, qubit are given by $CCX_1 = Swap_{01} CCX_0 Swap_{01}$ and $CCX_2 = Swap_{12} CCX_1 Swap_{12}$. The last notation uses a twice-controlled K' gate. Here $K' = KS^\dagger$ is a variant of the K -gate that has determinant 1. The reason we are not using a twice-controlled K -gate is that it has determinant i and is therefore not an element of $\mathcal{CS}(3)$.

4.2 Normal forms for finite subgroups of Clifford+CS operators

We will define normal forms and discuss the structure of the following finite subgroups of Clifford+CS operators. The inclusion relations between these subgroups are visualized in Figure 3.

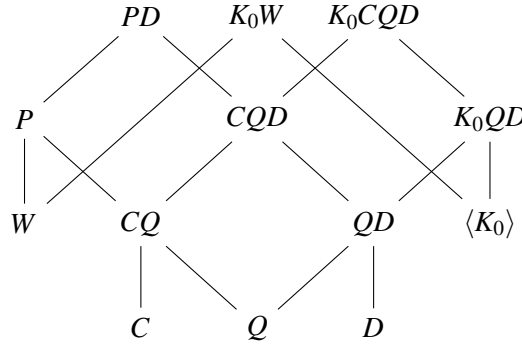


Figure 3: The inclusion graph of various finite subgroups of $\mathcal{CS}(3)$

- W , the subgroup of permutation matrices generated by $\mathcal{X}_W = \{\text{Swap}_{01}, \text{Swap}_{12}\}$.
- Q , the subgroup of permutation matrices generated by $\mathcal{X}_Q = \{X_0, CX_{10}, CX_{20}, CCX_0\}$.
- C , the subgroup of permutation matrices generated by $\mathcal{X}_C = \{X_1, CX_{12}, CX_{21}\}$.
- CQ , the subgroup generated by \mathcal{X}_C and \mathcal{X}_Q .
- P , the subgroup of permutation matrices generated by $\mathcal{X}_P = \{CX_{01}, CX_{10}, CX_{12}, CX_{21}, CCX_0, X_0\}$.
- D , the diagonal subgroup generated by $\mathcal{X}_D = \{i, S_0, S_1, S_2, CS_{01}, CS_{12}, CS_{02}, CCZ\}$.
- PD , the subgroup generated by \mathcal{X}_P and \mathcal{X}_D .
- QD , the subgroup generated by \mathcal{X}_Q and \mathcal{X}_D .
- CQD , the subgroup generated by \mathcal{X}_C , \mathcal{X}_Q and \mathcal{X}_D .
- K_0D the subgroup generated by $\{K_0\} \cup \mathcal{X}_D$. Note that this group contains Q , so it can also be denoted by K_0QD .
- K_0CD , the subgroup generated by $\{K_0\} \cup \mathcal{X}_C \cup \mathcal{X}_D$. Since this group contains Q , it can also be denoted by K_0CQD .
- K_0W , the subgroup generated by K_0 and \mathcal{X}_W .

Note that P is the group of all permutations of the computational basis vectors; we call its members “permutation operators”. Q , C , and CQ are subgroups of P . Similarly, D is the group of all diagonal operators in $\mathcal{CS}(3)$. The remaining subgroups play a technical role in our proofs.

Given that all claims about finite groups can be proved by just enumerating the elements, we will not give proofs of the following claims about finite subgroups of $\mathcal{CS}(3)$. Instead, we will illustrate the proofs with examples. Some of the proofs can be found in the Agda code.

The group W is the group of permutations of 3 qubits.

The generators of Q all commute with each other and are self-inverse. Therefore, each element of Q can be uniquely written of the form $X_0^a CX_{10}^b CX_{20}^c CCX_0^d$, where $a, b, c, d \in \{0, 1\}$. We say that the subgroup Q has the following normal form:

$$\bar{Q} ::= X_0^a CX_{10}^b CX_{20}^c CCX_0^d, \text{ where } a, b, c, d \in \{0, 1\}. \quad (20)$$

We use \overline{Q} to range over normal forms for Q . More generally, given any group G for which normal forms are defined, we use \overline{G} to range over the normal forms of G . The group Q has $2^4 = 16$ distinct normal forms corresponding to 16 distinct elements.

It is easy to see that $\text{Swap}_{12} \in C$, and therefore also $X_2 \in C$. The group C has the following normal form:

$$\overline{C} ::= c_4 c_3 c_2 \quad (21)$$

where

$$\begin{aligned} c_2 &\in \{\varepsilon, CX_{12}\}, \\ c_3 &\in \{\varepsilon, CX_{21}, CX_{12}CX_{21}\}, \\ c_4 &\in \{X_1^a X_2^b \mid a, b \in \{0, 1\}\}. \end{aligned}$$

There are $4! = 24$ distinct normal forms in C .

The group CQ is a semidirect product of C and Q with Q being normal. A semidirect product structure means that we have commuting relations of the form $qc = cq'$, or more precisely, for all $q \in Q$ and $c \in C$, there exists a unique $q' \in Q$ such that $qc = cq'$. Consequently, CQ has the following normal form:

$$\overline{CQ} ::= \overline{C} \overline{Q}.$$

The group P contains CQ as a subgroup with 105 cosets. We get the following normal form for P :

$$\overline{P} = c \overline{CQ}, \text{ where } c \text{ ranges over the set } V \text{ of 105 left coset representatives.} \quad (22)$$

One can easily spot a normal form for D , since all the generators commute with each other, CCZ has order 2, and all of the other generators have order 4. The normal form is:

$$\overline{D} ::= i^{n_0} S_0^{n_1} S_1^{n_2} S_2^{n_3} CS_{01}^{n_4} CS_{12}^{n_5} CS_{02}^{n_6} CCZ^{n_7}, \text{ where } n_0, \dots, n_6 \in \{0, 1, 2, 3\} \text{ and } n_7 \in \{0, 1\}. \quad (23)$$

The group PD is a semidirect product of P and D , with D being normal. It therefore has the following normal form:

$$\overline{PD} ::= \overline{P} \overline{D}. \quad (24)$$

Since Q is a subgroup of P , it follows that QD is also a semidirect product. It enjoys a similar normal form as (24), with P replaced by Q .

It is easy to see that group K_0D contains \mathcal{X}_Q , hence Q is a subgroup of K_0D . We have the following normal form:

$$\overline{K_0D} ::= e_4 e_3 e_2 e_1 \overline{D} \overline{Q}, \quad (25)$$

where

$$\begin{aligned} e_1 &\in \{\varepsilon, CCK'_0, CCK'_0 CCK'_0\}, \\ e_2 &\in \{\varepsilon, CK_{10}, S_0 CK_{10}\}, \\ e_3 &\in \{\varepsilon, CK_{20}, S_0 CK_{20}\}, \\ e_4 &\in \{\varepsilon, K_0, S_0 K_0\}. \end{aligned}$$

Note that CK_{10}, CK_{20} and K_0 commute with each other but not with CCK'_0 .

Notice that each element of \mathcal{X}_C commutes with K_0 . For any element of K_0CD , for example $w = X_1 K_0 CS_{01} K_0 CCZ$, we can commute X_1 all the way to the right using the commuting relations and the semidirect product structure of QD . For example, we get $w = K_0 CS_{01} CS_{01} CS_{01} S_0 K_0 CCZ CS_{02} CS_{02} X_1$. We will use the following normal form for K_0CD :

$$\overline{K_0CD} = (\overline{K_0D}) \overline{C} = e_4 e_3 e_2 e_1 \overline{D} \overline{Q} \overline{C}. \quad (26)$$

Note that this also proves that K_0CD is finite, which perhaps wasn't obvious from its definition.

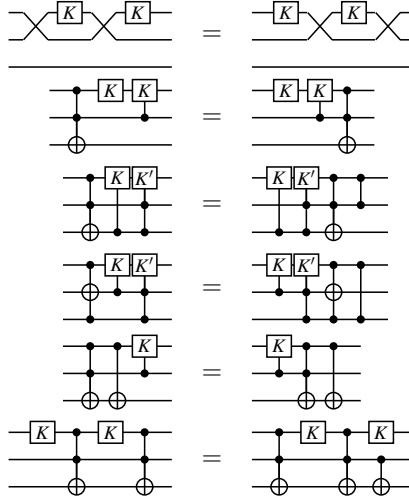


Figure 4: Some relations used to rewrite words of the form $ce_4e_3e_2e_1 ce_4e_3e_2e_1 \dots ce_4e_3e_2e_1$.

4.3 An almost-normal form for $\mathcal{CS}(3)$

Consider a Clifford+CS circuit. After replacing the generators K_1 and K_2 by $Swap_{01} K_0 Swap_{01}$ and $Swap_{12} Swap_{01} K_0 Swap_{01} Swap_{12}$, respectively, the circuit can be written as an alternating sequence of elements of PD and K_0 :

$$PK_0PK_0\dots PK_0PK_0$$

By repeatedly converting subcircuits to normal forms of the form (24), (22), and (26), we can rewrite this circuit as follows:

$$\begin{aligned}
& PK_0PK_0\dots PK_0PK_0 \\
& \xrightarrow{(24)(22)} c\overline{CQD}K_0c\overline{CQD}K_0\dots c\overline{CQD}K_0c\overline{CQD} \\
& \xrightarrow{(26)} ce_4e_3e_2e_1\overline{D}\overline{Q}\overline{C}c\overline{CQD}K_0\dots c\overline{CQD}K_0c\overline{CQD} \\
& \xrightarrow{(24)(22)} ce_4e_3e_2e_1c\overline{CQD}K_0\dots c\overline{CQD}K_0c\overline{CQD} \\
& \xrightarrow{repeat} ce_4e_3e_2e_1 ce_4e_3e_2e_1 \dots ce_4e_3e_2e_1c\overline{CQD}.
\end{aligned}$$

We can further rewrite the last expression, for example using relations in Figure 4. After this step, we might get some new gates that are not in V or of the form e_i . In this case, we continue with the first arrow step. We repeat the whole process until there is no further simplification. We call the resulting word an *almost-normal form*.

It turns out this almost-normal form is “canonical” enough. It can be used to show that a complete set of thousands of relations hold by rewriting both sides of each relation to almost-normal form. Moreover, all rewriting rules used to get an almost-normal form are consequences of the relations in Figure 2. This shows that the relations in Figure 2 are complete.

5 Clifford+CS is an amalgamated product of three finite groups

Let us first recall the definition of an amalgamated product of two monoids. For category theorists, this is simply a pushout: Given monoids M_1 , M_2 , and H with morphisms $H \rightarrow M_1$ and $H \rightarrow M_2$, the

amalgamated product $M_1 *_H M_2$ is the pushout

$$\begin{array}{ccc}
 H & \longrightarrow & M_2 \\
 \downarrow & & \downarrow \\
 M_1 & \dashrightarrow & M_1 *_H M_2.
 \end{array}$$

The amalgamated product of three monoids is defined similarly. Suppose $M_1, M_2, M_3, H_{12}, H_{23}, H_{13}$ are monoids with morphisms $H_{jk} \rightarrow H_j$ and $H_{jk} \rightarrow H_k$ for all relevant j and k . Then the amalgamated product P is the colimit of the following diagram, which generalizes a pushout:

$$\begin{array}{ccccc}
 & & H_{23} & & \\
 & & \downarrow & \searrow & \\
 & H_{13} & \longrightarrow & M_3 & \\
 & \downarrow & & \downarrow & \downarrow \\
 H_{12} & \longrightarrow & M_2 & \dashrightarrow & P \\
 & \searrow & \downarrow & & \\
 & & M_1 & \dashrightarrow & P.
 \end{array}$$

In terms of generators and relations, we have the following situation: Suppose we have three sets of generators X, Y , and Z , and three monoid presentations $M_1 = \langle X \cup Y \mid \Gamma_1 \rangle$, $M_2 = \langle X \cup Z \mid \Gamma_2 \rangle$, and $M_3 = \langle Y \cup Z \mid \Gamma_3 \rangle$. We can take $H_{12} = \langle X \rangle$, $H_{13} = \langle Y \rangle$ and $H_{23} = \langle Z \rangle$, with the obvious maps. Then the amalgamated product P has the presentation $\langle X \cup Y \cup Z \mid \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \rangle$.

In cases where P is an infinite monoid or group, it is remarkable when M_1, M_2 , and M_3 can be chosen to be finite. In that case, the slogan “the only relations that hold in P are relations that hold in a finite submonoid of P ” applies.

Using the results of this paper, we can show that $\mathcal{CS}(3)$ is an amalgamated product of three finite groups. We choose the sets of generators as follows:

$$\begin{aligned}
 X &= \{K_0, i\}, \\
 Y &= \{X_0, X_1, X_2, CX_{12}, CX_{21}, CX_{10}, CX_{20}, CCX_0, S_0, S_1, S_2, CS_{01}, CS_{12}, CS_{02}, CCZ, i\}, \\
 Z &= \{Swap_{01}, Swap_{12}\}.
 \end{aligned}$$

One can check that $\langle X \cup Y \rangle = K_0CQD$, $\langle X \cup Z \rangle = K_0W$, and $\langle Y \cup Z \rangle = PD$. Since $X \cup Y$, $X \cup Z$, and $Y \cup Z$ each generate a finite subgroup of $\mathcal{CS}(3)$, all that is left to show is that each relation of $\mathcal{CS}(3)$ is a consequence of relations in one of these three subgroups.

Before we prove this, we must adjust the relations of Figure 2 to fit the new set of generators $X \cup Y \cup Z$. This requires two adjustments. First, compared to the set of generators from Theorem 3.1, a number of new generators have been added, namely $X_0, X_1, X_2, CX_{12}, CX_{21}, CX_{10}, CX_{20}, CCX_0, CS_{02}, CCZ, Swap_{01}$, and $Swap_{12}$. For each of these, we must add a defining relation in terms of the old generators. These relations are as in Section 4.1. Second, the two generators K_1 and K_2 are no longer used, so where they appear in the relations, they must now be regarded as abbreviations for the words $Swap_{01} K_0 Swap_{01}$ and $Swap_{12} Swap_{01} K_0 Swap_{01} Swap_{12}$, respectively. With these adjustments, we still have a sound and complete presentation of $\mathcal{CS}(3)$ using the generators $X \cup Y \cup Z$.

Now we must show that each of the relations follows from relations that hold in $\langle X \cup Y \rangle$, $\langle X \cup Z \rangle$, or $\langle Y \cup Z \rangle$. Many of the relations, such as (C1), (C3), (C5)–(C9), and (C12) are already in one of the

three subgroups, so there is nothing else to show for them. The remaining relations must be proved individually; here, we give a proof of (C16) as a representative example. We have:

$$\begin{aligned}
& CS_{12} K_1 CS_{12} K_1 CS_{01} K_1 CS_{01} \\
&= CS_{12} Swap_{01} K_0 Swap_{01} CS_{12} Swap_{01} K_0 Swap_{01} CS_{01} Swap_{01} K_0 Swap_{01} CS_{01} \quad (1) \\
&= Swap_{01} CS_{02} K_0 CS_{02} K_0 CS_{01} K_0 CS_{01} Swap_{01} \quad (2) \\
&= Swap_{01} CS_{01} K_0 CS_{01} K_0 CS_{02} K_0 CS_{02} Swap_{01} \quad (3) \\
&= CS_{01} Swap_{01} K_0 Swap_{01} CS_{01} Swap_{01} K_0 Swap_{01} CS_{12} Swap_{01} K_0 Swap_{01} CS_{12} \quad (4) \\
&= CS_{01} K_1 CS_{01} K_1 CS_{12} K_1 CS_{12} \quad (5)
\end{aligned}$$

Here, steps (1) and (5) use the definition of K_1 , which is at this point merely an abbreviation for $Swap_{01} K_0 Swap_{01}$. Steps (2) and (4) uses the relations $Swap_{01}^2 = \varepsilon$ and $Swap_{01} CS_{12} Swap_{01} = CS_{02}$ and $Swap_{01} CS_{01} Swap_{01} = CS_{01}$. All three of these relations come from $\langle Y \cup Z \rangle$. Step (3) uses the relation $CS_{02} K_0 CS_{02} K_0 CS_{01} K_0 CS_{01} = CS_{01} K_0 CS_{01} K_0 CS_{02} K_0 CS_{02}$, which comes from $\langle X \cup Y \rangle$. In addition to (C16), there are a number of other relations to be proved, but they all follow a similar pattern.

As mentioned in the introduction, there is an analogous result for the 1-qubit Clifford+ T group, which is also an infinite group, and which is an amalgamated product of two finite subgroups. In this case, the finite subgroups are the Clifford group and the subgroup of diagonal and permutation operators, which is generated by T and X .

6 An overview of the accompanying Agda code

This paper is accompanied by a machine-checkable proof of Theorem 3.1 [6]. It has been formalized in the proof assistant Agda [1]. The proof assumes only the result of [4], i.e., the soundness and completeness of a certain set of relations for $U_n(\mathbb{Z}[\frac{1}{2}, i])$. Everything else is proved from first principles, including, for example, a complete proof of the version of the Reidemeister-Schreier theorem that we used.

Verifying the proof. Readers who are interested in verifying the proof only need to know the following: The *statement* of Theorem 3.1 is contained in the file `Theorem.agda`, and the final step of the *proof* of Theorem 3.1 is contained in the file `Proof.agda`. The reason we separated the statement of the theorem from its proof is to ensure that the statement assumes as little as possible: in fact, the file `Theorem.agda` is almost completely self-contained and only depends on a few definitions concerning generators, words, indices, and two-level relations. On the other hand, the proof requires a large number of auxiliary files with definitions, lemmas, tactics, and more. We checked the proof with Agda 2.6.4, and it took about 120 minutes on our laptop.

Reading the proof. For readers who are interested in inspecting our proof, here are some pointers. The folder `Lib` contains some general-purpose definitions, such as booleans and natural numbers, and some definitions and tactics related to monoids and relations. The main parts of the proof are contained in the folders `Step1` – `Step8`. Each of these steps transforms a set of generators and relations into an equivalent set of generators and relations, gradually simplifying the relations. The file `Gate.agda` provides the definitions for all gates used. The file `CosetNF.agda` contains definitions related to semidirect products and normal forms. The final proof witness is contained in the file `Proof.agda`.

7 Conclusion and future work

The main result of this paper is a presentation of the group of 3-qubit Clifford+CS operators by just 17 relatively simple relations. We proved this by a combination of a previous result from [4], the Reidemeister-Schreier method, and an Agda program that simplified several thousand large relations into the aforementioned 17 simple ones. Doing this simplification by brute force would not have been feasible. Instead, we proceeded by identifying a number of finite subgroups of the Clifford+CS operators, defining normal forms for these, and then combining them into carefully chosen rewrite rules. After months of fine-tuning, these rules eventually reduced the relations to a manageable size.

Unlike our previous work on generators and relations for 2-qubit Clifford+ T operators [5], which used a *Pauli rotation decomposition* to guide the rewriting, we found that the analog of the Pauli rotation decomposition, i.e., taking syllables that are conjugates of the CS gate under the action of the Clifford operators, does not work very well. Instead, we were surprised to find that a more useful decomposition was to take conjugates of K_0 (basically a Hadamard gate) under the action of diagonal and permutation operators. We may call this the *Hadamard decomposition* of Clifford+CS. In the process, we learned many interesting facts about finite subgroups of Clifford+CS. One of these facts is that the 3-qubits Clifford+CS group is an amalgamated product of three of its finite subgroups. Concretely, this means that every relation that holds in this group follows from relations that already hold in some finite subgroup of Clifford+CS.

This work suggests some interesting directions for future work. Many of our results about finite subgroups of Clifford+CS are valid for n qubits, so one may ask whether our generators and relations can also be extended to circuits with 4 or more qubits. Currently, the limiting factor is the prohibitive computational cost of applying the Reidemeister-Schreier method to a set of 2-level relations for 16×16 -matrices and then simplifying a massive set of relations. Perhaps a further study of the finite subgroups of Clifford+CS will open up an alternative path to this problem. For example, one may ask whether the n -qubit Clifford+CS group is an amalgamated product for all n . One may further ask the same question for the n -qubits Clifford+ T group or its other subgroups of interest, such as the Clifford+Toffoli group.

The fact that the Hadamard decomposition turned out to be more useful than the analog of the Pauli rotation decomposition raises the question whether our earlier work on Clifford+ T could benefit from the same insight. By applying these lessons, perhaps one can come up with a simpler complete set of relations. For example, our Clifford+ T axiomatization involved a number of obvious relations and three “non-obvious” ones. We were never able to resolve the question of whether these non-obvious relations actually follow from something simpler.

Another intriguing question is whether one can find a unique normal form for 3-qubit Clifford+CS circuits, like the Matsumoto-Amano normal form for 1-qubit Clifford+ T circuits. We currently only have an “almost-normal” form, but the fact that it efficiently reduced all of our relations is encouraging.

References

- [1] *Agda Documentation*. <https://agda.readthedocs.io/>. Accessed: 2023-03-17.
- [2] Matthew Amy, Andrew N. Glauddell & Neil J. Ross (2020): *Number-theoretic characterizations of some restricted Clifford+ T circuits*. *Quantum* 4, p. 252, doi:10.22331/q-2020-04-06-252. Also available from arXiv:1908.06076.
- [3] Michael Beverland, Earl Campbell, Mark Howard & Vadym Kliuchnikov (2020): *Lower bounds on the non-Clifford resources for quantum computations*. *Quantum Science and Technology* 5(3), p. 035009, doi:10.1088/2058-9565/ab8963.

- [4] Xiaoning Bian & Peter Selinger (2021): *Generators and relations for $U_n(\mathbb{Z}[1/2, i])$* . In: *Proceedings of the 18th International Conference on Quantum Physics and Logic, QPL 2021, Gdansk, Poland, Electronic Proceedings in Theoretical Computer Science* 343, pp. 145–164, doi:10.4204/EPTCS.343.8.
- [5] Xiaoning Bian & Peter Selinger (2022): *Generators and relations for 2-qubit Clifford+T operators*. To appear in *QPL 2022*. Available from arXiv:2204.02217.
- [6] Xiaoning Bian & Peter Selinger (2023): *Agda code accompanying this paper*. Available from <https://www.mathstat.dal.ca/~selinger/papers/downloads/cliffordcs3/>.
- [7] Shelly Garion & Andrew W Cross (2020): *Synthesis of CNOT-dihedral circuits with optimal number of two qubit gates*. *Quantum* 4, p. 369, doi:10.22331/q-2020-12-07-369.
- [8] Andrew N. Glauddell, Neil J. Ross & Jacob M. Taylor (2021): *Optimal two-qubit circuits for universal fault-tolerant quantum computation*. *npj Quantum Information* 7(1), p. 103, doi:10.1038/s41534-021-00424-z.
- [9] Jeongwan Haah & Matthew B Hastings (2018): *Codes and protocols for distilling T, controlled-S, and Toffoli gates*. *Quantum* 2, p. 71, doi:10.22331/q-2018-06-07-71.
- [10] Kurt Reidemeister (1927): *Knoten und Gruppen*. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 5(1), pp. 7–23, doi:10.1007/BF02952506.
- [11] Otto Schreier (1927): *Die Untergruppen der freien Gruppen*. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 5(1), pp. 161–183, doi:10.1007/BF02952517.
- [12] Peter Selinger (2015): *Generators and relations for n-qubit Clifford operators*. *Logical Methods in Computer Science* 11(2:10), pp. 1–17, doi:10.2168/LMCS-11(2:10)2015. Also available from arXiv:1310.6813.

Complete Equational Theories for the Sum-Over-Paths with Unbalanced Amplitudes

Matthew Amy

School of Computing Science
Simon Fraser University
meamy@sfu.ca

Vilmart recently gave a complete equational theory for the balanced sum-over-paths over Toffoli-Hadamard circuits, and by extension Clifford+diag($1, \zeta_{2^k}$) circuits. Their theory is based on the phase-free ZH-calculus which crucially omits the average rule of the full ZH-calculus, dis-allowing the local summation of amplitudes. Here we study the question of completeness in unbalanced path sums which naturally support local summation. We give a concrete syntax for the unbalanced sum-over-paths and show that, together with symbolic multilinear algebra and the interference rule, various formulations of the average and ortho rules of the ZH-calculus are sufficient to give complete equational theories over arbitrary rings and fields.

1 Introduction

The balanced sum-over-paths representation of a linear operator $\Psi : \mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n}$ introduced in [1] is a symbolic representation of Ψ over Boolean-valued variables, having the form

$$\Psi |\vec{x}\rangle = \mathcal{N} \sum_{\vec{y} \in \mathbb{Z}_2^k} e^{2\pi i P(\vec{x}, \vec{y})} |f(\vec{x}, \vec{y})\rangle \quad (1)$$

where $P : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{R}/2\pi$ and $f : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{Z}_2^n$ are represented by (systems of) polynomials in $m+k$ variables. The expression of Equation (1) is interpreted as a sum over the *paths* a system may take beginning from some initial configuration \vec{x} . If Ψ is taken as encoding some evolution of a physical (2^k -level) system, the expression Equation (1) coincides roughly with Richard Feynman's *path integral* [7]. As with Feynman's path integral the sum in Equation (1) is *balanced*, in that each path — indexed by the values of \vec{x} and \vec{y} — has the same amplitude \mathcal{N} but varies in the phase $e^{2\pi i P(\vec{x}, \vec{y})}$. As standard operators used in quantum computation can be represented in this form and the representation is closed over composition and tensor products, typical quantum computational processes admit representations by balanced path sums.

It was previously shown [1] that with a concrete representation in terms of polynomials, the balanced sum-over-paths admits a simple equational theory. This equational theory was shown to be relatively complete, and with a small modification complete [9], for Clifford operators. As the number of superfluous variables y_i in the sum, called *internal* and corresponding to a pair of paths between two end-points, offers an intuitive notion of complexity of the expression, the equational theory further gives rise to a natural re-write system which iteratively removes such variables from the sum. This re-write system was shown to terminate in polynomial time with a unique normal form for Clifford operators [2], and was further shown to perform well in practice on Clifford+ T circuits and channels for verification [1].

Vilmart [10] gave an equational theory for the balanced sum-over-paths over Toffoli and Hadamard gates and showed its completeness via translation of the phase-free ZH-calculus [11]. The phase-free

ZH-calculus crucially restricts the ZH-calculus [4] to balanced generators, allowing the direct translation of its equational theory into balanced sums — while the full ZH-calculus admits an encoding in the balanced sum-over-paths [8, 9], existing equational theories [4, 5] necessarily make use of unbalanced generators and the *average* rule. It was further shown that this equational theory is complete for operators over Clifford and $R_k := \text{diag}(1, \zeta_{2^k})$ gates through an embedding into the Toffoli+Hadamard fragment. However, their re-writing system lacks the desirable properties of confluence, normal forms, and a primitive equational theory for Clifford+ R_k . Moreover, the question of a complete equational theory for the sum-over-paths over \mathbb{C} was left open, as well as the development of a direct analogue of the full ZH-calculus in the sum-over-paths model.

In this work we address these questions, introducing a concrete representation of the *unbalanced* sums-over-paths and give complete equational theories for such sums over rings and fields. We consider general rings as fault-tolerantly constructible circuits are typically restricted to linear operators over subrings of \mathbb{C} [3]. Inspired by the ZH-calculus, completeness is attained by symbolically re-writing a sum to a unique normal form explicitly encoding the matrix entries. In essence, our system internalizes the method of falling back to explicit evaluation in cases where no progress can be made with re-writing [1]. As a result, the practical applicability of our equational theory is limited — our goals are instead to provide representations and complete theories for general sums-over-paths so that effective re-writing systems may be further developed.

The paper is organized as follows. In [Section 2](#) we review the balanced sum-over-paths and its equational theories. In [Section 3](#) we define a representation of the unbalanced sum-over-paths and give an equational theory which is complete for arbitrary rings. In [Section 4](#) we give a weaker equational theory which is not sound over rings, but is shown to be complete over any field.

2 The balanced sum-over-paths

In the path integral viewpoint, the action of a linear operator $\Psi : \mathcal{H}_1 \rightarrow \mathcal{H}_2$ between Hilbert spaces \mathcal{H}_1 and \mathcal{H}_2 on a state $|i\rangle$ of some orthonormal basis $\{|i\rangle\}$ of \mathcal{H}_1 can be described as a sum over some collection Π of paths, where the path $\pi \in \Pi$ has amplitude $\psi(\pi) \in \mathbb{C}$ and ends in a state $|f(\pi)\rangle$ of an orthonormal basis of \mathcal{H}_2 :

$$\Psi|i\rangle = \sum_{\pi \in \Pi} \psi(\pi) |f(\pi)\rangle.$$

In contrast to the operator representation $\Psi|i\rangle = \sum_j \alpha_{ij} |j\rangle$, in general there may be many superimposed paths leading to a particular basis state, their amplitudes adding in these cases and resulting in interference. Likewise, the sequential composition of two operators is described by composing paths along their endpoints and multiplying the amplitude along each segment, in essence delaying evaluation of any interfering paths. This provides flexibility in the evaluation of individual amplitudes, but on the flip side requires effective means of representing and reasoning about a system of paths to be useful.

In [1] a concrete representation of *amplitude-balanced* sums over 2^n -dimensional Hilbert spaces was given via multilinear polynomials. An amplitude-balanced sum is one in which every path π with non-zero amplitude has equal magnitude but may vary in the phase. By restricting to balanced sums, individual path amplitudes are described by unit-norm complex numbers, whose multiplicative group is isomorphic to the additive group $\mathbb{R}/2\pi$. In particular, $\psi(\pi) \approx e^{2\pi i P(\pi)}$ for some $P : \Pi \rightarrow \mathbb{R}/2\pi$ which has a unique representation as a multilinear polynomial, and if $\phi(\pi') \approx e^{2\pi i Q(\pi')}$, then $\psi(\pi) \cdot \phi(\pi') \approx e^{2\pi i [P(\pi) + Q(\pi')]}$ — i.e. the phase along a composite path — is uniquely representable in polynomial time. We review this representation below.

Definition 2.1 (Balanced sum-over-paths). A balanced path sum is an expression of the form

$$\Psi|\vec{x}\rangle = \mathcal{N} \sum_{\vec{y} \in \mathbb{Z}_2^k} e^{2\pi i P(\vec{x}, \vec{y})} |f(\vec{x}, \vec{y})\rangle,$$

where $\mathcal{N} \in \mathbb{C}$ and $P: \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{R}/2\pi$ and $f: \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{Z}_2^n$ are represented as a multilinear polynomial and a sequence of n multilinear polynomials in $m+k$ variables, respectively.

We use $|\Psi\rangle$ to denote the sum-over-paths representation of Ψ , or just Ψ when it is clear from the context that we mean the symbolic expression rather than the linear operator. The variables appearing in a sum-over-paths expression $|\Psi\rangle$ which are not summed over are called *free variables*. We denote the set of free variables of Ψ by $FV(\Psi)$ and for the purpose of substitution use the notation $|\Psi(x)\rangle$ to identify a variable which may appear free in $|\Psi\rangle$. Specifically, given a sum-over-paths $|\Psi(x)\rangle$, $|\Psi(f)\rangle$ denotes the (capture-avoiding) substitution of f for every free occurrence of x in $|\Psi\rangle$, of which there may be none. We say a sum $|\Psi\rangle$ is *closed* if $FV(\Psi) = \emptyset$, in which case $|\Psi\rangle$ corresponds to a vector. As a convention, we often use variable names x_i to denote the free variables of a path sum and y_i, z_i to denote variables which are summed, though it should be understood that this is not a rule and sum-over-paths expressions may contain arbitrary variables in free or summed positions. We further use letters f, g, h to refer to Boolean-valued functions or symbolic expressions, and uppercase letters P, Q, R to refer to symbolic expressions in other rings.

By linearity, compatible balanced sums may be sequentially composed through variable substitution:

$$\Phi\Psi|\vec{x}\rangle = \mathcal{N} \sum_{\vec{y} \in \mathbb{Z}_2^k} e^{2\pi i P(\vec{x}, \vec{y})} \Phi|f(\vec{x}, \vec{y})\rangle.$$

As substitution involves substituting symbolic expressions over \mathbb{Z}_2 in the phase, a *lifting* construction [1] is used to embed polynomial arithmetic over \mathbb{Z}_2 into polynomials over \mathbb{R} (or more generally, any unital ring \mathcal{R}). In particular, we define the lifting $\bar{\cdot}$ of $(\mathbb{Z}_2, \oplus, \cdot)$ into $(\mathcal{R}, +, \cdot)$ recursively by

$$\begin{aligned} \bar{0} &= 0_{\mathcal{R}} & \overline{f \cdot g} &= \bar{f} \cdot \bar{g} \\ \bar{1} &= 1_{\mathcal{R}} & \overline{f \oplus g} &= \bar{f} + \bar{g} - (2 \cdot \bar{f} \cdot \bar{g}). \end{aligned}$$

The tensor product, or parallel composition, can be defined via

$$(\Psi \otimes \Phi)(|\vec{x}\rangle \otimes |\vec{y}\rangle) = \Psi|\vec{x}\rangle \otimes \Phi|\vec{y}\rangle,$$

where the phases of $\Psi|\vec{x}\rangle$ and $\Phi|\vec{y}\rangle$ are multiplied and their final states are concatenated.

Example 2.2. The gates X and T admit the following representations as balanced sums:

- X $|x\rangle = |1 \oplus x\rangle$ and
- T $|y\rangle = \omega^y |y\rangle$, where $\omega = e^{\frac{2\pi i}{8}}$.

Composing T after X, we can compute the sum-over-paths expression of TX by substituting y in the expression of T with $x \oplus 1$, which lifts to $\overline{1 \oplus x} = 1 - x$ in the exponent of ω :

$$\text{TX}|x\rangle = \omega^{1-x} |1 \oplus x\rangle$$

While the paths in a balanced path sum with non-zero amplitude all have the same magnitude \mathcal{N} , matrices which have entries of different magnitudes can be represented as balanced sums with interfering paths, as the following example illustrates.

Example 2.3. The controlled-Hadamard gate

$$\Lambda(H) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes H = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{2} & 0 & 0 & 0 \\ 0 & \sqrt{2} & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

when viewed as a collection of *unique* transitions (i.e. non-interfering) in the computational basis is necessarily unbalanced. However, the $\Lambda(H)$ gate may be represented as a balanced sum-over-paths by using the control bit to cause the intermediate paths to interfere when it is in the 0 state:

$$\Lambda(H) |x_1 x_2\rangle = \frac{1}{\sqrt{2}} \sum_{y \in \mathbb{Z}_2} \omega^{(1-x_1)(2y-1)} (-1)^{x_1 x_2 y} |x_1\rangle |(1 \oplus x_1)x_2 \oplus x_1 y\rangle$$

Note that when $x_1 = 0$ we have $\frac{1}{\sqrt{2}} \sum_{y \in \mathbb{Z}_2} \omega^{(2y-1)} |0\rangle |x_2\rangle = \frac{\omega + \omega^\dagger}{\sqrt{2}} |0\rangle |x_2\rangle = |0\rangle |x_2\rangle$, while when $x_1 = 1$ we have the sum $\frac{1}{\sqrt{2}} \sum_{y \in \mathbb{Z}_2} (-1)^{x_2 y} |1\rangle |y\rangle$ which is the representation of $|1\rangle \otimes (H|x_2\rangle)$.

It has been shown that the balanced sum-over-paths is universal for linear operators over qubit (2^n -dimensional) Hilbert spaces, via a translation from the universal ZH-calculus [9]. Below we give a model of the universal ZX-calculus [6] which is more natural to specify in the balanced sum-over-paths as the ZX-calculus is generated by balanced operators, while the H-boxes of the ZH-calculus are unbalanced when the amplitude is non-unital.

Example 2.4. A simple model of the ZX-calculus via balanced sums can be defined over the universal generating set consisting of the Z-spider and Hadamard as so:

$$\llbracket n \text{ : } \alpha \text{ : } m \rrbracket |\vec{x}\rangle = \frac{1}{2^n} \sum_{\vec{y} \in \mathbb{Z}_2^n, z \in \mathbb{Z}_2} \alpha^z (-1)^{\sum_{i=1}^n y_i (x_i + z)} |zz \dots z\rangle \quad \llbracket - \square - \rrbracket |x\rangle = \frac{1}{\sqrt{2}} \sum_{y \in \mathbb{Z}_2} (-1)^{xy} |y\rangle$$

Note that the sum for the Z-spider forces any path with non-zero amplitude to satisfy $z = x_1 = x_2 = \dots = x_n$, since for any i , $\sum_{y_i \in \mathbb{Z}_2} (-1)^{y_i (x_i + z)} = 0$ whenever $z \neq x_i$, and 2 otherwise. Encodings of an arbitrary Z-spider using fewer variables are possible, for example $\frac{1}{2^n} \sum_{y, z \in \mathbb{Z}_2} \alpha^z (-1)^{y(1 + \prod_{i=1}^n (x_i + z + 1))} |zz \dots z\rangle$, but have size exponential in n .

We use \equiv_T to denote the equivalence of two path sums up to a theory T defined by a set of (sound) equations, together with the congruence

$$|\Psi\rangle \equiv_T |\Phi\rangle \implies \mathcal{N} \sum_{\vec{y} \in \mathbb{Z}_2^k} e^{2\pi i P(\vec{x}, \vec{y})} |\Psi\rangle \equiv_T \mathcal{N} \sum_{\vec{y} \in \mathbb{Z}_2^k} e^{2\pi i P(\vec{x}, \vec{y})} |\Phi\rangle$$

We use \equiv_e to denote equivalence up to an individual equation e . We say an equational theory T is *complete* for a subset \mathcal{C} of path sums if whenever $|\Psi\rangle, |\Phi\rangle \in \mathcal{C}$,

$$\Psi = \Phi \implies |\Psi\rangle \equiv_T |\Phi\rangle.$$

Figure 1 gives the (E), (H), and (ω) rules of the sum-over-paths which define the equational theory \equiv_{Cliff} . It was previously shown that \equiv_{Cliff} is complete for Clifford path sums [9, 2].

$$\sum_{y \in \mathbb{Z}_2} |\Psi\rangle \equiv 2 |\Psi\rangle \quad (\text{E})$$

$$\sum_{x, y \in \mathbb{Z}_2} (-1)^{y(x+f)} |\Psi(x)\rangle \equiv 2 |\Psi(f)\rangle \quad (\text{H})$$

$$\sum_{y \in \mathbb{Z}_2} i^y (-1)^{yf} |\Psi\rangle \equiv \omega \sqrt{2} (-i)^{\bar{f}} |\Psi\rangle \quad (\omega)$$

Figure 1: A Clifford-complete system of equations for the balanced sum-over-paths, denoted \equiv_{Cliff} . In all rules above $y \notin FV(\Psi)$ and f is some Boolean expression such that $x, y \notin FV(f)$.

Example 2.5. The following equalities are derivable:

$$\begin{aligned} \sum_{y \in \mathbb{Z}_2} |\Psi(y)\rangle &\equiv_{\text{Cliff}} \sum_{y \in \mathbb{Z}_2} |\Psi(y \oplus f)\rangle \text{ where } f \text{ is Boolean and } y \notin FV(f) \\ \frac{1}{\sqrt{2}} \sum_{y \in \mathbb{Z}_2} \omega^{(2y-1)} |0\rangle |x_2\rangle &\equiv_{\text{Cliff}} |0\rangle |x_2\rangle \\ \llbracket (-\square-)^{\otimes m} \circ \textcircled{\alpha} \circ (-\square)^{\otimes n} \rrbracket |\vec{x}\rangle &\equiv_{\text{Cliff}} \frac{1}{2^{n+m}} \sum_{y \in \mathbb{Z}_2, \vec{z} \in \mathbb{Z}_2^m} \alpha^y (-1)^{\sum_{i=1}^n x_i y} (-1)^{\sum_{j=1}^m y z_j} |\vec{z}\rangle \end{aligned}$$

The first equation is the *variable change* rule from the Clifford-complete equational theory of [9], whose derivation by the (E) rule was shown in [2]. The second encodes the evaluation of $\Lambda(\text{H}) |0\rangle |x_2\rangle = |0\rangle |x_2\rangle$ and follows from a single application of (ω). The third equation models the X-spider of the ZX-calculus by the color change law [6].

Interference, algebraic varieties, and completeness for Toffoli+Hadamard As noted in [1], (H) arises as an instance of a general (binary) *interference* rule,

$$\sum_{y \in \mathbb{Z}_2} (-1)^{yF} |\Psi\rangle \equiv 2 |\Psi|_{F=0}\rangle. \quad (\text{I})$$

where F is a polynomial over \mathbb{Z}_2 . Viewing F as a proposition on the paths indexed by $FV(F)$, the sum $\sum_{y \in \mathbb{Z}_2} (-1)^{yF} |\Psi\rangle$ filters out paths satisfying F , while paths which do not satisfy F pass through with double amplitude. However, to write the restriction $|\Psi|_{F=0}\rangle$ as an expression, $|\Psi\rangle = \mathcal{N} \sum_{\vec{y}} e^{2\pi i P(\vec{x}, \vec{y})} |f(\vec{x}, \vec{y})\rangle$ must be expressed as a sum over the solutions of the equation $F(\vec{x}, \vec{y}) = 0$. Recall that the algebraic variety $\mathcal{V}(I)$ of a polynomial ideal I consists of all points (a_1, \dots, a_k) such that $f(a_1, \dots, a_k) = 0$ for every polynomial f in I . We may hence write the restricted sum as

$$\mathcal{N} \sum_{(\vec{x}, \vec{y}) \in \mathcal{V}(I)} e^{2\pi i P(\vec{x}, \vec{y})} |f(\vec{x}, \vec{y})\rangle,$$

where $I = \langle F \rangle$. Note that P and f may be canonically written modulo the ideal I using Gröbner bases, though the resulting re-write system is not an equational theory in the sense we consider here. Instead we may restrict f to cases which can be solved by substitution. The simplest such cases are when $F = 0$ which is solved trivially for any point in the variety, and when $F = x + g$, $x \notin FV(g)$, which is solved by

$$\sum_{y \in \mathbb{Z}_2} \sum_{x \in \mathbb{Z}_2} (-1)^{y(x \cdot g + g \cdot f + 1)} |\Psi(x)\rangle \equiv \sum_{y \in \mathbb{Z}_2} (-1)^{y(g+1)} |\Psi(1+f)\rangle \quad (\text{Hgen})$$

$$\sum_{y \in \mathbb{Z}_2} \sum_{x \in \mathbb{Z}_2} (-1)^{y \cdot f + x \cdot g} |\Psi\rangle \equiv 2 \sum_{y \in \mathbb{Z}_2} (-1)^{y(f+g+f \cdot g)} |\Psi\rangle \quad (\text{Hrel})$$

$$\sum_{y \in \mathbb{Z}_2} \sum_{x \in \mathbb{Z}_2} (-1)^y |\Psi\rangle \equiv 0 |\Psi\rangle \quad (\text{Z})$$

Figure 2: A complete equational theory \equiv_{TH} for path sums over Toffoli and Hadamard gates [10]. In all rules $y \notin FV(\Psi)$ and f, g are Boolean expressions such that $x, y \notin FV(f) \cup FV(g)$

setting $x = g$. These two cases result in the (E) and (H) rules. Moreover, both equations are *complete* relative to the variety $\mathcal{V}(I)$ in that they completely characterize the solutions to f .

In [10] Vilmart gave a complete equational theory for path sums over Toffoli and Hadamard, via restricted cases of (I). Such sums can be written in the form

$$\Psi|\vec{x}\rangle = \frac{1}{\sqrt{2^k}} \sum_{\vec{y}} (-1)^{P(\vec{x}, \vec{y})} |f(\vec{x}, \vec{y})\rangle.$$

We define \equiv_{TH} to be equivalence of balanced sums up to \equiv_{Cliff} as well as the additional rules of Figure 2. As noted in [10], all three new equations arise as particular instances of the binary interference rule. The (Hgen) rule, which subsumes (H), arises from (I) when $F = x \cdot g + g \cdot f + 1$, in which case $x + g + 1 \in \langle F \rangle$ and hence $x = g + 1$ is a partial solution to $F = 0$. Likewise, (Hrel) arises from the intersection of the varieties $f = 0$ and $g = 0$, where since $\langle f, g \rangle = \langle f + g + f \cdot g \rangle$ over \mathbb{Z}_2 the two equations to be combined into a single one without affecting the variety. The (Z) rule arises when $F = 1$ and hence the variety is empty.

3 The unbalanced sum-over-paths

While computationally efficient for many problems, balanced sums are unwieldy for reasoning about vectors and matrices with entries of varying magnitude. Such states often arise in probabilistic quantum computations and algorithms, such as Grover's search or Shor's algorithm. Moreover, canonical forms for such operators are difficult to define and test for equality, as the following example illustrates.

Example 3.1. Consider the unit vector

$$|\psi\rangle = \frac{1}{\sqrt{1+p^2}}(|0\rangle + p|1\rangle)$$

where p is an odd prime. Any representation of $|\psi\rangle$ by a balanced sum with ± 1 phases must satisfy $\langle 0|\psi\rangle = 1$, $\langle 1|\psi\rangle = p$, and hence must consist of at least $p + 1$ distinct paths. Now let f and g be Boolean expressions in free variables $\{y_i\}$ with 1 and p satisfying assignments, respectively. Then

$$\frac{1}{2\sqrt{1+p^2}} \sum_{x \in \mathbb{Z}_2} \sum_{\vec{y} \in \mathbb{Z}_2^k} \sum_{z \in \mathbb{Z}_2} (-1)^{z[(1-x)(1+f(\vec{y})) + x(1+g(\vec{y}))]} |x\rangle$$

is a valid representation of $|\psi\rangle$, as is the representation above where f and g are replaced with any other Boolean expressions with the same number of solutions. By (I), we can re-write this sum over the variety

generated by the ideal $I = \langle (1-x)(1+f(\vec{y})) + x(1+g(\vec{y})) \rangle$ as $|\Psi\rangle = \sum_{(x,\vec{y}) \in \mathcal{V}(I)} |x\rangle$. However, if we take this as a normal form it is surely not unique, as any other variety with the same number of points for each x — for instance, any variety \mathcal{W} equal up to a permutation of the coordinates of \mathcal{V} — gives the same operator.

In order to allow the natural representation of linear algebraic objects with unbalanced magnitudes, we now develop a generalization to amplitudes which may be expressed as Boolean powers of elements taken from some ring \mathcal{R} . Recall that integer powers may be defined in any unital ring \mathcal{R} as

$$0^0 := 1, \quad r^0 := 1, \quad r^n := r \cdot r^{n-1}.$$

Our language of sums is comprised of expressions of three types: Boolean expressions used to denote paths, \mathcal{R} -valued expressions, and linear operators over \mathcal{R} in the computational basis.

Definition 3.2 (Unbalanced sum-over-paths). An unbalanced sum-over-paths is an expression $|\Psi\rangle$ of the following language

$$\begin{aligned} f &::= 0 \mid 1 \mid x \mid f_1 \cdot f_2 \mid f_1 \oplus f_2 \mid \neg f := 1 \oplus f \\ r &::= \alpha, \beta \in \mathcal{R} \mid r^f \mid r_1 r_2 \mid r_1 + r_2 \\ |\Psi\rangle &::= \sum_{\vec{y}} r |f_1 \cdots f_n\rangle. \end{aligned}$$

Variables x, y, z, \dots in all types of expressions range over Boolean values \mathbb{Z}_2 and \mathcal{R} is a commutative ring.

Expressions f , r , and $|\Psi\rangle$ are referred to as Boolean, \mathcal{R} , and sum-over-paths expressions, respectively. We denote by $FV(f)$, $FV(r)$, and $FV(\Psi)$ the free variables appearing in the given expression. As with balanced sums, variables \vec{y} which are summed over are bound and hence not included in the set of free variables of an unbalanced sum. An expression is *closed* if it contains no free variables. Note that closed unbalanced sums correspond to vectors.

\mathcal{R} -expressions in free variables $\{x_i\}$ may be interpreted as a non-standard syntax for the polynomial ring $\mathcal{R}[x_1, \dots, x_k] / \langle x_1^2 - x_1, \dots, x_k^2 - x_k \rangle$ which favours multiplication over addition in terms of computational efficiency. This allows our representation to coincide with balanced sums when possible, allowing a balanced representation to be used and manipulated normally, but providing an “escape hatch” in the form of ring sums. In [Section 4](#) we consider representations where the ring sum is dropped entirely.

In the balanced sum-over-paths it’s generally not obvious how a given matrix $A \in \mathcal{M}_{n \times m}(\mathcal{R})$ may be represented directly, hence universality is achieved via the construction of a known universal set of generators. By contrast, unbalanced sums allow the direct representation of A , as the amplitude function $\psi(\vec{x}, \vec{y}) = \langle \vec{y} | A | \vec{x} \rangle$ can be written directly as an \mathcal{R} -expression. We first define the notation $\vec{x} = \vec{y}$ as shorthand for the bitwise equality of \vec{x} and \vec{y} ,

$$\vec{x} = \vec{y} := \prod_i x_i \oplus \neg y_i.$$

Then we may write A as an unbalanced sum of the following form:

$$A |\vec{x}\rangle = \sum_{\vec{y}} \alpha_{00\dots 0}^{00\dots 0 = \vec{x}\vec{y}} \alpha_{00\dots 1}^{00\dots 1 = \vec{x}\vec{y}} \dots \alpha_{11\dots 1}^{11\dots 1 = \vec{x}\vec{y}} |\vec{y}\rangle$$

where $\vec{x}\vec{y}$ denotes the concatenation of \vec{x} and \vec{y} , and $\alpha_{\vec{x}\vec{y}}$ is equal to $\langle \vec{y} | A | \vec{x} \rangle$. Intuitively, for a given value of \vec{y} and \vec{x} , the (exponential-size) \mathcal{R} -expression above evaluates to $\alpha_{\vec{x}\vec{y}} = \langle \vec{y} | A | \vec{x} \rangle$, as the exponent of every of other $\alpha_{\vec{x}\vec{y}}$ evaluates to zero. We write out the product explicitly rather than as $\prod_z \alpha_z^{\vec{z} = \vec{x}\vec{y}}$ so as to avoid confusion with the use of Π as mathematical syntax and Σ as a syntactical element of path sums.

Proposition 3.3 (Universality). *Any linear operator $\mathcal{R}^{2^n} \rightarrow \mathcal{R}^{2^m}$ can be expressed as an unbalanced path sum over \mathcal{R} .*

Example 3.4. The $\Lambda(\text{H})$ gate can be expressed as the unbalanced sum

$$\Lambda(\text{H}) |x_1 x_2\rangle = \sum_y 0^{-x_1(x_2 \oplus y)} (1/\sqrt{2})^{x_1} (-1)^{x_1 x_2 y} |x_1 y\rangle.$$

Example 3.5. While the generalized H-boxes of the ZH-calculus [4] admit an indirect encoding in the balanced sum-over-paths via Euler angles [9], the ZH-calculus can be directly encoded in the unbalanced sum-over-paths as below.

$$\left[\begin{array}{c} n \\ \text{---} \\ \text{---} \\ m \end{array} \right] |\vec{x}\rangle = \sum_{\vec{y} \in \mathbb{Z}_2^n} \sum_{z \in \mathbb{Z}_2} 2^{-n} (-1)^{\sum_{i=1}^n y_i(x_i+z)} |zz \dots z\rangle \quad \left[\begin{array}{c} n \\ \text{---} \\ \text{---} \\ m \end{array} \right] |\vec{x}\rangle = \sum_{\vec{y} \in \mathbb{Z}_2^n} \alpha^{x_1 \dots x_n y_1 \dots y_m} |\vec{y}\rangle$$

As is customary, to simplify the notation and proofs we view a linear operator $A : \mathcal{R}^n \rightarrow \mathcal{R}^m$ as a vector $A \in \mathcal{R}^{nm}$ via the channel-state duality and define normal forms only on closed sums. Note that $\eta = \sum_y |yy\rangle$ and its adjoint $\varepsilon |xy\rangle = \frac{1}{2} \sum_z (-1)^{z(x \oplus y)}$, i.e. a ‘‘cup and cap,’’ are well-defined over any unital ring \mathcal{R} , hence we can move between the operator and vector view freely.

Definition 3.6 (Normal form). *A normal form is a closed, unbalanced sum of the following form:*

$$\sum_{\vec{x}} \alpha_{00 \dots 0}^{00 \dots 0 = \vec{x}} \alpha_{00 \dots 1}^{00 \dots 1 = \vec{x}} \dots \alpha_{11 \dots 1}^{11 \dots 1 = \vec{x}} |\vec{x}\rangle \quad (2)$$

Example 3.7. The $\Lambda(\text{H})$ gate has the normal form below, with 0’s suppressed:

$$\frac{1}{\sqrt{2}} \sum_{\vec{x}} \sqrt{2}^{0000 = \vec{x}} \sqrt{2}^{0101 = \vec{x}} 1^{1010 = \vec{x}} 1^{1011 = \vec{x}} 1^{1110 = \vec{x}} (-1)^{1111 = \vec{x}} |\vec{x}\rangle.$$

Note that x_1 and x_2 correspond to the first and second input bits, respectively, while x_3 and x_4 correspond to the first and second output.

We remark that normal forms are unique, which is a trivial consequence of the fact that they explicitly represent vectors in the computational basis by a sequence of 2^n amplitudes.

Proposition 3.8. *Let Ψ be a vector in \mathcal{R}^{2^n} . Then Ψ has a unique normal form.*

3.1 Equational theory of unbalanced sums over \mathcal{R}

Figure 3 gives an equational theory, denoted $\equiv_{\mathcal{R}}$, for unbalanced sums. We separate equations into three classes: equations on Boolean expressions, \mathcal{R} -expressions, and equations involving sums. The equational theory of Boolean expressions is simply the well-known equational theory of commutative Boolean rings and is only provided for completeness. The equational theory of \mathcal{R} -expressions includes the axioms of commutative, unital rings on the left, and equations specific to f -powers on the right. The two rules involving path sums are the usual (H) rule, and the new *sum* rule (S) which internalizes sums over variables as sums of \mathcal{R} -expressions.

To show completeness, we proceed by first defining a normal form for \mathcal{R} -expressions and showing that every \mathcal{R} -expression can be re-written in normal form. This forms the bulk of the proof, as the (S) rule can be used to force the evaluation of any internal variable by the \mathcal{R} -expression sub-language.

Definition 3.9 (\mathcal{R} -expression normal form). *An \mathcal{R} -expression over the variables $\{x_i\}$ is in normal form if it is of the form*

$$\alpha_{00 \dots 0}^{00 \dots 0 = \vec{x}} \alpha_{00 \dots 1}^{00 \dots 1 = \vec{x}} \dots \alpha_{11 \dots 1}^{11 \dots 1 = \vec{x}}.$$

$$\begin{array}{llll}
f \oplus 0 \equiv f & (f_1 \oplus f_2) \oplus f_3 \equiv f_1 \oplus (f_2 \oplus f_3) & f \cdot 1 \equiv f & (f_1 \cdot f_2) \cdot f_3 \equiv f_1 \cdot (f_2 \cdot f_3) \\
f \oplus f \equiv 0 & f_1 \oplus f_2 \equiv f_2 \oplus f_1 & f \cdot f \equiv f & f_1 \cdot f_2 \equiv f_2 \cdot f_1
\end{array}$$

$$f_1 \cdot (f_2 \oplus f_3) \equiv f_1 \cdot f_2 \oplus f_1 \cdot f_3$$

(a) Rules for Boolean expressions

$$\begin{array}{ll}
(r_1 + r_2) + r_3 \equiv r_1 + (r_2 + r_3) & r^0 \equiv 1 \equiv 1^f \\
r_1 + r_2 \equiv r_2 + r_1 & r^1 \equiv r \equiv r^f r^{-f} \\
r_1 + 0 \equiv r_1 & r^{f_1 \oplus f_2} \equiv r^{f_1} + r^{f_2} - (2r)^{f_1 \cdot f_2} \\
r - r \equiv 0 & r^{f_1 \cdot f_2} \equiv (r^{f_1})^{f_2} \\
(r_1 \cdot r_2) \cdot r_3 \equiv r_1 \cdot (r_2 \cdot r_3) & r_1^f r_2^f \equiv (r_1 r_2)^f \\
r_1 \cdot r_2 \equiv r_2 \cdot r_1 & r_1^f r_2^{-f} \equiv r_1^f + r_2^{-f} - 1 \\
r \cdot 1 \equiv r & r_1^f + r_2^f \equiv (r_1 + r_2)^f + 0^{-f} \\
r_1 \cdot (r_2 + r_3) \equiv r_1 \cdot r_2 + r_1 \cdot r_3 &
\end{array}$$

(b) Rules for \mathcal{R} -expressions.

$$\sum_{x,y} (-1)^{y(x \oplus f)} |\Psi(x)\rangle \equiv 2 |\Psi(f)\rangle \quad (\text{H})$$

$$\sum_y r(y) |\Psi\rangle \equiv (r(0) + r(1)) |\Psi\rangle \quad (\text{S})$$

(c) Rules for sum-over-paths expressions. In-scope variables are not free in any sub-expressions unless explicitly included in parentheses.

Figure 3: Equational theory $\equiv_{\mathcal{R}}$ for unbalanced sums over rings \mathcal{R} .

Proposition 3.10 (\mathcal{R} -expression normalization). *An \mathcal{R} -expression r can be brought into normal form over the variables $\{x_i\} \supseteq FV(r)$ using the equations of [Figure 3](#).*

A proof of [Proposition 3.10](#) is given in [Appendix A](#). We next turn our attention to normalization of expressions involving sums. Normalization proceeds by writing the closed sum as a sum over all basis vectors by equating outputs with fresh variables, then summing along each internal variable and normalizing the resulting \mathcal{R} -expression.

Theorem 3.11. *$\equiv_{\mathcal{R}}$ is complete for unbalanced sums over any commutative ring \mathcal{R} .*

Proof. Let $|\Psi\rangle = \sum_{\vec{x}} r |f_1 f_2 \cdots f_n\rangle$ be a closed, unbalanced sum. Then

$$\begin{aligned}
\sum_{\vec{x}} r |f_1 f_2 \cdots f_n\rangle &\equiv \sum_{\vec{x}} \sum_{\vec{y}} \sum_{\vec{z}} r (-1)^{y_1(z_1 \oplus f_1) + y_2(z_2 \oplus f_2) + \cdots + y_n(z_n \oplus f_n)} |z_1 z_2 \cdots z_n\rangle && \text{by (H)} \\
&\equiv \sum_{\vec{z}} \sum_{\vec{x}} \sum_{\vec{y}} r'(\vec{x}, \vec{y}) |\vec{z}\rangle \\
&\equiv \sum_{\vec{z}} (r'(00 \cdots 0) + r'(00 \cdots 1) + \cdots + r'(11 \cdots 1)) |\vec{z}\rangle && \text{by (S)} \\
&\equiv \sum_{\vec{z}} \alpha_{00 \cdots 0}^{00 \cdots 0 = \vec{z}} \alpha_{00 \cdots 1}^{00 \cdots 1 = \vec{z}} \cdots \alpha_{11 \cdots 1}^{11 \cdots 1 = \vec{z}} |\vec{z}\rangle && \text{by Proposition 3.10}
\end{aligned}$$

□

4 Weakening the sum rule

The equational theory developed in the preceding section is too strong for use in practice. Indeed, re-writing effectively amounts to explicit evaluation of the sum, e.g.,

$$\sum_x |\Psi(x)\rangle = |\Psi(0)\rangle + |\Psi(1)\rangle,$$

together with a set of rules for manipulating certain symbolic expressions over rings. This is made possible by the highly-expressive sub-language of \mathcal{R} -expressions, which allows for the summation of arbitrary \mathcal{R} -expressions and hence the super-powered sum rule. However, sums of \mathcal{R} expressions are difficult to re-write and generally require complete expansion of the expression to a normal form. In particular, with unrestricted use of the sum rule we are not likely to discover efficient proofs of equality.

To limit the power of the sum rule, in this section we define a fragment of the unbalanced sum-over-paths which eliminates sums of *symbolic* amplitudes, and give a complete equational theory over arbitrary fields.

Definition 4.1 (Multiplicative sum-over-paths). The multiplicative fragment of the unbalanced sum-over-paths over a field \mathcal{F} consists of unbalanced sums of the form

$$\begin{aligned} a &::= \alpha, \beta \in \mathcal{F} \mid a^f \mid a_1 a_2 \\ |\Psi\rangle &::= \sum_{\vec{y}} a |f_1 \cdots f_n\rangle. \end{aligned}$$

Boolean expressions f are defined as in the unbalanced sum-over-paths.

We call amplitude expressions of the form a \mathcal{F} -expressions. It can be readily observed that as normal forms live in the multiplicative fragment, the multiplicative fragment is again universal, and is equivalent to the full unbalanced sum-over-paths up to $\equiv_{\mathcal{R}}$.

Figure 4 defines an equational theory, denoted $\equiv_{\mathcal{F}}$, for the multiplicative fragment which is defined and sound when \mathcal{F} is a field. Note that the equation $a^{f_1 \oplus f_2} \equiv a^{f_1} a^{f_2} (a^{-2})^{f_1 \cdot f_2}$, which coincides with the lifting of $f_1 \oplus f_2$ to $f_1 + f_2 - 2f_1 \cdot f_2$, is not well defined in arbitrary rings. Otherwise, the equational theory coincides with the equational theory of \mathcal{R} -expressions with rules for sums of expressions removed. We omit the equational rules for Boolean expressions as they are the same as those of **Figure 3a**.

Proposition 4.2 (\mathcal{F} -expression normalization). *An \mathcal{F} -expression a can be brought into normal form over the variables $\{x_i\} \supseteq FV(a)$ using the equations of **Figure 4**.*

Proof. The proof follows identically to the proof of **Proposition 3.10**. The one different case of $a^{f_1 \oplus f_2}$ is handled similar to the corresponding case in **Proposition 3.10** by the fact that $a^{f_1 \oplus f_2} \equiv a^{f_1} a^{f_2} (a^{-2})^{f_1 \cdot f_2}$ where each of a^{f_1} , a^{f_2} , and $(a^{-2})^{f_1 \cdot f_2}$ can be normalized by the inductive hypothesis and the case of $a^{f_1 \cdot f_2}$. The case is then finished by the normalization of products. □

The **(A)** rule, which is a transliteration of the average rule of the ZH-calculus, replaces the **(S)** rule of $\equiv_{\mathcal{R}}$. Rather than summing “top-down” as in the normalization of the unbalanced sum-over-paths, the average rule allows amplitudes to only be summed “bottom-up,” i.e. by summing pure elements of \mathcal{F} .

$$\begin{array}{lll}
(a_1 \cdot a_2) \cdot a_3 \equiv a_1 \cdot (a_2 \cdot a_3) & a^0 \equiv 1 \equiv 1^f & a^{f_1 \oplus f_2} \equiv a^{f_1} a^{f_2} (a^{-2})^{f_1 \cdot f_2} \\
a_1 \cdot a_2 \equiv a_2 \cdot a_1 & a^1 \equiv a \equiv a^f a^{-f} & a^{f_1 \cdot f_2} \equiv (a^{f_1})^{f_2} \\
a \cdot 1 \equiv a & & a_1^f a_2^f \equiv (a_1 a_2)^f
\end{array}$$

(a) Rules for \mathcal{F} -expressions.

$$\sum_{x,y} (-1)^{y(x \oplus f)} |\Psi(x)\rangle \equiv 2 |\Psi(f)\rangle \quad (\text{H})$$

$$\sum_{y,z} a_1^x(y) a_2^{-x}(z) |\Psi(x)\rangle \equiv 2 \sum_y a_1^x(y) a_2^{-x}(y) |\Psi(x)\rangle \quad (\text{O})$$

$$\sum_y (\alpha^y \beta^{-y})^f |\Psi\rangle \equiv 2 \left(\frac{\alpha + \beta}{2} \right)^f |\Psi\rangle \quad (\text{A})$$

(b) Rules for unbalanced sums in the multiplicative fragment. In-scope variables are not free in sub-expressions unless explicitly listed in parentheses.

Figure 4: Equational theory $\equiv_{\mathcal{F}}$ of multiplicative sums over fields \mathcal{F} .

In particular, we may think about the normalization of a sum in (almost) normal form over one internal variable:

$$\sum_{\vec{x}} \sum_y \alpha_{00\dots 00}^{00\dots 00=\vec{x}y} \alpha_{00\dots 01}^{00\dots 01=\vec{x}y} \dots \alpha_{11\dots 11}^{11\dots 11=\vec{x}y} |\vec{x}\rangle.$$

Using the rules of $\equiv_{\mathcal{R}}$, we may factor out the y exponents and evaluate the sum over y top-down as

$$\begin{aligned}
\sum_{\vec{x}} \sum_y \alpha_{00\dots 00}^{00\dots 00=\vec{x}y} \alpha_{00\dots 01}^{00\dots 01=\vec{x}y} \dots \alpha_{11\dots 11}^{11\dots 11=\vec{x}y} |\vec{x}\rangle &\equiv_{\mathcal{R}} \sum_{\vec{x}} \sum_y (\alpha_{00\dots 00}^{00\dots 0=\vec{x}} \dots \alpha_{11\dots 10}^{11\dots 1=\vec{x}})^{-y} (\alpha_{00\dots 01}^{00\dots 0=\vec{x}} \dots \alpha_{11\dots 11}^{11\dots 1=\vec{x}})^y |\vec{x}\rangle \\
&\equiv_{\mathcal{R}} \sum_{\vec{x}} (\alpha_{00\dots 00}^{00\dots 0=\vec{x}} \dots \alpha_{11\dots 10}^{11\dots 1=\vec{x}} + \alpha_{00\dots 01}^{00\dots 0=\vec{x}} \dots \alpha_{11\dots 11}^{11\dots 1=\vec{x}}) |\vec{x}\rangle
\end{aligned}$$

Under the more restrictive rules of $\equiv_{\mathcal{F}}$, the y exponents must be brought inwards and the amplitudes summed in pairs:

$$\sum_{\vec{x}} \sum_y \alpha_{00\dots 00}^{00\dots 00=\vec{x}y} \alpha_{00\dots 01}^{00\dots 01=\vec{x}y} \dots \alpha_{11\dots 11}^{11\dots 11=\vec{x}y} |\vec{x}\rangle \equiv_{\mathcal{F}} \sum_{\vec{x}} \sum_y (\alpha_{00\dots 00}^{-y} \alpha_{00\dots 01}^y)^{00\dots 0=\vec{x}} \dots (\alpha_{11\dots 10}^{-y} \alpha_{11\dots 11}^y)^{11\dots 1=\vec{x}} |\vec{x}\rangle$$

In order to be able to apply the (A) rule to eliminate y with this factorization, a distinct variable is needed for each pair, which can be achieved with the (O) rule — a transliteration of the ortho rule of the ZH-calculus — since every pair's exponent varies in the polarity of at least one variable:

$$\begin{aligned}
&\sum_{\vec{x}} \sum_y (\alpha_{00\dots 00}^{-y} \alpha_{00\dots 01}^y)^{00\dots 0=\vec{x}} \dots (\alpha_{11\dots 10}^{-y} \alpha_{11\dots 11}^y)^{11\dots 1=\vec{x}} |\vec{x}\rangle \\
&\equiv_{\mathcal{F}} \sum_{\vec{x}} \sum_y \frac{1}{2^{2^n-1}} (\alpha_{00\dots 00}^{-y_1} \alpha_{00\dots 01}^{y_1})^{00\dots 0=\vec{x}} \dots (\alpha_{11\dots 10}^{-y_n} \alpha_{11\dots 11}^{y_n})^{11\dots 1=\vec{x}} |\vec{x}\rangle && \text{by (O)} \\
&\equiv_{\mathcal{F}} \sum_{\vec{x}} \frac{2^{2^n}}{2^{2^n-1}} \left(\frac{\alpha_{00\dots 00} + \alpha_{00\dots 01}}{2} \right)^{00\dots 0=\vec{x}} \dots \left(\frac{\alpha_{11\dots 10} + \alpha_{11\dots 11}}{2} \right)^{11\dots 1=\vec{x}} |\vec{x}\rangle && \text{by (A)} \\
&\equiv_{\mathcal{F}} \sum_{\vec{x}} (\alpha_{00\dots 00} + \alpha_{00\dots 01})^{00\dots 0=\vec{x}} \dots (\alpha_{11\dots 10} + \alpha_{11\dots 11})^{11\dots 1=\vec{x}} |\vec{x}\rangle
\end{aligned}$$

Note that in the final line above, each sum involves concrete values taken from \mathcal{F} and hence can be evaluated explicitly over \mathcal{F} .

Theorem 4.3. $\equiv_{\mathcal{F}}$ is complete for multiplicative sums over any field \mathcal{F} .

Proof. Let $|\Psi\rangle = \sum_{\vec{x}} a |f_1 f_2 \cdots f_n\rangle$ be a closed, multiplicative sum and note that $|\Psi\rangle \equiv_{\mathcal{F}} \sum_{\vec{x}} \sum_{\vec{y}} a' |\vec{x}\rangle$ by (H). Then by Proposition 4.2, a' can be normalized to give

$$|\Psi\rangle \equiv_{\mathcal{F}} \sum_{\vec{x}} \sum_{\vec{y}} \alpha_{00\dots 0}^{00\dots 0=\vec{xy}} \alpha_{00\dots 1}^{00\dots 1=\vec{xy}} \cdots \alpha_{11\dots 1}^{11\dots 1=\vec{xy}} |\vec{x}\rangle.$$

If \vec{y} is empty, then we're done. Otherwise, we can remove one variable at a time with the (O) and (A) rules as above until no internal variables are left. \square

5 Discussion

We have now given a concrete syntax for sum-over-paths expressions with unbalanced amplitudes. We gave equational theories for rings and fields, and showed that each is complete. While the equational theories we give here are simplistic and inefficient, our hope is that a complete equational theory will allow the development of effective, complete re-writing systems.

The equational theories we have developed — particularly $\equiv_{\mathcal{F}}$ — as well as our normal forms can be viewed as translations of the ZH-calculus with varying levels of freedom in the expression of *Boolean* data. In the ZH-calculus, propagation of Boolean expressions along wires is a *semantic* property, while in the sum-over-paths it is *syntactic*. This allows the (H) rule to take a more general form allowing the substitution of a variable with an expression, whereas in the ZH-calculus this logic is spread across several different rules.

As an exercise we could attempt to formulate a more ZH-like sum-over-paths by restricting the Boolean expression language further, e.g.,

$$\begin{aligned} f &::= 0 \mid 1 \mid x, y, z, \dots \mid f_1 \cdot f_2 \\ a &::= \alpha \in \mathcal{R} \mid \alpha^f \mid a_1 a_2 \\ |\Psi\rangle &::= \sum_{v \in V} a |x_1 \cdots x_n\rangle, \end{aligned}$$

and reformulate our equational theory for such a language. One natural formulation of the (H) rule is $\sum_{x,y} (-1)^{xy} (-1)^{yf} a(x) |\Psi\rangle \equiv 2a(f) |\Psi\rangle$, which corresponds to a slightly more general version of the HS1 rule of the ZH-calculus, which would be more accurately translated as $\sum_{x,y} (-1)^{xy} (-1)^{yf} \alpha^{xg} |\Psi\rangle \equiv 2\alpha^{fg} |\Psi\rangle$. With either formulation, a secondary rule is needed to cover propagation of Boolean sums, corresponding to expressions of the form

$$\sum_{x,y} (-1)^{xy} (-1)^{yf_1} (-1)^{yf_2} \cdots (-1)^{yf_k} a(x) |\Psi\rangle.$$

Semantically, this is equivalent to the expression $2a(f_1 \oplus f_2 \oplus \cdots \oplus f_k) |\Psi\rangle$, but since the language cannot express Boolean sums, we must distribute a over the sum in a single step. One natural option to avoid imposing restrictions on the underlying ring, as we did in this section, is to restrict a to the phase-free fragment. In particular,

$$\sum_{x,y} (-1)^{xy} (-1)^{yf_1} \cdots (-1)^{yf_k} (-1)^{xg} |\Psi\rangle \equiv 2(-1)^{f_1 g} \cdots (-1)^{f_k g} |\Psi\rangle,$$

which is equivalent up to (H) to the BA2 rule of the ZH-calculus.

On the one hand, it is unclear what the utility of such an exercise might be, beyond as a symbolic syntax for ZH diagrams. On the other hand, these investigations shed light on both the similarities and differences between the graphical and symbolic approach. Notably, the symbolic approach naturally allows highly expressive languages, and the use of existing theories developed within the framework of non-categorical algebra and symbolic computation. On the other hand, the use of an expressive symbolic language naturally makes it more challenging to apply *local* reasoning than in graphical theories. For these reasons we hypothesize that the sum-over-paths approach may be more amenable to *automated* reasoning, while the graphical approach may be more amenable to *interactive* reasoning. We leave it for future work to explore this premise further.

6 Acknowledgements

The author wishes to thank Louis Lemonnier and Aleks Kissinger for bringing the connection to the ZH-calculus and a path sum formulation of the ortho rule to their attention. The author also wishes to thank Julien Ross for many helpful discussions, as well as the anonymous reviewers for comments which have improved the presentation of the paper. This work was supported by Canada's AARMS and NSERC.

References

- [1] Matthew Amy (2018): *Towards Large-Scale Functional Verification of Universal Quantum Circuits*. In: *Proceedings of the 15th International Conference on Quantum Physics and Logic*, QPL'18, pp. 1–21, doi:10.4204/EPTCS.287.1. arXiv:<https://arxiv.org/abs/1805.06908>.
- [2] Matthew Amy, Owen Bennett-Gibbs & Neil J. Ross (2022): *Symbolic synthesis of Clifford circuits and beyond*. In: *Proceedings of the 19th International Conference on Quantum Physics and Logic*, QPL'22. arXiv:<https://arxiv.org/abs/2204.14205>.
- [3] Matthew Amy, Andrew N. Glauddell & Neil J. Ross (2020): *Number-Theoretic Characterizations of Some Restricted Clifford+T Circuits*. *Quantum* 4, p. 252, doi:10.22331/q-2020-04-06-252. arXiv:<https://arxiv.org/abs/1908.06676>.
- [4] Miriam Backens & Aleks Kissinger (2018): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity*. In: *Proceedings of the 15th International Conference on Quantum Physics and Logic*, QPL'18, pp. 23–42, doi:10.4204/eptcs.287.2. arXiv:<https://arxiv.org/abs/1805.02175>.
- [5] Miriam Backens, Aleks Kissinger, Hector Miller-Bakewell, John van de Wetering & Sal Wolffs (2023): *Completeness of the ZH-calculus*. *Compositionality* 5, p. 5, doi:10.32408/compositionality-5-5. arXiv:<https://arxiv.org/abs/2103.06610>.
- [6] Bob Coecke & Ross Duncan (2008): *Interacting Quantum Observables*. In: *Proceeds of the The 35th International Colloquium on Automata, Languages and Programming*, ICALP'08, pp. 298–310, doi:10.1007/978-3-540-70583-3_25.
- [7] Richard P. Feynman & Albert R. Hibbs (1965): *Quantum mechanics and path integrals*. International series in pure and applied physics, McGraw-Hill.
- [8] Louis Lemonnier, John van de Wetering & Aleks Kissinger (2020): *Hypergraph simplification: Linking the path-sum approach to the ZH-calculus*. In: *Proceedings of the 17th International Conference on Quantum Physics and Logic*, QPL'20, doi:10.4204/EPTCS.340.10. arXiv:<https://arxiv.org/abs/2003.13564>.
- [9] Renaud Vilmart (2021): *The Structure of Sum-Over-Paths, its Consequences, and Completeness for Clifford*. In: *Foundations of Software Science and Computation Structures, FoSSaCS'21* 12650, pp. 531–550, doi:10.1007/978-3-030-71995-1_27. arXiv:<https://arxiv.org/abs/2003.05678>.

- [10] Renaud Vilmart (2023): *Completeness of Sum-Over-Paths for Toffoli-Hadamard and the Dyadic Fragments of Quantum Computation*. In: *31st EACSL Annual Conference on Computer Science Logic, CSL'23* 252, pp. 36:1–36:17, doi:[10.4230/LIPIcs.CSL.2023.36](https://doi.org/10.4230/LIPIcs.CSL.2023.36). arXiv:<https://arxiv.org/abs/2205.02600>.
- [11] John van de Wetering & Sal Wolffs (2019): *Completeness of the Phase-free ZH-calculus*. arXiv:<https://arxiv.org/abs/1904.07545>.

A Proof of Proposition 3.10

Proposition A.1 (\mathcal{R} -expression normalization). *An \mathcal{R} -expression r can be brought into normal form over the variables $\{x_i\} \supseteq FV(r)$ using the equations of Figure 3.*

Proof. By induction on the structure of an \mathcal{R} -expression.

Case: α . By induction on the number of variables m . If $m = 0$ then α is already in normal form. For $m > 0$, let $\alpha \equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1}$ and observe that r can be brought into normal form involving one additional variable x_{m+1} :

$$\begin{aligned}
\alpha &\equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1} \\
&\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{\neg x_{m+1}} (\alpha_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{x_{m+1}} \\
&\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0})^{\neg x_{m+1}} \dots (\alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{\neg x_{m+1}} (\alpha_{00\dots 0}^{\vec{x}=00\dots 0})^{x_{m+1}} \dots (\alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{x_{m+1}} \\
&\equiv \alpha_{00\dots 0}^{(\vec{x}=00\dots 0)(x_{m+1}=0)} \dots \alpha_{11\dots 1}^{(\vec{x}=11\dots 1)(x_{m+1}=0)} \alpha_{00\dots 0}^{(\vec{x}=00\dots 0)(x_{m+1}=1)} \dots \alpha_{11\dots 1}^{(\vec{x}=11\dots 1)(x_{m+1}=1)} \\
&\equiv \alpha_{0\dots 00}^{\vec{x}x_{m+1}=00\dots 00} \dots \alpha_{11\dots 1}^{\vec{x}x_{m+1}=11\dots 10} \alpha_{00\dots 0}^{\vec{x}x_{m+1}=00\dots 01} \dots \alpha_{11\dots 1}^{\vec{x}x_{m+1}=11\dots 11}
\end{aligned}$$

Case: $r_1 r_2$. Let $r_1 \equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \alpha_{00\dots 1}^{\vec{x}=00\dots 1} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1}$ and $r_2 \equiv \beta_{00\dots 0}^{\vec{x}=00\dots 0} \beta_{00\dots 1}^{\vec{x}=00\dots 1} \dots \beta_{11\dots 1}^{\vec{x}=11\dots 1}$. Then

$$\begin{aligned}
r_1 r_2 &\equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1} \beta_{00\dots 0}^{\vec{x}=00\dots 0} \dots \beta_{11\dots 1}^{\vec{x}=11\dots 1} \\
&\equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \beta_{00\dots 0}^{\vec{x}=00\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1} \beta_{11\dots 1}^{\vec{x}=11\dots 1} \\
&\equiv (\alpha_{00\dots 0} \beta_{00\dots 0})^{\vec{x}=00\dots 0} \dots (\alpha_{11\dots 1} \beta_{11\dots 1})^{\vec{x}=11\dots 1}
\end{aligned}$$

Case: $r_1 + r_2$. Let r_1 be written in normal form as $r_1 \equiv \alpha_{00\dots 0}^{\vec{x}=00\dots 0} \alpha_{00\dots 1}^{\vec{x}=00\dots 1} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1}$ and factorize this as $s_1^{\neg x_m} s_2^{x_m}$. Likewise, let $r_2 \equiv \beta_{00\dots 0}^{\vec{x}=00\dots 0} \beta_{00\dots 1}^{\vec{x}=00\dots 1} \dots \beta_{11\dots 1}^{\vec{x}=11\dots 1} \equiv t_1^{\neg x_m} t_2^{x_m}$. Then we can observe:

$$\begin{aligned}
r_1 + r_2 &\equiv s_1^{\neg x_m} s_2^{x_m} + t_1^{\neg x_m} t_2^{x_m} \\
&\equiv s_1^{\neg x_m} + s_2^{x_m} - 1 + t_1^{\neg x_m} + t_2^{x_m} - 1 \\
&\equiv s_1^{\neg x_m} + s_2^{x_m} - 1 + t_1^{\neg x_m} + t_2^{x_m} - 1 \\
&\equiv (s_1 + t_1)^{\neg x_m} + 0^{x_m} + (s_2 + t_2)^{x_m} + 0^{\neg x_m} - 2 \\
&\equiv (s_1 + t_1)^{\neg x_m} + (s_2 + t_2)^{x_m} - 1 + 0^{\neg x_m} + 0^{x_m} - 1 \\
&\equiv (s_1 + t_1)^{\neg x_m} (s_2 + t_2)^{x_m} + 0^{\neg x_m} 0^{x_m} \\
&\equiv (s_1 + t_1)^{\neg x_m} (s_2 + t_2)^{x_m}
\end{aligned}$$

Now $s_1 + t_1$ and $s_2 + t_2$ are \mathcal{R} -expressions in $m - 1$ variables, so induction on the number of variables suffices to finish this case. Note that in the base case $m = 0$, $r_1 + r_2$ is an ordinary ring sum and hence can be evaluated in \mathcal{R} to the normal form α .

Case: r^f . We proceed by induction on the structure of f . If $f = 0$ then $r^f \equiv 1$ which can be brought into normal form by the previous case. If $f = 1$ then $r^f \equiv r$ which is already in normal form. If $f = x_i$ then

$$\begin{aligned} r^{x_i} &\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0} \alpha_{00\dots 1}^{\vec{x}=00\dots 1} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{x_i} \\ &\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0})^{x_i} (\alpha_{00\dots 1}^{\vec{x}=00\dots 1})^{x_i} \dots (\alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{x_i} \\ &\equiv 1^{\vec{x}=00\dots 0} \dots 1^{\vec{x}=01\dots 1} \alpha_{10\dots 0}^{\vec{x}=10\dots 0} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1} \end{aligned}$$

For the inductive cases, if $f = f_1 \cdot f_2$, then

$$\begin{aligned} r^{f_1 \cdot f_2} &\equiv (r^{f_1})^{f_2} \\ &\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0} \alpha_{00\dots 1}^{\vec{x}=00\dots 1} \dots \alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{f_2} \\ &\equiv (\alpha_{00\dots 0}^{\vec{x}=00\dots 0})^{f_2} (\alpha_{00\dots 1}^{\vec{x}=00\dots 1})^{f_2} \dots (\alpha_{11\dots 1}^{\vec{x}=11\dots 1})^{f_2} \\ &\equiv r_{00\dots 0} r_{00\dots 1} \dots r_{11\dots 1} \end{aligned}$$

where each $r_{\vec{x}}$ is in normal form and the $r_1 r_2$ case suffices to finish this case.

Finally, if $f \equiv f_1 \oplus f_2$, then $r^{f_1 \oplus f_2} \equiv r^{f_1} + r^{f_2} - (2r)^{f_1 \cdot f_2}$ where each term can be brought into normal form by the inductive hypothesis and the previous case. The $r_1 + r_2$ case then completes this case. \square

The Qudit ZH-Calculus: Generalised Toffoli+Hadamard and Universality

Patrick Roy*
University of Oxford
patrick.roy@udo.edu

John van de Wetering
University of Amsterdam
john@vdwetering.name

Lia Yeh
University of Oxford
Quantinuum, 17 Beaumont Street
Oxford OX1 2NA, United Kingdom
lia.yeh@cs.ox.ac.uk

We introduce the qudit ZH-calculus and show how to generalise the phase-free qubit rules to qudits. We prove that for prime dimensions d , the phase-free qudit ZH-calculus is universal for matrices over the ring $\mathbb{Z}[e^{2\pi i/d}]$. For qubits, there is a strong connection between phase-free ZH-diagrams and Toffoli+Hadamard circuits, a computationally universal fragment of quantum circuits. We generalise this connection to qudits, by finding that the two-qudit $|0\rangle$ -controlled X gate can be used to construct all classical reversible qudit logic circuits in any odd qudit dimension, which for qubits requires the three-qubit Toffoli gate. We prove that our construction is asymptotically optimal up to a logarithmic term. Twenty years after the celebrated result by Shi proving universality of Toffoli+Hadamard for qubits, we prove that circuits of $|0\rangle$ -controlled X and Hadamard gates are approximately universal for qudit quantum computing for any odd prime d , and moreover that phase-free ZH-diagrams correspond precisely to such circuits allowing postselections.

1 Introduction

For qubits there are essentially three different graphical calculi: ZX, ZW and ZH [9]. Each of these is suitable for reasoning about different types of structures and quantum gates. The ZX-calculus [10, 11] is the most well-studied of these, and can naturally reason about the Clifford+Phases gate set (containing CNOT, Hadamard, S as well as arbitrary Z phase gates) and the useful primitives of phase gadgets and Pauli gadgets [12, 44]. Its *phase-free* fragment, where the spiders cannot be labelled by a non-trivial phase, corresponds to CNOT circuits (together with ancillae and postselection) and can alternatively be interpreted into a category of linear relations [24]. The ZW-calculus [22, 39] instead can reason about photonic and fermionic computations [23]. The W-spider helps to easily represent sums of linear maps [25, 31, 42]. Its phase-free fragment is universal and complete for matrices over \mathbb{Z} , and here again the W-spider is used to sum up numbers.

The calculus we will be interested in here is the ZH-calculus [2, 3]. Its H-box generator allows for easy representation of gates involving multilinear logic, like the Toffoli or other many-controlled gates. It can represent hyper-graph states [26], the path-sum formalism [27, 35, 36], quantum binary decision diagrams [34] and more [15, 16]. Its phase-free fragment represents the Toffoli+Hadamard gate set and is universal for matrices over \mathbb{Z} [3]. The H-box here allows for representing the AND operation $|x\rangle \otimes |y\rangle \mapsto |x \wedge y\rangle$.

The last few years have seen a push towards generalising graphical calculi to work for higher-dimensional qudits. For ZX there is now work on qutrits [33, 37, 41], the prime-dimensional qudit stabiliser fragment [6, 30], and the universal algebraic qudit ZX-calculus [38, 40]. For ZW there are several

*Work completed before the author joined Amazon UK, Inc.

different proposals for qudit generalisations [29, 39]. Missing from these proposals is a generalisation for the ZH-calculus.

In this paper we present for the first time a qudit generalisation of the ZH-calculus. We base this translation on extending the representation of Boolean logic in the qubit ZH generators of [3] to arithmetic over \mathbb{Z}_d . Then the Z- and X-spiders represent respectively the copy $x \mapsto (x, x)$ and addition/negation $(x, y) \mapsto -_d(x +_d y)$, while the H-box represents (up to some Hadamards) the multiplication $(x, y) \mapsto x \cdot_d y$, where the subscript d denotes an operation modulo d . This correspondence makes it easy to represent qudit generalisations of Toffoli-like gates.

In order to motivate this connection, we will first study the qudit generalisation of the Toffoli+Hadamard gate set, which for qubits is known to be computationally universal for quantum circuits [32]. First, we show that whereas the Toffoli suffices to construct all classical reversible qubit logic circuits, for odd-dimensional qudits we can do the same with the $|0\rangle$ -controlled X gate. We find that our construction for these qudit classical reversible circuits from the $|0\rangle$ -controlled X gate is asymptotically optimal up to a logarithmic factor. Second, we show that the gate set consisting of the $|0\rangle$ -controlled X and Hadamard¹ gates is approximately universal for quantum computing in all odd prime dimensions. Third, we find that phase-free qudit ZH-diagrams represent precisely postselected circuits over this Hadamard+ $|0\rangle$ -controlled X gate set.

A considerable part of the paper is devoted to proving that the phase-free ZH-calculus for prime-dimensional qudits is universal for matrices over $\mathbb{Z}[\omega]$ where $\omega = e^{2\pi i/d}$ is a d th root of unity. While proving universality for qubit ZH is straightforward, the qudit case brings several difficulties, since the structure of the matrix of the H-box is a lot more complicated. Our proof involves an encoding of propositional formulae over \mathbb{Z}_d into polynomials and a construction of Pascal's triangle into a matrix.

In Section 2 we present our results regarding classical reversible dit logic and the $|0\rangle$ -controlled X gate. Then in Section 3 we introduce the phase-free qudit ZH-calculus and show its connection to the previously introduced gates. In Section 4 we extend the calculus to allow labels over arbitrary rings and prove its universality over this ring. Then in Section 5 we tackle the harder problem of proving universality of the phase-free ZH-calculus.

2 The qudit Toffoli+Hadamard gate set

In this paper, we let d denote the dimension of our qudits, so that a single wire in a (circuit) diagram corresponds to \mathbb{C}^d . Note that many of our results only work if d is an odd prime. We let $\omega := e^{2\pi i/d}$ denote a d th root of unity. Then the qudit Paulis correspond to $Z|a\rangle = \omega^a|a\rangle$ and $X|a\rangle = |a +_d 1\rangle$, where we use subscripts on operators like $+_d$ to denote operations modulo d . The controlled X gate (CX) then becomes $|x, y\rangle \mapsto |x, x +_d y\rangle$. The qudit Hadamard acts as $H|x\rangle = \frac{1}{\sqrt{d}} \sum_y \omega^{x \cdot y} |y\rangle$. For qubits, we can write the action of the Toffoli as $|x, y, z\rangle \mapsto |x, y, (x \cdot_2 y) +_2 z\rangle$. This definition extends straightforwardly to the qudit setting, where we just take the multiplication and addition to be modulo d instead of modulo 2. When allowing *zeroed* ancillae, i.e. qubits prepared in the $|0\rangle$ state, the Toffoli together with the X gate (which acts as the NOT gate) suffice to construct an arbitrary classical reversible logic circuit. It turns out however that for certain qudit dimensions, just a two-qudit gate suffices to achieve the analogous result.

¹Technically in mathematics a Hadamard matrix is a ± 1 matrix of maximum possible determinant, named after Hadamard's 1893 article on the matter [21]. However, we follow the convention of other qudit graphical calculi to refer to the d -dimensional Discrete Fourier Transform as the Hadamard [6].

We define the $|0\rangle$ -controlled X gate as acting on the computational basis as follows:

$$|c, t\rangle \mapsto \begin{cases} |c, t+d-1\rangle, & \text{if } c = 0 \\ |c, t\rangle, & \text{else} \end{cases} \quad (1)$$

i.e. by applying an X gate to the target iff the control is $|0\rangle$.

Note that the $|0\rangle$ -controlled X gate is not Clifford for any prime qudit dimensions except for the qubit case (for which it is a CNOT gate conjugated by NOTs on the control).

Theorem 2.1. For any odd qudit dimension d , any d -ary classical reversible function $f : \mathbb{Z}_d^n \rightarrow \mathbb{Z}_d^n$ on n dits can be constructed by a circuit of $O(d^n n)$ many $|0\rangle$ -controlled X gates and $O(n)$ ancillae prepared in the $|0\rangle$ state.

Proposition 2.2. For any qudit dimension d , there exist d -ary classical reversible functions $f : \mathbb{Z}_d^n \rightarrow \mathbb{Z}_d^n$ that require at least $O(nd^n / \log n)$ single-qudit and two-qudit gates to construct, even when allowed $\Omega(n)$ ancillae.

We present the proofs of Theorem 2.1 and Proposition 2.2 in the appendix.

Interestingly, we only need a two-qudit gate—the $|0\rangle$ -controlled X gate—to construct any d -ary classical reversible gate (i.e. bijective maps of the form $f : \mathbb{Z}_d^n \rightarrow \mathbb{Z}_d^n$) with the help of $|0\rangle$ ancillae. Hence, the $|0\rangle$ -controlled X gate is universal for all classical reversible logic—generalising to all odd d what the three-qubit Toffoli gate does for $d = 2$. Hence, it makes sense to consider the generalization of the qubit Toffoli+H gate set to be the qudit gate set containing $|0\rangle$ -controlled X and Hadamard, which by Theorem 2.1 generates all possible qudit generalized Toffoli gates (since they are all classically reversible).

For qubits, adding the Hadamard gate to all the classical reversible gates (which is generated by the Toffoli gate and zeroed ancillae) suffices for approximately universal quantum computation [32]. By combining Theorems 2.1 and 2.3 we find that this is in fact true in any prime qudit dimension.

Theorem 2.3. The $|0\rangle$ -controlled X gate and the H gate form an approximately universal gate set for qudits of any odd prime dimension. In other words, permitting the help of ancillae, this gate set can deterministically approximate any qudit computation up to arbitrarily small error.

Proof. The proof below suffices for the case where the qudit dimension d is a prime $d > 3$. The proof for the $d = 3$ case consists of constructing all the Cliffords as follows, and the metaplectic gate (a single-qudit non-Clifford gate) which we construct in Appendix B similarly to our construction in Ref. [19, Section 3].

Define the single-qudit gates $Q[i]$ by $Q[i]|j\rangle = \omega^{\delta_{ij}}|j\rangle$ where $\delta_{ij} = 1$ iff $i = j$. In [43] it is shown that CX, H, and the $Q[i]$ gates are universal for quantum computing for prime $d > 3$; for $d = 3$ this generates the Clifford group. It hence suffices to show that our gate set generates these gates. Clearly, inputting a zeroed ancilla to the control of the $|0\rangle$ -controlled X gate yields the X gate. From here, the CX gate is easy to build from X and $|0\rangle$ -controlled X gates. We can also exactly synthesize the $Q[0]$ gate deterministically (up to an irrelevant global phase) with just $|0\rangle$ -controlled X gates, H gates and a zeroed ancilla:

$$\begin{array}{c} \boxed{Q[0]} \\ \hline \end{array} = \begin{array}{c} \textcircled{0} \\ |1\rangle \text{---} \boxed{Z} \text{---} \langle 1| \end{array} = \begin{array}{c} \textcircled{0} \\ |0\rangle \text{---} \boxed{X} \text{---} \boxed{H^\dagger} \text{---} \boxed{X} \text{---} \boxed{H} \text{---} \boxed{X^\dagger} \text{---} \langle 0| \end{array} \quad (2)$$

Conjugating by X gates then yields all $Q[i]$ gates. □

Remark 2.4. Theorem 2.1 and Proposition 2.2 build upon previous work of some of the authors [46], which showed for qutrits how to explicitly construct any ternary classical reversible gate using $O(3^n)$

$|0\rangle^{\otimes n}$ -controlled X_{01} gates² each with gate count polynomial in n , and that there exist ternary classical reversible gates requiring at least $O(n3^n/\log n)$ gates to construct. In this work, we generalise these results to any odd qudit dimension d and we additionally find a construction of the $|0\rangle^{\otimes n}$ -controlled X_{01} gate using $O(n)$ gates. Combining these results gives us the $O(nd^n)$ gate count construction of d -ary classical reversible gates which is hence near asymptotically optimal in gate count up to a $\log n$ factor.

Remark 2.5. A recent preprint [47] appearing after submission of this paper independently discovered a version of Lemmas A.5 and A.8 for any odd qudit dimension. They additionally provide a separate $O(n)$ gate count $|0\rangle^{\otimes n}$ -controlled X_{01} gate construction applicable to any even qudit dimension. By generalisation of Ref. [46], they independently derived our Proposition 2.2 and a version of our Theorem 2.1 which uses more types of gates than just the $|0\rangle$ -controlled X , but which does work for all qudit dimensions.

3 The qudit ZH-Calculus

Now let us introduce the qudit ZH-calculus, which allows for graphical reasoning about qudit Toffoli-like gates. Diagrams will flow from inputs at the bottom, to outputs on the top (but because our generators will be flexsymmetric [7, 8] the orientation of diagrams in this paper will not matter much).

As is the case for the qubit ZH-calculus, the qudit ZH-calculus will consist of string diagrams built out of two types of generators: Z-spiders and H-boxes. We define these as follows:

$$\underbrace{\begin{array}{c} n \\ \cdots \\ \text{---} \\ \text{---} \\ \text{---} \\ \cdots \\ m \end{array}} := \sum_{i=0}^{d-1} |i\rangle^{\otimes n} \langle i|^{\otimes m}, \quad \underbrace{\begin{array}{c} n \\ \cdots \\ \text{---} \\ \text{---} \\ \text{---} \\ \cdots \\ m \end{array}} := \frac{1}{\sqrt{d}} \sum_{i_1, \dots, i_m, j_1, \dots, j_n \in \mathbb{Z}_d} \omega^{i_1 \dots i_m \cdot j_1 \dots j_n} |j_1 \dots j_n\rangle \langle i_1 \dots i_m|.$$

This matches the qubit-ZH definitions of [2], except that now the sums go from 0 to $d - 1$ instead of from 0 to 1, and we use the d th root of unity $\omega = e^{2\pi i/d}$ instead of -1 . Additionally, we have included a normalization factor of $1/\sqrt{d}$ in the definition of the H-box that will prevent some tedious constants from appearing everywhere [5, Ap. E]. As a consequence of this choice of normalisation, the 1-input, 1-output phase-free H-box corresponds exactly to the qudit Hadamard $|x\rangle \mapsto \frac{1}{\sqrt{d}} \sum_y \omega^{x \cdot y} |y\rangle$. Note that while the matrix of the qubit H-box consists of just 1's, with a single entry equal to -1 , for qudits the matrix has a more complicated structure, with different powers of ω appearing throughout the matrix. In the next section we will also introduce labelled H-boxes, so we will sometimes refer to diagrams containing just the above generators as *phase-free* ZH-diagrams, following [3].

Apart from these generators we have the standard structural generators—identity, swap, cup and cap—needed to make a compact-closed PROP. Note that the qudit Z-spider and H-box satisfy the same symmetries as their qubit counterparts, meaning we get a flexsymmetric PROP [7, 8]:



Remark 3.1. Note that the actual choice of d th root of unity $\omega = e^{2\pi i/d}$ is not important. We can choose any primitive d th root of unity (i.e. a complex number ω satisfying $\omega^d = 1$ while $\omega^k \neq 1$ for any $0 \leq k < d$), and the rest of our results will also go through.

² X_{01} maps $|0\rangle$ and $|1\rangle$ to each other and is identity on all other basis states.

There are a couple of useful derived generators we will need:

(3)

The first of these is the well-known X -spider. The second realizes the Pauli X gate, e.g. the map $X|i\rangle = |i+d\rangle$. The last two generators represent the scalars \sqrt{d} and $1/\sqrt{d}$ respectively.

The (derived) generators of the qubit ZH-calculus can be motivated by a correspondence to Boolean logic [3, Eq. 5]. Similarly, our generators turn out to correspond with arithmetic operations over \mathbb{Z}_d :

(4)

Note that here for multiplication we have a sequence of three Hadamard instead of just the one in the qubit version. This is because for qudits $H^4 = \text{id}$, but not $H^2 = \text{id}$. Instead we have $H^2|i\rangle = |-_d i\rangle$. This map is sometimes called the *antipode* or *dualiser* [9], and we will use it throughout the diagrams in this paper. It turns out to also be equal to a single-input, single-output X -spider.

This interpretation gives a straightforward way to represent the Toffoli and the $|0\rangle$ -controlled X gate (writing our diagrams here from left-to-right to match circuit notation):

(5)

The correctness of the Toffoli construction follows easily from the interpretation given in Eq. (4). For the other, note that in the first step we use the trick that a gate controlled on some value, followed by its adjoint, is the same thing as controlling the adjoint on all the other values. Then the correctness of the ZH-diagram follows from Fermat's little theorem: for all $x \in \mathbb{Z}_d$ for d prime, $x^{d-1} = 0$ if $x = 0$ and $x^{d-1} = 1$ otherwise. The full diagram hence adds 1 if the control is not 0.

Many of the rules of the qubit ZH-calculus generalise to qudits; see Figure 1. For their soundness we refer to Appendix C.

The Z -spider fusion rule generalises as expected, but the H -box fusion rule generalizes into something that allows *contraction* of odd-length sequences of H -boxes interspersed by Hadamards. For the bialgebra rules, the Z/X version generalises up to global scalars, while the Z/H bialgebra needs some additional Hadamards which would cancel in the qubit case (furthermore for (ba1), if $(n-1)(m-1) < 0$, introduce $\blacktriangle^{-(n-1)(m-1)}$ to the LHS instead). Lastly, we have the generalization of the *identity* and *multiply* rules. We rename the latter *cyclic* (cy) because what it really captures is the cyclic structure of the group \mathbb{Z}_d .

Note how the above ruleset neither contains a rule stating that $H^4 = \text{id}$, nor an inverted color change rule. That is because we can derive them from the rules presented above:

(h4)

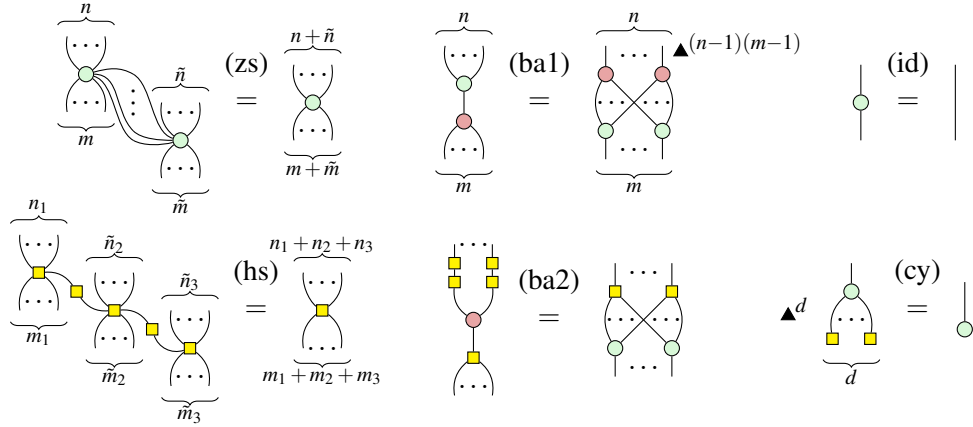
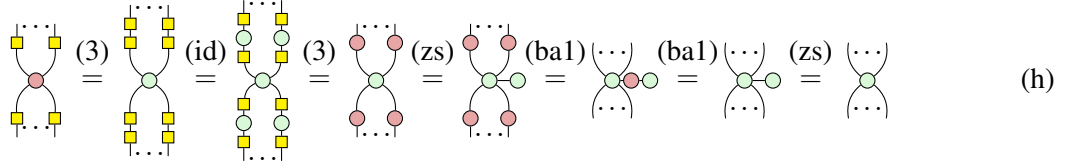


Figure 1: Basic rules of the phase-free qudit ZH-calculus. Some additional (derived) rules are presented in Appendices D and E. The rules hold for all n and m . Here d is the dimension of the qudit.



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 the notation of [33, Fig. 1].

Since these derivations hold for arbitrary dimension d , they particularly hold for $d = 2$. This means that due to the qubit H-box fusion rule, (h4) actually implies the self-inverseness of Hadamard gates, making the (hh) rules of Backens et al’s ruleset redundant [3, Tab. 1].

In Appendix D, we also present a generalisation of the ortho rule from the phase-free qubit ZH-calculus [3]. Hence, we have a prime-dimensional qudit generalisation of all the phase-free qubit ZH-calculus rewrite rules [3]. While those rules are complete for the qubit phase-free calculus, it is not clear whether this continues to hold for qudits. We leave this question for future work, for instance building upon the recent completeness for all qudit dimensions in the ZXW-calculus [29].

3.1 Translating ZH-diagrams to ZX-diagrams

The qudit ZX-calculus is universal, and hence can represent any linear map between qudits [41]. So in particular, there must be some way to interpret ZH-diagrams into ZX-diagrams. As the only generator of ZH-diagrams that is different from the ZX-calculus is the H-box, this is the only one we will have to translate. In fact, we only need to translate the three-legged and one-legged H-box, as the two-legged H-box is just the Hadamard gate. We can then obtain diagrams for higher-arity H-boxes by unfusing them into three-legged H-boxes. However, we will also introduce a direct construction for n -legged H-boxes, which arises from the asymptotically efficient circuit constructions for any multiple-controlled prime-dimensional qudit Toffoli gate presented in the Appendix.

First, note that there is a close correspondence between an H-box and the qudit CCZ gate, which acts

like $|x, y, z\rangle \mapsto \omega^{x \cdot y \cdot z} |x, y, z\rangle$:

(6)

Hence, in particular the three-legged H-box is equal to one copy of the qudit CCZ gate acting on $|+++ \rangle$:

(7)

Since a CCZ gate is just the Toffoli from Eq. (5) with the target qudit conjugated by Hadamards, to construct an H-box in the ZX-calculus it then suffices to show how to construct the qudit Toffoli in the ZX-calculus. But by Theorem 2.1 we can construct the Toffoli from the $|0\rangle$ -controlled X gate, so that it remains to show how this gate is constructed as a ZX-diagram.

We will write phases on Z-spiders in the ZX-calculus, as vectors $\vec{\alpha}$ of length $d-1$:

$$\textcircled{\vec{\alpha}} = |0 \dots 0\rangle\langle 0 \dots 0| + e^{i\alpha_1} |1 \dots 1\rangle\langle 1 \dots 1| + \dots + e^{i\alpha_{d-1}} |(d-1) \dots (d-1)\rangle\langle (d-1) \dots (d-1)| \quad (8)$$

Lemma 3.2 ([45]). The prime-dimensional qudit $|0\rangle$ -controlled X gate can be decomposed into the Clifford+Phases gate set (decomposing H as phase gates [40, Remark 2.3]), and written as a ZX-diagram:

(9)

where $\vec{p} = \left(\omega^{-\frac{d-1}{2}}, \omega^{-\frac{d-1}{2}}, \dots, \omega^{-\frac{d-1}{2}} \right)$ and $\vec{r} = \left(\omega^{\frac{1}{d}}, \omega^{\frac{2}{d}}, \dots, \omega^{\frac{d-1}{d}} \right)$ represents the d th root of Z gate from Ref. [45].

Theorem 3.3. Any prime-dimensional qudit ZH-diagram composed of m Z spiders and n H-boxes each with no more than g legs, can be written as a composition of those m Z spiders, and $O(ng)$ of either $|0\rangle$ -controlled X, Hadamard or $|0\rangle$.

Proof. Up to cups and caps, an H-box with two legs is the Hadamard gate, while an H-box with one leg is $Z|+\rangle$: $\textcircled{\vec{Z}} = \textcircled{\vec{Z}} = \textcircled{\vec{Z}}$ where $\vec{Z} = \left(\omega, \omega^2, \dots, \omega^{d-1} \right)$ indicates the Z gate.

Eq (7) shows how to relate the three-legged H-box to the CCZ gate. We can then invoke Theorem 2.1 to build the Toffoli gate from $|0\rangle$ -controlled X gates, which is related to the CCZ gate by conjugating the target by Hadamards. Eq. (9) shows how to construct this gate in the ZX-calculus. Any H-box with $n > 3$ legs can then be built from unfusing to one- to three-legged H-boxes by applying rule (HS). \square

Note that phase-free Z-spiders are just GHZ states $|0 \dots 0\rangle + |1 \dots 1\rangle + \dots + |(d-1) \dots (d-1)\rangle$, up to cups and caps. Hence, using a typical decomposition of any size qudit GHZ state into a CX circuit on $|+0 \dots 0\rangle$,

(10)

we can further decompose the Z-spiders in Theorem 3.3 into $\{|0\rangle\text{-controlled X, H}, |0\rangle\}$ as $|+\rangle = H|0\rangle$. This then gives us a way to write any phase-free ZH diagram as a $\{|0\rangle\text{-controlled X, H}\}$ circuit where ancillae and postselections on $|0\rangle$ and $\langle 0|$ are allowed. By Eq. (5), we can also write any circuit composed of $\{|0\rangle\text{-controlled X, H}, |0\rangle, \langle 0|\}$ as a phase-free ZH diagram. Therefore, prime-dimensional phase-free ZH-diagrams correspond to $\{|0\rangle\text{-controlled X, H}\}$ circuits with ancillae and postselections. This thus generalizes the same correspondence which holds in the qubit case, between the qubit phase-free ZH-calculus and the Toffoli+Hadamard gate set.

is true iff $\lambda_{|x_1\dots x_n\rangle, |y_1\dots y_n\rangle} = 1$:

$$\varphi_L(x_1, \dots, x_n, y_1, \dots, y_m) = \bigvee_{\substack{i_1, \dots, i_n, j_1, \dots, j_m \\ \in \{0, \dots, d-1\} \\ \lambda_{i_1 \dots i_n, j_1 \dots j_m} = 1}} \bigwedge_{k=1}^n (x_k = i_k) \wedge \bigwedge_{\ell=1}^m (y_\ell = j_\ell). \quad (13)$$

Logical formulae unfortunately do not correspond to something we can easily directly express in ZH_R . However, we can translate these formulae into polynomials, which we *can* represent in ZH_R .

Proposition 4.1. If d is prime, then for every propositional formula φ over $(\mathbb{Z}_d, -, +, \cdot, =)$ in n free variables there exists a polynomial $p_\varphi \in (\mathbb{Z}_d)[X_1, \dots, X_n]$ such that $p_\varphi(x_1, \dots, x_n) = 0 \iff \varphi(x_1, \dots, x_n)$.

Proof. Let φ be a formula over $(\{1, \dots, d\}, -, +, \cdot, =)$ 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 and disjunction³.

- 1) When $\varphi = (p_1(x_1, \dots, x_n) = p_2(x_1, \dots, x_n))$ for $p_1, p_2 \in (\mathbb{Z}_d)[X_1, \dots, X_n]$, set $p_\varphi = p_1 - p_2$.
- 2) When $\varphi = \neg\varphi'$, set $p_\varphi = 1 - (p_{\varphi'})^{d-1}$.
- 3) When $\varphi = \varphi_1 \vee \varphi_2$, set $p_\varphi = p_{\varphi_1} \cdot p_{\varphi_2}$.

The only non-obvious part of the construction is the construction for negation. This step follows from the fact that for d prime, exponentiating with $d-1$ in \mathbb{Z}_d 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. \square

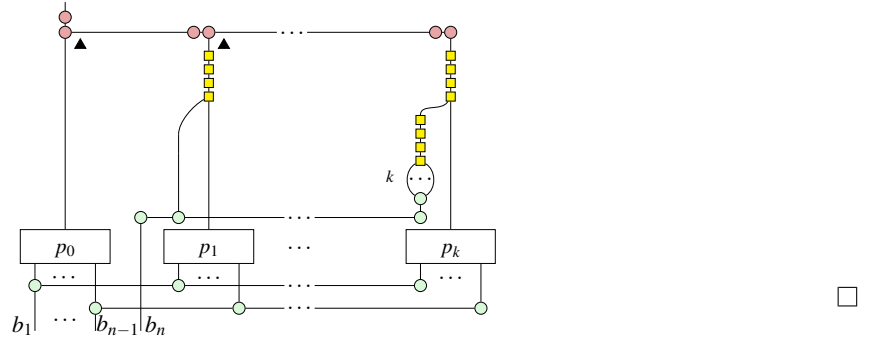
Lemma 4.2. Assume d is prime. Given a polynomial $p \in (\mathbb{Z}_d)[X_1, \dots, X_n]$ in n variables, we can construct an n -input 0-output ZH_R -diagram that evaluates to 1 on states $|b_1\dots b_n\rangle$ such that $p(b_1, \dots, b_n) = 0$, and to r on all other states.

Proof. First suppose that we had a diagram implementing the map $|b_1\dots b_n\rangle \mapsto |p(b_1, \dots, b_n)\rangle$. For d prime, the map $x \mapsto x^{d-1}$ in \mathbb{Z}_d maps 0 to 0, and everything else to 1. By (12), we know how to realize this operation as a ZH-diagram. Apply this operation to the output of the diagram implementing the polynomial, and postselect with the effect $H(r)^T = (1, r, r^2, \dots, r^{d-1})$. This gives the desired map. So let's see how to implement the map $|b_1\dots b_n\rangle \mapsto |p(b_1, \dots, b_n)\rangle$.

We do this by induction on the number of variables n . If $n = 0$, then $p \in \mathbb{Z}_d$ is a constant, which we know how to realize using (12). Now suppose we know how to construct diagrams for polynomials with $n-1$ variables. By definition of polynomial rings we can abuse notation slightly to write $p \in (\mathbb{Z}_d[X_1, \dots, X_{n-1}])[X_n]$, e.g. $p = \sum_{i=0}^k p_i X_n^i$ for $p_0, \dots, p_k \in \mathbb{Z}_d[X_1, \dots, X_{n-1}]$. By induction, we have ZH-diagrams realizing p_0, \dots, p_k , which we denote by boxes labeled “ p_0 ” through “ p_k ”. Then a diagram for

³We do not need to deal with conjunction, since \neg and \vee are functionally complete.

the desired map can be constructed as follows (using the correspondence to algebraic operations of (4)):



In light of (13), this means that using a polynomial $p \in (\mathbb{Z}_d)[X_1, \dots, X_m, Y_1, \dots, Y_n]$ such that $p(x_1, \dots, x_m, y_1, \dots, y_n) = 0 \iff \phi(x_1, \dots, x_m, y_1, \dots, y_n)$, we can use Lemma 4.2 and map-state-duality to construct arbitrary $r, 1$ -pseudobinary linear maps as ZH_R -diagrams.

Corollary 4.3. For prime d , every $r, 1$ -pseudobinary linear map $L : (\mathbb{C}^d)^{\otimes n} \rightarrow (\mathbb{C}^d)^{\otimes m}$ has a qudit ZH_R -diagram realizing L .

Proof. Use (13) to construct a formula $\phi(\vec{x}, \vec{y})$ that is true when $\langle \vec{y} | L | \vec{x} \rangle = 1$ and r otherwise. Then use Proposition 4.1 to transform ϕ into a polynomial p that is 0 when ϕ is true, and finally use Lemma 4.2 to construct a diagram with $n + m$ inputs that evaluates to 1 when you input $|\vec{x}\rangle \otimes |\vec{y}\rangle$ with $p(\vec{x}, \vec{y}) = 0$ and to r on other inputs. Bending the last m wires up to be outputs gives a diagram that is exactly equal to L . \square

We give a worked out example of this entire procedure in Appendix F.

Theorem 4.4. Let $R \supset \mathbb{Z}[\omega, \frac{1}{\sqrt{d}}]$ be a commutative ring. Then ZH_R is universal for matrices over R .

Proof. By Proposition 4.3 we can construct ZH_R -diagrams for arbitrary $r, 1$ -pseudobinary matrices for $r \in R$. By taking Schur products of these matrices, any matrix over R can be realised. \square

5 Universality of the phase-free ZH-calculus

We now set our sights on establishing the universality of the phase-free ZH-calculus, where we are only allowed ω -labelled (i.e. phase-free) H-boxes, for matrices over the ring $\mathbb{Z}[\omega]$. We will use the structure of the previous proof, reducing the problem to the ability to construct diagrams for $r, 1$ -pseudobinary matrices, where now $r \in R = \mathbb{Z}[\omega]$. The only obstacle to using this approach is that in the proof of Lemma 4.2 we require a postselection to the state $H(r)$, which we don't a priori have access to. To prove universality of the phase-free ZH-calculus we hence need to show that we can construct diagrams for states of the form $H(r) = (1, r, r^2, \dots, r^{d-1})^T$ where $r = a_1 + a_2\omega + \dots + a_{d-1}\omega^{d-1} \in \mathbb{Z}[\omega]$.

Backens *et al.* [3] established the analogous results in the qubit case: that ZH is universal for integer-valued matrices even without introducing labeled H -boxes as new generators. To show this, they construct an equivalent to all the integer labelled H -boxes: there is a simple expression with the same linear map as the $H(0)$ -box, and there is a successor gadget that increments the label of an arbitrary H -box by 1. Construction of negative integers is done by using a negation gadget. We will follow a similar path.

First, we already have a representation of $H(0) = |0\rangle$ (see Eq. (12) and take $k = 0$). Our immediate goal is then to construct a successor gadget to increment H -box labels. This will give us H -boxes with natural numbers as labels. The other possible labelled H -boxes are then straightforward to construct.

5.1 The qudit successor gadget

A successor gadget $S = (s_{ij})_{0 \leq i, j < d}$ that increments the label of an H -box by 1 has to satisfy the equation $SH(a) = H(a + 1)$ for any a . Looking at the definition of the qudit H -box, this means the coefficients s_{ij} of S have to satisfy the equations $(a + 1)^i = \sum_{j=0}^{d-1} s_{ij} a^j$. To solve this, we recall the binomial theorem, which states that $(a + 1)^i = \sum_{j=0}^i \binom{i}{j} a^j$. Hence, we must have $s_{ij} = \binom{i}{j}$, with the convention $\binom{i}{j} = 0$ for $j > i$. This means that the matrix S encodes *Pascal's triangle* in the form of a lower triangular matrix.

Note that because we already have a representation of $H(0)$, that we can use Lemma 4.2 to construct a ZH-diagram for any *binary* matrix: a matrix whose entries are only 0's and 1's. Our task then is to construct a ZH-diagram for S using only binary matrices. We achieve this by constructing each row of S individually and then *multiplexing* between them. To see how this works, first consider the linear map $R : \mathbb{C}^d \rightarrow \mathbb{C}^d, |i\rangle \mapsto |i\rangle + |i + 1\rangle$. One readily verifies that the coefficients of $R^j|0\rangle$ for $0 \leq j < d$ correspond to the $(j + 1)$ th row of Pascal's triangle. Hence, our desired successor gadget S satisfies the equation $R^j|0\rangle = S^T|j\rangle$. Therefore, we need some way to apply a different power of R to different inputs (and then take the transpose, which is straightforward). To do this we need a *multiplexer*.

Consider the linear map $M : (\mathbb{C}^d)^{d+1} \rightarrow \mathbb{C}^d$ defined by

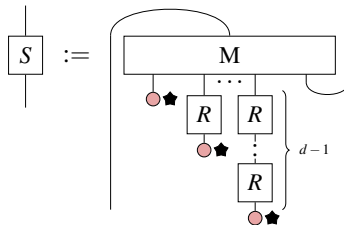
$$|x_0 \dots x_{d-1}\rangle \otimes |c\rangle \mapsto \begin{cases} |x_c\rangle & x_j = 0 \text{ for all } j \neq c \\ 0 & \text{otherwise.} \end{cases}$$

Let $|\varphi^i\rangle = \sum_{j=0}^{d-1} \lambda_{ij} |j\rangle$ be a collection of states for $0 \leq i < d$ where for all i the $|0\rangle$ coefficient λ_{i0} equals 1. Then for a fixed control value $0 \leq c < d$ we calculate:

$$\begin{aligned} M(|\varphi^0\rangle \otimes \dots \otimes |\varphi^{d-1}\rangle \otimes |c\rangle) &= \sum_{j_0=0}^{d-1} \dots \sum_{j_{d-1}=0}^{d-1} \lambda_{0j_0} \dots \lambda_{(d-1)j_{d-1}} M(|j_0 \dots j_{d-1}\rangle \otimes |c\rangle) \\ &= \lambda_{00} \dots \lambda_{(d-1)0} \sum_{j_c=0}^{d-1} \lambda_{cj_c} |j_c\rangle = \sum_{j_c=0}^{d-1} \lambda_{cj_c} |j_c\rangle = |\varphi^c\rangle. \end{aligned}$$

Hence, M multiplexes between these input states, using $|c\rangle$ as a control. As each row of Pascal's triangle starts with 1, the states $R^j|0\rangle$ have the right property. Hence $M(R^0|0\rangle \otimes \dots \otimes R^{d-1}|0\rangle \otimes |c\rangle) = R^c|0\rangle = S^T|c\rangle$. So by combining M and powers of R , and applying some appropriate transposes, we get S .

Both maps R and M are binary, meaning we can realize them as a phase-free ZH-diagram using Proposition 4.3. We perform this construction for R in Appendix F, while for M we only outline the first few steps, without actually constructing the diagram, due to its immense size. Using placeholders for M and R , we get the following diagram for our successor map S :



We can then realize any integer-labeled H -box where the label is non-negative: $H(n) = S^n H(0)$. Combining this with Lemma 4.2 means we can already construct arbitrary \mathbb{N} -valued matrices. To get all integer labeled H -boxes, we construct -1 in the next subsection.

5.2 Constructing all the labelled H-boxes

To construct more complicated labelled H-boxes we first realise that by taking the Schur product of two labelled H-boxes $H(a)$ and $H(b)$, we calculate the product of the labels (up to global scalar): $H(a) \star H(b) = \frac{1}{\sqrt{d}} H(a \cdot b)$. Since we already know how to construct $H(n)$ for any $n \in \mathbb{N}$, and we have the phase-free H-box $H(\omega)$ we can then also construct $H(\omega n)$ for any $n \in \mathbb{N}$. The second observation is that the successor gadget S adds 1 to the label regardless of the label, including non-integers. We can hence construct $S^m H(\omega n) = H(\omega n + m)$. Iterating these two steps we can then build $H(\omega^{d-1} n_1 + \omega^{d-2} n_2 + \dots + \omega n_{d-1} + n_d)$ where all the $n_j \in \mathbb{N}$.

Recall that for a d th root of unity $\omega \neq 1$ we have the identity $\sum_{j=1}^{d-1} \omega^j = -1$. Hence using the above procedure we can also construct $H(-1)$ as $H(-1) = H(\omega + \omega^2 + \dots + \omega^{d-1})$. Combining this with our construction of $H(\sum_j n_j \omega^j)$ for positive n_j above, we can then construct any $H(\sum_j a_j \omega^j)$ where $a_j \in \mathbb{Z}$. For instance, if we want to construct $H(n_2 \omega^2 - n_1 \omega - n_0)$ we do it with the following sequence of operations:

$$\begin{aligned} H(n_2) &\rightarrow H(n_2 \omega) \rightarrow H(-n_2 \omega) \rightarrow H(-n_2 \omega + n_1) \rightarrow H(-n_2 \omega^2 + n_1 \omega) \\ &\rightarrow H(-n_2 \omega^2 + n_1 \omega + n_0) \rightarrow H(n_2 \omega^2 - n_1 \omega - n_0). \end{aligned}$$

To summarise the whole construction of this section: we started out with the observation that with Lemma 4.2 we can represent arbitrary polynomials in phase-free ZH, and in this way represent arbitrary binary matrices (where every entry is either 0 or 1). We then found a way to construct a ‘‘successor gadget’’ S that increments the label of an H-box, $SH(a) = H(a + 1)$, from building blocks that are binary matrices which we know how to construct. Together with using the Schur product as a multiplication operation for H-box labels, this then allowed us to create H-boxes with arbitrary labels from $\mathbb{Z}[\omega]$. But then we can appeal to the same construction in Proposition 4.3 and Theorem 4.4 to conclude the following:

Theorem 5.1. The phase-free ZH-calculus for qudits of prime dimension d is universal for matrices over the ring $\mathbb{Z}[\omega]$, where $\omega = e^{2\pi i/d}$ is a d th root of unity.

Note that phase-free ZH-diagrams in fact represent a slightly larger fragment, corresponding to matrices $\frac{1}{\sqrt{d}} M$ where M has entries in $\mathbb{Z}[\omega]$. This is because we have global factors of $\frac{1}{\sqrt{d}}$ that cannot be made ‘local’ inside of the matrix. This is analogous to the qubit result for the phase-free ZH-calculus [3, Section 8.3] and the Toffoli+Hadamard circuit fragment [1].

6 Conclusion

We have introduced a qudit ZH-calculus, and showed how to generalise all the rules of the phase-free qubit calculus. We have established a universality result both for qudit ZH over an arbitrary ring, as well as for the phase-free ZH-calculus. We found that phase-free ZH-diagrams correspond to postselected circuits of Hadamard and $|0\rangle$ -controlled X gates. We showed that this gate set is approximately universal for qudit computation, and we found an almost asymptotically optimal strategy for compiling classical reversible qudit logic to this gate set.

The most immediate question about our qudit ZH-calculus is whether our generalisation of the qubit phase-free rules remains complete for qudits. It is possible to generalise the unique normal form for qubits from [3], but it is far from clear how to prove that we can reduce arbitrary diagrams to this normal form. Another open question is whether our construction in Theorem 2.1 is optimal, or whether it can be improved by a logarithmic factor. An interesting future direction would be to translate to and from

the ZXW-calculus [29] to achieve completeness of the qudit ZH-calculus and thereafter improve the challenging compilation of classical reversible logic in photonic quantum computing [17].

Acknowledgements: LY is supported by an Oxford - Basil Reeve Graduate Scholarship at Oriel College with the Clarendon Fund. PR was supported by the German Academic Scholarship Foundation. We thank the anonymous reviewers for their feedback.

References

- [1] Matthew Amy, Andrew N. Glauddell & Neil J. Ross (2020): *Number-Theoretic Characterizations of Some Restricted Clifford+T Circuits*. *Quantum* 4, p. 252, doi:10.22331/q-2020-04-06-252.
- [2] Miriam Backens & Aleks Kissinger (2019): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity*. In Peter Selinger & Giulio Chiribella, editors: *Proceedings of the 15th International Conference on Quantum Physics and Logic, Halifax, Canada, 3-7th June 2018, Electronic Proceedings in Theoretical Computer Science* 287, Open Publishing Association, pp. 23–42, doi:10.4204/EPTCS.287.2.
- [3] Miriam Backens, Aleks Kissinger, Hector Miller-Bakewell, John van de Wetering & Sal Wolffs (2021): *Completeness of the ZH-calculus*. doi:10.48550/arXiv.2103.06610.
- [4] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin & Harald Weinfurter (1995): *Elementary gates for quantum computation*. *Physical Review A* 52(5), pp. 3457–3467, doi:10.1103/physreva.52.3457.
- [5] Niel de Beaudrap (2021): *Well-tempered ZX and ZH Calculi*. In Benoît Valiron, Shane Mansfield, Pablo Arrighi & Prakash Panangaden, editors: *Proceedings 17th International Conference on Quantum Physics and Logic, Paris, France, June 2 - 6, 2020, Electronic Proceedings in Theoretical Computer Science* 340, Open Publishing Association, pp. 13–45, doi:10.4204/EPTCS.340.2.
- [6] Robert I. Booth & Titouan Carette (2022): *Complete ZX-calculi for the stabiliser fragment in odd prime dimensions*. doi:10.48550/arxiv.2204.12531.
- [7] Titouan Carette (2021): *When Only Topology Matters*. doi:10.48550/arxiv.2102.03178.
- [8] Titouan Carette (2021): *Wielding the ZX-calculus, Flexsymmetry, Mixed States, and Scalable Notations*. Theses, Université de Lorraine. Available at <https://hal.science/tel-03468027>.
- [9] Titouan Carette & Emmanuel Jeandel (2020): *A Recipe for Quantum Graphical Languages*. In Artur Czumaj, Anuj Dawar & Emanuela Merelli, editors: *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020), Leibniz International Proceedings in Informatics (LIPIcs)* 168, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 118:1–118:17, doi:10.4230/LIPIcs.ICALP.2020.118.
- [10] Bob Coecke & Ross Duncan (2008): *Interacting quantum observables*. In: *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, Lecture Notes in Computer Science, doi:10.1007/978-3-540-70583-3_25.
- [11] Bob Coecke & Ross Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016.
- [12] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons & Seyon Sivarajah (2020): *Phase Gadget Synthesis for Shallow Circuits*. In Bob Coecke & Matthew Leifer, editors: *Proceedings 16th International Conference on Quantum Physics and Logic, Chapman University, Orange, CA, USA., 10-14 June 2019, Electronic Proceedings in Theoretical Computer Science* 318, Open Publishing Association, pp. 213–228, doi:10.4204/EPTCS.318.13.
- [13] Shawn X. Cui & Zhenghan Wang (2015): *Universal quantum computation with metaplectic anyons*. *Journal of Mathematical Physics* 56(3), p. 032202, doi:10.1063/1.4914941.
- [14] Ross Duncan & Kevin Dunne (2016): *Interacting Frobenius Algebras Are Hopf*. In: *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16*, Association for Computing Machinery, New York, NY, USA, p. 535–544, doi:10.1145/2933575.2934550.

- [15] Richard D. P. East, Pierre Martin-Dussaud & John van de Wetering (2021): *Spin-networks in the ZX-calculus*. doi:10.48550/ARXIV.2111.03114.
- [16] Richard D.P. East, John van de Wetering, Nicholas Chancellor & Adolfo G. Grushin (2022): *AKLT-States as ZX-Diagrams: Diagrammatic Reasoning for Quantum States*. *PRX Quantum* 3, p. 010302, doi:10.1103/PRXQuantum.3.010302.
- [17] Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang & Bob Coecke (2023): *Light-matter interaction in the ZXW-calculus*. In: *Quantum Physics and Logic*.
- [18] Craig Gidney (2015): *Constructing Large Controlled Nots*. Available at <https://algassert.com/circuits/2015/06/05/Constructing-Large-Controlled-Nots.html>.
- [19] Andrew N. Glauddell, Neil J. Ross, John van de Wetering & Lia Yeh (2022): *Qutrit Metaplectic Gates Are a Subset of Clifford+T*. In François Le Gall & Tomoyuki Morimae, editors: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022), Leibniz International Proceedings in Informatics (LIPIcs)* 232, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 12:1–12:15, doi:10.4230/LIPIcs.TQC.2022.12. Available at <https://drops.dagstuhl.de/opus/volltexte/2022/16519>.
- [20] Daniel Gottesman (1999): *Fault-Tolerant Quantum Computation with Higher-Dimensional Systems*. *Chaos, Solitons & Fractals* 10(10), p. 1749–1758, doi:10.1016/s0960-0779(98)00218-5.
- [21] Jacques Hadamard (1893): *Resolution d'une question relative aux determinants*. *Bull. des sciences math.* 2, pp. 240–246.
- [22] Amar Hadzihasanovic (2017): *The algebra of entanglement and the geometry of composition*. Ph.D. thesis, University of Oxford, doi:10.48550/ARXIV.1709.08086. Available at <https://arxiv.org/abs/1709.08086>.
- [23] Amar Hadzihasanovic, Giovanni de Felice & Kang Feng Ng (2019): *A diagrammatic calculus of fermionic quantum circuits*. *Logical Methods in Computer Science* 15, doi:10.23638/LMCS-15(3:26)2019.
- [24] Aleks Kissinger (2022): *Phase-free ZX diagrams are CSS codes (...or how to graphically grok the surface code)*, doi:10.48550/ARXIV.2204.14038.
- [25] Mark Koch (2022): *Quantum Machine Learning using the ZXW-Calculus*. Master's thesis, University of Oxford. Available at <https://www.cs.ox.ac.uk/people/aleks.kissinger/theses/koch-thesis.pdf>.
- [26] Mateusz Kupper (2019): *Analysis of quantum hypergraph states in the ZH-calculus*. Master's thesis, University of Edinburgh. Available at https://www.researchgate.net/publication/339377025_Analysis_of_quantum_hypergraph_states_in_the_ZH-calculus.
- [27] Louis Lemonnier, John van de Wetering & Aleks Kissinger (2021): *Hypergraph Simplification: Linking the Path-sum Approach to the ZH-calculus*. In Benoît Valiron, Shane Mansfield, Pablo Arrighi & Prakash Panangaden, editors: *Proceedings 17th International Conference on Quantum Physics and Logic, Paris, France, June 2 - 6, 2020, Electronic Proceedings in Theoretical Computer Science* 340, Open Publishing Association, pp. 188–212, doi:10.4204/EPTCS.340.10.
- [28] Kang Feng Ng (2018): *Completeness of the ZW and ZX calculi*. Ph.D. thesis, University of Oxford.
- [29] Boldizsár Poór, Quanlong Wang, Razin Shaikh A., Lia Yeh, Richie Yeung & Bob Coecke (2023): *Completeness for arbitrary finite dimensions of ZXW-calculus, a unifying calculus*. doi:10.48550/ARXIV.2302.12135.
- [30] Boldizsár Poór, Robert I. Booth, Titouan Carette, John van de Wetering & Lia Yeh (2023): *The Qutip Stabiliser ZX-travaganza: Simplified Axioms, Normal Forms and Graph-Theoretic Simplification*. In: *To appear in proceedings of the 2023 International Conference on Quantum Physics and Logic*.
- [31] Razin A. Shaikh, Quanlong Wang & Richie Yeung (2022): *How to sum and exponentiate Hamiltonians in ZXW calculus*. doi:10.48550/ARXIV.2212.04462.
- [32] Yaoyun Shi (2003): *Both Toffoli and Controlled-NOT Need Little Help to Do Universal Quantum Computing*. *Quantum Info. Comput.* 3(1), p. 84–92, doi:10.5555/2011508.2011515.

- [33] John van de Wetering & Lia Yeh (2022): *Building Qutrit Diagonal Gates from Phase Gadgets*. In: *Quantum Physics and Logic*, Oxford. Available at https://www.qplconference.org/proceedings2022/QPL_2022_paper_16.pdf.
- [34] Renaud Vilmart (2021): *Quantum Multiple-Valued Decision Diagrams in Graphical Calculi*. In Filippo Bonchi & Simon J. Puglisi, editors: *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021), Leibniz International Proceedings in Informatics (LIPIcs) 202*, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 89:1–89:15, doi:10.4230/LIPIcs.MFCS.2021.89.
- [35] Renaud Vilmart (2021): *The Structure of Sum-Over-Paths, its Consequences, and Completeness for Clifford*. In Stefan Kiefer & Christine Tasson, editors: *Foundations of Software Science and Computation Structures*, Springer International Publishing, Cham, pp. 531–550, doi:10.1007/978-3-030-71995-1_27.
- [36] Renaud Vilmart (2022): *Completeness of Sum-Over-Paths for Toffoli-Hadamard and the Dyadic Fragments of Quantum Computation*, doi:10.48550/ARXIV.2205.02600.
- [37] Quanlong Wang (2018): *Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics*. In Bob Coecke & Aleks Kissinger, editors: *Proceedings 14th International Conference on Quantum Physics and Logic, Nijmegen, The Netherlands, 3-7 July 2017, Electronic Proceedings in Theoretical Computer Science 266*, Open Publishing Association, pp. 58–70, doi:10.4204/EPTCS.266.3.
- [38] Quanlong Wang (2021): *An Algebraic Axiomatisation of ZX-calculus*. In Benoît Valiron, Shane Mansfield, Pablo Arrighi & Prakash Panangaden, editors: *Proceedings 17th International Conference on Quantum Physics and Logic, Paris, France, June 2 - 6, 2020, Electronic Proceedings in Theoretical Computer Science 340*, Open Publishing Association, pp. 303–332, doi:10.4204/EPTCS.340.16.
- [39] Quanlong Wang (2021): *A non-anyonic qudit ZW-calculus*, doi:10.48550/arXiv.2109.11285.
- [40] Quanlong Wang (2021): *Qufinite ZX-calculus: a unified framework of qudit ZX-calculi*. doi:10.48550/ARXIV.2104.06429.
- [41] Quanlong Wang & Xiaoning Bian (2014): *Qutrit Dichromatic Calculus and Its Universality*. In Bob Coecke, Ichiro Hasuo & Prakash Panangaden, editors: *Proceedings of the 11th workshop on Quantum Physics and Logic, Kyoto, Japan, 4-6th June 2014, Electronic Proceedings in Theoretical Computer Science 172*, Open Publishing Association, pp. 92–101, doi:10.4204/EPTCS.172.7.
- [42] Quanlong Wang & Richie Yeung (2022): *Differentiating and Integrating ZX Diagrams*. doi:10.48550/ARXIV.2201.13250.
- [43] Yuchen Wang, Zixuan Hu, Barry C. Sanders & Sabre Kais (2020): *Qudits and High-Dimensional Quantum Computing*. *Frontiers in Physics* 8, doi:10.3389/fphy.2020.589504.
- [44] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*. doi:10.48550/arxiv.2012.13966.
- [45] Lia Yeh (2023): *Scaling W states in the qudit Clifford hierarchy*. In: *Proceedings of the 1st International Workshop on the Art, Science, and Engineering of Quantum Programming*, doi:10.48550/arXiv.2304.12504.
- [46] Lia Yeh & John van de Wetering (2022): *Constructing All Qutrit Controlled Clifford+T gates in Clifford+T*. In Claudio Antares Mezzina & Krzysztof Podlaski, editors: *Reversible Computation*, Springer International Publishing, Cham, pp. 28–50, doi:10.1007/978-3-031-09005-9_3.
- [47] Wei Zi, Qian Li & Xiaoming Sun (2023): *Optimal Synthesis of Multi-Controlled Qudit Gates*. In: *To appear in Design Automation Conference*, San Francisco, CA, doi:10.48550/arXiv.2303.12979.

A Building all prime-dimensional classical reversible gates

In this section of the appendix, we derive Theorem 2.1. First, we define a few gates which we will use. When an explicit proof is not given, it is because the construction can then be easily checked for any fixed odd prime dimension, and realising that they only affect the $\{|0\rangle, |1\rangle\}$ subspace.

The first is a permutation cycle of length 3 which maps $|00\rangle \mapsto |01\rangle$, $|01\rangle \mapsto |10\rangle$, and $|10\rangle \mapsto |00\rangle$, and is identity on all other computational basis states:

Lemma A.1.

$$\boxed{\begin{pmatrix} 00, \\ 10, \\ 01 \end{pmatrix}} = \begin{array}{c} \text{---} \textcircled{0} \text{---} \text{X}^\dagger \text{---} \textcircled{0} \text{---} \text{X} \text{---} \\ | \\ \text{---} \text{X}^\dagger \text{---} \textcircled{0} \text{---} \text{X} \text{---} \textcircled{0} \text{---} \end{array} \quad (14)$$

Proof. Consider the action on a basis state $|x, y\rangle$ where $x, y \in \mathbb{Z}_d$. After the first gate, the state is $|x, y + x^{d-1} - 1\rangle$. After the second gate, the state is $|x + (y + x^{d-1} - 1)^{d-1} - 1, y + x^{d-1} - 1\rangle$. After the third gate, the state is $|x + (y + x^{d-1} - 1)^{d-1} - 1, y + x^{d-1} - (x + (y + x^{d-1} - 1)^{d-1} - 1)^{d-1}\rangle$. At this point, by case distinctions on x and y being 0, 1, or otherwise, we can compute that the bottom output state must be:

$$\begin{cases} |0\rangle, & \text{if } x = 0 \text{ and } y = 1 \\ |1\rangle, & \text{if } x = 1 \text{ and } y = 0 \\ |y\rangle, & \text{else} \end{cases} \quad (15)$$

From here on, the fourth and final gate can be seen to apply when either both $x \neq 1$ and $y = 0$, or both $x = 0$ and $y = 1$. Hence the top output state must be:

$$\begin{cases} |0\rangle, & \text{if } x = 1 \text{ and } y = 0 \\ |1\rangle, & \text{if } x = 0 \text{ and } y = 0 \\ |x\rangle, & \text{else} \end{cases} \quad (16)$$

Upon inspection, the circuit sends $|00\rangle \mapsto |01\rangle$, $|01\rangle \mapsto |10\rangle$, and $|10\rangle \mapsto |00\rangle$, and is identity on all other computational basis states. \square

Note that here the $|0\rangle$ -controlled X^\dagger gate can be constructed from the $|0\rangle$ -controlled X gate, by repeating that one $d - 1$ times.

Using this, we can build the $|0\rangle$ - and $|1\rangle$ -controlled X_{01} gate, where the X_{01} gate is a single-qudit permutation gate that maps $|0\rangle \mapsto |1\rangle$, $|1\rangle \mapsto |0\rangle$, and is identity on all other computational basis states.

Lemma A.2.

$$\begin{array}{c} \text{---} \textcircled{0} \text{---} \textcircled{1} \text{---} \\ | \\ \text{---} \text{X}_{01} \text{---} \text{X}_{01} \text{---} \end{array} = \begin{array}{c} \text{---} \text{X}^\dagger \text{---} \Lambda \text{---} \boxed{\begin{pmatrix} 00, \\ 10, \\ 01 \end{pmatrix}} \text{---} \text{X} \text{---} \Lambda \text{---} \boxed{\begin{pmatrix} 00, \\ 10, \\ 01 \end{pmatrix}} \text{---} \\ | \\ \text{---} \Lambda \text{---} \boxed{\begin{pmatrix} 00, \\ 10, \\ 01 \end{pmatrix}} \text{---} \Lambda \text{---} \boxed{\begin{pmatrix} 00, \\ 10, \\ 01 \end{pmatrix}} \text{---} \end{array} \quad (17)$$

Note that here we have a Λ -controlled gate which for $\Lambda(U)$ implements $|x, y\rangle \mapsto U^x|y\rangle$. In this particular case, $\Lambda(X)|x, y\rangle = |x, x + y\rangle$ is the CX gate, and is Clifford.

Corollary A.3. The X_{01} gate can be synthesized, by setting the control qudit as an ancilla in the $|0\rangle$ state.

We can then build the $|0\rangle$ -controlled X_{01} gate.

Lemma A.4.

$$\begin{array}{c} \text{---} \textcircled{0} \text{---} \\ | \\ \text{---} \text{X}_{01} \text{---} \end{array} = \underbrace{\begin{array}{c} \text{---} \text{X} \text{---} \text{X} \text{---} \textcircled{0} \text{---} \textcircled{1} \text{---} \text{X} \text{---} \\ | \\ \text{---} \text{X}_{01} \text{---} \text{X}_{01} \text{---} \text{X}_{01} \text{---} \end{array}}_{\text{repeat } \frac{d-1}{2} \text{ times}} \quad (18)$$

We then utilise the following generalisation to all odd qudit dimensions d , for which the qubit analogue is in [4, Lemma 7.5] and the qutrit analogue is in [46, Lemma 5]:

Lemma A.5.

$$\text{---} \begin{array}{c} \textcircled{0} \\ | \\ \textcircled{0} \\ | \\ \boxed{X_{01}} \end{array} \text{---} = \text{---} \begin{array}{c} \textcircled{0} \quad \textcircled{0} \quad \textcircled{0} \\ | \quad | \quad | \\ \boxed{X_{01}^\dagger} \quad \Lambda \quad \boxed{X} \\ | \quad | \quad | \\ \boxed{X_{01}} \quad \boxed{X_{01}} \quad \boxed{X_{01}} \end{array} \text{---} \quad (19)$$

where $\Lambda(X_{01})$ can be further decomposed into controlling X_{01} on all odd computational basis states.

Corollary A.6. Any of these controls can be changed to any other computational basis by conjugating by X 's, or to Λ controls by repeating the construction once for each odd computational basis state control.

Corollary A.7. As we will shortly discuss in Proposition A.9, any permutation can be generated by 2-cycles i.e. permutations exchanging only two elements. For example, the $|00\rangle$ -controlled X gate can be obtained by $|00\rangle$ -controlling each gate in the decomposition:

$$\text{---} \boxed{X} \text{---} = \text{---} \boxed{X_{(d-2)(d-1)}} \boxed{X_{(d-3)(d-2)}} \cdots \boxed{X_{12}} \boxed{X_{01}} \text{---} \quad (20)$$

In previous work [46], a construction was found that has polynomial Clifford+ T gate count to decompose any tritstring controlled qutrit Toffoli. It was left open whether there was a better construction with linear gate count. Specifically, whether it was possible to generalise Gidney's construction (reprinted from [18]):

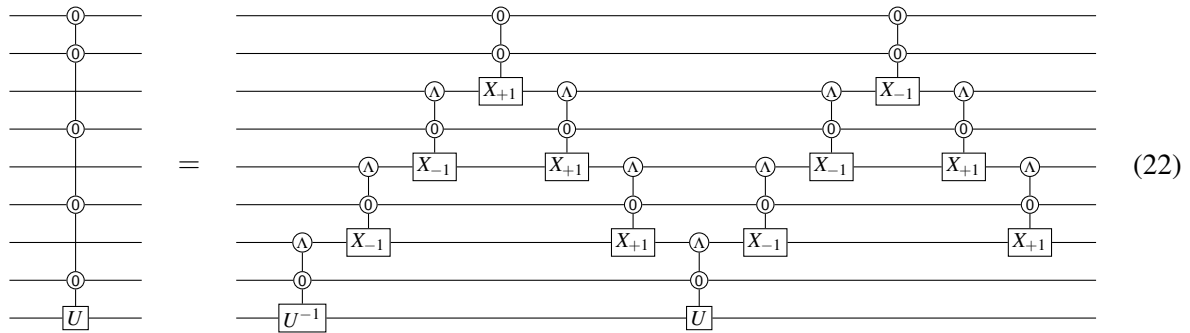
CⁿNOT from n-2 Borrowed bits

$$\begin{array}{c} \text{Controls} \\ \begin{array}{c} \text{A} \\ \text{B} \\ \text{C} \\ \text{D} \\ \text{E} \end{array} \\ \text{Target} \\ \text{T} \end{array} \text{---} = \begin{array}{c} \text{A} \\ \text{B} \\ \text{X}_1 \\ \text{C} \\ \text{X}_2 \\ \text{D} \\ \text{X}_3 \\ \text{E} \\ \text{T} \end{array} \text{---} \quad (21)$$

If such a construction did not exist, it would be hard to justify ever using qudit Toffolis as opposed to qubit Toffolis, as they would be asymptotically more expensive. However, it turns out that there is a qudit version with an analogous structure.

Lemma A.8. Any odd-dimensional qudit gate controlled on n qudits, admits a decomposition with $n - 2$ borrowed ancillae qudits, with $O(n)$ gate count.

Proof. Any gate U can be controlled on $|0\rangle^{\otimes n}$ with $n - 2$ borrowed ancillae:



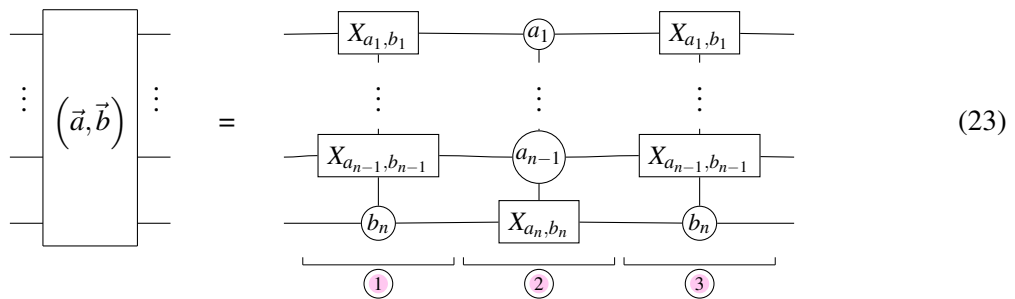
This approach is adaptable to arbitrary n . The control qudits can be conjugated by X gates to generalize this from $|0\dots 0\rangle$ -controlled, to control on any ditstring. This also applies if any of the control qudits are other types of controls, for instance Λ -controlled.

As an added note, if it is preferred to construct $\Lambda(U)$ instead of $\Lambda(|0\rangle\text{-controlled } U)$, the bottommost control qudit in the above decomposition can be omitted at the cost of adding one borrowed ancilla. \square

This lets us immediately apply the following proposition reprinted from Ref. [46] for qutrits, which holds for arbitrary qudit dimension.

Proposition A.9. Let $\vec{a} = (a_1, \dots, a_n)$ and $\vec{b} = (b_1, \dots, b_n)$ be any two ditstrings of length n . Then we can exactly implement a unitary which maps the basis states $|\vec{a}\rangle \mapsto |\vec{b}\rangle$ and $|\vec{b}\rangle \mapsto |\vec{a}\rangle$, and is identity on all other computational basis states, with gate count asymptotically the same as the Toffoli controlled on a ditstring of length n , which from Lemma A.8 is $O(n)$ for any odd qudit dimension.

Proof. We assume $\vec{a} \neq \vec{b}$, or the permutation 2-cycle (\vec{a}, \vec{b}) would just be the identity operation on all inputs. As \vec{a} and \vec{b} differ, they must differ by at least one character. Without loss of generality suppose that $a_n \neq b_n$. Consider the following circuit:



Here the circles denote controls on the value of an a_j or b_j , which control whether a X_{a_j, b_j} operation is applied (which we take to be the identity if $a_j = b_j$). Hence, the gate in Step 2 is a many-controlled X_{a_n, b_n} gate, which for odd d we know how to build by Lemma A.8 using $O(n)$ gates. We conjugate this gate, in Steps 1 and 3, by $n - 1$ gates that are each Clifford equivalent to the $|0\rangle$ -controlled X_{01} gate. Hence for odd d , the above circuit requires $O(n)$ gates to implement.

This circuit indeed implements the (\vec{a}, \vec{b}) 2-cycle, which we can see by enumerating the possible input cases.

- When the input is \vec{a} : Only steps 2 and 3 fire (as $b_n \neq a_n$), outputting \vec{b} .
- When the input is \vec{b} : Steps 1 and 2 fire, outputting \vec{a} .

Observe that when Step 2 does not fire, Steps 1 and 3 always combine to the identity gate. Therefore, we only need to consider the remaining cases where Step 2 does fire.

- When both Steps 1 and 2 fired: The input had to have been \vec{b} .
- When Step 2 fired, but Step 1 didn't fire: Either the input was \vec{a} , or the last input character was neither a_n nor b_n in which case the overall operation is the identity.

Therefore, the circuit in Eq. (23) maps \vec{a} to \vec{b} , \vec{b} to \vec{a} , and is identity on all other ditstrings. \square

As any permutation can be written as product of disjoint 2-cycles (this is well known, and explained in Ref. [46]), we can thus break down any d -ary classical reversible circuit on n dits as a permutation of length at most d^n , which can always be broken down into a product of $d^n - 1$ 2-cycles. Thus, we can apply the following qudit equivalent of the qutrit result in Ref. [46]:

Theorem A.10 (Restatement of Theorem 2.1). For any odd qudit dimension d , any d -ary classical reversible function $f : \mathbb{Z}_d^n \rightarrow \mathbb{Z}_d^n$ on n dits can be constructed by a circuit of $O(d^n)$ $|0\rangle$ -controlled X gates and $O(n)$ ancillae prepared in the $|0\rangle$ state.

Proof. We view f as a permutation of size d^n . This permutation consists of cycles, each of which can be decomposed into 2-cycles. This full decomposition requires at most $d^n - 1$ 2-cycles. Implementing each of these 2-cycles requires $O(n)$ gate count. Therefore, the asymptotic gate count of the overall construction is $O(d^n)$. \square

This is within a $\log(n)$ factor of the gate count necessary, by generalizing our proof from Ref. [46] to the qudit setting:

Proposition A.11 (Restatement of Proposition 2.2). For any qudit dimension d , there exist d -ary classical reversible functions $f : \mathbb{Z}_d^n \rightarrow \mathbb{Z}_d^n$ that require at least $O(nd^n / \log n)$ single-qudit and two-qudit gates to construct, even when allowed $\Omega(n)$ ancillae.

Proof. Fix any finite gate set consisting of single-qudit and two-qudit gates, and suppose we have $O(n)$ ancillae. Then taking into account positioning, we have $O(n^2)$ possible single-gate circuits (the square comes from positioning the two-qudit gates). Let's suppose we can bound this by cn^2 for some constant c . Hence, using N gates from this gate set, we can construct at most $(cn^2)^N = c^N n^{2N}$ different circuits. There are exactly $(d^n)!$ different d -ary classical reversible functions on ditstrings of length n (where $k!$ denotes the factorial of k). In order to write down every such permutation we must hence have a number of gates N such that at least $c^N n^{2N} \geq (d^n)!$. Taking the logarithm on both sides and using $\log(k!) \geq \frac{1}{2}k \log k$ we can rewrite this inequality to $N \log c + 2N \log n \geq \frac{1}{2}d^n \cdot n \log d$. Factoring out N gives $N \geq \frac{\log d}{2} \frac{nd^n}{\log c + 2 \log n} \geq \frac{\log d}{6} \frac{nd^n}{\log n}$ for $n \geq c$, which shows that we must have $N = O(nd^n / \log n)$. \square

B Building the qutrit metaplectic gate

This section mostly follows Ref. [19, Section 3] to construct the qutrit metaplectic gate R , a single-qutrit non-Clifford gate with matrix $\text{diag}(1, 1, -1)$. However, the key difference is that here we restrict the allowed gate set to $H+|0\rangle$ -controlled X instead of qutrit Clifford+ T .

In all prime qudit dimensions, adding a single-qudit non-Clifford gate to gates generating the Clifford group achieves approximately universal quantum computation [20]. This was explicitly proven for qutrits in Ref. [13, Theorem 2].

Therefore, explicit construction of the R gate, in addition to the qutrit Clifford gates constructed in the proof of Theorem 2.3, suffice to show approximately universality of the $H+|0\rangle$ -controlled X gate set.

We remark that in this section only, we match the definition of the H gate in Ref. [19], which differs from the definition in the main body of the paper by a global phase of i . In any case, the below construction exactly synthesizes the R gate with the correct global phase, regardless of the global phase in the definition of the H gate.

A code implementation of the below is available at <https://github.com/lia-approves/qudit-circuits/blob/main/qutrit-Toffoli-Hadamard/ToffH.m>.

By [19, Equation 12], where \mathbb{I} denotes the single-qutrit identity matrix,

$$\begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline -H^\dagger \\ \hline \end{array} \begin{array}{|c|} \hline -H^\dagger \\ \hline \end{array} \begin{array}{|c|} \hline -H^2 \\ \hline \end{array} = \begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline -\mathbb{I} \\ \hline \end{array} = \begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline R \\ \hline \end{array} \quad (24)$$

By [46, Lemma 11], where Z^\dagger is the conjugation of the X gate by H ,

$$\begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} = \begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline \omega\mathbb{I} \\ \hline \end{array} = \begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline X_{02} \\ \hline \end{array} \begin{array}{|c|} \hline Z^\dagger \\ \hline \end{array} \begin{array}{|c|} \hline X_{02} \\ \hline \end{array} \begin{array}{|c|} \hline Z^\dagger \\ \hline \end{array} \quad (25)$$

By a modification of [19, Lemma 21] to instead use the gate in Equation (25), leveraging the decomposition of H into Z and X rotations by [40, Remark 2.3],

$$\begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline -H^\dagger \\ \hline \end{array} = \begin{array}{c} \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H^2 \\ \hline \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H \\ \hline \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H^2 \\ \hline \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H^3 \\ \hline \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H^2 \\ \hline \end{array} \begin{array}{|c|} \hline Q[2] \\ \hline \end{array} \begin{array}{|c|} \hline H^2 \\ \hline \end{array} \quad (26)$$

Finally, we note that for qutrits $X_{12} = -H^2$, enabling us to substitute the $|0\rangle$ -controlled X_{12} gate into the $|0\rangle$ -controlled H^2 gate in Equation (24) to realise the qutrit metaplectic gate.

C Soundness of qudit ZH rewrite rules

In this appendix, we argue for the soundness of the rewrite rules introduced in Figure 1. The soundness of the qudit Z -spider fusion rule (zs) is already well known from ZX -literature [6], and similarly for the identity rule. For the remaining rules, we appeal in parts to the algebraic interpretation of ZH -generators from Eq. (4), which we will prove along the way.

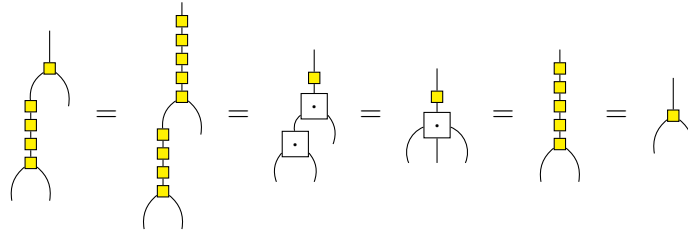
First, to verify our claim that the H -box indeed corresponds to multiplication, observe that

$$\frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \omega^{ijk} H^3 |k\rangle \langle i| \langle j| = \frac{1}{d} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \sum_{v=0}^{d-1} \omega^{ijk-vk} |v\rangle \langle i| \langle j| = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} |i \cdot j\rangle \langle i| \langle j|$$

using the identity $\sum_{k=0}^{d-1} \sum_{\mu=0}^{d-1} (\omega^\mu)^k = d$, since for any root of unity $\zeta \neq 1$ we have $\sum_{k=0}^{d-1} \zeta^k = 0$. Similarly, we get

$$H^2 |i\rangle = \frac{1}{d} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \omega^{j+jk} |k\rangle = \frac{1}{d} \sum_{k=0}^{d-1} |k\rangle \sum_{j=0}^{d-1} (\omega^{i+k})^j = |-i\rangle.$$

Then, using the identity $H^4 = \text{id}$, we get



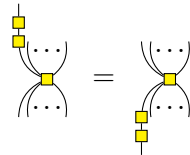
Using flexsymmetry and inductive application of the above identity, we get a generalization of the qubit H -fusion rule, with fusion happening via H^3 instead of H . Only fusion of 1-ary H -boxes is not covered. For this, however, observe that

$$HH(\omega) = \frac{1}{d} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \omega^{i+j} |j\rangle = |-1\rangle$$

and thus $H^3H(\omega) = |1\rangle$. Since 1 is the unit of multiplication modulo d , it merges into H -boxes.

To arrive at our H-box contraction rule, we first introduce the following lemma:

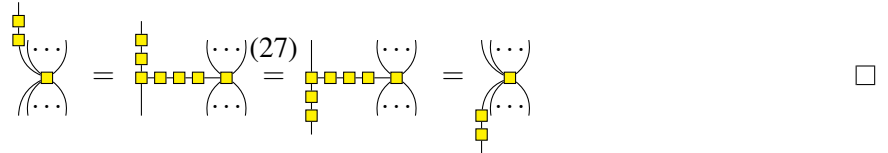
Lemma C.1. We can freely transfer a double quantum Fourier transform between the legs of a H -box, e.g.



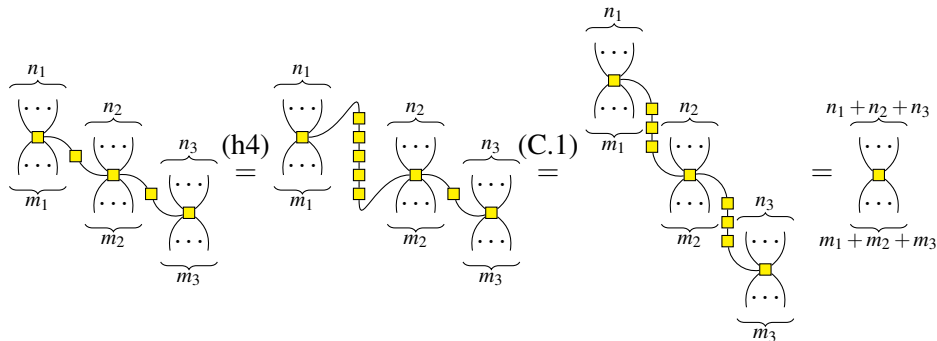
Proof. We have

$$\begin{aligned} \begin{array}{c} \square \\ \square \\ | \end{array} &= \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \omega^{ijk} |-k\rangle \langle i| \langle j| = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \bar{\omega}^{ijk} |k\rangle \langle i| \langle j| = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \sum_{k=0}^{d-1} \omega^{ijk} |k\rangle \langle i| \langle -j| = \begin{array}{c} | \\ \square \\ \square \end{array} \quad (27) \end{aligned}$$

Using this we then have

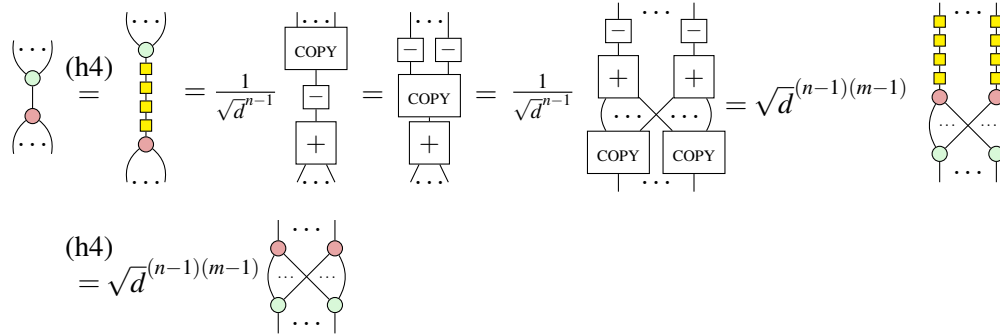


Then we get:



To argue soundness of the two bialgebra rules, we adopt the proof strategy of Backens et al [3, 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 Eq. (4) that we can interpret them as arithmetic operations in \mathbb{Z}_d .

Proof of Z/X-bialgebra rule (ba1). To see why the X-spider correspond to addition modulo d , we refer to existing literature on ZX-calculus, for instance [33]. Using this, we then get



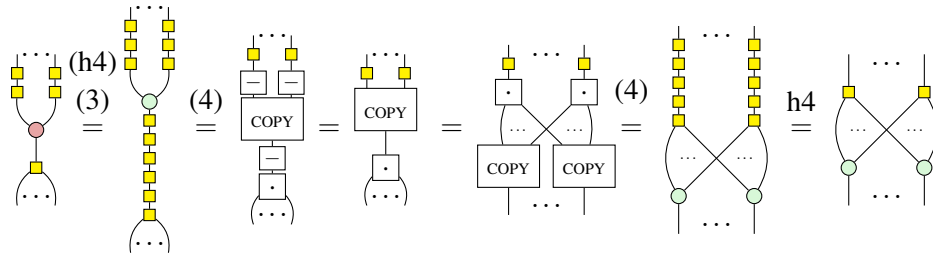
Lastly, observe that

$$\blacktriangle = \begin{matrix} (3) & (3) \\ \circ & \circ \\ \bullet & \bullet \end{matrix} = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} \zeta^{ij} = \sqrt{d}$$

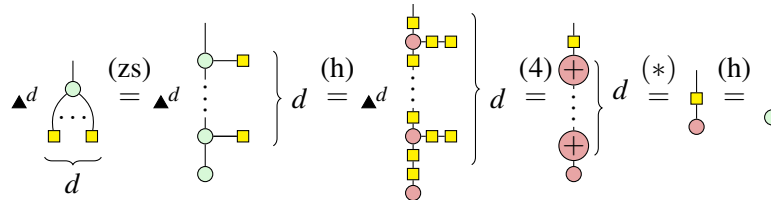
to see that \blacktriangle indeed corresponds to \sqrt{d} . □

The remaining proofs are straightforward:

Proof of Z/H-bialgebra rule (ba2).



Proof of cyclic rule (cy).



Here the step marked (*) uses the fact that repeating Pauli-X a total of d times in succession is just the identity. To see our gadget of H-boxes and X-spider really represents Pauli-X, push the two H -boxes from the input through to the output to get an addition gadget, and observe that $H^3H(\omega) = |1\rangle$. □

D Further Rewrite Rules

In the qubit ZH-calculus of Backens *et al.*, there is one further rewrite rule (the *ortho* rule) we have not yet considered in the qudit setting. We make up for that in this appendix. Additionally, we look at two simpler rules [3, Lem. 2.28, Lem. 5.1] Backens *et al.* have proven to be equivalent to ortho in the qubit setting [3, Thm. 8.6], and generalise those as well (though we do not show equivalence).

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

$$\forall x_0, x_1, y: x_0y = x_1(y + 1) \iff x_0y = 0 = x_1(y + 1).$$

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 x_0y and $x_1(y + 1)$ is always zero. Thus, if $x_0y = x_1(y + 1)$, both products must equal 0.

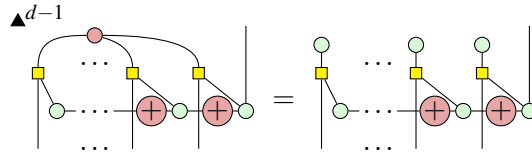
We can generalise this argument to $\mathbb{Z}/d\mathbb{Z}$ for arbitrary d :

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

and thus

$$\forall x_0, \dots, x_{d-1}, y: x_0y = \dots = x_{d-1}(y + d - 1) \iff \forall i \in \{0, \dots, d - 1\}: x_i(y + i) = 0.$$

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:

$$\forall x_0, \dots, x_{d-1}, y: x_0y = \dots = x_{d-1}(y + d - 1) \iff \text{at most one } x_i \neq 0,$$

however, this condition does not seem to be easily expressible as a ZH-diagram.

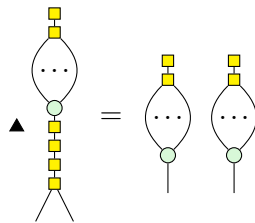
Backens *et al.*'s Lemma 2.28 of [3] states that

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

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

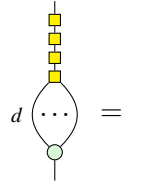
$$\forall x, y: xy \neq 0 \iff x \neq 0 \wedge y \neq 0$$

Obviously, this does not hold for d composite, since then $\mathbb{Z}/d\mathbb{Z}$ admits zero-divisors. Using Eq. (12) we can realize this as the diagrammatic equation



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

Lastly, Backens *et al.*'s Lemma 5.1 is a diagrammatic version of the *Frobenius identity*, which states that for prime d , we have $x^d = x$ for all $x \in \mathbb{Z}_d$. We have used statements similar to this all throughout this paper. Diagrammatically, the identity becomes



E Derived Rewrite Rules

In this appendix we derive simple, yet often useful, rewrite rules using the qudit ZH-calculus. If a rule generalises a known rule for qubit ZH or ZX, we provide references to where the rules were first proven. In these cases, the proofs are often virtually identical and only need some adjustments regarding the number of Hadamards.

Note that Lemmas E.3 and E.4 also work with the colors of the spiders inverted. Lemma E.6 also works with arbitrarily many wires connecting the spiders, similar to (zs). Lemmas E.8 through E.11 are copy rules resulting from special cases of our bialgebra rules. Following, we derive some rules for Z- and X-spiders connected via multiple wires, including the *ZX Hopf-rule* (Lemma E.12), the qudit version of *complementarity* (Lemma E.13), as well as a generalisation of Yeh and van de Wetering's qutrit *special* rule (Lemma E.14). Lastly, we have some rules about scalar cancellation (Lemma E.15 and E.16).

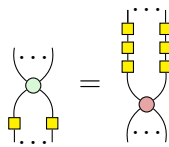
Lemma E.1.



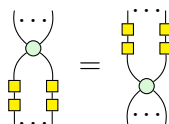
Lemma E.2.



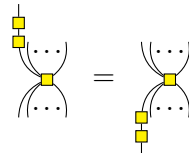
Lemma E.3.



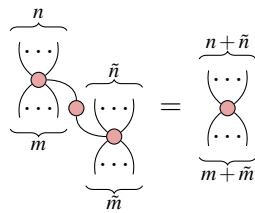
Lemma E.4.



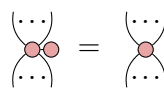
Lemma E.5.



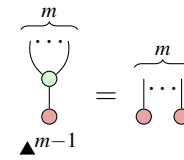
Lemma E.6.



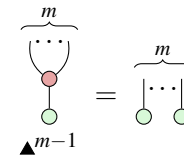
Lemma E.7.



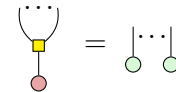
Lemma E.8.



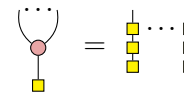
Lemma E.9.



Lemma E.10.

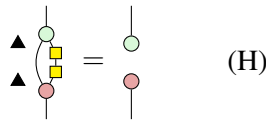


Lemma E.11.

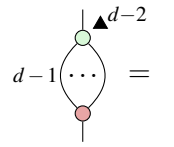


(xs)

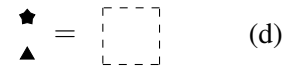
Lemma E.12 (Hopf).



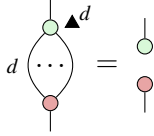
Lemma E.14.



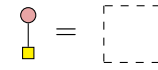
Lemma E.16.



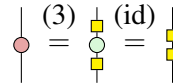
Lemma E.13.



Lemma E.15.



Proof of E.1 [3, Lem. 2.11].



□

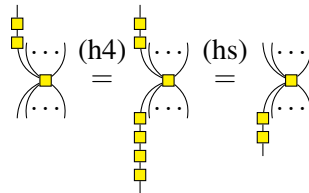
Proof of E.2 [3, Lem. 2.11]. Follows immediately from (h4).

□

Proof of E.3 [3, Lem. 2.15, Lem. 2.16] and E.4. Follow by alternating application of (h), (3) and then (h4).

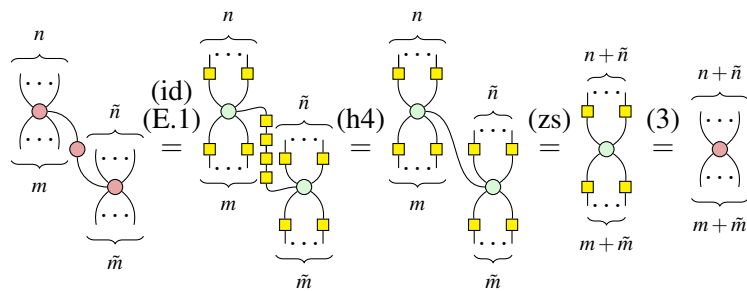
□

Proof of E.5.



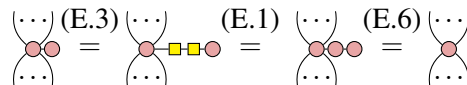
□

Proof of E.6 [3, Lem. 2.10].



□

Proof of E.7.

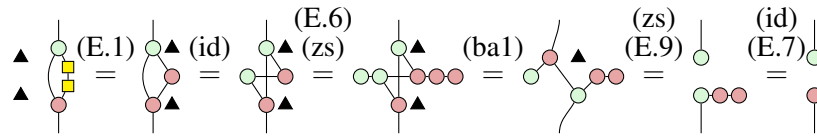


□

Proofs of E.8 through E.11 [3, Lem. 2.21, Lem. 2.23, Lem. 2.26]. Follow immediately from (ba1) and (ba2).

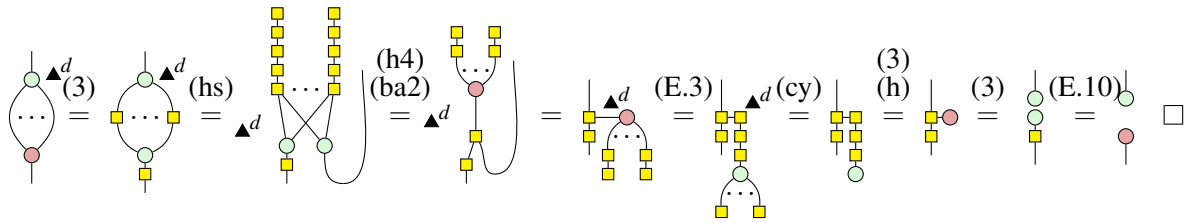
□

Proof of E.12.

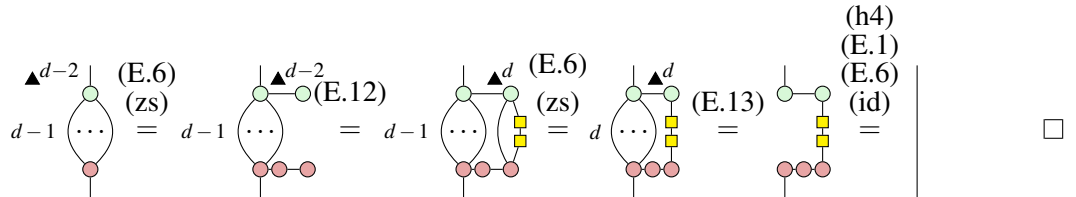


Note that this proof is very similar to the one given by Feng [28, Eq. 4.3] and the abstract one done using the dualizer by Duncan and Dunne [14, Thm. 4.6]. Lastly, Booth and Carette considered this Lemma to be a rewrite rule [6, Eq. 21] □

Proof of E.13 [3, Lem 2.30].

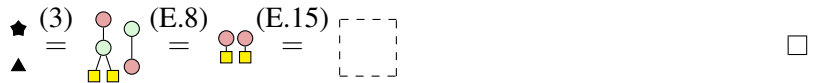


Proof of E.14 [6, Eq. 21].



Proof of E.15 [3, Lem. 2.5]. Unchanged from the qubit case via (ba2). □

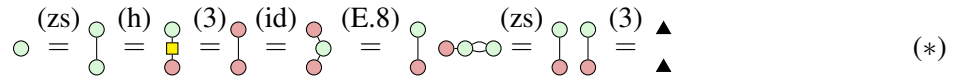
Proof of E.16 [3, Lem. 2.3, Lem 2.4].



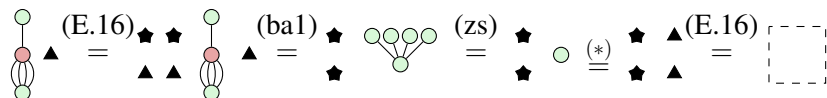
Remark E.17. Note that we can also express \star using just Z- and X-spiders in a slightly more intuitive way via



The proof of this is slightly more complicated. It is based upon the identity



We then have



and adding a \blackstar to both sides yields the desired identity. Furthermore, Booth and Carette [6, Fig 1.] give

$$\blackstar = \text{diagram of a loop with a red dot at the bottom and a green dot at the top}$$

For this version, we have

$$\blacktriangle \text{ (loop with red dot)} \stackrel{\text{(id)}}{=} \blacktriangle \text{ (loop with red dot)} \stackrel{\text{(E.6) (zs)}}{=} \blacktriangle \text{ (loop with red dot)} \stackrel{\text{(ba1)}}{=} \blacktriangle \text{ (loop with red dot)} \stackrel{\text{(E.16) (E.8)}}{=} \blackstar \text{ (loop with red dot)} \stackrel{\text{(E.6) (3)}}{=} \text{dashed box}$$

F Building Diagrams

In this appendix, we apply the algorithm outlined in Section 4 to construct a phase-free ZH-diagram for the operator R we used in Section 5 to construct our successor gadget. We also outline how one would go about constructing a phaseless ZH-diagram for M , our multiplexer map.

Recall that we defined $R|i\rangle = |i\rangle + |i+d-1\rangle$, meaning we have

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

This means that the formula φ_R describing the locations of the ones in this matrix is

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

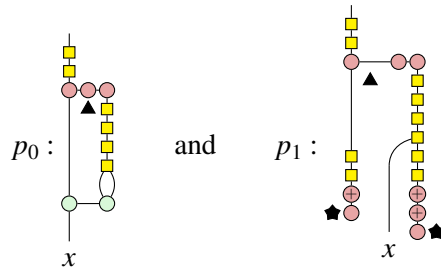
Following the construction of Proposition 4.1, the polynomial

$$p_R(x, y) = (x - y) \cdot (x + 1 - y)$$

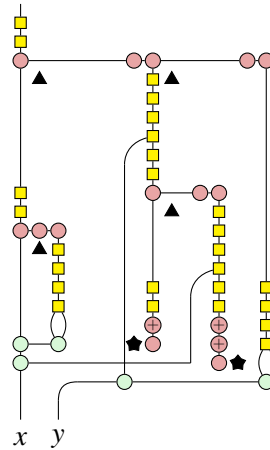
has roots whenever $\varphi_R(x, y)$ is true. To turn this polynomial into a ZH-diagram realizing R , we first realize the map $|x, y\rangle \mapsto |p_R(x, y)\rangle$ via the inductive procedure presented in Lemma 4.2 to get

$$p_R(x, y) = x^2 + x - xy - yx - y + y^2 = y^2 + (-2x - 1)y + (x^2 + x).$$

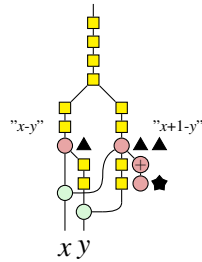
We thus need to start by constructing diagrams for the polynomials $p_1(x) = -2x - 1$ and $p_0(x) = x^2 + x$ (technically, the proof of Lemma 4.2 starts by constructing diagrams for 0-ary polynomials, e.g. constants such as -1 and -2 . For brevity, we inline this step into the construction of p_0 and p_1):



This then leads to the following diagram for p_R



Applying $|x\rangle \mapsto |x^{d-1}\rangle$, post-selecting with $H(0)^T$ and bending up the y -wire then yields a diagram for R . We could now simplify this diagram into something more manageable. Alternatively, we can observe that we do not need to follow the construction of $|x, y\rangle \mapsto |p_R(x, y)\rangle$ given in the proof of Lemma 4.2. Often, the structure of a polynomial allows a much simpler ad-hoc construction based on the gadgets given in (12) and (4). In our case, we get



which is significantly easier to simplify, in particular due to the absence of exponentiator gadgets.

Now, recall our definition of M :

$$|x_0 \dots x_{d-1}\rangle \otimes |c\rangle \mapsto \begin{cases} |x_c\rangle & x_j = 0 \text{ for all } j \neq c \\ 0 & \text{otherwise.} \end{cases}$$

Here, we get the formula

$$\varphi_M(x_0, \dots, x_{d-1}, c, y) = \bigvee_{i=0}^{d-1} \left((c = i) \wedge (y = x_i) \wedge \bigwedge_{\substack{j=0 \\ j \neq i}}^{d-1} (x_j = 0) \right).$$

To translate φ_M into a polynomial, we first need to apply deMorgan's law ($A \wedge B = \neg(\neg A \vee \neg B)$) to turn the conjunctions into disjunctions, to get

$$\varphi_M(x_1, \dots, x_{d-1}, c, y) = \bigvee_{i=0}^{d-1} \neg \left((c \neq i) \vee (y \neq x_i) \vee \bigvee_{\substack{j=0 \\ j \neq i}}^{d-1} (x_j \neq 0) \right)$$

which then translates into the polynomial

$$p_M(x_0, \dots, x_{d-1}, c, y) = \prod_{i=0}^{d-1} \left(1 - \left((1 - (c - i)^{d-1}) \cdot (1 - (y - x_i)^{d-1}) \cdot \prod_{\substack{j=0 \\ j \neq i}}^{d-1} (1 - x_j^{d-1}) \right)^{d-1} \right).$$

We omit the construction of the associated phase-free ZH-diagram, due to the immense size of the resulting diagram.

Moore-Penrose Dagger Categories

Robin Cockett*

Department of Computer Science
University of Calgary, Canada
robin@ucaalgary.ca

Jean-Simon Pacaud Lemay†

School of Mathematical and Physical Sciences
Macquarie University, Australia
js.lemay@mq.edu.au

The notion of a Moore-Penrose inverse (M-P inverse) was introduced by Moore in 1920 and rediscovered by Penrose in 1955. The M-P inverse of a complex matrix is a special type of inverse which is unique, always exists, and can be computed using singular value decomposition. In a series of papers in the 1980s, Puystjens and Robinson studied M-P inverses more abstractly in the context of dagger categories. Despite the fact that dagger categories are now a fundamental notion in categorical quantum mechanics, the notion of a M-P inverse has not (to our knowledge) been revisited since their work. One purpose of this paper is, thus, to renew the study of M-P inverses in dagger categories.

Here we introduce the notion of a Moore-Penrose dagger category and provide many examples including complex matrices, finite Hilbert spaces, dagger groupoids, and inverse categories. We also introduce generalized versions of singular value decomposition, compact singular value decomposition, and polar decomposition for maps in a dagger category, and show how, having such a decomposition is equivalent to having M-P inverses. This allows us to provide precise characterizations of which maps have M-P inverses in a dagger idempotent complete category, a dagger kernel category with dagger biproducts (and negatives), and a dagger category with unique square roots.

1 Introduction

The Moore-Penrose inverse of an $n \times m$ complex matrix A is an $m \times n$ complex matrix A° such that: $AA^\circ A = A$, $A^\circ AA^\circ = A^\circ$, $(AA^\circ)^\dagger = AA^\circ$, and $(A^\circ A)^\dagger = A^\circ A$, where \dagger is the conjugate transpose operator. For any complex matrix, its Moore-Penrose inverse exists, is unique, and can be computed using singular value decomposition – see Example 2.9. The Moore-Penrose inverse is named after E. H. Moore and R. Penrose. Moore first described the notion in 1920 in terms of orthogonal projectors [18]. Without knowing about Moore’s work, in 1955 Penrose described the notion using the identities above [20]. Curious readers can learn more about the fascinating history of the Moore-Penrose inverse and its origin in [1, 4]. Many useful – and quite recent – applications of the Moore-Penrose inverse in mathematics, physics, and computer science are described by Baksalary and Trenkler in [1].

The Moore-Penrose inverse can be generalized to other contexts besides complex matrices. For example, one may consider the Moore-Penrose inverse of a matrix over an involutive ring. While the Moore-Penrose inverse may not always exist, for certain involutive rings it is possible to precisely characterize which matrices have Moore-Penrose inverses. One can also consider Moore-Penrose inverses in involutive semigroups, and in particular in C^* -algebras. It is also possible to define the notion of Moore-Penrose inverses for bounded linear operators between Hilbert spaces, and to characterize precisely which have a Moore-Penrose inverse. Following in this direction, one can in fact define the notion of a Moore-Penrose inverse for maps in *dagger categories*.

Selinger in [27] introduced the term “dagger category”, based on the use in physics of the symbol \dagger for conjugate transpose. Dagger categories are simply categories equipped with an involution on maps

*Partially funded by NSERC.

†Financially supported by a JSPS PDF, Award #: P21746 and a ARC DECRA, Award #: DE230100303

(Def 2.1). In a dagger category, a Moore-Penrose inverse of a map $f : A \rightarrow B$ is a map in the reverse direction $f^\circ : B \rightarrow A$ satisfying the equations above (Def 2.3). The existence and computations of Moore-Penrose inverses for maps in general dagger categories were studied by Puystjens and Robinson in a series of papers in the 1980s [21, 22, 23, 24, 25]. Since Puystjens and Robinson's work, there does not appear to have been any further development of Moore-Penrose inverses in dagger categories. This, despite the fact that the theory of dagger categories itself has undergone significant development. Indeed, in the last decade, dagger categories have become a fundamental component of categorical quantum mechanics (see Heunen and Vicary's introductory level book on the subject [15]). Therefore, it makes perfect sense to revisit Moore-Penrose inverses in the context of dagger categories.

The main objective of this paper is to revisit and renew the study of Moore-Penrose inverses in dagger categories, in the hope that this will lead to new applications in categorical quantum mechanics and elsewhere. We shall apply techniques which have been developed since Puystjens and Robinson's work, such as dagger idempotent splitting and dagger kernels, to Moore-Penrose inverses. We also introduce and study the natural concept of a **Moore-Penrose dagger category**, which is a dagger category where every map has a Moore-Penrose inverse. We provide many examples of Moore-Penrose dagger categories including well-known ones, such as the category of complex matrices or finite-dimensional Hilbert spaces, and also various new ones, such as dagger groupoids and inverse categories.

As was mentioned above, singular value decomposition can be used to compute Moore-Penrose inverses of complex matrices. In Section 4, we introduce a generalized version of singular value decomposition for maps in a dagger category with dagger biproducts. Then, by using dagger kernels, we show how having a generalized singular value decomposition is equivalent to having a Moore-Penrose inverse (Thm 4.9). Another way to compute the Moore-Penrose inverse is by using *compact* singular value decomposition: this is often easier to compute than full singular value decomposition. In Section 3, we introduce a generalized version of compact singular value decomposition for maps in any dagger category and then prove that having a generalized compact singular value decomposition is equivalent to having a Moore-Penrose inverse when dagger idempotents split (Prop 3.9). Therefore, we obtain a precise characterization of maps that have a Moore-Penrose inverse in any dagger idempotent complete category (Thm 3.10). Lastly in Section 5, we give a novel application of Moore-Penrose inverses by introducing the notion of a Moore-Penrose polar decomposition, which captures precisely polar decomposition for complex matrices.

Acknowledgements: The authors would like to thank Chris Heunen for useful discussions and support of this project, as well as thank Ben MacAdam and Cole Comfort for initial discussions on Moore-Penrose inverses and possible relations to restriction categories. The authors would also like to thank Masahito Hasegawa and RIMS at Kyoto University for helping fund research visits so that the authors could work together on this project.

2 Moore-Penrose Inverses

In this section, we discuss Moore-Penrose inverses and some basic properties thereof. In addition, Moore-Penrose dagger categories are introduced and various examples are provided. To set up notation and terminology, we begin by quickly reviewing the basics of dagger categories. For a more in-depth introduction to dagger categories, we refer the reader to [15]. For an arbitrary category \mathbb{X} , we denote objects by capital letters A, B, X, Y , etc. and maps by lowercase letters f, g, h , etc. Identity maps are denoted as $1_A : A \rightarrow A$. Composition is written in *diagrammatic order*, that is, the composition of a map $f : A \rightarrow B$ followed by $g : B \rightarrow C$ is denoted $fg : A \rightarrow C$.

Definition 2.1 [15, Def 2.32] A **dagger** on a category \mathbb{X} is a contravariant functor $(-)^{\dagger} : \mathbb{X} \rightarrow \mathbb{X}$ which is the identity on objects and involutive. We refer to f^{\dagger} as the **adjoint** of f . A **dagger category** is a pair (\mathbb{X}, \dagger) consisting of a category \mathbb{X} equipped with a dagger \dagger .

Concretely, a dagger category can be described as a category \mathbb{X} where for each map $f : A \rightarrow B$, there is a chosen map of dual type $f^{\dagger} : B \rightarrow A$ such that $1_A^{\dagger} = 1_A$, $(fg)^{\dagger} = g^{\dagger}f^{\dagger}$, and $(f^{\dagger})^{\dagger} = f$. Thus, $(-)^{\dagger}$ is a contravariant functor which is, furthermore, an involution – so the adjoint of the adjoint of f is f itself. It is important to note that a category \mathbb{X} can have multiple different daggers. This means that a dagger on a category is structure which must be chosen. Examples of dagger categories can be found below. Here are some special maps in a dagger category:

Definition 2.2 [15, Def 2.34] In a dagger category (\mathbb{X}, \dagger) :

- (i) A map $s : A \rightarrow B$ is an **isometry** if $ss^{\dagger} = 1_A$;
- (ii) A map $r : A \rightarrow B$ is a **coisometry** if $r^{\dagger}r = 1_B$;
- (iii) A map $u : A \rightarrow B$ is a **unitary isomorphism** if $uu^{\dagger} = 1_A$ and $u^{\dagger}u = 1_B$;
- (iv) A map $q : A \rightarrow B$ is a **partial isometry** if $qq^{\dagger}q = q$;
- (v) A map $h : A \rightarrow A$ is **self-adjoint** (or **Hermitian**) if $h^{\dagger} = h$;
- (vi) A map $p : A \rightarrow A$ is **positive** if there exists a map $f : A \rightarrow X$ such that $p = ff^{\dagger}$;
- (vii) A map $e : A \rightarrow A$ is a **\dagger -idempotent** if it self-adjoint and idempotent, that is, $e^{\dagger} = e$ and $ee = e$.

This allows us to define the main concept of interest for this paper:

Definition 2.3 In a dagger category (\mathbb{X}, \dagger) , a **Moore-Penrose inverse** (M-P inverse) of a map $f : A \rightarrow B$ is a map $f^{\circ} : B \rightarrow A$ such that the following equalities hold:

$$[\text{MP.1}] \quad ff^{\circ}f = f \qquad [\text{MP.2}] \quad f^{\circ}ff^{\circ} = f^{\circ} \qquad [\text{MP.3}] \quad (ff^{\circ})^{\dagger} = ff^{\circ} \qquad [\text{MP.4}] \quad (f^{\circ}f)^{\dagger} = f^{\circ}f$$

If f has a M-P inverse, we say that f is **Moore-Penrose invertible** (M-P invertible). A **Moore-Penrose dagger category** is a dagger category such that every map is M-P invertible.

[MP.1] and [MP.2] say that f° is a “regular” inverse of f , while [MP.3] and [MP.4] say that ff° and $f^{\circ}f$ are self-adjoint. This allows us to interpret ff° as the projection of the domain of f , while $f^{\circ}f$ is the projection of the range of f . Examples of Moore-Penrose dagger categories can be found below. However, before looking at examples, we state some basic results for M-P inverses. Most importantly, M-P inverses (if they exist) are unique:

Lemma 2.4 In a dagger category (\mathbb{X}, \dagger) , if a map $f : A \rightarrow B$ has a M-P inverse $f^{\circ} : B \rightarrow A$, then f° is the unique map which satisfies [MP.1] to [MP.4].

PROOF: Suppose that for a map $f : A \rightarrow B$, there exist maps $f^{\circ} : B \rightarrow A$ and $f^{\bullet} : B \rightarrow A$ which are both M-P inverses of f . Then we first compute that:

$$f^{\bullet}f = f^{\bullet}ff^{\circ}f = (f^{\bullet}f)^{\dagger}(f^{\circ}f)^{\dagger} = f^{\dagger}(f^{\bullet})^{\dagger}f^{\dagger}(f^{\circ})^{\dagger} = (ff^{\bullet}f)^{\dagger}(f^{\circ})^{\dagger} = f^{\dagger}(f^{\circ})^{\dagger} = (f^{\circ}f)^{\dagger} = f^{\circ}f.$$

So $f^{\bullet}f = f^{\circ}f$ and, similarly, we can also compute that $ff^{\bullet} = ff^{\circ}$. This allows the observation that:

$$f^{\bullet} = f^{\bullet}ff^{\bullet} = f^{\circ}ff^{\bullet} = f^{\circ}ff^{\circ} = f^{\circ}$$

So $f^\circ = f^\bullet$ and therefore Moore-Penrose inverses are unique. \square

An important consequence of the above lemma is that, for a dagger category, being Moore-Penrose is a property rather than a structure. That said, it is important to note that a map can have a M-P inverse with respect to one dagger structure but fail to have one for another, see Example 2.10. Having a M-P inverse, has a number of consequences:

Lemma 2.5 *In a dagger category (\mathbb{X}, \dagger) , if f has a M-P inverse f° then:*

- (i) f° is also M-P invertible where $f^{\circ\circ} = f$;
- (ii) f^\dagger is also M-P invertible where $f^{\dagger\circ} = f^{\circ\dagger}$;
- (iii) ff° and $f^\circ f$ are \dagger -idempotents and M-P invertible where $(ff^\circ)^\circ = ff^\circ$ and $(f^\circ f)^\circ = f^\circ f$;
- (iv) ff^\dagger and $f^\dagger f$ are M-P invertible where $(ff^\dagger)^\circ = f^{\dagger\circ} f^\circ$ and $(f^\dagger f)^\circ = f^\circ f^{\dagger\circ}$;
- (v) $ff^\circ = f^{\dagger\circ} f^\dagger$ and $f^\circ f = f^\dagger f^{\dagger\circ}$;
- (vi) $f = ff^\dagger f^{\dagger\circ} = f^{\dagger\circ} f^\dagger f$;
- (vii) $f^\circ = f^\circ f^{\dagger\circ} f^\dagger = f^\dagger f^{\dagger\circ} f^\circ$;
- (viii) $f^\dagger = f^\dagger ff^\circ = f^\circ ff^\dagger$;
- (ix) If f is self-adjoint, then f° is also self-adjoint (i.e. $f^{\circ\dagger} = f^\circ$) and $f^\circ f = ff^\circ$;
- (x) If $f^\circ = f^\dagger$, then f is a partial isometry.

PROOF: These are straightforward to check, so we leave them as an exercise for the reader. \square

It is known that computing M-P inverses of complex matrices can be reduced to computing the M-P inverses of Hermitian positive semi-definite matrices. The same is true in dagger categories:

Lemma 2.6 *In a dagger category (\mathbb{X}, \dagger) , for any map $f : A \rightarrow B$, the following are equivalent:*

- (i) f is M-P invertible;
- (ii) $f^\dagger f$ is M-P invertible and $f(f^\dagger f)^\circ f^\dagger f = f$;
- (iii) ff^\dagger is M-P invertible and $ff^\dagger(ff^\dagger)^\circ f = f$

Therefore (\mathbb{X}, \dagger) is Moore-Penrose if and only if every map f satisfies (ii) or (iii).

PROOF: Lemma 2.5.(iv) and (vii) gives us (i) \Rightarrow (ii) and (i) \Rightarrow (iii). Conversely, if $f^\dagger f$ (resp. ff^\dagger) is M-P invertible, then $(f^\dagger f)^\circ f^\dagger$ (resp. $f^\dagger (ff^\dagger)^\circ$) will always satisfy [MP.2], [MP.3], and [MP.4]. The extra assumption that $f(f^\dagger f)^\circ f^\dagger f = f$ (resp. $ff^\dagger(ff^\dagger)^\circ f = f$) is precisely [MP.1]. So we have that f is M-P invertible, giving (ii) \Rightarrow (i) and (iii) \Rightarrow (i). \square

In any dagger category, there are some maps that always have M-P inverses:

Lemma 2.7 *In a dagger category (\mathbb{X}, \dagger) :*

- (i) Identity maps 1_A are M-P invertible where $1_A^\circ = 1_A$;
- (ii) If f is an isomorphism, then f is M-P invertible where $f^\circ = f^{-1}$;
- (iii) If f is a partial isometry or a (co)isometry or unitary, then f is M-P invertible where $f^\circ = f^\dagger$;
- (iv) If e is a \dagger -idempotent, then e is M-P invertible where $e^\circ = e$;

- (v) If p is a positive map such that there exists a M-P invertible map f such that $p = ff^\dagger$, then p is M-P invertible where $p^\circ = f^\circ \dagger f^\circ$, and so p° is also positive;
- (vi) If p is a positive map and M-P invertible, then for any map f such that $p = ff^\dagger$ and $pp^\circ f = f$, f is also M-P invertible where $f^\circ = f^\dagger p^\circ$.

PROOF: These are straightforward to check, so we leave them as an exercise for the reader. □

It is important to note that, in general, Moore-Penrose inverses are not compatible with composition. Indeed, even if f and g have M-P inverses, fg might not have a M-P inverse and, even if it does, $(fg)^\circ$ is not necessarily equal to $g^\circ f^\circ$. Here are some conditions for when $(fg)^\circ = g^\circ f^\circ$ holds:

Lemma 2.8 *In a dagger category (\mathbb{X}, \dagger) , if $f : A \rightarrow B$ and $g : B \rightarrow C$ are M-P invertible then:*

- (i) fg is M-P invertible with $(fg)^\circ = g^\circ f^\circ$ if and only if $f^\circ f g g^\circ$ and $g g^\circ f^\circ f$ are idempotent, and both $f g g^\circ f^\circ = f^\circ \dagger g g^\circ f^\dagger$ and $g^\circ f^\circ f g = g^\dagger f^\circ f g^\circ \dagger$;
- (ii) The following conditions¹ are equivalent and imply $(fg)^\circ = g^\circ f^\circ$:
 - (a) $g g^\circ f^\circ f$, $f g g^\circ f^\circ$ and $g^\circ f^\circ f g$ are self-dual;
 - (b) $g g^\dagger f^\circ f$ and $f^\dagger f g g^\circ$ are self-dual;
 - (c) $f^\circ f g g^\dagger f^\dagger = g g^\dagger f^\dagger$ and $g g^\circ f^\dagger f g = f^\dagger f g$.

PROOF: These can be checked by lengthy and brute-force calculations. □

Here are some examples of Moore-Penrose dagger categories, as well as some non-examples but where we can still fully characterize the M-P invertible maps:

Example 2.9 *Let \mathbb{C} be the field of complex numbers and let $\text{MAT}(\mathbb{C})$ be the category whose objects are natural numbers $n \in \mathbb{N}$ and where a map $A : n \rightarrow m$ is an $n \times m$ complex matrix. $(\text{MAT}(\mathbb{C}), \dagger)$ is a dagger category where \dagger is the conjugate transpose operator, $A^\dagger(i, j) = \overline{A(j, i)}$. Furthermore, $(\text{MAT}(\mathbb{C}), \dagger)$ is also a Moore-Penrose dagger category where the M-P inverse of a matrix can be constructed from its singular value decomposition (SVD). For a $n \times m$ \mathbb{C} -matrix A , let d_1, \dots, d_k be the non-zero singular values of A , so $d_i \in \mathbb{R}$ with $d_i > 0$, and $k \leq \min(n, m)$. Then there exists a unitary $n \times n$ matrix U and a unitary $m \times m$ matrix V such that:*

$$A = U \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}_{n \times m} V^\dagger \quad \text{where } D \text{ is the diagonal } k \times k \text{ matrix } D = \begin{bmatrix} d_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & d_k \end{bmatrix}$$

Then the M-P inverse of A is the $m \times n$ matrix A° defined as follows:

$$A^\circ = V \begin{bmatrix} D^{-1} & 0 \\ 0 & 0 \end{bmatrix}_{m \times n} U^\dagger \quad \text{where } D^{-1} \text{ is the diagonal } k \times k \text{ matrix } D^{-1} = \begin{bmatrix} \frac{1}{d_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \frac{1}{d_k} \end{bmatrix}$$

Since M-P inverses are unique, the construction does not depend on the choice of SVD.

¹For complex matrices, the conditions of (ii) are equivalent to $(fg)^\circ = g^\circ f^\circ$ [5, Sec 1.4 & 1.5]. However, for general dagger categories it appears that the conditions in (ii) are sufficient – but not necessary – to obtain $(fg)^\circ = g^\circ f^\circ$.

Example 2.10 On the other hand, $\text{MAT}(\mathbb{C})$ has another dagger given instead simply by the transpose operator, $A^\top(i, j) = A(j, i)$. However, the dagger category $(\text{MAT}(\mathbb{C}), \top)$ is not Moore-Penrose. For example, the matrix $\begin{bmatrix} i & 1 \end{bmatrix}$ does not have a M-P inverse with respect to the transpose. If it did, one can obtain the contradiction that $i = 0$, which we leave as an exercise for the reader.

Example 2.11 Recall that an involutive ring is a ring R equipped with a unary operation $*$, called the involution, such that $(x + y)^* = x^* + y^*$, and $(xy)^* = y^*x^*$, and $x^{**} = x$. Let $\text{MAT}(R)$ be the category of matrices over R , that is, the category whose objects are natural numbers $n \in \mathbb{N}$ and where a map $A : n \rightarrow m$ is an $n \times m$ matrix A with coefficients in R . Then $(\text{MAT}(R), \dagger)$ is a dagger category where \dagger is given by the involution transpose operator, that is, $A^\dagger(i, j) = A(j, i)^*$. In general $(\text{MAT}(R), \dagger)$ will not necessarily be Moore-Penrose. However, in certain cases, it is possible to precisely characterize which R -matrices do have a M-P inverse. For example, if R is an involutive field, then an R -matrix A has a M-P inverse if and only if $\text{rank}(AA^\dagger) = \text{rank}(A) = \text{rank}(A^\dagger A)$ [19, Thm 1]. Necessary and sufficient conditions for when an R -matrix has a M-P inverse have also been described in the case when R is an integral domain [2], a commutative ring [3], or even a semi-simple artinian ring [17].

Example 2.12 Let HILB be the category of (complex) Hilbert spaces and bounded linear operators between them. Then (HILB, \dagger) is a dagger category where the dagger is given by the adjoint, that is, for a bounded linear operator $f : H_1 \rightarrow H_2$, $f^\dagger : H_2 \rightarrow H_1$ is the unique bounded linear operator such that $\langle f(x)|y \rangle = \langle x|f^\dagger(y) \rangle$ for all $x \in H_1$ and $y \in H_2$. (HILB, \dagger) is not Moore-Penrose but there is a characterization of the M-P invertible maps: a bounded linear map $f : H_1 \rightarrow H_2$, let $\text{Ker}(f) \subseteq H_1$ be its kernel and $\text{im}(f) \subseteq H_2$ be its range, and let $\text{Ker}(f)^\perp$ and $\text{im}(f)^\perp$ be their orthogonal complements. If $\text{im}(f)$ is closed, then we have that $H_2 = \text{im}(f) \oplus \text{im}(f)^\perp$ and also that $f|_{\text{Ker}(f)^\perp} : \text{Ker}(f)^\perp \rightarrow \text{im}(f)$ is a bounded linear isomorphism. Then define the M-P inverse $f^\circ : H_2 \rightarrow H_1$ as $f^\circ(y) = f^{-1}|_{\text{Ker}(f)^\perp}(y)$ for $y \in \text{im}(f)$ and $f^\circ(y) = 0$ for $y \in \text{im}(f)^\perp$. For more details, see [11, Ex 2.16]. Now let FHILB be the subcategory of finite dimensional Hilbert spaces. Then (FHILB, \dagger) is also a dagger category and it is well known that $(\text{FHILB}, \dagger) \simeq (\text{MAT}(\mathbb{C}), \dagger)$. As such, (FHILB, \dagger) is also a Moore-Penrose dagger category where we this time use SVD on linear operators to construct the M-P inverse. So let H_1 be a Hilbert space of dimension n and H_2 a Hilbert space of dimension m . Then for any linear operator $f : H_1 \rightarrow H_2$, if $d_1, \dots, d_k \in \mathbb{R}$ are the non-zero singular values of f (so $k \leq \min(n, m)$), then there exists orthonormal bases $u_i \in H_1$ and $v_j \in H_2$ such that $f(x) = \sum_{i=1}^k d_i \langle u_i|x \rangle v_i$ for all $x \in H_1$. Then $f^\circ : H_2 \rightarrow H_1$ is defined as follows $f^\circ(y) := \sum_{i=1}^k \frac{1}{d_i} \langle v_i|y \rangle u_i$.

Example 2.13 Any field gives a simple example of a Moore-Penrose dagger category. So let k be a field, and let \bullet_k be the category with one object and whose maps are elements of k , where composition is given by the multiplication and the identity map is the unit of k . Then (\bullet_k, \dagger) is a Moore-Penrose dagger category where for all $x \in k$, $x^\dagger = x$ and $x^\circ = x^{-1}$ if $x \neq 0$ or $x^\circ = 0$ if $x = 0$. In fact, a Moore-Penrose dagger category with only one object is precisely a $*$ -regular monoid [8].

Example 2.14 Let REL be the category of sets and relations, that is, the category whose objects are sets and where a map $R : X \rightarrow Y$ is a subset $R \subseteq X \times Y$. (REL, \dagger) is a dagger category where \dagger is given by the converse relation, that is, $(y, x) \in R^\dagger \subseteq Y \times X$ if and only if $(x, y) \in R \subseteq X \times Y$. While (REL, \dagger) is not a Moore-Penrose dagger category, it turns out that the M-P invertible maps are precisely the partial isometries (which recall by Lemma 2.7.(iii) always have M-P inverses). A partial isometry in (REL, \dagger) is a difunctional relation [10, Def 1], which is a relation $R \subseteq X \times Y$ which satisfies that if $(x, b), (a, b)$ and $(a, y) \in R$, then $(x, y) \in R$. It was previously observed that a relation between finite sets has M-P inverse if and only if it was a difunctional relation/partial isometry – since relations between finite sets

correspond to Boolean matrices, and Boolean matrices with M-P inverses were fully characterized in [26, Thm 4.3]. From this, it is not difficult to see that this can be extended to relations between arbitrary sets. Thus, in (REL, \dagger) , $R \subseteq X \times Y$ has a M-P inverse if and only if R is a difunctional relation/partial isometry, which in this case means that the M-P inverse is the converse relations $R^\circ = R^\dagger \subseteq Y \times X$. In fact, the same is true for allegories. Briefly, an **allegory** [9, Chap 2] is a dagger category (\mathbb{X}, \dagger) which is poset enriched and has meets, so in particular each homset $\mathbb{X}(A, B)$ is a poset with order \leq and binary meets \cap , and such that the modular law $fg \cap h \leq (f \cap hg)^\dagger$ holds. Well-known examples of allegories include (REL, \dagger) and more generally the category of relations of a regular category [9, Sec 2.111]. From the modular law, it follows that every map f in an allegory (\mathbb{X}, \dagger) satisfies $f \leq ff^\dagger f$ [9, Sec 2.112]. Therefore, if f has a M-P inverse, using Lemma 2.5.(vii) and (viii), we easily compute that:

$$\begin{aligned} f^\dagger &= f^\circ ff^\dagger \leq f^\circ f^\circ f^\dagger f^\circ ff^\dagger = f^\circ f^\circ f^\dagger f^\dagger = f^\circ \\ f^\circ &= f^\circ f^\circ f^\dagger f^\dagger \leq f^\circ f^\circ f^\dagger f^\dagger ff^\dagger = f^\circ ff^\dagger = f^\dagger. \end{aligned}$$

So we conclude that $f^\circ = f^\dagger$, and so by Lemma 2.5.(x), f is a partial isometry. Thus, a map f in an allegory (\mathbb{X}, \dagger) has a M-P inverse if and only if f is a partial isometry, which means that its M-P inverse is its adjoint $f^\circ = f^\dagger$.

Example 2.15 A **dagger groupoid** is a dagger category (\mathbb{X}, \dagger) where every map in \mathbb{X} is an isomorphism (though not necessarily a unitary). Every dagger groupoid (\mathbb{X}, \dagger) is a Moore-Penrose dagger category where $f^\circ = f^{-1}$. In particular, from any dagger category, we can always construct a dagger groupoid via its subcategory of isomorphisms. So for any category \mathbb{X} , let \mathbb{X}_{iso} be the subcategory of isomorphisms of \mathbb{X} . If (\mathbb{X}, \dagger) is a dagger category, then $(\mathbb{X}_{\text{iso}}, \dagger)$ is a dagger groupoid since if f is an isomorphism, then so is f^\dagger with inverse $f^{\dagger^{-1}} := f^{-1\dagger}$. Therefore $(\mathbb{X}_{\text{iso}}, \dagger)$ is a Moore-Penrose dagger category.

Example 2.16 An **inverse category** [6, Sec 2.3.2] is a dagger category (\mathbb{X}, \dagger) where $ff^\dagger f = f$ for all maps f and $ff^\dagger gg^\dagger = gg^\dagger ff^\dagger$ for all parallel maps f and g . Inverse categories play an important role in the theory of restriction categories [6], since the subcategory of partial isomorphisms of a restriction category is an inverse category. Every inverse category (\mathbb{X}, \dagger) is a Moore-Penrose dagger category where the M-P inverse of f is its adjoint $f^\circ = f^\dagger$ (since every map in an inverse category is a partial isometry by definition). So in particular, for any restriction category, its subcategory of partial isomorphisms is a Moore-Penrose dagger category. As a concrete example, let PINJ be the category of sets and partial injections, which is the subcategory of partial isomorphisms of the restriction category of sets and partial functions. Then (PINJ, \dagger) is an inverse category where for a partial injection $f : X \rightarrow Y$, $f^\dagger : Y \rightarrow X$ is defined as $f^\dagger(y) = x$ if $f(x) = y$ and is undefined otherwise.

Example 2.17 If $(\mathbb{X}_1, \dagger_1)$ and $(\mathbb{X}_2, \dagger_2)$ are both Moore-Penrose dagger categories, then their product $(\mathbb{X}_1 \times \mathbb{X}_2, \dagger_1 \times \dagger_2)$ is also a Moore-Penrose dagger category. In particular, we can combine Example 2.13 and Example 2.16. So if (\mathbb{X}, \dagger) is an inverse category and k is a field, let \mathbb{X}_k be the category whose objects are those of \mathbb{X} but whose maps are pairs (f, x) consisting of a map f in \mathbb{X} and an element $x \in k$, so we may think of x as adding a weight or a cost to f . Then (\mathbb{X}_k, \dagger) is a Moore-Penrose dagger category where $(f, x)^\dagger = (f^\dagger, x)$ and $(f, x)^\circ = (f^\dagger, x^\circ)$.

3 Compact Singular Value Decomposition

In Example 2.9, we explained how to construct the M-P inverse of a complex matrix using SVD. However, there is an alternative way to construct the M-P inverse using *compact* singular value decomposition

(CSVD). This decomposition tells us that for any $n \times m$ complex matrix, A , again with singular values d_1, \dots, d_k and associated diagonal matrix D , there exists an $n \times k$ matrix R and an $m \times k$ matrix S such that $A = RDS^\dagger$ and $R^\dagger R = S^\dagger S = I_k$. The decomposition allows one to construct the M-P inverse as $A^\circ := SD^{-1}R^\dagger$. In dagger categorical terms, R and S are coisometries, and D is an isomorphism². Thus, generalized CSVD in an arbitrary dagger category is a factorization into a coisometry, followed by an isomorphism, followed by an isometry. We shall discuss the generalization of CSVD for dagger categories before discussing SVD because generalizing SVD requires dagger biproducts and dagger kernels, while generalizing CSVD can be explained without introducing further structure.

This generalized CSVD not only provides a simple way of computing M-P inverses, but is also directly related to the splitting of dagger idempotents, an important dagger category concept that was introduced by Selinger in [28]. Generalized CSVD allows us to precisely characterize the M-P invertible maps in dagger categories which are dagger idempotent complete. Furthermore, the dagger idempotent splitting completion leads us to an important reinterpretation of M-P inverses as being *actual* inverses between dagger idempotents. As such, we begin this section by discussing the relationship between M-P inverses and dagger idempotent splitting.

Definition 3.1 [28, Def 3.6] *In a dagger category (\mathbb{X}, \dagger) , a **dagger idempotent** $e : A \rightarrow A$ is an idempotent which is self-adjoint, $ee = e = e^\dagger$. A dagger idempotent is said to **\dagger -split** if there exists a map $r : A \rightarrow X$ such that $rr^\dagger = e$ and $r^\dagger r = 1_X$ (so r is a coisometry). A **dagger idempotent complete category** is a dagger category (\mathbb{X}, \dagger) such that all \dagger -idempotents \dagger -split.*

In Lemma 2.5.(iii), we saw that in any dagger category (\mathbb{X}, \dagger) , if a map f has a M-P inverse, then ff° and $f^\circ f$ were both \dagger -idempotents. As such, we may ask these \dagger -idempotents to also be \dagger -split:

Definition 3.2 *In a dagger category (\mathbb{X}, \dagger) , a map f is **Moore-Penrose split** (M-P split) if f has a M-P inverse f° and the \dagger -idempotents ff° and $f^\circ f$ \dagger -split. A Moore-Penrose category in which all maps are M-P split is said to be **Moore-Penrose complete**.*

A dagger category which is Moore-Penrose complete is the same thing as a Moore-Penrose category in which *all* dagger idempotents split:

Proposition 3.3 *A dagger category (\mathbb{X}, \dagger) is Moore-Penrose complete if and only if (\mathbb{X}, \dagger) is dagger idempotent complete and Moore-Penrose.*

PROOF: The \Leftarrow direction is immediate by definition. For the \Rightarrow direction, suppose that (\mathbb{X}, \dagger) is Moore-Penrose complete. By definition, this means every map has a M-P inverse, so (\mathbb{X}, \dagger) is indeed Moore-Penrose. Now let $e : A \rightarrow A$ be a \dagger -idempotent. By Lemma 2.7.(iv), e is its own M-P inverse, so $e^\circ = e$, and therefore $e^\circ e = e = ee^\circ$. However, by assumption, e is M-P split, which therefore implies that e is \dagger -split. So (\mathbb{X}, \dagger) is indeed \dagger -idempotent complete. \square

We will now explain how every Moore-Penrose dagger category embeds into a Moore-Penrose complete dagger category. Let us first review how every dagger category embeds into a dagger idempotent complete category via the dagger version of the idempotent splitting completion, also called the dagger Karoubi envelope [28, Def 3.13]. So for a dagger category (\mathbb{X}, \dagger) , define the dagger category $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ whose objects are pairs (A, e) consisting of an object A and a \dagger -idempotent $e : A \rightarrow A$ in (\mathbb{X}, \dagger) , and whose maps $f : (A_1, e_1) \rightarrow (A_2, e_2)$ in are maps $f : A_1 \rightarrow A_2$ in \mathbb{X} such that $e_1 f e_2 = f$ (or equivalently $e_1 f = f = f e_2$). Composition in $\text{Split}_\dagger(\mathbb{X})$ is defined as in \mathbb{X} , while identity maps

²The fact that D is a diagonal matrix of singular values is not relevant to this way of constructing the M-P inverse.

$1_{(A,e)} : (A,e) \rightarrow (A,e)$ are defined as $1_{(A,e)} := e$. Lastly, the dagger of $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ is defined as in (\mathbb{X}, \dagger) , and furthermore $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ is a dagger idempotent complete category [28, Prop 3.12]. There is also an embedding $\mathcal{S} : (\mathbb{X}, \dagger) \rightarrow (\text{Split}_\dagger(\mathbb{X}), \dagger)$ which is defined on objects as $\mathcal{S}(A) = (A, 1_A)$ and on maps as $\mathcal{S}(f) = f$.

Lemma 3.4 *Let (\mathbb{X}, \dagger) be a Moore-Penrose dagger category. Then $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ is a Moore-Penrose complete category.*

PROOF: Let $f : (A,e) \rightarrow (B,e')$ be a map in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Since composition and the dagger of $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ are the same as in (\mathbb{X}, \dagger) , it suffices to show that $f^\circ : B \rightarrow A$ is also a map of type $(B,e') \rightarrow (A,e)$ in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. So we must show that $e'f^\circ e = f^\circ$. To do so we use Lemma 2.5.(vii) and that $f^\dagger : (A_2, e_2) \rightarrow (A_1, e_1)$ is also a map in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$:

$$e'f^\circ e = e'f^\circ f^{\dagger\circ} f^\dagger e = e'f^\circ f^{\dagger\circ} f^\dagger = e'f^\circ = e'f^\dagger f^{\dagger\circ} f^\circ = f^\dagger f^{\dagger\circ} f^\circ = f^\circ$$

So $f^\circ : (B,e') \rightarrow (A,e)$ is a map in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. □

We are now ready to discuss a generalization of CSVD in an arbitrary dagger category, and show that having a generalized CSVD is equivalent to being M-P split.

Definition 3.5 *In a dagger category, a **generalized compact singular value decomposition** (GCSVD) of a map $f : A \rightarrow B$ is a triple $(r : A \rightarrow X, d : X \rightarrow Y, s : Y \rightarrow B)$, where r is a coisometry, d is an isomorphism, and s is an isometry, such that $f = rds$.*

Lemma 3.6 *In a dagger category (\mathbb{X}, \dagger) , if the two triples $(r_1 : A \rightarrow X_1, d_1 : X_1 \rightarrow Y_1, s_1 : Y_1 \rightarrow B)$ and $(r_2 : A \rightarrow X_2, d_2 : X_2 \rightarrow Y_2, s_2 : Y_2 \rightarrow B)$ are GCSVDs of $f : A \rightarrow B$, then there exist unique unitary maps $u : X_1 \rightarrow X_2$ and $v : Y_1 \rightarrow Y_2$ such that $r_1 u = r_2$, $d_1 v = u d_2$, and $s_1 = v s_2$.*

PROOF: Define u and v as the composites, $u := r_1^\dagger r_2$ and $v := s_1 s_2^\dagger$. The necessary identities are checked via some straightforward diagram chasing. □

In order to show that having a GCSVD is equivalent to being M-P split, it will be useful to first observe that maps with M-P inverses in the base dagger category are actual isomorphisms in the dagger idempotent splitting completion:

Lemma 3.7 *A map $f : A \rightarrow B$ in a dagger category (\mathbb{X}, \dagger) has a M-P inverse if and only if there exists \dagger -idempotents $e_1 : A \rightarrow A$ and $e_2 : B \rightarrow B$ such that $f : (A, e_1) \rightarrow (B, e_2)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Explicitly:*

- (i) *If $f : A \rightarrow B$ has a M-P inverse $f^\circ : B \rightarrow A$, then $f : (A, f f^\circ) \rightarrow (A, f^\circ f)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ with inverse $f^\circ : (A, f^\circ f) \rightarrow (A, f f^\circ)$;*
- (ii) *If $f : (A, e_1) \rightarrow (B, e_2)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ with inverse $f^\circ : (B, e_2) \rightarrow (A, e_1)$, then f is M-P invertible in (\mathbb{X}, \dagger) with M-P inverse f° .*

PROOF: To start, let us explicitly spell out what it means for $f : (A, e_1) \rightarrow (B, e_2)$ to be an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Firstly, we need that $e_1 f e_2 = f$ (or equivalently $e_1 f = f = f e_2$). Secondly, we also need a map $g : (B, e_2) \rightarrow (A, e_1)$ in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$, so $e_2 g e_1 = g$ (or equivalently $e_2 g = g = g e_1$), and such that $f g = 1_{(A, e_1)} = e_1$ and $g f = 1_{(B, e_2)} = e_2$.

Suppose that $f : A \rightarrow B$ has a M-P inverse $f^\circ : B \rightarrow A$. By Lemma 2.5.(iii), $(A, f f^\circ)$ and $(A, f^\circ f)$ are well-defined objects in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. On the other hand, by [MP.1] and [MP.2], it is easy to check that

$f : (A, ff^\circ) \rightarrow (A, f^\circ f)$ and $f^\circ : (A, f^\circ f) \rightarrow (A, ff^\circ)$ are well-defined maps in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Lastly, by definition we have that $ff^\circ = 1_{(A, ff^\circ)}$ and $f^\circ f = 1_{(A, f^\circ f)}$. Thus we conclude that $f : (A, ff^\circ) \rightarrow (A, f^\circ f)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$.

Conversely, suppose that $f : (A, e_1) \rightarrow (B, e_2)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ with inverse $f^\circ : (B, e_2) \rightarrow (A, e_1)$. In particular, this implies that $ff^\circ = e_2$ and $f^\circ f = e_1$. So ff° and $f^\circ f$ are both \dagger -idempotents, thus [MP.3] and [MP.4] hold. By the assumed properties of maps in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$, we have that $ff^\circ f = e_1 f = f$ and $f^\circ f f^\circ = e_2 f^\circ = f^\circ$, and so [MP.1] and [MP.2] hold. Therefore, we conclude that f is M-P invertible with M-P inverse f° . \square

Corollary 3.8 *A map $f : A \rightarrow B$ in a dagger category (\mathbb{X}, \dagger) is M-P split if and only if there exists \dagger -split \dagger -idempotents $e_1 : A \rightarrow A$ and $e_2 : B \rightarrow B$ such that $f : (A, e_1) \rightarrow (B, e_2)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$.*

PROOF: Suppose that $f : A \rightarrow B$ is M-P split with M-P inverse $f^\circ : B \rightarrow A$. By definition ff° and $f^\circ f$ are \dagger -split \dagger -idempotents and by Lemma 3.7, $f : (A, ff^\circ) \rightarrow (A, f^\circ f)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Conversely, suppose that $e_1 : A \rightarrow A$ and $e_2 : B \rightarrow B$ are \dagger -idempotents that \dagger -split via the coisometry $r : A \rightarrow X$ and isometry $s : Y \rightarrow B$ respectively, and also that $f : (A, e_1) \rightarrow (B, e_2)$ is an isomorphism in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$ with inverse $f^\circ : (B, e_2) \rightarrow (A, e_1)$. Then by Lemma 3.7, f° is the M-P inverse of f , and by assumption we also have that $ff^\circ = e_1$ and $f^\circ f = e_2$. So ff° and $f^\circ f$ are \dagger -split, and therefore we conclude that f is M-P split. \square

We may now state the main result of this section:

Proposition 3.9 *In a dagger category (\mathbb{X}, \dagger) , a map f has a GCSVD if and only if f is M-P split.*

PROOF: Suppose that $f : A \rightarrow B$ has a GCSVD ($r : A \rightarrow X, d : X \rightarrow Y, s : Y \rightarrow B$). Define $f^\circ := s^\dagger d^{-1} r^\dagger$. First note that rr^\dagger and $s^\dagger s$ are \dagger -split \dagger -idempotents, so (A, rr^\dagger) and $(B, s^\dagger s)$ are well-defined objects in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. We then compute that:

$$\begin{aligned} rr^\dagger fs^\dagger s &= rr^\dagger rds s^\dagger s = rds = f \\ s^\dagger s f^\circ rr^\dagger &= s^\dagger s s^\dagger d^{-1} r^\dagger rr^\dagger = s^\dagger d^{-1} r^\dagger = f^\circ \end{aligned}$$

So $f : (A, rr^\dagger) \rightarrow (B, s^\dagger s)$ and $f^\circ : (B, s^\dagger s) \rightarrow (A, rr^\dagger)$ are maps in $(\text{Split}_\dagger(\mathbb{X}), \dagger)$. Furthermore, we can also compute that:

$$\begin{aligned} ff^\circ &= rds s^\dagger d^{-1} r^\dagger = rdd^{-1} r^\dagger = rr^\dagger = 1_{(A, rr^\dagger)} \\ f^\circ f &= s^\dagger d^{-1} r^\dagger rds = s^\dagger d^{-1} ds = s^\dagger s = 1_{(B, s^\dagger s)} \end{aligned}$$

Therefore, $f : (A, ff^\circ) \rightarrow (A, f^\circ f)$ is an isomorphism with inverse $f^\circ : (B, s^\dagger s) \rightarrow (A, rr^\dagger)$. So by Corollary 3.8, f is M-P split with M-P inverse f° .

Conversely, suppose that $f : A \rightarrow B$ is M-P split, where ff° and $f^\circ f$ both \dagger -split via, respectively, the coisometry $r : A \rightarrow X$ and isometry $s : Y \rightarrow B$. Now define $d : X \rightarrow Y$ as the composite $d := r^\dagger f s^\dagger$. We then immediately have $f = rds$. So it remains to show that d is an isomorphism. So define $d^{-1} : Y \rightarrow X$ as the composite $d^{-1} = s^\dagger f^\circ r$. We compute that:

$$\begin{aligned} dd^{-1} &= r^\dagger f s^\dagger s f^\circ r = r^\dagger f f^\circ f s^\dagger r = r^\dagger r r^\dagger r r^\dagger r = 1_X \\ d^{-1} d &= s f^\circ r r^\dagger f s^\dagger = s f^\circ f f^\circ f s^\dagger = s s^\dagger s s^\dagger s s^\dagger = 1_Y \end{aligned}$$

Therefore, $(r : A \rightarrow X, d : X \rightarrow Y, s : Y \rightarrow B)$ is a CSVD of f . \square

We can now precisely characterize M-P invertible maps in a dagger idempotent complete category:

Theorem 3.10 *In a dagger idempotent complete category, a map is M-P invertible if and only if it has a GCSVD.*

Corollary 3.11 *A dagger category is Moore-Penrose complete if and only if every map has a GCSVD.*

Observe that Lemma 3.4 tells us that every Moore-Penrose dagger category embeds into a dagger category where every map has a GCSVD.

4 Singular Value Decomposition

The objective of this section is to generalize SVD for maps in a dagger category in such a way that we may compute M-P inverses in the same way that was done in Example 2.9. So generalized SVD can be described as a special factorization in terms of two unitaries and an isomorphism. However, in order to describe the middle component as a square matrix with the isomorphism in the top corner and zeroes everywhere else, we need to work in a setting with *dagger biproducts*. It is worth mentioning that in [22], Puystjens and Robinson do discuss how the existence of a M-P inverse for a map with an epic-monic factorization is essentially equivalent to a factorization via dagger biproducts. Here, we drop the epic-monic factorization requirement, which allows us to provide a story of how M-P inverses are equivalent to a dagger biproduct factorization which more closely resembles the generalized version of SVD.

Let us begin by quickly recalling the definition of dagger biproducts. For a refresher on biproducts and zero objects, we refer the reader to [15, Chap 2]. So for category \mathbb{X} that has finite biproducts, we denote the biproduct as \oplus , the projections as $\pi_j : A_1 \oplus \dots \oplus A_n \rightarrow A_j$, the injections as $\iota_j : A_j \rightarrow A_1 \oplus \dots \oplus A_n$, the zero object as 0 , the sum of maps as $f + g$, and lastly the zero maps as 0 .

Definition 4.1 [15, Def 2.39] *A dagger category (\mathbb{X}, \dagger) has **finite \dagger -biproducts** if \mathbb{X} has finite biproducts such that the adjoints of the projections are the injections, that is, $\pi_j^\dagger = \iota_j$.*

Using dagger biproducts, we may now introduce generalized SVD:

Definition 4.2 *In a dagger category (\mathbb{X}, \dagger) with finite \dagger -biproducts, a **generalized singular value decomposition (GSVD)** of a map $f : A \rightarrow B$ is a triple of maps $(u : A \rightarrow X \oplus Z, d : X \rightarrow Y, v : Y \oplus W \rightarrow B)$ such that u and v are unitary and d is an isomorphism, and such that $f = u(d \oplus 0)v$.*

Lemma 4.3 *In a dagger category (\mathbb{X}, \dagger) with finite \dagger -biproducts, if for a map $f : A \rightarrow B$, we have that $(u_1 : A \rightarrow X_1 \oplus Z_1, d : X_1 \rightarrow Y_1, v_1 : Y_1 \oplus W_1 \rightarrow B)$ and $(u_2 : A \rightarrow X_2 \oplus Z_2, d : X_2 \rightarrow Y_2, v_2 : Y_2 \oplus W_2 \rightarrow B)$ are both GSVDs of f , then there exists unique unitary maps $x : X_1 \rightarrow X_2$, $y : Y_1 \rightarrow Y_2$, $z : Z_1 \rightarrow Z_2$, and $w : W_1 \rightarrow W_2$ such that $u_1(x \oplus z) = u_2$, $d_1 y = x d_2$, and $v_1 = (y \oplus w)v_2$.*

PROOF: Define x , y , z , and w as the composites $x := \iota_1 u_1^\dagger u_2 \pi_1$, $y := \iota_1 v_1 v_2^\dagger \pi_1$, $z := \iota_2 u_1^\dagger u_2 \pi_2$, and lastly $w := \iota_2 v_1 v_2^\dagger \pi_2$. By straightforward diagram chasing, one can check all the necessary identities. \square

We will explain below why this recaptures precisely SVD for complex matrices. We first observe that every GSVD induces a GCSVD. Therefore by applying the results of the previous section, having a GSVD implies that we have a M-P inverse:

Proposition 4.4 *In a dagger category (\mathbb{X}, \dagger) with \dagger -biproducts, suppose that a map $f : A \rightarrow B$ has a GSVD $(u : A \rightarrow X \oplus Z, d : X \rightarrow Y, v : Y \oplus W \rightarrow B)$. Then $(u\pi_1 : A \rightarrow X, d : X \rightarrow Y, v\iota_1 : Y \rightarrow B)$ is a GCSVD of f , and therefore f is M-P split where $f^\circ := v^\dagger(d^{-1} \oplus 0)u^\dagger$.*

PROOF: A unitary composed with a (co)isometry is always a (co)isometry. So $u\pi_1$ is a coisometry and $\iota_1 v$ is an isometry. Next, note that the \dagger -biproduct structure gives us that $a \oplus b = \pi_1 a \iota_1 + \pi_2 a \iota_2$. So in our case, we have that $d \oplus 0 = \pi_1 d \iota_1$. Therefore, we have that $f = u\pi_1 d \iota_1 v$. So we conclude that $(u\pi_1, d, \iota_1 v)$ is a GCSVD of f . Applying Proposition 3.9 we get that $f^\circ := v^\dagger \pi_1 d^{-1} \iota_1 u^\dagger$, which can alternatively be written as $f^\circ := v^\dagger (d^{-1} \oplus 0) u^\dagger$. \square

Let us explain how GSVD does indeed generalize how SVD is used to compute M-P inverses for matrices. As explained in [15, Sec 2.2.4], in a dagger category with finite dagger biproducts, a map $F : A_1 \oplus \dots \oplus A_n \rightarrow B_1 \oplus \dots \oplus B_m$ is uniquely determined by a family of maps $f_{i,j} : A_i \rightarrow B_j$. Therefore F can be represented as a $n \times m$ matrix where the term in the i -th row and j -th column is $f_{i,j}$. So if f has a GSVD (u, d, v) , we may expand $d \oplus 0$ as a 2×2 matrix, and therefore write f and f° as:

$$f = u \begin{bmatrix} d & 0 \\ 0 & 0 \end{bmatrix} v \quad f^\circ = v^\dagger \begin{bmatrix} d^{-1} & 0 \\ 0 & 0 \end{bmatrix} u^\dagger$$

which recaptures precisely how M-P inverses were constructed using SVD in Example 2.9. We now wish to go in the other direction, that is, going from a M-P inverse to a GSVD. To do so, we will need to use dagger kernels. For a refresher on ordinary kernels, we refer the reader to [15, Sec 2.4.2].

Definition 4.5 [12, Def 2.1] *In a dagger category (\mathbb{X}, \dagger) with a zero object, a map $f : A \rightarrow B$ has a \dagger -kernel if f has a kernel $k : \ker(f) \rightarrow A$ such that k is an isometry. A **dagger kernel category** is a dagger category with a zero object such that every map has a dagger kernel.*

In [24], Puystjens and Robinson describe many necessary and sufficient conditions for when a map that has a kernel has a M-P inverse in a dagger category which is enriched over Abelian groups. However, dagger kernels are not discussed in [24]. Therefore, one could specialize certain results in [24] for dagger kernels instead. In this paper, we will show that having a M-P inverse *and* a dagger kernel is equivalent to having a GSVD. Also note that, unlike in [24], we do not assume that we are working in a setting with negatives (i.e. additive inverses). Because of this, the statement does require a modest extra compatibility condition between the M-P inverse and the dagger kernel.

Proposition 4.6 *In a dagger category (\mathbb{X}, \dagger) with \dagger -biproducts, a map f has a GSVD if and only if f is M-P split and f has a \dagger -kernel $k : \ker(f) \rightarrow A$ and f^\dagger has a \dagger -kernel $c : \ker(f^\dagger) \rightarrow B$ such that $f f^\circ + k^\dagger k = 1_A$ and $f^\circ f + c^\dagger c = 1_B$.*

PROOF: Suppose that $f : A \rightarrow B$ has a GSVD $(u : A \rightarrow X \oplus Z, d : X \rightarrow Y, v : Y \oplus W \rightarrow B)$. We have already explained why f is M-P split in the above lemma. Using the \dagger -biproduct identity that $\iota_i \pi_j = 0$ if $i \neq j$ and $\iota_j \pi_j = 1$, it is straightforward to check that $\iota_2 u^\dagger : Z \rightarrow A$ is a \dagger -kernel of f and that $\iota_2 v : W \rightarrow B$ is a \dagger -kernel of f^\dagger . For the extra identities, we first note that $f f^\circ = u(1_X \oplus 0)u^\dagger$ and $f^\circ f = v^\dagger(1_Y \oplus 0)v$, which we can alternatively write as $f f^\circ = u\pi_1 \iota_1 u^\dagger$ and $f^\circ f = v^\dagger \pi_1 \iota_1 v$. Then using the other \dagger -biproduct identity that $\pi_1 \iota_1 + \pi_2 \iota_2 = 1$, it follows that $f f^\circ + k^\dagger k = 1_A$ and $f^\circ f + c^\dagger c = 1_B$ as desired.

Conversely, suppose that f is M-P split, and has a \dagger -kernel $k : \ker(f) \rightarrow A$ and f^\dagger has a \dagger -kernel $c : \ker(f^\dagger) \rightarrow B$ such that the two equalities $f f^\circ + k^\dagger k = 1_A$ and $f^\circ f + c^\dagger c = 1_B$ also hold. Then by Prop 3.9, f also has a GCSVD $(r : A \rightarrow X, d : X \rightarrow Y, s : Y \rightarrow B)$, so, in particular, $f = rds$ and d is an isomorphism. Then, using matrix notation, define $u : A \rightarrow X \oplus \ker(f)$ and $v : Y \oplus \ker(f^\dagger) \rightarrow B$ respectively as $u := \begin{bmatrix} r & k^\dagger \end{bmatrix}$ and $v := \begin{bmatrix} s \\ c \end{bmatrix}$. We first compute that:

$$u(d \oplus 0)v = \begin{bmatrix} r & k^\dagger \end{bmatrix} \begin{bmatrix} d & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ c \end{bmatrix} = \begin{bmatrix} rd & 0 \end{bmatrix} \begin{bmatrix} s \\ c \end{bmatrix} = rds = f$$

So $f = u(d \oplus 0)v$ as desired. We must also show that u and v are unitary. To show that u is unitary, recall that $rr^\dagger = f^\circ f$ and also that since $krds = kf = 0$, it follows that $kr = 0$. Therefore we compute:

$$uu^\dagger = \begin{bmatrix} r & k^\dagger \end{bmatrix} \begin{bmatrix} r & k^\dagger \end{bmatrix}^\dagger = \begin{bmatrix} r & k^\dagger \end{bmatrix} \begin{bmatrix} r^\dagger \\ k \end{bmatrix} = rr^\dagger + k^\dagger k = ff^\circ + k^\dagger k = 1_A$$

$$u^\dagger u = \begin{bmatrix} r^\dagger \\ k \end{bmatrix} \begin{bmatrix} r & k^\dagger \end{bmatrix} = \begin{bmatrix} r^\dagger r & r^\dagger k^\dagger \\ kr & kk^\dagger \end{bmatrix} = \begin{bmatrix} 1_A & 0 \\ 0 & 1_{\ker(f)} \end{bmatrix} = 1_{X \oplus \ker(f)}$$

So u is unitary. Similarly, we can show that v is unitary. So we conclude that (u, d, v) is a GSVD of f . \square

Corollary 4.7 *In a dagger category (\mathbb{X}, \dagger) with finite \dagger -biproducts and negatives, a map f has a GSVD if and only if f is M-P split and both f and f^\dagger have \dagger -kernels.*

PROOF: We need only show that $ff^\circ + k^\dagger k = 1_A$ and $f^\circ f + c^\dagger c = 1_B$. First note that $(1_A - ff^\circ)f = 0$ and $(1_B - f^\circ f)f^\dagger = 0$ (the latter of which is by Lemma 2.5.(viii)). So by universal property of the kernel, there exist unique maps z_1 and z_2 such that $z_1 k = 1_A - ff^\circ$ and $z_2 c = 1_B - f^\circ f$. By post-composing by k^\dagger and c^\dagger respectively, and also by using that $f^\circ k^\dagger = 0$ (which follows from Lemma 2.5.(vii)) and $fc^\dagger = 0$, we then obtain that $z_1 = k^\dagger$ and $z_2 = c^\dagger$. Therefore, $k^\dagger k = 1_A - ff^\circ$ and $c^\dagger c = 1_B - f^\circ f$, which in turn implies the desired equalities. \square

Therefore, in a setting with negatives and all dagger kernels, we may state that:

Corollary 4.8 *In a dagger kernel category (\mathbb{X}, \dagger) with finite \dagger -biproducts and negatives, a map f has a GSVD if and only if f is M-P split.*

Finally, assuming also that we are in a dagger idempotent complete setting, we obtain a precise characterization of M-P invertible maps in terms of a generalized version of SVD:

Theorem 4.9 *In a dagger kernel category (\mathbb{X}, \dagger) that is \dagger -idempotent complete and which has finite \dagger -biproducts and negatives, a map f is M-P invertible if and only if f has a GSVD.*

5 Polar Decomposition

It is straightforward to give a generalized version of polar decomposition (PD) in a dagger category, it is the statement that a map factorizes as a partial isometry followed by a positive map. However, the statement of PD for bounded linear maps between Hilbert spaces is stronger: it also involves a requirement on the kernel (or range) of the partial isometry. In [16, Thm 8.3], Higham nicely explains how M-P inverses can play a role in the PD of complex matrices and can be used to replace that extra requirement. So recall that for an $n \times m$ complex matrix, A , there exists a unique partial isometry U and unique a positive semi-definite Hermitian matrix H such that $A = UH$ and $\text{range}(U^\dagger) = \text{range}(H)$. The matrix H is given by the square root of the matrix $A^\dagger A$, so $H = (A^\dagger A)^{\frac{1}{2}}$, while the matrix U is constructed using the M-P inverse of H , so $U = AH^\circ$. Furthermore, the condition $\text{range}(U^*) = \text{range}(H)$ can be equivalently described in terms of M-P inverses as the equality $U^\dagger U = HH^\circ$ (where note that $U^\circ = U^\dagger$ since U is a partial isometry). Therefore PD of complex matrices can be completely expressed in terms of M-P inverses. As such in this section, we introduce the notion of a Moore-Penrose polar decomposition of maps in an arbitrary dagger category, which recaptures precisely PD for complex matrices.

Definition 5.1 *In a dagger category (\mathbb{X}, \dagger) , for a map $f : A \rightarrow B$,*

- (i) A **generalized polar decomposition (GPD)** of f is a pair of maps $(u : A \rightarrow B, h : B \rightarrow B)$ where u is a partial isometry and h is a positive map such that $f = uh$;
- (ii) A **Moore-Penrose polar decomposition (M-P PD)** of f is a GPD $(u : A \rightarrow B, h : B \rightarrow B)$ of f such that h is M-P invertible and $u^\dagger u = hh^\circ$.

We will show that for f to have a M-P PD is equivalent to requiring that f be M-P invertible and $f^\dagger f$ has a square-root. The following definition is a Moore-Penrose version of Selinger's definition [28, Def 5.13].

Definition 5.2 In a dagger category (\mathbb{X}, \dagger) , a M-P invertible positive map $p : A \rightarrow A$ has a **Moore-Penrose square root (M-P square root)** if there exists a M-P invertible positive map $\sqrt{p} : A \rightarrow A$ such that $\sqrt{p}\sqrt{p} = p$. A dagger category is said to have **(unique) M-P square roots** if all M-P invertible positive maps have a (unique) M-P square root.

Proposition 5.3 In a dagger category (\mathbb{X}, \dagger) , a map f has a M-P PD if and only if f is M-P invertible and $f^\dagger f$ has a M-P square root.

PROOF: Suppose that $(u : A \rightarrow B, h : B \rightarrow B)$ is a M-P PD of a map $f : A \rightarrow B$. Since u is a partial isometry, u is M-P invertible where $u^\circ = u^\dagger$. We also have that since h is positive, it is self-dual $h^\dagger = h$, and by Lemma 2.5.(ix) we have that $h^\circ h = hh^\circ$. Therefore by the assumption that $u^\dagger u = hh^\circ$, it easily follows that $u^\dagger uhh^\circ = hh^\circ u^\dagger u$. Therefore by Lemma 2.8.(ii), we have that $f = uh$ is M-P invertible whose M-P inverse is $f^\circ = (uh)^\circ = h^\circ u^\dagger$. By Lemma 2.5.(iv), $f^\dagger f$ is a M-P invertible positive map. So it remains to compute that:

$$hh = hhh^\circ h = hu^\dagger uh = (uh)^\dagger uh = f^\dagger f$$

So $hh = f^\dagger f$, and therefore h is a M-P square root of $f^\dagger f$.

Conversely, suppose that f is M-P invertible and $f^\dagger f$ has a M-P square root $\sqrt{f^\dagger f}$. So define $h : B \rightarrow B$ as $h := \sqrt{f^\dagger f}$, and define $u : A \rightarrow B$ as the composite $u := fh^\circ$. We then compute the following:

$$\begin{aligned} uu^\dagger u &= fh^\circ (fh^\circ)^\dagger fh^\circ = fh^\circ h^\circ f^\dagger fh^\circ = fh^\circ h^\circ hhh^\circ = fh^\circ hh^\circ hh^\circ = fh^\circ hh^\circ = fh^\circ = u \\ u^\dagger u &= (fh^\circ)^\dagger fh^\circ = h^\circ f^\dagger fh^\circ = h^\circ hhh^\circ = hh^\circ hh^\circ = hh^\circ \\ uh &= fh^\circ h = fh^\circ hh^\circ h = fh^\circ h^\circ hh = f(hh)^\circ hh = f(f^\dagger f)^\circ f^\dagger f = ff^\circ f = f \end{aligned}$$

So u is an isometry, $u^\dagger u = hh^\circ$, and $f = uh$. So we conclude that (u, h) is a M-P PD of f . \square

Corollary 5.4 In a dagger category (\mathbb{X}, \dagger) with M-P square roots, a map f is M-P invertible if and only if f has a M-P PD.

Unlike PD which is always unique, M-P PD is not necessarily unique in an arbitrary dagger category. The reason PD is unique is due to the fact that positive semi-definite Hermitian matrices have unique square roots. Therefore, if we work in a dagger category where the positive maps do have unique square roots, then M-P PD is also unique as desired.

Lemma 5.5 In a dagger category (\mathbb{X}, \dagger) with unique M-P square roots, M-P PDs are unique.

PROOF: Suppose that $(u : A \rightarrow B, h : B \rightarrow B)$ and $(v : A \rightarrow B, k : B \rightarrow B)$ are both M-P PDs of a map $f : A \rightarrow B$. By Proposition 5.3, we have that $u = fh^\circ$ and $v = fk^\circ$, and that h and k are positive maps such that $hh = f^\dagger f = kk$. By the uniqueness of M-P square roots, this implies that $h = k$. In turn, this also implies that $u = v$. So we conclude that a M-P PD is unique. \square

6 Conclusion

In this paper, we revisited and added to the story of Moore-Penrose inverses in a dagger category. This work was motivated in part by wishing to understand how partial isomorphisms (in the restriction categories sense) generalize to dagger categories: Moore-Penrose inverses seem to provide the appropriate generalization. However, their theory is more sophisticated and could be better understood. In particular, although a start has been made here, there is more to be understood about their compositional behaviour and their relation to dagger idempotents. Moore-Penrose inverses should also be considered in relation to other dagger structures, such as dagger limits [14], dagger monads [13], and dagger compact closedness [27]. One should also find other interesting examples of Moore-Penrose dagger categories. We conjecture that certain fragments of the ZX-calculus [7] and possibly PROPs with weights on strings will be Moore-Penrose dagger categories. Finally, as the Moore-Penrose inverse has many practical applications, it would also be worthwhile generalizing these applications to Moore-Penrose dagger categories. This may in turn lead to further applications for Moore-Penrose inverses.

References

- [1] O. M. Baksalary & G. Trenkler (2021): *The Moore–Penrose inverse: a hundred years on a frontline of physics research*. *The European Physical Journal H* 46, pp. 1–10, doi:10.1140/epjh/s13129-021-00011-y.
- [2] R. B. Bapat, K. P. S. B. Rao & K. M. Prasad (1990): *Generalized inverses over integral domains*. *Linear Algebra and its Applications* 140, pp. 181–196, doi:10.1016/0024-3795(90)90229-6.
- [3] R. B. Bapat & D. W. Robinson (1992): *The Moore-Penrose inverse over a commutative ring*. *Linear algebra and its applications* 177, pp. 89–103, doi:10.1016/0024-3795(92)90318-5.
- [4] A. Ben-Israel (2002): *The Moore of the Moore-Penrose Inverse*. *The Electronic Journal of Linear Algebra* 9, pp. 150–157, doi:10.13001/1081-3810.1083.
- [5] S. Campbell & C. Meyer (2009): *Generalized inverses of linear transformations*. SIAM, doi:10.1137/1.9780898719048.
- [6] R. Cockett & S. Lack (2002): *Restriction categories I: categories of partial maps*. *Theoretical computer science* 270(1-2), pp. 223–259, doi:10.1016/S0304-3975(00)00382-0.
- [7] B. Coecke & A. Kissinger (2018): *Picturing quantum processes: A first course on quantum theory and diagrammatic reasoning*. In: *Diagrammatic Representation and Inference: 10th International Conference, Diagrams 2018, Edinburgh, UK, June 18-22, 2018, Proceedings 10*, Springer, pp. 28–31, doi:10.1017/9781316219317.
- [8] M. P. Drazin (1979): *Regular semigroups with involution*. In: *Proc. Symp. on Regular Semigroups*, pp. 29–46.
- [9] P. J. Freyd & A. Scedrov (1990): *Categories, Allegories*. Elsevier.
- [10] H. P. Gumm & M. Zarrad (2014): *Coalgebraic simulations and congruences*. In: *Coalgebraic Methods in Computer Science: 12th IFIP WG 1.3 International Workshop, CMCS 2014, Colocated with ETAPS 2014, Grenoble, France, April 5-6, 2014, Revised Selected Papers*, Springer, pp. 118–134, doi:10.1007/978-3-662-44124-4_7.
- [11] R. Hagen, S. Roch & B. Silbermann (2000): *C*-algebras and numerical analysis*. CRC Press, doi:10.1201/9781482270679.
- [12] C. Heunen & B. Jacobs (2010): *Quantum logic in dagger kernel categories*. *Order* 27(2), pp. 177–212, doi:10.1016/j.entcs.2011.01.024.
- [13] C. Heunen & M. Karvonen (2016): *Monads on dagger categories*. *Theory and Applications of Categories* 31(35), pp. 1016–1043, doi:10.48550/arXiv.1602.04324.

- [14] C. Heunen & M. Karvonen (2019): *Limits in dagger categories*. *Theory and Applications of Categories* 34(18), pp. 468–513, doi:10.48550/arXiv.1803.06651.
- [15] C. Heunen & J. Vicary (2019): *Categories for Quantum Theory: an introduction*. Oxford University Press, doi:10.1093/oso/9780198739623.001.0001.
- [16] N. J. Higham (2008): *Functions of matrices: theory and computation*. SIAM, doi:10.1137/1.9780898717778.
- [17] D. Huylebrouck, R. Puystjens & J. Van Geel (1984): *The Moore-Penrose inverse of a matrix over a semi-simple artinian ring*. *Linear and Multilinear Algebra* 16(1-4), pp. 239–246, doi:10.1080/03081088408817625.
- [18] E. H. Moore (1920): *On the reciprocal of the general algebraic matrix*. *Bull. Am. Math. Soc.* 26, pp. 394–395, doi:10.1090/S0002-9904-1920-03322-7.
- [19] M. H. Pearl (1968): *Generalized inverses of matrices with entries taken from an arbitrary field*. *Linear Algebra and its Applications* 1(4), pp. 571–587, doi:10.1016/0024-3795(68)90028-1.
- [20] R. Penrose (1955): *A generalized inverse for matrices*. In: *Mathematical proceedings of the Cambridge philosophical society*, 51, Cambridge University Press, pp. 406–413, doi:10.1017/S0305004100030401.
- [21] R. Puystjens & D. W. Robinson (1981): *The Moore-Penrose inverse of a morphism with factorization*. *Linear Algebra and its Applications* 40, pp. 129–141, doi:10.1016/0024-3795(81)90145-2.
- [22] R. Puystjens & D. W. Robinson (1984): *The Moore-Penrose inverse of a morphism in an additive category*. *Communications in algebra* 12(3), pp. 287–299, doi:10.1080/00927878408823004.
- [23] R. Puystjens & D. W. Robinson (1985): *EP morphisms*. *Linear algebra and its applications* 64, pp. 157–174, doi:10.1016/0024-3795(85)90273-3.
- [24] R. Puystjens & D. W. Robinson (1987): *Generalized inverses of morphisms with kernels*. *Linear Algebra and Its Applications* 96, pp. 65–86, doi:10.1016/0024-3795(87)90336-3.
- [25] R. Puystjens & D. W. Robinson (1990): *Symmetric morphisms and the existence of Moore-Penrose inverses*. *Linear Algebra and Its Applications* 131, pp. 51–69, doi:10.1016/0024-3795(90)90374-L.
- [26] P. S. S. N. V. P. Rao & K. P. S. B. Rao (1975): *On Generalized Inverses of Boolean Matrices*. *Linear Algebra and its applications* 11(2), pp. 135–153, doi:10.1016/0024-3795(75)90054-3.
- [27] P. Selinger (2007): *Dagger compact closed categories and completely positive maps*. *Electronic Notes in Theoretical computer science* 170, pp. 139–163, doi:10.1016/j.entcs.2006.12.018.
- [28] P. Selinger (2008): *Idempotents in dagger categories*. *Electronic Notes in Theoretical Computer Science* 210, pp. 107–122, doi:10.1016/j.entcs.2008.04.021.

Generalised Winograd Schema and its Contextuality

Kin Ian Lo Mehrnoosh Sadrzadeh

University College London
London, UK

{kin.lo.20,m.sadrzadeh}@ucl.ac.uk

Shane Mansfield

Quandela
Paris, France

shane.mansfield@quandela.com

Ambiguities in natural language give rise to probability distributions over interpretations. The distributions are often over multiple ambiguous words at a time; a multiplicity which makes them a suitable topic for sheaf-theoretic models of quantum contextuality. Previous research showed that different quantitative measures of contextuality correlate well with Psycholinguistic research on lexical ambiguities. In this work, we focus on coreference ambiguities and investigate the Winograd Schema Challenge (WSC), a test proposed by Levesque in 2011 to evaluate the intelligence of machines. The WSC consists of a collection of multiple-choice questions that require disambiguating pronouns in sentences structured according to the Winograd schema, in a way that makes it difficult for machines to determine the correct referents but remains intuitive for human comprehension. In this study, we propose an approach that analogously models the Winograd schema as an experiment in quantum physics. However, we argue that the original Winograd Schema is inherently too simplistic to facilitate contextuality. We introduce a novel mechanism for generalising the schema, rendering it analogous to a Bell-CHSH measurement scenario. We report an instance of this generalised schema, complemented by the human judgements we gathered via a crowdsourcing platform. The resulting model violates the Bell-CHSH inequality by 0.192, thus exhibiting contextuality in a coreference resolution setting.

1 Introduction

The Winograd Schema Challenge (WSC) originated from the ideas of the American computer scientist Terry Winograd in the 1970s. Winograd was interested in situations where machine understanding could fall behind human understanding. He constructed hypothetical experiments where humans and machines would read a given description, and then answer some questions about it. The descriptions would provide humans with enough context and thus they could answer the questions correctly. However, machine understanding would fall short, as machines did not learn from the context in the same way as humans did. An example description is the sentence “The city councilmen refused the demonstrators a permit because they feared violence.”. The question following it is “Who feared violence?” and the correct answer is “The city councilmen”. If we change the word “feared” to “advocated”, the question will have the opposite answer, namely “the demonstrators”. Winograd’s examples were picked up by the Canadian AI scientist Hector Levesque in 2011. He created a suite of descriptions and questions, proposing them as a test of machine intelligence - an alternative to the Turing Test [26]. Later, the AI company Nuance put forwards a cash prize of USD 25,000 for any AI that could solve the challenge with an accuracy close to humans, 92-96%. No AI system managed to achieve the target, and as a result, the prize was withdrawn in 2018. It was not until the 2020s that large pre-trained language models, employing transformer architectures, eventually reached a performance level comparable to human accuracy [24]. Despite these advancements, the WSC continues to present significant challenges for AI systems lacking extensive data resources and computational power.

In previous work, we showed how natural language notions of context can be modelled by the mathematics of quantum contextuality [36, 37, 34]. In particular, we modelled anaphoric context in [28]. Inspired by the reliance of the WSC on anaphoric context, we decided to explore whether quantum contextuality could potentially provide a solution to the challenge.

Our initial examination found that the WSC in its original form lacked the complexity required to be of interest from a quantum contextuality standpoint. Upon modelling the WSC within the sheaf theoretic framework, it became evident that the scenario was too simplistic to exhibit contextuality, as the models derived from it were deterministic.

This motivated us to extend the schema and allow it to be non-deterministic such that it can, in principle, host contextuality. This was achieved by introducing additional linguistic context, namely, (1) two special words rather than one and (2) two ambiguous pronouns instead of one. Consequently, we obtained more observables and more measurement contexts, leading to a scenario that resembles the Bell-CHSH scenario.

The above outlines the first contribution of this paper. Our second contribution lies in the instantiation of our generalized Winograd Schema and the collection of human judgments via a crowdsourcing platform. This allowed us to calculate the violation of the Bell-CHSH inequality and thereby establish the contextuality of our model, which was constructed based on human judgments. We also modelled the data using the Contextuality-by-Default (CbD) framework of contextuality and calculated a corresponding CbD degree of contextuality. It was found that our probabilistic model exhibited contextuality in both the Bell-CHSH and CbD models.

2 Contextuality

The origins of contextuality research can be traced back to 1935, with the work of Einstein, Podolsky, and Rosen (EPR) [15]. In their work, they posited that the quantum mechanical description of physics was incomplete when two spatially separated parties were permitted to make measurements on an entangled system. A way of formalising such theories is in terms of hidden variables, which, if known, might fully determine the outcome that would result from any given measurement. Bell's theorem [7, 6] in the 1960s showed that no hidden-variable theory exists for quantum mechanics unless the measurement outcomes were allowed to be dependent on which other measurements are performed simultaneously. Around the same time, Kochen and Specker [23] independently demonstrated that there exists a set of measurements in a 3-dimensional Hilbert space such that a non-contextual hidden-variable theory cannot exist, regardless of the state of the system. These two results, collectively known as the Bell-Kochen-Specker theorem, showed that a hidden-variable theory for quantum mechanics must be contextual, providing some clarity to the debate on a more fundamental theory conforming to certain classical intuitions for quantum mechanics. The first attempt at experimentally verifying Bell's inequality was performed by Aspect et al. [5], with the most recent ones closing all known loopholes in the earlier experiments [18, 20, 32]. Thus it has been established that quantum physics is vastly different from classical physics – a description of quantum physics that agrees with our classical intuition must be contextual.

Other than the philosophical implications, contextuality has been shown to possess computational power through non-classical correlations. Anders and Browne first showed that certain measurements on GHZ states can be used to lift a linear classical computation into a universal classical computation [4]; Raussendorf later showed that the probability of success of such computation is bounded by the degree of contextuality [30], as measured by the contextual fraction [2, 1]. Subsequent work by Howard et al. revealed that contextuality is an essential ingredient for *magic state distillation*, a process that yields

specific quantum states known as *magic states* [21]. The current most promising fault-tolerant quantum computing scheme, the surface code [22], only permits fault-tolerant computation with a subset of quantum operations which can be efficiently simulated by classical computers. Via state injection, these magic states can be used with surface code to allow for fully fault-tolerant universal quantum computation. Thus, one might argue that contextuality carries an intrinsic computational power that is absent in non-contextual systems.

A variety of frameworks for modelling contextuality have been developed. These including the sheaf-theoretic framework [2, 3, 1], the Contextuality-by-Default (CbD) framework [11, 14, 12], the graph-theoretic framework [8], a framework based on simplicial sets [29]. Generally speaking, these frameworks enable the formalisation of the notion of measurement through the use of various mathematical structures. Bell's inequalities, or in general inequalities that witness contextuality, can be derived systematically within these frameworks. Although we will mainly use the terminology from the sheaf-theoretic framework to describe our examples, our results are framework-agnostic.

2.1 Sheaf Theoretic Framework

Here, we provide a concise overview of the sheaf-theoretic framework of contextuality proposed by Abramsky and Brandenburger [2]

A measurement scenario is defined as a triplet $\langle X, \mathcal{M}, O \rangle$, where X refers to a collection of observables, O is the possible outcomes, and \mathcal{M} denotes an abstract simplicial complex composed of subsets from X .

Every element in X is an observable of the system under consideration. Upon measurement, each observable yields one of the outcomes contained in O . The characterization of \mathcal{M} as an abstract simplicial complex implies a particular structural feature: if a subset C belongs to \mathcal{M} , then every subset nested within C must also be an element of \mathcal{M} .

A necessity of contextuality is that one cannot measure all the observables in X simultaneously, at least not without altering the state of the system. Thus, every framework for contextuality must provide a description of the compatibility between observables. Within the sheaf-theoretic framework, each simplex in the simplicial complex \mathcal{M} constitutes a subset of observables in X that are measurable simultaneously, i.e. they are mutually compatible. A *measurement context*, or simply *context*, is defined as a maximal simplex in \mathcal{M} , which is not a proper subset of any other simplex in \mathcal{M} .

For instance, the measurement scenario in the Bell-CHSH settings is specified by $X = \{a_1, a_2, b_1, b_2\}$; $\mathcal{M} = \{\{a_1, b_1\}, \{a_1, b_2\}, \{a_2, b_1\}, \{a_2, b_2\}\}$; $O = \{0, 1\}$. The simplicial complex \mathcal{M} can be geometrically realized as the boundary of a square, where each vertex corresponds to an observable and each edge represents a context (see Figure 1(a)). Two parties are involved in this scenario: Alice is allowed to measure either a_1 or a_2 , and Bob is allowed to measure either b_1 or b_2 . The measurements are dichotomic, i.e. the outcomes are either 0 or 1.

Every subset of observables which is a context in \mathcal{M} can be measured jointly. Thus we can define a (local) joint probability distribution over the observables in the context. Such a joint probability distribution can either be estimated by performing the measurements in an experiment, or be calculated according to a theory of the system under consideration. A collection of all such joint probability distributions is called an *empirical model*, or simply *model*, of the system. For instance, using a set of appropriately chosen measurement bases, the Bell state $|\Psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ produces the empirical model depicted in Figure 1(b). This state exhibits the highest violation of the Bell-CHSH inequality among all quantum states.

An empirical model is said to be *signalling* if the marginalised distribution of a set of observables

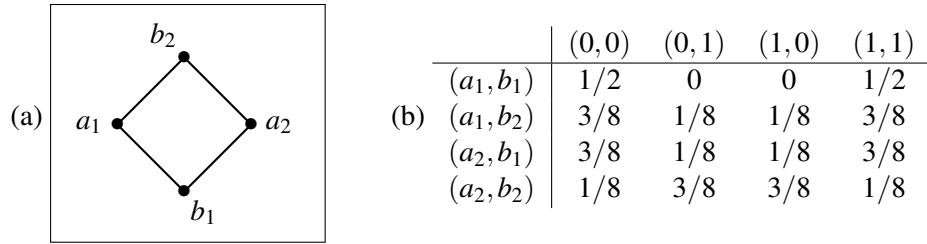


Figure 1: (a) The simplicial complex \mathcal{M} in the Bell-CHSH scenario. Every vertex represents an observable and every edge represents a context. Alice chooses between a_1 and a_2 ; Bob chooses between b_1 and b_2 . The absence of edges between a_1 and a_2 , and between b_1 and b_2 , indicates their incompatibility. (b) An empirical model of the Bell-CHSH scenario. Each row represents a joint probability distribution over the observables in the context. For example, the bottom-right entry $1/8$ is the probability of observing $a_2 = 1$ and $b_2 = 1$ when measuring the observables in the context (a_2, b_2) .

differs from one context to another. In contrast, non-signalling implies that the observed probabilities remain invariant under different contexts, thereby preventing the transmission of information through the choice of context.

A prevalent misconception is a belief that *signalling is contextuality*, often based on the incorrect reasoning that the *probabilities* in a signalling model are generally context-dependent, leading to the conclusion that the model is contextual. However, it is essential to recognize a fundamental distinction between the two concepts: signalling pertains to the observed probabilities, while contextuality relates to the underlying hidden-variable theories of the model.

The qualitative criterion for contextuality of a model in the sheaf-theoretic framework is based on Fine's theorem [17], which states that a model is contextual if and only if there exists a global probability distribution that is compatible with every local probability distribution in the model.

The quantitative degree of contextuality of a model is measured by the *contextual fraction* CF [1]. Given an empirical model e , the contextual fraction $\text{CF}(e)$ is defined as the minimum λ such that e admits a convex decomposition¹:

$$e = (1 - \lambda)e^{NC} + \lambda e^C, \quad (1)$$

where e^{NC} is a non-contextual (and non-signalling) empirical model and e^C is an empirical model that may be contextual.

Suppose a given model e is non-contextual, then λ can be set to zero by choosing $e^{NC} = e$. Otherwise, λ must be taken to be greater than zero to make the decomposition valid. Therefore, for non-signalling models, the sheaf-theoretic criterion of contextuality is

$$\text{CF}(e) > 0. \quad (2)$$

The calculation of CF can be reduced to solving for a linear program, for which numerical solvers are readily available. The CF of a model has a nice interpretation as the maximum amount of *normalised violation* of all possible general Bell's inequalities [1].

In the case of signalling models, the above decomposition cannot hold because e^{NC} and e^C are, by definition, non-signalling. We could consider allowing e^C to be signalling. However, this adjustment would lead to the misleading conclusion that all signalling models are contextual, assuming we maintain our interpretation of CF as a measure of contextuality for these models.

¹Here, we represent the empirical models as empirical tables. Addition and scalar multiplication are then interpreted as standard matrix operations, where the empirical tables are treated as matrices.

2.2 Contextuality by Default

In the setting of Contextuality-by-Default (CbD), there are two important notions: *contents*, denoted by q_i , which are measurements, or more generally, questions about the system; and *contexts*, denoted by c^j , which represent the conditions under which the questions are asked, e.g. their ordering. Every q_i in a c^j gives rise to a random variable R_i^j taking values in $\{\pm 1\}$, and representing possible answers and their probabilities. All random variables in a given context are jointly distributed.

A well-studied class of CbD systems are the cyclic systems [11, 12, 13], where each context has exactly 2 contents and every content is in exactly 2 contexts. The rank of a cyclic system is the number of contents, or equivalently, the number of contexts.

A cyclic system of rank n is contextual if and only if CNT_1 is positive, where CNT_1 is defined as:

$$\text{CNT}_1 := s_{\text{odd}} \left(\left\{ \langle R_{i_j}^j R_{i'_j}^j \rangle \right\}_{j=1, \dots, n} \right) - \Delta - n + 2 > 0 \quad (3)$$

where $i_j \neq i'_j$ for all j and $R_{i_j}^j, R_{i'_j}^j$ are well-defined for all j . Quantities $s_{\text{odd}} : \mathbb{R}^n \rightarrow \mathbb{R}$ and Δ are defined as follows:

$$s_{\text{odd}}(\underline{x}) = \max_{\substack{\sigma \in \{\pm 1\}^k; \\ \text{p}(\underline{\sigma} = -1)}} \underline{\sigma} \cdot \underline{x}; \quad \Delta = \sum_{i=1}^n \left| \langle R_i^{i_i} \rangle - \langle R_i^{i'_i} \rangle \right| \quad (4)$$

where $\text{p}(\underline{\sigma}) = \prod_{i=1}^n \sigma_i$ (p is the parity function of $\underline{\sigma}$). The quantity Δ measures the degree of signalling in the system. Thus, a non-signalling system has $\Delta = 0$.

For a rank 4 cyclic system, i.e. the Bell-CHSH scenario, the above inequality reduces to the maximum violation of the Bell-CHSH inequalities over the choices of the four signs:

$$\text{CNT}_1 = \pm \langle R_0^0 R_1^0 \rangle \pm \langle R_1^1 R_2^1 \rangle \pm \langle R_2^2 R_3^2 \rangle \pm \langle R_3^3 R_0^3 \rangle - 2 \quad (5)$$

where the number of minus signs has to be taken odd. Therefore, the CbD criterion of contextuality coincides with the Bell-CHSH inequalities for the Bell-CHSH scenario.

2.3 Ambiguous words as observables

Ambiguities in natural language have posed a challenge to natural language processing. Lexical ambiguity, where a word has multiple meanings, is one of the most common types of ambiguity in natural language. For instance, the word *produce* has two possible meanings: *to give birth* and *to make something*.

Without any context, it is not possible to determine which of the two meanings is intended. Another type of ambiguity is *coreference ambiguity*, where a word can potentially refer to different entities. For instance, the pronoun *it* can refer to the *dog* or the *cat* in the sentence *The dog chased the cat. It barked..* In this paper, we focus on the latter type of ambiguity.

A method to formalise the notion of contextuality in natural language is by viewing an ambiguous word as an observable, with its interpretations as possible outcomes. For instance, the word *produce* has (at least) two possible interpretations: *to give birth* and *to make something*. Measuring the word *produce* amounts to selecting one of these interpretations by a reader.

We can assign probabilities to these interpretations based on the frequency of the interpretations in an experiment where a group of readers is asked to interpret the word *produce*, or a single reader is asked to assign a probability to each of the interpretations. The first approach is more costly as it requires a

large number of readers to be involved in the experiment. However, the latter approach is better suited to machine learning models since they can be trained to assign probabilities to different interpretations.

This way of treating ambiguous words as observables was first proposed by Wang et al. [36, 37]. The authors considered subject-verb and verb-object phrases where each word carries at least two possible interpretations. Measurement contexts were constructed by selecting different pairs of nouns and verbs, in a way similar to how Alice and Bob select their measurements in the Bell-CHSH scenario. The probabilities in the results were estimated from a group of crowd workers who were asked to assign a score to the different interpretations.

3 Winograd Schema Challenge

Commonsense reasoning, the inherent human capacity to logically comprehend the world around us, has long been a focal point in the field of artificial intelligence, with the aim to cultivate this ability in machines.

The Winograd Schema Challenge (WSC) emerged as a measure of this commonsense reasoning capability. The challenge was inspired by Terry Winograd's seminal paper [38], wherein he contended that syntax alone falls short in the interpretation of natural language, necessitating commonsense or world knowledge as well. The challenge presents a collection of sentences, each with an ambiguous pronoun whose meaning can be clarified via the context. A machine is deemed to have passed the test if it can disambiguate the pronoun with an accuracy on par with human performance.

The classic example of a Winograd schema, originally constructed by Winograd himself, is the following pair of sentences:

- (1) a. The city councilmen refused the demonstrators a permit because **they** *feared* violence.
- b. The city councilmen refused the demonstrators a permit because **they** *advocated* violence.

Note that the two sentences differ only in the words *feared* and *advocated*. In both sentences, there is an ambiguous pronoun **they** which can either refer to the *city councilmen* or the *demonstrators*. In the first sentence, it can be inferred through commonsense reasoning that the pronoun **they** refers to the *city councilmen*, as it is within our common sense that city councilmen are the ones who tend to prevent violence in demonstrations. In the second sentence, the pronoun *they* refers to the *demonstrators*, as it is within our common sense (stereotype) that demonstrators tend to advocate violence and that doing so would lead to the refusal of a permit for a demonstration.

Another classic example of a Winograd schema is the following pair of sentences:

- (2) The trophy doesn't fit into the suitcase because it's too [*small / large*].

Here we adopt a compact notation in which the pair of square brackets encloses the two possible word choices, each leading to a different sentence. This notation will be employed throughout the paper.

In a WSC, the participant is asked to identify the correct interpretation of the ambiguous pronoun. Success in the test is defined by the participant's accuracy equalling or approximating human performance. The evaluation of responses to a WSC question is straightforward, either the correct referent of the ambiguous pronoun is identified or not.

In contrast, the Turing Test has been criticised for being too difficult to evaluate. Originated as the imitation game by Turing [33], the test involves a human judge interrogating a machine via a textual interface. The conversation between the judge and the machine is unrestricted. If the judge or a panel of judges cannot distinguish the machine from a human based on the conversation, the machine is deemed

to have passed the test. However, this unrestricted nature of the Turing Test opens doors to potential deception. In fact, for a machine to pass the test, it must deceive as machines lack physical bodies. If questioned about its physical attributes, like height or weight, the machine must lie to successfully pose as a human. Due to this advantage of the ease of evaluation over the Turing Test, the WSC was proposed as a replacement for the Turing Test.

Unlike the Turing Test, the WSC is a structured binary-choice test. The major issue with the WSC is that it is over-constrained - it is unexpectedly difficult to construct examples of it, due to the numerous requirements that must be satisfied. A valid Winograd schema must satisfy the following requirements:

1. A Winograd Schema comprises a pair of sentences that differ slightly from each other. The first sentence includes a *special* word which, when replaced by an *alternate* word, yields the second sentence. For instance, in the *trophy-suitcase* example, *small* is the *special* word, and *large* is its *alternate*.
2. The sentences should contain two noun phrases. In the *trophy-suitcase* example, *the trophy* and *the suitcase* serve as the two noun phrases.
3. A pronoun, which agrees with the two noun phrases in number and gender, must be present in the sentences. For example, in the *trophy-suitcase* scenario, the pronoun *it* aligns with both *the trophy* and *the suitcase* regarding number and gender.
4. The pronoun's referent should be easily identifiable from a natural reading of the sentence, and the correct referent should differ between the two sentences.
5. Each sentence in the pair should be fluid and natural to read, to the extent that they could feasibly appear in regular text sources like news articles or Wikipedia pages.

The outlined requirements ensure the preservation of both linguistic structure and the test's integrity:

1. The first requirement ensures grammatical consistency across the pair of sentences.
2. The fourth requirement necessitates a change in the correct referent of the pronoun when the special word is replaced with the alternate. This stipulation indicates that grammatical structure alone does not determine the correct pronoun referent.
3. The fifth requirement safeguards the authenticity of the language used in the test, ensuring it remains aligned with naturally occurring language.

Crafting valid examples of the Winograd schema is a complex task due to the set restrictions and requirements. The challenge of creating such schemas is evidenced by the limited number of examples in the original Winograd Schema Challenge set, which includes only 285 instances².

In 2018, the first system achieved a better-than-chance accuracy of 57.1% [16] on the original 285 examples of the WSC. In 2019, a fine-tuned RoBERTa [27] model achieved a human-like accuracy of 90.1% [31]. The WSC has suffered from the same problem that plagued the Turing Test – there are weaknesses in the test that can be exploited without having to demonstrate the desired human-level intelligence. Simply put, the WSC has been defeated [24].

It is even more so for the WSC precisely because of its ease of evaluation. Proposals to increase the difficulty of the WSC, such as requiring the test-taker to select a correct explanation for their answer from a list of options [39, 19], emerged as potential solutions. However, these suggestions further complicate the already challenging task of question set construction. An alternative could involve requiring free-form explanations from the test-taker, though this would likely introduce additional ambiguity and make the evaluation process more difficult.

²Available at <https://cs.nyu.edu/davise/papers/WinogradSchemas/WS.html>.

4 Generalised Winograd Schema

In this section, we present our approach for the generalisation of the Winograd Schema, enabling the potential observation of contextuality. We will first discuss why the original Winograd Schema is insufficiently complex to exhibit contextuality, and then propose a generalised Winograd Schema that is sophisticated enough to host contextuality.

4.1 Modelling Winograd Schemas as measurement scenarios

To study the contextuality in the Winograd Schema, we model it with a measurement scenario in the sheaf-theoretic framework. This way of treating ambiguity in language is akin to the way ambiguous phrases are treated in [36], where an ambiguous word is considered an observable in a measurement scenario.

However, the same ambiguous word, i.e. the ambiguous pronoun, is shared across the twin pair of sentences in a Winograd Schema. Thus, if we follow the approach of “words as observables” strictly, then we will end up with a trivial measurement scenario, where there is only one observable, i.e. the ambiguous pronoun. Moreover, this naive approach deviates from the spirit of the Winograd Schema, which is to disambiguate a pronoun by considering the linguistic context. Instead, We argue that there should be exactly two contexts in the measurement scenario, one for each sentence in the twin pair. Recall that in the original Winograd Schema, the twin pair of sentences are identical except for the special word and the alternate word. In a rough sense, the special word and the alternate word provide the *linguistical context* for disambiguating the pronoun. This way of defining the measurement contexts provides a concrete link between *context in language* and *contextuality in quantum mechanics*.

Following from the above discussion, we define an observable as a tuple: **(pronoun, special word)** or **(pronoun, alternate word)**, to distinguish between the two pronouns in different linguistic contexts. The possible outcomes of each of the two observables are the candidate referents of the pronoun.

Definition 1 (Winograd Schema scenario) *Given a Winograd Schema with two noun phrases A and B ; an ambiguous pronoun \mathbf{p} which refers to either A or B ; a special word (s) and an alternate word (a), the corresponding measurement scenario is defined by the data:*

- observables $X = \{(\mathbf{p}, s), (\mathbf{p}, a)\}$;
- contexts $\mathcal{M} = \{\{(\mathbf{p}, s)\}, \{(\mathbf{p}, a)\}\}$;
- outcomes $O = \{A, B\}$.

We call such a measurement scenario a Winograd Schema scenario, or a WS scenario in short.

With the *councilmen-demonstrators* example, the measurement scenario would be given by the data:

- observables $X = \{(\mathbf{they}, \text{feared}), (\mathbf{they}, \text{advocated})\}$;
- contexts $\mathcal{M} = \{\{(\mathbf{they}, \text{feared})\}, \{(\mathbf{they}, \text{advocated})\}\}$;
- outcomes $O = \{\text{city councilmen}, \text{demonstrators}\}$.

It becomes apparent that any Winograd Schema scenario is too simplistic to accommodate any contextual model due to the absence of overlapping contexts. One can always construct a compatible global distribution by taking the product of the local distributions.

4.2 Generalising the Winograd Schema scenario

Before proceeding to the generalisation of Winograd Schema, we point out an interpretation of the WS scenario as an analogy to an experiment in quantum physics. Consider an imaginary experimenter, Alice, who decides whether to measure the pronoun with the special word, or with the alternate word. That is, Alice chooses between the two observables: (\mathbf{p}, s) and (\mathbf{p}, a) . This is exactly analogous to Alice choosing between two projection axes in an experiment measuring a spin-1/2 particle.

A natural and obvious way to generalise the WS scenario would be to add one more experimenter, Bob. This results in the Bell-CHSH scenario, which is well-known to be able to host contextual models. That amounts to introducing one more pronoun, one more special word and its alternate word, to the original Winograd Schema. We use the subscript 1 to denote objects relating to the first pronoun and the subscript 2 to denote objects relating to the second pronoun.

Here we give a set of requirements for the generalised Winograd Schema, in the style of the original WSC:

1. A generalised schema consists of four slightly differing sentences. The first sentence contains two special words s_1 and s_2 . Similar to the original Winograd Schema, s_1 can be replaced by an alternate word a_1 and s_2 can be replaced by an a_2 . The possibility of replacing special words with alternate words creates the rest of the four sentences.
2. There are a pair of noun phrases.
3. There are two pronouns in the sentences. The first pronoun refers to one of the noun phrases in the first pair of noun phrases. The second pronoun refers to either one noun phrase in the second pair of noun phrases.
4. All four sentences should be natural to read.

In short, a generalised Winograd Schema is two Winograd Schemas put together in a single discourse.

Definition 2 (Generalised Winograd Schema scenario) *Given a Generalised Winograd Schema with two noun phrases A and B ; two ambiguous pronouns \mathbf{p}_1 and \mathbf{p}_2 can each refer to either A or B ; two special words (s_1) and (s_2) ; two alternate words (a_1) and (a_2) , the corresponding measurement scenario is defined by the data:*

- observables $X = \{(\mathbf{p}_1, s_1), (\mathbf{p}_1, a_1), (\mathbf{p}_2, s_2), (\mathbf{p}_2, a_2)\}$
- contexts $\mathcal{M} = \{(\mathbf{p}_1, s_1), (\mathbf{p}_2, s_2)\}, \{(\mathbf{p}_1, s_1), (\mathbf{p}_2, a_2)\}, \{(\mathbf{p}_1, a_1), (\mathbf{p}_2, s_2)\}, \{(\mathbf{p}_1, a_1), (\mathbf{p}_2, a_2)\}$;
- outcomes $O = \{A, B\}$.

Such a measurement scenario is called a Generalised Winograd Schema scenario, or a generalised WS scenario in short.

The generalised WS scenario is isomorphic, i.e. identical upon relabelling, to the Bell-CHSH scenario shown in Figure 1. It has long been known that the Bell-CHSH scenario can host contextual models [6, 9]. Thus a carefully designed generalised Winograd Schema would be able to demonstrate contextuality.

Here we provide a straightforward example of a generalized Winograd Schema scenario, built upon the original *trophy-suitcase* example:

- (3) The trophy doesn't fit into the suitcase because \mathbf{it}_1 is too [$s_1 = small / a_1 = large$]. Nonetheless, \mathbf{it}_2 is [$s_1 = light / a_2 = heavy$].

The corresponding generalised WS scenario is given by:

- observables $X = \{(\mathbf{it}_1, \textit{small}), (\mathbf{it}_1, \textit{large}), (\mathbf{it}_2, \textit{light}), (\mathbf{it}_2, \textit{heavy})\}$
- contexts $\mathcal{M} = \{ \{(\mathbf{it}_1, \textit{small}), (\mathbf{it}_2, \textit{light})\}, \{(\mathbf{it}_1, \textit{small}), (\mathbf{it}_2, \textit{heavy})\}, \{(\mathbf{it}_1, \textit{large}), (\mathbf{it}_2, \textit{light})\}, \{(\mathbf{it}_1, \textit{large}), (\mathbf{it}_2, \textit{heavy})\} \};$
- outcomes $O = \{\textit{trophy}, \textit{suitcase}\}$.

Interestingly, it was in the original set of Winograd Schemas (WSC285) that Davis designed a special example making use of two pronouns:

- (4) Sid explained his theory to Mark but **he** couldn't [*convince / understand*] **him**.

The author deemed this example a “Winograd schema in the broad sense” since using more than one pronoun violates the requirements of the original Winograd Schema. Yet, this example is not a proper generalised Winograd Schema defined in this paper, as it only employs one special word and one alternate word.

Other than the fact that its scenario is too simple, there is another reason why the original Winograd Schema is not contextual: the intended referent of the pronoun should be obvious to a human reader. That means an empirical model constructed with judgement data collected from human subjects on the original Winograd Schema would be deterministic or nearly deterministic. It is known that deterministic systems are not contextual [10]. On the other extreme, a completely random model is trivially non-contextual. Intriguingly, it seems that only a system with a moderate level of intelligence, in between that of humans and that of complete randomness, would have the possibility of being contextual.

There are two directions to where we could take the generalised Winograd Schema: (1) to continue its mission to be a test of intelligence or commonsense reasoning; (2) to become a well-structured linguistic setting under which contextual models could be found.

Recent results from large language models have demonstrated human-like accuracies in solving the Winograd Schema Challenge. The introduction of one more pronoun might increase the difficulty of the challenge, possibly stipulating advancements in the field of natural language processing. However, it is our goal to find bridges between natural language and contextuality. Therefore the second direction will be the focus of this paper.

4.3 An example of the generalised Winograd Schema

As our goal is to uncover contextual models in natural language, we need to gather judgment data from human participants to build empirical models for generalized Winograd Schema instances. Crucially, deterministic systems lack contextuality. Therefore, our generalized Winograd Schema examples should be inherently ambiguous to human readers, unlike the original Winograd Schema where humans can easily resolve the pronoun.

Due to the requirement of having two almost identical pairs of naturally-sounding sentences, it is a difficult task to come up with examples of the original Winograd Schema. The extra requirements we put forward for the generalised Winograd Schema make it even harder to come up with naturally-sounding examples. Here we report an example of the generalised Winograd Schema³:

- (5) A and B belong to the same [*cannibalistic / herbivorous*]₁ species of animal. On a hot afternoon in the south Sahara, **one of them**₁ was very hungry. They noticed each other when they were roaming in the field. After a while, **one of them**₂ is no longer [*hungry / alive*]₂.

³It was pointed out by one of the reviewers that the original version of the example contains several incorrect uses of English. Here we provide the corrected version of the example.

Note that we had to violate the requirement of having a single sentence because it is difficult to come up with a naturally-sounding sentence that contains every ingredient of the generalised Winograd Schema. We also decided to use the referring phrase **one of them** instead of the third-person pronoun **it** to improve the naturalness of the example.

We used the alphabetic symbols A and B as the two noun phrases as we wanted to make the two symmetric. That is, any empirical model of the scenario is invariant to the interchanging of A and B. It turns out that all symmetric models are non-signalling, at least for cyclic scenarios such as that Bell-CHSH scenario. Dealing with symmetric models carries two disadvantages: (1) it is more difficult to assert the contextuality of a signalling model; (2) the sheaf-theoretic criterion of contextuality applies to non-signalling models only. By considering only symmetric models, we thereby avoid the complications of dealing with non-signalling models.

4.4 Human judgements on the example

We collected human judgments on this example on the crowd-sourcing platform Amazon Mechanical Turk in the form of a questionnaire. There were four versions of the questionnaire, each corresponding to one of the four contexts in the generalised WS scenario. The respondents were asked to read the example and answer a question about the correct referents, A or B, of the two referring phrases **one of them₁** and **one of them₂**. A screenshot of the questionnaire is shown in Figure 2.

Instruction: Please read the following short story which contains some ambiguities, then select the interpretations you think are the most appropriate.

Story: A and B belong to the same $\{\text{word1}\}$ species of animals. In a hot afternoon in south Sahara, one of them was very hungry. They notice each other when they were roaming in the field. In a while, one of them is no longer $\{\text{word2}\}$.

Question: The following are 4 different interpretations of the story. Please select the **2** most appropriate interpretations.

- A was the very hungry $\{\text{word1}\}$ animal. A is no longer $\{\text{word2}\}$.
- A was the very hungry $\{\text{word1}\}$ animal. B is no longer $\{\text{word2}\}$.
- B was the very hungry $\{\text{word1}\}$ animal. A is no longer $\{\text{word2}\}$.
- B was the very hungry $\{\text{word1}\}$ animal. B is no longer $\{\text{word2}\}$.

Figure 2: A screenshot of the template of the questionnaire. The placement holders $\{\text{word1}\}$ and $\{\text{word2}\}$ are instantiated with the two special words or the alternate words of the generalised Winograd Schema. In this example, $\{\text{word1}\}$ can be either *cannibalistic* or *herbivorous* and $\{\text{word2}\}$ can be either *hungry* or *alive*. Four versions of the questionnaire were created, each corresponding to one of the four contexts in the generalised WS scenario. *Note that the story contains several incorrect uses of English. Unfortunately, we did not notice these until a reviewer pointed them out, after data collection.*

Since each referring phrase can be interpreted in two ways, there are 4 possible combinations of interpretations, (A, A), (A, B), (B, A), (B, B), of the two referring phrases. The symmetry between A and B in the example ensures that the combinations (A, A) and (B, B) are equally plausible and (A, B) and (B, A) are also equally plausible. Therefore we asked the respondents to pick two out of the four combinations. This design choice also allows the detection of invalid answers, that is, those that do not

(a)	(A, A)	(A, B)	(B, A)	(B, B)	(b)	(A, A)	(A, B)	(B, A)	(B, B)
(<i>canni</i> , <i>hungry</i>)	0.402	0.097	0.097	0.402	...	1/2	0	0	1/2
(<i>canni</i> , <i>alive</i>)	0.044	0.455	0.455	0.044	...	0	1/2	1/2	0
(<i>herbi</i> , <i>hungry</i>)	0.345	0.154	0.154	0.345	...	1/2	0	0	1/2
(<i>herbi</i> , <i>alive</i>)	0.344	0.155	0.155	0.344	...	1/2	0	0	1/2

Table 1: (a) The empirical model constructed with the 410 human judgments collected from Amazon Mechanical Turk. The violation of Bell’s inequality of the model is 0.192 ± 0.176 . For brevity, the special word *cannibalistic* is shortened to *canni* and the alternate word *herbivorous* is shortened to *herbi*. The model generally resembled the PR model shown in Table (b) on the right.

respect the symmetry between A and B.

A total of 410 responses were collected on Amazon Mechanical Turk separately on two dates: 20th Oct 2022 and 23rd Nov 2022. Out of the 410 responses, 110 were to the context (*cannibalistic*, *hungry*) and 100 each were to the rest of the three contexts. Out of all the responses, 348 were valid, i.e. their responses respected the symmetry between A and B. The respondents were each financially rewarded USD 1.00, regardless of the validity of their responses.

The collected valid data were used to build an estimated probability distribution for each of the four contexts. The resulting empirical model is shown in Table 1. The model violates the Bell-CHSH inequality by 0.192 with a standard deviation of 0.176. Since the model is symmetric in the outcomes by construction, it is non-signalling and thus the measure of contextuality CNT_1 in the Cbd framework coincides with the degree of violation [25]. The symmetry in the outcomes also allows the violation to saturate the bound defined by CF in sheaf-theoretic framework [1], i.e. the following equality is attained

$$\max \left\{ 0, \frac{1}{2} \text{ violation of Bell-CHSH inequality} \right\} = \text{CF}. \quad (6)$$

Thus, our model is considered contextual in both the sheaf-theoretic framework and the Cbd framework.

To establish the significance of the contextuality result, we conducted bootstrap resampling to estimate the spread of the violation to the Bell-CHSH inequality. Simulated datasets were generated by random sampling with replacement from the original dataset. The resulting distribution of violations is depicted in Figure 3. Among the resampled datasets, 87% of them exhibited a positive violation, indicating that our experimental model demonstrates contextuality with a significance level of 87%.

5 Conclusions and Future Work

In this work, we employed the sheaf-theoretic framework for contextuality to model the Winograd Schema, originally formulated as an ambiguous coreference resolution task. Our findings revealed that the original Winograd Schema scenario lacked the necessary complexity to exhibit contextuality. To address this limitation, we introduced an additional ambiguous pronoun and a new pair of special and alternate words, creating a generalized Winograd Schema reminiscent of the Bell-CHSH scenario. Through crowdsourcing, we collected human judgments on an example of the generalized Winograd Schema and observed a contextual empirical model with a significance level of 87

An intriguing direction for future research involves constructing a comprehensive set of examples based on the proposed generalized Winograd Schema, thereby establishing it as a new challenge in the field of natural language processing. One potential approach is to leverage state-of-the-art generative

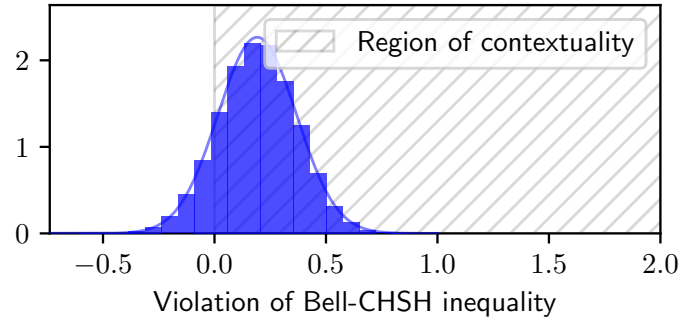


Figure 3: A normalised histogram of the Bell-CHSH inequality violation for 100,000 bootstrap samples from the model shown in Table 1. A positive violation, indicative of contextuality, is observed in 87% of the resampled models. The standard deviation of the distribution is 0.176.

language models such as GPT-4 to systematically generate examples of the schema with minimal human intervention. Careful prompt engineering would be needed to ensure that the generated examples are of high quality.

As collecting human judgments is costly and time-consuming, another alternative approach for constructing empirical models of the generalized Winograd Schema involves utilizing generative language models to generate responses to examples. This approach also offers an opportunity to explore the extent to which the responses generated by language models align with human responses. By comparing and analysing the correspondence between model-generated responses and human responses, one could gain insights into the capabilities and limitations of language models in capturing the way human beings understand language.

This paper presents an approach that consists of deliberately constructing sentences that exhibit contextuality. This strategy of “detecting contextuality in natural language” may invite criticism for its contrived nature.

An alternative approach could involve the application of mathematical frameworks designed for contextuality to analyze pre-existing natural language data, moving away from the intentional construction of examples with distinct features [35]. The aim of this strategy would not be to pursue contextuality within natural language. Instead, it would focus on developing novel methods for modelling natural language phenomena from a different perspective.

Acknowledgements

We are grateful to Daphne Wang for insightful discussions and the anonymous reviewers for their constructive comments. KL is supported by the Engineering and Physical Sciences Research Council [grant number EP/S021582/1]. MS is supported by the Royal Academy of Engineering research chair RCSR2122-14-152 on Engineered Mathematics for Modelling Typed Structures.

References

- [1] Samson Abramsky, Rui Soares Barbosa & Shane Mansfield (2017): *Contextual Fraction as a Measure of Contextuality*. *Physical Review Letter* 119, p. 050504, doi:10.1103/PhysRevLett.119.050504.
- [2] Samson Abramsky & Adam Brandenburger (2011): *The sheaf-theoretic structure of non-locality and contextuality*. *New Journal of Physics* 13(11), p. 113036, doi:10.1088/1367-2630/13/11/113036.
- [3] Samson Abramsky, Shane Mansfield & Rui Soares Barbosa (2012): *The Cohomology of Non-Locality and Contextuality*. *Electronic Proceedings in Theoretical Computer Science* 95, pp. 1–14, doi:10.4204/EPTCS.95.1.
- [4] Janet Anders & Dan E. Browne (2009): *Computational Power of Correlations*. *Physical Review Letter* 102, p. 050502, doi:10.1103/PhysRevLett.102.050502.
- [5] Alain Aspect, Jean Dalibard & Gérard Roger (1982): *Experimental Test of Bell's Inequalities Using Time-Varying Analyzers*. *Phys. Rev. Lett.* 49, pp. 1804–1807, doi:10.1103/PhysRevLett.49.1804.
- [6] J. S. Bell (1964): *On the Einstein Podolsky Rosen paradox*. *Physics Physique Fizika* 1(3), pp. 195–200, doi:10.1103/PhysicsPhysiqueFizika.1.195.
- [7] John S. Bell (1966): *On the Problem of Hidden Variables in Quantum Mechanics*. *Reviews of Modern Physics* 38(3), pp. 447–452, doi:10.1103/RevModPhys.38.447.
- [8] Adán Cabello, Simone Severini & Andreas Winter (2014): *Graph-theoretic approach to quantum correlations*. *Physical Review Letters* 112(4), pp. 1–5, doi:10.1103/PhysRevLett.112.040401.
- [9] John F. Clauser, Michael A. Horne, Abner Shimony & Richard A. Holt (1969): *Proposed Experiment to Test Local Hidden-Variable Theories*. *Physical Review Letters* 23(15), pp. 880–884, doi:10.1103/PhysRevLett.23.880.
- [10] Ehtibar N. Dzhafarov (2019): *The Contextuality-by-Default View of the Sheaf-Theoretic Approach to Contextuality*. doi:10.48550/arXiv.1906.02718.
- [11] Ehtibar N. Dzhafarov & Janne V. Kujala (2013): *All-Possible-Couplings Approach to Measuring Probabilistic Context*. *PLoS ONE* 8(5), p. e61712, doi:10.1371/journal.pone.0061712.
- [12] Ehtibar N. Dzhafarov & Janne V. Kujala (2016): *Contextuality-by-Default 2.0: Systems with Binary Random Variables*. doi:10.48550/arXiv.1604.04799.
- [13] Ehtibar N. Dzhafarov, Janne V. Kujala & Victor H. Cervantes (2015): *Contextuality-by-Default: A Brief Overview of Ideas, Concepts, and Terminology*. *Lecture Notes in Computer Science* 9535, 12–23, 2016, doi:10.1007/978-3-319-28675-4-2.
- [14] Ehtibar N. Dzhafarov, Janne V. Kujala, Víctor H. Cervantes, Ru Zhang & Matt Jones (2016): *On contextuality in behavioural data*. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374(2068), p. 20150234, doi:10.1098/rsta.2015.0234.
- [15] Albert Einstein, Boris Podolsky & Nathan Rosen (1935): *Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?* *Phys. Rev.* 47, pp. 777–780, doi:10.1103/PhysRev.47.777.
- [16] Ali Emami, Noelia De La Cruz, Adam Trischler, Kaheer Suleman & Jackie Chi Kit Cheung (2018): *A Knowledge Hunting Framework for Common Sense Reasoning*. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, pp. 1949–1958, doi:10.18653/v1/D18-1220.
- [17] Arthur Fine (1982): *Hidden Variables, Joint Probability, and the Bell Inequalities*. *Phys. Rev. Lett.* 48, pp. 291–295, doi:10.1103/PhysRevLett.48.291.
- [18] Marissa Giustina, Marijn A.M. Versteegh, Sören Wengerowsky, Johannes Handsteiner, Armin Hochrainer, Kevin Phelan, Fabian Steinlechner, Johannes Kofler, Jan-Åke Larsson, Carlos Abellán, Waldimar Amaya, Valerio Pruneri, Morgan W. Mitchell, Jörn Beyer, Thomas Gerrits, Adriana E. Lita, Lynden K. Shalm,

- Sae Woo Nam, Thomas Scheidl, Rupert Ursin, Bernhard Wittmann & Anton Zeilinger (2015): *Significant-Loophole-Free Test of Bell's Theorem with Entangled Photons*. *Physical Review Letters* 115(25), p. 250401, doi:10.1103/PhysRevLett.115.250401.
- [19] Weinan He, Canming Huang, Yongmei Liu & Xiaodan Zhu (2021): *WinoLogic: A Zero-Shot Logic-based Diagnostic Dataset for Winograd Schema Challenge*. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp. 3779–3789, doi:10.18653/v1/2021.emnlp-main.307.
- [20] B. Hensen, H. Bernien, A. E. Dreaú, A. Reiserer, N. Kalb, M. S. Blok, J. Ruitenberg, R. F. L. Vermeulen, R. N. Schouten, C. Abellán, W. Amaya, V. Pruneri, M. W. Mitchell, M. Markham, D. J. Twitchen, D. Elkouss, S. Wehner, T. H. Taminiua & R. Hanson (2015): *Loophole-free Bell inequality violation using electron spins separated by 1.3 kilometres*. *Nature* 526(7575), pp. 682–686, doi:10.1038/nature15759.
- [21] Mark Howard, Joel Wallman, Victor Veitch & Joseph Emerson (2014): *Contextuality supplies the 'magic' for quantum computation*. *Nature* 510(7505), pp. 351–355, doi:10.1038/nature13460.
- [22] Alexei Y. Kitaev (2003): *Fault-tolerant quantum computation by anyons*. *Annals of Physics* 303(1), pp. 2–30, doi:10.1016/s0003-4916(02)00018-0.
- [23] Simon Kochen & Ernst Specker (1968): *The Problem of Hidden Variables in Quantum Mechanics*. *Indiana Univ. Math. J.* 17(1), pp. 59–87, doi:10.1512/iumj.1968.17.17004.
- [24] Vid Kocijan, Ernest Davis, Thomas Lukasiewicz, Gary Marcus & Leora Morgenstern (2022): *The Defeat of the Winograd Schema Challenge*. doi:10.48550/arXiv.2201.02387.
- [25] Janne V. Kujala & Ehtibar N. Dzhafarov (2019): *Measures of Contextuality and Noncontextuality*. *Philosophical Transactions of the Royal Society A* 377:20190149, 2019 377(2157), p. 20190149, doi:10.1098/rsta.2019.0149.
- [26] Hector J. Levesque, Ernest Davis & Leora Morgenstern (2012): *The Winograd Schema Challenge*. In: *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning, KR'12*, AAAI Press, pp. 552–561, doi:10.5555/3031843.3031909.
- [27] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer & Veselin Stoyanov (2019): *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. doi:10.48550/arXiv.1907.11692.
- [28] Kin Ian Lo, Mehrnoosh Sadrzadeh & Shane Mansfield (2022): *A Model of Anaphoric Ambiguities using Sheaf Theoretic Quantum-like Contextuality and BERT*. *EPTCS* 366, 2022, pp. 23-34, doi:10.4204/EPTCS.366.5.
- [29] Cihan Okay, Aziz Kharoof & Selman Ipek (2022): *Simplicial quantum contextuality*. doi:10.48550/arXiv.2204.06648.
- [30] Robert Raussendorf (2013): *Contextuality in measurement-based quantum computation*. *Physical Review A - Atomic, Molecular, and Optical Physics* 88(2), pp. 1–7, doi:10.1103/PhysRevA.88.022322.
- [31] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula & Yejin Choi (2021): *WinoGrande: An Adversarial Winograd Schema Challenge at Scale*. *Commun. ACM* 64(9), pp. 99–106, doi:10.1145/3474381.
- [32] Lynden K. Shalm, Evan Meyer-Scott, Bradley G. Christensen, Peter Bierhorst, Michael A. Wayne, Martin J. Stevens, Thomas Gerrits, Scott Glancy, Deny R. Hamel, Michael S. Allman, Kevin J. Coakley, Shellie D. Dyer, Carson Hodge, Adriana E. Lita, Varun B. Verma, Camilla Lambrocco, Edward Tortorici, Alan L. Migdall, Yanbao Zhang, Daniel R. Kumor, William H. Farr, Francesco Marsili, Matthew D. Shaw, Jeffrey A. Stern, Carlos Abellán, Waldimar Amaya, Valerio Pruneri, Thomas Jennewein, Morgan W. Mitchell, Paul G. Kwiat, Joshua C. Bienfang, Richard P. Mirin, Emanuel Knill & Sae Woo Nam (2015): *Strong Loophole-Free Test of Local Realism*. *Phys. Rev. Lett.* 115, p. 250402, doi:10.1103/PhysRevLett.115.250402.
- [33] Alan M. Turing (1950): *Computing Machinery and Intelligence*. *Mind* 59(236), pp. 433–460, doi:10.1093/mind/LIX.236.433.
- [34] Daphne Wang (2018): *Distributional Models of Meaning The Contextuality of a Text*. MSc project report.

- [35] Daphne Wang & Mehrnoosh Sadrzadeh (2023): *The Causal Structure of Semantic Ambiguities*. doi:10.48550/arXiv.2206.06807.
- [36] Daphne Wang, Mehrnoosh Sadrzadeh, Samson Abramsky & Victor Cervantes (2021): *On the Quantum-like Contextuality of Ambiguous Phrases*. In: *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, Association for Computational Linguistics, Groningen, The Netherlands, pp. 42–52, doi:10.48550/arXiv.2107.14589.
- [37] Daphne Wang, Mehrnoosh Sadrzadeh, Samson Abramsky & Víctor H. Cervantes (2021): *Analysing Ambiguous Nouns and Verbs with Quantum Contextuality Tools*. *Journal of Cognitive Science* 22(3), pp. 391–420, doi:10.17791/jcs.2021.22.3.391.
- [38] Terry Winograd (1972): *Understanding natural language*. *Cognitive Psychology* 3(1), pp. 1–191, doi:10.1016/0010-0285(72)90002-3.
- [39] Hongming Zhang, Xinran Zhao & Yangqiu Song (2020): *WinoWhy: A Deep Diagnosis of Essential Commonsense Knowledge for Answering Winograd Schema Challenge*. doi:10.48550/arXiv.2005.05763.

Flow-preserving ZX-calculus Rewrite Rules for Optimisation and Obfuscation

Tommy McElvanney

School of Computer Science
University of Birmingham
t.xm639@student.bham.ac.uk

Miriam Backens

School of Computer Science
University of Birmingham
m.backens@cs.bham.ac.uk

In the one-way model of measurement-based quantum computation (MBQC), computation proceeds via measurements on a resource state. So-called flow conditions ensure that the overall computation is deterministic in a suitable sense, with Pauli flow being the most general of these. Computations, represented as measurement patterns, may be rewritten to optimise resource use and for other purposes. Such rewrites need to preserve the existence of flow to ensure the new pattern can still be implemented deterministically. The majority of existing work in this area has focused on rewrites that reduce the number of qubits, yet it can be beneficial to increase the number of qubits for certain kinds of optimisation, as well as for obfuscation.

In this work, we introduce several ZX-calculus rewrite rules that increase the number of qubits and preserve the existence of Pauli flow. These rules can be used to transform any measurement pattern into a pattern containing only (general or Pauli) measurements within the XY-plane. We also give the first flow-preserving rewrite rule that allows measurement angles to be changed arbitrarily, and use this to prove that the ‘neighbour unfusion’ rule of Staudacher et al. preserves the existence of Pauli flow. This implies it may be possible to reduce the runtime of their two-qubit-gate optimisation procedure by removing the need to regularly run the costly gflow-finding algorithm.

1 Introduction

The ZX-calculus is a graphical language for representing and reasoning about quantum computations, allowing us to conveniently represent and reason about computations in both the quantum circuit model and the one-way model of measurement based quantum-computation (MBQC), as well as to translate between the two. The ZX-calculus has various complete sets of rewrite rules, meaning that any two diagrams representing the same linear map can be transformed into each other entirely graphically [1, 14, 15] and provide tools for uses in optimization [24] [12], obfuscation [7] and other areas of research in quantum computing.

The one-way model of MBQC involves the implementation of quantum computations by performing successive adaptive single-qubit measurements on a resource state [21], largely without using any unitary operations. This contrasts with the more commonly-used circuit model and has applications in server-client scenarios as well as for certain quantum error-correcting codes.

An MBQC computation is given as a *pattern*, which specifies the resource state – usually a graph state – and a sequence of measurements of certain types [11]. As measurements are non-deterministic, future measurements need to be adapted depending on the outcomes of past measurements to obtain an overall deterministic computation. Yet not every pattern can be implemented deterministically. Sufficient (and in some cases necessary) criteria for determinism are given by the different kinds of *flow*, which define a partial order on the measured qubits and give instructions for how to adapt the future computation if a measurement yields the undesired outcome [6, 10] (cf. Section 2.3).

In addition to the applications mentioned above, the flexible structure of MBQC patterns is also useful as a theoretical tool. For example, translations between circuits and MBQC patterns have been used to trade off circuit depth versus qubit number [5] or to reduce the number of T -gates in a Clifford+ T circuit [17]. When translating an MBQC pattern (back) into a circuit, it is important that the pattern still have flow, as circuit extraction algorithms rely on flow [3, 10, 12, 20].

ZX-calculus diagrams directly corresponding to MBQC-patterns are said to be in MBQC-form. Many of the standard ZX-calculus rewrite rules do not preserve the MBQC-form structure nor the existence of a flow, which we often want to preserve, thus circuit optimisation using MBQC and the ZX-calculus relies on proofs that preserve both MBQC-form and flow [3, 12]. Much of the previous work on this has focused on rewrite rules that maintain or reduce the number of qubits, which find direct application in T-count optimisation [12]. Nevertheless, it is sometimes desirable to increase the number of qubits in an MBQC pattern while preserving the existence of flow, such as for more involved optimisation strategies [23] or for obfuscation.

In this work we introduce several ZX-calculus rewrite rules that preserve the MBQC-form structure as well as Pauli flow [6], alongside proofs of this preservation. These rules have various applications, such as being used in obfuscation techniques for blind quantum computation [7]. Notably, we introduce the first Pauli flow preserving rewrite rule that allows us to change measurement angles arbitrarily, with all previous rules only allowing for changes that are integer multiples of $\frac{\pi}{2}$. Using this, we prove that the ‘neighbour unfusion’ rule of [24] always preserves the existence of Pauli flow. Additionally, we provide a sufficient and a necessary condition for neighbour unfusion to preserve the existence of gflow [6], a more restricted flow condition.

2 Preliminaries

In this section, we give an overview of the ZX-calculus and then use it to introduce measurement-based quantum computing. We discuss the notion of flow that will be used in this paper and some existing rewrite rules which preserve the existence of this flow.

2.1 The ZX-calculus

The ZX-calculus is a diagrammatic language for reasoning about quantum computations. We will provide a short introduction here; for a more thorough overview, see [8, 25].

A ZX-diagram consists of *spiders* and *wires*. Diagrams are read from left to right: wires entering a diagram from the left are inputs while wires exiting the diagram on the right are outputs, like in the quantum circuit model. ZX-diagrams compose in two distinct ways: *horizontal composition*, which involves connecting the output wires of one diagram to the input wires of another, and *vertical composition* (or the tensor product), which just involves drawing one diagram vertically above the other. The linear map corresponding to a ZX-diagram D is denoted by $\llbracket D \rrbracket$.

ZX-diagrams are generated by two families of spiders which may have any number of inputs or outputs, corresponding to the Z and X bases respectively. Z-spiders are drawn as green dots and X-spiders as red dots; with m inputs, n outputs, and using $(\cdot)^{\otimes k}$ to denote a k -fold tensor power, we have:

$$\left[\begin{array}{c} \text{---} \\ \diagup \quad \diagdown \\ \alpha \\ \diagdown \quad \diagup \\ \text{---} \\ \vdots \end{array} \right] = |0\rangle^{\otimes n} \langle 0|^{\otimes m} + e^{i\alpha} |1\rangle^{\otimes n} \langle 1|^{\otimes m} \quad \left[\begin{array}{c} \text{---} \\ \diagup \quad \diagdown \\ \alpha \\ \diagdown \quad \diagup \\ \text{---} \\ \vdots \end{array} \right] = |+\rangle^{\otimes n} \langle +|^{\otimes m} + e^{i\alpha} |-\rangle^{\otimes n} \langle -|^{\otimes m}$$

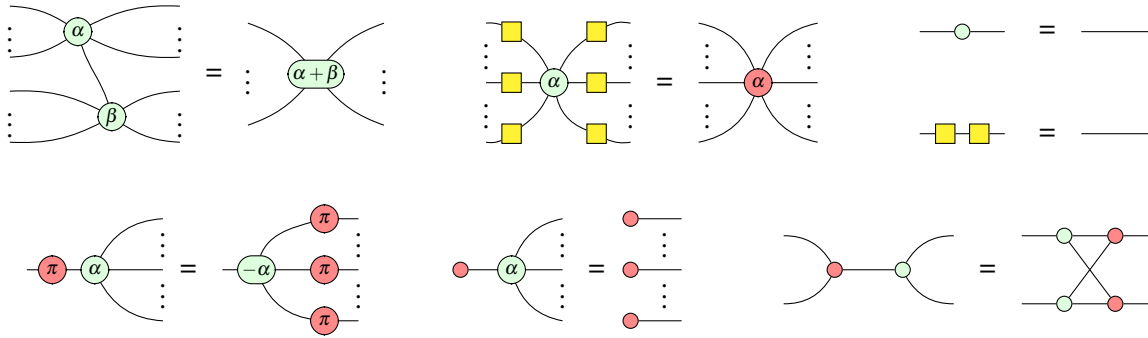


Figure 1: A complete set of rewrite rules for the scalar-free stabilizer ZX-calculus. Each rule also holds with the colours or the directions reversed.

Spiders with exactly one input and output are unitary, in particular $\llbracket -\alpha - \rrbracket = |0\rangle\langle 0| + e^{i\alpha} |1\rangle\langle 1| = Z_\alpha$ and $\llbracket -\alpha \rrbracket = |+\rangle\langle +| + e^{i\alpha} |-\rangle\langle -| = X_\alpha$.

Two diagrams D and D' are said to be equivalent (written $D \cong D'$) if $\llbracket D \rrbracket = z \llbracket D' \rrbracket$ for some non-zero complex number z . For the rest of the paper, whenever we write a diagram equality we will mean equality up to some global scalar in this way. For treatments of the ZX-calculus which do not ignore scalars see [2] for the stabilizer fragment, [15] for the Clifford+T fragment and [14, 16] for the full ZX-calculus.

The Hadamard gate $H = |+\rangle\langle 0| + |-\rangle\langle 1| \cong Z_{\frac{\pi}{2}} \circ X_{\frac{\pi}{2}} \circ Z_{\frac{\pi}{2}}$ will be widely used throughout the paper. It has two common syntactic sugars – a yellow square, or a blue dotted line – with the latter only used between spiders:



The ZX-calculus is equipped with a set of rewrite rules which can be used to transform a ZX-diagram into another diagram representing the same linear map. The following rules, along with the definition of $-\square-$, is complete for stabilizer ZX-diagrams: any two stabilizer ZX-diagrams which correspond (up to non-zero scalar factor) to the same linear map can be rewritten into one another using these rules [1].

2.2 Measurement-based Quantum computation

Measurement-based Quantum computation (MBQC) is a particularly interesting model of quantum computation with no classical analogue. In the one-way model of MBQC, one first constructs a highly entangled resource state that can be independent of the specific computation that one wants to perform (only depending on the ‘size’ of the computation) by preparing qubits in the $|+\rangle$ state and applying CZ-gates to certain pairs of qubits. The computation then proceeds by performing single qubit measurements in a specified order. MBQC is a universal model for quantum computation – any computation can be performed by choosing an appropriate resource state and then performing a certain combination of measurements on said state.

Measurement-based computations are traditionally expressed as *measurement patterns*, which use a sequence of commands to describe how the resource state is constructed and how the computation proceeds [11]. As the resource states are graph states, a graphical representation of MBQC protocols can be more intuitive; we shall therefore introduce MBQC with ZX-diagrams.

Definition 2.1 ([13]). A *graph state diagram* is a ZX-diagram where each vertex is a (phase-free) green spider, each edge connecting spiders has a Hadamard gate on it, and there is a single output wire incident on each vertex.

operator	$\langle +_{XY,\alpha} _i$	$\langle +_{XZ,\alpha} _i$	$\langle +_{YZ,\alpha} _i$	$\langle +_{X,0} _i$	$\langle +_{Y,0} _i$	$\langle +_{Z,0} _i$	$\langle +_{X,\pi} _i$	$\langle +_{Y,\pi} _i$	$\langle +_{Z,\pi} _i$
diagram									

Table 1: MBQC measurement effects in Dirac notation and their corresponding ZX-diagrams

Definition 2.2. [3, Definition 2.18] A ZX-diagram is in *MBQC-form* if it consists of a graph state diagram in which each vertex of the graph may furthermore be connected to an input (in addition to its output), and a measurement effect instead of its output.

MBQC restricts the allowed single-qubit measurements to three planes of the Bloch sphere: those spanned by the eigenstates of two Pauli matrices, called the XY, YZ and XZ planes. Each time a qubit u is measured in a plane $\lambda(u)$ at an angle α , one may obtain either the desired outcome, denoted $\langle +_{\lambda(u),\alpha} |$, or the undesired outcome $\langle -_{\lambda(u),\alpha} | = \langle +_{\lambda(u),\alpha+\pi} |$. Measurements where the angle is an integer multiple of $\frac{\pi}{2}$ are Pauli measurements; the corresponding measurement type is denoted by simply X, Y, or Z. The ZX-diagram corresponding to each (desired) measurement outcome is given in Table 1. The structure of an MBQC protocol is formalised as follows.

Definition 2.3. A *labelled open graph* is a tuple $\Gamma = (G, I, O, \lambda)$, where $G = (V, E)$ is a simple undirected graph, $I \subseteq V$ is a set of input vertices, $O \subseteq V$ is a set of output vertices, and $\lambda : V \setminus O \rightarrow \{X, Y, Z, XY, XZ, YZ\}$ assigns a measurement plane or Pauli measurement to each non-output vertex.

2.3 Pauli flow

Measurement-based computations are inherently probabilistic because measurements are probabilistic. Computations can be made deterministic overall (up to Pauli corrections on the outputs) by tracking which measurements result in undesired outcomes and then correcting for these by adapting future measurements. A sufficient (and in some cases necessary) condition for this to be possible on a given labelled open graph is *Pauli flow*. In the following, $\mathcal{P}(A)$ denotes the powerset of a set A , $N_G(v) = \{w \in V \mid (v, w) \in E\}$ is the set of neighbours of a vertex v in a graph $G = (V, E)$. Furthermore, $\text{Odd}_G(A) = \{v \in V \mid |N_G(v) \cap A| \equiv 1 \pmod{2}\}$ is the set of vertices in the graph $G = (V, E)$ which have an odd number of neighbours in $A \subseteq V$; this is referred to as the odd neighbourhood (in G) of A .

Definition 2.4 ([6, Definition 5]). A labelled open graph (G, I, O, λ) has Pauli flow if there exists a map $p : V \setminus O \rightarrow \mathcal{P}(V \setminus I)$ and a partial order \prec over V such that for all $u \in V \setminus O$,

1. if $v \in p(u)$, $v \neq u$ and $\lambda(v) \notin \{X, Y\}$, then $u \prec v$.
2. if $v \in \text{Odd}_G(p(u))$, $v \neq u$ and $\lambda(v) \notin \{Y, Z\}$, then $u \prec v$.
3. if $\neg(u \prec v)$, $u \neq v$ and $\lambda(v) = Y$, then $v \in p(u) \iff v \in \text{Odd}_G(p(u))$.
4. if $\lambda(u) = XY$, then $u \notin p(u)$ and $u \in \text{Odd}_G(p(u))$.
5. if $\lambda(u) = XZ$, then $u \in p(u)$ and $u \in \text{Odd}_G(p(u))$.
6. if $\lambda(u) = YZ$, then $u \in p(u)$ and $u \notin \text{Odd}_G(p(u))$.
7. if $\lambda(u) = X$, then $u \in \text{Odd}_G(p(u))$.
8. if $\lambda(u) = Z$, then $u \in p(u)$.
9. if $\lambda(u) = Y$ then either $u \in p(u)$ and $u \notin \text{Odd}_G(p(u))$ or $u \notin p(u)$ and $u \in \text{Odd}_G(p(u))$.

Here, the partial order is related to time order in which the qubits need to be measured. The set $p(u)$ denotes qubits that are modified by Pauli-X to compensate for an undesired measurement outcome on u , $\text{Odd}_G(p(u))$ denotes the set of vertices that are modified by Pauli-Z.

Pauli flow is a sufficient condition for strong, stepwise and uniform determinism: this means all branches of the computation should implement the same linear operator up to a phase, any interval of the computation should be deterministic on its own, and the computation should be deterministic for all choices of measurement angles that satisfy λ [6, p. 5]. Pauli flow (and related flow conditions) are particularly interesting from a ZX-calculus perspective as there are polynomial-time algorithms for extracting circuits from MBQC-form ZX-diagrams with flow [3, 12, 22], while circuit extraction from general ZX-diagrams is #P-hard [4].

There are certain Pauli flows where the side effects of any correction are particularly limited, these are called *focused* and they exist whenever a labelled open graph has Pauli flow.

Definition 2.5 (rephrased from [22, Definition 4.3]). Suppose the labelled open graph (G, I, O, λ) has a Pauli flow (p, \prec) . Define $S_u = V \setminus (O \cup \{u\})$ for all $u \in V$. Then (p, \prec) is *focused* if for all $u \in V \setminus O$:

- Any $v \in S_u \cap p(u)$ satisfies $\lambda(v) \in \{XY, X, Y\}$.
- Any $v \in S_u \cap \text{Odd}(p(u))$ satisfies $\lambda(v) \in \{XZ, YZ, Y, Z\}$.
- Any $v \in S_u$ such that $\lambda(v) = Y$ satisfies $v \in p(u)$ if and only if $v \in \text{Odd}(p(u))$.

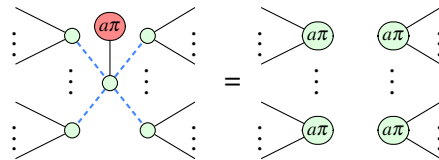
Lemma 2.6 ([22, Lemma 4.6]). *If a labelled open graph has Pauli flow, then it has a focused Pauli flow.*

If the codomain of λ in Definition 2.4 is $\{XY, XZ, YZ\}$, the flow is called a gflow [6]. Similarly, for Definition 2.5, the restriction to measurement labels $\{XY, XZ, YZ\}$ is called a focused gflow [3]. If a labelled open graph has gflow, then it has a focused gflow [3, Proposition 3.14].

2.4 Existing flow-preserving rewrite rules

The basic ZX-calculus rewrite rules in Figure 1 do not generally preserve even the MBQC-form structure of a ZX-calculus diagram. Yet there are some more complex derived rewrite rules that are known to preserve both the MBQC-form structure and the existence of a flow. These rules were previously considered in the context of gflow [12] and extended gflow [3]; the Pauli-flow preservation proofs are due to [22]. The simplest of these rules are Z-deletion and Z-insertion:

Lemma 2.7 ([22, Lemma D.6]). *Deleting a Z-measured vertex preserves the existence of Pauli flow.*



Lemma 2.8 ([18, Proposition 4.1]). *Inserting a Z-measured vertex (i.e. the inverse of Z-deletion) also preserves the existence of Pauli flow.*

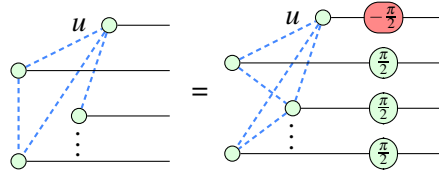
Other rewrite rules are based around quantum generalisations of two graph-theoretic operations.

Definition 2.9. Let $G = (V, E)$ be a graph and $u \in V$. The *local complementation* of G about u is the operation which maps G to $G \star u := (V, E \Delta \{(b, c) \mid (b, u), (c, u) \in E \text{ and } b \neq c\})$, where Δ is the symmetric difference operator given by $A \Delta B = (A \cup B) \setminus (A \cap B)$. The *pivot* of G about the edge (u, v) is the operation mapping G to the graph $G \wedge uv := G \star u \star v \star u$.

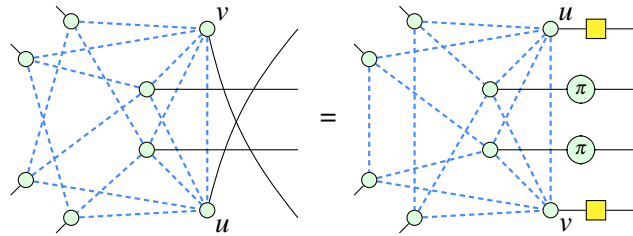
Local complementation keeps the vertices of the graph the same but toggles some edges: for each pair of neighbours of u , i.e. $v, v' \in N_G(u)$, there is an edge connecting v and v' in $G \star u$ if and only if there is no edge connecting v and v' in G . Pivoting is a series of three local complementations about two neighbouring vertices, and is denoted by $G \wedge uv = G \star u \star v \star u$.

Both local complementation and pivoting give rise to operations on MBQC-form diagrams which preserve the MBQC form as well as the existence of Pauli flow (after some simple merging of single-qubit Cliffords into measurement effects, cf. [3, Section 4.2]). We illustrate the operations with examples as they are difficult to express in ZX-calculus in generality.

Lemma 2.10 ([22, Lemma D.12]). *A local complementation about a vertex u preserves the existence of Pauli flow.*



Lemma 2.11 ([22, Lemma D.21]). *A pivot about an edge (u, v) preserves the existence of Pauli flow.*



Observation 2.12. *Lemmas 2.10 and 2.11 provide their own inverses: four successive local complementations about the same vertex or two successive pivots about the same edge leave a diagram invariant.*

3 Converting planar measurements to XY-measurements

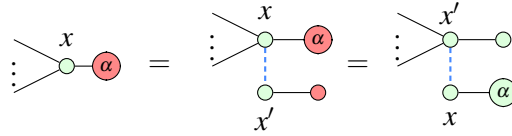
In the *graph-like diagrams* [12] used in PyZX, all spiders are green and all edges are Hadamard edges. ‘Phase gadgets’ consist of a degree-1 green spider connected to a phase-free green spider by a Hadamard edge as in the left-most diagram of (1). When converting graph-like diagrams to MBQC-form, it is difficult to know whether to interpret phase gadgets as a single YZ-measured vertex (middle diagram) or as an X-measured vertex connected to a degree-1 XY-measured vertex (right-most diagram):



The following proposition shows that both interpretations are valid and can be interconverted.

Proposition 3.1. *Let (G, I, O, λ) be a labelled open graph with Pauli flow where $G = (V, E)$, and suppose there exists $x \in V$ with $\lambda(x) = YZ$. Then (G', I, O, λ') has Pauli flow, where $V' = V \cup \{x'\}$, $E' = E \cup \{x, x'\}$, and $\lambda'(x) = X$, $\lambda'(x') = XY$, and $\lambda'(v) = \lambda(v)$ otherwise.*

Proof. Consider the following sequence of rewrites.

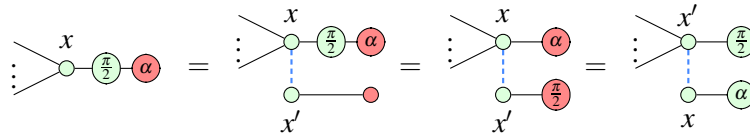


Here we insert the Z -measured vertex x' connected only to x , then pivot along the edge (x, x') . Both Z -insertion and pivoting preserve the existence of Pauli flow, thus our new rewrite rule also preserves the existence of Pauli flow. \square

A similar sequence of rewrites allows us to rewrite XZ -measurements in terms of just Y -measurements and XY -measurements

Proposition 3.2. *Let (G, I, O, λ) be a labelled open graph with Pauli flow where $G = (V, E)$, and suppose there exists $x \in V$ with $\lambda(x) = XZ$. Then (G', I, O, λ') has Pauli flow, where $V' = V \cup \{x'\}$, $E' = E \cup \{(x, x')\}$, and $\lambda'(x) = Y$, $\lambda'(x') = XY$ and $\lambda'(v) = \lambda(v)$ otherwise.*

Proof. Consider the following sequence of rewrites.



Here we insert a Z -measured vertex x' connected to the XZ -measured vertex x , perform local complementation about x' , then pivot along the edge (x, x') . Each of these rewrites preserves the existence of Pauli flow, thus the resulting pattern has Pauli flow. \square

Using the two previous propositions, we are able to re-write any XZ - and YZ -planar measurements into a Pauli measurement plus an XY -measurement. This implies the following.

Proposition 3.3. *Let (G, I, O, λ) be an arbitrary MBQC-form diagram with Pauli flow. Then there exists an equivalent diagram (G', I', O', λ') with Pauli flow where $\lambda'(v) \in \{X, Y, XY\}$ for all $v \in V'$.*

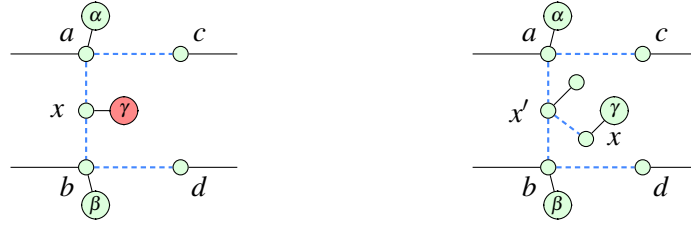
Proof. We begin by applying Z -deletion (Lemma 2.7) to all Z -measured vertices, leaving us with only X, Y, XY, XZ and YZ vertices. It remains to remove all XZ and YZ measurements.

By Proposition 3.1, we can convert every YZ -measured vertex into an X -measured vertex connected to an XY -measured vertex while preserving the existence of Pauli flow. Then, by Proposition 3.2 we can convert every XZ -measured vertex into a Y -measured vertex connected to an XY -measured vertex. We now only have X, Y and XY measured vertices remaining, and each rewrite rule used to get here preserves the existence of Pauli flow, thus the resulting graph has Pauli flow. \square

Remark 3.4. Note that Pauli flow is important here: the gflow conditions need not be satisfied if the newly-introduced Pauli measurements were taken to be arbitrary XY -measurements instead.

For example, the first of the following two diagrams has a gflow (g, \prec) with $g(a) = \{c\}$, $g(b) = \{d\}$, $g(x) = \{c, d, x\}$ and $a, b, x \prec c, d$. The second diagram has Pauli flow by Proposition 3.1, but it does not have gflow: any flow (p, \prec') on the second diagram must have $x \in p(x')$ to satisfy $x' \in \text{Odd}(p(x'))$, as

inputs a, b do not appear in correction sets. Similarly, $x' \in p(x)$ as it is the only neighbour. Thus the gflow conditions would require $x \prec' x'$ and $x' \prec' x$ simultaneously, which is not possible.



4 Subdividing an edge

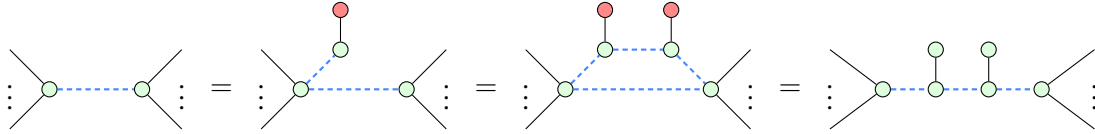
Research on flow-preserving rewrite rules so far has been geared towards optimization, which usually involves reducing the number of vertices in a pattern. Yet there are also cases where it is desirable to introduce new vertices. An example of this is the obfuscation protocol for blind quantum computing of [7], which used an unpublished rewrite rule proved by one of the authors. We give the proof below.

Proposition 4.1. *Let $G = (V, E)$ be a graph with vertices V and edges E . Suppose the labelled open graph (G, I, O, λ) has Pauli flow. Pick an edge $\{v, w\} \in E$ and subdivide it twice, i.e. let $G' := (V', E')$, where $V' := V \cup \{v', w'\}$ contains two new vertices v', w' , and*

$$E' := (E \setminus \{\{v, w\}\}) \cup \{\{v, w'\}, \{w', v'\}, \{v', w\}\}.$$

Then (G', I, O, λ') has Pauli flow, where $\lambda'(v') = \lambda'(w') = X$ and $\lambda'(u) = \lambda(u)$ for all $u \in V \setminus O$.

Proof. We may subdivide an edge by inserting two Z -measured vertices as shown in the following diagram, then pivoting about these two Z -measured vertices.



As inserting Z -measured vertices and pivoting both preserve the existence of Pauli flow, subdividing an edge also preserves the existence of Pauli flow. \square

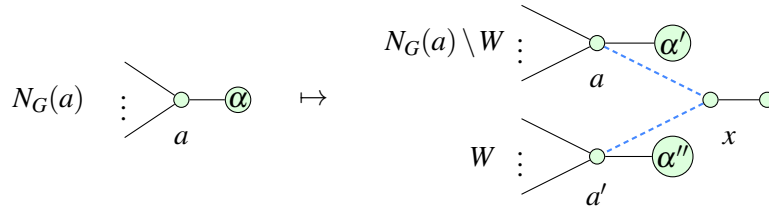
5 Splitting a vertex

Each of the previously mentioned Pauli-flow preserving rewrite rules only changes measurement angles by integer multiples of $\frac{\pi}{2}$. Here we introduce the first Pauli-flow preserving rewrite rule which allows us to change measurement angles arbitrarily. To simplify the proof, the proposition requires that all measurements in the pattern are XY , X or Y ; by Proposition 3.3 this is without loss of generality.

Proposition 5.1. *Let $G = (V, E)$ be a graph with vertices V and edges E . Suppose the labelled open graph (G, I, O, λ) has Pauli flow and satisfies $\lambda(u) \in \{XY, X, Y\}$ for all $u \in O^c$. Pick a vertex $a \in O^c$ such that $\lambda(a) = XY$ and split it, i.e. let $G' := (V', E')$, where $V' := V \cup \{x, a'\}$ contains two new vertices x, a' , and choose some (possibly empty) subset $W \subseteq N(x)$ such that*

$$E' := (E \setminus \{\{a, w\} \mid w \in W\}) \cup \{\{a', w\} \mid w \in W\} \cup \{\{a, x\}, \{x, a'\}\}.$$

Then (G', I, O, λ') has Pauli flow, where $\lambda'(x) = X$, $\lambda'(a') = XY$, and $\lambda'(u) = \lambda(u)$ for all $u \in V \setminus O$.



Proof. Let (p, \prec) be a focused Pauli flow for (G, I, O, λ) ; this exists as the labelled open graph has Pauli flow. Since all measurements are XY , X or Y , the focusing conditions from Definition 2.5 reduce to:

- For all $u \in V \setminus O$, if $v \in \text{Odd}_G(p(u)) \setminus (O \cup \{u\})$ then $\lambda(v) = Y$.
- For all $u \in V \setminus O$ and all $v \in V \setminus (O \cup \{u\})$ such that $\lambda(v) = Y$, we have $v \in p(u) \leftrightarrow v \in \text{Odd}_G(p(u))$.

Now, for all $u \in V \setminus O$, define

$$p'(u) := \begin{cases} p(u) \cup \{x, a'\} & \text{if } a \in p(u) \text{ and } |p(u) \cap W| \equiv 1 \pmod{2} \\ p(u) \cup \{a'\} & \text{if } a \in p(u) \text{ and } |p(u) \cap W| \equiv 0 \pmod{2} \\ p(u) \cup \{x\} & \text{if } a \notin p(u) \text{ and } |p(u) \cap W| \equiv 1 \pmod{2} \\ p(u) & \text{if } a \notin p(u) \text{ and } |p(u) \cap W| \equiv 0 \pmod{2}, \end{cases}$$

then it is straightforward to check that $\text{Odd}_{G'}(p'(u)) = \text{Odd}_G(p(u))$. For example, in the first case, note that \cup can be replaced by Δ since x, a' are new vertices that cannot appear in $p(u)$. Thus

$$\begin{aligned} \text{Odd}_{G'}(p'(u)) &= \text{Odd}_{G'}(p(u) \Delta \{x, a'\}) \\ &= \text{Odd}_{G'}(p(u)) \Delta \text{Odd}_{G'}(\{x, a'\}) \\ &= \text{Odd}_{G'}(a) \Delta \left(\bigtriangleup_{w \in p(u) \setminus (W \cup \{a\} \cup O)} \text{Odd}_{G'}(w) \right) \Delta \left(\bigtriangleup_{w \in (p(u) \cap W) \setminus O} \text{Odd}_{G'}(w) \right) \Delta \{a, x, a'\} \Delta W \\ &= \text{Odd}_G(a) \Delta \left(\bigtriangleup_{w \in p(u) \setminus (W \cup \{a\} \cup O)} \text{Odd}_G(w) \right) \Delta \left(\bigtriangleup_{w \in (p(u) \cap W) \setminus O} \text{Odd}_G(w) \Delta \{a, a'\} \right) \Delta \{a, a'\} \\ &= \text{Odd}_G(p(u)), \end{aligned}$$

where the third step uses $\text{Odd}_{G'}(a) = \text{Odd}_G(a) \Delta W \Delta \{x\}$, and the final step uses $|p(u) \cap W| \equiv 1 \pmod{2}$.

If a is an input in G , then it remains an input in G' (the ‘input’ is not a neighbour so cannot be transferred to a' during the splitting process). This is without loss of generality: if a' is desired to be an input, replace W by $N_G(a) \setminus W$ and swap the labels α' and α'' to get a labelled open graph that is equivalent to the desired one up to relabelling $a \leftrightarrow a'$. Having a be an input is compatible with the correction set for x in the next step below. If a is not an input, there is actually a choice of whether to correct x via a or a' ; we shall always choose the latter for some slight notational convenience.

Let $p'(x) := \{a'\}$, and let $p'(a') := p'(a) \Delta \{x\}$, resulting in the following odd neighbourhoods:

$$\text{Odd}_{G'}(p'(x)) = \text{Odd}_{G'}(\{a'\}) = W \cup \{x\} \tag{2}$$

and

$$\begin{aligned} \text{Odd}_{G'}(p'(a')) &= \text{Odd}_{G'}(p'(a)) \Delta \text{Odd}_{G'}(\{x\}) \\ &= \text{Odd}_G(p(a)) \Delta \{a, a'\} \\ &= (\text{Odd}_G(p(a)) \cup \{a'\}) \setminus \{a\} \end{aligned} \tag{3}$$

where the final step uses the fact that $a \in \text{Odd}_G(p(a))$ and $a' \notin \text{Odd}_G(p(a))$ (since a' is not even in G).

Let \prec' be the transitive closure of

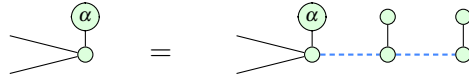
$$\prec \cup \left\{ (w, a') \mid w \prec a \right\} \cup \left\{ (a', w) \mid a \prec w \right\} \cup \left\{ (x, w) \mid w \in W \right\} \cup \left\{ (x, a') \right\}.$$

This is a partial order since a' has the same relationships as a (except for being a successor of x) and x only has successors.

The proof that (p', \prec') is a Pauli flow for G' can be found in Appendix A. \square

We are able to obtain other useful rewrite rules as immediate corollaries of this.

Corollary 5.2. *Using Proposition 5.1 with $W = \emptyset$ and $\alpha'' = 0$, we obtain the following rule used in [7].*

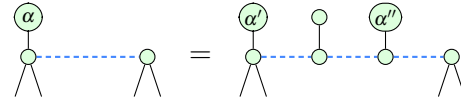


This rule can alternatively be derived in a more round-about way from Z-insertion and pivoting, but we next prove a rule that truly requires vertex splitting.

6 Neighbour unfusion

In [24], a rewrite rule called *neighbour unfusion* was used to reduce the number of two-qubit gates in circuits via the ZX-calculus. Using neighbour unfusion allowed for the two-qubit gate count to be greatly reduced, but introduced a new problem: neighbour unfusion, which introduces two new qubits in each application, was found to not always preserve gflow. Yet a flow is needed to be able to translate back to a circuit after the application of the two-qubit gate count reduction algorithm. We now show that neighbour unfusion preserves the existence of Pauli flow, so circuit re-extraction is always possible.

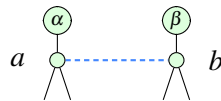
Corollary 6.1. *By applying vertex splitting with $|W| = 1$, we obtain the following ‘neighbour unfusion’ rule, where $\alpha = \alpha' + \alpha''$ (the measurement for the right-most vertex is not drawn as it can be measured in any plane, or even be an output).*



Staudacher et al. [24] state that, in the case of only XY-measurements, neighbour unfusion empirically fails to preserve gflow if the two vertices to which neighbour unfusion is applied are extracted to different qubits in the circuit extraction process. While we have not fully formalised this idea, we give a condition which guarantees that neighbour unfusion preserves gflow.

Proposition 6.2. *Let $\Gamma = (G, I, O, \lambda)$ be a labelled open graph and suppose a, b are two adjacent vertices in G with $\lambda(a) = \lambda(b) = XY$. Suppose Γ has focused gflow (g, \prec) where $b \in g(a)$ and for all $w \in V \setminus \{a, b\}$, $w \prec b \implies w \prec a$ and $a \prec w \implies b \prec w$. Let Γ' be the labelled open graph after applying neighbour unfusion to a and b . Then Γ' has gflow. The same holds with the roles of a and b reversed.*

Proof. Consider a labelled open graph which has a focused gflow (g, \prec) and contains the subdiagram



Assume without loss of generality that $b \in g(a)$ and that for all $w \in V \setminus \{a, b\}$, $w \prec b$ implies $w \prec a$ and $a \prec w$ implies $b \prec w$; the other case is symmetric. Neighbour unfusion yields a labelled open graph $\Gamma' = (G', I, O, \lambda')$ with the following subdiagram, where $\alpha' + \alpha'' = \alpha$:



We can construct a focused gflow for the new pattern by defining the correction sets as follows:

$$g'(v) = \begin{cases} g(v) \cup \{x, a'\} & \text{if } a \in g(v) \wedge b \in g(v) \\ g(v) \cup \{a'\} & \text{if } a \in g(v) \wedge b \notin g(v) \\ g(v) \cup \{x\} & \text{if } a \notin g(v) \wedge b \in g(v) \\ g(b) \cup \{a'\} & \text{if } v = x \\ g(a) & \text{if } v = a' \\ g(v) & \text{otherwise.} \end{cases}$$

This choice leaves invariant the odd neighbourhoods of the correction sets of any original (non-output) vertices. Furthermore we have

$$\text{Odd}_{G'}(g'(x)) = \text{Odd}_{G'}(g(b) \cup \{a'\}) = \text{Odd}_{G'}(g(b) \Delta \{a'\}) = \text{Odd}_G(g(b)) \Delta \{x, b\}$$

since $a, a' \notin g(b)$, and

$$\text{Odd}_{G'}(g'(a')) = \text{Odd}_{G'}(g(a)) = \text{Odd}_G(g(a)) \Delta \{a, a'\}$$

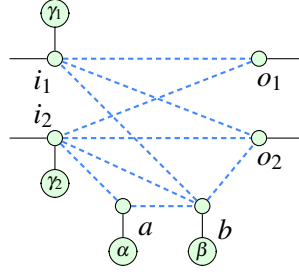
since $b \in g(a)$. Therefore $x \in \text{Odd}_{G'}(g'(x))$ and $a' \in \text{Odd}_{G'}(g'(a'))$ as desired, and furthermore the correction sets for the new vertices are focused. Take \prec' to be the transitive closure of $\prec \cup \{(a, x), (x, a'), (a', b)\}$, then (g', \prec') is a focused gflow for Γ' : Firstly, the relation \prec' is a strict partial order. To show that the gflow conditions are satisfied by (g', \prec') , it suffices to consider that the modified correction sets are compatible with the new partial order:

- For any $v \in V$, we have $a' \in g'(v)$ only if $a \in g(v)$. The latter implies $v \prec a$ and thus $v \prec' a \prec' a'$.
- For any $v \in V$, we have $x \in g'(v)$ only if $b \in g(v)$. The latter implies $v \prec b$ and thus by assumption $v \prec a$. Then, as in the previous case, $v \prec' a \prec' x$.
- If $w \in g'(x)$, then either $w = a'$ or $w \in g(b)$. The former is straightforward as $x \prec' a'$ by definition. For the latter, we have $b \prec w$ since (g, \prec) is a gflow, and thus $x \prec' b \prec' w$.
- If $w \in g'(a')$, then $w \in g(a)$. This implies $a \prec w$ since (g, \prec) is a gflow, and furthermore $b \prec w$ by assumption. Thus, $a' \prec' b \prec' w$.

All of the other gflow conditions are satisfied as (g, \prec) is a gflow for Γ . □

We will illustrate the neighbour unfusion process with an example that shows some choices of unfusion which do preserve gflow and others which do not.

Example 6.3. Consider the following MBQC-form diagram, which appeared in a different context in [20].



This has a focused gflow (g, \prec) with $g(i_1) = \{a, o_2\}$, $g(i_2) = \{a, b, o_2\}$, $g(a) = \{b, o_2\}$ and $g(b) = \{o_1, o_2\}$ and partial order $i_1, i_2 \prec a \prec b \prec o_1, o_2$. Then neighbour unfusion along one of the edges (i_2, a) , (a, b) or (b, o_2) preserves gflow by Proposition 6.2.

On the other hand, the pair (i_2, o_2) satisfies the condition $o_2 \in g(i_2)$ but satisfies neither $w \prec o_2 \implies w \prec i_2$ nor $i_2 \prec w \implies o_2 \prec w$ since a and b sit in between the pair in the partial order. Applying neighbour unfusion to i_2 and o_2 does not preserve the existence of gflow since the odd neighbourhood of $\{o_1, o_2\}$ would become $\{i_2, i_2', b\}$ and thus no vertex can be corrected solely using the outputs in the resulting diagram. An analogous argument holds for the pair (i_2, o_1) for which $o_1 \notin g(i_2)$.

We now show that, for MBQC-form diagrams with equal numbers of inputs and outputs, the condition $b \in g(a)$ (or instead $a \in g(b)$) is necessary for neighbour unfusion to preserve gflow.

Proposition 6.4. *Let $\Gamma = (G, I, O, \lambda)$ be a labelled open graph with $|I| = |O|$ with focused gflow (g, \prec) . Suppose a, b are two adjacent vertices in G with $\lambda(a) = \lambda(b) = \text{XY}$. Let Γ' be the labelled open graph after applying neighbour unfusion to a and b in Γ , and suppose Γ' has gflow. Then we must have $b \in g(a)$ or $a \in g(b)$ for the focused gflow on Γ .*

Proof. Suppose we have a pattern Γ' with the subdiagram (4) and assume that Γ' has a focused gflow (g', \prec') . Let $\Gamma'' = (G'', I, O, \lambda'')$ be the induced sub-pattern containing only those vertices of G' that are either outputs or measured in the XY-plane; this must include all inputs since those cannot be measured in planes XZ or YZ. This new measurement pattern still contains the subdiagram (4) and it has gflow [3, Lemma 3.15]. In fact, since (g', \prec') is focused, it implicitly follows from [3, Proposition 3.14 and Lemma 3.15] that the gflow of the new pattern is just the restriction of the old gflow function to a smaller domain, and this is still focused; denote it by (g'', \prec'') .

Now every focused gflow in a pattern with only XY-plane measurements and equal numbers of inputs and outputs can be reversed in a very strict sense: Let $\Gamma''' = (G''', O, I, \lambda''')$ be the reversed pattern with the roles of inputs and outputs swapped and λ''' mapping all non-outputs to XY. Then there exists a focused gflow $(g'''_{rev}, \prec'''_{rev})$ for Γ''' where \prec'''_{rev} is the reverse of \prec'' and $u \in g'''_{rev}(v)$ if and only if $v \in g''(u)$ [19], cf. also [3, Corollary 2.47].

As x is XY-measured and has two neighbours, to satisfy $x \in \text{Odd}_{G''}(g''(x))$ and $x \in \text{Odd}_{G''}(g'''_{rev}(x))$ we require the following to hold, where \oplus is the exclusive-or operator:

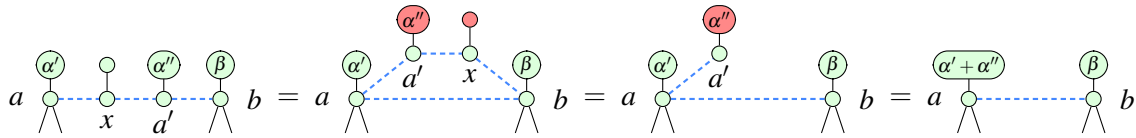
$$(a \in g''(x) \wedge x \in g''(x')) \oplus (x' \in g''(x) \wedge x \in g''(a)).$$

As a' is also XY-measured and has two neighbours, by the same reasoning we obtain:

$$(b \in g''(a') \wedge a' \in g''(x)) \oplus (x \in g''(a') \wedge a' \in g''(b)).$$

Then, as we cannot have both $x \in g''(a')$ and $a' \in g''(x)$, we have either that $a \in g''(x)$, $x \in g''(a')$ and $a' \in g''(b)$ or that $b \in g''(a')$, $a' \in g''(x)$ and $x \in g''(a)$ for (g'', \prec'') to be a gflow for Γ'' . But (g'', \prec'') is the restriction of (g', \prec') to the XY-measured vertices in Γ' . Thus either $a \in g'(x)$, $x \in g'(a')$ and $a' \in g'(b)$ or that $b \in g'(a')$, $a' \in g'(x)$ and $x \in g'(a)$.

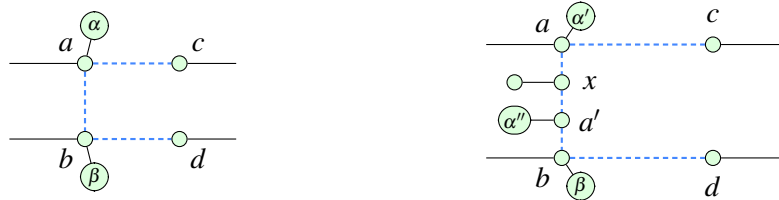
Now, consider the following sequence of rewrites corresponding to the inverse of neighbour unfusion:



where we first pivot along the edge (x, a') , then apply Z -deletion to x' and finally apply the phase gadget identity rule of [17] to add the phase of x' to that of a . Each of these rules preserves the existence of gflow, thus the inverse of neighbour unfusion preserves the existence of gflow. Moreover, if $x \in g'(a)$, $x' \in g'(x)$ and $b' \in g'(x')$, then after applying the inverse of neighbour unfusion we get a focused gflow (g, \prec) for Γ with $b \in g(a)$ (and similarly if $a \in g'(x)$, $x \in g'(x')$ and $x' \in g'(b)$ we get a focused gflow with $a \in g(b)$). Therefore, if the measurement pattern after neighbour unfusion has gflow, then the original pattern has a focused gflow where b is in the correction set of a , or a focused gflow where a is in the correction set of b . \square

An analogous argument to the above works if b is an output, in which case the only option is for b to be in the correction set of a . Therefore the above proposition covers all the cases relevant to Staudacher et al.'s work on patterns where all measurements are in the XY-plane.

Example 6.5. The following two measurement patterns are related by neighbour unfusion along the edge between vertices a and b :



In the first pattern, a and b are both inputs and thus cannot appear in correction sets. Hence the pattern does not have a gflow where a is in the correction set of b or where b is in the correction set of a . Yet it does have a gflow (g, \prec) with $g(a) = \{c\}$, $g(b) = \{d\}$ and $a, b \prec c, d$.

For the second pattern to have a flow (p, \prec') , we require $a' \in p(x)$ and $x \in p(a')$ since both vertices need to be in the odd neighbourhood of their correction set and inputs cannot appear in correction sets. This diagram can therefore not have a gflow, as the gflow conditions would require that $x \prec' a'$ and $a' \prec' x$ simultaneously, so \prec' would not be strict. This diagram does have a Pauli flow however, as the X -measured vertex x does not need to come after a' in the partial order in the case of Pauli flow. The Pauli flow satisfies $p(a) = \{c\}$, $p(b) = \{d\}$, $p(x) = \{d, a'\}$ and $p(a') = \{c, x\}$ with $x \prec a, b, a' \prec c, d$.

The sufficient condition for neighbour unfusion to preserve gflow in Proposition 6.2 and the necessary condition in Proposition 6.4 do not quite match up: we leave the question of a condition that is both necessary and sufficient to future work.

7 Conclusion

We have introduced several rewrite rules which preserve the existence of Pauli flow, including the first flow-preserving rewrite rule which allows us to change phases arbitrarily, rather than just by multiples of $\frac{\pi}{2}$. An immediate corollary of this rule preserving Pauli flow is that the neighbour unfusion rule of [24] also preserves Pauli flow, potentially leading to a reduced runtime for their two-qubit gate count reduction algorithm.

At present, the circuit extraction algorithm for diagrams with Pauli flow introduces more two-qubit gates than the corresponding circuit extraction algorithm for diagrams with gflow – future work could involve using known work on Pauli gadget optimization, such as that of [9], to reduce the number of two-qubit gates obtained when performing circuit extraction on diagrams with Pauli flow.

Other future work could involve finding an analogous result to the stabiliser completeness proof of [18] for a more general fragment of the MBQC-form ZX-calculus, using Proposition 5.1 to introduce phases that are not just integer multiples of $\frac{\pi}{2}$.

Acknowledgements

We would like to thank Korbinian Staudacher and Shuxiang Cao for bringing the topic of rewriting measurement patterns to add new qubits to our attention. Additional thanks to Korbinian Staudacher for providing a counterexample to our original claim in Section 6 and for countless discussions during the development of this paper. We would also like to thank Will Simmons for useful conversations on related topics. Finally, thanks to Piotr Mitosek for suggesting the formulation of the sufficient condition for neighbour unfusion to preserve gflow, and to Simon Perdrix for pointing out an edge case in Proposition 5.1, which is now treated correctly.

References

- [1] Miriam Backens (2014): *The ZX-calculus is complete for stabilizer quantum mechanics*. *New Journal of Physics* 16(9), p. 093021, doi:10.1088/1367-2630/16/9/093021.
- [2] Miriam Backens (2015): *Making the stabilizer ZX-calculus complete for scalars*. *Electronic Proceedings in Theoretical Computer Science* 195, p. 17–32, doi:10.4204/eptcs.195.2.
- [3] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and back again: A circuit extraction tale*. *Quantum* 5, p. 421, doi:10.22331/q-2021-03-25-421.
- [4] Niel de Beaudrap, Aleks Kissinger & John van de Wetering (2022): *Circuit Extraction for ZX-Diagrams Can Be #P-Hard*. In Mikołaj Bojańczyk, Emanuela Merelli & David P. Woodruff, editors: *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 229, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 119:1–119:19, doi:10.4230/LIPIcs.ICALP.2022.119.
- [5] Anne Broadbent & Elham Kashefi (2009): *Parallelizing quantum circuits*. *Theoretical Computer Science* 410(26), pp. 2489–2510, doi:10.1016/j.tcs.2008.12.046.
- [6] Daniel E Browne, Elham Kashefi, Mehdi Mhalla & Simon Perdrix (2007): *Generalized flow and determinism in measurement-based quantum computation*. *New Journal of Physics* 9(8), p. 250–250, doi:10.1088/1367-2630/9/8/250.
- [7] Shuxiang Cao (2022): *Multi-agent blind quantum computation without universal cluster state*, doi:10.48550/ARXIV.2206.13330.
- [8] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, doi:10.1017/9781316219317.

- [9] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons & Seyon Sivarajah (2020): *Phase Gadget Synthesis for Shallow Circuits*. *Electronic Proceedings in Theoretical Computer Science* 318, pp. 213–228, doi:10.4204/eptcs.318.13.
- [10] Vincent Danos & Elham Kashefi (2006): *Determinism in the one-way model*. *Phys. Rev. A* 74, p. 052310, doi:10.1103/PhysRevA.74.052310.
- [11] Vincent Danos, Elham Kashefi & Prakash Panangaden (2005): *Parsimonious and robust realizations of unitary maps in the one-way model*. *Physical Review A* 72(6), p. 064301, doi:10.1103/PhysRevA.72.064301.
- [12] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.
- [13] Ross Duncan & Simon Perdrix (2009): *Graph States and the Necessity of Euler Decomposition*. In Klaus Ambos-Spies, Benedikt Löwe & Wolfgang Merkle, editors: *Mathematical Theory and Computational Practice*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 167–177, doi:10.1007/978-3-642-03073-4_18.
- [14] Amar Hadzihasanovic, Kang Feng Ng & Quanlong Wang (2018): *Two complete axiomatisations of pure-state qubit quantum computing*. In: *LICS '18: 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, doi:10.1145/3209108.3209128.
- [15] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *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*, Association for Computing Machinery, New York, NY, USA, p. 559–568, doi:10.1145/3209108.3209131.
- [16] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics*. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '18*, Association for Computing Machinery, New York, NY, USA, p. 569–578, doi:10.1145/3209108.3209139.
- [17] Aleks Kissinger & John van de Wetering (2020): *Reducing the number of non-Clifford gates in quantum circuits*. *Physical Review A* 102(2), p. 022406, doi:10.1103/PhysRevA.102.022406.
- [18] Tommy McElvanney & Miriam Backens (2022): *Complete flow-preserving rewrite rules for MBQC patterns with Pauli measurements*, doi:10.48550/ARXIV.2205.02009. To appear in *Proceedings QPL '22*.
- [19] Mehdi Mhalla, Mio Muraio, Simon Perdrix, Masato Someya & Peter S Turner (2011): *Which graph states are useful for quantum information processing?* In: *Conference on Quantum Computation, Communication, and Cryptography*, Springer, pp. 174–187, doi:10.1007/978-3-642-54429-3_12.
- [20] Jisho Miyazaki, Michal Hajdušek & Mio Muraio (2015): *Analysis of the trade-off between spatial and temporal resources for measurement-based quantum computation*. *Physical Review A* 91(5), p. 052302, doi:10.1103/PhysRevA.91.052302.
- [21] Robert Raussendorf & Hans J. Briegel (2001): *A One-Way Quantum Computer*. *Phys. Rev. Lett.* 86, pp. 5188–5191, doi:10.1103/PhysRevLett.86.5188.
- [22] Will Simmons (2021): *Relating Measurement Patterns to Circuits via Pauli Flow*. In Chris Heunen & Miriam Backens, editors: *Proceedings 18th International Conference on Quantum Physics*

and Logic, Gdansk, Poland, and online, 7-11 June 2021, *Electronic Proceedings in Theoretical Computer Science* 343, Open Publishing Association, pp. 50–101, doi:10.4204/EPTCS.343.4.

- [23] Korbinian Staudacher (2021): *Optimization Approaches for Quantum Circuits using ZX-calculus*. Master’s thesis, Ludwig-Maximilians-Universität, München. Available at <https://www.mnm-team.org/pub/Diplomarbeiten/stau21/PDF-Version/stau21.pdf>.
- [24] Korbinian Staudacher, Tobias Guggemos, Sophia Grundner-Culemann & Wolfgang Gehrke (2022): *Reducing 2-qubit gate count for ZX-calculus based quantum circuit optimization*. Available at <https://elib.dlr.de/188470/>. To appear in Proceedings QPL ’22.
- [25] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*, doi:10.48550/ARXIV.2012.13966.

A Splitting a vertex preserves the existence of Pauli flow

The following case distinction forms part of the proof of Proposition 5.1.

Let G' , p' and \prec' be defined as in Proposition 5.1. We shall show that (p', \prec') satisfy the nine conditions of Pauli flow.

Claim 1: For all $u \in O^c$, if $v \in p'(u)$ and $u \neq v$ and $\lambda'(v) \notin \{X, Y\}$, then $u \prec' v$.

- For original vertices $u \in V \setminus O$, $v \in p'(u)$ implies $v \in p(u)$ or $v \in \{x, a'\}$. If $v \in p(u)$ then either $u \prec v$ and thus $u \prec' v$ by the definition of \prec' or $\lambda(u) = \lambda'(u) \in \{X, Y\}$. If $v = x$ then $\lambda'(v) = X$ and thus we don’t need to consider this case. Finally, if $v = a'$ then $a \in p(u)$, thus $u \prec a$ which gives us $u \prec' v = a'$ by the definition of \prec' .
- For $u = x$, the only element of $p(x)$ is a' and we have $x \prec' a'$.
- For $u = a'$, $v \in p'(a')$ implies $v \in p(a)$ or $v = x$. For the latter case, $\lambda'(x) = X$ and thus we do not need to consider this. In the former case, $v \in p(a)$ gives us that $a \prec v$ or $\lambda(v) = \lambda'(v) \in \{X, Y\}$ as (p, \prec) is a Pauli flow. So either $a' \prec v$ by the definition of \prec' or the property is trivial anyway.

Claim 2: For all $u \in O^c$, if $v \in \text{Odd}_{G'}(p'(u))$ and $u \neq v$ and $\lambda'(v) \notin \{Y, Z\}$, then $u \prec' v$.

- For original vertices $u \in V \setminus O$, we have defined p' in such a way that $\text{Odd}_{G'}(p'(u)) = \text{Odd}_G(p(u))$, thus this property is inherited from (p, \prec) being a Pauli flow.
- For $u = x$, we have $\text{Odd}_{G'}(p'(x)) = W \cup \{x\}$, and $x \prec' w$ for any $w \in W$ by the definition of \prec' .
- For $u = a'$, $v \in \text{Odd}_{G'}(p'(a'))$ and $v \neq a'$ implies $v \in \text{Odd}_G(p(a))$ by (3). As a' has the same successors as a , we get that $a' \prec' v$ as desired.

Claim 3: For all $u \in O^c$, if $\neg(u \prec' v)$ and $u \neq v$ and $\lambda'(v) = Y$, then $v \in p'(u) \iff v \in \text{Odd}_{G'}(p'(u))$.

- For original vertices $u \in V \setminus O$, this is inherited from (p, \prec) , as the only changes to correction sets and odd neighbourhoods involve adding or removing x or a' , which are not Y -measured.
- For $u = x$, we have $p'(x) \cup \text{Odd}_{G'}(p'(x)) = W \cup \{x, a'\}$. By the definition of the partial order, $x \prec a'$ and $x \prec' w$ for all $w \in W$, so the claim is trivially true.

- For $u = a'$,

$$\lambda'(v) = Y \text{ and } v \in p'(a') \text{ implies } v \in p(a).$$

As (p, \prec) is a focused Pauli flow, we must have $v \in \text{Odd}_G(p(a))$, thus $v \in \text{Odd}_{G'}(p'(a))$ and finally $v \in \text{Odd}_{G'}(p'(a'))$ by (3).

For the other direction,

$$\lambda'(v) = Y \text{ and } v \in \text{Odd}_{G'}(p'(a')) \text{ implies } v \in \text{Odd}_G(p(a)).$$

As (p, \prec) is a Pauli flow, $v \in p(a)$ thus $v \in p'(a)$ and finally $v \in p'(a')$, as desired.

Claim 4: For all $u \in O^c$, if $\lambda'(u) = XY$, then $u \notin p'(u)$ and $u \in \text{Odd}_{G'}(p(u))$.

- If $u = a'$, then this is true by inspection.
- If $u = x$, then the claim is true trivially since $\lambda'(x) \neq XY$.
- If $u \in V \setminus O$, then this property is inherited from (p, \prec) .

Claim 5: For all $u \in O^c$, if $\lambda'(u) = XZ$, then $u \in p'(u)$ and $u \in \text{Odd}_{G'}(p'(u))$.

- This is trivially true as we have no XZ -measured vertices.

Claim 6: For all $u \in O^c$, if $\lambda'(u) = YZ$, then $u \in p'(u)$ and $u \notin \text{Odd}_{G'}(p'(u))$.

- This is trivially true as we have no YZ -measured vertices.

Claim 7: For all $u \in O^c$, if $\lambda'(u) = X$, then $u \in \text{Odd}_{G'}(p'(u))$.

- if $u = a'$, then this is true trivially since $\lambda'(a') \neq X$.
- If $u = x$, then the claim is true by (2).
- If $u \in V \setminus O$, then this property is inherited from (p, \prec) .

Claim 8: For all $u \in O^c$, if $\lambda'(u) = Z$, then $u \in p'(u)$.

- This is trivially true as we have no Z -measured vertices.

Claim 9: For all $u \in O^c$, if $\lambda'(u) = Y$, then $u \in p'(u)$ and $u \notin \text{Odd}_G(p'(u))$, or $u \notin p'(u)$ and $u \in \text{Odd}_G(p'(u))$.

- This is true trivially for a, x and a' , and inherited for all other vertices.

All nine properties are satisfied, therefore (p', \prec') is a Pauli flow for G' .

The Qupit Stabiliser ZX-travaganza: Simplified Axioms, Normal Forms and Graph-Theoretic Simplification

Boldizsár Poór

University of Oxford

Quantinuum, 17 Beaumont Street
Oxford, OX1 2NA, United Kingdom

`boldizsar.poor@quantinuum.com`

Robert I. Booth

University of Edinburgh

`robert.booth@ed.ac.uk`

Titouan Carette

Centre for Quantum Computer Science,
Faculty of Computing, University of Latvia,
Raina 19, Riga, Latvia, LV-1586

`titouan.carette@lu.lv`

John van de Wetering

University of Amsterdam

`john@vdwetering.name`

Lia Yeh

University of Oxford

Quantinuum, 17 Beaumont Street
Oxford, OX1 2NA, United Kingdom

`lia.yeh@cs.ox.ac.uk`

We present a smorgasbord of results on the stabiliser ZX-calculus for odd prime-dimensional qudits (i.e. *qupits*). We derive a simplified rule set that closely resembles the original rules of qubit ZX-calculus. Using these rules, we demonstrate analogues of the spider-removing local complementation and pivoting rules. This allows for efficient reduction of diagrams to the *affine with phases* normal form. We also demonstrate a reduction to a unique form, providing an alternative and simpler proof of completeness. Furthermore, we introduce a different reduction to the *graph state with local Cliffords* normal form, which leads to a novel layered decomposition for qupit Clifford unitaries. Additionally, we propose a new approach to handle scalars formally, closely reflecting their practical usage. Finally, we have implemented many of these findings in `DiZX`, a new open-source Python library for qudit ZX-diagrammatic reasoning.

1 Introduction

A helpful tool to reason about quantum computation is the *ZX-calculus* [22, 21], a graphical language which can represent any qubit computation. It has been used, for example, in measurement-based quantum computing [36, 4, 53], error-correcting codes [34, 37, 29], quantum circuit optimisation [7, 33, 50], classical simulation [51, 19, 52], quantum natural language processing [20, 54], quantum chemistry [61], and quantum machine learning [67, 74].

All the above results use the *qubit* ZX-calculus, but recent years have seen a surge of interest in studying quantum computation using d -dimensional systems, called *qudits*. Qudit-based quantum computation has been experimentally realised in a variety of physical systems, such as ion traps [60, 45], photonic devices [18], and superconducting devices [11, 69, 71, 44, 40].

On the theory side, there has been work in translating work on qubits to qudits in quantum algorithms [68], fault-tolerant quantum computing [42, 15], quantum communication [25], and more [31, 38, 12, 55].

This raises the question of how we can use the ZX-calculus to reason about qudit systems. There exist several variations of the ZX-calculus that extend it to higher-dimensional qudits. Many have focused on the specific case of qutrit systems [65, 39, 65, 62], with applications in quantum computation [70, 63], and complexity theory [62]. Recent papers have focused on the stabiliser fragment of odd prime dimensional qudits, including Ref. [24] that explores error correction and detection in this context, and also Ref. [13] mentioned below. Some proposals capture all finite or infinite dimensions [59, 66, 57, 30], but lack many of the nicer features of the qubit calculus. Of particular importance to our paper is Ref. [13], which constructs a calculus for odd prime dimensions while retaining many of these desirable properties and establishing completeness for the stabiliser fragment. Despite these advancements, practical utilisation of the rewrites in these calculi has received limited attention, leaving room for further exploration and development.

To understand the usefulness of rewrite rules, we can take a look at the original qubit calculus. In qubit ZX, we can distinguish between ‘standard’ rules — spider fusion, identity removal, state copying, bialgebra, and colour change — and ‘harder’ rules — complementarity, Euler angle colour permutation, and the rules dealing with the triangle generator. The standard rules, with minor modifications, were those originally discovered [21], and they are the most commonly used in practice. For instance, all the rewrites used in the PyZX compiler [49] can be proved using just these standard rules [33]. These rules are sufficient to prove completeness for the *stabiliser fragment* of the ZX-calculus [1], while the harder rules were developed to prove completeness for larger fragments. This suggests that carefully studying the qudit stabiliser fragment could be a fruitful avenue for developing useful qudit ZX rewrite rules.

Recall that the stabiliser fragment corresponds to Clifford computation, which is an efficiently simulable subset of quantum computation [41] that forms the basis of many quantum protocols, such as error-correcting codes [48, 47], superdense coding [10], quantum teleportation [9], and quantum key distribution [8]. Completeness of the qubit stabiliser fragment of ZX was proved in [1], while for qutrits it was proved in [65]. Recently, completeness was proved for the stabiliser fragment for any odd-dimensional prime qudit dimension in [13]. The proofs of all these results work essentially the same way: first, they show that any state diagram can be reduced to a Graph State with Local Cliffords (GSLC), and then they show that any pair of GSLCs implementing the same state can be rewritten to a common reduced form.

In this paper, we take this last complete calculus for prime-dimensional qudits [13] as a starting point, and extend it in several ways:

1. We simplify the rules to a smaller set that has a clearer relation to the original qubit stabiliser calculus, and for most of which we can prove the necessity.
2. We incorporate a well-tempered axiomatisation for our calculus following the convention of [27], removing most of the scalars in our rewrite rules, and thus, simplifying our calculations.
3. We introduce a new approach to handle scalars, formalising the often-used convention of writing scalar numbers alongside diagrams.
4. We discover the qupit versions of the spider-removing *local complementation* and *pivoting* rules found in [33] and generalised to qutrits in [62]. These rules serve as the foundation

for optimisation and simulation strategies in the qubit setting [33, 50, 7, 49]. Our findings demonstrate that these strategies can be adapted to work for prime-dimensional qudits, thus extending their applicability beyond qubits.

5. Using these rewrite rules, we simplify the original completeness proof of [13] by reducing the number of case distinctions required.¹ Specifically, we demonstrate that these rewrites reduce diagrams to a normal form that we call the *affine with phases* (AP) form, which originally appeared in [32]. Then, given an AP-form diagram, we show how to reduce it further to a unique form, resulting in completeness.²
6. Additionally, we demonstrate how to rewrite diagrams into a *graph-state with local Cliffords* (GSLC) form, which yields a layered decomposition for Clifford unitaries similar to the one proposed for qubits in [33].

Our findings highlight that qupit stabiliser diagrams share many familiar properties with their qubit counterparts. Furthermore, many results regarding optimisation and normal forms extend seamlessly to the odd prime-dimensional qudit setting.

Finally, we have implemented many of these findings in `DiZX`, a new open-source Python library for qudit ZX-diagrammatic reasoning based on `PyZX` [49].³

Related work Subsequent to submission, we were made aware of a related, parallel work, Ref. [28], which also concerns well-tempered axiomatisations for qudit ZX-calculi.

2 The qupit Clifford ZX-calculus

In this section, we introduce the qudit stabiliser ZX-calculus for odd prime dimensions.

We let p denote an arbitrary odd prime, and $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ the ring of integers modulo p . Since p is prime, \mathbb{Z}_p is a field, implying that every non-zero element in \mathbb{Z}_p has a multiplicative inverse. We denote the group of units (i.e. invertible elements) as $\mathbb{Z}_p^* := \mathbb{Z}_p \setminus \{0\}$. We also define the Legendre symbol, for $x \in \mathbb{Z}_p^*$, as follows:

$$\left(\frac{x}{p}\right) = \begin{cases} 1 & \text{if } \exists y \in \mathbb{Z}_p^* \text{ s.t. } x = y^2; \\ -1 & \text{otherwise;} \end{cases} \quad (1)$$

The Hilbert space of a qupit is $\mathcal{H} = \text{span}\{|m\rangle \mid m \in \mathbb{Z}_p\} \cong \mathbb{C}^p$. Letting $\omega := e^{i\frac{2\pi}{p}}$ be a p -th primitive root of unity, we can write down the following standard operators Z and X , occasionally known as the *clock* and *shift* operators: $Z|m\rangle := \omega^m|m\rangle$ and $X|m\rangle := |m+1\rangle$ for any $m \in \mathbb{Z}_p$. Notably, $ZX = \omega XZ$.

A *Pauli operator* is defined as any operator of the form $\omega^k X^a Z^b$ for $k, a, b \in \mathbb{Z}_p$. We consider Pauli operator *trivial* if it is proportional to the identity. Each Pauli operator has a spectrum given by $\{\omega^k \mid k \in \mathbb{Z}_p\}$, and we denote $|k : Q\rangle$ as the eigenvector of a Pauli operator Q associated with the eigenvalue ω^k . It follows from the definition of Z that we can identify $|k : Z\rangle = |k\rangle$.

¹In addition to being aesthetically and ergonomically preferable, reducing the number of case distinctions also makes the proof more easily verifiable. During the preparation of this manuscript, we identified and communicated several errors and omissions in [13], which were subsequently fixed.

²A similar normal form for qubits was independently found in [53]. It is worth noting that our formulation was already employed for qubits in the Oxford Quantum Software course prior to the preprint [53] appeared online.

³See <https://github.com/jvdwetering/dizx>.

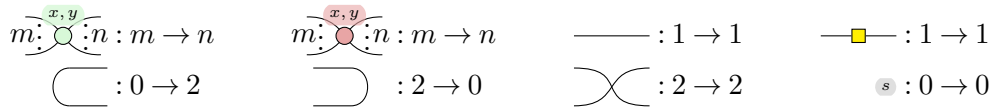
The collection of all Pauli operators is denoted \mathcal{P}_1 and called the *Pauli group*. For $n \in \mathbb{N}^*$, the *generalised Pauli group* \mathcal{P}_n is defined as $\bigotimes_{k=1}^n \mathcal{P}_1$. Of particular importance to us are the (*generalised*) *Clifford groups*. These groups are defined for each $n \in \mathbb{N}^*$ as the (unitary) normaliser of \mathcal{P}_n . In other words, a unitary operator C on $\mathcal{H}^{\otimes n}$ belongs to the Clifford group if, for any $P \in \mathcal{P}_n$, the conjugation CPC^\dagger is also an element of \mathcal{P}_n . While every Pauli operator is Clifford, there exist non-Pauli Clifford operators.

In the case of prime qudit dimensions, the group of Clifford unitaries can be generated by three gates: the *Hadamard gate* defined as $H := \sum_{k \in \mathbb{Z}_p} |k : Z\rangle\langle k : X|$, the *S gate* defined as $S := \sum_{k \in \mathbb{Z}_p} \omega^{2^{-1}k(k-1)} |k : Z\rangle\langle k : Z|$, and the *CX gate* defined as $CX := \sum_{j,k \in \mathbb{Z}_p} |j, j+k : Z\rangle\langle j, k : Z|$ [42]. Note that in this context the Hadamard gate is sometimes also just called the *Fourier transform*.

Stabiliser quantum mechanics is operationally described as a fragment of quantum mechanics where the allowed operations include initialisations and measurements in the eigenbases of Pauli operators, as well as unitary operations from the generalised Clifford groups.

2.1 Generators

We define the symmetric monoidal category $\mathbb{Z}\mathbb{X}_p^{\text{Stab}}$ as having objects \mathbb{N} and morphisms generated by the following diagrams, for any $x, y \in \mathbb{Z}_p$ and $s \in \mathbb{C}$:



In addition to the “standard” generators of $\mathbb{Z}\mathbb{X}$, we have introduced a new generator represented by a light-grey bubble with a scalar written inside it, which we refer to as an *explicit scalar*. These explicit scalars offer a convenient way to streamline the often cumbersome reasoning related to scalars that is typically involved in many graphical completeness papers. Note that the presence of the red X-spider as a generator is in principle unnecessary since the Z-spider surrounded by Hadamard boxes is equivalent to it. However, our goal is not to provide a minimal set of generators, but rather a convenient one.

Diagrams in our framework can be composed in two ways: sequentially, by connecting output wires to input wires, or vertically, by “stacking” diagrams, corresponding to the tensor product operation which is defined as $n \otimes m = n + m$ on objects.

2.2 Interpretation







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



$$\begin{aligned} \llbracket \text{light-grey bubble} \rrbracket &= p^{\frac{n+m-2}{4}} \sum_{k \in \mathbb{Z}_p} \omega^{2^{-1}(xk+yk^2)} |k : Z\rangle^{\otimes n} \langle k : Z|^{\otimes m} & \llbracket \text{horizontal line} \rrbracket &= \sum_{k \in \mathbb{Z}_p} |k : Z\rangle\langle k : Z| \\ \llbracket \text{red X-spider} \rrbracket &= p^{\frac{n+m-2}{4}} \sum_{k \in \mathbb{Z}_p} \omega^{2^{-1}(xk+yk^2)} |-k : X\rangle^{\otimes n} \langle k : X|^{\otimes m} & \llbracket \text{horizontal line with yellow square} \rrbracket &= \sum_{k \in \mathbb{Z}_p} |k : Z\rangle\langle k : X| \\ \llbracket \text{U-shaped wire} \rrbracket &= \sum_{k \in \mathbb{Z}_p} |kk : Z\rangle & \llbracket \text{C-shaped wire} \rrbracket &= \sum_{k \in \mathbb{Z}_p} \langle kk : Z| & \llbracket \text{crossing} \rrbracket &= \sum_{k, \ell \in \mathbb{Z}_p} |k, \ell : Z\rangle\langle \ell, k : Z| \end{aligned}$$

and $\llbracket s \rrbracket = s$.

There are a couple of things we should remark about this interpretation. First, the definition of the X-spider does not follow the standard convention. It is defined in such a way that it maps X-eigenstates to their additive inverse (modulo p). This definition is used in order to satisfy the property of *flexsymmetry* [16, 17], which allows us to treat diagrams as undirected graphs. Second, note that the interpretation of phases on the spiders has an additional 2^{-1} factor which is necessary for the later stated **EULER** and **GAUSS** axioms to be sound. This factor is considered modulo p , so for instance, for $p = 5$ we have $2^{-1} \equiv 3$. Finally, the spiders are defined with a global scalar factor of $p^{\frac{n+m-2}{4}}$ to follow the *well-tempered normalisation* convention of [27]. This allows us to present the axioms later on with significantly fewer scalar factors floating around.

While the conventional qudit ZX-calculus represents spiders using a $(d-1)$ -dimensional vector [59], we employ a different approach by leveraging a useful property of the Clifford group for prime-dimensional qudits: the phases of its spiders are p^m -th roots of unity raised to polynomial functions with a maximum degree of 2 [26]. This property enables us to capture the essence of Clifford spiders using only two parameters: the coefficients of the linear and square terms. As a result, we develop a more elegant and intuitive framework for reasoning about stabiliser maps, requiring only two parameters in any odd-prime dimension. To establish a connection between our convention and the original qudit ZX-calculus, we define a mapping where a spider with phase parameter (x, y) corresponds to the spider described in [59] with parameter $\vec{\alpha} := (\alpha_1, \dots, \alpha_{d-1})$, where $\alpha_k = \omega^{2^{-1}(xk+yk^2)}$.

For any $a \in \mathbb{Z}_p$, the diagrams  and  correspond to the single qudit Pauli Z^a and X^a gates, respectively. Similarly, the diagrams  and  correspond to Clifford unitaries for any $a, b \in \mathbb{Z}_p$. As a result, we designate spiders with a phase $(a, 0)$ as *Pauli spiders*, and spiders with a phase (a, b) as *Clifford spiders*. Furthermore, spiders with a phase $(0, b)$ are referred to as *purely-Clifford spiders*, while spiders with a phase (a, z) where $z \neq 0$ are termed *strictly-Clifford spiders*. When the parameters of a spider are all zero, i.e. $x = y = 0$, we call the spider *phase-free* and we denote it without label as , and similarly for the X-spider. Lastly, we designate the phase-free X-spider  as the *antipode* since it implements the map $|k : Z\rangle \mapsto |-k : Z\rangle$.

Contrary to the qubit case, the qudit Hadamard gate is not self-inverse. Instead, it follows the property that four successive applications of the Hadamard gate results in the identity, that is, $H^4 = I$. Therefore, the inverse of the Hadamard gate is given by H^3 . To maintain the clarity and simplicity of diagrams, we introduce the shorthand notation  :=  to represent the inverse of the Hadamard box.

2.3 Axioms

We present the axioms of our calculus in Figure 1. In addition to these concrete rules, our calculus also follows the structural rules of a compact-closed PROP. This property implies that “only connectivity matters”, allowing us to treat our diagrams as undirected graphs while preserving their interpretation as linear maps.

These rewrite rules are essentially a simplified version of the complete set of rewrite rules found in [13]. We can show these rules are equivalent to those found in that paper, by deriving the missing axioms.

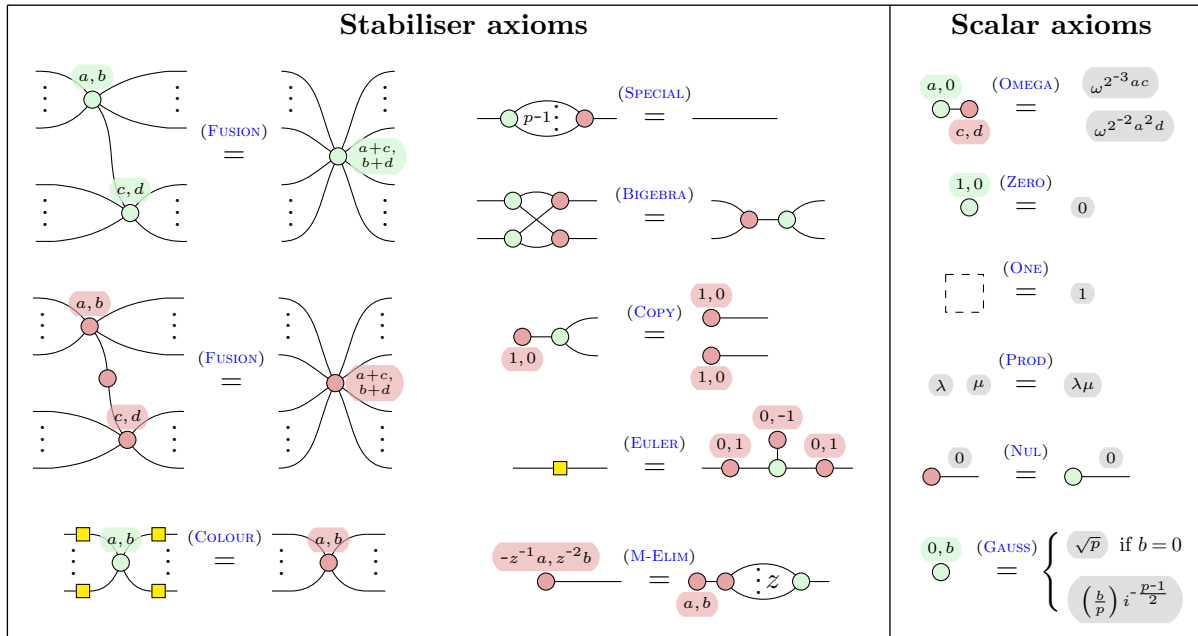
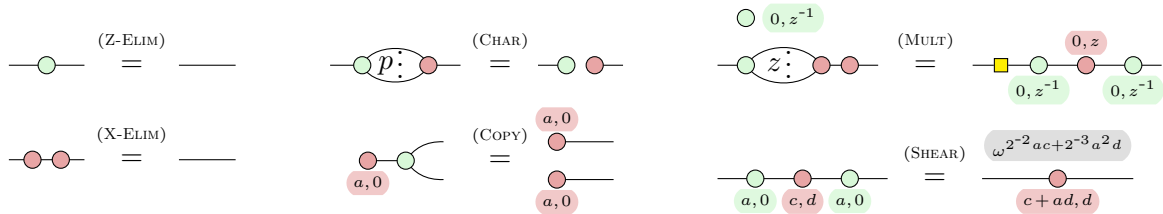


Figure 1: The rewrite rules of the qudit stabiliser ZX-calculus for any odd prime dimension p . Here $a, b, c, d \in \mathbb{Z}_p$, $z \in \mathbb{Z}_p^*$ and $\lambda, \mu \in \mathbb{C}$. $\left(\frac{b}{p}\right)$ is the Legendre symbol, as defined in Equation (1). The dotted square in **ONE** depicts the empty diagram.

Proposition 1. For any $z \in \mathbb{Z}_p^*$ and $a, c, d \in \mathbb{Z}_p$, $\text{ZX}_p^{\text{Stab}}$ proves the following axioms from [13]:



Note that all the proofs in the paper can be found in the appendices.

We also change the presentation of scalars, but we can rely on the reduction in [13] of the scalar fragment to the elementary scalar fragment:

Definition 2. An *elementary scalar* is a diagram $A \in \text{ZX}_p^{\text{Stab}}[0, 0]$ which is a (possibly empty) tensor product of diagrams from $\{\lambda, s, 0 \circ 1, 0 \circ 0, 0 \circ 1 \mid \lambda \in \mathbb{C}, s \in \mathbb{Z}_p\}$.

Lemma 3. $\text{ZX}_p^{\text{Stab}}$ is complete for elementary scalars. Explicitly, if $s : 0 \rightarrow 0$ is an elementary scalar, then $\boxed{s} = \boxed{[s]}$.

With these results, we can see that every derivation of [13] is also valid in our calculus, so that the rules of Figure 1 are complete. For this reason, we freely use the lemmas of [13] in the rest of this paper.

In deriving **MULT** and **SHEAR** in Proposition 1, as well as in the reduction to AP-form of Section 3, we make extensive use of the following “strictly-Clifford” state colour-change rules:

Lemma 4. Strictly-Clifford states can all be represented both using Z- and X-spiders: for any $a \in \mathbb{Z}_p$ and $b \in \mathbb{Z}_p^*$,

$$\begin{array}{c} \text{a, b} \\ \circ \text{---} \end{array} = \begin{array}{c} \text{ab}^{-1}, -b^{-1} \\ \bullet \text{---} \\ \circ \text{---} \\ \text{-ab}^{-1}, b^{-1} \end{array} \qquad \begin{array}{c} \text{a, b} \\ \bullet \text{---} \end{array} = \begin{array}{c} \text{-ab}^{-1}, -b^{-1} \\ \circ \text{---} \\ \circ \text{---} \\ \text{-ab}^{-1}, b^{-1} \end{array}$$

This lemma gives a qupit version of the well-known qubit ZX rule $\begin{array}{c} \pm \frac{\pi}{2} \\ \circ \text{---} \end{array} \propto \begin{array}{c} \mp \frac{\pi}{2} \\ \bullet \text{---} \end{array}$.

On the way to proving this lemma, we also prove the qupit Clifford version of *supplementarity*, originally introduced for the qubit case in Ref. [56]:

Lemma 5. For any $b \in \mathbb{Z}_p^*$,

$$\begin{array}{c} \text{0, b} \\ \bullet \\ \text{0, -b} \\ \bullet \end{array} \begin{array}{c} \circ \\ \text{---} \\ \bullet \end{array} = \begin{array}{c} \circ \\ \text{---} \end{array} \qquad \begin{array}{c} \text{0, b} \\ \circ \\ \text{0, -b} \\ \circ \end{array} \begin{array}{c} \bullet \\ \text{---} \\ \bullet \end{array} = \begin{array}{c} \bullet \\ \text{---} \end{array}$$

A generalisation of this rule is known to be necessary, but not sufficient, for the completeness of the Clifford+T fragment in the qubit case [56, 46].

2.4 A word on scalars

Handling scalars in a graphical language is always a delicate issue. Scalars are essential to guarantee the soundness of rewriting rules but can sometimes be seen as a cumbersome bureaucracy that can be omitted in practice and recovered through a quick normalisation check at the end of a calculation. As a result, some textbooks prefer to work up to non-zero scalars [23], and in [1], a first proof of completeness is presented without scalars, which are addressed in a subsequent article [2]. There is no perfect solution to this situation.

In this paper, we adopt an intermediary approach that can be extended to other graphical languages: the introduction of grey scalar boxes. This approach bears resemblance to how the ZH-calculus handles scalars [3], although in the ZH-calculus, the scalar boxes are directly representable within the calculus itself, requiring no extension as described here. Given any prop \mathbf{P} , the set of scalars $\mathbf{P}[0,0]$ forms a commutative monoid [43]. We view $\mathbf{P}[0,0]$ as a monoidal category with a single object, where the \otimes and \circ operations are identified. We then consider the product category $\mathbf{P}[0,0] \times \mathbf{P}$, which also forms a prop, with arrows represented as pairs (s, f) , where $f : n \rightarrow m$ is an arrow of \mathbf{P} and s is a scalar. Graphically, such a pair is depicted as a diagram representing f together with a floating grey scalar box containing s . The principal equations governing the behaviour of scalar boxes are then **ONE** and **PROD**. In $\mathbf{P}[0,0] \times \mathbf{P}$, grey boxes and diagrams are treated independently. To achieve the desired axiomatization of \mathbf{P} , we need to quotient the equational theory by the equation $\boxed{s} = \boxed{[s]}$ for all $s : 0 \rightarrow 0$. This can be accomplished by introducing rules that guarantee the desired result for a family of well-chosen elementary scalars. Then it is enough to show that any diagram $0 \rightarrow 0$ can be reduced to elementary scalars as we do in Lemma 3.

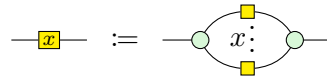
3 Normal forms

In this section, we show that we can simplify stabiliser diagrams into two distinct normal forms: the *affine with phases* (AP) form and the *graph state with local Cliffords* (GSLC) form. The AP form can be efficiently transformed into a unique reduced form, offering an alternative proof

of completeness. On the other hand, the GSLC form is particularly useful for rewriting and decomposing stabiliser unitaries.

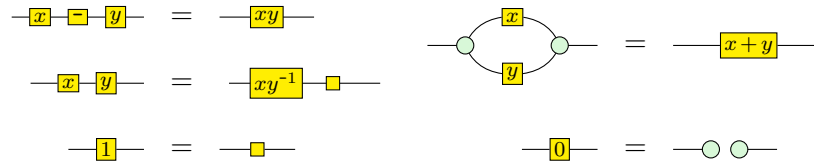
3.1 Graph simplifications

Before reducing the diagrams to our normal forms, we first need to simplify them into a *graph-like* form. In this form, the diagrams consist only of Z-spiders and *H-edges*. To define the qubit graph-like diagrams, we first define *H-boxes* as:



where $x \in \mathbb{Z}_p$ is the *weight* of the H-box. Unlike the *multipliers* in [13], H-boxes are undirected, thus, we can treat diagrams that contain only generators and H-boxes as undirected (weighted) graphs.

Proposition 6. ZX_p^{Stab} proves the following equations:



Since edges that contain H-boxes are central to the subsequent proofs, we define *H-edges*, similarly to the qubit case, as a blue dashed line with the corresponding weight on top:



Definition 7. A ZX-diagram is *graph-like* when:

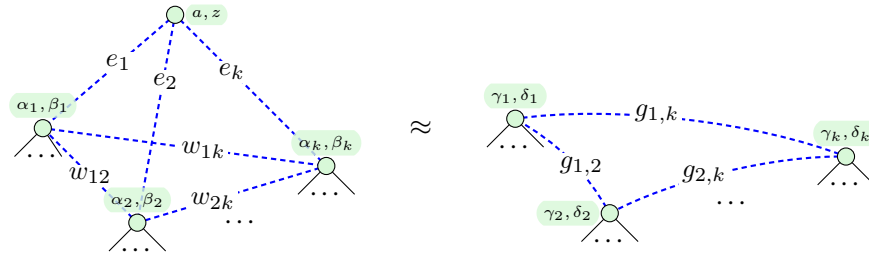
1. All spiders are Z-spiders.
2. Z-spiders are only connected via H-edges.
3. There are no self-loops.
4. Every input or output is connected to a Z-spider.
5. Every Z-spider is connected to at most one input or output.

Using standard techniques [33], it is evident that any ZX-diagram can be transformed into a graph-like form. This transformation involves several steps: performing a colour change on all X-spiders, fusing all Z-spiders, removing self-loops, and introducing identity elements to ensure that each input and output is correctly connected to a Z-spider. Once in graph-like form, the diagram can be represented as an open, weighted graph, where the edge weights are elements of \mathbb{Z}_p and each vertex is labelled by a phase $(a, b) \in \mathbb{Z}_p^2$.

Now that we have a graph-like diagram, we can differentiate between *boundary* spiders, those directly connected to an input or output, and *interior* spiders, those that are only connected to other spiders. Subsequently, we demonstrate that many of the internal spiders can be removed from a diagram using similar techniques to the qubit case [33].

The local complementation simplification enables the removal of a strictly-Clifford interior spider by introducing phases and wires to the spiders it is connected to. This technique is analogous to the qubit version described in [33].

Lemma 8 (Local complementation simplification). For any $z \in \mathbb{Z}_p^*$ and for all $a, \alpha_i, \beta_i, e_i, w_{i,j} \in \mathbb{Z}_p$ where $i, j \in \{1, \dots, k\}$ such that $i < j$ we have:

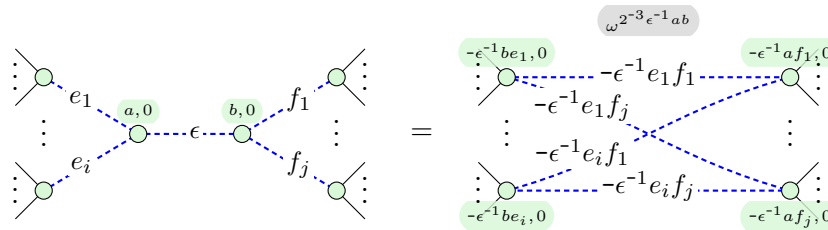


Here $\gamma_i = \alpha_i - e_i a z^{-1}$, $\delta_i = \beta_i - z^{-1} e_i^2$, and $g_{i,j} = w_{ij} - z^{-1} e_i e_j$.

We also have an analogue of the pivot rewrite rule. This rule enables us to eliminate connected interior Pauli spiders by introducing additional phases and connections to the spiders they are connected to.

First, we prove a simplified version of pivoting:

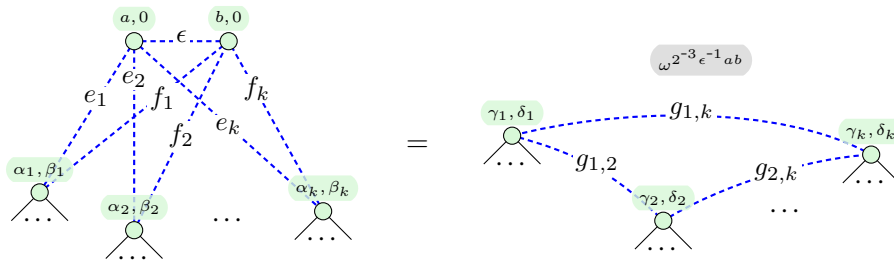
Lemma 9. The following version of pivoting is derivable in $\text{ZX}_p^{\text{Stab}}$:



Here $\epsilon \in \mathbb{Z}_p^*$ and all the other variables are allowed arbitrary values.

Then the general version can be derived from that:

Lemma 10 (Pivoting simplification). General pivoting is derivable in $\text{ZX}_p^{\text{Stab}}$:



Here again $\epsilon \in \mathbb{Z}_p^*$ with every other variable on the left-hand side allowed arbitrary values. On the right-hand side $\gamma_i = \alpha_i - \epsilon^{-1}(a f_i + b e_i)$, $\delta_i = \beta_i - 2\epsilon^{-1} e_i f_i$, and $g_{i,j} = -\epsilon^{-1}(e_i f_j + e_j f_i)$.

3.2 AP-form

The above results suggest that through the application of local complementation and pivoting, it is possible to transform any state diagram (a diagram without inputs) into a graph-like diagram where only Pauli spiders remain internal spiders, and they are exclusively connected to boundary spiders. This is achieved through a two-step process. Firstly, any internal spider that

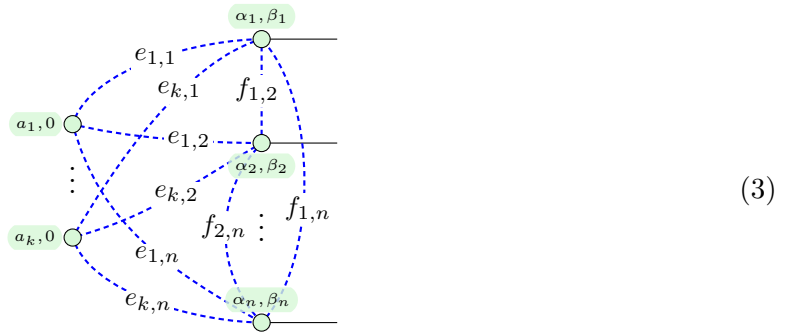
is Clifford is eliminated through local complementation. This ensures that only Pauli spiders remain internal. Secondly, given that the diagram contains only Pauli internal spiders, any connected pair of internal spiders can be removed using pivoting. We give a name to this type of diagram:

Definition 11. We say that a graph-like diagram is in *Affine with Phases form* (AP-form) when:

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

We refer to this class of diagrams as “Affine with Phases” because they correspond to states described by an affine subspace of Z basis states, with an additional phase function applied to the output. This characterisation is supported by the following lemma:

Lemma 12. A general non-zero n -qupit diagram in AP-form is described by the diagram:



where $a_l, \alpha_i, \beta_i, e_{h,i}, f_{i,j} \in \mathbb{Z}_p$ with $l \in \{1, \dots, k\}$ and $i, j \in \{1, \dots, n\}$ such that $i < j$. The interpretation of this diagram is (up to some non-zero scalar) equal to a state

$$\sum_{E\vec{x}=\vec{a}} \omega^{\phi(\vec{x})} |\vec{x}\rangle \tag{4}$$

where E is the weighted bipartite adjacency matrix of the internal and boundary spiders, \vec{a} describes the Pauli phases of the internal spiders, and ϕ is a phase function that describes the connectivity and phases of the boundary spiders:

$$E = \begin{bmatrix} e_{1,1} & \cdots & e_{1,n} \\ e_{2,1} & \cdots & e_{2,n} \\ \vdots & & \vdots \\ e_{k,1} & \cdots & e_{k,n} \end{bmatrix}, \quad \vec{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix}, \quad \phi(\vec{x}) = \sum_{\substack{i,j \in \{1, \dots, n\} \\ i < j}} 2^{-3} x_i \alpha_i + 2^{-2} x_i^2 \beta_i - 2^{-3} f_{i,j} x_i x_j$$

Notably, states described by AP-form diagrams correspond to the stabiliser normal forms described in Ref. [64].

With AP-form diagrams, we can prove a qupit version of the Gottesman-Knill theorem, which states that we can efficiently sample from the probability distribution of a stabiliser computation. Let us consider an AP-form diagram represented by (E, \vec{b}, ϕ) . When we measure this state in the computational basis, we observe that the phase function ϕ has no impact on the measurement

outcomes, allowing us to disregard it. Hence, we can describe the state as $N \sum_{E\vec{x}=\vec{a}} |\vec{x}\rangle$, where N is a normalisation constant. This state represents a uniform superposition of the states $|\vec{x}\rangle$ that satisfy the equation $E\vec{x} = \vec{a}$.

To sample from such states, we need to generate solutions to this equation uniformly at random. Efficiently achieving this involves finding any solution $E\vec{x}' = \vec{a}$ and then obtaining a basis $\vec{v}_1, \dots, \vec{v}_\ell$ for the linear space $\{E\vec{x} = \vec{0}\}$. We can then return $\vec{x}' + \sum_i^\ell b_i \vec{v}_i$, where the $b_i \in \mathbb{Z}_p$ are chosen uniformly at random.

AP-form diagrams also enable us to provide an alternative, more direct proof of the completeness of $\text{ZX}_p^{\text{Stab}}$ through reduction to a unique normal form. In the context of graphical calculi, completeness means that the rewrite rules of the calculus can prove any true equation. In other words, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, then it is possible to rewrite diagram A into diagram B .

Definition 13. We say that a diagram in AP-form defined by (E, \vec{a}, ϕ) is in *reduced AP-form* if it is either zero, or it is non-zero and satisfies the following conditions:

- E is in reduced row echelon form (RREF), i.e., it is fully reduced using Gaussian elimination.
- E contains no fully zero rows.
- ϕ only contains free variables from the equation system of E , i.e., variables that do not correspond to *pivot* columns in E .

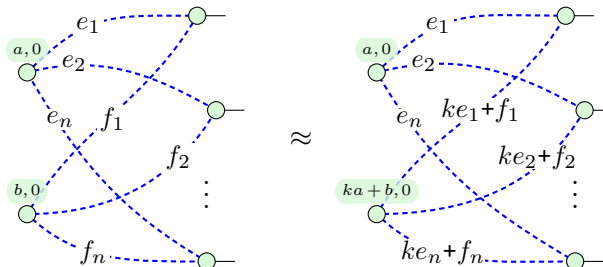
Lemma 14. For any non-zero state $|\psi\rangle$, there is at most one triple (E, \vec{a}, ϕ) satisfying the conditions of reduced AP-form such that:

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

Therefore, a diagram in reduced AP-form is unique.

Now, our objective is to demonstrate that we can rewrite a ZX-diagram in AP-form in a manner that transforms its biadjacency matrix E into RREF. Additionally, we need to show that we can modify the diagram so that the corresponding phase function ϕ only includes free variables from the equation system $E\vec{x} = \vec{a}$. Put simply, we need to prove that we can perform primitive row operations on a ZX-diagram in AP-form as well as eliminate any phase or Hadamard edge from a pivot spider.

Lemma 15. We can perform primitive row operations on a ZX-diagram in AP-form, i.e., we can “add” one inner spider to another. For any $k, a, b, e_i, f_j \in \mathbb{Z}_p$ where $i \in 1, \dots, n$ and $j \in 1, \dots, m$:



Using this result, we can apply primitive row operations to E in AP-form diagram and hence reduce it to RREF. Through diagrammatic rewrites, we can show that when E is in RREF, we can eliminate all the phases and H-edges associated with the non-free variables of E .

Lemma 16. If an AP-form diagram has its biadjacency matrix E in RREF, we can rewrite the diagram so that the boundary spiders corresponding to non-free variables of E have zero phases, and there are no H-edges connecting them to other boundary spiders.

Lemma 17. Any diagram in ZX_p^{Stab} can be converted into one in reduced AP-form.

The completeness result follows immediately from the above lemma.

Theorem 18 (Completeness). For any pair of ZX-diagrams $A, B \in ZX_p^{\text{Stab}}$, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, we can provide a sequence of rewrites that transforms A into B .

3.3 GSLC form

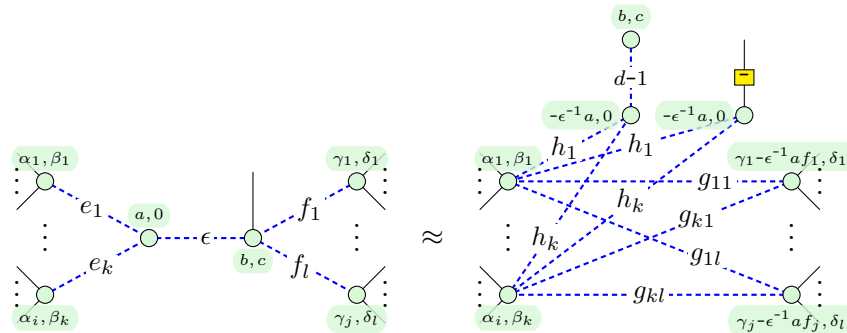
The AP-form is advantageous as it can be directly transformed into a unique normal form, and allows for straightforward classical sampling. However, it may be less suitable for other applications. For instance, when applying the algorithm described above to a diagram originating from a Clifford unitary, it becomes challenging to establish a clear relationship between the resulting simplified diagram and a corresponding quantum circuit.

In this section, we introduce the qubit version of the well-known qubit GSLC-form diagrams.

Definition 19. We say a diagram is in *GSLC form* (Graph State with Local Cliffords) when it is graph-like, up to Hadamards on input and output wires, and it has no internal spiders.

The algorithm for reducing a diagram to AP-form may still yield diagrams with internal spiders, specifically Pauli spiders connected to boundaries. However, we can eliminate these internal spiders by using a *boundary pivot*.

Lemma 20. The following boundary pivot rule is derivable in ZX_p^{Stab} :



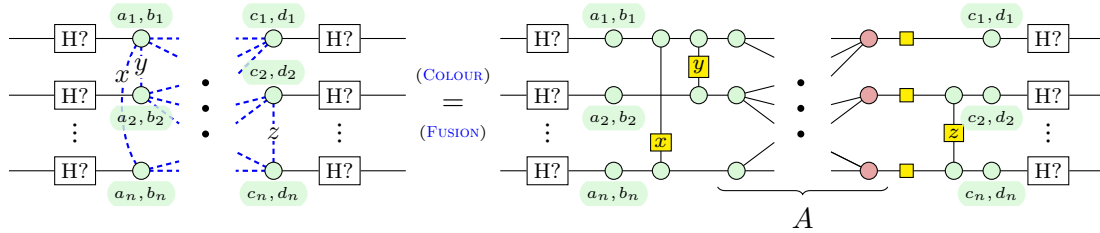
Here $g_{ij} := -\epsilon^{-1}e_i f_j$ and $h_i := -\epsilon^{-1}e_i$. This rule holds for all choices of phases as long as $\epsilon \neq 0$.

To observe how this rewrite aids in eliminating internal spiders, consider that the spider with a phase of (b, c) now becomes an internal spider connected to an internal Pauli spider. Consequently, if $c = 0$, we can eliminate the pair using standard pivoting. On the other hand, if $c \neq 0$, we can employ a local complementation to remove the (b, c) spider. This alteration modifies the phase of its sole neighbour, subsequently enabling its removal through another local complementation.

Lemma 20 can be straightforwardly modified, similar to Lemma 10, to accommodate arbitrary connectivity between the internal spider and the boundary. By incorporating additional

spider unfusions, we can extend the application of Lemma 20 to boundary spiders that are connected to multiple inputs or outputs. It is worth noting that when applying Lemma 20 multiple times to the same boundary, different powers of the Hadamard gate may appear on the input or output wire. For instance, applying it twice yields $(H^3)^2 = H^2$, and another iteration reverts back to H .

Hence, we can observe that it is indeed possible to eliminate all internal spiders from a diagram, allowing for an efficient reduction of diagrams to GSLC form. This is particularly significant for diagrams derived from unitaries, as we can then rewrite them in the following manner:



Here, the boxes labelled with $H?$ represent a possible power of a Hadamard gate acting on the qupit. By applying spider unfusion and colour change operations, we observe that the diagram can be decomposed into several layers consisting of Hadamard gates, Z phase gates, CZ gates, and a middle portion represented by a weighted biadjacency matrix A . This part of the circuit implements a map of the form $|\vec{x}\rangle \mapsto |A\vec{x}\rangle$, where $\vec{x} \in \mathbb{Z}_p^n$ and A is an $n \times n$ matrix over \mathbb{Z}_p . Since we assume the entire map to be unitary, A must also be invertible. Consequently, such a ‘linear’ qupit map can always be implemented through a series of CX gates, transforming $|x, y\rangle$ to $|x, x + y\rangle$ (the decomposition is achieved via standard Gaussian elimination over \mathbb{Z}_p). Thus, we arrive at the following result.

Theorem 21. Any odd-prime-dimensional qudit Clifford unitary can be efficiently decomposed into a quantum circuit consisting of the following layers:

$$H-Z-S-CZ-CX-H-CZ-Z-S-H$$

To the best of our knowledge, such a Clifford normal form for qudits has not been described before in the existing literature. It is worth noting, though, that this result bears a striking resemblance to the qubit normal form for Clifford circuits outlined in [33].

4 Conclusion

We presented a simplified version of the qudit ZX-calculus for odd prime dimensions based on the work in Ref. [13]. This version includes fewer rules and a new scalar gadget to bring the reasoning about scalars more in line with practice. We also extended the spider-removing versions of local complementation and pivoting to qupits. This extension enabled us to reduce diagrams efficiently to AP-form and its unique version, the reduced AP-form. As a result, we obtained a new completeness proof for the qupit stabiliser fragment, which is more straightforward compared to previous proofs. Additionally, we discovered a reduction to GSLC form, leading to a novel layered decomposition of qupit Clifford unitaries. To support these developments, we implemented our rewrites into DiZX, a port of PyZX that now supports qudit stabiliser diagrams of arbitrary dimension.

For future work, it would be interesting to investigate whether our techniques can be applied to develop a useful circuit optimisation pipeline for qudits. It would also be valuable to identify specific circuits that would benefit from such optimisation.

Acknowledgements: We would like to thank Razin A. Shaikh for his contributions to the development of DiZX. LY is supported by an Oxford - Basil Reeve Graduate Scholarship at Oriel College with the Clarendon Fund. Some of this work was done while BP was a student at the University of Oxford. The results of Sections 3.1 to 3.2, Lemma 4, and the explicit scalars are also presented in his Master’s thesis [58]. TC was supported by the ERDF project 1.1.1.5/18/A/020 “Quantum algorithms: from complexity theory to experiment”.

References

- [1] Miriam Backens (2014): *The ZX-calculus Is Complete for Stabilizer Quantum Mechanics*. *New Journal of Physics* 16(9), p. 093021, doi:[10.1088/1367-2630/16/9/093021](https://doi.org/10.1088/1367-2630/16/9/093021).
- [2] Miriam Backens (2015): *Making the Stabilizer ZX-calculus Complete for Scalars*. In Chris Heunen, Peter Selinger & Jamie Vicary, editors: *Proceedings of the 12th International Workshop on Quantum Physics and Logic*, Electronic Proceedings in Theoretical Computer Science, Open Publishing Association, pp. 17–32, doi:[10.4204/EPTCS.195.2](https://doi.org/10.4204/EPTCS.195.2).
- [3] Miriam Backens & Aleks Kissinger (2019): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity*. In Peter Selinger & Giulio Chiribella, editors: *Proceedings of the 15th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 287, Open Publishing Association, Halifax, Canada, pp. 23–42, doi:[10.4204/EPTCS.287.2](https://doi.org/10.4204/EPTCS.287.2).
- [4] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and Back Again: A Circuit Extraction Tale*. *Quantum* 5, p. 421, doi:[10.22331/q-2021-03-25-421](https://doi.org/10.22331/q-2021-03-25-421).
- [5] Miriam Backens, Simon Perdrix & Quanlong Wang (2017): *A Simplified Stabilizer ZX-calculus*. In Ross Duncan & Chris Heunen, editors: *Proceedings 13th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 236, Open Publishing Association, Glasgow, Scotland, pp. 1–20, doi:[10.4204/EPTCS.236.1](https://doi.org/10.4204/EPTCS.236.1).
- [6] Miriam Backens, Simon Perdrix & Quanlong Wang (2020): *Towards a Minimal Stabilizer ZX-calculus*. *Logical Methods in Computer Science* Volume 16, Issue 4, doi:[10.23638/LMCS-16\(4:19\)2020](https://doi.org/10.23638/LMCS-16(4:19)2020).
- [7] Niel de Beaudrap, Xiaoning Bian & Quanlong Wang (2020): *Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities*. In Steven T. Flammia, editor: *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020), Leibniz International Proceedings in Informatics (LIPIcs)* 158, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, p. 11:1–11:23, doi:[10.4230/LIPIcs.TQC.2020.11](https://doi.org/10.4230/LIPIcs.TQC.2020.11).
- [8] Charles H. Bennett & Gilles Brassard (2014): *Quantum Cryptography: Public Key Distribution and Coin Tossing*. *Theoretical Computer Science* 560, pp. 7–11, doi:[10.1016/j.tcs.2014.05.025](https://doi.org/10.1016/j.tcs.2014.05.025).
- [9] Charles H. Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres & William K. Wootters (1993): *Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels*. *Physical Review Letters* 70(13), pp. 1895–1899, doi:[10.1103/PhysRevLett.70.1895](https://doi.org/10.1103/PhysRevLett.70.1895).
- [10] Charles H. Bennett & Stephen J. Wiesner (1992): *Communication via One- and Two-Particle Operators on Einstein-Podolsky-Rosen States*. *Physical Review Letters* 69(20), pp. 2881–2884, doi:[10.1103/PhysRevLett.69.2881](https://doi.org/10.1103/PhysRevLett.69.2881). Available at <https://courses.engr.illinois.edu/phys513/sp2019/reading/week9/PhysRevLett.69.2881.pdf>.

- [11] M. S. Blok, V. V. Ramasesh, T. Schuster, K. O'Brien, J. M. Kreikebaum, D. Dahlen, A. Morvan, B. Yoshida, N. Y. Yao & I. Siddiqi (2021): *Quantum Information Scrambling on a Superconducting Qutrit Processor*. *Physical Review X* 11(2), p. 021010, doi:[10.1103/PhysRevX.11.021010](https://doi.org/10.1103/PhysRevX.11.021010).
- [12] Alex Bocharov, Martin Roetteler & Krysta M. Svore (2017): *Factoring with Qutrits: Shor's Algorithm on Ternary and Metaplectic Quantum Architectures*. *Physical Review A* 96(1), p. 012306, doi:[10.1103/PhysRevA.96.012306](https://doi.org/10.1103/PhysRevA.96.012306). arXiv:[1605.02756](https://arxiv.org/abs/1605.02756).
- [13] Robert I. Booth & Titouan Carette (2022): *Complete ZX-Calculi for the Stabiliser Fragment in Odd Prime Dimensions*. In Stefan Szeider, Robert Ganian & Alexandra Silva, editors: *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 241, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 24:1–24:15, doi:[10.4230/LIPIcs.MFCS.2022.24](https://doi.org/10.4230/LIPIcs.MFCS.2022.24).
- [14] Robert I. Booth & Titouan Carette (2023): *Complete ZX-calculi for the Stabiliser Fragment in Odd Prime Dimensions*. arXiv:[2204.12531v3](https://arxiv.org/abs/2204.12531v3).
- [15] Earl T. Campbell (2014): *Enhanced Fault-Tolerant Quantum Computing in d-Level Systems*. *Physical Review Letters* 113(23), p. 230501, doi:[10.1103/PhysRevLett.113.230501](https://doi.org/10.1103/PhysRevLett.113.230501). arXiv:[1406.3055](https://arxiv.org/abs/1406.3055).
- [16] Titouan Carette (2021): *When Only Topology Matters*. arXiv:[2102.03178](https://arxiv.org/abs/2102.03178).
- [17] Titouan Carette (2021): *Wielding the ZX-calculus, Flexsymmetry, Mixed States, and Scalable Notations*. Ph.D. thesis, Loria, Université de Lorraine. Available at <https://hal.archives-ouvertes.fr/tel-03468027>.
- [18] Yulin Chi, Jieshan Huang, Zhanchuan Zhang, Jun Mao, Zinan Zhou, Xiaojiong Chen, Chonghao Zhai, Jueming Bao, Tianxiang Dai, Huihong Yuan, Ming Zhang, Daoxin Dai, Bo Tang, Yan Yang, Zhihua Li, Yunhong Ding, Leif K. Oxenløwe, Mark G. Thompson, Jeremy L. O'Brien, Yan Li, Qihuang Gong & Jianwei Wang (2022): *A Programmable Qudit-Based Quantum Processor*. *Nature Communications* 13(1), p. 1166, doi:[10.1038/s41467-022-28767-x](https://doi.org/10.1038/s41467-022-28767-x).
- [19] Julien Codsı & John van de Wetering (2023): *Classically Simulating Quantum Supremacy IQP Circuits through a Random Graph Approach*. arXiv:[2212.08609](https://arxiv.org/abs/2212.08609).
- [20] Bob Coecke, Giovanni de Felice, Konstantinos Meichanetzidis & Alexis Toumi (2020): *Foundations for Near-Term Quantum Natural Language Processing*. arXiv:[2012.03755](https://arxiv.org/abs/2012.03755).
- [21] Bob Coecke & Ross Duncan (2008): *Interacting Quantum Observables*. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir & Igor Walukiewicz, editors: *Automata, Languages and Programming*, *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp. 298–310, doi:[10.1007/978-3-540-70583-3_25](https://doi.org/10.1007/978-3-540-70583-3_25). Available at <http://personal.strath.ac.uk/ross.duncan/papers/iqo-icalp.pdf>.
- [22] Bob Coecke & Ross Duncan (2011): *Interacting Quantum Observables: Categorical Algebra and Diagnostics*. *New Journal of Physics* 13(4), p. 043016, doi:[10.1088/1367-2630/13/4/043016](https://doi.org/10.1088/1367-2630/13/4/043016).
- [23] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes*. Cambridge University Press, doi:[10.1017/9781316219317](https://doi.org/10.1017/9781316219317).
- [24] Cole Comfort (2023): *The Algebra for Stabilizer Codes*. arXiv:[2304.10584](https://arxiv.org/abs/2304.10584).
- [25] Daniele Cozzolino, Beatrice Da Lio, Davide Bacco & Leif Katsuo Oxenløwe (2019): *High-Dimensional Quantum Communication: Benefits, Progress, and Future Challenges*. *Advanced Quantum Technologies* 2(12), p. 1900038, doi:[10.1002/qute.201900038](https://doi.org/10.1002/qute.201900038).
- [26] Shawn X. Cui, Daniel Gottesman & Anirudh Krishna (2017): *Diagonal Gates in the Clifford Hierarchy*. *Physical Review A* 95(1), p. 012329, doi:[10.1103/PhysRevA.95.012329](https://doi.org/10.1103/PhysRevA.95.012329). arXiv:[1608.06596](https://arxiv.org/abs/1608.06596).
- [27] Niel de Beaudrap (2021): *Well-Tempered ZX and ZH Calculi*. In Benoît Valiron, Shane Mansfield, Pablo Arrighi & Prakash Panangaden, editors: *Proceedings 17th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 340, Open Publishing Association, Paris, France, pp. 13–45, doi:[10.4204/EPTCS.340.2](https://doi.org/10.4204/EPTCS.340.2).

- [28] Niel de Beaudrap & Richard D. P. East (2023): *Simple ZX and ZH Calculi for Arbitrary Finite Dimensions, via Discrete Integrals*. arXiv:[2304.03310](#).
- [29] Niel de Beaudrap & Dominic Horsman (2020): *The ZX Calculus Is a Language for Surface Code Lattice Surgery*. *Quantum* 4, p. 218, doi:[10.22331/q-2020-01-09-218](#).
- [30] Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang & Bob Coecke (2023): *Light-Matter Interaction in the ZXW Calculus*. arXiv:[2306.02114](#).
- [31] Nadish de Silva (2021): *Efficient Quantum Gate Teleportation in Higher Dimensions*. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 477(2251), p. 20200865, doi:[10.1098/rspa.2020.0865](#). arXiv:[2011.00127](#).
- [32] Jeroen Dehaene & Bart De Moor (2003): *Clifford Group, Stabilizer States, and Linear and Quadratic Operations over $GF(2)$* . *Physical Review A* 68(4), p. 042318, doi:[10.1103/PhysRevA.68.042318](#). arXiv:[quant-ph/0304125](#).
- [33] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:[10.22331/q-2020-06-04-279](#).
- [34] Ross Duncan & Maxime Lucas (2014): *Verifying the Steane Code with Quantomatic*. In Bob Coecke & Matty Hoban, editors: *Proceedings of the 10th International Workshop on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 171, Open Publishing Association, Castelldefels (Barcelona), Spain, pp. 33–49, doi:[10.4204/EPTCS.171.4](#).
- [35] Ross Duncan & Simon Perdrix (2009): *Graph States and the Necessity of Euler Decomposition*. In Klaus Ambos-Spies, Benedikt Löwe & Wolfgang Merkle, editors: *Mathematical Theory and Computational Practice, Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp. 167–177, doi:[10.1007/978-3-642-03073-4_18](#).
- [36] Ross Duncan & Simon Perdrix (2010): *Rewriting Measurement-Based Quantum Computations with Generalised Flow*. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide & Paul G. Spirakis, editors: *Automata, Languages and Programming, Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp. 285–296, doi:[10.1007/978-3-642-14162-1_24](#).
- [37] Liam Garvie & Ross Duncan (2018): *Verifying the Smallest Interesting Colour Code with Quantomatic*. In Bob Coecke & Aleks Kissinger, editors: *Proceedings 14th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 266, Open Publishing Association, Nijmegen, The Netherlands, pp. 147–163, doi:[10.4204/EPTCS.266.10](#).
- [38] Pranav Gokhale, Jonathan M. Baker, Casey Duckering, Natalie C. Brown, Kenneth R. Brown & Frederic T. Chong (2019): *Asymptotic Improvements to Quantum Circuits via Qutrits*. In: *Proceedings of the 46th International Symposium on Computer Architecture, ISCA '19*, Association for Computing Machinery, New York, NY, USA, pp. 554–566, doi:[10.1145/3307650.3322253](#).
- [39] Xiaoyan Gong & Quanlong Wang (2017): *Equivalence of Local Complementation and Euler Decomposition in the Qutrit ZX-calculus*. arXiv:[1704.05955](#).
- [40] Noah Goss, Alexis Morvan, Brian Marinelli, Bradley K. Mitchell, Long B. Nguyen, Ravi K. Naik, Larry Chen, Christian Jünger, John Mark Kreikebaum, David I. Santiago, Joel J. Wallman & Irfan Siddiqi (2022): *High-Fidelity Qutrit Entangling Gates for Superconducting Circuits*. *Nature Communications* 13(1), p. 7481, doi:[10.1038/s41467-022-34851-z](#).
- [41] Daniel Gottesman (1998): *The Heisenberg Representation of Quantum Computers*. arXiv:[quant-ph/9807006](#).
- [42] Daniel Gottesman (1999): *Fault-Tolerant Quantum Computation with Higher-Dimensional Systems*. In Colin P. Williams, editor: *Quantum Computing and Quantum Communications, Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp. 302–313, doi:[10.1007/3-540-49208-9_27](#). arXiv:[quant-ph/9802007](#).

- [43] Chris Heunen & Jamie Vicary (2019): *Categories for Quantum Theory: An Introduction*. Oxford University Press, doi:[10.1093/oso/9780198739623.001.0001](https://doi.org/10.1093/oso/9780198739623.001.0001).
- [44] Alexander D. Hill, Mark J. Hodson, Nicolas Didier & Matthew J. Reagor (2021): *Realization of Arbitrary Doubly-Controlled Quantum Phase Gates*. arXiv:[2108.01652](https://arxiv.org/abs/2108.01652).
- [45] Pavel Hrmó, Benjamin Wilhelm, Lukas Gerster, Martin W. van Mourik, Marcus Huber, Rainer Blatt, Philipp Schindler, Thomas Monz & Martin Ringbauer (2023): *Native Qudit Entanglement in a Trapped Ion Quantum Processor*. *Nature Communications* 14(1), p. 2242, doi:[10.1038/s41467-023-37375-2](https://doi.org/10.1038/s41467-023-37375-2).
- [46] Emmanuel Jeandel, Simon Perdrix, Renaud Vilmart & Quanlong Wang (2017): *ZX-Calculus: Cyclo-tomic Supplementarity and Incompleteness for Clifford+T Quantum Mechanics*. In Kim G. Larsen, Hans L. Bodlaender & Jean-Francois Raskin, editors: *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017), Leibniz International Proceedings in Informatics (LIPIcs)* 83, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 11:1–11:13, doi:[10.4230/LIPIcs.MFCS.2017.11](https://doi.org/10.4230/LIPIcs.MFCS.2017.11).
- [47] Andrey Boris Khesin, Jonathan Z. Lu & Peter W. Shor (2023): *Graphical Quantum Clifford-encoder Compilers from the ZX Calculus*. arXiv:[2301.02356](https://arxiv.org/abs/2301.02356).
- [48] Aleks Kissinger (2022): *Phase-free ZX diagrams are CSS codes (...or how to graphically grok the surface code)*. arXiv:[2204.14038](https://arxiv.org/abs/2204.14038).
- [49] Aleks Kissinger & John van de Wetering (2020): *PyZX: Large Scale Automated Diagrammatic Reasoning*. In: *Proceedings 16th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 318, Open Publishing Association, CA, USA., pp. 229–241, doi:[10.4204/EPTCS.318.14](https://doi.org/10.4204/EPTCS.318.14).
- [50] Aleks Kissinger & John van de Wetering (2020): *Reducing the Number of Non-Clifford Gates in Quantum Circuits*. *Physical Review A* 102(2), p. 022406, doi:[10.1103/PhysRevA.102.022406](https://doi.org/10.1103/PhysRevA.102.022406). arXiv:[1903.10477](https://arxiv.org/abs/1903.10477).
- [51] Aleks Kissinger, John van de Wetering & Renaud Vilmart (2022): *Classical Simulation of Quantum Circuits with Partial and Graphical Stabiliser Decompositions*. In François Le Gall & Tomoyuki Morimae, editors: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022), Leibniz International Proceedings in Informatics (LIPIcs)* 232, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 5:1–5:13, doi:[10.4230/LIPIcs.TQC.2022.5](https://doi.org/10.4230/LIPIcs.TQC.2022.5).
- [52] Tuomas Laakkonen, Konstantinos Meichanetzidis & John van de Wetering (2022): *A Graphical #SAT Algorithm for Formulae with Small Clause Density*. arXiv:[2212.08048](https://arxiv.org/abs/2212.08048).
- [53] Tommy McElvanney & Miriam Backens (2022): *Complete Flow-Preserving Rewrite Rules for MBQC Patterns with Pauli Measurements*. arXiv:[2205.02009](https://arxiv.org/abs/2205.02009).
- [54] Konstantinos Meichanetzidis, Stefano Gogioso, Giovanni de Felice, Nicolò Chiappori, Alexis Toumi & Bob Coecke (2021): *Quantum Natural Language Processing on Near-Term Quantum Computers*. In Benoît Valiron, Shane Mansfield, Pablo Arrighi & Prakash Panangaden, editors: *Proceedings 17th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 340, Open Publishing Association, Paris, France, pp. 213–229, doi:[10.4204/EPTCS.340.11](https://doi.org/10.4204/EPTCS.340.11).
- [55] A. S. Nikolaeva, E. O. Kiktenko & A. K. Fedorov (2022): *Decomposing the Generalized Toffoli Gate with Qutrits*. *Physical Review A* 105(3), p. 032621, doi:[10.1103/PhysRevA.105.032621](https://doi.org/10.1103/PhysRevA.105.032621). arXiv:[2112.14535](https://arxiv.org/abs/2112.14535).
- [56] Simon Perdrix & Quanlong Wang (2016): *Supplementarity Is Necessary for Quantum Diagram Reasoning*. In Piotr Faliszewski, Anca Muscholl & Rolf Niedermeier, editors: *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016), Leibniz Interna-*

- tional Proceedings in Informatics (LIPIcs)* 58, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 76:1–76:14, doi:[10.4230/LIPIcs.MFCS.2016.76](https://doi.org/10.4230/LIPIcs.MFCS.2016.76).
- [57] Boldizsár Poór, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung & Bob Coecke (2023): *Completeness for Arbitrary Finite Dimensions of ZXW-calculus, a Unifying Calculus*. In: *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, Boston, MA, USA, pp. 1–14, doi:[10.1109/LICS56636.2023.10175672](https://doi.org/10.1109/LICS56636.2023.10175672). arXiv:[2302.12135](https://arxiv.org/abs/2302.12135).
- [58] Boldizsár Poór (2022): *A Unique Normal Form for Prime-Dimensional Qudit Clifford ZX-calculus*. Master's thesis, University of Oxford. Available at <https://www.cs.ox.ac.uk/people/aleks.kissinger/theses/poor-thesis.pdf>.
- [59] André Ranchin (2014): *Depicting Qudit Quantum Mechanics and Mutually Unbiased Qudit Theories*. In Bob Coecke, Ichiro Hasuo & Prakash Panangaden, editors: *Proceedings of the 11th Workshop on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 172, Open Publishing Association, Kyoto, Japan, pp. 68–91, doi:[10.4204/EPTCS.172.6](https://doi.org/10.4204/EPTCS.172.6).
- [60] Martin Ringbauer, Michael Meth, Lukas Postler, Roman Stricker, Rainer Blatt, Philipp Schindler & Thomas Monz (2022): *A Universal Qudit Quantum Processor with Trapped Ions*. *Nature Physics* 18(9), pp. 1053–1057, doi:[10.1038/s41567-022-01658-0](https://doi.org/10.1038/s41567-022-01658-0). arXiv:[2109.06903](https://arxiv.org/abs/2109.06903).
- [61] Razin A. Shaikh, Quanlong Wang & Richie Yeung (2022): *How to Sum and Exponentiate Hamiltonians in ZXW Calculus*. arXiv:[2212.04462](https://arxiv.org/abs/2212.04462).
- [62] Alex Townsend-Teague & Konstantinos Meichanetzidis (2022): *Simplification Strategies for the Qutrit ZX-Calculus*. arXiv:[2103.06914](https://arxiv.org/abs/2103.06914).
- [63] John van de Wetering & Lia Yeh (2022): *Phase Gadget Compilation for Diagonal Qutrit Gates*. arXiv:[2204.13681](https://arxiv.org/abs/2204.13681).
- [64] Maarten Van den Nest (2010): *Classical Simulation of Quantum Computation, the Gottesman-Knill Theorem, and Slightly Beyond*. *Quantum Information & Computation* 10(3&4), pp. 258–271, doi:[10.26421/QIC10.3-4-6](https://doi.org/10.26421/QIC10.3-4-6). arXiv:[0811.0898](https://arxiv.org/abs/0811.0898).
- [65] Quanlong Wang (2018): *Qutrit ZX-calculus Is Complete for Stabilizer Quantum Mechanics*. In Bob Coecke & Aleks Kissinger, editors: *Proceedings 14th International Conference on Quantum Physics and Logic, Electronic Proceedings in Theoretical Computer Science* 266, Open Publishing Association, Nijmegen, The Netherlands, pp. 58–70, doi:[10.4204/EPTCS.266.3](https://doi.org/10.4204/EPTCS.266.3).
- [66] Quanlong Wang (2022): *Qufinite ZX-calculus: A Unified Framework of Qudit ZX-calculi*. arXiv:[2104.06429](https://arxiv.org/abs/2104.06429).
- [67] Quanlong Wang, Richie Yeung & Mark Koch (2022): *Differentiating and Integrating ZX Diagrams with Applications to Quantum Machine Learning*. arXiv:[2201.13250](https://arxiv.org/abs/2201.13250).
- [68] Yuchen Wang, Zixuan Hu, Barry C. Sanders & Sabre Kais (2020): *Qudits and High-Dimensional Quantum Computing*. *Frontiers in Physics* 8, doi:[10.3389/fphy.2020.589504](https://doi.org/10.3389/fphy.2020.589504).
- [69] Biaoliang Ye, Zhen-Fei Zheng, Yu Zhang & Chui-Ping Yang (2018): *Circuit QED: single-step realization of a multiqubit controlled phase gate with one microwave photonic qubit simultaneously controlling $n - 1$ microwave photonic qubits*. *Optics Express* 26(23), pp. 30689–30702, doi:[10.1364/OE.26.030689](https://doi.org/10.1364/OE.26.030689).
- [70] Lia Yeh & John van de Wetering (2022): *Constructing All Qutrit Controlled Clifford+T Gates in Clifford+T*. In Claudio Antares Mezzina & Krzysztof Podlaski, editors: *Reversible Computation, Lecture Notes in Computer Science*, Springer International Publishing, Cham, pp. 28–50, doi:[10.1007/978-3-031-09005-9_3](https://doi.org/10.1007/978-3-031-09005-9_3). arXiv:[2204.00552](https://arxiv.org/abs/2204.00552).
- [71] M. A. Yurtalan, J. Shi, M. Kononenko, A. Lupascu & S. Ashhab (2020): *Implementation of a Walsh-Hadamard Gate in a Superconducting Qutrit*. *Physical Review Letters* 125(18), p. 180504, doi:[10.1103/PhysRevLett.125.180504](https://doi.org/10.1103/PhysRevLett.125.180504). arXiv:[2003.04879](https://arxiv.org/abs/2003.04879).
- [72] Yutsumura (2017): *Each Element in a Finite Field Is the Sum of Two Squares*. Available at <https://yutsumura.com/each-element-in-a-finite-field-is-the-sum-of-two-squares>.

- [73] Fabio Zanasi (2018): *Interacting Hopf Algebras: The Theory of Linear Systems*. Ph.D. thesis, Ecole Normale Supérieure de Lyon. arXiv:[1805.03032](https://arxiv.org/abs/1805.03032).
- [74] Chen Zhao & Xiao-Shan Gao (2021): *Analyzing the Barren Plateau Phenomenon in Training Quantum Neural Networks with the ZX-calculus*. *Quantum* 5, p. 466, doi:[10.22331/q-2021-06-04-466](https://doi.org/10.22331/q-2021-06-04-466).

Appendix

A Necessity of the rules

We can demonstrate that most of the non-scalar rules of our axiomatisation are *necessary*, meaning that they cannot be derived from the other rules.

Note that the standard approach to showing necessity involves defining an alternative interpretation of the diagrams in which every rewrite rule remains sound, except for the specific rule being examined for necessity. This approach reveals that the other rules cannot establish the rule that undermines soundness. Several examples of this approach can be found in the works of Backens, Perdrix, and Wang [5, 6]. In particular, we may define an interpretation into projective Hilbert spaces (quotienting by all non-zero scalars), in order to automatically satisfy all the scalar axioms and focus on the non-scalar axioms. This automatically satisfies all the scalar axioms, allowing us to focus solely on the non-scalar axioms. Another approach involves using graph properties that are invariant under all but one rule. In the following discussion, we rely on the invariants of non-emptiness and connectivity.

We can demonstrate the necessity of all but two of the stabiliser rules:

- At least one of the **FUSION** rules is necessary, as they are the only rules that allow the decomposition of a spider with an arbitrary number of legs into spiders with fewer legs. In other words, these rules are not sound for an interpretation that assigns zero to all spiders with at least p legs.
- **SPECIAL** is the only axiom that enables the removal of all non-identity generators from a diagram. This breaks the interpretation where every generator is zero, except for the identity.
- **COLOUR** is necessary. To see this, consider the interpretation into projective Hilbert spaces where we redefine the X-spiders to swap the sign of the Pauli phase. It can be easily verified that this new interpretation satisfies all axioms except for **COLOUR**.
- **COPY** is necessary since it is the only axiom that can transform a connected diagram into a disconnected one.
- **EULER** is necessary, as shown by a modified interpretation similar to those in Refs. [35, 39].

We propose a conjecture regarding the necessity of **M-ELIM**, as it stands out as the only rule that establishes a connection between elements in \mathbb{Z}_p^* and their multiplicative inverses. Although we lack a formal proof for this intuition, we believe it to be true.

It is worth noting that despite its centrality in most derivations, there remains one stabiliser axiom for which we have no knowledge of its necessity, even in the qubit case [6] or in the setting of graphical linear algebra [73]: **BIGEBRA**. We leave this intriguing open problem for the particularly motivated reader to explore further.

As for the scalar rules, at least one subcase of each is necessary:

- One subcase of **OMEGA** is necessary because it is the only rule that allows the introduction of an ω scalar box, thereby breaking the interpretation where we redefine the ω scalar box.
- **ZERO** is the only rule that relates a diagram without a zero scalar box to one that includes it. This means that when we interpret the zero scalar box as equal to 1 and set all other generators to zero, this rule becomes necessary.
- **ONE** is necessary as it is the only rule that connects a non-empty diagram to an empty diagram.
- **PROD** is necessary because there are complex numbers that cannot be expressed within the fragment of the language without scalar boxes. This rule is the only one that allows the multiplication of two such numbers.
- **NUL** is necessary, following an analogous argument as of Ref. [5].
- In **GAUSS**, the subcase $b = 0$ is necessary since it is the only rule that allows one to interpret a diagram to a scalar box with a non-unit modulus. Additionally, at least one subcase $b \neq 0$ is necessary because these are the only rules that introduce a -1 scalar box.

B Qubit Clifford ZX-calculus

B.1 Multipliers

We extend our language by *multipliers* [13], which are defined as:

$$\begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} := \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} \quad \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} := \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} \quad (5)$$

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

$$\begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} \quad \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle x \rangle \text{---} \\ \text{---} \langle x \rangle \text{---} \end{array} \quad (6)$$

The following equations hold for multipliers and are proved in Ref. [13]:

Proposition 22.

$$\begin{array}{c} \text{---} \langle x \rangle \langle y \rangle \text{---} \\ \text{---} \langle x \rangle \langle y \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle xy \rangle \text{---} \\ \text{---} \langle xy \rangle \text{---} \end{array} \quad \begin{array}{c} \text{---} \langle z^{-1} \rangle \text{---} \\ \text{---} \langle z^{-1} \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle z \rangle \text{---} \\ \text{---} \langle z \rangle \text{---} \end{array}$$

$$\begin{array}{c} \text{---} \langle -1 \rangle \text{---} \\ \text{---} \langle -1 \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle -1 \rangle \text{---} \\ \text{---} \langle -1 \rangle \text{---} \end{array} \quad \begin{array}{c} \text{---} \langle x \rangle \langle y \rangle \text{---} \\ \text{---} \langle x \rangle \langle y \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle x+y \rangle \text{---} \\ \text{---} \langle x+y \rangle \text{---} \end{array}$$

$$\text{---} \text{---} = \begin{array}{c} \text{---} \langle 1 \rangle \text{---} \\ \text{---} \langle 1 \rangle \text{---} \end{array} \quad \begin{array}{c} \text{---} \langle p \rangle \text{---} \\ \text{---} \langle p \rangle \text{---} \end{array} = \begin{array}{c} \text{---} \langle 0 \rangle \text{---} \\ \text{---} \langle 0 \rangle \text{---} \end{array}$$

B.2 Recovering the derivations of Ref. [13]

In this appendix, we recover all of the lemmas that were proved in [13]. We do this by proving that all of the axioms used there are derivable from the simplified set given in section 2 (up to scalars). We also show that the language is complete for the scalar fragment, which completes the proof. Since many of these proofs are entirely analogous to their counterparts in Ref. [13], we omit them and refer to Ref. [13] instead. In order to avoid ambiguity, we refer to the proofs in the specific version of Ref. [13] cited as Ref. [14].

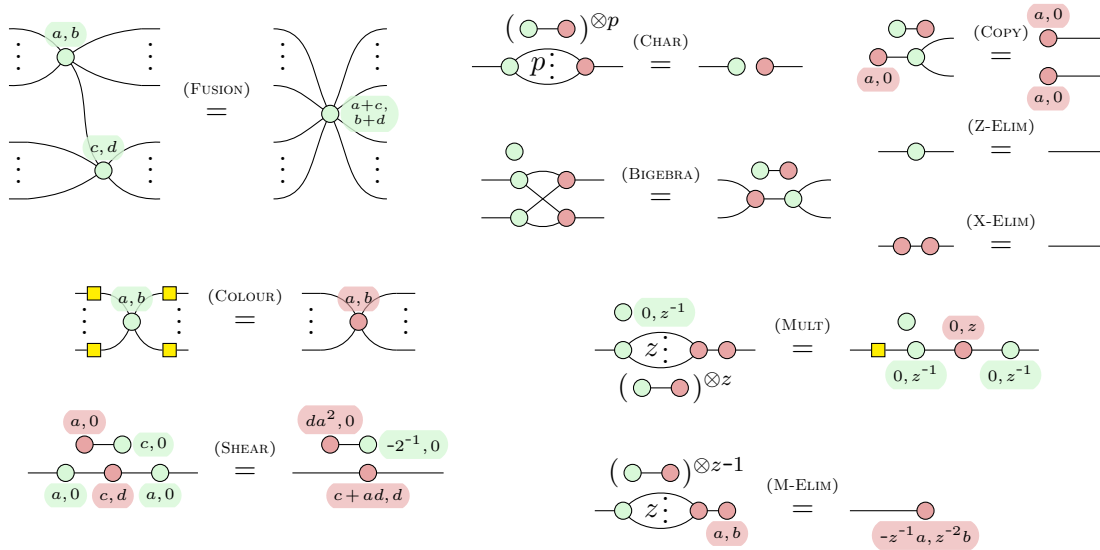


Figure 2: The original rule set of the qupit stabiliser ZX-calculus of [13].

In [13], the calculus was axiomatised using the equations presented in Figure 2.

Comparing with the axioms of this paper (and ignoring scalars for now), the missing axioms are Z-ELIM, X-ELIM, CHAR, MULT, SHEAR. In addition, the axioms of FUSION and COPY were made more minimalistic.

Lemma 23. Green $1 \rightarrow 1$ spiders are trivial:

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

Proof.

$$\text{---} \circ \text{---} \stackrel{\text{(SPECIAL)}}{=} \text{---} \circ \text{---} \circ \text{---} \stackrel{\text{(FUSION)}}{=} \text{---} \circ \text{---} \stackrel{\text{(SPECIAL)}}{=} \text{---} \quad \square$$

Lemma 24. Products of Hadamards are antipodes:

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

Proof. Same as Lemma 37 of Ref. [14]. □

Lemma 25. Antipodes are self-inverse:

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

Proof.

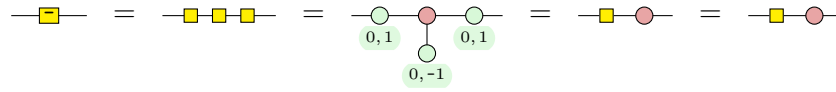
$$\text{---} \circ \text{---} \stackrel{\text{(SPECIAL)}}{=} \text{---} \circ \text{---} \circ \text{---} \stackrel{\text{(FUSION)}}{=} \text{---} \circ \text{---} \stackrel{\text{(SPECIAL)}}{=} \text{---} \quad \square$$

Lemma 26. Hadamards and antipodes commute:

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

Proof. Same as Lemma 38 of Ref. [14]. □

Lemma 27. The inverse Hadamard is a product of Hadamards, and admits a “tree” Euler decomposition:



Proof. The first part is the same as Lemma 39 of Ref. [14], and the second part follows using **EULER** and **COLOUR**. The last two follow from the first equation and Lemma 24. □

Lemma 28. The product of the Hadamard and its inverse equals the identity:



Proof.

 The second equation can be proved similarly. □

Lemma 29. Units absorb antipodes:



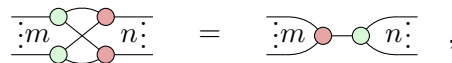
Proof. Same as Lemma 40 of Ref. [14]. □

Lemma 30. For any $x, y \in \mathbb{Z}_p$,



Proof.
□

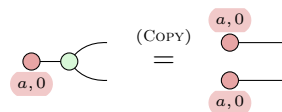
Lemma 31. The bigebra law holds for multiple legs: for any $m, n \in \mathbb{N}$, $2 \leq n, m$,



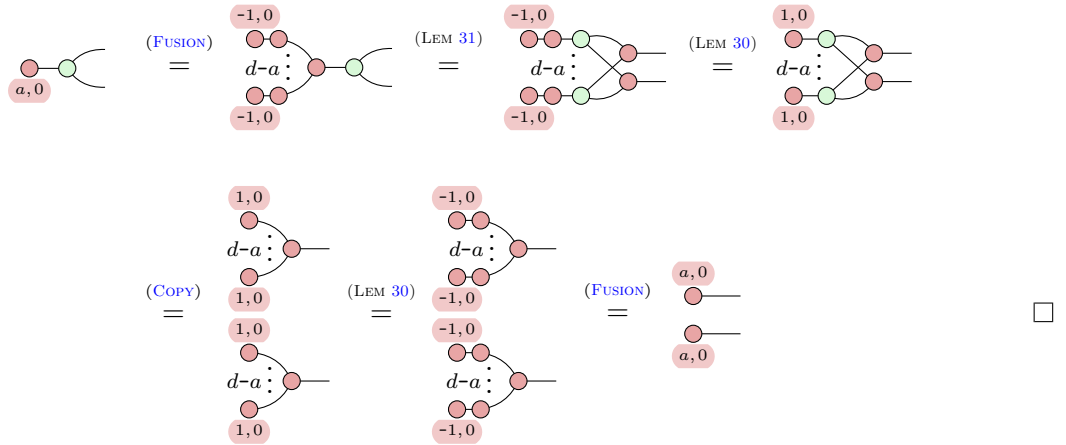
where in the diagram on the LHS, there are m green and n red spiders, and each green spider is connected to each red spider by a single wire.

Proof. Follows from straightforward induction (which furthermore is analogous to the qubit case). □

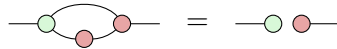
Lemma 32. Using the **COPY** rule in Figure 1, the COPY rule in Ref. [13] is derivable:



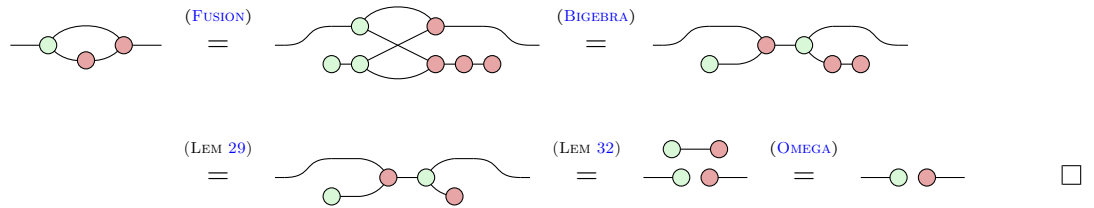
Proof. The following holds for $a = 2, \dots, p$. As $p \bmod p = 0$, this also proves the $a = 0$ subcase of COPY. The $a = 1$ case is just COPY.



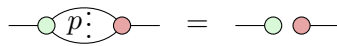
Lemma 33. The Hopf identity is derivable in ZX_p^{Stab} :



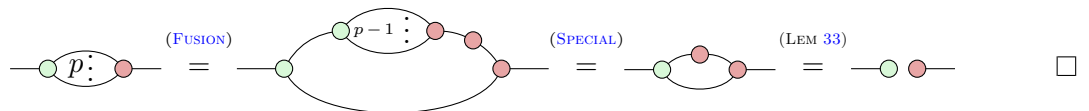
Proof.



Lemma 34. The axiom CHAR of [13] is derivable:



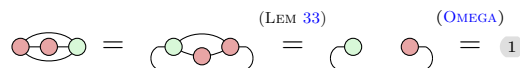
Proof.



We are now ready to prove the completeness of the calculus for elementary scalars.

Lemma 3. ZX_p^{Stab} is complete for elementary scalars. Explicitly, if $s : 0 \rightarrow 0$ is an elementary scalar, then $\boxed{s} = \llbracket s \rrbracket$.

Proof. First, note that we have:



We can use this rule and the scalar axioms to rewrite every scalar in Definition 2, as well as the zero scalar $\circlearrowleft_{1,0}$, into an explicit scalar. Then, we apply PROD to rewrite this collection

| of explicit scalars into a single one. □

Lemma 35. Self-loops on green spiders can be eliminated:

$$\begin{array}{c} \text{green spider with self-loop} \end{array} = \begin{array}{c} \sqrt{p} \\ \text{green spider} \end{array} \qquad \begin{array}{c} \text{red spider with self-loop} \end{array} = \begin{array}{c} \sqrt{p} \\ \text{red spider} \end{array}$$

We include the colour-swapped version of this rule for completeness, even though it no longer includes a genuine self-loop.

Proof.

$$\begin{array}{c} \text{green spider with self-loop} \end{array} \stackrel{\text{(LEM 25)}}{=} \begin{array}{c} \text{red spider with self-loop} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{red spider with green spider} \end{array} \stackrel{\text{(LEM 33)}}{=} \begin{array}{c} \text{red spider} \end{array} \begin{array}{c} \text{green spider} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \sqrt{p} \\ \text{green spider} \end{array} \stackrel{\text{(GAUSS)}}{=} \begin{array}{c} \text{green spider} \end{array}$$

The red version follows from COLOUR and Lemma 24. □

Lemma 36. Green units absorb red rotations and vice-versa:

$$\begin{array}{c} \text{red spider} \begin{array}{c} x, y \\ \text{green spider} \end{array} \end{array} = \begin{array}{c} \text{red spider} \end{array} \qquad \begin{array}{c} \text{green spider} \begin{array}{c} x, y \\ \text{red spider} \end{array} \end{array} = \begin{array}{c} \text{green spider} \end{array}$$

Proof.

$$\begin{array}{c} \text{red spider} \begin{array}{c} x, y \\ \text{green spider} \end{array} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} \text{green spider} \\ x, y \end{array} \end{array} \stackrel{\text{(LEM 32)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} x, y \\ \text{red spider} \end{array} \end{array} \stackrel{\text{(OMEGA)}}{=} \begin{array}{c} \text{red spider} \end{array}$$

The red version follows from COLOUR. □

Lemma 37. The green co-multiplication copies antipodes:

$$\begin{array}{c} \text{red spider} \begin{array}{c} \text{green spider} \end{array} \end{array} = \begin{array}{c} \text{green spider} \begin{array}{c} \text{red spider} \end{array} \end{array}$$

Proof.

$$\begin{array}{c} \text{red spider} \begin{array}{c} \text{green spider} \end{array} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} \text{red spider} \end{array} \begin{array}{c} \text{green spider} \end{array} \end{array} \stackrel{\text{(LEM 29)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} \text{red spider} \end{array} \begin{array}{c} \text{red spider} \end{array} \end{array} \stackrel{\text{(BIGEBRA)}}{=} \begin{array}{c} \text{green spider} \begin{array}{c} \text{red spider} \end{array} \end{array} \begin{array}{c} \text{red spider} \end{array} \\
 \stackrel{\text{(LEM 32)}}{=} \begin{array}{c} \text{green spider} \begin{array}{c} \text{red spider} \end{array} \end{array} \begin{array}{c} \text{red spider} \end{array} \stackrel{\text{(LEM 29)}}{=} \begin{array}{c} \text{green spider} \begin{array}{c} \text{red spider} \end{array} \end{array} \begin{array}{c} \text{red spider} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{green spider} \begin{array}{c} \text{red spider} \end{array} \end{array}$$

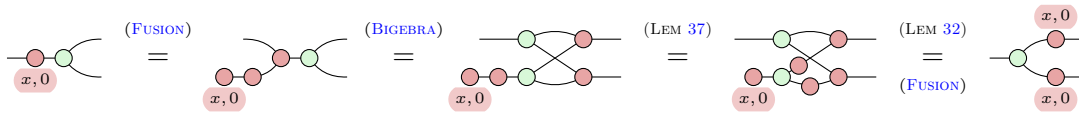
Lemma 38. Green Pauli states copy through red rotations and vice-versa:

$$\begin{array}{c} \text{red spider} \begin{array}{c} a, 0 \\ \text{green spider} \end{array} \end{array} \begin{array}{c} c, d \end{array} = \begin{array}{c} \text{red spider} \begin{array}{c} a, 0 \end{array} \end{array} \begin{array}{c} \omega^{2^{-3}ac+2^{-2}a^2d} \\ \text{red spider} \end{array} \qquad \begin{array}{c} \text{green spider} \begin{array}{c} a, 0 \\ \text{red spider} \end{array} \end{array} \begin{array}{c} c, d \end{array} = \begin{array}{c} \text{green spider} \begin{array}{c} -a, 0 \end{array} \end{array} \begin{array}{c} \omega^{2^{-3}ac+2^{-2}a^2d} \\ \text{green spider} \end{array}$$

Proof.

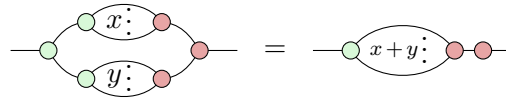
$$\begin{array}{c} \text{red spider} \begin{array}{c} a, 0 \\ \text{green spider} \end{array} \end{array} \begin{array}{c} c, d \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} a, 0 \end{array} \end{array} \begin{array}{c} \text{green spider} \begin{array}{c} c, d \end{array} \end{array} \stackrel{\text{(LEM 32)}}{=} \begin{array}{c} \text{red spider} \begin{array}{c} a, 0 \\ \text{red spider} \end{array} \end{array} \begin{array}{c} c, d \end{array} \stackrel{\text{(OMEGA)}}{=} \begin{array}{c} \omega^{2^{-3}ac+2^{-2}a^2d} \\ \text{red spider} \end{array} \begin{array}{c} a, 0 \end{array}$$

Proof.



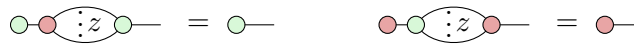
The second equation follows from Lemmas 28 and 41. □

Lemma 44. Parallel multipliers sum: for any $x, y \in \mathbb{Z}_p$:

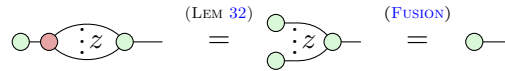


Proof. This is a straightforward consequence of FUSION. □

Lemma 45. For any $z \in \mathbb{Z}_p^*$,

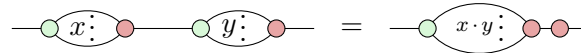


Proof.

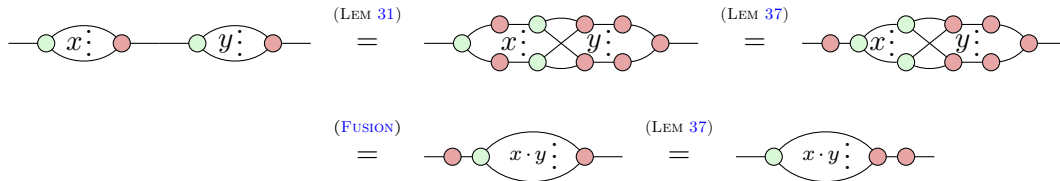


The other rule follows from COLOUR. □

Lemma 46. For any $x, y \in \mathbb{N}$,



Proof.

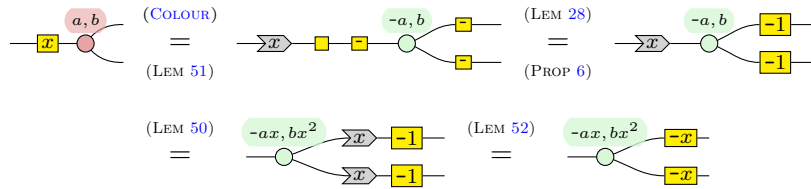


The second equality follows from COLOUR. □

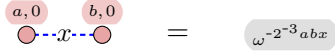
Lemma 47. For any $x \in \mathbb{Z}_p^*$,



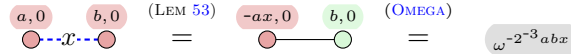
The other equation can be proved similarly,



Lemma 54.



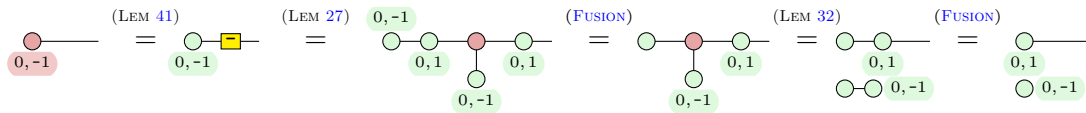
Proof.



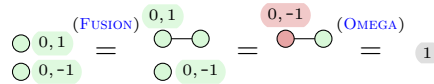
Lemma 55. Any purely-Clifford states can be represented in both the red and green fragment: for any $x \in \mathbb{Z}_p^*$,



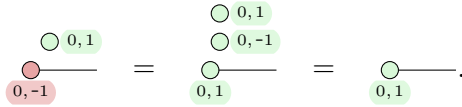
Proof. Firstly, we prove the subcase $x = 1$ of (a):



so that



and



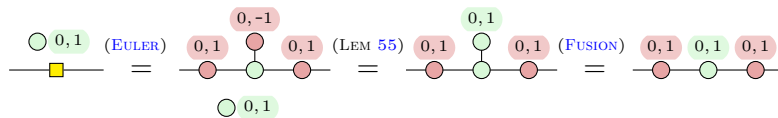
Then the general case for any invertible x follows using lemma 49.

(b) follows once again using COLOUR. □

Lemma 56. The Hadamards admit more standard Euler decompositions (originally shown for qudit ZX in [66]):



Proof.



We obtain the second derivation, as always, using COLOUR. □

In the next few proofs, we make frequent use of the following fact:

Lemma 57. For any $x \in \mathbb{Z}_p$, there are $a, b \in \mathbb{Z}_p$ such that $x = a^2 + b^2$.

Lemma 4. Strictly-Clifford states can all be represented both using Z- and X-spiders: for any $a \in \mathbb{Z}_p$ and $b \in \mathbb{Z}_p^*$,

$$\begin{array}{c} a, b \\ \circ \text{---} \end{array} = \begin{array}{c} ab^{-1}, -b^{-1} \\ \circ \text{---} \\ \circ -ab^{-1}, b^{-1} \end{array} \qquad \begin{array}{c} a, b \\ \circ \text{---} \end{array} = \begin{array}{c} -ab^{-1}, -b^{-1} \\ \circ \text{---} \\ \circ -ab^{-1}, b^{-1} \end{array}$$

Proof. Firstly,

$$\begin{array}{c} \circ \text{---} \\ ab^{-1}, b^{-1} \end{array} \stackrel{\text{(LEM 53)}}{=} \begin{array}{c} \circ \text{---} \\ -a, b \end{array} \stackrel{\text{(LEM 59)}}{=} \begin{array}{c} \circ \text{---} \\ -a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(LEM 40)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array}$$

so that

$$\begin{array}{c} a, b \\ \circ \text{---} \\ \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(LEM 23)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ ab^{-1}, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ ab^{-1}, 0 \end{array}$$

whence

$$\begin{array}{c} a, b \\ \circ \text{---} \\ \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(LEM 25)}}{=} \begin{array}{c} a, b \\ \circ \text{---} \\ \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} a, b \\ \circ \text{---} \\ \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(LEM 5)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(LEM 30)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ ab^{-1}, 0 \end{array} \stackrel{\text{(LEM 32)}}{=} \begin{array}{c} \circ \text{---} \\ -ab^{-1}, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ -ab^{-1}, 0 \end{array}$$

and, finally,

$$\begin{array}{c} \circ \text{---} \\ a, b \end{array} \stackrel{\text{(LEM 23)}}{=} \begin{array}{c} \circ \text{---} \\ a, b \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, b^{-1} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ -ab^{-1}, 0 \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ -ab^{-1}, -b^{-1} \end{array}$$

The change of colour in the scalar, as well as the second derivation, follow using **COLOUR**. \square

Lemma 61. The following states are equivalent:

$$\begin{array}{c} 0, z \\ \circ \text{---} \end{array} = \begin{array}{c} 0, z^{-1} \\ \circ \text{---} \end{array}$$

Proof.

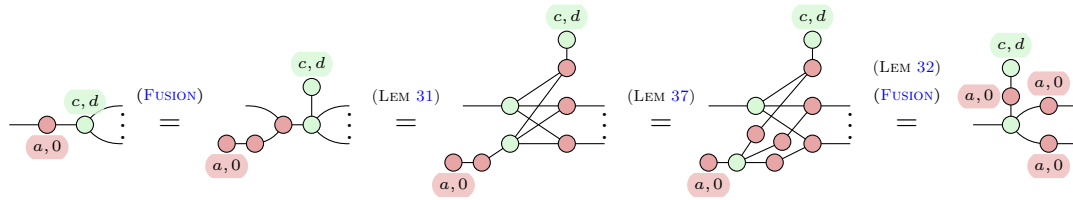
$$\begin{array}{c} 0, z \\ \circ \text{---} \end{array} \stackrel{\text{(FUSION)}}{=} \begin{array}{c} \circ \text{---} \\ 0, z \end{array} \stackrel{\text{(LEM 4)}}{=} \begin{array}{c} \circ \text{---} \\ 0, z^{-1} \end{array} \begin{array}{c} \circ \text{---} \\ 0, -z^{-1} \end{array} \stackrel{\text{(OMEGA)}}{=} \begin{array}{c} \circ \text{---} \\ 0, z^{-1} \end{array}$$

\square

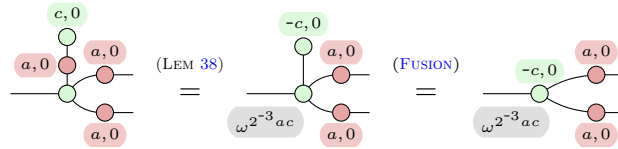
Lemma 62. X-spiders with arbitrary phases copy Pauli Z-spiders and vice-versa:

$$\begin{array}{c} c, d \\ \circ \text{---} \\ \circ \text{---} \\ a, 0 \end{array} \approx \begin{array}{c} ad-c, d \\ \circ \text{---} \\ \circ \text{---} \\ a, 0 \end{array} \begin{array}{c} a, 0 \\ \circ \text{---} \\ \circ \text{---} \\ a, 0 \end{array} \qquad \begin{array}{c} c, d \\ \circ \text{---} \\ \circ \text{---} \\ a, 0 \end{array} \approx \begin{array}{c} c-ad, d \\ \circ \text{---} \\ \circ \text{---} \\ -a, 0 \end{array} \begin{array}{c} -a, 0 \\ \circ \text{---} \\ \circ \text{---} \\ -a, 0 \end{array}$$

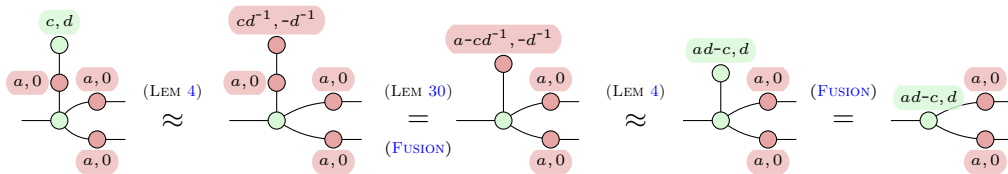
Proof. First of all,



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



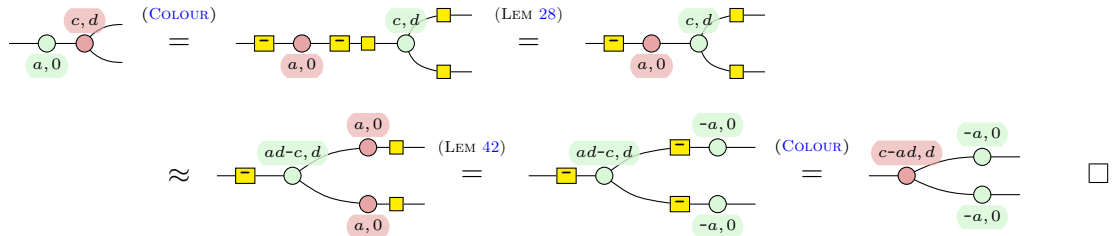
Note that if $d = 0$, then $ad - c = -c$ and so the lemma holds. Otherwise, $d \neq 0$ and therefore d^{-1} exists, so we can apply the state-change lemma:



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

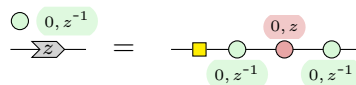
$$-(a - cd^{-1})(-d^{-1})^{-1}, -(-d^{-1})^{-1} = -(a - cd^{-1})(-d), d = ad - c, d$$

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



We are now ready to prove that axioms MULT and SHEAR of [13] are derivable from our simplified set of axioms:

Proposition 63. For any $z \in \mathbb{Z}_p^*$,



|

B.3 Graph-like diagrams

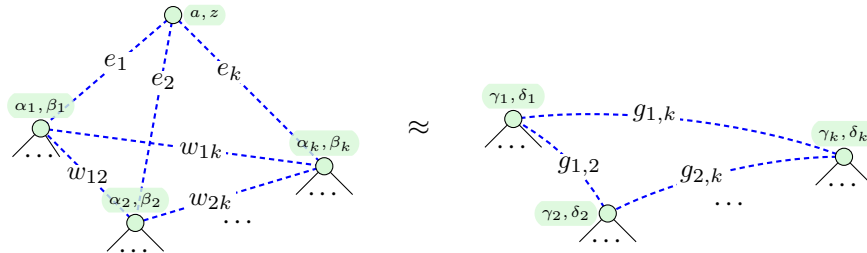
Proposition 65. γ -weighted local \mathbb{Z}_d -complementation is derivable in $\text{ZX}_p^{\text{Stab}}$, for any graph $G = (V, E)$, $\gamma \in \mathbb{Z}_p$ and $u \in V$,

$$G \left. \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{array} \right\} N_G(u) \approx G \star u \quad (8)$$

| Proof. Same as Lemma 12 of Ref. [14]. □

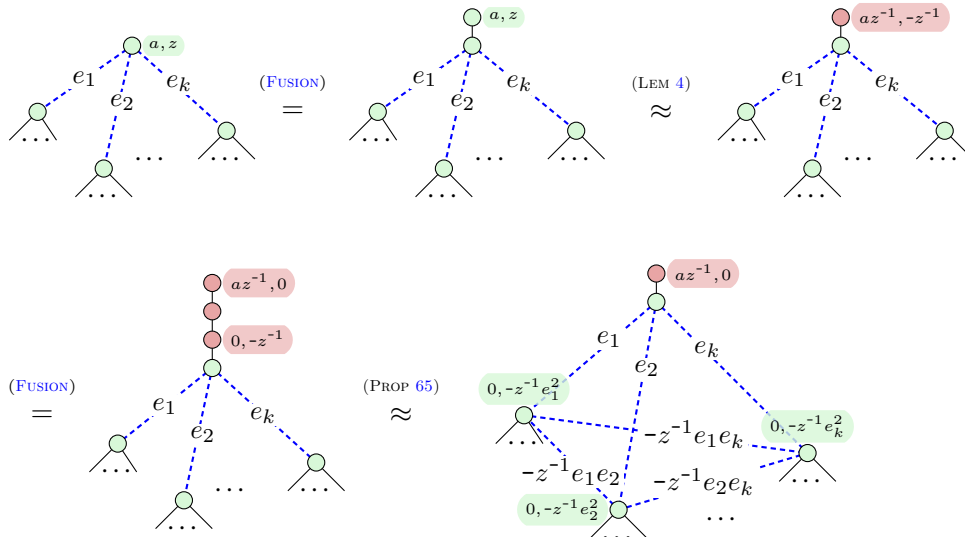
B.3.1 Local complementation simplification

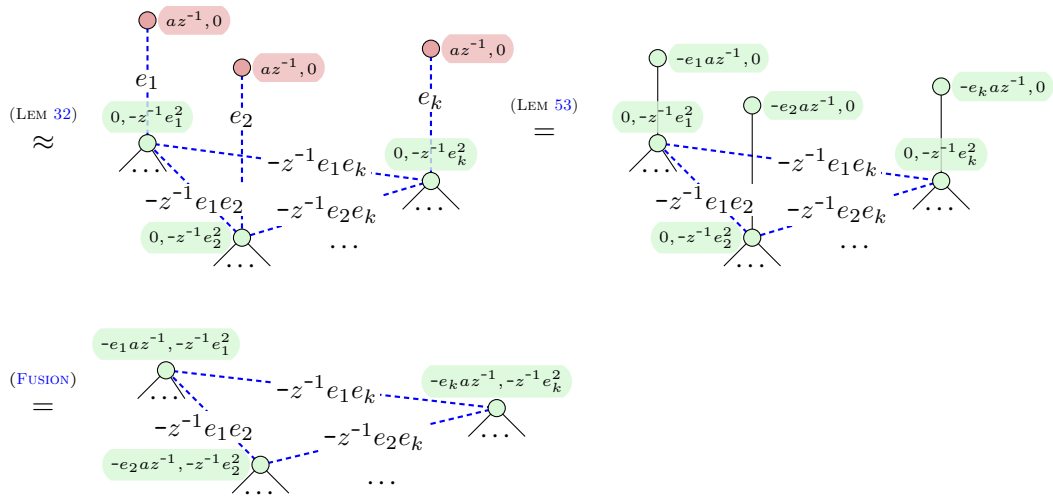
Lemma 8 (Local complementation simplification). For any $z \in \mathbb{Z}_p^*$ and for all $a, \alpha_i, \beta_i, e_i, w_{i,j} \in \mathbb{Z}_p$ where $i, j \in \{1, \dots, k\}$ such that $i < j$ we have:



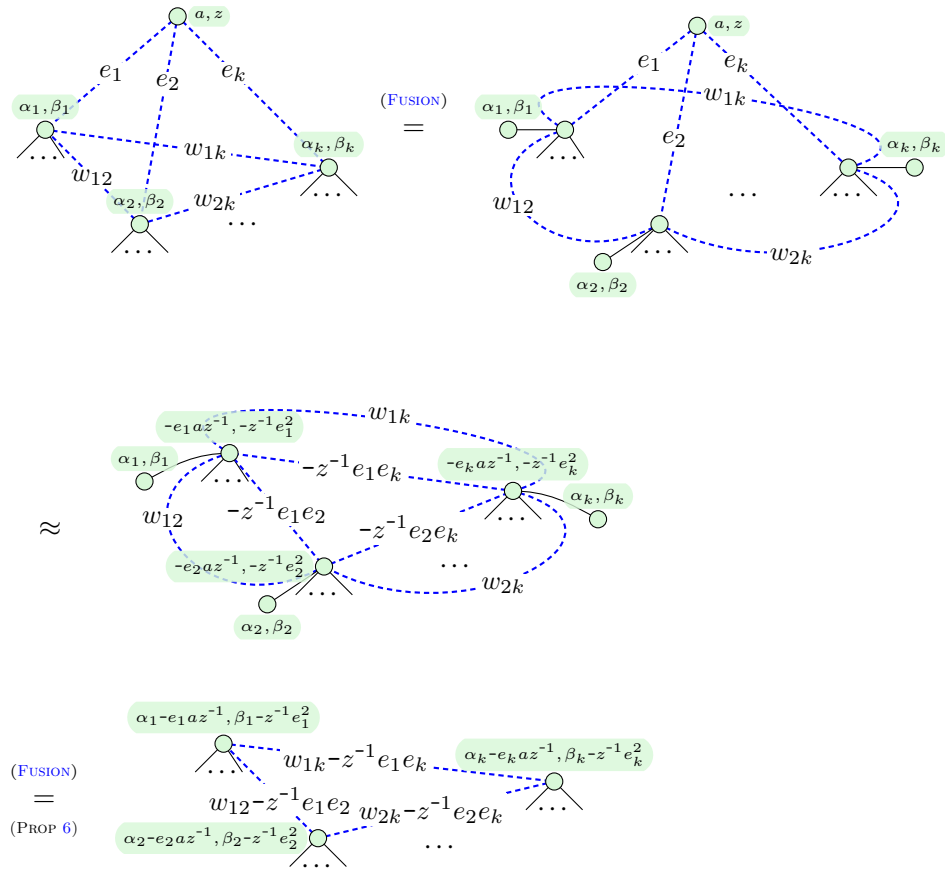
Here $\gamma_i = \alpha_i - e_i a z^{-1}$, $\delta_i = \beta_i - z^{-1} e_i^2$, and $g_{i,j} = w_{ij} - z^{-1} e_i e_j$.

Proof. First, we can prove a simplified version of the lemma without phases of the boundary spiders and H-edges as follows:





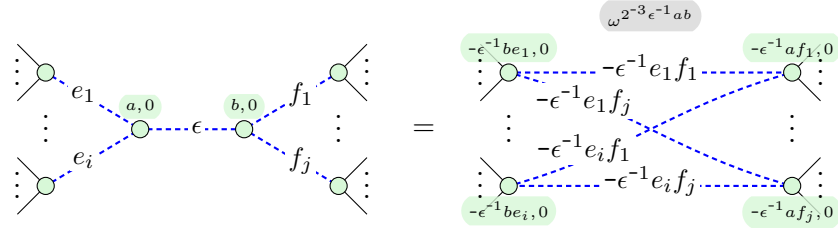
Then, we can use the previous equation to prove the lemma.



□

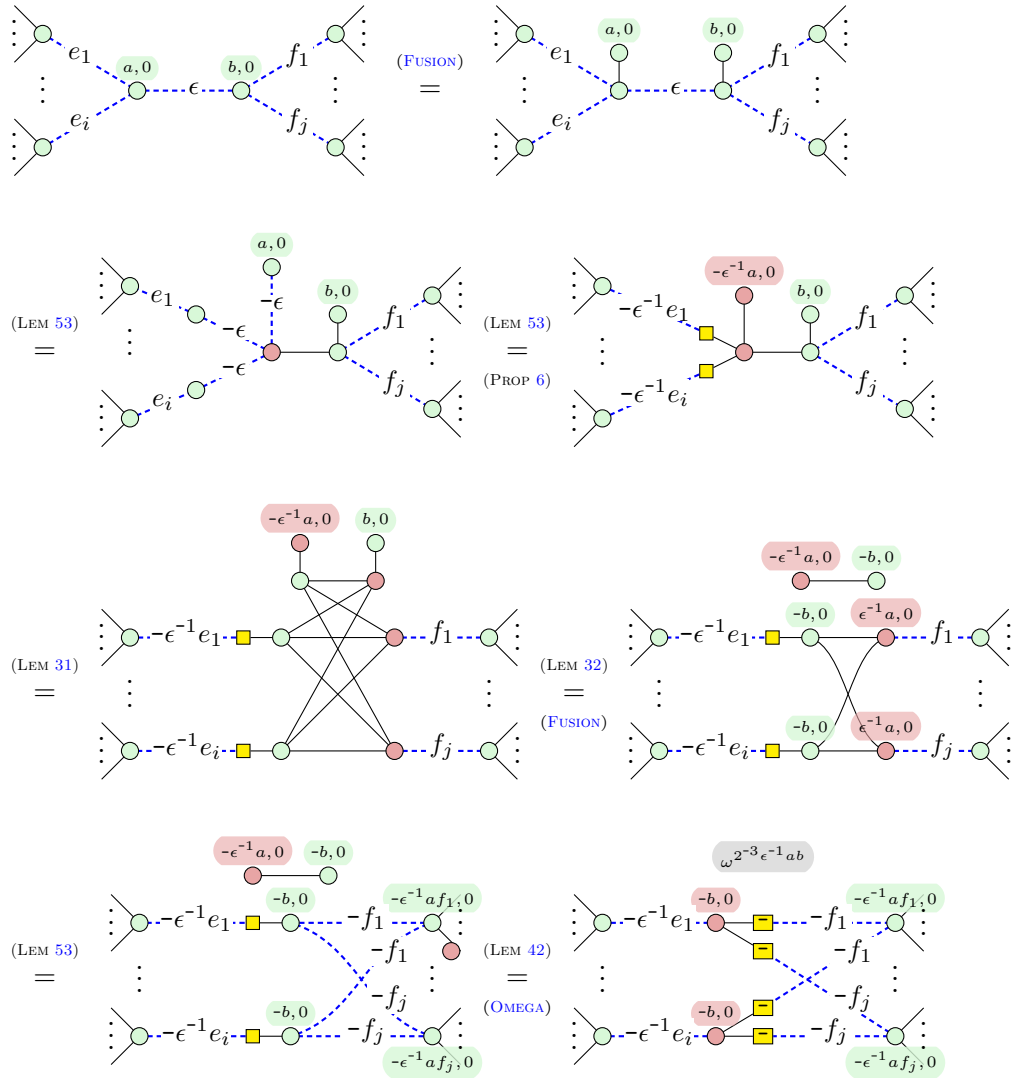
B.3.2 Pivoting simplification

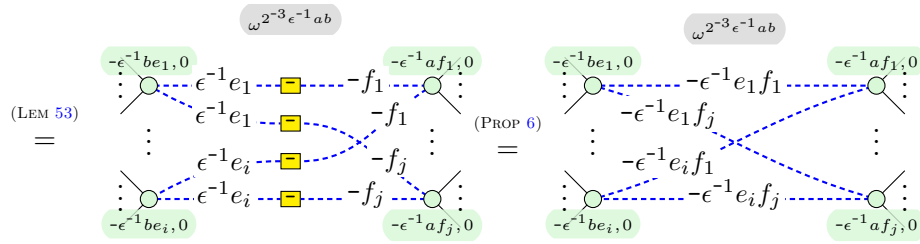
Lemma 9. The following version of pivoting is derivable in ZX_p^{Stab} :



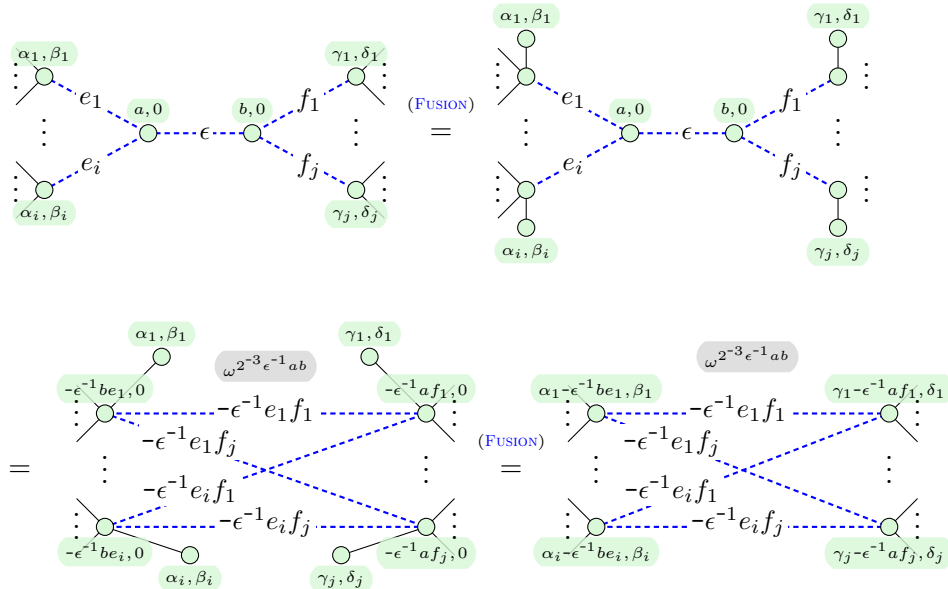
Here $\epsilon \in \mathbb{Z}_p^*$ and all the other variables are allowed arbitrary values.

Proof. First, we can prove a simplified version of the equation that omits the phases of boundary spiders as follows,





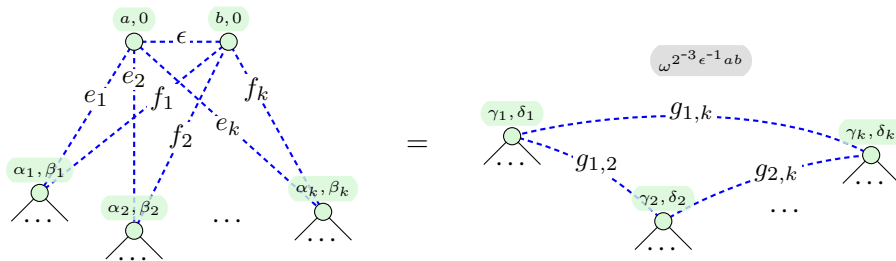
Then, we can use the previous equation to prove the lemma as follows:



□

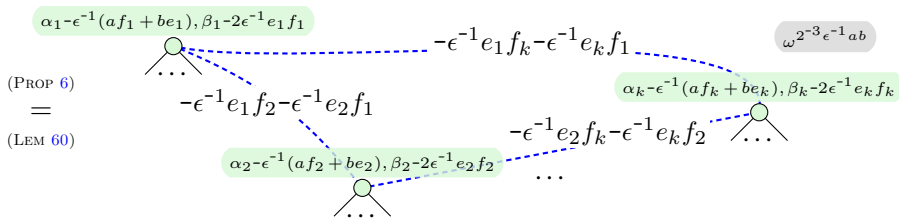
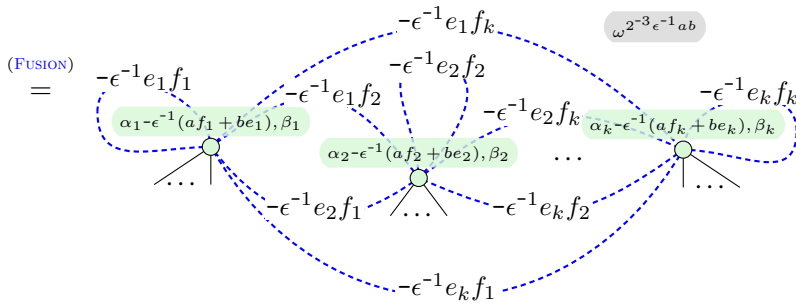
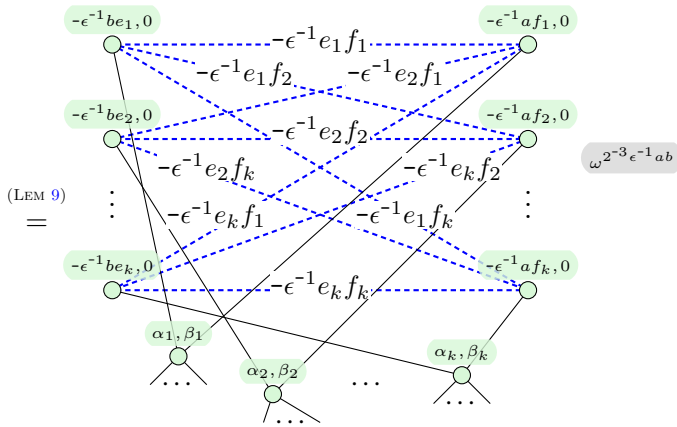
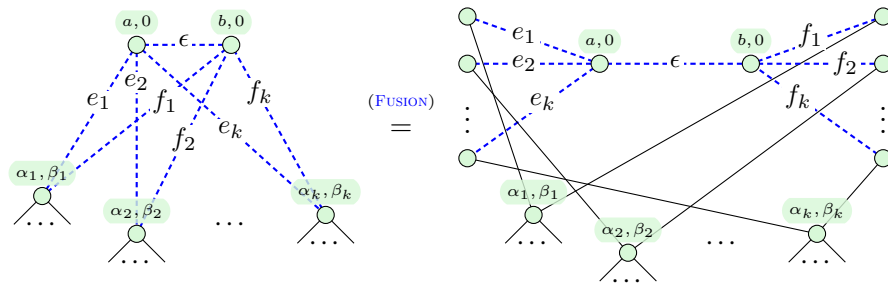
Now, we prove the general version of pivoting.

Lemma 10 (Pivoting simplification). General pivoting is derivable in ZX_p^{Stab} :



Here again $\epsilon \in \mathbb{Z}_p^*$ with every other variable on the left-hand side allowed arbitrary values. On the right-hand side $\gamma_i = \alpha_i - \epsilon^{-1}(af_i + be_i)$, $\delta_i = \beta_i - 2\epsilon^{-1}e_i f_i$, and $g_{i,j} = -\epsilon^{-1}(e_i f_j + e_j f_i)$.

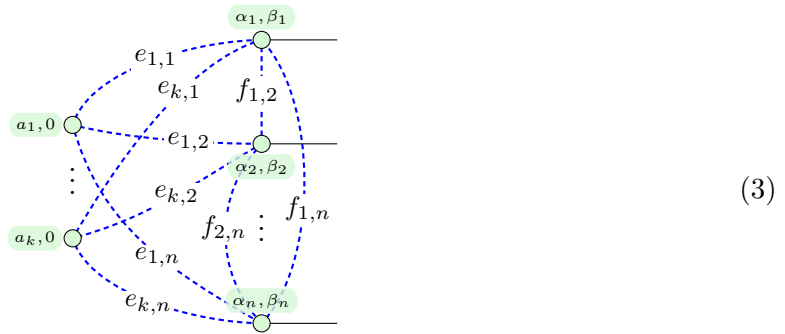
Proof.



□

C A normal form

Lemma 12. A general non-zero n -qupit diagram in AP-form is described by the diagram:



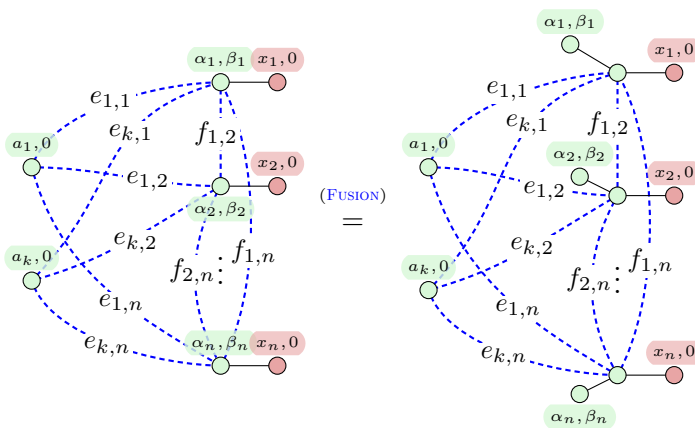
where $a_l, \alpha_i, \beta_i, e_{h,i}, f_{i,j} \in \mathbb{Z}_p$ with $l \in \{1, \dots, k\}$ and $i, j \in \{1, \dots, n\}$ such that $i < j$. The interpretation of this diagram is (up to some non-zero scalar) equal to a state

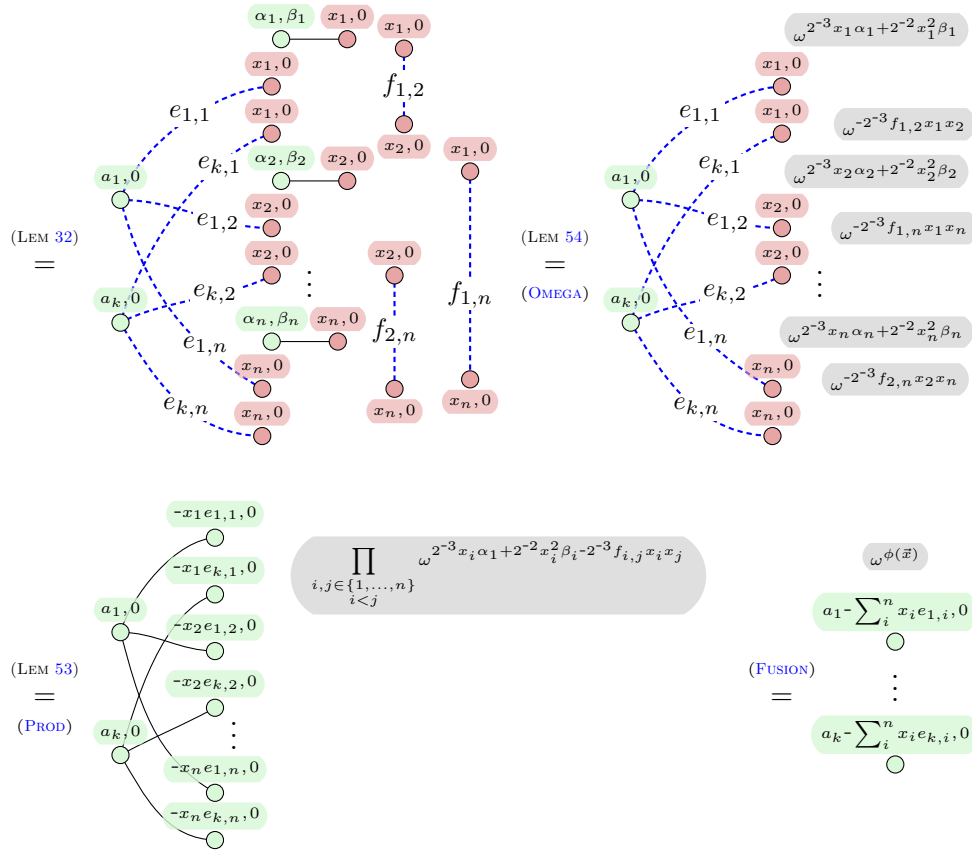
$$\sum_{E\vec{x}=\vec{a}} \omega^{\phi(\vec{x})} |\vec{x}\rangle \tag{4}$$

where E is the weighted bipartite adjacency matrix of the internal and boundary spiders, \vec{a} describes the Pauli phases of the internal spiders, and ϕ is a phase function that describes the connectivity and phases of the boundary spiders:

$$E = \begin{bmatrix} e_{1,1} & \cdots & e_{1,n} \\ e_{2,1} & \cdots & e_{2,n} \\ \vdots & & \vdots \\ e_{k,1} & \cdots & e_{k,n} \end{bmatrix}, \quad \vec{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix}, \quad \phi(\vec{x}) = \sum_{\substack{i,j \in \{1, \dots, n\} \\ i < j}} 2^{-3} x_i \alpha_i + 2^{-2} x_i^2 \beta_i - 2^{-3} f_{i,j} x_i x_j$$

Proof. We can prove this claim purely diagrammatically, by composing the diagram of Equation (3) with an effect that corresponds to the vector $|x\rangle$. By rewriting the diagram while keeping track of the scalars, we can prove that the diagram indeed represents the one described in Equation (4). These transformations are as follows:





Note that if a Z-spider with no legs has phase $(z, 0)$ for any $z \in \mathbb{Z}_p^*$, then it equals the zero scalar. This means that the probability of such an effect is 0. Therefore, the above diagram allows only such \vec{x} vectors that satisfy the equation $E\vec{x} = \vec{a}$. Furthermore, the scalars that are copied from the phases part of the diagram equal the $\omega^{\phi(\vec{x})}$ component of the equation. We conclude that a diagram in Equation (3) indeed equals the state presented in Equation (4). \square

C.1 Completeness

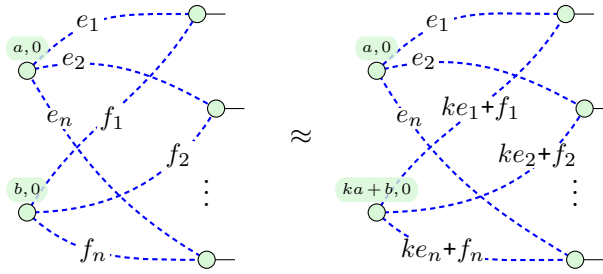
Lemma 14. For any non-zero state $|\psi\rangle$, there is at most one triple (E, \vec{a}, ϕ) satisfying the conditions of reduced AP-form such that:

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

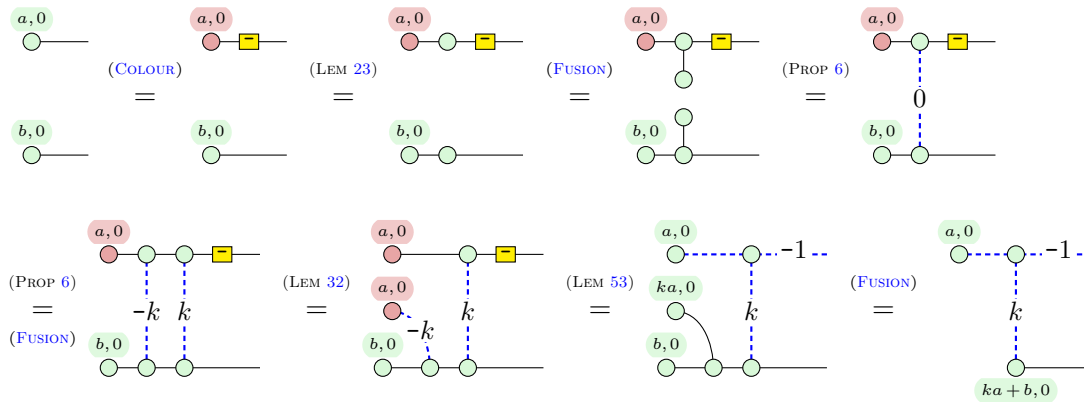
Proof. Since $|\psi\rangle \neq 0$, the set $\mathcal{A} = \{\vec{x} \mid E\vec{x} = \vec{a}\}$ is non-empty. Therefore, there is a unique system of equations in RREF that define \mathcal{A} . This means that E and \vec{a} are uniquely fixed. Now, for any assignment $\{x_{i_1} := c_1, \dots, x_{i_k}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

Lemma 15. We can perform primitive row operations on a ZX-diagram in AP-form, i.e., we can

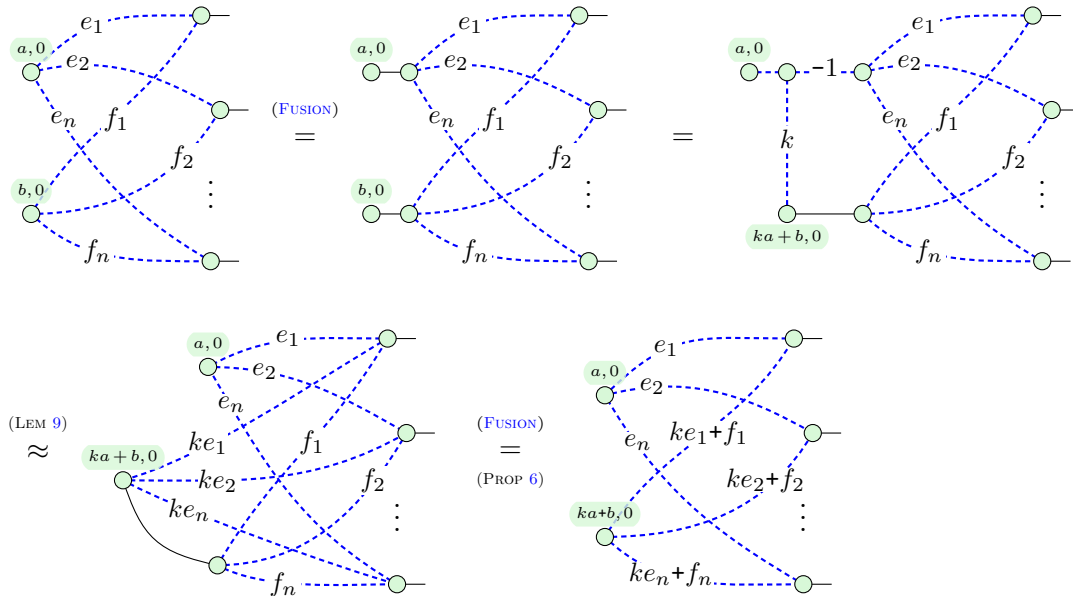
“add” one inner spider to another. For any $k, a, b, e_i, f_j \in \mathbb{Z}_p$ where $i \in 1, \dots, n$ and $j \in 1, \dots, m$:



Proof. Firstly, we show that we can transform two disconnected X-states:



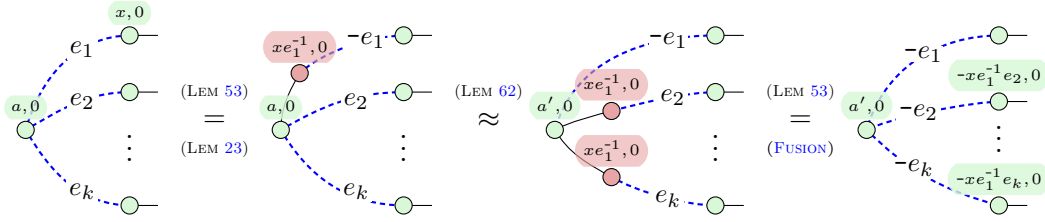
Then, we can show that we can transform a diagram in AP-form as follows:



□

Lemma 66. We can remove Pauli-phases from the pivot spiders of diagrams in AP-form.

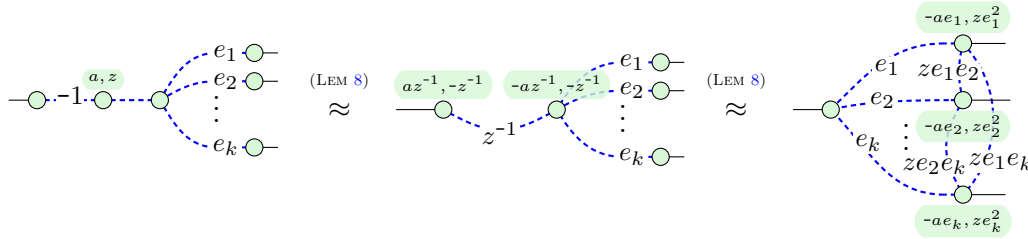
Proof. For any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $e_1 \in \mathbb{Z}_p^*$:



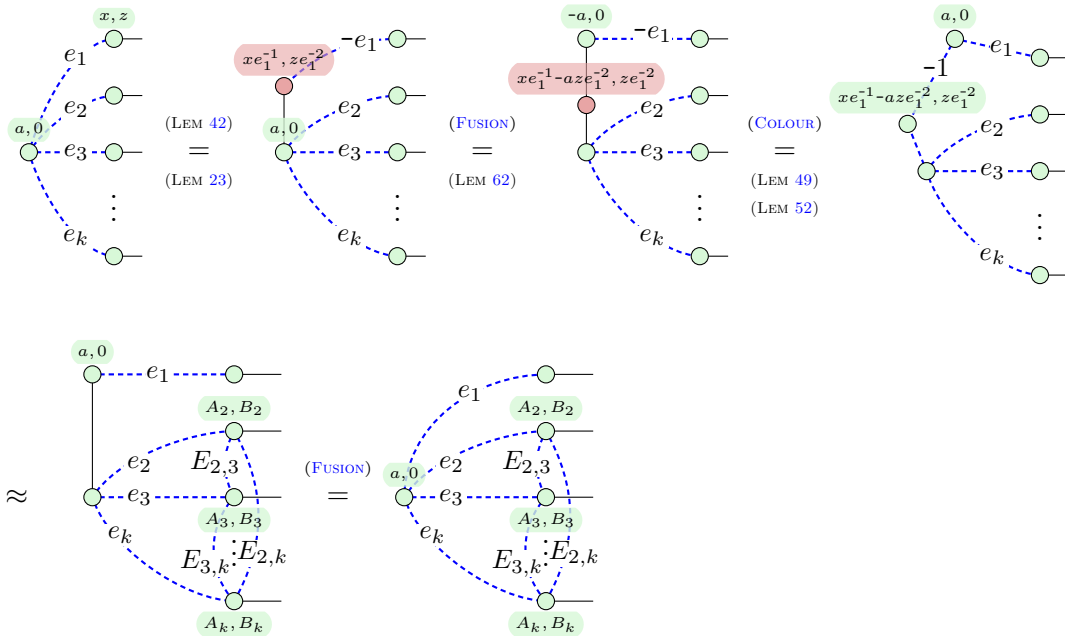
where $a' := -(a + xe_1^{-1})$. □

Lemma 67. We can remove strictly-Clifford phases from the pivot spiders of diagrams in AP-form.

Proof. To prove this case, we first show that we can push strictly-Clifford Z-spider through an X-spider with weighted outputs. That is, for any $a, e_i \in \mathbb{Z}_p$ where $i \in \{1, \dots, k\}$ and $z \in \mathbb{Z}_p^*$:



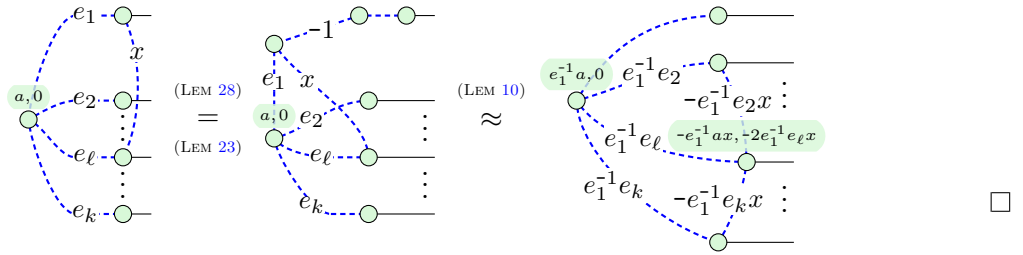
Therefore, for any $a, x, e_i \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$ and $z, e_1 \in \mathbb{Z}_p^*$:



where $A_i = (aze^{-1} - x)e^{-1}e_i$, $B_i = ze_1^{-2}e_i^2$, and $E_{i,j} = ze_1^{-2}e_i e_j$. □

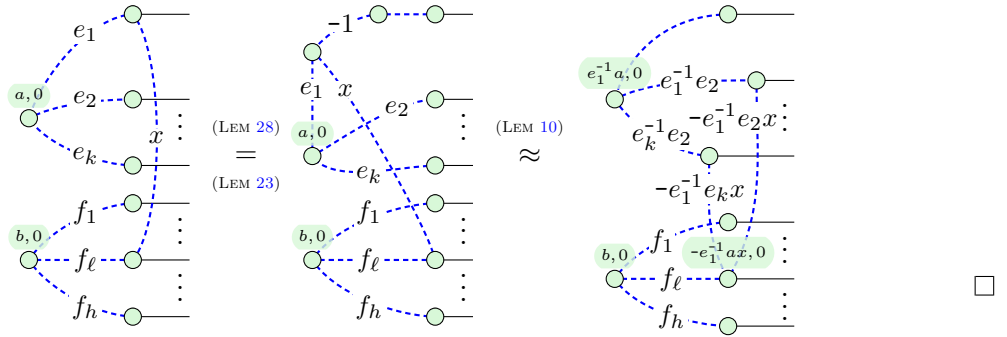
Lemma 68. We can remove an H-edge between the pivot spider and a boundary spider that connects to the same internal spider as the pivot.

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



Lemma 69. We can remove an H-edge between the pivot spider and a boundary spider that does not connect to the same internal spider as the pivot.

Proof. For any $a, b, x, e_i, f_h \in \mathbb{Z}_p$ where $i \in \{2, \dots, k\}$, $h \in \{1, \dots, j\}$ and $e_1 \in \mathbb{Z}_p^*$:



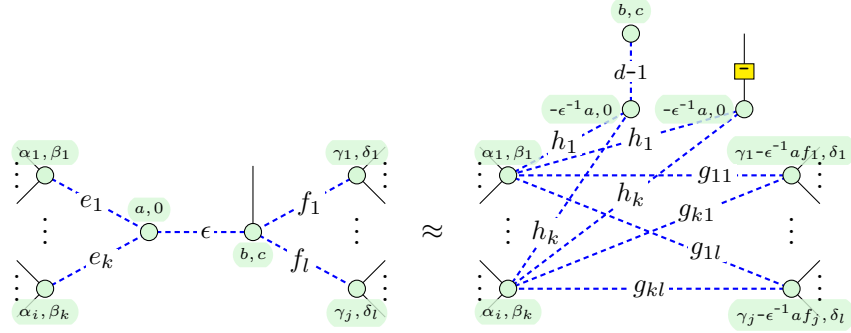
Lemma 17. Any diagram in ZX_p^{Stab} can be converted into one in reduced AP-form.

Proof. First, we can convert any diagram in ZX_p^{Stab} into one in AP-form using local complementation and pivoting. Then, we can translate such a diagram into AP-form with a biadjacency matrix in RREF using Gaussian elimination, as demonstrated in Lemma 15. Furthermore, we have established the proofs for removing any phase from the pivot spider (Lemma 66 and Lemma 67), as well as removing any H-edge connected to the pivot spider (Lemma 68 and Lemma 69). These results allow us to transform a diagram in such a way that its phase function ϕ only contains free variables from the equation system $E\vec{x} = \vec{a}$. Consequently, we can conclude that any diagram in ZX_p^{Stab} can be rewritten into a form that satisfies the necessary properties to be considered a diagram in reduced AP-form. \square

Theorem 18 (Completeness). For any pair of ZX-diagrams $A, B \in ZX_p^{\text{Stab}}$, if $\llbracket A \rrbracket = \llbracket B \rrbracket$, we can provide a sequence of rewrites that transforms A into B .

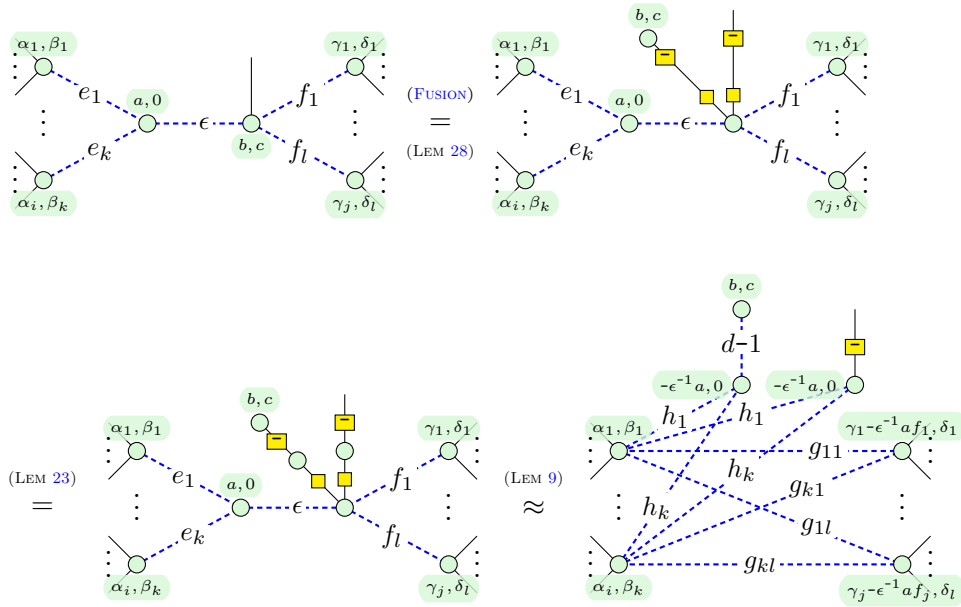
Proof. Without loss of generality, we can assume that both A and B are states by map-state duality. If A and B represent the same linear map, i.e. $\llbracket A \rrbracket = \llbracket B \rrbracket$, then their reduced AP-forms are identical, thanks to the uniqueness of the form proved in Lemma 14. Therefore, we can transform both A and B into diagrams in reduced AP-form using Lemma 17. The sequence of transformations from A to A in reduced AP-form, composed with the series of rewrites from B in reduced AP-form to B , provides us with a sequence of rewrites that transforms A into B . \square

Lemma 20. The following boundary pivot rule is derivable in ZX_p^{Stab} :



Here $g_{ij} := -\epsilon^{-1}e_i f_j$ and $h_i := -\epsilon^{-1}e_i$. This rule holds for all choices of phases as long as $\epsilon \neq 0$.

Proof. Unfuse spiders and introduce Hadamards as follows:



Where in the last step we applied the regular pivot Lemma 9. □

Floquetifying the Colour Code

Alex Townsend-Teague, Julio Magdalena de la Fuente and Markus Kesselring

Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany

alex.townsend-teague@outlook.com

Floquet codes are a recently discovered type of quantum error correction code. They can be thought of as generalising stabilizer codes and subsystem codes, by allowing the logical Pauli operators of the code to vary dynamically over time. In this work, we use the ZX-calculus to create new Floquet codes that are in a definable sense equivalent to known stabilizer codes. In particular, we find a Floquet code that is equivalent to the colour code, but has the advantage that all measurements required to implement it are of weight one or two. Notably, the qubits can even be laid out on a square lattice. This circumvents current difficulties with implementing the colour code fault-tolerantly, while preserving its advantages over other well-studied codes, and could furthermore allow one to benefit from extra features exclusive to Floquet codes. On a higher level, as in Ref. [BLN⁺23], this work shines a light on the relationship between ‘static’ stabilizer and subsystem codes and ‘dynamic’ Floquet codes; at first glance the latter seems a significant generalisation of the former, but in the case of the codes that we find here, the difference is essentially just a few basic ZX-diagram deformations.

1 Introduction

In 2021, Hastings and Haah discovered the honeycomb code [HH21, [E_C₂₀₀](#)]. At first glance, it looked a lot like a subsystem code [KLP05, [E_C₂₀₀](#)]; it was defined via a sequence of non-commuting Pauli measurements whose individual outcomes were random, but combined to give deterministic outcomes suitable for use in catching errors that occur during quantum computation. But something didn’t quite add up. Viewed exactly as a subsystem code, it encoded no logical information. This was because the logical information was instead ‘dynamically’ encoded. This made it the first member of a new class of codes, which have come to be called *Floquet codes* [Vui21, [E_C₂₀₀](#)]. Since its publication, it seems a number of people have been holed up in their offices thinking about Floquet codes, because of late a new one has popped up every month or so [KFT⁺22, DTB22, AWH22, ZAV22, Bau23, SWP23]. In particular, in both Ref. [KFT⁺22] and Ref. [DTB22], a whole family of Floquet codes is described, of which the honeycomb code is a single member. We call this family the *condensed colour codes*, as in [KFT⁺22].

Plot-twist 1.1. *Before the honeycomb code paper was published, another Floquet code had already independently been discovered.*

In Ref. [BLN⁺23], the authors write that Hector Bombín had previously discovered an equivalence between the rotated surface code [BMD07, [E_C₂₀₀](#)] and a (then unnamed) condensed colour code. This equivalence is shown in Ref. [BLN⁺23] using the ZX-calculus, a flexible but rigorous graphical formalism for quantum mechanics [CD11, CK17]. Given any ZX-diagram representing an error correction protocol, something the ZX-calculus is particularly good at is identifying ways of rewriting high-weight Pauli measurements as weight-two or weight-one Pauli measurements. The authors of Ref. [BLN⁺23] applied this idea to a ZX-diagram representing multiple rounds of rotated surface code measurements. Interestingly, Craig Gidney had himself previously done something very similar in Ref. [Gid22] to find

an implementation of the rotated surface code that uses only weight-two measurements. The only difference is that Gidney applied the idea to a circuit representing a *single* round of rotated surface code measurements. In both cases all measurements were reduced to weight-two, but only in the former case was the result a Floquet code¹.

The colour code [BMD06, \mathbb{E}_{260}] has certain properties that make it arguably more appealing than the surface code, such as a higher encoding rate [LRA14], transversal Clifford gates [BMD06] and more efficient lattice surgery operations [TKBB22]. The high-weight measurements naively required for its implementation, however, are the major obstacle to realising it practically [CKYZ20]. So in the same way that Ref. [BLN⁺23] ‘Floquetified’ the rotated surface code, it would be great to have a ‘Floquetified’ colour code - that is, a Floquet code that is in a definable sense equivalent to the colour code - in which all measurements are weight-two or less. Perhaps even more excitingly, Floquet codes can come equipped with the ability to perform logical Clifford gates both fault-tolerantly and at no extra effort; this is discussed in detail in Ref. [AWH22]. The honeycomb code, for example, naturally performs a fault-tolerant logical Hadamard gate every three timesteps. Exactly which logical gates can be implemented by a Floquet code in this manner is restricted by the automorphism group of what condensed matter theorists call the *anyonic defects* of the code. The honeycomb code can be shown to have exactly one non-trivial such automorphism, which corresponds exactly to the logical Hadamard gate. A Floquetified colour code, however, would in principle inherit its automorphism group from the colour code - this is much richer, containing 72 elements [Yos15, KPEB18]. So in such a code, the set of logical Clifford gates that could potentially be fault-tolerantly implemented in this way is larger.

To this end, we aimed to use the ideas from Ref. [BLN⁺23] to Floquetify the colour code. We succeeded, finding a Floquet code with period 13 whose qubits can be laid out on a square lattice, and in which all measurements are weight one or two. This paper proceeds as follows. In [Section 2](#), we introduce the definitions and notation we’ll need throughout the paper. First, we introduce *ISG codes*, of which stabilizer codes [Got97, \mathbb{E}_{260}], subsystem codes and Floquet codes are subtypes (thus far, we’re not aware of any universally accepted formal definition of a Floquet code in the literature). We also import the graphical formalism of *Pauli webs* from Ref. [BLN⁺23] (generalised to *stabilizer flow* in Ref. [MBG23]), which allows us to reason graphically about stabilizers, logical operators and *detectors*. In [Section 3](#), we jump in the shallow end by Floquetifying the $[[4, 2, 2]]$ code [VGW96, \mathbb{E}_{260}], demonstrating the key ideas behind this Floquetification process on a simple example. The deep end awaits in [Section 4](#), where we Floquetify the colour code. We include many extra details in appendices; these will be signposted throughout the main text.

2 Preliminaries

We will assume familiarity with stabilizer codes and the stabilizer formalism [Got97], as well as the ZX-calculus. For the uninitiated, good introductory references are Ref. [NC10, Section 10.5] or Ref. [Got09] for the former, and Ref. [Wet20] for the latter. In the appendix, we also include a reminder of how measurement works in the stabilizer formalism - see [Theorem A.5](#).

¹Similar in spirit to Ref. [BLN⁺23] is Ref. [Bau23]; both can be viewed as using graphical tensor network approaches to construct new Floquet codes from existing stabilizer codes. However, the latter’s author approaches things through the lens of *topological spacetime path-integrals*, and uses a tensor network approach that is closely related to, but isn’t exactly the same as, the ZX-calculus. Nonetheless, for *topological codes*, we believe the two approaches are equivalent.

2.1 ISG codes

We begin by introducing *ISG codes*, where ‘ISG’ stands for *instantaneous stabilizer group*, a term introduced in Ref. [HH21]. But first, some notation; we’ll use $\sigma_0 = I$, $\sigma_1 = X$, $\sigma_2 = Y$ and $\sigma_3 = Z$ to denote the *Pauli matrices*, and \mathcal{P}_1 to denote the *single qubit Pauli group* they form under composition. \mathcal{P}_n will then denote the *n-qubit Pauli group* $\mathcal{P}_n = \{p_1 \otimes \dots \otimes p_n \mid \forall j : p_j \in \mathcal{P}_1\}$ for $n > 1$. Every element of this group can be written in the form $i^\ell (\sigma_{j_1} \otimes \dots \otimes \sigma_{j_n})$, for $\ell \in \{0, 1, 2, 3\}$. This i^ℓ is often called a *phase*. One such element we’ll use a lot is $(\sigma_j)_k = I \otimes \dots \otimes I \otimes \sigma_j \otimes I \otimes \dots \otimes I$, which has an I in each tensor factor except the k -th, where we insert the Pauli matrix σ_j . Another common element is $\mathbb{1} = I \otimes \dots \otimes I$. By a slight abuse of notation, we will usually write $i^\ell \mathbb{1}$ or $i^\ell I$ as just i^ℓ . The *weight* of any element $i^\ell (\sigma_{j_1} \otimes \dots \otimes \sigma_{j_n})$ is the number of Pauli matrices σ_{j_k} that aren’t I .

Rather than defining qubits via Hilbert spaces, we’ll stick to group theory; we’ll simply define that we have a *system of n qubits* whenever we have any group \mathcal{G} isomorphic to \mathcal{P}_n . A particularly important example will be $\mathcal{G} = N(\mathcal{S})/\mathcal{S} \cong \mathcal{P}_{n-r}$, for any stabilizer group $\mathcal{S} \leq \mathcal{P}_n$ with *rank* (size of any minimal generating set) r [Got09, Section 3.4]. Recall that a stabilizer group is just any subgroup of \mathcal{P}_n that doesn’t contain $-\mathbb{1}$, and this forces it to be Abelian. Here, $N(\mathcal{S}) = \{p \in \mathcal{P}_n \mid p\mathcal{S}p^{-1} = \mathcal{S}\}$ denotes the *normalizer* of \mathcal{S} in \mathcal{P}_n . So $N(\mathcal{S})/\mathcal{S}$ is the quotient group consisting of *left cosets* of \mathcal{S} in $N(\mathcal{S})$, whose elements $p\mathcal{S} = \{ps \mid s \in \mathcal{S}\}$ will often be denoted \bar{p} for short. The *weight* of such a coset is the minimum weight over all its elements. For any groups $\mathcal{G}_1, \dots, \mathcal{G}_\ell$, we’ll write the product $\mathcal{G}_1 \dots \mathcal{G}_\ell$ to mean the group generated by the union of generating sets for $\mathcal{G}_1, \dots, \mathcal{G}_\ell$.

It can be shown that \mathcal{P}_n has presentation $\mathcal{P}_n = \langle i, X_1, Z_1, X_2, Z_2, \dots, X_n, Z_n \rangle$. Thus any group \mathcal{G} isomorphic to \mathcal{P}_n has presentation $\langle \iota, x_1, z_1, x_2, z_2, \dots, x_n, z_n \rangle$, for some elements ι, x_j, z_j in \mathcal{G} , with group isomorphism $\mathcal{G} \cong \mathcal{P}_n$ given by $\iota \mapsto i, x_j \mapsto X_j$ and $z_j \mapsto Z_j$. In particular, all the x_j and z_j generators obey the same commutativity relations as the Paulis X_j and Z_j . If we think of \mathcal{G} as defining n qubits, we can identify qubit j with the subgroup $\langle \iota, x_j, z_j \rangle \cong \langle i, X_j, Z_j \rangle \cong \mathcal{P}_1$.

We can now define an *ISG code*. Given n qubits, an ISG code is defined entirely by a *measurement schedule* \mathcal{M} , which is an ordered list $[\mathcal{M}_0, \mathcal{M}_1, \dots]$ of Abelian subgroups of \mathcal{P}_n . The schedule \mathcal{M} can be finite or infinite. If it’s finite, with length ℓ , say, then we let the subscript in \mathcal{M}_j be modulo ℓ . Given any such \mathcal{M} , there exists a subgroup \mathcal{S}_t of \mathcal{P}_n for all $t \in \mathbb{Z}$ called the *instantaneous stabilizer group* (ISG). This is defined recursively: for $t < 0$, it’s always the trivial group $\{\mathbb{1}\}$. For $t \geq 0$, \mathcal{S}_t is formed from \mathcal{S}_{t-1} by measuring a generating set for \mathcal{M}_t ; the effect of this can be determined using the stabilizer formalism (Theorem A.5). This is well-defined, in that it doesn’t depend on the choice of generating set for \mathcal{M}_t . We’ll often call t the *timestep* (or just *time*). At every timestep $t \in \mathbb{Z}$, let r_t denote the rank of \mathcal{S}_t , and let $k_t := n - r_t$. Then $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_{k_t}$. That is, we can consider ourselves to have a system of k_t qubits. This is the idea behind an *ISG code*.

Definition 2.1. An $[[n, k, d]]$ *ISG code* is given by a measurement schedule \mathcal{M} , with the property that for some $T \in \mathbb{Z}_{\geq 0}$ and all $t \geq T$, the group \mathcal{S}_t has some fixed rank r . We say that for all such timesteps $t \geq T$, the code is *established*. This encodes $k = n - r$ **logical qubits** whenever $t \geq T$, via the **logical Pauli group** $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$. The **distance** d is the minimum weight of any element of $N(\mathcal{S}_t)/\mathcal{S}_t$, over all $t \geq T$. The **period** is the length $\ell \in \mathbb{Z}_{\geq 0} \cup \{\infty\}$ of the list $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$, and can be finite or infinite.

For a slightly longer discussion of this definition, see Appendix C. As advertised, stabilizer codes, subsystem codes and Floquet codes are types of ISG code.

Definition 2.2. A *stabilizer code* is an ISG code $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$ such that the group $\mathcal{M}_0\mathcal{M}_1 \dots$

generated by the union of generating sets of all \mathcal{M}_t in \mathcal{M} is itself Abelian.

A stabilizer code has the property that, after establishment at time T , the ISG \mathcal{S}_t is the same for all $t \geq T$. There thus exist fixed Paulis $x_1, z_1, \dots, x_k, z_k \in \mathcal{P}_n$ such that $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$ is a presentation for $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$ for all $t \geq T$. This latter property is baked into the next definition, which is admittedly a bit of a mouthful, and can be skipped by any readers not already familiar with subsystem codes. Therein, for a group \mathcal{G} , we let $Z(\mathcal{G}) = \{g \in \mathcal{G} \mid gh = hg \ \forall h \in \mathcal{G}\}$ denote its *center*, and \mathcal{P}_k° be the ‘almost Pauli group’ $\langle X_1, Z_1, \dots, X_k, Z_k \rangle$.

Definition 2.3. A *subsystem code* is an ISG code $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$ that establishes at time T , such that the group $\mathcal{M}_0\mathcal{M}_1\dots$ generated by the union of generating sets of all \mathcal{M}_t in \mathcal{M} satisfies the following: letting $\mathcal{G} = \langle i \rangle \mathcal{M}_0\mathcal{M}_1\dots$ and \mathcal{S} be a stabilizer group such that $\langle i \rangle \mathcal{S} = Z(\mathcal{G})$, there exist fixed Paulis $x_1, z_1, \dots, x_k, z_k$ such that $\langle x_1\mathcal{G}, z_1\mathcal{G}, \dots, x_k\mathcal{G}, z_k\mathcal{G} \rangle$ is a presentation for $N(\mathcal{S})/\mathcal{G} \cong \mathcal{P}_k^\circ$ and, for all $t \geq T$, $\langle i\mathcal{S}_t, x_1\mathcal{S}_t, z_1\mathcal{S}_t, \dots, x_k\mathcal{S}_t, z_k\mathcal{S}_t \rangle$ is a presentation for $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$.

Unlike a stabilizer code, after establishment at time T , the ISG \mathcal{S}_t of a subsystem code may change from one timestep to another (while always having the same rank). However, such a code still has the property that there exist fixed Paulis that can represent $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$ for all $t \geq T$. This is what is meant when stabilizer and subsystem codes are labelled *static*.

Our stabilizer code definition above agrees exactly with the usual one; though our definition requires a measurement schedule $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$ to be specified, the fact that $\mathcal{M}_0\mathcal{M}_1\dots$ is Abelian makes this irrelevant. Our subsystem code definition, however, slightly deviates from the usual one; here the fact that a measurement schedule is required is very relevant. We go into more detail on this in [Appendix D](#).

Definition 2.4. A *Floquet code* is an ISG code with a finite period.

We do not attribute so much importance to whether or not an ISG code has finite period, and hence whether it’s labelled a Floquet code or not. Indeed, there are ISG codes that don’t fall into any of the three categories above - examples include the *dynamic tree codes* of Ref. [DTB22]. More interesting to us is the fact that for a general ISG code that establishes at time T , there need not exist fixed Paulis

Code type	\mathcal{S}_t	$N(\mathcal{S}_t)/\mathcal{S}_t$	Period
Stabilizer	Static	Static	Finite or infinite
Subsystem	Dynamic	Static	Finite or infinite
Floquet	Dynamic	Dynamic	Finite
ISG	Dynamic	Dynamic	Finite or infinite

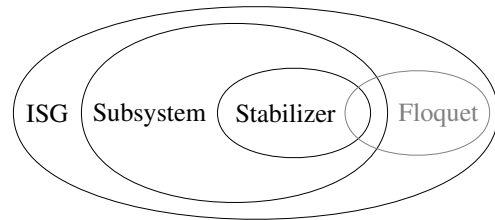


Figure 1: Table and Venn diagram showing relationships between stabilizer, subsystem, Floquet and ISG codes. We attribute little importance to whether an ISG code has finite period, hence this column in the table is drawn in grey. If this column is ignored, there is no difference between a Floquet code and a general ISG code, hence the former’s row in the table, as well as its bubble in the Venn diagram, are also drawn in grey. The Venn diagram shows that all stabilizer codes are subsystem codes, and all subsystem codes are ISG codes. Though all Floquet codes are ISG codes, only stabilizer and subsystem codes with finite period are Floquet codes.

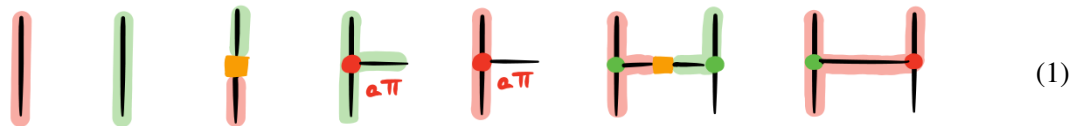
$x_1, z_1, \dots, x_k, z_k \in \mathcal{P}_n$ such that $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$ is a presentation for $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$ for all $t \geq T$. This is what is meant when such codes are labelled *dynamic*.

In [Figure 1](#), we show a table and a Venn diagram characterising the relationships between these code types, and in [Appendix B](#) we give a simple example of an ISG code and its evolution. When working with ISG codes, calculating the effect on $N(\mathcal{S}_t)/\mathcal{S}_t$ of measuring a Pauli $p \in \mathcal{P}_n$ is paramount. To this end, a vital tool is a corollary of the stabilizer formalism which we informally call the *normalizer formalism*; we state and prove it in [Appendix A](#).

2.2 Pauli webs

Though we’ve now defined ISG codes, we haven’t said how to actually detect errors on them, nor how to perform logical (Pauli) operations. Both of these can be viewed elegantly in the ZX-calculus via *Pauli webs*, as defined in Ref. [\[BLN⁺23\]](#). This is analogous to *firing spiders* in Ref. [\[Bor19\]](#), and is generalised to *stabilizer flow* in Ref. [\[MBG23\]](#). Here we’ll only introduce it in a limited and informal way, since this is all we’ll need for [Sections 3](#) and [4](#). For a more rigorous discussion, see Ref. [\[BLN⁺23\]](#) or Ref. [\[MBG23\]](#).

Given a *Clifford ZX-diagram* (one in which all spider phases are integer multiples of $\frac{\pi}{2}$), we’ll define an (*unsigned CSS*) *Pauli web* to be a highlighting of wires green or red (corresponding to Z and X , respectively), according to certain rules. Essentially, the green highlighted edges correspond to how a Z gate can propagate through the diagram, and likewise for red edges and the X gate. Specifically, a highlighted wire can only end at a *Pauli spider* (one whose phase is an integer multiple of π), a Hadamard box², or an input or output node of the overall diagram; a green Pauli spider must have an even number of legs highlighted green (and likewise for red Pauli spiders and red edges); a green Pauli spider must have no legs or every leg highlighted red (and likewise for red Pauli spiders and green edges); and if one leg of a Hadamard box is highlighted green, the other must be red. Examples of Pauli webs on small ZX-diagrams are shown below. Throughout this paper, ZX-diagrams should be read bottom-to-top:



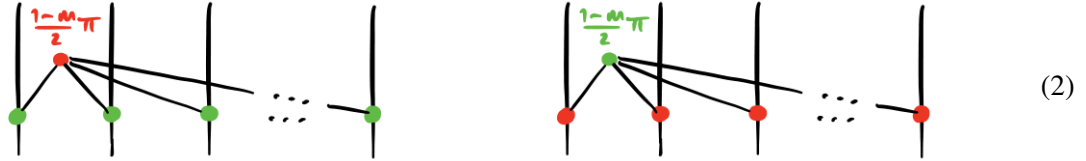
2.2.1 Detectors

A detector is a set of measurement outcomes $m_j \in \{-1, 1\}$ whose product is deterministic in the absence of noise [\[Gid21, HG23\]](#). We can write them as *formal products*³ with powers taken modulo 2. For example, given a qubit in state $|0\rangle$, a Z -basis measurement should deterministically give outcome $m = 1$. Thus the formal product m is a detector. On the other hand, if the qubit is in state $|+\rangle$, a Z -basis measurement’s outcome m_1 is completely random. But a second Z -basis measurement should give outcome m_2 identical to m_1 ; that is, $m_1 m_2$ should be 1. So the formal product $m_1 m_2$ is a detector. Specifically, it detects

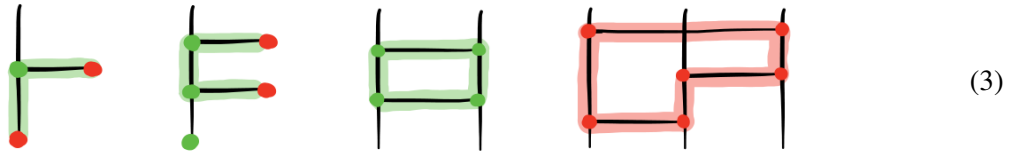
²Under the hood, a Hadamard box is actually a composition of three spiders with phases $\pm \frac{\pi}{2}$. The fact that an unsigned CSS Pauli web can end here might seem to contradict the fact we just said it can only end at Pauli spiders. However, this is just a consequence of the limited way in which we’ve imported Pauli webs here. More general (unsigned) Pauli webs can terminate at any *Clifford spider* - one with phase $k \frac{\pi}{2}$, for $k \in \mathbb{Z}$.

³By *formal product*, we mean we forget that symbols like m_j are actually stand-ins for values like -1 and 1 , and treat the symbols just as objects to be moved around algebraically. By taking powers modulo 2, we mean - for example - the formal product m^3 is the same as $m^1 = m$.

Pauli X errors - if one occurs between the first and second measurement, we'll get $m_2 = -m_1$ and hence $m_1 m_2 = -1$. We say in this case that the detector $m_1 m_2$ is *violated*. In an ISG code context, detectors occur whenever we measure a Pauli p such that p or $-p$ is in the ISG \mathcal{S}_t (*Case 2* in Appendix A). Following [MBG23], we'll define an (*unsigned CSS*) *detecting region* to be a Pauli web with the additional constraint that no input or output nodes of the overall diagram are incident to highlighted edges. Now, recall that in the ZX-calculus, $Z \otimes Z \otimes \dots \otimes Z$ and $X \otimes X \otimes \dots \otimes X$ measurements with outcome m can be represented respectively as:

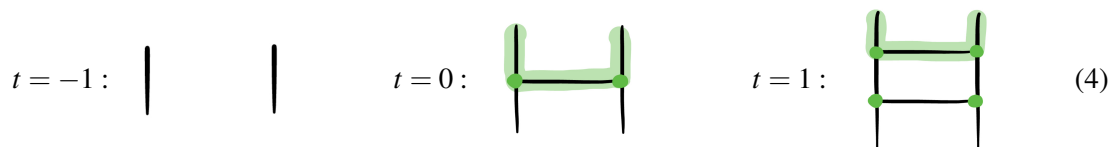


In particular, note that the measurement outcome m parametrises a spider phase. A detecting region then corresponds to a detector $m_1 \dots m_d$ as follows: the measurement outcomes m_j in the detector are all those that parametrise a red spider incident to a green highlighted edge, or a green spider incident to a red highlighted edge. In fact, throughout this paper we will always be able to post-select; that is, we can assume all measurement outcomes m_j are 1. See Appendix G for a longer discussion of this. Below are some simple detecting regions; in each diagram, horizontal wires correspond to measurements (or rather, post-selections; we can thus omit the spider phases that correspond to the measurement outcomes). The resulting detectors consist of exactly the outcomes of the measurements represented by these horizontal wires.

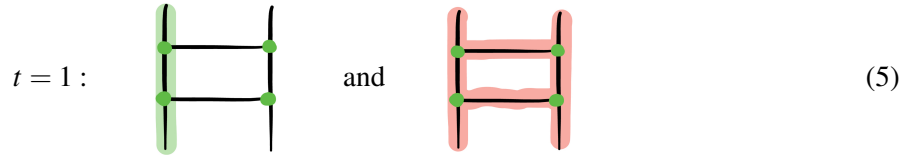


2.2.2 Stabilizers and logical operators

Given an ISG code, the stabilizers (elements of \mathcal{S}_t) and logical operators (members of cosets of $N(\mathcal{S}_t)/\mathcal{S}_t$) can also be seen via Pauli webs. In analogy with a detecting region, we can define an (*unsigned CSS*) *stabilizing region* on a ZX-diagram to be a Pauli web in which none of the diagram's input nodes are incident to a highlighted edge, but at least one output node is. Supposing the ZX-diagram has n output wires, the stabilizer corresponding to such a stabilizing region is (up to ± 1 sign) the Pauli $p = \sigma_{j_1} \otimes \dots \otimes \sigma_{j_n} \in \mathcal{P}_n$, where σ_{j_ℓ} is I if output wire ℓ isn't highlighted, Z if it's highlighted green, and X if it's highlighted red. For an ISG code with measurement schedule $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$, we can draw a ZX-diagram that corresponds to measuring a generating set for \mathcal{M}_0 , then \mathcal{M}_1 , and so on. If we do this up to \mathcal{M}_t , the non-trivial elements of \mathcal{S}_t are exactly the stabilizers derived from the stabilizing regions for this diagram. Below we show this for timesteps $t \in \{-1, 0, 1\}$ of the distance-two repetition code. This is a $[[2, 1, 1]]$ stabilizer code defined by $\mathcal{M} = [\langle Z_1 Z_2 \rangle]$. Its ISG \mathcal{S}_t is thus trivial for $t < 0$ and $\langle m Z_1 Z_2 \rangle$ for $t \geq 0$, for some measurement outcome m .



Logical operators get a very similar treatment. We can define an (*unsigned CSS*) *operating region* to be a Pauli web in which at least one input *and* output node of the diagram are incident to highlighted edges. If the diagram has n output legs, the corresponding operator $p \in \mathcal{P}_n$ is again found by looking at the output wires, in exactly the same way as for a stabilizer above. Given any ISG code, if we again draw a ZX-diagram that corresponds to sequentially measuring generating sets for \mathcal{M}_0 up to \mathcal{M}_t , then representatives of non-trivial elements of $N(\mathcal{S}_t)/\mathcal{S}_t$ are exactly the operators derived from the operating regions for this diagram. The distance-two repetition code has $N(\mathcal{S}_t)/\mathcal{S}_t = \langle \bar{i}, \bar{Z}_1, \bar{X}_1, \bar{Z}_2, \bar{X}_2 \rangle \cong \mathcal{P}_2$ for $t < 0$, and $N(\mathcal{S}_t)/\mathcal{S}_t = \langle \bar{i}, \bar{Z}_1, \bar{X}_1 \bar{X}_2 \rangle \cong \mathcal{P}_1$ for $t \geq 0$. Below, we show operating regions at time $t = 1$:



We close this preliminary section with the comment that detectors and logical operators together provide an alternative view of an ISG code. That is, one can think of such a code not as a group-theoretic object, but as a Clifford ZX-diagram that is suitably covered by detecting regions, and contains a non-empty set of pairs of operating regions whose corresponding operators satisfy the Pauli commutativity relations. This corresponds to the unifying view of fault-tolerance put forward recently in Ref. [BLN⁺23], and is in the same spirit as the *spacetime codes* of Ref. [DP23].

3 Floquetifying the $[[4, 2, 2]]$ code

Let’s warm up with one of the simplest interesting codes around: the $[[4, 2, 2]]$ code. This is a stabilizer code, which we’ll define as $\mathcal{M} = [\langle Z_1 Z_2 Z_3 Z_4 \rangle, \langle X_1 X_2 X_3 X_4 \rangle]$. The aim of this section is to prove the following:

Theorem 3.1. *The $[[4, 2, 2]]$ stabilizer code is equivalent as a ZX-diagram to a $[[12, 2, 2]]$ Floquet code with period 6, which we call the **double hexagon code**.*

In **Figure 2**, we show three equivalent ZX-diagrams depicting seven timesteps of this code. Here the grey squares, grey numbers, wire colours and wire styles (solid/dashed) have no meaning in the ZX-calculus; they’re just visual aids. The grey squares denote timesteps of the $[[4, 2, 2]]$ code, and the grey numbers and coloured/styled lines will be explained shortly. The leftmost diagram is the most natural one; it shows measurements of $Z_1 Z_2 Z_3 Z_4$ and $X_1 X_2 X_3 X_4$ alternating at each timestep. The second diagram is obtained from the first by unfixing every spider in the center of a grey square into two spiders, and unfixing every spider in the corner of a grey square into three spiders. The third is identical to the second in the ZX-calculus - all we’ve done is coloured and styled certain wires, and labelled all one-legged spiders and black wires with an integer. Now, in the leftmost diagram, we interpret the four vertical lines as the world-lines of the four qubits of the code. But we need not do this! The ZX-diagram remains equivalent if we choose to interpret different wires as qubit world-lines. This is exactly what the colours and styles in the rightmost diagram are for; each colour-style pair denotes a different qubit world-line. Since there are twelve world-lines, we’re now viewing this as a system of twelve qubits, rather than four.

Let’s make some observations about this rightmost diagram. Firstly, if we follow the world-line of any particular qubit up the page, the integer labels incident to it form an increasing sequence. For example,

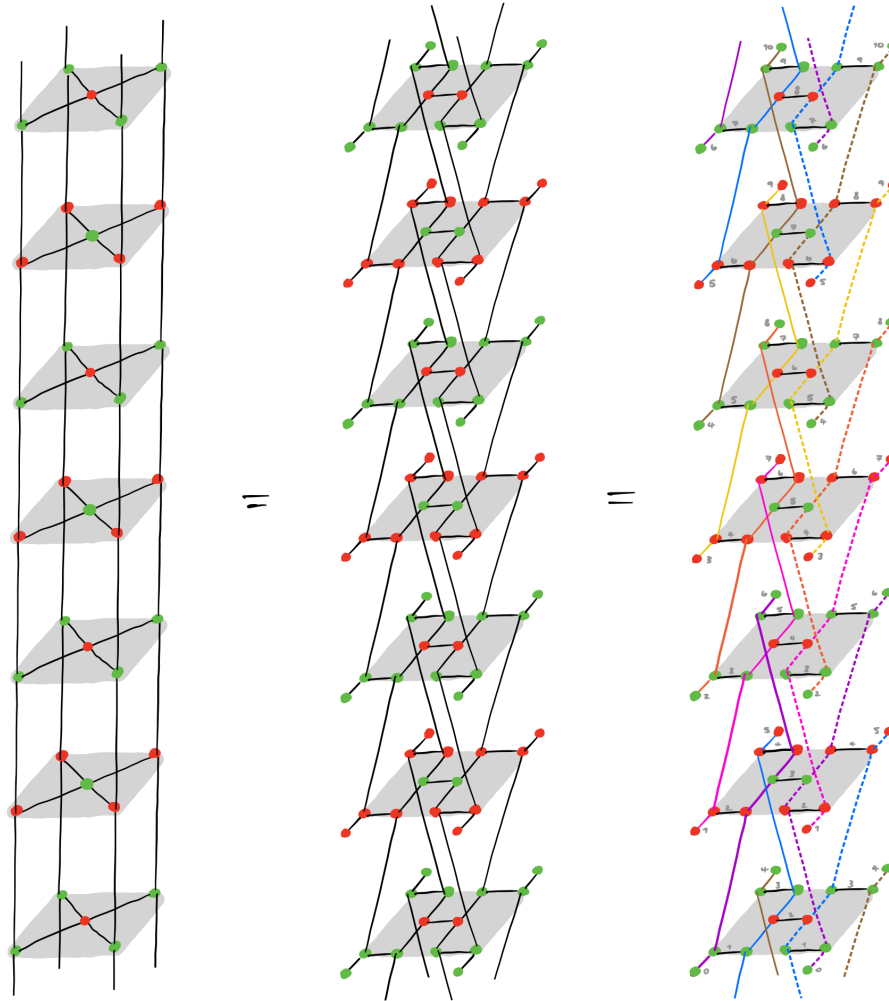


Figure 2: Three equivalent ZX-diagrams for seven timesteps of the $[[4, 2, 2]]$ code.

starting from the bottom of the diagram and following the solid purple qubit upwards, the integers incident to it form the sequence $[0, 1, \dots, 6]$. We can thus think of these integers as a new set of timesteps for this diagram. Next, notice that we can interpret all the uncoloured black wires like $\text{---} \text{---}$ and $\text{---} \text{---}$ as weight two $Z_u Z_v$ and $X_u X_v$ measurements (respectively) between qubits. Furthermore, the qubit world-lines only have limited interactions with each other via these measurements. Specifically, if we define ordered lists **colours** = [purple, pink, orange, yellow, brown, blue] and **styles** = [solid, dashed], and let qubit (i, j) denote the qubit with the i -th colour and j -th style, where i and j are taken modulo 6 and 2 respectively, then looking closely we see that qubit (i, j) is only ever involved in a measurement with the three qubits $(i + 1, j)$, $(i - 1, j)$ and $(i, j + 1)$. So supposing we now wanted to lay out these qubits on a planar 2D chip, a natural geometry would be a ‘double hexagon’, as in the rightmost diagram of Figure 3.

In fact, recalling the diagrammatic equation $\text{---} \text{---} = \text{---} \text{---}$, which says non-destructive single qubit Pauli measurements disconnect wires, we can also interpret all one-legged spiders as one half of such a measurement. This interpretation is valid, in that the timestep at which the world-line of qubit (i, j) ‘ends’

at a one-legged spider is the same as the timestep at which it ‘resumes’ via another one-legged spider further up the page. Finally, we can see that this pattern of colours and styles repeats itself every six timesteps.

From this analysis, we can now interpret the rightmost ZX-diagram as an ISG code; we can write down the measurements that each qubit undergoes at each timestep, and consequently we can define the measurement schedule described by this diagram. We get $\mathcal{M} = [\mathcal{M}_0, \dots, \mathcal{M}_5]$, where:

$$\mathcal{M}_t = \left\langle \begin{array}{l} X_{(t,0)}, \\ X_{(t,1)}, \\ X_{(t+1,0)}X_{(t+2,0)}, \\ X_{(t+1,1)}X_{(t+2,1)}, \\ X_{(t+3,0)}X_{(t+3,1)}, \\ X_{(t+4,0)}X_{(t+5,0)}, \\ X_{(t+4,1)}X_{(t+5,1)} \end{array} \right\rangle \text{ if } t \text{ even}, \quad \mathcal{M}_t = \left\langle \begin{array}{l} Z_{(t,0)}, \\ Z_{(t,1)}, \\ Z_{(t+1,0)}Z_{(t+2,0)}, \\ Z_{(t+1,1)}Z_{(t+2,1)}, \\ Z_{(t+3,0)}Z_{(t+3,1)}, \\ Z_{(t+4,0)}Z_{(t+5,0)}, \\ Z_{(t+4,1)}Z_{(t+5,1)} \end{array} \right\rangle \text{ if } t \text{ odd}. \quad (6)$$

One can then calculate the group \mathcal{S}_t , and consequently $N(\mathcal{S}_t)/\mathcal{S}_t$. It turns out \mathcal{S}_t is established whenever $t \geq T = 3$, and can be minimally generated by the seven generators of \mathcal{M}_t , plus three weight-six Paulis s_{t-1}, s_{t-2} and s_{t-3} , where:

$$s_t = \begin{cases} X_{(t,0)}X_{(t,1)}X_{(t-1,0)}X_{(t-1,1)}X_{(t-2,0)}X_{(t-2,1)} & \text{if } t \text{ even} \\ Z_{(t,0)}Z_{(t,1)}Z_{(t-1,0)}Z_{(t-1,1)}Z_{(t-2,0)}Z_{(t-2,1)} & \text{if } t \text{ odd} \end{cases} \quad (7)$$

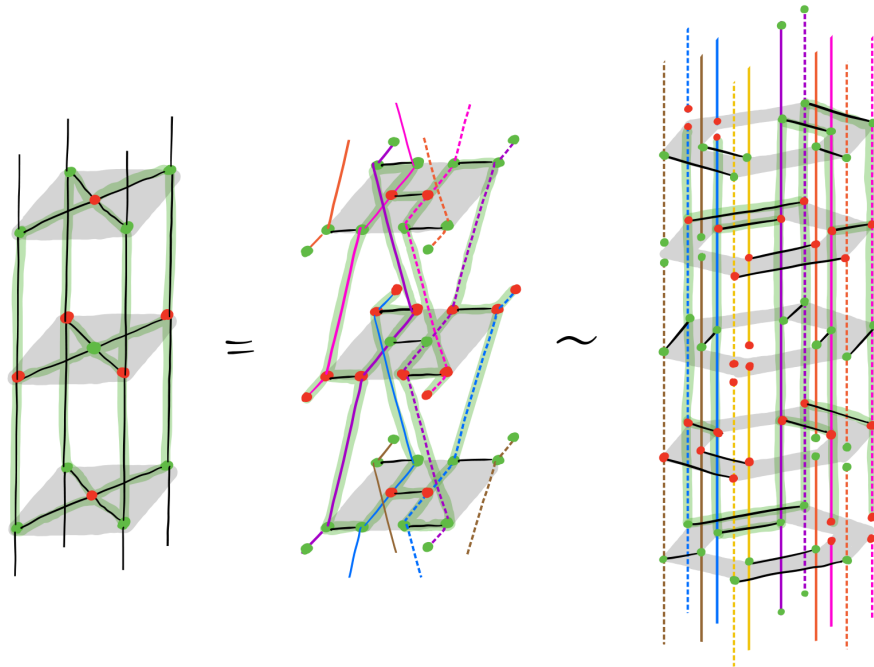


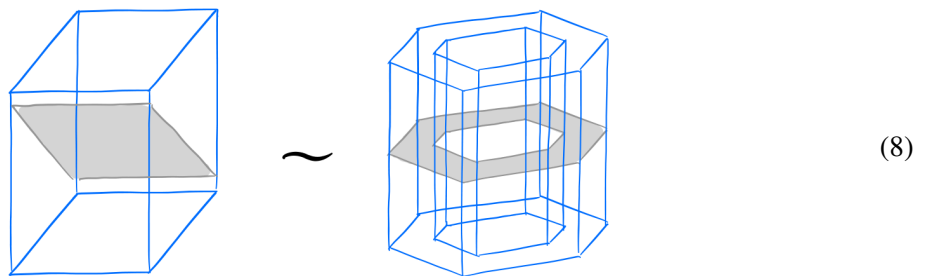
Figure 3: A detecting region in the $[[4, 2, 2]]$ code and its image in the double hexagon code. We use the (\sim) symbol between the second and third diagrams rather than an equals sign because, although the two codes can be viewed as having equivalent ZX-diagrams, these two particular subdiagrams are not equivalent - the rightmost one has more input and output wires, for example.

Since we have 10 independent generators on 12 qubits, we can conclude that $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_2$ for all $t \geq 3$. This then proves most of [Theorem 3.1](#); namely that the double hexagon code encodes 2 logical qubits and has period 6. The proof that the distance of the new Floquet code remains two is deferred to [Appendix E](#).

In addition to being able to work with detectors, stabilizers and logical operators algebraically, as above, the mapping of these objects from the $[[4, 2, 2]]$ code to the double hexagon code can be seen graphically via Pauli webs. In [Figure 3](#) we show a detecting region in the $[[4, 2, 2]]$ code and its image in the double hexagon code. From the rightmost diagram of this figure, and recalling the rules for mapping a detecting region to a detector from [Subsection 2.2.1](#), one can see that the corresponding detector in the double hexagon code consists of eight measurements. Specifically, in the bottom layer, we include two $Z \otimes Z$ measurements between blue and purple qubits, and two single qubit Z measurements on pink qubits. In the top layer, we include two $Z \otimes Z$ measurements between purple and pink qubits, and two single qubit Z measurements on blue qubits. Since every detecting region in the $[[4, 2, 2]]$ code is equivalent to the one on the left of this figure (up to a space-time translation and exchanging the roles of Z and X), every detecting region in the double hexagon code is equivalent to the one on the right (again up to a space-time translation and $Z \leftrightarrow X$ interchange).

One could justifiably point out here that we seem to have made things worse; we've taken a $[[4, 2, 2]]$ ISG code and turned it into a $[[12, 2, 2]]$ ISG code, and what's more, each detector now consists of eight measurements rather than two, so would seem to be noisier! The trade-off is that now every measurement is weight-two or weight-one, rather than weight-four. As a general rule, the higher the measurement weight, the noisier it will be. In particular, weight-one and weight-two measurements can be performed natively in some architectures, whereas higher-weight measurements are implemented via extraction circuits, which give more opportunities for noise to interfere.

On a higher-level, one can view this Floquetification process as a reinterpretation of the time direction in a ZX-diagram. Below, we use two blue prisms (square and hexagonal) as abstractions of ZX-diagrams for the $[[4, 2, 2]]$ code and double hexagon code respectively. In grey we show how a timeslice in the double hexagon code corresponds to an angled slice of the $[[4, 2, 2]]$ code:



4 Floquetifying the colour code

We can apply the same ideas as in the last section to stabilizer codes more interesting than the $[[4, 2, 2]]$ code. In this section, we take this more interesting stabilizer code to be the colour code. Or, more accurately, we take it to be the *bulk* of the colour code - i.e. ignoring the code's global topology (whether it lives on a torus, or is planar). A discussion of global topology is deferred to [Subsection 4.2](#) at the end of this section, and continued in detail in [Appendix F](#).

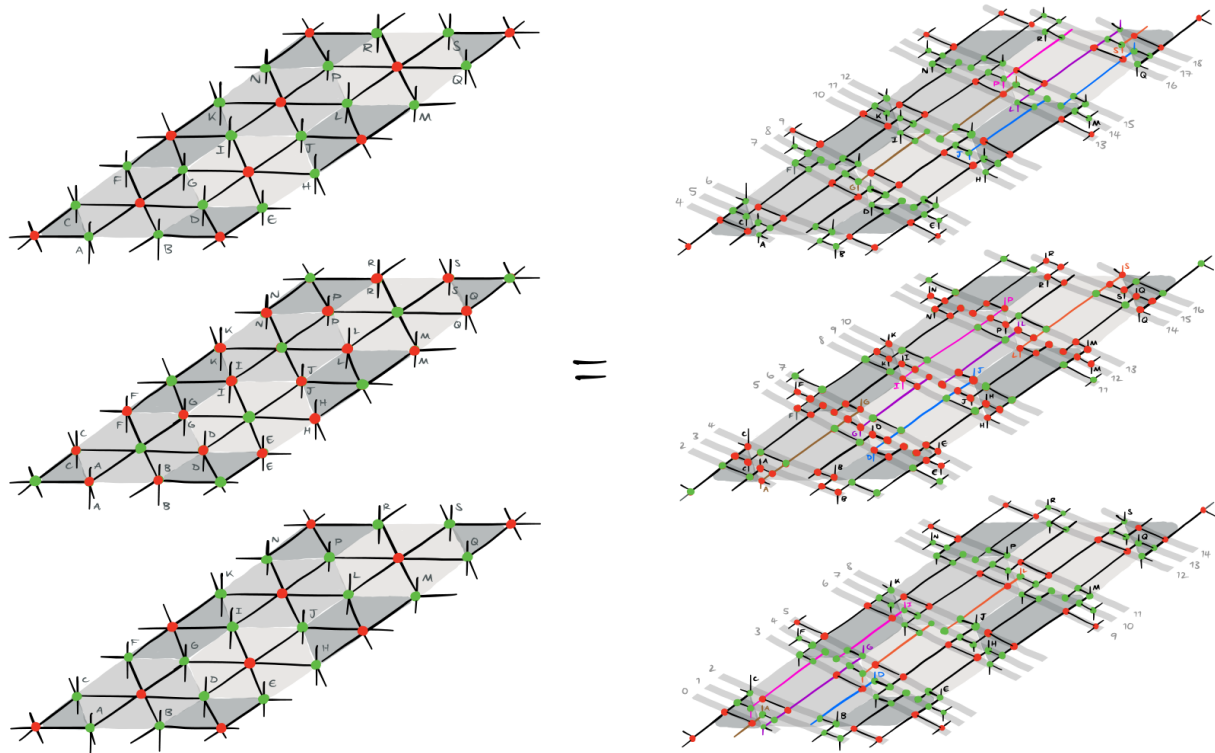


Figure 4: Two equivalent ZX-diagrams for a patch of the colour code at three consecutive timesteps. Ideally we’d draw vertical wires connecting each column of patches, like we were able to do for the $[[4, 2, 2]]$ code in the last section, but doing this renders the diagrams pretty much unreadable. Instead, we draw very small vertical wires going up and/or down from certain spiders, labelled by letters. These letters define how a wire going vertically upwards in one layer is actually connected to a wire coming vertically downwards from the layer above it; wires labelled by the same letter are connected.

4.1 The bulk

The 6.6.6 *colour code* is a stabilizer code defined on a honeycomb lattice, with qubits placed at vertices. We write $v \in f$ to mean that a vertex v is incident to a hexagonal face f . For any such face f , we define weight-six Paulis $X_f = \prod_{v \in f} X_v$ and $Z_f = \prod_{v \in f} Z_v$. On a torus, the code is defined by the measurement schedule $\mathcal{M} = [\langle \{Z_f : \text{face } f\} \rangle, \langle \{X_f : \text{face } f\} \rangle]$. On a planar geometry, slightly different Paulis are measured at the boundaries [KPEB18].

As before, we start with a ZX-diagram of the colour code over multiple timesteps; see the left hand side of Figure 4. We then unfuse every spider into four or five spiders to get the diagram on the right of the figure. In these diagrams, grey hexagons, bars and integers, as well as letter labels and coloured wires, all have no meaning in the ZX-calculus; they’re just visual aids. In the diagram on the right, wires within grey bars correspond to weight-two measurements, while all other wires correspond to qubit world-lines. The focus is on a single qubit’s world-line, which we’ve coloured purple; we can see that it’s only involved in measurements with four other qubits, which we’ve coloured orange, pink, blue and brown. It turns out that all qubit world-lines have this property of only interacting with four other qubits. Furthermore, these interactions are such that the qubits of the new code can be laid out on a square lattice. So henceforth

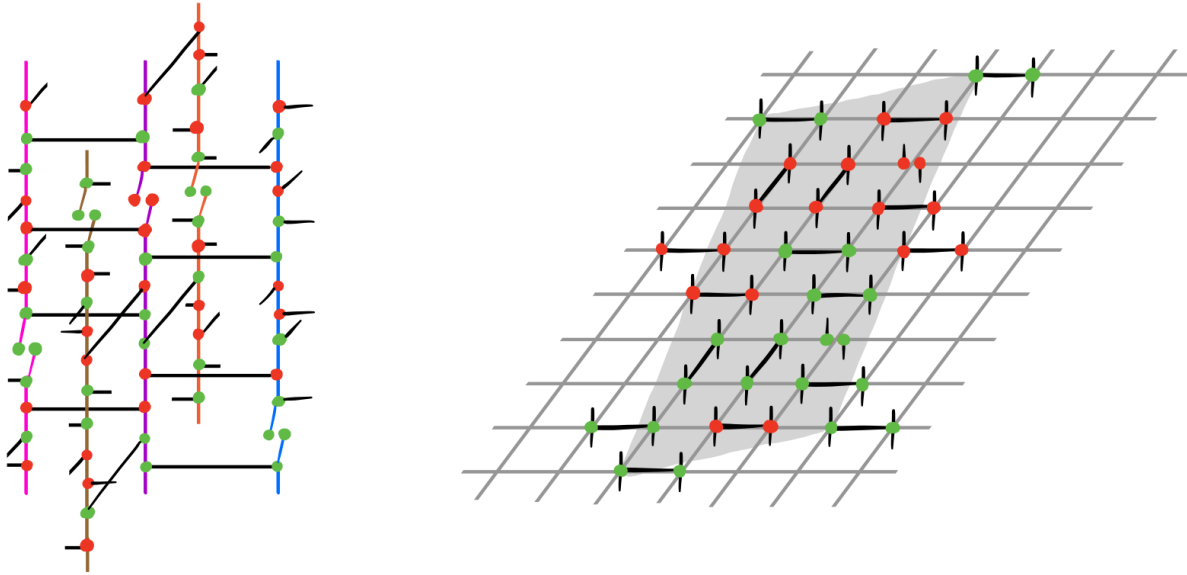


Figure 5: Left: measurements undergone by a single qubit and its four nearest neighbours over one full period of 13 timesteps. Right: One ‘tile’ of one timestep of the measurement schedule in the bulk of the Floquetified colour code. Here the grey lines are *not* ZX-diagram wires - they are just visual aids that show this all sits on a square lattice.

we’ll label qubits of the new code with a pair of integer coordinates (x, y) .

The grey bars labelled by integers denote the timesteps of the new code. These are well-defined; picking any qubit world-line and following it up the page while noting down the integer label of every grey bar it crosses produces an increasing sequence. For example, doing this for the purple qubit produces the contiguous sequence $[0, 1, 2, \dots]$. As before, we can view all one-legged spiders as halves of single-qubit measurements. The pattern repeats after every two timesteps of the old code (the colour code); one can see this by noting that the bottom and top of the diagram are identical, up to a translation in space and an increase by 13 in the labels of the grey bars. In other words, this new code has a period of 13.

Again, we can now write down the measurements that each qubit undergoes at each timestep. In [Figure 5](#), we show a ZX-diagram of this, focused on the purple qubit from [Figure 4](#). Each qubit undergoes essentially the same pattern of measurements in each period. Specifically, whatever measurement qubit (x, y) undergoes at timestep t , qubit $(x, y + 1)$ undergoes it at time $t - 2$, but with the roles of Z and X exchanged. Likewise for the remaining neighbours $(x + 1, y)$, $(x, y - 1)$ and $(x - 1, y)$, but at times $t - 8$, $t + 2$ and $t + 8$ respectively. Knowing this, we can then write down the measurement schedule for the bulk of the new code (i.e. what measurements are happening at any single timestep). This has a periodic structure; we draw a ZX-diagram for a single timestep t and single ‘tile’ of this on the right of [Figure 5](#). To see the measurements happening across the whole bulk at this timestep, one should tile these grey rectangles across the square lattice. That is, whatever measurement qubit (x, y) undergoes at time t , qubits $(x + 3, y + 1)$ and $(x - 2, y + 8)$ undergo this too. Then to get the measurements happening at the next timestep, one should translate the measurements from time t by $(-1, -3)$. That is, whatever measurement qubit (x, y) undergoes at time t , qubit $(x - 1, y - 3)$ undergoes it at time $t + 1$.

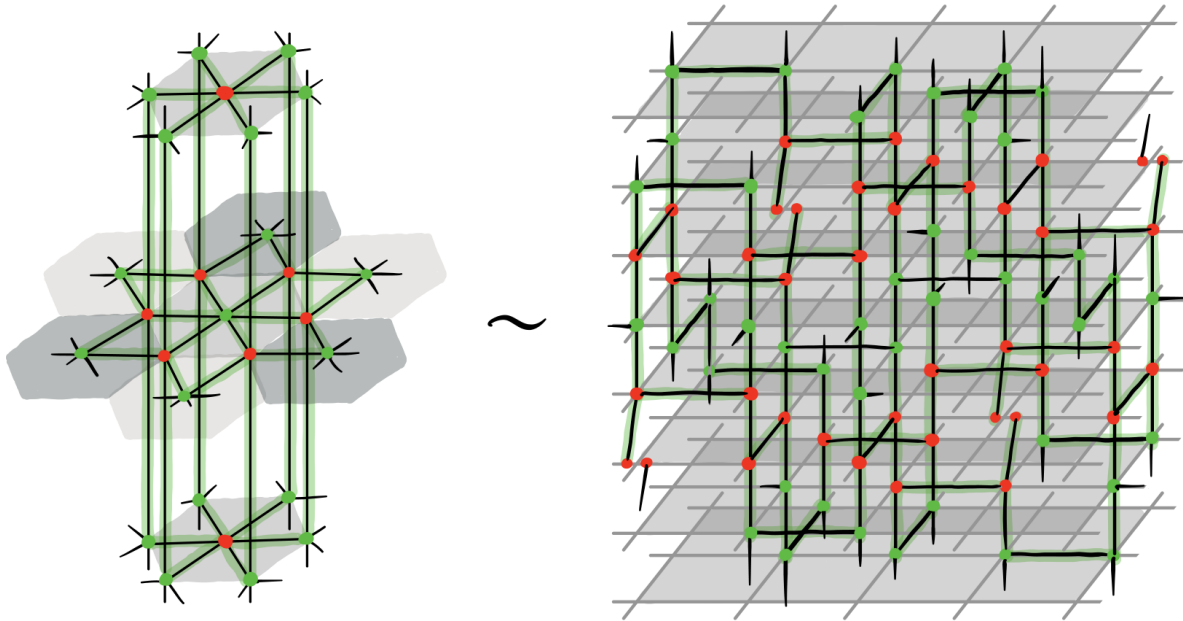


Figure 6: A detecting region in the colour code and the corresponding detecting region in the Floquetification. Again, grey lines are *not* ZX-diagram wires - they are just a visual guide showing the square lattice.

Detectors and logical operators can again not only be worked with algebraically, but also graphically, via Pauli webs. In [Figure 6](#) we show a detecting region in the colour code bulk and its image in the bulk of the new Floquet code. The corresponding detector in the new code consists of 14 weight-two measurements and 4 single-qubit measurements, spread over 8 timesteps. Every detector in the bulk of the new code is identical to this one, up to a space-time translation and exchanging Z and X . Just as in the colour code, these detecting regions are tiled such that unique errors violate unique sets of detectors, so decoding can be performed, though we leave investigating specific decoding strategies to future work. A similar exercise can be repeated for the logical operators; one can draw the operating regions corresponding to the known logical operators of the colour code, and see how these map to operating regions in the Floquetified code, from which one can write down the new code's logical operators.

4.2 Beyond the bulk

The Floquetification process described above can be applied directly to planar colour codes, and will lead to a new code that is itself planar. But since the boundaries of the original code look different to the bulk, extra work needs to be done to Floquetify these correctly. Furthermore, the resulting Floquet code can exhibit a 'drifting' behaviour, which we discuss in more detail in [Appendix F.1](#). There are potential perks of this behaviour - e.g. for removing leakage - but it's also handy to have a code that doesn't drift. One might think we could get around these two issues by starting with the colour code defined on a torus, which has no boundaries to worry about. But viewing our procedure as tilting the time direction in a ZX-diagram for a code, as described in the last section, we find we can only give well-defined new timesteps when the code we start with is planar - this is discussed in more depth in [Appendix F.2](#). Instead, if we want to avoid boundaries, what we can do is Floquetify only the bulk, which will give the bulk of

a potential new Floquet code, then see if placing this new bulk on a torus still encodes logical qubits. In [Appendix F.3](#) we apply the two ideas described above - we Floquetify a planar colour code, and place the Floquetified colour code bulk from this section on a torus.

5 Conclusion and future work

In this work, we introduced ISG codes, which describe a large family of codes driven by sequentially measuring sets of Pauli operators; this includes stabilizer, subsystem⁴ and Floquet codes, and more. We then used the ZX-calculus to find a new ISG code (specifically, a Floquet code) that is equivalent to the colour code, and can be implemented on a square lattice. The main disadvantage of the colour code versus the surface code is its high-weight measurements - our construction removes this obstacle, since all its measurements are of weight one or two. For it to be a genuine candidate for practical implementation, we would need to investigate its decoding capabilities, its boundaries and its logical gates - we leave these for future work.

One direction we find very interesting relates to the latter; in Ref. [\[AWH22\]](#), it is shown that Floquet codes can natively implement certain logical Clifford gates fault-tolerantly at no extra effort. The set of such implementable gates is restricted by the automorphisms of the code's anyonic defects; we believe our Floquetified colour code should inherit a rich set of such automorphisms from the colour code. We would like to verify whether this is the case, and then see whether this can be leveraged to perform fault-tolerant logical gates.

As it happens, the authors of an upcoming paper [\[DTB⁺23\]](#) do exactly this for an independently discovered Floquet code that is also in a definable sense equivalent to the colour code. Specifically, using the *anyon condensation* framework of Ref. [\[KFT⁺22\]](#), they construct a code that can implement the full logical Clifford group via sequences of measurements of weight at most three. What's more, a further upcoming paper [\[DSTE23\]](#) contains a third independent 'Floquetified colour code' construction, this time by starting with a subsystem code (specifically, that of Ref. [\[Bom10\]](#)) and passing to an *associated ISG code* ([Definition D.1](#)). We are excited to learn more about both of these works, and to think about the connections between our various different constructions. More generally, understanding our own Floquetified colour code in terms of topological phases and symmetries of the ZX-diagram representing it seems an exciting research avenue that bridges between the fields of diagrammatic calculi, (topological) error correction and condensed matter physics.

Further work is warranted around ISG codes more broadly. For example, there are interesting questions to be answered around equivalences of such codes, how best to define a notion of distance on them, and how subsystem codes as they're usually defined fit into this framework. Though we go into more detail on these in [Appendices C and D](#), a more thorough investigation would be welcome. This could perhaps lead to interesting re-evaluations of long-established quantum codes.

The other obvious further research direction would be to develop the ideas here into a full-blown general-purpose Floquetification algorithm, which can take as input a stabilizer code (presumably satisfying certain conditions), and return a Floquetified version of it, with all the advantages and trade-offs that brings; e.g. lower-weight measurements, but more measurements per detector, typically. One could then

⁴Up to the caveat that our definition of a subsystem code as in [Definition 2.3](#) differs slightly from the usual subsystem code definition, a point which we explore in [Appendix D](#).

compare the performance of these stabilizer codes with their Floquetifications, to see if any practical advantages emerge in the general case.

On a higher level, this work makes the same point as Ref. [BLN⁺23], in that it suggests *static* stabilizer and subsystem codes are perhaps not so different from *dynamic* ISG codes (like Floquet codes) after all, and that a unifying way to think of ISG codes could be as ZX-diagrams that are suitably covered by detecting regions and contain pairs of operating regions satisfying Pauli commutativity relations. An avenue for further work would be to develop this perspective further - for example, by taking it as a starting point for designing new codes.

6 Acknowledgements

We wish to thank Daniel Litinski and Fernando Pastawski above all, for chatting with us about applications of the ZX-calculus to quantum error correction [BLN⁺23], and to David Aasen for discussions about his automorphism codes [AWH22]. We also thank the reviewers from QPL 2023, who gave very helpful feedback on our initial submission, as well as Jens Eisert, Peter-Jan Derks and Daniel Litinski, who gave helpful feedback on the final draft. We are grateful to the authors of Ref. [DTB⁺23] and Ref. [DSTE23] for discussing their own Floquetified colour code constructions with us. Alex wishes to thank Drew Vandeth, Anamaria Rojas and all those involved in organising the IBM QEC Summer School for the inspiring four weeks that sowed the seeds of this project. Julio wants to thank Tyler Ellison for inspiring discussion on Floquet codes in the colour code phase. This work was supported by the Einstein Foundation (Einstein Research Unit on quantum devices), the DFG (CRC 183), the Munich Quantum Valley (K8) and the BMBF (RealistiQ, QSolid).

References

- [AWH22] David Aasen, Zhenghan Wang & Matthew B. Hastings (2022): *Adiabatic paths of Hamiltonians, symmetries of topological order, and automorphism codes*. *Physical Review B* 106(8), doi:[10.1103/physrevb.106.085122](https://doi.org/10.1103/physrevb.106.085122).
- [Bac06] Dave Bacon (2006): *Operator quantum error-correcting subsystems for self-correcting quantum memories*. *Physical Review A* 73(1), doi:[10.1103/physreva.73.012340](https://doi.org/10.1103/physreva.73.012340).
- [Bau23] Andreas Bauer (2023): *Topological error correcting processes from fixed-point path integrals*, doi:[10.48550/arXiv.2303.16405](https://doi.org/10.48550/arXiv.2303.16405). arXiv:[2303.16405](https://arxiv.org/abs/2303.16405).
- [BLN⁺23] Hector Bombin, Daniel Litinski, Naomi Nickerson, Fernando Pastawski & Sam Roberts (2023): *Unifying flavors of fault tolerance with the ZX calculus*, doi:[10.48550/ARXIV.2303.08829](https://doi.org/10.48550/ARXIV.2303.08829).
- [BMD06] H. Bombin & M. A. Martin-Delgado (2006): *Topological Quantum Distillation*. *Physical Review Letters* 97(18), doi:[10.1103/physrevlett.97.180501](https://doi.org/10.1103/physrevlett.97.180501).
- [BMD07] H. Bombin & M. A. Martin-Delgado (2007): *Optimal resources for topological two-dimensional stabilizer codes: Comparative study*. *Physical Review A* 76(1), doi:[10.1103/physreva.76.012305](https://doi.org/10.1103/physreva.76.012305).
- [Bom10] H. Bombin (2010): *Topological subsystem codes*. *Physical Review A* 81(3), doi:[10.1103/physreva.81.032301](https://doi.org/10.1103/physreva.81.032301).

- [Bor19] Coen Borghans (2019): *ZX-Calculus and Quantum Stabilizer Theory*. Master's thesis, Radboud University. <https://www.cs.ox.ac.uk/people/aleks.kissinger/papers/borghans-thesis.pdf>.
- [CD11] Bob Coecke & Ross Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:[10.1088/1367-2630/13/4/043016](https://doi.org/10.1088/1367-2630/13/4/043016).
- [CK17] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, doi:[10.1017/9781316219317](https://doi.org/10.1017/9781316219317).
- [CKYZ20] Christopher Chamberland, Aleksander Kubica, Theodore J Yoder & Guanyu Zhu (2020): *Triangular color codes on trivalent graphs with flag qubits*. *New Journal of Physics* 22(2), p. 023019, doi:[10.1088/1367-2630/ab68fd](https://doi.org/10.1088/1367-2630/ab68fd).
- [DP23] Nicolas Delfosse & Adam Paetznick (2023): *Spacetime codes of Clifford circuits*, doi:<https://doi.org/10.48550/arXiv.2304.05943>.
- [DSTE23] Arpit Dua, Joseph Sullivan, Nathanan Tantivasadakarn & Tyler Ellison (2023): *Rewinding floquet codes*. To appear.
- [DTB22] Margarita Davydova, Nathanan Tantivasadakarn & Shankar Balasubramanian (2022): *Floquet codes without parent subsystem codes*, doi:[10.48550/ARXIV.2210.02468](https://doi.org/10.48550/ARXIV.2210.02468).
- [DTB⁺23] Margarita Davydova, Nathanan Tantivasadakarn, Shankar Balasubramanian, & David Aasen (2023): *Quantum computation from dynamic automorphism codes*. To appear.
- [Gid21] Craig Gidney (2021): *Stim: a fast stabilizer circuit simulator*. *Quantum* 5, p. 497, doi:[10.22331/q-2021-07-06-497](https://doi.org/10.22331/q-2021-07-06-497).
- [Gid22] Craig Gidney (2022): *A Pair Measurement Surface Code on Pentagons*, doi:[10.48550/ARXIV.2206.12780](https://doi.org/10.48550/ARXIV.2206.12780).
- [Gid23] Craig Gidney (2023): *A cleaned up version of the axioms I actually use when applying the ZX calculus to quantum error correction*. Twitter. Available at <https://twitter.com/CraigGidney/status/1643848850711662592>. Accessed on 21.06.2023.
- [Got97] Daniel Gottesman (1997): *Stabilizer Codes and Quantum Error Correction*, doi:[10.48550/ARXIV.QUANT-PH/9705052](https://doi.org/10.48550/ARXIV.QUANT-PH/9705052).
- [Got09] Daniel Gottesman (2009): *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation*, doi:[10.48550/ARXIV.0904.2557](https://doi.org/10.48550/ARXIV.0904.2557).
- [HG23] Oscar Higgott & Craig Gidney (2023): *Sparse Blossom: correcting a million errors per core second with minimum-weight matching*, doi:<https://doi.org/10.48550/arXiv.2303.15933>.
- [HH21] Matthew B. Hastings & Jeongwan Haah (2021): *Dynamically Generated Logical Qubits*. *Quantum* 5, p. 564, doi:[10.22331/q-2021-10-19-564](https://doi.org/10.22331/q-2021-10-19-564).
- [KFT⁺22] Markus S. Kesselring, Julio C. Magdalena de la Fuente, Felix Thomsen, Jens Eisert, Stephen D. Bartlett & Benjamin J. Brown (2022): *Anyon condensation and the color code*, doi:[10.48550/ARXIV.2212.00042](https://doi.org/10.48550/ARXIV.2212.00042).

- [KLP05] David Kribs, Raymond Laflamme & David Poulin (2005): *Unified and Generalized Approach to Quantum Error Correction*. *Physical Review Letters* 94(18), doi:[10.1103/physrevlett.94.180501](https://doi.org/10.1103/physrevlett.94.180501).
- [KPEB18] Markus S. Kesselring, Fernando Pastawski, Jens Eisert & Benjamin J. Brown (2018): *The boundaries and twist defects of the color code and their applications to topological quantum computation*. *Quantum* 2, p. 101, doi:[10.22331/q-2018-10-19-101](https://doi.org/10.22331/q-2018-10-19-101).
- [Lit19] Daniel Litinski (2019): *A Game of Surface Codes: Large-Scale Quantum Computing with Lattice Surgery*. *Quantum* 3, p. 128, doi:[10.22331/q-2019-03-05-128](https://doi.org/10.22331/q-2019-03-05-128).
- [LRA14] Andrew J. Landahl & Ciaran Ryan-Anderson (2014): *Quantum computing by color-code lattice surgery*, doi:[10.48550/ARXIV.1407.5103](https://doi.org/10.48550/ARXIV.1407.5103).
- [MBG23] Matt McEwen, Dave Bacon & Craig Gidney (2023): *Relaxing Hardware Requirements for Surface Code Circuits using Time-dynamics*, doi:[10.48550/ARXIV.2302.02192](https://doi.org/10.48550/ARXIV.2302.02192).
- [NC10] Michael A. Nielsen & Isaac L. Chuang (2010): *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, doi:[10.1017/CBO9780511976667](https://doi.org/10.1017/CBO9780511976667).
- [Pou05] David Poulin (2005): *Stabilizer Formalism for Operator Quantum Error Correction*. *Physical Review Letters* 95(23), doi:[10.1103/physrevlett.95.230504](https://doi.org/10.1103/physrevlett.95.230504).
- [SWP23] Joseph Sullivan, Rui Wen & Andrew C. Potter (2023): *Floquet codes and phases in twist-defect networks*, doi:[10.48550/arXiv.2303.17664](https://doi.org/10.48550/arXiv.2303.17664).
- [TKBB22] Felix Thomsen, Markus S. Kesselring, Stephen D. Bartlett & Benjamin J. Brown (2022): *Low-overhead quantum computing with the color code*, doi:[10.48550/ARXIV.2201.07806](https://doi.org/10.48550/ARXIV.2201.07806).
- [VGW96] Lev Vaidman, Lior Goldenberg & Stephen Wiesner (1996): *Error prevention scheme with four particles*. *Physical Review A* 54(3), pp. R1745–R1748, doi:[10.1103/physreva.54.r1745](https://doi.org/10.1103/physreva.54.r1745).
- [Vui21] Christophe Vuillot (2021): *Planar Floquet Codes*, doi:[10.48550/ARXIV.2110.05348](https://doi.org/10.48550/ARXIV.2110.05348).
- [Wet20] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*, doi:[10.48550/ARXIV.2012.13966](https://doi.org/10.48550/ARXIV.2012.13966).
- [Yos15] Beni Yoshida (2015): *Topological color code and symmetry-protected topological phases*. *Physical Review B* 91(24), doi:[10.1103/physrevb.91.245131](https://doi.org/10.1103/physrevb.91.245131).
- [ZAV22] Zhehao Zhang, David Aasen & Sagar Vijay (2022): *The X-Cube Floquet Code*, doi:[10.48550/ARXIV.2211.05784](https://doi.org/10.48550/ARXIV.2211.05784).

A A corollary of the stabilizer formalism

Here we state and prove the effect that measuring a Hermitian Pauli $p \in \mathcal{P}_n$ has on the group $N(\mathcal{S})/\mathcal{S}$, for any stabilizer group $\mathcal{S} \leq \mathcal{P}_n$. We will often refer to this as (measurement in) the *normalizer formalism*. A few lemmas will be required in order to prove this; the first two are so fundamental that we will use them without explicitly referencing them.

Lemma A.1. *Any two elements $p, q \in \mathcal{P}_n$ either commute or anti-commute. That is, $pq = \pm qp$.*

Corollary A.2. *If $\mathcal{S} \leq \mathcal{P}_n$ is a stabilizer group, the normalizer $N(\mathcal{S}) = \{p \in \mathcal{P}_n : p\mathcal{S}p^{-1} = \mathcal{S}\}$ is equal to the centralizer $C(\mathcal{S}) = \{p \in \mathcal{P}_n : \forall s \in \mathcal{S}, psp^{-1} = s\}$.*

The next two are less elementary; their proofs can be found in Ref. [Got09, Section 3.4].

Lemma A.3. *If $\mathcal{S} \leq \mathcal{P}_n$ is a stabilizer group with rank r , then $N(\mathcal{S})/\mathcal{S} \cong \mathcal{P}_k$, where $k = n - r$.*

Lemma A.4. *If $\mathcal{S} \leq \mathcal{P}_n$ is a stabilizer group with rank r , and $k = n - r$, then for any maximally Abelian subgroup $\langle \bar{x}_1, \dots, \bar{x}_k \rangle$ of $N(\mathcal{S})/\mathcal{S} \cong \mathcal{P}_k$ we might choose, there exists a second Abelian subgroup $\langle \bar{z}_1, \dots, \bar{z}_k \rangle$, such that $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$ is isomorphic to $\mathcal{P}_k = \langle i, X_1, Z_1, \dots, X_k, Z_k \rangle$ via the map $\bar{i} \mapsto i, \bar{x}_j \mapsto X_j$ and $\bar{z}_j \mapsto Z_j$ for all j .*

First, we'll remind ourselves of how measurement works in the *stabilizer formalism*.

Theorem A.5 (Measurement in the stabilizer formalism). *Suppose we have a stabilizer group $\mathcal{S} = \langle s_1, \dots, s_r \rangle \leq \mathcal{P}_n$ with rank r , and let $k = n - r$. Measuring a Hermitian Pauli p produces a measurement outcome $m \in \{-1, 1\}$ and a new stabilizer group $\mathcal{S}' \leq \mathcal{P}_n$. We have three cases:*

Case 1 (only possible when $r < n$): *p commutes with all generators s_j but $\pm p \notin \mathcal{S}$. In this case, the measurement outcome $m \in \{1, -1\}$ is random, and $\mathcal{S}' = \langle mp, s_1, s_2, \dots, s_r \rangle$.*

Case 2: *p commutes with all generators s_j and $\pm p \in \mathcal{S}$. Here the outcome m is deterministically ± 1 , and $\mathcal{S}' = \mathcal{S}$.*

Case 3: *p anti-commutes with at least one s_j . In fact, it can be shown that we can always pick a generating set such that p anti-commutes with exactly one generator s_1 , and commutes with the remaining generators s_2, \dots, s_r . In this case, the outcome m is again random, and $\mathcal{S}' = \langle mp, s_2, \dots, s_r \rangle$.*

We can then define measurement in the normalizer formalism via the same three cases.

Theorem A.6 (Measurement in the normalizer formalism). *Given a stabilizer group $\mathcal{S} = \langle s_1, \dots, s_r \rangle \leq \mathcal{P}_n$ with rank r , and letting $k = n - r$, we know that:*

$$N(\mathcal{S})/\mathcal{S} = \langle \bar{i}, \bar{x}_1, \bar{z}_1, \bar{x}_2, \bar{z}_2, \dots, \bar{x}_k, \bar{z}_k \rangle \cong \mathcal{P}_k \quad (9)$$

for some Paulis x_j and z_j in $N(\mathcal{S})$. Measuring a Hermitian Pauli $p \in \mathcal{P}_n$ produces a measurement outcome $m \in \{1, -1\}$ and a new stabilizer group \mathcal{S}' ; we can describe the new logical Pauli group $N(\mathcal{S}')/\mathcal{S}'$ in terms of the old one $N(\mathcal{S})/\mathcal{S}$ according to the same three cases:

Case 1 (only possible when $r < n$): *p commutes with all generators s_j but $\pm p \notin \mathcal{S}$. Equivalently, \bar{p} is an element of $N(\mathcal{S})/\mathcal{S}$ other than $\bar{1}$ or $-\bar{1}$. By choosing a different generating set for $N(\mathcal{S})/\mathcal{S}$ if needed, we may assume without loss of generality that $\bar{p} = \bar{x}_1$. We then find that:*

$$N(\mathcal{S}')/\mathcal{S}' = \langle \bar{i}, \bar{x}_2, \bar{z}_2, \dots, \bar{x}_k, \bar{z}_k \rangle \cong \mathcal{P}_{k-1} \quad (10)$$

where \mathcal{S}' is now $\langle mp, s_1, \dots, s_r \rangle$.

Case 2: p commutes with all generators s_j and $\pm p \in \mathcal{S}$. Equivalently, \bar{p} is either $\bar{\mathbb{1}}$ or $-\bar{\mathbb{1}}$ in $N(\mathcal{S})/\mathcal{S}$. This case is trivial: $\mathcal{S}' = \mathcal{S}$, hence $N(\mathcal{S}')/\mathcal{S}' = N(\mathcal{S})/\mathcal{S}$.

Case 3: p anti-commutes with at least one s_j . Equivalently, $\bar{p} \notin N(\mathcal{S})/\mathcal{S}$. Without loss of generality, we assume p anti-commutes with s_1 and commutes with s_2, \dots, s_r . Then by multiplying representatives of generators of $N(\mathcal{S})/\mathcal{S}$ by s_1 if needed, we can also assume $x_1, z_1, \dots, x_k, z_k$ all commute with p . We then find that:

$$N(\mathcal{S}')/\mathcal{S}' = \langle \bar{i}, \bar{x}_1, \bar{z}_1, \bar{x}_2, \bar{z}_2, \dots, \bar{x}_k, \bar{z}_k \rangle \cong \mathcal{P}_k \tag{11}$$

where \mathcal{S}' is now $\langle mp, s_2, \dots, s_r \rangle$.

Proof. Let's first note that since p is Hermitian, we have $p^2 = \mathbb{1}$. Thus if \bar{p} is in $N(\mathcal{S})/\mathcal{S}$, it must be of order 2. This is important because it means that the three cases above are disjoint and cover all possibilities; \bar{p} is either in $N(\mathcal{S})/\mathcal{S}$ or isn't, and if it is, it's either $\bar{\mathbb{1}}$, $-\bar{\mathbb{1}}$ or another order-2 element. In the following, we will occasionally need to be verbose as to whether \bar{q} denotes $q\mathcal{S} \in N(\mathcal{S})/\mathcal{S}$ or $q\mathcal{S}' \in N(\mathcal{S}')/\mathcal{S}'$.

We start with **Case 1**. If p commutes with all generators of \mathcal{S} but $\pm p \notin \mathcal{S}$, then by definition this means \bar{p} is an order-2 element of $N(\mathcal{S})/\mathcal{S}$ other than $\bar{\mathbb{1}}$ or $-\bar{\mathbb{1}}$. The converse also holds, hence our use of the word 'equivalently' above was justified. By **Lemma A.4**, we can always choose a presentation $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$ for $N(\mathcal{S})/\mathcal{S}$ such that $\bar{x}_1 = \bar{p}$ and the Pauli commutativity relations are satisfied by all \bar{x}_j and \bar{z}_j . Since \mathcal{S}' has minimal presentation $\langle mp, s_1, \dots, s_r \rangle$, **Lemma A.3** tells us that $N(\mathcal{S}')/\mathcal{S}'$ is isomorphic to \mathcal{P}_{k-1} .

We then claim that a presentation for $N(\mathcal{S}')/\mathcal{S}'$ is $\langle \bar{i}, \bar{x}_2, \bar{z}_2, \dots, \bar{x}_k, \bar{z}_k \rangle$. Let's first check this is well-defined, in that the representative q of each generator \bar{q} is in $N(\mathcal{S}')$. This follows from the fact that each q was in $N(\mathcal{S})$, so commutes with s_1, \dots, s_r , and because each \bar{q} commuted with $\bar{x}_1 = \bar{p}$ in $N(\mathcal{S})/\mathcal{S}$, hence q commutes with mp . Next, we check these generators remain independent in $N(\mathcal{S}')/\mathcal{S}'$. Similarly, this follows from the fact that they were independent of one another and of $\bar{p} = \bar{x}_1$ in $N(\mathcal{S})/\mathcal{S}$. Specifically, let's assume for a contradiction that a generator \bar{q} can be written in terms of the remaining $2(k-1)$ generators $\bar{q}_1, \dots, \bar{q}_{2(k-1)}$ as a word $\bar{w} := \bar{q}_1^{a_1} \dots \bar{q}_{2(k-1)}^{a_{2(k-1)}}$. Then this would imply $q = ws'$ for some $s' \in \mathcal{S}'$. This s' can in turn be written as $(mp)^{b_0} s_1^{b_1} \dots s_r^{b_r}$. But noting that $m = i^{1-m}$, we can define $w' = w(i^{1-m}p)^{b_0}$ as a new word over representatives of generators of $N(\mathcal{S})/\mathcal{S}$, and can also define $s = s_1^{b_1} \dots s_r^{b_r} \in \mathcal{S}$, so that $q = w's$. But then this says $\bar{q} = \bar{w}'$ in $N(\mathcal{S})/\mathcal{S}$, contradicting the fact that the generators $\bar{p}, \bar{q}, \bar{q}_1, \dots, \bar{q}_{2(k-1)}$ were independent in $N(\mathcal{S})/\mathcal{S}$.

Finally, we check that they obey the Pauli commutativity relations in $N(\mathcal{S}')/\mathcal{S}'$; this follows directly from the fact that they did so in $N(\mathcal{S})/\mathcal{S}$. Thus this presentation is a subgroup of $N(\mathcal{S}')/\mathcal{S}'$ isomorphic to \mathcal{P}_{k-1} . But since $N(\mathcal{S}')/\mathcal{S}'$ is itself isomorphic to \mathcal{P}_{k-1} , this presentation must be exactly $N(\mathcal{S}')/\mathcal{S}'$, as claimed.

Next up is **Case 2**, where p commutes with all generators s_j and $\pm p \in \mathcal{S}$. By definition, if $\pm p \in \mathcal{S}$ then $\bar{p} = \pm \bar{\mathbb{1}}$ in $N(\mathcal{S})/\mathcal{S}$, and vice versa, so again our use of 'equivalently' in the theorem statement was justified. Then there's nothing left to prove; since $\mathcal{S}' = \mathcal{S}$ in this case, we know $N(\mathcal{S}')/\mathcal{S}' = N(\mathcal{S})/\mathcal{S}$.

Finally, in **Case 3**, we know p anti-commutes with at least one generator of \mathcal{S} . Thus it cannot be in $N(\mathcal{S})$, and hence $\bar{p} \notin N(\mathcal{S})/\mathcal{S}$. Again, the converse also holds, justifying our use of 'equivalently' above.

We may assume without loss of generality that $\mathcal{S} = \langle s_1, \dots, s_r \rangle$ where p anti-commutes with s_1 but commutes with the remaining generators. We have a presentation $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$ for $N(\mathcal{S})/\mathcal{S}$. If any generator $q\mathcal{S} = \bar{q} \in \{\bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k\}$ is such that its representative q anti-commutes with p , we can pick a new representative $qs_1 \in q\mathcal{S}$ that must commute with p . Hence without loss of generality we may assume all representatives $x_1, z_1, \dots, x_k, z_k$ commute with p . Since \mathcal{S}' has minimal presentation $\langle mp, s_2, \dots, s_r \rangle$, we know $N(\mathcal{S}')/\mathcal{S}'$ is isomorphic to \mathcal{P}_k .

We claim that a presentation for $N(\mathcal{S}')/\mathcal{S}'$ is $\langle \bar{i}, \bar{x}_1, \bar{z}_1, \dots, \bar{x}_k, \bar{z}_k \rangle$. To prove this, we repeat similar steps as for [Case 1](#) above; we first show this is well-defined, in that each generator representative is in $N(\mathcal{S}')$. This follows from the fact that each one was in $N(\mathcal{S})$, so commutes with s_2, \dots, s_r , and from the assumption above that we picked them to commute with p , and hence mp too. Then we must show these generators remain independent. To see this, suppose for a contradiction that some generator \bar{q} can be written in terms of the other $2k$ generators $\bar{q}_1, \dots, \bar{q}_{2k}$, as some word $\bar{w} := \bar{q}_1^{a_1} \dots \bar{q}_{2k}^{a_{2k}}$, for $a_j \in \{0, 1\}$. This would mean $q = ws'$, for some $s' \in \mathcal{S}'$. This s' can itself be written as $(mp)^{b_1} s_2^{b_2} \dots s_r^{b_r}$. If $b_1 = 0$, then this contradicts the fact that the generators $\bar{q}, \bar{q}_1, \dots, \bar{q}_{2k}$ were independent in $N(\mathcal{S})/\mathcal{S}$. But equally if $b_1 = 1$, then s_1 anticommutes with ws' but not q , which is again a contradiction, since $q = ws'$. So the generators must all be independent. Finally, we must prove the generators still satisfy the Pauli commutation relations; this follows directly from the fact that they did so in $N(\mathcal{S})/\mathcal{S}$. We can conclude this presentation is a subgroup of $N(\mathcal{S}')/\mathcal{S}'$ isomorphic to \mathcal{P}_{n-k} , and since $N(\mathcal{S}')/\mathcal{S}'$ is itself isomorphic to $N(\mathcal{S}')/\mathcal{S}'$, it must be that this presentation is the whole of $N(\mathcal{S}')/\mathcal{S}'$, as claimed. \square

B ISG code example

Here we give a simple example of an ISG code and the evolution over time of its ISG \mathcal{S}_t and logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t$. Once we know the ISG's evolution, we can then immediately write down detectors for the code too. We do all this algebraically, using the stabilizer and normalizer formalisms of [Appendix A](#). We will consider the ISG code defined by the schedule $\mathcal{M} = [\langle X_1X_2, X_3X_4 \rangle, \langle Z_1Z_3, Z_2Z_4 \rangle]$. Those who've seen a bit of error correction before may recognise this as the distance-two Bacon-Shor subsystem code, which can in turn be thought of as a subsystem implementation of the distance-two rotated surface code.

B.1 Instantaneous stabilizer group

The initial ISG \mathcal{S}_{-1} is the trivial group $\{\mathbb{1}\}$, as ever. Using the stabilizer formalism we can see how \mathcal{S}_t evolves. In the zero-th round we measure X_1X_2 and X_3X_4 , so we are in [Case 1](#) for both; we get random outcomes $m_{X_1X_2}^{(0)}$ and $m_{X_3X_4}^{(0)}$ respectively, and the ISG immediately afterwards is:

$$\mathcal{S}_0 = \langle m_{X_1X_2}^{(0)} X_1X_2, m_{X_3X_4}^{(0)} X_3X_4 \rangle \quad (12)$$

By multiplying the second generator by the first, we get a new presentation for the same group that'll be more convenient for figuring out what \mathcal{S}_1 is in a second:

$$\mathcal{S}_0 = \langle m_{X_1X_2}^{(0)} X_1X_2, m_{X_1X_2}^{(0)} m_{X_3X_4}^{(0)} X_1X_2X_3X_4 \rangle \quad (13)$$

In the next round we measure Z_1Z_3 and Z_2Z_4 . First, consider Z_1Z_3 . Since it anticommutes with X_1X_2 but commutes with $X_1X_2X_3X_4$, we're in [Case 3](#); $m_{X_1X_2}^{(0)} X_1X_2$ is removed from the generating set of \mathcal{S}_1 , and replaced by $m_{Z_1Z_3}^{(1)} Z_1Z_3$, where $m_{Z_1Z_3}^{(1)}$ is the random outcome of measuring Z_1Z_3 . Next, consider Z_2Z_4 :

since it commutes with Z_1Z_3 and $X_1X_2X_3X_4$ but neither it nor its negation is in \mathcal{S}_0 , we're in *Case 1*. So $m_{Z_2Z_4}^{(1)}Z_2Z_4$ is added as a generator of \mathcal{S}_1 , where $m_{Z_2Z_4}^{(1)}$ is the random outcome of measuring Z_2Z_4 . So altogether:

$$\mathcal{S}_1 = \langle m_{Z_1Z_3}^{(1)}Z_1Z_3, m_{Z_2Z_4}^{(1)}Z_2Z_4, m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_2X_3X_4 \rangle \quad (14)$$

For every subsequent timestep $t > 1$, the ISG \mathcal{S}_t will have rank 3. That is, this code is established after $T = 1$ timesteps. Let's spell this out for $t = 2$, wherein we measure X_1X_2 and X_3X_4 again. We'll first use a different generating set for \mathcal{S}_1 , like we did just a second ago:

$$\mathcal{S}_1 = \langle m_{Z_1Z_3}^{(1)}Z_1Z_3, m_{Z_1Z_3}^{(1)}m_{Z_2Z_4}^{(1)}Z_1Z_2Z_3Z_4, m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_2X_3X_4 \rangle \quad (15)$$

We measure X_1X_2 , which anticommutes with Z_1Z_3 , commutes with the other generators, and neither it nor its negation is in \mathcal{S}_1 . So we're in *Case 3*; we get random outcome $m_{X_1X_2}^{(2)}$, and $m_{X_1X_2}^{(2)}X_1X_2$ replaces $m_{Z_1Z_3}^{(1)}Z_1Z_3$ as a generator. In particular, the product $m_{X_1X_2}^{(2)}m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_3X_4$ of $m_{X_1X_2}^{(2)}X_1X_2$ and $m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_2X_3X_4$ is now in \mathcal{S}_1 . So when we then measure X_3X_4 , we're in *Case 2*; we get deterministic outcome $m_{X_3X_4}^{(2)} = m_{X_1X_2}^{(2)}m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}$, and \mathcal{S}_2 remains unchanged:

$$\mathcal{S}_2 = \langle m_{X_1X_2}^{(2)}X_1X_2, m_{Z_1Z_3}^{(1)}m_{Z_2Z_4}^{(1)}Z_1Z_2Z_3Z_4, m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_2X_3X_4 \rangle \quad (16)$$

By multiplying the third generator by the first and using $m_{X_3X_4}^{(2)} = m_{X_1X_2}^{(2)}m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}$, we get the following more convenient presentation:

$$\mathcal{S}_2 = \langle m_{X_1X_2}^{(2)}X_1X_2, m_{X_3X_4}^{(2)}X_3X_4, m_{Z_1Z_3}^{(1)}m_{Z_2Z_4}^{(1)}Z_1Z_2Z_3Z_4 \rangle \quad (17)$$

And indeed this has rank 3 again, as promised. If we continued this sort of analysis, we'd see that we get:

$$\mathcal{S}_t = \begin{cases} \{\mathbb{1}\} & \text{if } t < 0 \\ \langle m_{X_1X_2}^{(0)}X_1X_2, m_{X_3X_4}^{(0)}X_3X_4 \rangle & \text{if } t = 0 \\ \langle m_{Z_1Z_3}^{(t)}Z_1Z_3, m_{Z_2Z_4}^{(t)}Z_2Z_4, m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_2X_3X_4 \rangle & \text{if } t > 0 \text{ odd} \\ \langle m_{X_1X_2}^{(t)}X_1X_2, m_{X_3X_4}^{(t)}X_3X_4, m_{Z_1Z_3}^{(1)}m_{Z_2Z_4}^{(1)}Z_1Z_2Z_3Z_4 \rangle & \text{if } t > 0 \text{ even} \end{cases} \quad (18)$$

Note that the ISG here is *dynamic* even after establishment at $T = 1$; it changes from one timestep to the next. As pointed out in the table in [Figure 1](#), this is characteristic of a subsystem code. In fact, we can be even more specific; for any subsystem code, the way in which the ISG changes between timesteps (after establishment) is more commonly called *gauge fixing* - we give more detail on this in [Appendix D.1](#).

B.2 Detectors

By analysing the evolution of the ISG of the code, we also uncover the code's detectors. Specifically, any-time we measure a Pauli p and find ourselves in *Case 2*, we learn a detector of the code. Moreover, a generating set of detectors of the code can be found this way. To see why this is, recall that a detector is defined to be a set of measurement outcomes whose product is deterministic in the absence of noise. In any ISG code, at any timestep t , every element of the ISG \mathcal{S}_t is of the form $m_{p_1}^{(t_1)} \dots m_{p_\ell}^{(t_\ell)} p_1 \dots p_\ell$, where each p_j is a Pauli and $m_{p_j}^{(t_j)}$ is the outcome of measuring p_j at timestep t_j . If at time t we measure some Pauli p

and land in *Case 2*, it means that $\pm p$ was already in \mathcal{S}_t . In other words, $\pm p = m_{p_1}^{(t_1)} \dots m_{p_\ell}^{(t_\ell)} p_1 \dots p_\ell$. So the measurement outcome $m_p^{(t)}$ is deterministically equal to $m_{p_1}^{(t_1)} \dots m_{p_\ell}^{(t_\ell)}$; that is, the product $m_p^{(t)} m_{p_1}^{(t_1)} \dots m_{p_\ell}^{(t_\ell)}$ is deterministic (it's 1, in this case). Hence the formal product $m_p^{(t)} m_{p_1}^{(t_1)} \dots m_{p_\ell}^{(t_\ell)}$ is a detector.

For example, in the analysis above, the measurement of X_3X_4 at time $t = 2$ was handled by *Case 2*. We found that the measurement outcome $m_{X_3X_4}^{(2)}$ was deterministically equal to the product $m_{X_1X_2}^{(2)} m_{X_1X_2}^{(0)} m_{X_3X_4}^{(0)}$ of previous measurements. Thus the formal product $m_{X_1X_2}^{(2)} m_{X_3X_4}^{(2)} m_{X_1X_2}^{(0)} m_{X_3X_4}^{(0)}$ is a detector. Indeed, were we to continue this analysis, we would find detectors:

$$\begin{aligned} m_{X_1X_2}^{(t)} m_{X_3X_4}^{(t)} m_{X_1X_2}^{(t-2)} m_{X_3X_4}^{(t-2)} & \text{ at every even timestep } t \geq 2 \\ m_{Z_1Z_3}^{(t)} m_{Z_2Z_4}^{(t)} m_{Z_1Z_3}^{(t-2)} m_{Z_2Z_4}^{(t-2)} & \text{ at every odd timestep } t \geq 2 \end{aligned} \quad (19)$$

The reason we said this gives us a *generating set* of detectors is because detectors form a group. Suppose we have two detectors $m_1 \dots m_u$ and $m'_1 \dots m'_v$. Since by definition these products are deterministic, the total product $m_1 \dots m_u m'_1 \dots m'_v$ must be deterministic too. Hence the formal product $m_1 \dots m_u m'_1 \dots m'_v$ is a detector. The group's unit is the trivial formal product, which corresponds to an empty set of measurement outcomes. We can write this as $\prod_{m_j \in \emptyset} m_j = 1$. Every detector is then its own inverse; as a formal product whose powers are taken mod 2, we have $m_1 \dots m_u m_1 \dots m_u = m_1^2 \dots m_u^2 = m_1^0 \dots m_u^0 = 1$.

B.3 Logical Pauli group

Tracking the evolution of the logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t$ has a very similar feel as for \mathcal{S}_t , but there's a little bit more to picking the right presentation for the group now, in order to be able to apply the normalizer formalism.

As a reminder, we're considering the schedule $\mathcal{M} = [\langle X_1X_2, X_3X_4 \rangle, \langle Z_1Z_3, Z_2Z_4 \rangle]$. Since the initial ISG \mathcal{S}_{-1} is trivial, the most obvious presentation for the initial logical Pauli group is:

$$N(\mathcal{S}_{-1})/\mathcal{S}_{-1} = \langle \bar{i}, \bar{X}_1, \bar{Z}_1, \bar{X}_2, \bar{Z}_2, \bar{X}_3, \bar{Z}_3, \bar{X}_4, \bar{Z}_4 \rangle \quad (20)$$

But at $t = 0$ we measure X_1X_2 and X_3X_4 , so in order to use the normalizer formalism, we need a different presentation. First, we consider measuring X_1X_2 . We know we'll be in *Case 1*; we can see this either by observing that $\bar{X}_1\bar{X}_2$ is an element of $N(\mathcal{S}_{-1})/\mathcal{S}_{-1}$ other than $\bar{1}$ or $-\bar{1}$, or equivalently by observing that X_1X_2 is not in \mathcal{S}_{-1} but commutes with all its generators (vacuously, because there are none). To apply the formalism, we can first multiply \bar{X}_1 by \bar{X}_2 , so that $\bar{X}_1\bar{X}_2$ is a generator. One might think that we're then good to go, but we must be careful: this multiplication means the generators no longer satisfy the Pauli commutativity relations. To account for this, we can multiply \bar{Z}_2 by \bar{Z}_1 , to get:

$$N(\mathcal{S}_{-1})/\mathcal{S}_{-1} = \langle \bar{i}, \bar{X}_1\bar{X}_2, \bar{Z}_1, \bar{X}_2, \bar{Z}_1\bar{Z}_2, \bar{X}_3, \bar{Z}_3, \bar{X}_4, \bar{Z}_4 \rangle \quad (21)$$

Now we can apply the formalism! Doing this simply removes $\bar{X}_1\bar{X}_2$ and \bar{Z}_1 from the generating set. We can repeat something similar for the measurement of X_3X_4 ; namely, we can multiply \bar{X}_3 by \bar{X}_4 and \bar{Z}_4 by \bar{Z}_3 , then apply *Case 1* to get:

$$N(\mathcal{S}_0)/\mathcal{S}_0 = \langle \bar{i}, \bar{X}_2, \bar{Z}_1\bar{Z}_2, \bar{X}_4, \bar{Z}_3\bar{Z}_4 \rangle \quad (22)$$

where $\mathcal{S}_0 = \langle m_{X_1X_2}^{(0)} X_1X_2, m_{X_3X_4}^{(0)} X_3X_4 \rangle$. At the next timestep $t = 1$, we first measure Z_1Z_3 . This puts us in *Case 3*, and transforms \mathcal{S}_0 to $\mathcal{S}'_0 = \langle m_{Z_1Z_3}^{(1)} Z_1Z_3, m_{X_1X_2}^{(0)} m_{X_3X_4}^{(0)} X_1X_2X_3X_4 \rangle$. Since all representatives of the

generators of $N(\mathcal{S}_0)/\mathcal{S}_0$ already commute with Z_1Z_3 , we don't need to change our presentation to be able to apply the formalism. We get:

$$N(\mathcal{S}'_0)/\mathcal{S}'_0 = \langle \bar{i}, \bar{X}_2, \bar{Z}_1\bar{Z}_2, \bar{X}_4, \bar{Z}_3\bar{Z}_4 \rangle \quad (23)$$

The other measurement we make at timestep $t = 1$ is Z_2Z_4 . This means we're in *Case 1*, but we need an alternative presentation first, in which $\bar{Z}_2\bar{Z}_4$ is a generator. We can do this by multiplying $\bar{Z}_3\bar{Z}_4$ by $\bar{Z}_1\bar{Z}_2$ to get $\bar{Z}_1\bar{Z}_2\bar{Z}_3\bar{Z}_4$, then, since $m_{Z_1Z_3}^{(1)}Z_1Z_3 \in \mathcal{S}'_0$, we can pick a new representative $m_{Z_1Z_3}^{(1)}Z_2Z_4$. If necessary, we can multiply this by \bar{i}^2 to get just $\bar{Z}_2\bar{Z}_4$, for convenience. But now our generators don't satisfy the Pauli commutativity relations again. Fortunately we can fix this by multiplying \bar{X}_2 by \bar{X}_4 , picking a new representative $m_{X_1X_2}^{(0)}m_{X_3X_4}^{(0)}X_1X_3$, and for convenience multiplying by \bar{i}^2 if needed to get just $\bar{X}_1\bar{X}_3$. Altogether, this gives:

$$N(\mathcal{S}'_0)/\mathcal{S}'_0 = \langle \bar{i}, \bar{X}_1\bar{X}_3, \bar{Z}_1\bar{Z}_2, \bar{X}_4, \bar{Z}_2\bar{Z}_4 \rangle \quad (24)$$

We can now directly apply *Case 1*; the resulting new logical Pauli group is:

$$N(\mathcal{S}_1)/\mathcal{S}_1 = \langle \bar{i}, \bar{X}_1\bar{X}_3, \bar{Z}_1\bar{Z}_2 \rangle \quad (25)$$

And this is how things stay: for every measurement p performed at any future timestep $t > 1$, we're either in *Case 2* or *Case 3* - and when it's the latter, the given representatives of generators of $N(\mathcal{S}_1)/\mathcal{S}_1$ even commute with p already. Hence:

$$N(\mathcal{S}_t)/\mathcal{S}_t = \begin{cases} \langle \bar{i}, \bar{X}_1, \bar{Z}_1, \bar{X}_2, \bar{Z}_2, \bar{X}_3, \bar{Z}_3, \bar{X}_4, \bar{Z}_4 \rangle \cong \mathcal{P}_4 & \text{if } t < 0 \\ \langle \bar{i}, \bar{X}_2, \bar{Z}_1\bar{Z}_2, \bar{X}_4, \bar{Z}_3\bar{Z}_4 \rangle \cong \mathcal{P}_2 & \text{if } t = 0 \\ \langle \bar{i}, \bar{X}_1\bar{X}_3, \bar{Z}_1\bar{Z}_2 \rangle \cong \mathcal{P}_1 & \text{if } t > 0 \end{cases} \quad (26)$$

We can thus see that the logical Pauli group is *static* after becoming established at time $T = 1$. Again, as noted in the table in [Figure 1](#), this is characteristic of a subsystem code.

C Remarks on ISG codes

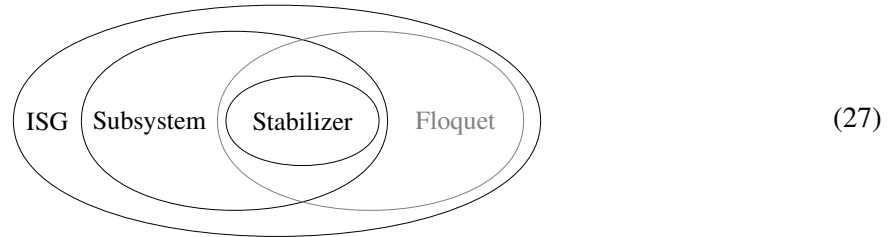
Here we comment on some aspects of ISG codes that we didn't have space for in the main text. First, we note that $|N(\mathcal{S}_t)/\mathcal{S}_t|$ is a non-increasing sequence in t . That is, the number of qubits encoded in an ISG code can only ever stay the same or decrease, but never increase. One can see this from the normalizer formalism of the previous section; $N(\mathcal{S}_t)/\mathcal{S}_t$ is initially isomorphic to \mathcal{P}_n , so encodes n qubits, then evolves exclusively by measuring Paulis $p \in \mathcal{P}_n$. Such a measurement can only ever cause the number of encoded qubits to decrease by one (*Case 1*) or stay the same (*Cases 2 and 3*).

Second, we note that, for all t , \mathcal{S}_t has rank $r \iff N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_{n-r}$. Thus when we defined 'establishment' as being the existence of a T such that \mathcal{S}_t has fixed rank r for all $t \geq T$, we could equally have defined it in terms of logical qubits, as a T such that $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$ for fixed $k = n - r$ for all $t \geq T$.

We confess that we have used a very naive definition of distance; namely, the minimum weight of any element of $N(\mathcal{S}_t)/\mathcal{S}_t$ over all $t \geq T$. This only considers problematic Pauli operators within a single timestep, and fails to take into account that there can be combinations of Pauli operators over multiple timesteps which anti-commute with logical operators but do not violate any detectors. Our definition

should perhaps be downgraded to a *spacelike distance*, and a better definition of distance could instead be directly related to detecting and operating regions.

An interesting unanswered question concerns when two ISG codes should be considered equivalent. For example, suppose we have an ISG code $\mathcal{M} = [\dots, \mathcal{M}_t, \mathcal{M}_{t+1}, \dots]$. If $\mathcal{M}_t \mathcal{M}_{t+1}$ is Abelian (i.e. all Paulis in \mathcal{M}_t and \mathcal{M}_{t+1} commute with each other), should this be considered equivalent to the ISG code $\mathcal{M}' = [\dots, \mathcal{M}_t \mathcal{M}_{t+1}, \dots]$? Certainly the ISGs \mathcal{S}_{t+1} of \mathcal{M} and \mathcal{S}'_t of \mathcal{M}' will be identical, and thus so too the logical Pauli groups $N(\mathcal{S}_{t+1})/\mathcal{S}_{t+1}$ and $N(\mathcal{S}'_t)/\mathcal{S}'_t$. If we think the answer to this question is yes, then any stabilizer code $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_t, \dots]$ is equivalent to the period-1 code $\mathcal{M} = [\mathcal{M}_0 \mathcal{M}_1 \dots \mathcal{M}_t \dots]$. In particular, this would make every stabilizer code a Floquet code, demanding we update our Venn diagram from [Figure 1](#) to:



Other cases also arise - should the periodic ISG code $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_{\ell-2}, \mathcal{M}_{\ell-1}]$ be considered equivalent to one like $\mathcal{M}' = [\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_{\ell-1}, \mathcal{M}_0]$, which has the same measurement sequence only cyclically permuted? We would be interested to hear more opinions on this matter.

One could also point out that ‘establishment’ isn’t a strictly necessary part of the definition of an ISG code. For example, one could imagine a measurement schedule $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$ such that the sequence $|N(\mathcal{S}_t)/\mathcal{S}_t|$ never becomes constant, but decreases so slowly that one can still use a subset of the encoded logical qubits to perform quantum computation. We would be very interested to hear of any non-trivial such \mathcal{M} . Finally, we remark that the definition could (and probably should) be extended such that, in addition to Pauli measurements, Clifford operations are also allowed to be applied to the qubits of the code.

D Remarks on subsystem codes

Here, we give more detail on the relationship between subsystem codes and ISG codes. In [Definition 2.3](#), we defined a subsystem code as a type of ISG code, and confessed that this definition actually deviates slightly from the usual one. We now discuss this deviation at length. We first introduce new notation - for any group $\mathcal{G} \leq \mathcal{P}_n$, we use \mathcal{G}^* to denote $\mathcal{G}/\langle i \rangle$, the same group modulo phases - and we remind ourselves of previous notation; for groups $\mathcal{G}_1, \dots, \mathcal{G}_\ell \leq \mathcal{P}_n$, we write the product $\mathcal{G}_1 \dots \mathcal{G}_\ell$ to mean the group generated by the union of generating sets for $\mathcal{G}_1, \dots, \mathcal{G}_\ell$, and we let \mathcal{P}_k° denote the ‘almost Pauli group’ $\langle X_1, Z_1, \dots, X_k, Z_k \rangle$, which satisfies $\langle i \rangle \mathcal{P}_k^\circ = \mathcal{P}_k$.

At a high level, subsystem codes are stabilizer codes in which some logical qubits aren’t used to store logical information. So the formalism doesn’t give rise to new codes, but rather gives rise to more flexible ways of correcting errors on these codes [[Bac06](#)]. These unused logical qubits are referred to as *gauge qubits*. Much like stabilizer codes can be defined entirely by a stabilizer group, subsystem codes can be

defined entirely by a subgroup $\mathcal{G} \leq \mathcal{P}_n$ of the Pauli group. The only requirement on this \mathcal{G} is that it must contain i (and hence need not be Abelian). We must also pick a presentation $\langle i, s_1, \dots, s_R \rangle$ of $Z(\mathcal{G})$ such that $\mathcal{S} = \langle s_1, \dots, s_R \rangle$ is a stabilizer group, but the actual choice is arbitrary. Crucially, at no point above was a measurement schedule required, which makes this definition more general than ours. Throughout the rest of this section, we will forget our [Definition 2.3](#); whenever we mention a subsystem code, we now just mean a gauge group \mathcal{G} .

It was originally envisioned that, in order to use a subsystem code for error correction, one would repeatedly measure a generating set for \mathcal{S} [[Pou05](#)]. Later, it was pointed out that actually one could potentially measure a generating set for a group \mathcal{G}' satisfying $\langle i \rangle \mathcal{G}' = \mathcal{G}$, and combine the outcomes in order to infer the measurement outcomes of a generating set for \mathcal{S} [[Bom10](#)]. Since the generators of such a \mathcal{G}' are typically of lower-weight than those of \mathcal{S} , this provides a significant practical advantage, and is thus generally how subsystem codes are now envisioned to be implemented. But since \mathcal{G} can be non-Abelian, we can't necessarily measure a full generating set for \mathcal{G}' at once. Instead, if choosing this route, we must specify a schedule in which to measure mutually commuting sets of generators. This leads us to define an *associated ISG code* for a subsystem code:

Definition D.1. *Given a subsystem code \mathcal{G} , an **associated ISG code** for it is an ISG code $\mathcal{M}_{\mathcal{G}} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$ that establishes at some time $T \geq 0$, such that $\langle i \rangle \mathcal{M}_0 \mathcal{M}_1 \dots = \mathcal{G}$, and the ISG \mathcal{S}_t satisfies $\mathcal{S}^* \leq \mathcal{S}_t^*$ for all $t \geq T$.*

In the other direction, we can define the *associated subsystem code* for an ISG code:

Definition D.2. *Given an ISG code $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$, we say its **associated subsystem code** is defined by the gauge group $\mathcal{G}_{\mathcal{M}} = \langle i \rangle \mathcal{M}_0 \mathcal{M}_1 \dots$*

Note that a single subsystem code \mathcal{G} can have many associated ISG codes $\mathcal{M}_{\mathcal{G}}$, but an ISG code \mathcal{M} has a unique associated subsystem code $\mathcal{G}_{\mathcal{M}}$. As an example, in [Appendix B](#) we said we looked at the distance-two Bacon-Shor code, which is normally defined via the gauge group $\mathcal{G} = \langle i, X_1 X_2, X_3 X_4, Z_1 Z_3, Z_2 Z_4 \rangle$. However, since this group is non-Abelian, if we choose to implement it by measuring generators of a \mathcal{G}' satisfying $\langle i \rangle \mathcal{G}' = \mathcal{G}$ (the obvious choice being $\mathcal{G}' = \langle X_1 X_2, X_3 X_4, Z_1 Z_3, Z_2 Z_4 \rangle$), then we require a measurement schedule. In that section, we thus actually looked at the associated ISG code $\mathcal{M}_{\mathcal{G}} = [\langle X_1 X_2, X_3 X_4 \rangle, \langle Z_1 Z_3, Z_2 Z_4 \rangle]$. Let's also point out quickly that, while any partition of generators of \mathcal{G} into mutually commuting subsets produces an ISG code, some fall foul of the definition of an *associated* ISG code for \mathcal{G} . The schedule $\mathcal{M}_{\mathcal{G}} = [\langle X_1 X_2 \rangle, \langle Z_2 Z_4 \rangle, \langle X_3 X_4 \rangle, \langle Z_1 Z_3 \rangle]$ is one such example for the Bacon-Shor code above. Here the ISG at any time t is (up to phases) exactly the group \mathcal{M}_t whose generating set was just measured. That is, letting $m_p^{(t)}$ denote the outcome of measuring Pauli p at time $t \geq 0$, we have:

$$\mathcal{S}_t = \begin{cases} \langle m_{X_1 X_2}^{(t)} X_1 X_2 \rangle & \text{if } t = 0 \pmod{4} \\ \langle m_{Z_2 Z_4}^{(t)} Z_2 Z_4 \rangle & \text{if } t = 1 \pmod{4} \\ \langle m_{X_3 X_4}^{(t)} X_3 X_4 \rangle & \text{if } t = 2 \pmod{4} \\ \langle m_{Z_1 Z_3}^{(t)} Z_1 Z_3 \rangle & \text{if } t = 3 \pmod{4} \end{cases} \quad (28)$$

Since $\mathcal{S}^* = Z(\mathcal{G})^* = \langle \overline{X_1 X_2 X_3 X_4}, \overline{Z_1 Z_3 Z_2 Z_4} \rangle$, this ISG code doesn't satisfy $\mathcal{S}^* \leq \mathcal{S}_t^*$ for any t , so isn't an associated ISG code for \mathcal{G} .

We are very interested in precisely pinning down the relationship between a subsystem code \mathcal{G} and any associated ISG code $\mathcal{M}_{\mathcal{G}}$ for it. To this end, we state below one result and one open question. For a

subsystem code \mathcal{G} , the logical (almost!) Pauli group is defined as $N(\mathcal{S})/\mathcal{G}$, and is isomorphic to \mathcal{P}_k° , for some $k \in \mathbb{Z}_{\geq 0}$. This k is defined to be the number of logical qubits of the subsystem code. The following theorem effectively says that the logical qubits of a subsystem code are a subset of the logical qubits of any associated ISG code. More precisely:

Theorem D.3. *For any subsystem code \mathcal{G} and any associated ISG code $\mathcal{M}_{\mathcal{G}}$, there exist fixed Paulis $x_1, z_1, \dots, x_k, z_k$ such that $\langle x_1 \mathcal{G}, z_1 \mathcal{G}, \dots, x_k \mathcal{G}, z_k \mathcal{G} \rangle$ is a presentation for $N(\mathcal{S})/\mathcal{G} \cong \mathcal{P}_k^\circ$, and, for all t , $\langle i \mathcal{S}_t, x_1 \mathcal{S}_t, z_1 \mathcal{S}_t, \dots, x_k \mathcal{S}_t, z_k \mathcal{S}_t \rangle$ is a subgroup of $N(\mathcal{S}_t)/\mathcal{S}_t$ isomorphic to \mathcal{P}_k .*

We prove the statement in two steps. First we show that every non-trivial coset of the subsystem code's logical Pauli group $N(\mathcal{S})/\mathcal{G}$ has a representative - a *bare logical* - which is also a non-trivial logical operator for any associated ISG code at all times. Secondly, we show that inequivalent non-trivial logical operators of the subsystem code are also inequivalent non-trivial logical operators of any associated ISG code. Both statements rely on the fact that, for any subsystem code \mathcal{G} and associated ISG code $\mathcal{M}_{\mathcal{G}} = [\mathcal{M}_0, \mathcal{M}_1, \dots]$, the ISG is always a subgroup of the gauge group: $\forall t \in \mathbb{Z}, \mathcal{S}_t \leq \mathcal{G}$. This can be seen directly from the fact that \mathcal{S}_t is formed by measuring generating sets for groups $\mathcal{M}_t, \mathcal{M}_{t-1}, \dots, \mathcal{M}_0 \leq \mathcal{G}$. Thus every element of \mathcal{S}_t is a product of elements of \mathcal{G} , multiplied by some product ± 1 of measurement outcomes - both of which are again in \mathcal{G} . We start in earnest by importing the following lemma separately, so that we can refer to it again later:

Lemma D.4 ([Pou05]). *For any subsystem code $\mathcal{G} \leq \mathcal{P}_n$, the gauge group can be written as $\mathcal{G} = \mathcal{S}\langle i \rangle \langle x_1, z_1, \dots, x_K, z_K \rangle$, for some Paulis $x_1, z_1, \dots, x_K, z_K \in \mathcal{P}_n$, with $\langle x_1, z_1, \dots, x_K, z_K \rangle$ isomorphic to \mathcal{P}_K° .*

The group $\langle x_1, z_1, \dots, x_K, z_K \rangle \cong \mathcal{P}_K^\circ$ above corresponds exactly to the K gauge qubits of the subsystem code. Next we define bare and logical operators.

Definition D.5. *Given a subsystem code \mathcal{G} , a logical operator (a member of a coset of $N(\mathcal{S})/\mathcal{G}$) is **bare** if it commutes with all gauge qubit logical operators $x_1, z_1, \dots, x_K, z_K$, and **dressed** if not.*

Every logical operator coset in $N(\mathcal{S})/\mathcal{G}$ contains at least one bare logical operator. We now relate these bare logicals to the *centralizer* $C(\mathcal{G}) = \{p \in \mathcal{P}_n \mid \forall g \in \mathcal{G}, pg = gp\}$ of \mathcal{G} in \mathcal{P}_n - i.e. all the Pauli operators that commute with all elements of \mathcal{G} .

Lemma D.6. *Given a subsystem code \mathcal{G} , the set of all non-trivial bare logical operators is exactly $C(\mathcal{G}) - \mathcal{G}$.*

Proof. A logical operator is a member of a coset of $N(\mathcal{S})/\mathcal{G}$, so - ignoring the group structure and just thinking in terms of sets - the non-trivial logical operators are exactly the set $N(\mathcal{S}) - \mathcal{G}$. For stabilizer groups like \mathcal{S} , the normalizer and centralizer coincide: $N(\mathcal{S}) = C(\mathcal{S})$ [Got09, Section 3.2]. So a logical operator commutes with all elements of \mathcal{S} . A *bare* logical operator by definition also commutes with all gauge qubit logical operators $x_1, z_1, \dots, x_K, z_K$. From **Lemma D.4** we know $\mathcal{G} = \mathcal{S}\langle i \rangle \langle x_1, z_1, \dots, x_K, z_K \rangle$, and since every Pauli operator commutes with i , we deduce that a bare logical operator commutes with all of \mathcal{G} . So all non-trivial bare logical operators are in $C(\mathcal{G}) - \mathcal{G}$. The converse can also be seen to hold; every element of $C(\mathcal{G}) - \mathcal{G}$ is a non-trivial bare logical operator. \square

We now tie this all together.

Proof of Theorem D.3. A non-trivial subsystem code logical operator is an element of $C(\mathcal{G}) - \mathcal{G}$, by **Lemma D.6**. Since $\mathcal{S}_t \leq \mathcal{G}$ for all t , we have $C(\mathcal{G}) \leq C(\mathcal{S}_t)$, and hence $C(\mathcal{G}) - \mathcal{G} \leq C(\mathcal{S}_t) - \mathcal{S}_t$. If we

again use the fact that centralizers and normalizers coincide for stabilizer groups like \mathcal{S}_t , we see that any non-trivial bare logical operator is a member of $N(\mathcal{S}_t) - \mathcal{S}_t$, and is hence a non-trivial logical operator of the ISG code at all timesteps.

It remains to show that inequivalent logical operators of the subsystem code correspond to inequivalent logical operators of any associated ISG code. Let $p, q \in N(\mathcal{S}) - \mathcal{G}$ be inequivalent logicals of the subsystem code - i.e. $p\mathcal{G} \neq q\mathcal{G}$ in $N(\mathcal{S})/\mathcal{G}$. Since \mathcal{S}_t is a subgroup of \mathcal{G} for all $t \in \mathbb{Z}$, we can directly infer that $p\mathcal{S}_t \neq q\mathcal{S}_t$. So we're done; we've shown that choosing *bare* logical representatives $x_1, z_1, \dots, x_k, z_k$ of $\langle x_1\mathcal{G}, z_1\mathcal{G}, \dots, x_k\mathcal{G}, z_k\mathcal{G} \rangle = N(\mathcal{S})/\mathcal{G}$ gives us a presentation $\langle i\mathcal{S}_t x_1 \mathcal{S}_t, z_1 \mathcal{S}_t, \dots, x_k \mathcal{S}_t, z_k \mathcal{S}_t \rangle \cong \mathcal{P}_k$ of a subgroup of $N(\mathcal{S}_t)/\mathcal{S}_t$ for all $t \in \mathbb{Z}$. \square

As a corollary, we can now bound the distance of any associated ISG code. For any subsystem code \mathcal{G} , let $d_{\mathcal{G}}$ be its *bare distance*; that is, the minimum weight of any bare logical operator of the code. Then the distance d of any associated ISG code must be at most $d_{\mathcal{G}}$. Similarly, letting $k_{\mathcal{G}}$ be the number of logical qubits in the subsystem code, the number k of logical qubits of any associated ISG code must be at least $k_{\mathcal{G}}$. For this latter bound, it is natural to ask whether we can always find an associated ISG code which saturates it - in other words, can we always find an associated ISG code whose logical qubits are effectively exactly those of the subsystem code?

Open question D.7. *For every subsystem code \mathcal{G} , does there always exist an associated ISG code $\mathcal{M}_{\mathcal{G}}$ that establishes at time T , and fixed Paulis $x_1, z_1, \dots, x_k, z_k$, such that $\langle x_1\mathcal{G}, z_1\mathcal{G}, \dots, x_k\mathcal{G}, z_k\mathcal{G} \rangle$ is a presentation for $N(\mathcal{S})/\mathcal{G} \cong \mathcal{P}_k^{\circ}$, and, for all t , $\langle i\mathcal{S}_t, x_1\mathcal{S}_t, z_1\mathcal{S}_t, \dots, x_k\mathcal{S}_t, z_k\mathcal{S}_t \rangle$ is a presentation for $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_k$?*

For many subsystem codes, we can find such an associated ISG. In the small Bacon-Shor code of [Appendix B](#), for example, we analysed the associated ISG code $[\langle X_1X_2, X_3X_4 \rangle, \langle Z_1Z_3, Z_2Z_4 \rangle]$, and found that after establishment it had logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t = \langle i\mathcal{S}_t, X_1X_3\mathcal{S}_t, Z_1Z_2\mathcal{S}_t \rangle$ - see [Equation \(26\)](#). The same representatives generate $N(\mathcal{S})/\mathcal{G} = \langle X_1X_3\mathcal{G}, Z_1Z_2\mathcal{G} \rangle$ too. For some other subsystem codes, however, we have not managed to find such an associated ISG code - examples include Bombín's *topological subsystem code* introduced in Ref. [\[Bom10, Section IV\]](#). We plan to investigate this subsystem code through the lens of ISG codes in future work.

D.1 Gauge fixing

We close this section with a word on what *gauge fixing* means from an ISG code perspective. For any subsystem code $\mathcal{G} \leq \mathcal{P}_n$ implemented by measuring generators of some \mathcal{G}' satisfying $\langle i \rangle \mathcal{G}' = \mathcal{G}$, gauge fixing is the name given to tracking what happens to the group stabilizing the system of n qubits as we measure these generators. The name comes from the following: as shown in [Lemma D.4](#), \mathcal{G} can be written as $\mathcal{S} \langle i \rangle \langle x_1, z_1, \dots, x_K, z_K \rangle$, where the subgroup $\langle x_1, z_1, \dots, x_K, z_K \rangle \leq \mathcal{G}$ isomorphic to \mathcal{P}_k° is the group of logical operators for the gauge qubits. When we measure one of these logical operators, we are then said to be fixing the overall system into an eigenstate of one of these gauge qubits. Hence: *gauge fixing*. If we instead view these measurements of generators of \mathcal{G}' as implementing an associated ISG code for \mathcal{G} , gauge fixing is just another way of saying 'tracking the ISG \mathcal{S}_t over time'. Note that any associated ISG code will potentially only gauge fix a subset of the gauge qubits. This would then lead to a higher number of logical qubits in the ISG code compared to the subsystem code, a distinctive feature of Floquet codes [\[HH21, DTB22, DSTE23\]](#).

E Double hexagon code

We give here a little more detail on the double hexagon code of [Section 3](#), showing that it still has distance two, like the $\llbracket 4, 2, 2 \rrbracket$ code that it Floquetifies. In [Figures 7 and 8](#), we show operating regions corresponding to logical operators \bar{x}_1 and \bar{z}_1 respectively. Recall ([Subsection 2.2.2](#)) that given an operating region, the corresponding logical operator is found by looking at which output legs are highlighted with which colours. For example, in the rightmost diagram of [Figure 7](#), we show seven timesteps $t \in \{0, 1, \dots, 6\}$ of the period-six double hexagon code. So looking at the output wires of this diagram tells us a representative of the logical \bar{x}_1 at timestep $t = 6$. Namely, re-using the notation that assigns label (i, j) to the qubit with the i -th colour and j -th style, where i and j are taken modulo 6 and 2 respectively, and the ordered lists of colours and styles are **[purple, pink, orange, yellow, brown, blue]** and **[solid, dashed]**, we see the corresponding logical operator at this timestep is $X_{(0,0)}X_{(0,1)}X_{(5,0)}X_{(5,1)}$.

To see the corresponding logical operator at a different timestep t , we can just truncate the diagram after timestep t and again look at the highlighted output edges. This is shown for timesteps $t \in \{0, 1, 2\}$ in [Figure 9](#) below; the corresponding operators are $X_{(0,0)}X_{(0,1)}X_{(5,0)}X_{(5,1)}$ after timesteps $t = 0$ and $t = 1$, and $X_{(2,0)}X_{(2,1)}X_{(1,0)}X_{(1,1)}$ after timestep $t = 2$. In fact, truncating the rightmost diagram of [Figure 7](#) after *any* timestep t gives an operating region whose corresponding \bar{x}_1 representative has weight four. The same is true for all possible truncations of the \bar{z}_1 operating region in [Figure 8](#), and would also be true if we repeated this process for the other logical operator cosets \bar{x}_2 and \bar{z}_2 . That is, truncating after timestep t would always give us an operating region whose corresponding operator had weight four. It would be tempting to conclude, therefore, that the double hexagon code has distance four, and that Floquetifying the $\llbracket 4, 2, 2 \rrbracket$ code has increased its distance. But this would be wrong.

The reason is that while this process gives us logical operators in the Floquetification, it doesn't necessarily give us minimum-weight ones, even though we started with minimum-weight operating regions of the $\llbracket 4, 2, 2 \rrbracket$ code in the leftmost diagrams of [Figures 7 and 8](#). Indeed, one can find stabilizers at a given timestep t with which to multiply these logical operators, in order to reduce their weight to two. One can do this algebraically, making use of the fact that we already worked out the ISG \mathcal{S}_t of the double hexagon code for all t (see [Equation \(7\)](#) and the preceding sentence). Alternatively, one can do this graphically, using the fact that Pauli webs on the same diagram form a group and can thus be multiplied together [[MBG23](#), Appendix A.3]. Graphically, multiplying together two (unsigned CSS) Pauli webs just means laying one diagram on top of the other, up to the rule that if a wire is highlighted twice by the same colour, these cancel out and the wire becomes unhighlighted⁵. In [Figure 10](#) below, we thus multiply the weight-four operating regions by appropriate stabilizing regions to get weight-two operating regions.

Since this code is fairly small, one can use these ideas to check by hand that for all t , no logical operator at time t has weight one. Hence, since representatives of weight two exist for the logical operator coset \bar{x}_1 , the code has distance two, as promised.

⁵If laying one diagram on top of another results in a wire being highlighted both red and green, then the resulting Pauli web is no longer CSS, in the language of this paper. It is, however, a valid Pauli web according to the more general definition used in Refs. [[BLN⁺23](#), [MBG23](#)].

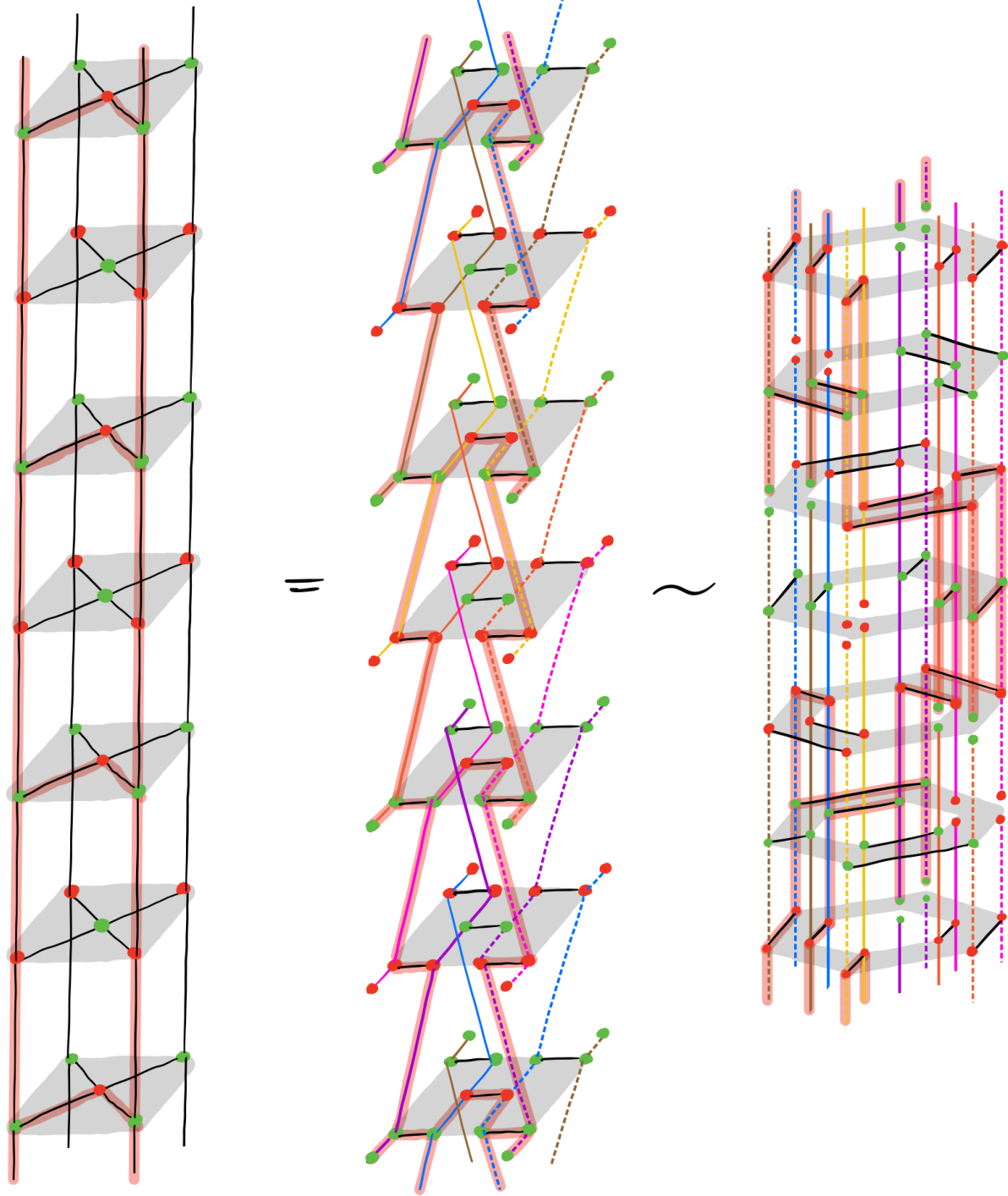


Figure 7: An operating region corresponding to a representative of the logical \bar{x}_1 in the $[[4, 2, 2]]$ code (left and middle) and in its Floquetification, the double hexagon code (right).

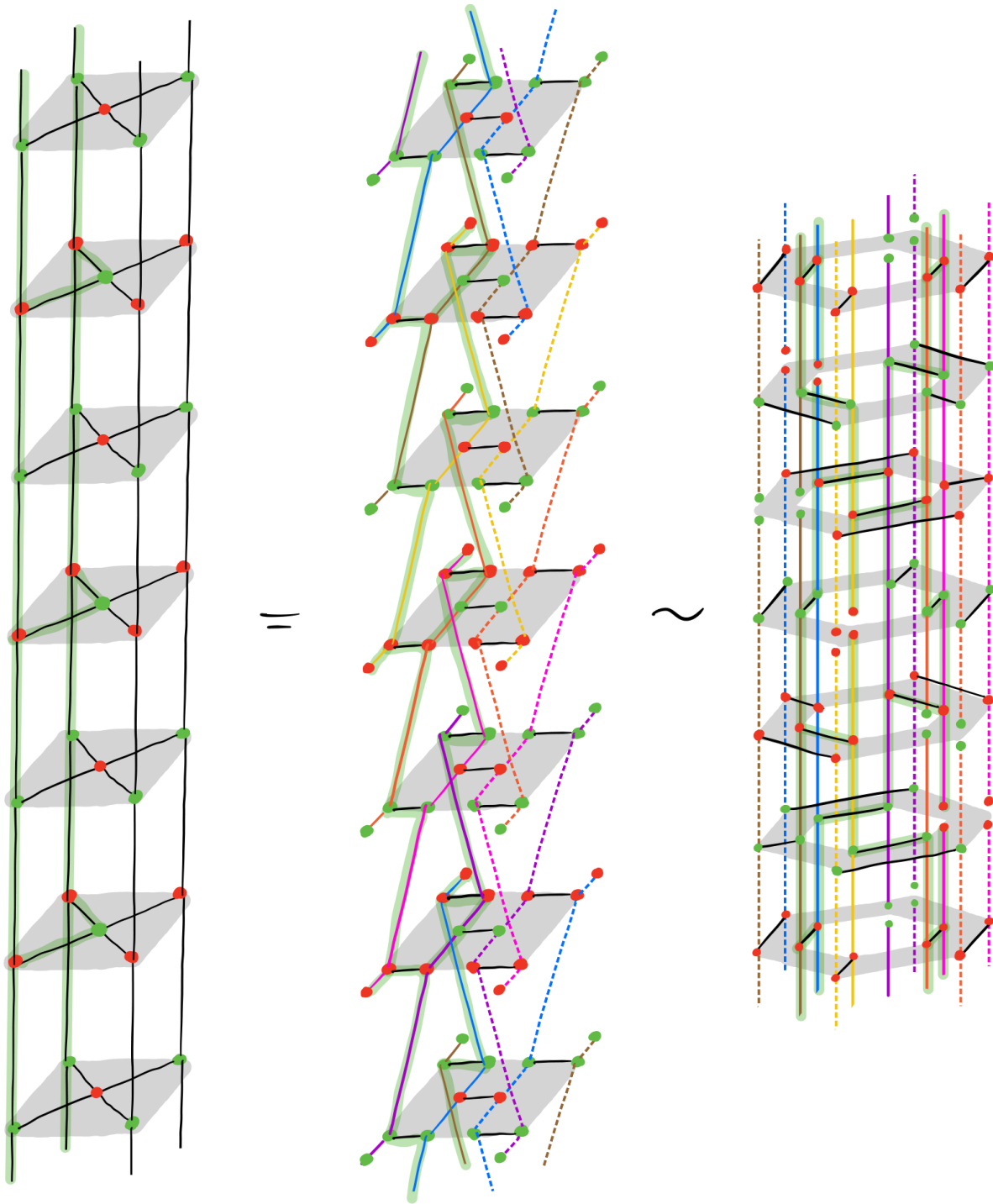


Figure 8: An operating region corresponding to a representative of the logical \bar{z}_1 in the $[[4, 2, 2]]$ code (left and middle) and in its Floquetification, the double hexagon code (right).

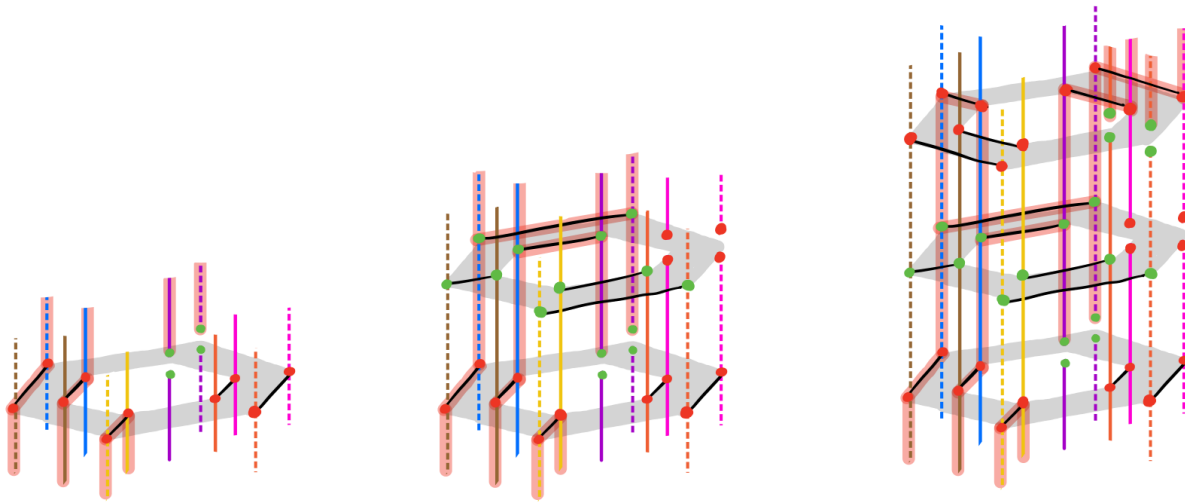


Figure 9: The double hexagon code after timesteps $t = 0, t = 1$ and $t = 2$ respectively, annotated with an operating region corresponding to a representative of the logical \bar{x}_1 .

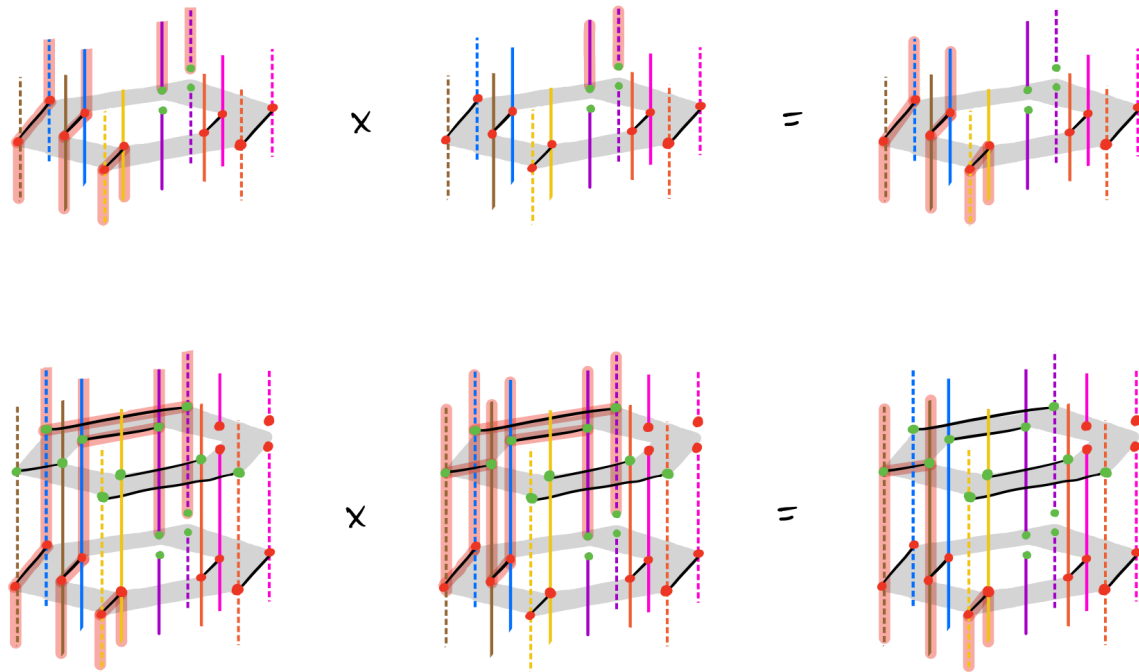


Figure 10: For timesteps $t = 0$ and $t = 1$ respectively, we multiply the weight-four operating regions of Figure 9 by appropriate stabilizing regions, which gives us weight-two operating regions. This corresponds exactly to composition of the corresponding Pauli operators - for example, the topmost graphical equation corresponds to the algebraic equation $X_{(0,0)}X_{(0,1)}X_{(5,0)}X_{(5,1)} \circ X_{(0,0)}X_{(0,1)} = X_{(5,0)}X_{(5,1)}$.

F Floquetifying the colour code: beyond the bulk

In this section, we expand on what we mean by a code exhibiting *drift*, and sketch some ideas as to how it might be avoided. Then we describe what goes wrong when attempting to Floquetify a non-planar code, before turning our attention back to our specific example of the colour code.

F.1 Drift

In a drifting Floquet code, at any timestep a new qubit might need to be initialised and entangled into the code, or an old qubit might be measured out and not used again. Were such a code to be implemented on hardware that required qubits to be in fixed positions (e.g. on a superconducting chip), the code might appear to slowly drift across the hardware in some direction. For example, if the qubits of the code roughly form a square, and qubits on the top boundary of this square are being regularly measured out, while on the bottom boundary new qubits are regularly being entangled in, then the code as a whole will seem to drift downwards. Viewing our Floquetification procedure as tilting the time direction in a ZX-diagram shows that drift is a natural consequence - see the left hand side of [Figure 11](#).

There are many ways we can try to avoid drifting boundaries. In fact, we saw one of them already, when we Floquetified the $[[4, 2, 2]]$ code in [Section 3](#). Ordinarily, the resulting code would exhibit drift, in exactly the fashion visualised on the left hand side of [Figure 11](#). However, our Floquetification re-used

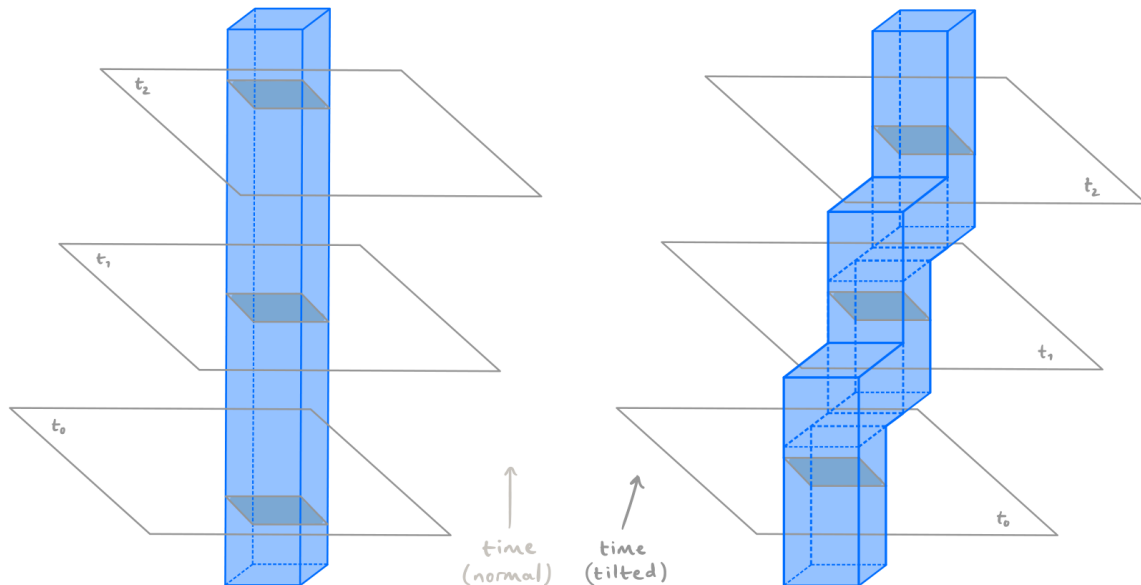


Figure 11: Here blue prisms represent ZX-diagrams of codes, while grey planes are timeslices $t_0 < t_1 < t_2$ after the time direction has been tilted. The shaded grey squares show the intersection of the code's ZX-diagram with a particular timestep. On the left, we see that taking a stabilizer code and tilting the time direction to get an equivalent Floquet code naturally causes drift - notice how the shaded grey intersections change position within the timeslices t_0 , t_1 and t_2 . On the right, we show that one way to avoid drifting boundaries is to Floquetify a code that is itself moving; here the grey intersections are always in the same place within the timeslices t_0 , t_1 and t_2 .

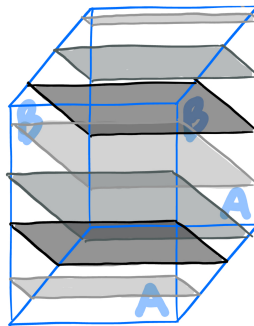
qubits cyclically; once the pair of qubits on the top boundary of the code were measured out, they were immediately re-entangled into the bottom boundary of the code at the next timestep - see the rightmost diagram of [Figure 2](#). In this sense, we made our code drift around in a circle, which gives it its double hexagon shape. While this works nicely for the very small $[[4, 2, 2]]$ code, it's not a good general solution from a practical perspective, since for hardware with qubits in fixed positions, an unusual connectivity would be required in order to implement a large code that drifts circularly. (For other types of hardware, where qubits can be moved around and re-used, this might be less of a problem).

Another potential solution is to Floquetify a code that is already moving, so as to cancel out the drift that results from Floquetifying it. For example, the rotated surface code can be grown outwards from one boundary (the top one, say), then contracted from the opposite one (the bottom), such that it appears to move in a particular direction (upwards) - see, for example, Ref. [[Lit19](#), Figure 40(c)]. The same idea can be used to move other stabilizer and subsystem codes. If such a code can be moved in an appropriate direction and at an appropriate speed, the resulting Floquetification will remain put; this is sketched abstractly on the right hand side of [Figure 11](#).

Other systematic ways one might hope to obtain non-drifting boundaries involve changing the measurement schedule. Suppose a drifting Floquetification has schedule $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_{\ell-1}]$. Consider a new Floquet code with schedule $\mathcal{M} = [\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_{\ell-1}, \mathcal{M}_{\ell-1}, \mathcal{M}_{\ell-2}, \dots, \mathcal{M}_0]$. One could hope that this new code drifts in one direction in timesteps $0 \leq t < \ell$, but then drifts in exactly the opposite direction in timesteps $\ell \leq t < 2\ell$, and so overall stays within a bounded region, while still having the required error correcting properties.

F.2 Periodic boundaries

Recall once again that we can interpret our Floquetification procedure as tilting the time direction in a ZX-diagram. If we attempt to do this for a code with periodic boundary conditions, this tilting can have the effect of making time run circularly. The diagram below illustrates the problem: a ‘thickened’ torus $T^2 \times I$ is shown in blue (faces of the cube marked with the same letters *A* and *B* are glued together), and three planes through it are drawn in different shades of grey. The thickened torus is an abstraction for a ZX-diagram showing the time evolution of a code that lives on a torus, while the three grey planes are potential new timeslices after Floquetifying this code. However, it's impossible to label these three planes with unique integers such that a line perpendicular to them passes through them an increasing sequence:



(29)

A possible alternative strategy in the case that we're given a code that lives on a torus is the following: one can first try to Floquetify a patch of the bulk, by reinterpreting which wires in the code's ZX-diagram represent qubit world-lines. After doing this for as large a portion of the bulk as needed, one can then

try to impose periodic boundary conditions on this Floquetified bulk patch, and see whether the resulting ZX-diagram encodes the expected logical qubits.

F.3 The colour code

Having discussed drift and periodic boundaries in general, we now discuss strategies for completing our Floquetification of the colour code bulk from the main text. Option one is to Floquetify a planar colour code, which will give us a Floquet code with drifting boundaries. We show this explicitly in Figures 12 to 14 below, for a small colour code that lives on a parallelogram, but the idea extends to a planar colour code of any size.

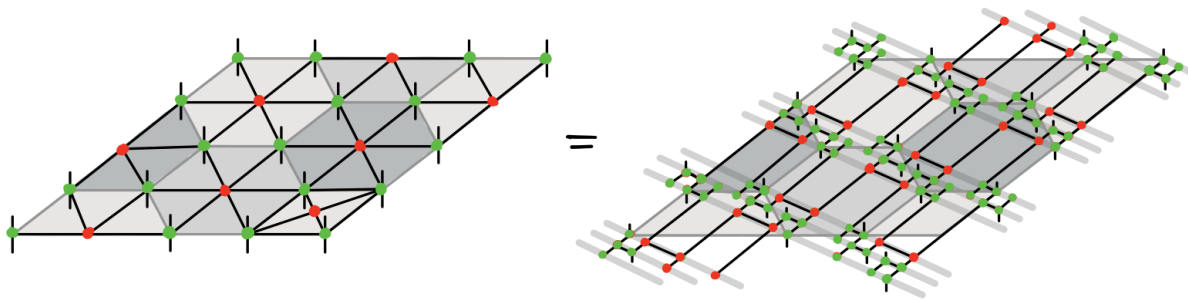


Figure 12: Two equivalent ZX-diagrams for one round of a small planar colour code on a parallelogram, in which all $Z \otimes Z \otimes \dots \otimes Z$ measurements are performed. As in Figure 4, the grey highlighted bars have no meaning in the ZX-calculus, and instead denote timesteps of the corresponding Floquetified code.

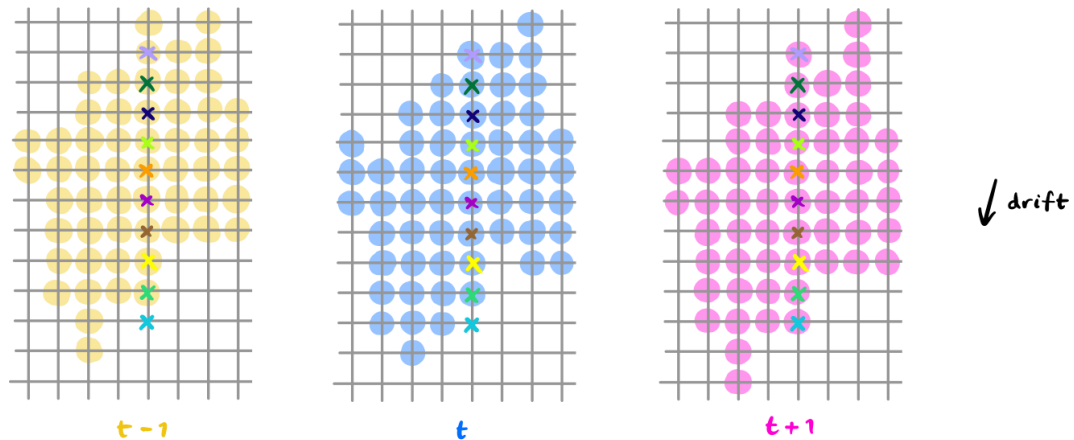


Figure 13: The square lattice layout of the Floquetified planar colour code, with qubits on vertices, at consecutive timesteps $t - 1$, t and $t + 1$. Vertices marked with a coloured cross correspond to qubit world-lines of the same colour in Figure 14. Vertices highlighted yellow, blue or pink are *active* at the respective timestep $t - 1$, t or $t + 1$. Here *active* means the qubit is initialised for the first time on or before this timestep, and is measured out for the last time on or after this timestep. If we were to plot the active qubits across all timesteps, we would see that the code drifts downwards and slightly to the left.

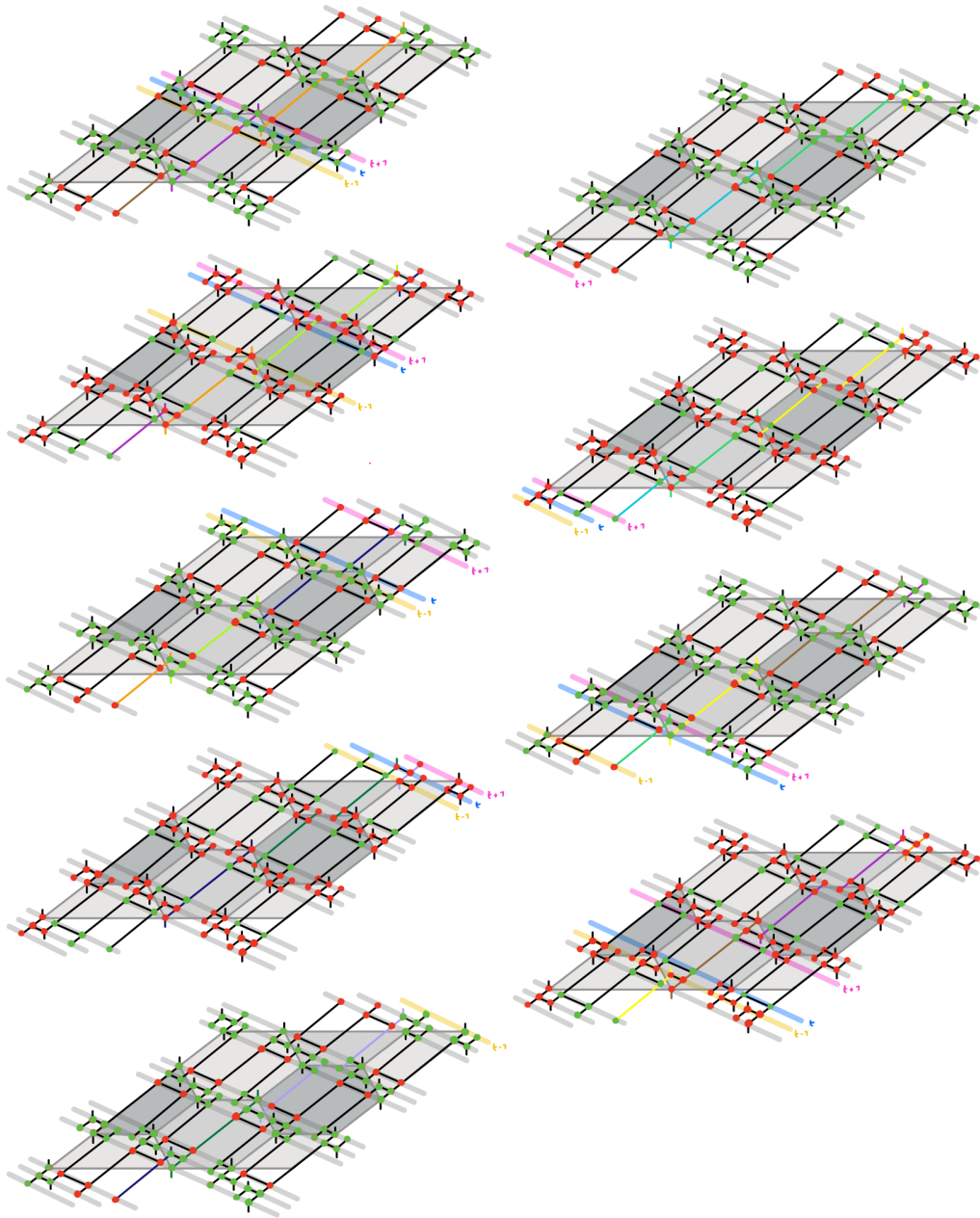
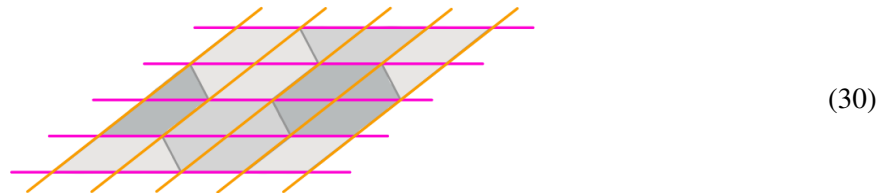


Figure 14: ZX-diagram of nine rounds of the planar colour code. This figure should be read columnwise up the page; first the left column, then the right. Much like in [Figure 4](#), vertical ZX-wires between consecutive layers have been omitted for clarity. Instead we state in words that, for a vertex v of the underlying honeycomb lattice, the small vertical upwards wire closest to v going upwards from one layer is connected to the small vertical wire closest to v protruding downwards from the next layer. Certain wires have been coloured - these indicate the world-lines of different qubits. Specifically, these are the world-lines of qubits that in a single column on the square lattice of the Floquetified code, as denoted by coloured crosses in [Figure 13](#). Grey highlighted bars again denote timesteps of the Floquetified code; three of them have been coloured yellow, blue and pink respectively, denoting three consecutive timesteps $t - 1$, t and $t + 1$.

The sequence of measurements that a particular qubit undergoes will be almost the same as that described in [Section 4](#), with the exception that qubits near a boundary can be measured out earlier or initialised later than usual. Additionally, as a consequence, a pairwise measurement that would ordinarily have been performed is omitted if one of the two qubits it would have applied to has been measured out early or initialised late. Detectors and logicals can again be found via Pauli webs, by drawing them on the ZX-diagram for the planar colour code and seeing how they map to the Floquetification. In the bulk, they will look as described in [Section 4](#).

If we wish to prevent the Floquetified code from drifting, we could instead force our parallelogrammatic colour code to regularly expand outwards from its top and right boundaries, and contract inwards from its bottom and left boundaries, such that this cancels out the drift that results from Floquetifying it. The expansion is performed by preparing pairs of qubits in Bell states, then entangling them into the code by measuring the stabilizers of a bigger colour code. Similarly, contraction is done by measuring out pairs of qubits in the Bell basis. We must be a little careful here to maintain the code's timelike distance; when measuring the stabilizers of the new, bigger code during an expansion phase, we must repeatedly measure these stabilizers for d rounds, where d is the distance of the original code. Contraction, however, can be done instantaneously. The distance by which we need to expand is determined by how fast the world-lines of qubits in the Floquetified code move forwards and rightwards through the ZX-diagram for the underlying colour code, as in [Figure 14](#) or the right hand side of [Figure 4](#). Let *rows* and *columns* in the honeycomb lattice be defined as the pink and orange lines respectively in the following diagram:



Suppose we now have a parallelogrammatic colour code with distance d . The geometry of the parallelogram forces d to be even. Such a code occupies $\frac{3d}{2} - 1$ rows and $\frac{3d}{2} - 1$ columns in the honeycomb lattice. After every d rounds of stabilizer measurements, the code needs to have moved $4d$ rows forwards and d columns rightwards in the honeycomb lattice in order to cancel out the drift the Floquetification process would bring. It's most convenient to expand and contract the colour code by multiples of 3 rows or columns, so let's also assume that d is an even multiple of 3. Thus, during an expansion phase, the code occupies $\frac{3d}{2} - 1 + 4d$ rows and $\frac{3d}{2} - 1 + d$ columns. The distance of the code during such a phase is therefore $\frac{11d}{3}$ in the forwards direction and $\frac{5d}{3}$ in the rightwards direction. We sketch this in [Figure 15](#).

The extra complication that this idea entails is that we now have also timelike boundaries to Floquetify. These are points at which pairs of qubits either become Bell states and get entangled into the code, or get measured out in the Bell basis. We could just do the same thing in our Floquetified code, but while the pairs of qubits to be entangled/measured are all neighbours in the honeycomb lattice, they are not all neighbours in the square lattice of the Floquetified code. This is shown in [Figure 16a](#). Fortunately, as shown in that [Figure 16d](#), we can construct a circuit on groups of four qubits which achieves the same thing but respects the nearest neighbour connectivity of the square lattice.

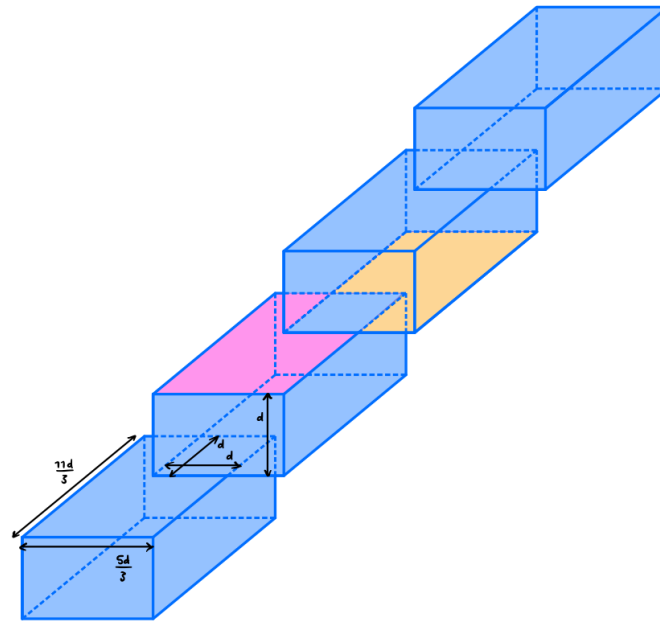


Figure 15: An abstraction of a ZX-diagram of a parallelogrammatic colour code which expands and contracts in the fashion specified in the text. In this figure, time goes directly upwards. The distance of the original colour code is denoted d . Two timelike boundaries are shaded; at the pink one, pairs of qubits are measured in the Bell basis, while at the orange one, Bell pairs are entangled into the code.

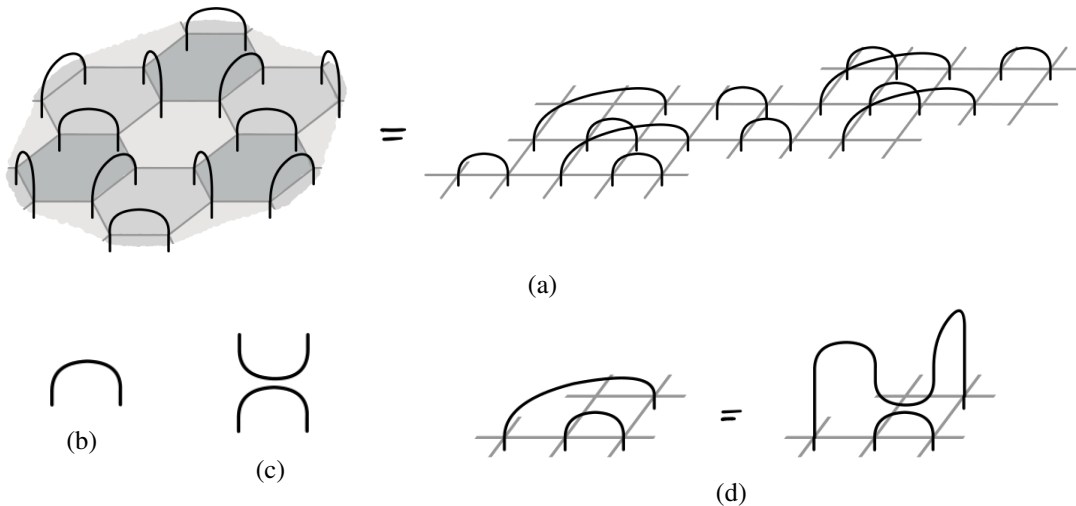


Figure 16: **(a)** The Bell basis measurements on a timelike boundary of the colour code, such as the pink one in [Figure 15](#), and their image in the square lattice Floquetification. **(b)** A destructive Bell basis measurement (or rather, post-selection) in the ZX-calculus. **(c)** A non-destructive Bell basis measurement (or rather, post-selection) in the ZX-calculus. **(d)** Two equivalent ZX-diagrams for performing a pair of Bell basis measurements on four qubits. Whereas the diagram on the left involves a measurement between two non-neighbouring qubits on the square lattice, the diagram on the right respects this connectivity constraint, but costs one extra Bell basis measurement.

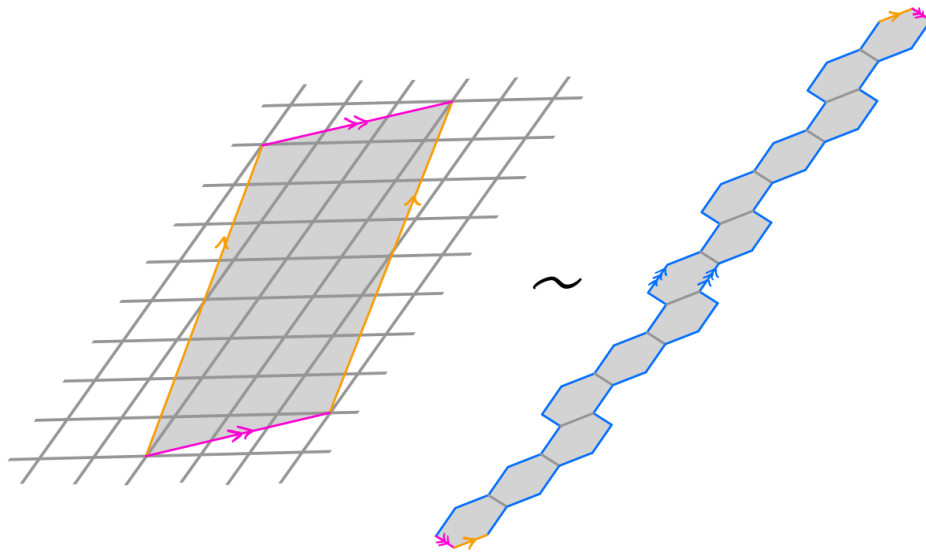


Figure 17: Imposing periodic boundary conditions on a single rectangle of the Floquetified colour code bulk, as in [Figure 5](#), defines a Floquet code with no logical qubits. As a patch of the Floquetified colour code bulk, this rectangle corresponds to the strip of colour code bulk on the right. Since this strip is not three-colourable, imposing toric boundary conditions on it prevents it from supporting a colour code.

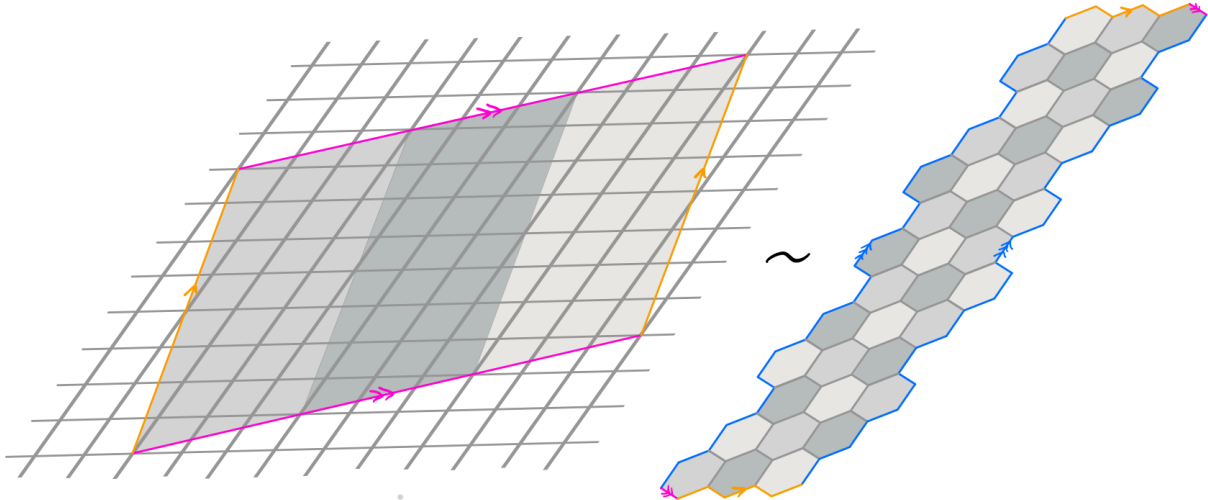


Figure 18: If we instead tile together a multiple of three columns of these rectangles, we define a Floquet code with four logical qubits - the same number as for the toric colour code. The left diagram of this figure is thus the support of the smallest member of our family of toric Floquetified colour codes. As a patch of the Floquetified colour code bulk, this corresponds to the wider strip of colour code bulk on the right. Unlike in [Figure 17](#), this strip is three-colourable, and thus supports a colour code when toric boundary conditions are imposed on it.

An alternative option is to impose periodic boundary conditions on a patch of Floquetified colour code bulk. On the right hand side of [Figure 5](#) we drew a grey rectangle, and said that by tiling copies of these rectangles across the square lattice, we can define the whole bulk of the Floquetified colour code. An obvious candidate, then, for the patch of bulk on which to impose a toric geometry, is this rectangle. But it turns out that we can't just tile the torus any way we like with these rectangles; in order to actually encode any logical information, we must choose a tiling in which the number of columns of such rectangles is a multiple of three. That is, if we choose a tiling in which this is not the case, then we find that the logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t$ becomes trivial after a full period of measurements. When we choose a tiling in which it is the case, on the other hand, we find that $N(\mathcal{S}_t)/\mathcal{S}_t \cong \mathcal{P}_4$ after establishment, just as for the toric colour code. One explanation for this restriction is the fact that the underlying colour code is three-coloured; indeed, analogously, one can't define a colour code on a toric honeycomb lattice that isn't three-colourable. We sketch this correspondence in [Figures 17](#) and [18](#).

G Measurement and post-selection

Throughout this paper, we've used post-selection instead of measurement in all of our ZX-diagrams. Here we justify this. ISG codes detect and correct errors using detectors, as defined in [Subsection 2.2.1](#). The logical information they encode is characterised by their logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t$. Suppose we draw the ZX-diagram for a given ISG code, and actually use measurement rather than post-selection. That is, we parametrise spiders representing measurements by $\frac{1-m}{2}\pi$, where m is the measurement outcome (we showed an example of this in [Equation \(2\)](#) for CSS measurements). The set of all detecting regions in this ZX-diagram corresponds exactly to the set of all detectors for the code. Likewise the set of all operating regions corresponds exactly to the logical Pauli group $N(\mathcal{S}_t)/\mathcal{S}_t$. Thus these two types of Pauli webs capture all of the error-correcting and information-encoding capabilities of an ISG code.

The admissible Pauli webs for a Pauli spider are independent of whether that spider's phase is 0 or π . Hence detecting regions and operating regions are independent of this too. In particular, they are unaffected by whether a spider representing a measurement has phase $\frac{1-1}{2}\pi = 0$ or $\frac{1-(-1)}{2}\pi = \pi$. Thus from an error correction perspective there is no loss of generality in assuming all the measurements have outcome $m = 1$, say - i.e. in using post-selection, rather than measurement. In other contexts, of course, this is not justifiable.

This is hinted at in Ref. [\[MBG23, Appendix A\]](#) and stated even more clearly in Ref. [\[BLN⁺23, III. Checks\]](#) in their own language. Craig Gidney has also tweeted about the idea of a 'Pauli-free' ZX-calculus [\[Gid23\]](#) suitable for use in error correction, in which a spider with phase α is equivalent to one with phase $\alpha + k\pi$, for any $k \in \mathbb{Z}$.

H 'Colour code' vs 'color code'

For fear of having their British passport revoked, one third of the authors of this paper insisted on using 'colour' instead of 'color' throughout. We apologise for any distress caused.

I 'Floquet': how to say it

It rhymes with 'okay'!