

Transdimensional quantum computation: the graphical novel

Lia Yeh


Oriel College
UNIVERSITY OF OXFORD

Department of Computer Science

This dissertation is submitted on 26th April 2025 for the degree of Doctor of Philosophy

Abstract

Transdimensional quantum computation: the graphical novel

Lia Yeh

This thesis lays down groundwork for researching towards heralding the age of useful quantum computers. It begins with background on *quantum graphical calculi*, which the work of this PhD has contributed to developing and utilizing. This leads into the first question this PhD research sought to answer: In what ways can quantum graphical calculi facilitate the discovery of new abstractions to close the gaps between algorithms and hardware, and theory and practice? We will focus on what I believe are two of the most promising and fundamental research avenues in this vein. One is *quantum error correction*, centering on the necessary protocols for quantum algorithms to be realizable on error-prone quantum devices. Building on a fully graphical reformulation of error correcting codes, we introduce a new normal form for Clifford isometries and show its correspondence to stabilizer tableaus. The other is *qudit* computation, the generalization of qubits (2-level quantum systems) to d-level quantum systems, bringing new possibilities for quantum information processing. To this end, we study the interesting properties of two novel qudit gate sets. One of these gate sets we show is the canonical generalization of the qubit Toffoli+Hadamard gate set to arbitrary prime qudit dimension. We further use these results to solve the multiqutrit exact synthesis problem, advancing the original solution for qubits by Giles and Selinger in 2013. In building upon the foundations in these areas, we seek to uncover paradigm shifts in how we process quantum information.

Thesis Contents

I Introduction **9**

1 Thesis Start **11**

1.1 On quantum graphical calculi 13

1.2 On reading this thesis and its contributions 14

1.3 Software Libraries 17

II Background **21**

2 The ZX-calculus **25**

2.1 Classical functions in quantum algorithms 34

3 Quantum information theory **41**

3.1 Stabilizer formalism 41

4 Quantum error correcting codes **49**

4.1 Classical error correcting codes 49

4.2 A bad quantum error correcting code 51

4.3 Stabilizer quantum error correcting codes 54

4.4 CSS codes with the phase-free ZX-calculus 58

5 About qudits **65**

5.1 Diagonal gates in the Clifford hierarchy 67

5.2 Qutrit Hamiltonians and Gell-Mann matrices 68

5.3 Rings 71

III Qubitland **77**

6 Computational representations of stabilizer theory **81**

6.1 Quantum error correcting codes 81

6.2 The Phase-free Z-X Generalized Parity Form (GPF) 81

6.3 The RREF_X Normal Form for Complete Stabilizer Tableau Rows 85

6.4 Circuit Extraction . . . 88
6.5 Transversal Gates and Magic State Distillation . . . 93
6.6 Sample: Code switching and morphing . . . 100

7 Isometries in the stabilizer fragment 103
7.1 Graph State with Local Cliffords Normal Form for Graph Codes . . . 104
7.2 Affine with Phases Normal Form for stabilizer codes . . . 106
7.3 The stabilizer tableau equivalent of AP normal form . . . 108
7.4 Adapting AP normal form to isometries . . . 110

IV Qutritland 119

8 The metaplectic gate set is a subset of Clifford+T 125
8.1 The qutrit Clifford+T gates . . . 127
8.2 Finding the metaplectic gate in qutrit Clifford+T . . . 132

9 Controlling all of qutrit Clifford+T 137
9.1 Controlled unitaries . . . 138
9.2 Adding controls to Clifford+T gates . . . 141
9.3 Building multiple-controlled Clifford+T gates . . . 145

10 Building diagonal gates with phase gadgets 151
10.1 The qutrit ZX-calculus . . . 152
10.2 Constructing phase gadgets, controlled phase gates, and phase multipliers . . . 156
10.3 Emulating diagonal qubit unitaries with qutrit phase gadgets . . . 165
10.4 Sample: Superconducting implementation of qutrit phase multipliers . . . 167
10.5 Sample: Emulating qubits with qutrits . . . 172

V Quditland 189

11 Scaling W state circuits in the Clifford hierarchy 193
11.1 The $\sqrt[d]{Z}$ gate . . . 196
11.2 A fault-tolerant circuit for the $|0\rangle$-controlled prime-dimensional qudit X gate . . . 197
11.3 The d-qubit W state . . . 199
11.4 The controlled H gate with phase gadgets . . . 201
11.5 Building any size W states . . . 205

12 Qudit Toffoli+Hadamard and the ZH-calculus 211
12.1 All odd prime-dimensional classical reversible gates can be built

with just the $|0\rangle$-controlled X gate . . . 214
12.2 The qudit Toffoli+Hadamard gate set . . . 220
12.3 The qudit ZH-Calculus . . . 221
12.4 Translating ZH-diagrams to ZX-diagrams . . . 225

13 Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits 229
13.1 Multiqutrit Exact Synthesis . . . 232
13.2 Gate Set Equivalences . . . 236
13.3 Rings and Groups . . . 240
13.4 Generators for $\mathrm{U}_n(\mathbb{Z}[1/3, \omega])$ . . . 243
13.5 Circuit Representations for the Elements of $\mathcal{S}_{3^n}$ . . . 245
13.6 Exact Synthesis of Toffoli+Hadamard Circuits . . . 250
13.7 Catalytic Embeddings . . . 251
13.8 Exact Synthesis of Clifford-Cyclotomic Circuits . . . 254
13.9 Circuit Complexity . . . 255

VI Thesis End 259

14 What didn't make it into the thesis 261

Bibliography 262

Preface

At the start of this PhD, I'd wanted to work on qudit algorithms, but then realized that even the basic building blocks that could be taken for granted for qubits since before I was born wasn't there: how to build qudit multiple-controlled Toffolis efficiently and fault-tolerantly. In my second year of this PhD, I had struggled with writing the qualifying dissertation, which is the first main milestone of the degree requirements of this PhD program. At that time I had only many questions and little to no answers. It turns out that in retrospect, those questions were onto something. If I were to go back in time and tell myself to buckle up and enjoy the ride, that I would go on to publish eight papers each answering a different question raised in that qualifying dissertation about qudit computation, I would have laughed in disbelief. But as I sit here and mash my keyboard, I think, who knows what the future holds.

Welcome to this thesis.

Part I

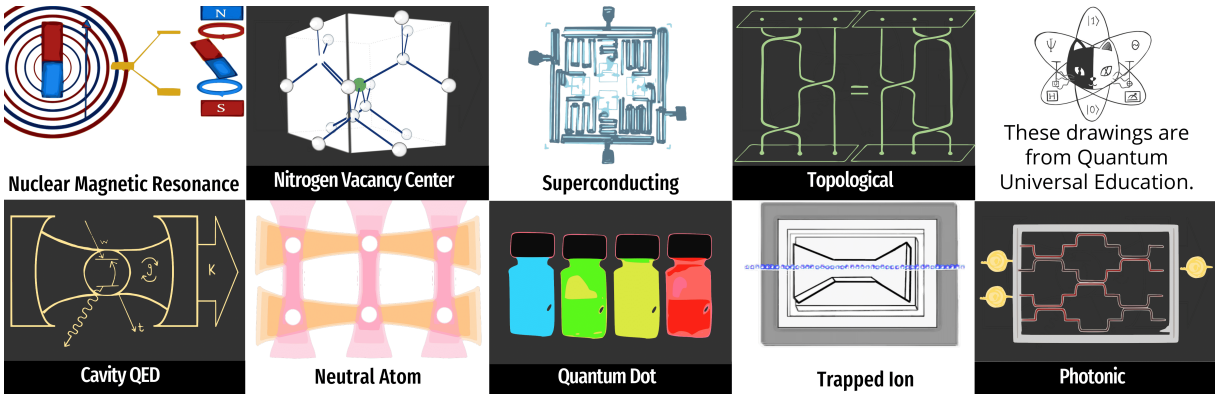
Introduction

1

Thesis Start

Quantum computation was conceptualized in the 70's and 80's to tackle impactful problems far beyond the capabilities of present-day computers, in health sciences, chemical engineering, atmospheric sciences, materials science, astrophysics, communications, … and more possibilities to discover. It is said we are now in the formative years of the second quantum revolution, where unlike the first a century ago with the theorization and experimental agreements with quantum mechanics, quantum devices can now precisely control individual atoms, electrons, and photons to do certain computations contesting our most powerful supercomputers. As we anticipate these awesome new technologies to take flight well within our lifetimes, we also risk the double-edged sword where scientists and ordinary people alike lack a foothold for comprehending what the overall tech is doing or being used for.

One view is that Quantum Processing Units will be special-purpose devices, as GPUs were for graphics, for problems with underlying mathematical structure suited to quantum information. Another view is that quantum will be integral into some next big AI thing. At this stage, no one knows exactly what a large enough future quantum computer will look like, or the best way to program them. We can already see that there are a lot of different types of quantum computing hardware platforms, including but not limited to:



These each have different physical operations, error models, or even different quantum computational models.

Faced with an entirely new model of computation, to realize quantum computation will require us to overcome many challenges along the way. We must continually improve and reinvent the modules and layers that will take us from quantum programs, to logical quantum gates and subroutines, all the way down to the many physical qubits that encode each logical qubit. I strongly believe that it is not merely a goal worthy of many PhD theses, but a matter of both importance and relevance to all quantum computing researchers, to think critically about the types of abstractions with the agility to act as a common language between the different layers of the full quantum computing stack. To accelerate quantum computing developments, we should be flexible and consider many models and architectures beyond qubits, the circuit model, and the same old gate sets. Just as classical computing outgrew programming in binary, quantum computing is in the era of inventing new computational abstractions aspiring to become expressive enough for a hybrid model of computation with both classical and quantum processing, with modularity and layering necessary for scalability. They should be verifiably correct, compatible with optimisation, and enable people from different backgrounds to communicate with one another.

This thesis is broadly about advancing and reconciling graph-, process-, and number-theoretic frameworks for quantum computation. With stabilizer theory, we can protect quantum information from faults by engineering error correcting codes and streamlining their implementations. With exact synthesis, we can figure out exactly what quantum computations are achievable with just a finite set of operations, so that we can focus on doing just a few operations very well to do any quantum computation. With higher-dimensional quantum information, we can store and compute more than just a two-dimensional system, breaking the abstraction of bits to enable more realistic and powerful capabilities. With graphical languages, we can work directly with the underlying mathematical representations and zoom in and out to different levels of computational abstraction.

1.1 On quantum graphical calculi

After the ZX-calculus was first formalized in 2007 [CD08], it was not until a decade later in 2017 that the first proof of *completeness* was achieved for the ZW-calculus — that is, a finite set of rules of the calculus are sufficient to prove any equality between linear maps in all of qubit quantum mechanics [Had17]. Completeness of the ZX- [HNW18; JPV18] and ZH- [BKM+23] calculi for qubit quantum mechanics have followed since, thus completing the trifecta of all non-trivial qubit graphical calculi[1] as classified in Ref. [CJ20]. In spite of there being infinitely many linear maps on qubits and infinitely many ways to implement any of them, Ref. [Vil19] reduced the complete qubit ZX-calculus ruleset to 8 rules which are each conceptually simple. This is crucial as there is mathematical structure inherent to quantum information that any good abstraction should leverage. Conversely, expressing gates in terms of their matrices is a bad abstraction not only because it is unwieldy, but because it is problematic for the soundness of quantum programs if they allow operations that are physically impossible on the quantum devices. The graphical calculi we use and develop are grounded in categorical quantum mechanics [AC04], from which the ability to reason using diagrams consisting of a few generators and wires between them arises from the underlying formalization of the interacting algebras.

Towards the objective of realizing practical quantum computation, quantum graphical calculi have been successfully applied to many areas across quantum information. To name a few:

1. Expressing and converting between different models of quantum computing — including the circuit model, measurement-based quantum computing [DP10; DKPvdW20; BMdF+21], the Pauli Fusion model [dBDHP20], linear optical quantum computing [dFC23; CHM+22], and fusion-based quantum computing [dFPC+26]
2. Reasoning about and translating between different gate sets, with ZX-calculus corresponding to Clifford + single-qubit phase gates [CD08], ZH-calculus to hypergraph states and circuits of Toffoli + Hadamard [BK19], and ZW-calculus to the GHZ and W algebras [CK10]
3. Optimization and classical simulation of quantum circuits [KvdWV22], with formal verification [LCR22]
4. Circuit simplification of hybrid quantum-classical circuits [BPV21]
5. Differentiation and integration of ZX diagrams, applied to the barren plateau problem in quantum machine learning [WYK24], and summation and exponentiation of Hamiltonians [SWY23]
6. Quantum error correction including lattice surgery [dBH20] and braiding [HEMN20] on surface codes, fault-tolerance [BLN+24], and coherent parity check codes [CKZ+23]

[1] To be precise, those whose structure consists of two Frobenius algebras interacting via a bialgebra rule.

1.2 Hitchhiker's guide to reading this thesis and its contributions

The organization of this thesis is quite standard. There is also the addition of *sample* (surprise) sections which are in the spirit of free samples at the grocery store, where you can try a bit (of some concept I researched during this PhD but opted not to include more details in this thesis). The section titles in this thesis's Table of Contents is intended to let you navigate to the topics you are (or aren't) looking for. Throughout the text, it mentions when there are related topics or concepts that are detailed later. This is to soften the learning curve and avoid getting distracted by every rabbit hole.

Following this introductory part, Part II contains background on topics prerequisite to the remainder of the thesis. In particular, for those new to the ZX-calculus, Chapter 2 is a small part of the intro to ZX-calculus tutorial materials I have refined over the years. The rest of the background chapter covers quantum information theory, quantum error correction, and qudits.

Part III is all about quantum error correcting codes in the qubit ZX-calculus. We present quantum error correction from a graphical perspective, drawing connections between stabilizer codes, graphs, projectors, tableaus, and encoder circuits, through normal forms in the ZX-calculus. The foremost contribution of this chapter is proof of a new graph-theoretic and circuit normal form for any Clifford isometry, which is moreover (up to logical operators) in one-to-one correspondence with the stabilizer tableau of the associated stabilizer code.

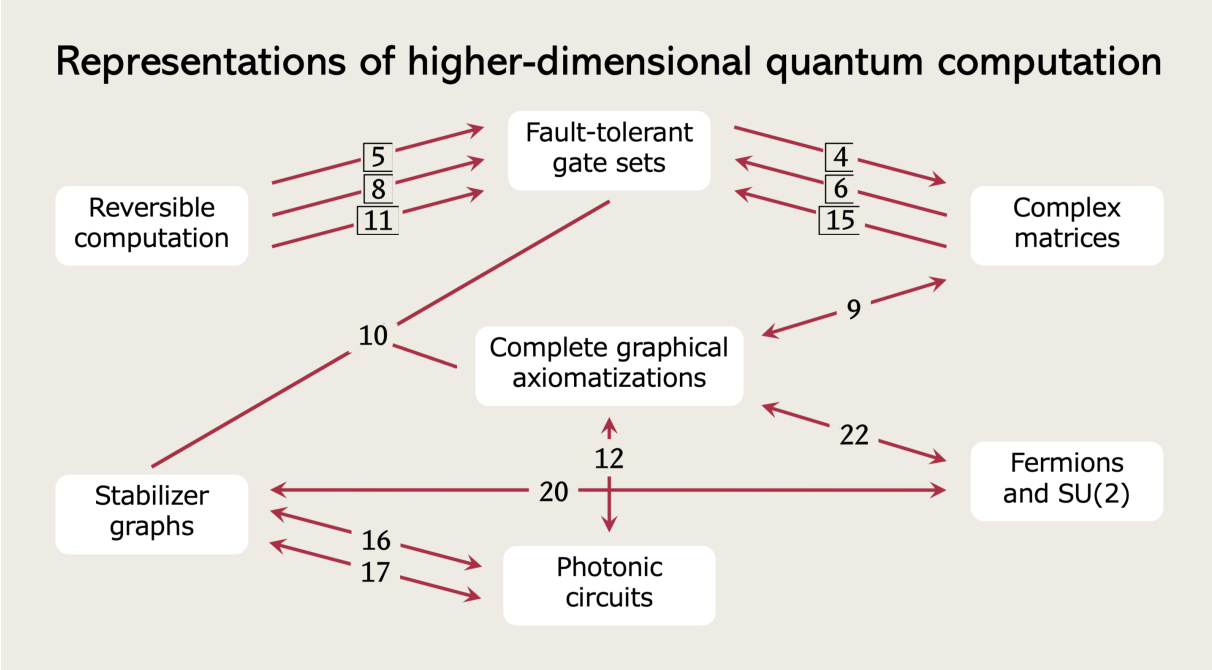
Part IV is the qutritlogy of three qutrit papers published in Summer 2022. They respectively dive into qutrit computation on three fronts: a number-theoretic approach leveraging exact circuit synthesis, an approach centering on ternary classical reversible circuits and controlled operators, and a phase gadgets (i.e. phase polynomials) approach to constructing diagonal gates. The foremost contributions of these sections are respectively: proving a strict subset relation between the two most well-studied fault-tolerant qutrit gate sets, constructing all possible controlled Clifford+T gates unitarily in the same gate set, and different ways to multiple-control phase gates and emulate arbitrary diagonal qubit unitaries with qutrits.

Part V is again about three papers, published in 2023 and 2024, going beyond qutrits to higher dimensions. The first two sections respectively introduce two novel prime-dimensional qudit gate sets, Clifford+$\sqrt[d]{Z}$ and Toffoli+Hadamard, with highly interesting properties. The former utilizes this to efficiently construct arbitrarily large qudit and emulated qubit W states. The latter moreover introduces the first qudit generalization of the ZH-calculus, and shows that the phase-free fragment corresponds precisely to circuits over the Toffoli+Hadamard gate set. The third and final section of the chapter and thesis, is about the first multiqudit exact synthesis

proof in any dimension higher than 2. Given a unitary matrix over a certain ring, we give an algorithm to exactly synthesize a circuit over the corresponding gate set in the multiqutrit Clifford-cyclotomic hierarchy of gate sets.

Overview of research during this PhD

In case you are looking for a specific paper(s), click on the *boxed* numbers below to jump to the related part in this thesis.



The chart above organizes the work done during my years of this PhD across seven representations of higher-dimensional quantum computing.

23 Christine Li and Lia Yeh. Transversal AND in Quantum Codes. arXiv:2603.04548

22 Quanlong Wang, Richard D. P. East, Razin A. Shaikh, Lia Yeh, Boldizsár Poór, and Bob Coecke. Beyond Penrose tensor diagrams with the ZX calculus: Applications to quantum computing, quantum machine learning, condensed matter physics, and quantum gravity. arXiv:2511.06012

21 Lia Yeh, Jiaxin Huang, Aleks Kissinger, Sarah Meng Li, and John van de Wetering. A Three-Way Normal Form for Stabiliser Codes across ZX Diagrams, Circuits, and Tableaus. Manuscript in preparation.

20 Haytham McDowall-Rose, Razin A. Shaikh, and Lia Yeh. From Fermions to Qubits: A ZX-Calculus Perspective. arXiv:2505.06212

19 Edwin Agnew, Lia Yeh, and Richie Yeung. Algebraic Structure of Quantum Controlled States and Operators. arXiv:2603.13454

18 Selma Dündar-Coecke, Caterina Puca, Lia Yeh, Muhammad Hamza Waseem, Emmanuel M. Pothos, Thomas Cervoni, Sieglinde M.-L. Pfaendler, Vincent Wang-Maścianica, Peter Sigrist,

Ferdi Tomassini, Vincent Anandraj, Ilyas Khan, Stefano Gogioso, Aleks Kissinger, and Bob Coecke. Making the quantum world accessible to young learners through Quantum Picturalism: An experimental study. arXiv:2504.01013

17 Giovanni de Felice, Boldizsár Poór, Lia Yeh, and William Cashman. Fusion and flow: formal protocols to reliably build photonic graph states. arXiv:2409.13541
and a substantially updated version: Giovanni de Felice, Boldizsár Poór, Cole Comfort, Lia Yeh, Mateusz Kupper, William Cashman, and Bob Coecke. A dataflow programming framework for linear optical distributed quantum computing. doi:10.22331/q-2026-01-19-1972

16 Razin A. Shaikh, Lia Yeh, and Stefano Gogioso. The Focked-up ZX Calculus: Picturing Continuous-Variable Quantum Computation. arXiv:2406.02905

15 Andrew Glaudell, Neil J. Ross, John van de Wetering, and Lia Yeh. Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits. In *Proceedings of the 21st International Conference on Quantum Physics and Logic* (July 2024). Buenos Aires, Argentina. doi:10.4204/EPTCS.406.2

14 Selma Dündar-Coecke, Lia Yeh, Caterina Puca, Sieglinde M.-L. Pfaendler, Muhammad Hamza Waseem, Thomas Cervoni, Stefano Gogioso, Aleks Kissinger, Bob Coecke. Quantum Picturalism: Learning Quantum Theory in High School. In *Proceedings of The 2nd Quantum Science and Engineering Education Conference*, co-located with *The 4th IEEE International Conference on Quantum Computing and Engineering* (September 2023). Bellevue, Washington, USA. doi:10.1109/QCE57702.2023.20321

13 Jiaxin Huang*, Sarah Meng Li*, Lia Yeh*, Aleks Kissinger, Michele Mosca, Michael Vasmer (*equal authorship). Graphical CSS Code Transformation Using ZX Calculus. In *Proceedings of the 20th International Conference on Quantum Physics and Logic* (July 2023). Paris, France. doi:10.4204/EPTCS.384.1

12 Giovanni de Felice, Razin Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang, Bob Coecke. Light-matter rewriting in the ZXW calculus. In *Proceedings of the 20th International Conference on Quantum Physics and Logic* (July 2023). Paris, France. doi:10.4204/EPTCS.384.2

11 Patrick Roy, John van de Wetering, Lia Yeh. The Qudit ZH-Calculus: Generalised Toffoli+Hadamard and Universality. In *Proceedings of the 20th International Conference on Quantum Physics and Logic* (July 2023). Paris, France. doi:10.4204/EPTCS.384.9

10 Boldizsár Poór, Robert I. Booth, Titouan Carette, John van de Wetering, Lia Yeh. The Qupit Stabiliser ZX-travaganza: Simplified Axioms, Normal Forms and Graph-Theoretic Simplification. In *Proceedings of the 20th International Conference on Quantum Physics and Logic* (July 2023). Paris, France. doi:10.4204/EPTCS.384.13

9 Boldizsár Poór, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung, Bob Coecke. Completeness for arbitrary finite dimensions of ZXW-calculus, a unifying calculus. In *Proceedings of the 38th Annual ACM/IEEE Symposium on Logic in Computer Science* (June 2023). doi:10.1109/LICS56636.2023.10175672

8 Lia Yeh. Scaling W State Circuits in the qudit Clifford Hierarchy. In *Proceedings of the 1st*

International Workshop on the Art, Science, and Engineering of Quantum Programming (March 2023). doi:10.1145/3594671.3594687

7 Alberto Maldonado-Romo, Lia Yeh. Quantum computing online workshops and hackathon for Spanish speakers: A case study. In *Proceedings of The 2nd Quantum Science and Engineering Education Conference*, co-located with *The 4th IEEE International Conference on Quantum Computing and Engineering* (September 2022). doi:10.1109/QCE53715.2022.00096

6 John van de Wetering, Lia Yeh. Building qutrit diagonal gates from phase gadgets. In *Proceedings of the 19th International Conference on Quantum Physics and Logic* (July 2022). doi:10.4204/EPTCS.394.4.

5 Lia Yeh, John van de Wetering. Constructing all Qutrit Controlled Clifford+T Gates in Clifford+T. In *Proceedings of the 14th International Conference on Reversible Computation* (July 2022). doi:10.1007/978-3-031-09005-9

4 Andrew N. Glaudell, Neil J. Ross, John van de Wetering, Lia Yeh. Qutrit metaplectic gates are a subset of Clifford+T. In *Proceedings of the 17th Conference on the Theory of Quantum Computation, Communication and Cryptography* (July 2022). doi:10.4230/LIPIcs.TQC.2022.12

3 Zeki C. Seskir, Piotr Migdał, Carrie Weidner, Aditya Anupam, Nicky Case, Noah Davis, Chiara Decaroli, İlke Ercan, Caterina Foti, Paweł Gora, Klementyna Jankiewicz, Brian R. La Cour, Jorge Yago Malo, Sabrina Maniscalco, Azad Naeemi, Laurentiu Nita, Nassim Parvin, Fabio Scafirimuto, Jacob F. Sherson, Elif Surer, James R. Wootton, Lia Yeh, Olga Zabello, Marilù Chiofalo. Quantum games and interactive tools for quantum technologies outreach and education. In *Optical Engineering* 61, 8 (July 2022). doi:10.1117/1.OE.61.8.081809

2 Razin A. Shaikh*, Lia Yeh*, Benjamin Rodatz*, Bob Coecke (*equal authorship). Composing Conversational Negation. In *Proceedings of the 4th Annual International Applied Category Theory Conference* (July 2021). Cambridge, UK. doi:10.4204/EPTCS.372.25

1 Benjamin Rodatz*, Razin A. Shaikh*, Lia Yeh* (*equal authorship). Conversational Negation using Worldly Context in Compositional Distributional Semantics. In *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, co-located with *the 14th International Conference on Computational Semantics* (June 2021). Association for Computational Linguistics, pp. 53-65. Groningen, The Netherlands. https://aclanthology.org/2021.semspace-1.6.pdf

Paper from before this PhD

Lia Yeh, Lincoln Satterthwaite, David Patterson. 2019. Automated, context-free assignment of asymmetric rotor microwave spectra. *The Journal of Chemical Physics* 150, 20 (May 2019), 204122. doi:10.1063/1.5085794

1.3 Software Libraries

1.3.1.0 Software accompanying this thesis

For five of the publications comprising this thesis, code implementation is available in the below three open-source repositories:

- https://github.com/lia-approves/qudit-circuits/tree/main [GRvdWY22a; YvdW22; RvdWY23]
- https://github.com/lia-approves/qutrit-Clifford-cyclotomic [GRvdWY24]
- https://github.com/jvdwetering/dizx/tree/main [PBC+23]

The first two repositories above verify correctness of key qutrit and qudit circuit constructions. The base functionality is essentially as a glorified circuits to matrices calculator. Perhaps the most helpful addition is a helper function designed for use for classical reversible circuits, which parses the resulting matrix and prints a truth table of the output computational basis state with coefficient, that each input computational basis state was mapped to.

The code for Ref. [RvdWY23] builds the circuits for the proof that $\{X, CX, CCX\}$ generates all d-ary classical reversible circuits. Arbitrary odd prime qudit dimension d can be specified as a parameter[2].

The second repository supplements our paper on multiqutrit exact circuit synthesis in the Clifford-cyclotomic gate set [GRvdWY24]. While not implementing the exact synthesis algorithm itself nor the catalytic embeddings, it constructs each level operator, which the final circuits from our exact synthesis algorithm are built out of.

The third repository develops *DiZX*, a software library to extend some of the core functionalities of PyZX to prime-dimensional qudit ZX-diagrams, through implementing the algorithms for prime-dimensional qudit stabilizer computation presented in Ref. [PBC+23].

1.3.2.0 Qubit ZX-calculus Software Libraries

It is helpful to overview a few current software libraries for programming with the qubit ZX-calculus. This is not a complete list, but a starting point to peruse and use.

- *Quantomatic* is a library for more general graphical rewriting. Although not maintained, it is predecessor to the below libraries. https://github.com/zxcalc/quantomatic [KZ15]
- The most popular library is *PyZX* (in Python), which in focusing on the qubit ZX-calculus, was significantly faster than Quantomatic. https://github.com/zxcalc/pyzx [KvdW20a]
- *QuiZX* (in Rust) is $\approx 100\times$ faster than PyZX. It is more specialized implementation, with fewer functionalities compared to PyZX. Two ZX-diagram simplifiers it implements are from Refs. [DKPvdW20] and [KvdW20b] respectively. https://github.com/zxcalc/quizx

[2]Unavoidably, too large choice of the dimension d will induce memory-related slowdown or fault. In this event, one option is to allocate additional memory to the computation at the start of the program.

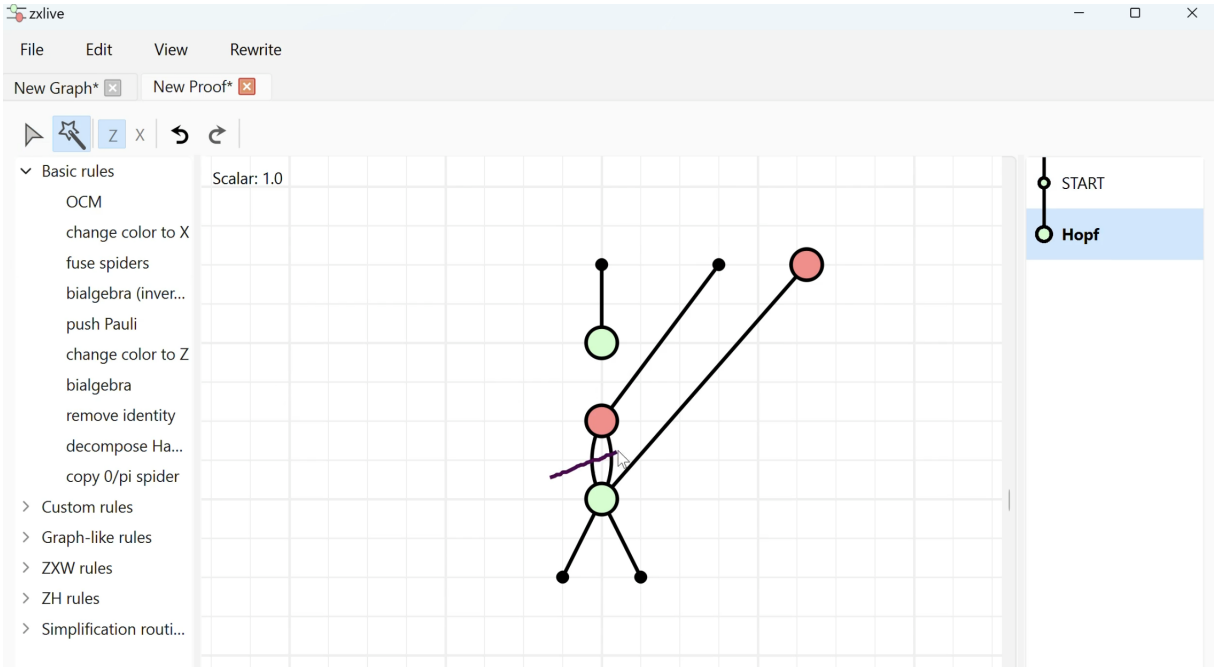


Figure 1.1: This is a screenshot of ZXLive. The ZX-diagram for the three mixed-state quantum measurement operators of the Stern-Gerlach experiment were drawn by drag-and-drop in the GUI application. Depicted here is the second application of the magic wand tool to apply the Hopf rule, which removes any pair of wires between different color spiders.

- *ParamZX* (in C++ & CUDA) is an even more specialized library for running on GPUs, which is up to 100× faster than QuiZX. https://github.com/mjsutcliffe99/ParamZX [SK25]
- *ZXLive* is a drag-and-drop GUI for drawing PyZX diagrams, and applying rewrite rules and algorithms with the click of a button or a wave of the magic wand tool. It has been the object of our affection for a number of mini-hackathons in recent years. https://github.com/zxcalc/zxlive
- *Stim ZX* lets you get the stabilizers of a ZX-calculus graph. It is part of the fast stabilizer circuit library *Stim* [Gid21]. https://github.com/quantumlib/Stim/tree/main/glue/zx
- *Tikzit* lets you typeset computer-drawn ZX-diagrams in a drag-and-drop GUI. You can copy or save the rendered image, or can input as Tikz files when you compile LaTeX. You can specify a style file to customize how nodes, bendy wires, and even shaded regions look—or you can start with a style file from the source of papers with ZX-diagrams on arXiv. Almost all of the diagrams in this thesis are typeset with Tikzit. https://tikzit.github.io/

In addition to contacting the maintainers, for questions and support for using these libraries, you can ask in the ZX Discord server linked at https://zxcalculus.com/.

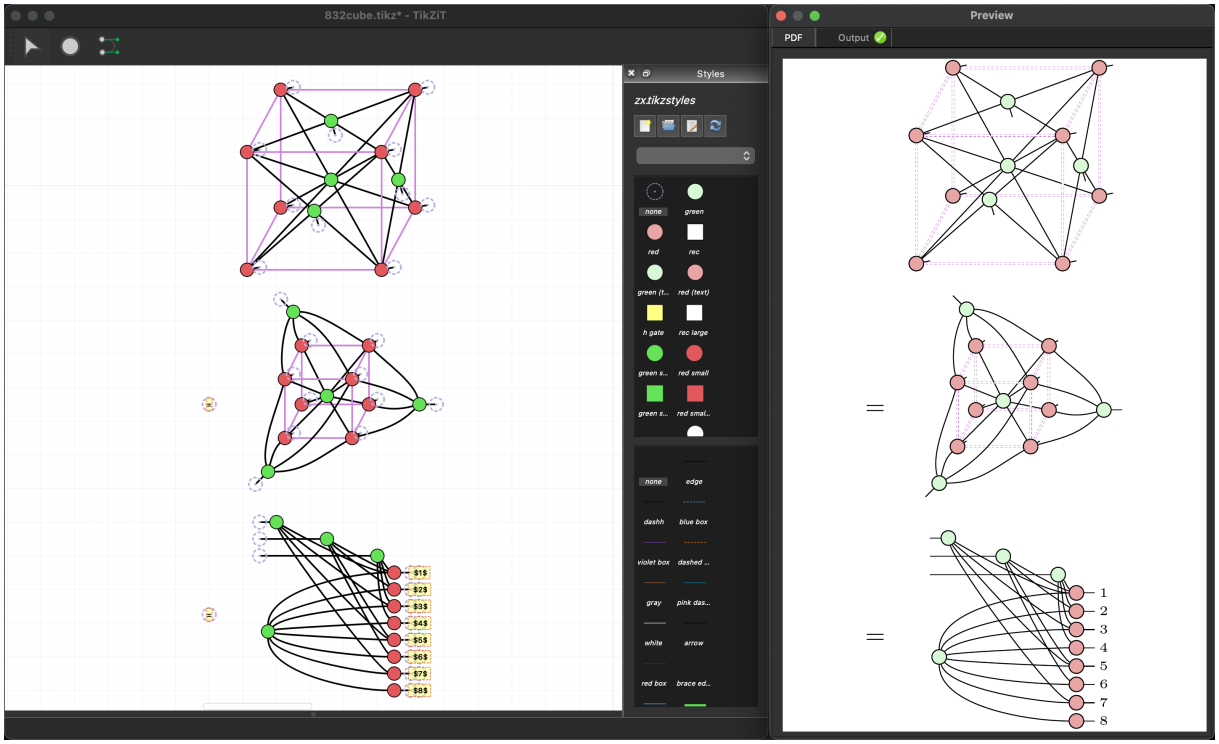


Figure 1.2: This is a screenshot of Tikzit. **Left:** The drag-and-drop GUI window to draw these ZX-diagrams for the encoder of the $[\![8, 3, 2]\!]$ quantum error correcting code. **Right:** That image, rendered by Tikzit according to the loaded .tikzstyles file.

Part II

Background

Chapter Contents

2 The ZX-calculus **25**
2.1 Classical functions in quantum algorithms 34

3 Quantum information theory **41**
3.1 Stabilizer formalism 41

4 Quantum error correcting codes **49**
4.1 Classical error correcting codes 49
4.2 A bad quantum error correcting code 51
4.3 Stabilizer quantum error correcting codes 54
4.4 CSS codes with the phase-free ZX-calculus 58

5 About qudits **65**
5.1 Diagonal gates in the Clifford hierarchy 67
5.2 Qutrit Hamiltonians and Gell-Mann matrices 68
5.3 Rings 71

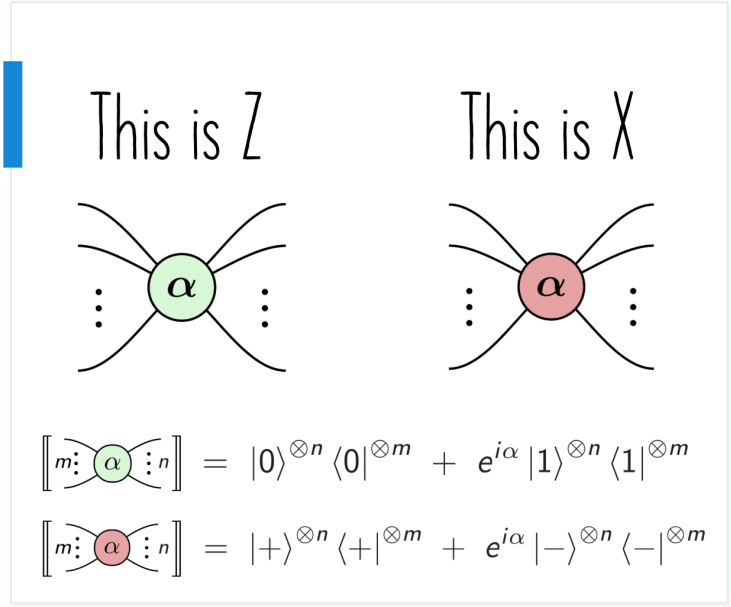
2

The ZX-calculus

a crash course

The goal of this section is to get through a speedy but practical introduction to how to use the ZX-calculus. Accordingly, rather than focusing on how the calculus was defined through category theory, the objective here is to cover a range of concepts in qubit quantum computing in this formalism, to make the later chapters of this thesis more accessible to readers who are not already familiar with the ZX-calculus. We assume familiarity with matrices, complex numbers, and bra-ket notation for translating these diagrams to the Hilbert space formalism, but this is not necessary to work with the ZX-calculus itself. In this section we use a variety of different formattings for qubit ZX-diagrams, reflecting the range of educational resources on the topic; with this noted, in the rest of this thesis we will be consistent with adhering to the defined symbols and conventions.

Every *diagram* in the ZX-calculus is built from the *generators*, the Z and X spiders:



There are also the *structural generators*, which enable wires to bend and cross over each other.

$$[\![|]\!] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad [\![\times]\!] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad [\![\cap]\!] = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad [\![\cup]\!] = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}$$

Each diagram has an *interpretation*[1] as a qubit linear map, given above for the Z and X spiders[2]. Composing diagrams in sequence gives matrix multiplication, whereas in parallel gives tensor product. The number of inputs and outputs of a diagram determine the shape of its matrix:

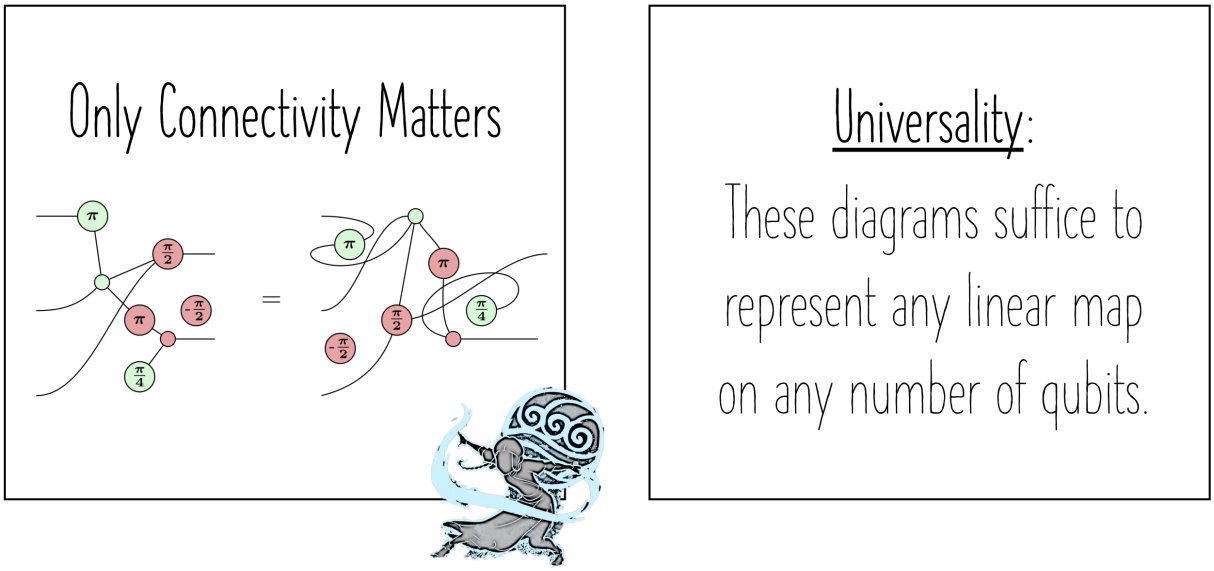
Inputs	Outputs	Interpretation	As a matrix
0	>0	State	Column vector
>0	0	Effect	Row vector
>0	>0	Map	Matrix
0	0	Scalar	Complex number

Calculi have *rules*, which indicate equality between any two diagrams that represent the same linear map. The ZX-calculus has the property of *soundness*, which means that any application of its rules necessarily implies equality of the underlying linear maps; in other words, you cannot prove anything incorrect under correct usage of the calculus. The simplest rule is *Only Connectivity Matters (OCM)*: the wires can move, bend, or cross freely, so long as how

[1] Strictly speaking, the interpretation is better denoted by a symbol other than equality, due to difference possible choices of how to interpret the semantics for a given calculus of syntactical rules. We use equal signs for interpretation in this thesis because we stick to a single interpretation for each diagram for which the calculus is sound, taking the qudit dimension(s) that could be interpreted into to be apparent from context.

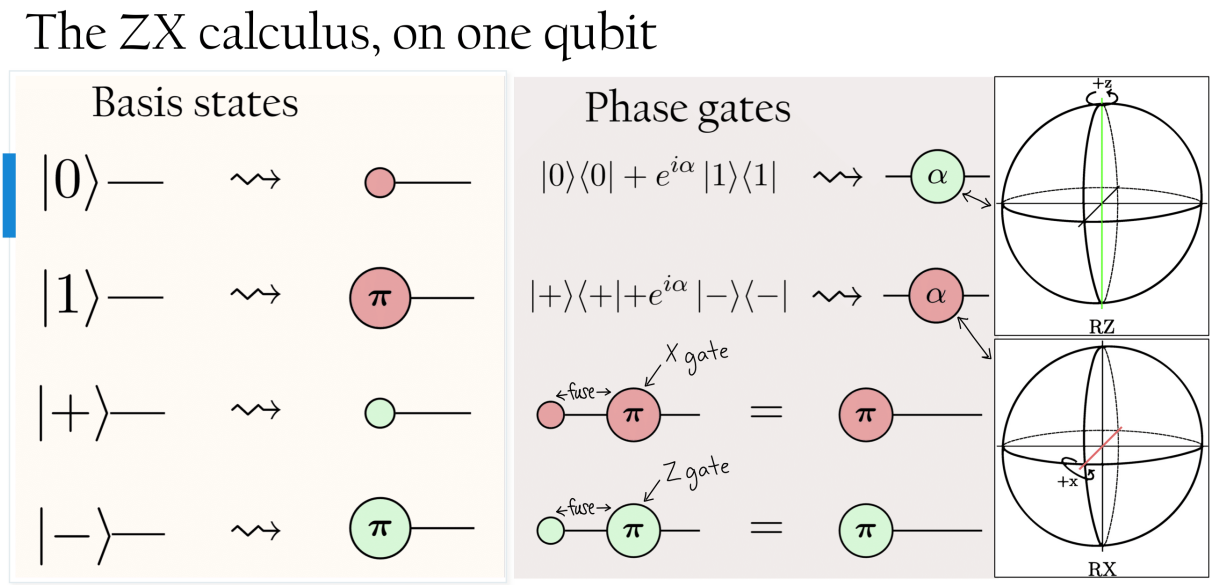
[2] Conventions differ up to a nonzero scalar factor. In practice, in most situations and everywhere in this thesis (with the exception of the Deutsch-Jozsa algorithm in this section), you may not need to keep track of nonzero scalar factors, in which case you can take the equational theory modulo by them, as done here.

everything is connected stays the same.



The ZX-calculus has the property of *universality* for all qubit linear maps: This means that for any linear map on qubits, there always exists a (non-unique) ZX diagram that interprets to it. One way to see this, is that any linear map can always be decomposed as a (possibly post-selected) quantum circuit over the universal gate set of $\{CX, H, Z(\alpha)\}$[3].

Below, we show some common single-qubit ZX diagrams: the Z-basis states (which are eigenvectors of the Z gate, but they are X spiders) $|0\rangle$ and $|1\rangle$, the X-basis states (eigenvectors of the X gate but are Z spiders) $|+\rangle$ and $|-\rangle$, and the single-qubit rotation gates $Z(\alpha)$ and $X(\alpha)$.

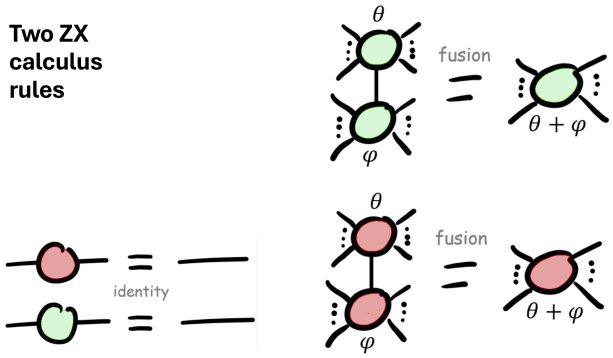


Any single-qubit gate is visualizable (isomorphic up to global phase) to a rotation of a 3D sphere; the Z and X rotation gates rotate by angle α about the Z and X axes of the sphere

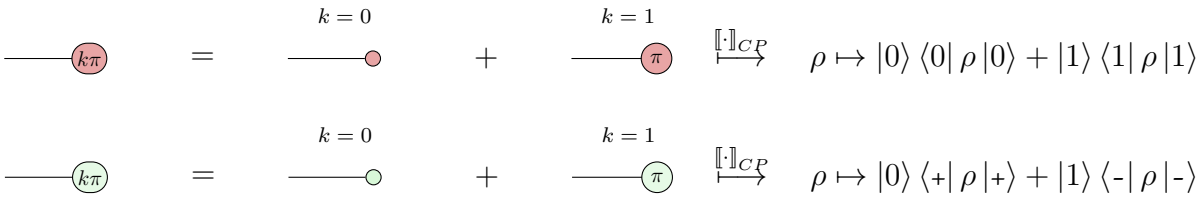
[3]In the next section, we will define these gates for arbitrary qudit dimension.

respectively. Rotating by an angle of 0 is the same as the identity operation, which is why by convention we omit spider labels of 0.

Applying the $X(\pi) = X$ gate to the $|0\rangle$ state gives the $|1\rangle$ state, and the $Z(\pi) = Z$ gate to $|+\rangle$ gives $|-\rangle$. These are single-qubit examples of the *Fusion* rule in the ZX-calculus: fusing two connected same-color spiders and adding their phases, preserves the linear map.



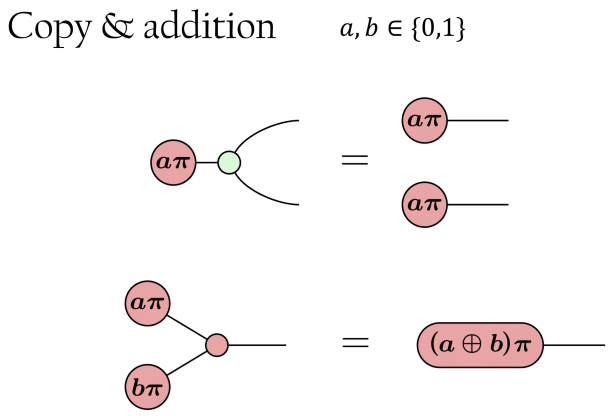
Destructively measuring a qubit is conventionally drawn as a diagram parametrized by a bit k that fixes the effect of the measurement with outcome k. We formally defined this notation in Ref. [dFPYC24] as an implicit sum over diagrams, one per each term in the Kraus decomposition of the measurement operator. For Z and X measurement respectively, the interpretation as a completely positive (CP) map is:



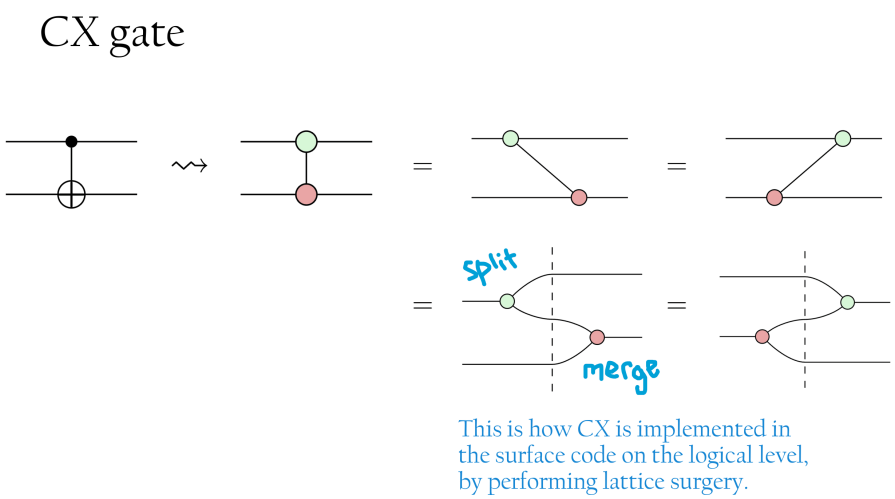
For measuring a qubit in any other basis/planes/angles, we can use the fact that any single-qubit measurement can be decomposed into U followed by Z measurement for some single-qubit unitary U. For Y-basis measurement, $U = Z(\frac{\pi}{2})$.

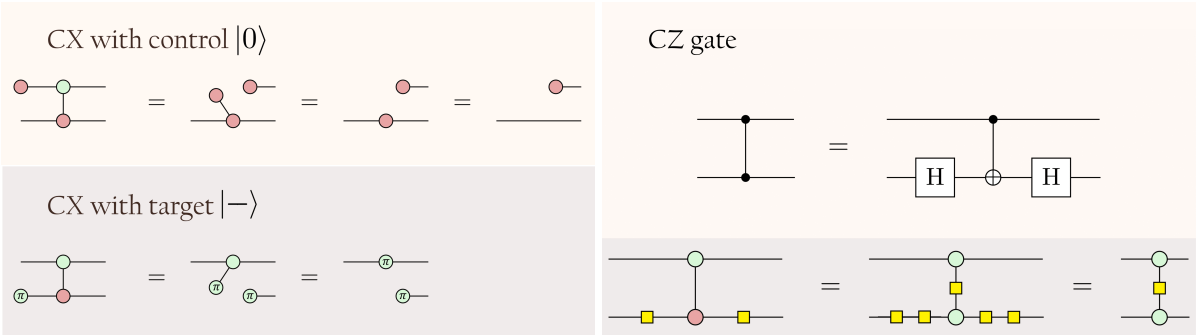
The Z and X spiders respectively perform copy[4] and addition modulo 2 on Z-basis states. Likewise, the X and Z spiders respectively perform copy and addition on X-basis states.

[4]This does not violate the no-cloning theorem, because quantum states in a known basis can be copied.



The CX gate in the ZX-calculus resembles how it looks in quantum circuits, but it is in two halves that each have a separate meaning[5]. The action of the CX gate when the input states are in the Z-basis is: the Z spider copies the control qubit, and the X spider adds this to the target qubit modulo 2. When the input states are in the X-basis, it is the X spider that copies the target qubit, and the Z spider that adds this to the control qubit.





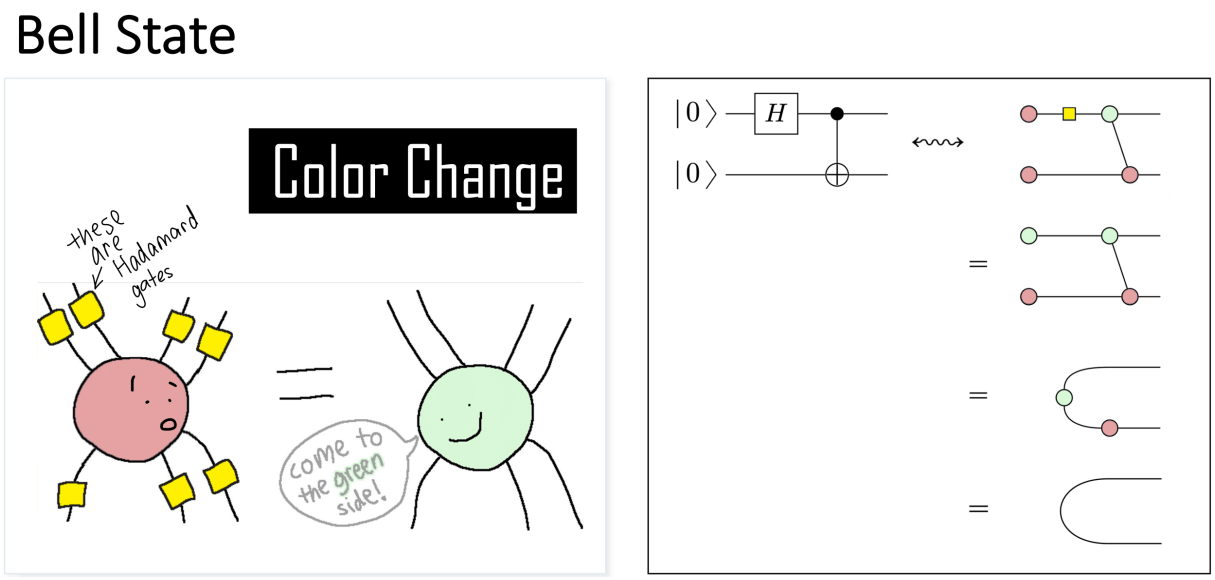
We can introduce another generator to the ZX-calculus called H, which is drawn as a square

[5]Some implementations of the CX gate do these halves separately, such as surface code lattice surgery [dBH20].

with 1 input and 1 output, and interprets to the Hadamard gate matrix:

$$\boxed{} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.1}$$

This is a *derived* generator in the ZX-calculus, because we can write H as a diagram made of Z and X spiders, as done on the next page in Figure 2.2. As the single-qubit gate that switches between the Z and X bases, applying H to every leg of a spider changes its color (without changing its phase label). An example is above, where by conjugating the target of a CX gate by H gates, we can see that the ZX diagram for the CZ gate is symmetric with respect to both qubits, and any Z phase gates can be commuted through it (by fusion then unfusion). Another example where this is applied is in the standard Bell state preparation circuit, of gates applied to two $|0\rangle$ states. This is depicted below; note that the H gate here is blue instead of yellow as that was the conventional color choice of the source material.



What we can see from the above derivation that the simplest ZX-calculus diagram for the Bell state $|0\rangle|0\rangle + |1\rangle|1\rangle$ is precisely a wire (which interprets to the single-qubit identity gate $I = |0\rangle\langle 0| + |1\rangle\langle 1|$), but with the input bent $180°$ into an output. The Bell state and Bell measurement are central to the textbook example of the ZX-calculus, quantum teleportation. Figure 2.1 shows proof in the ZX-calculus that after corrections for all four possible Bell measurement outcomes, the standard quantum teleportation circuit implements the identity linear map from Alice to Bob.

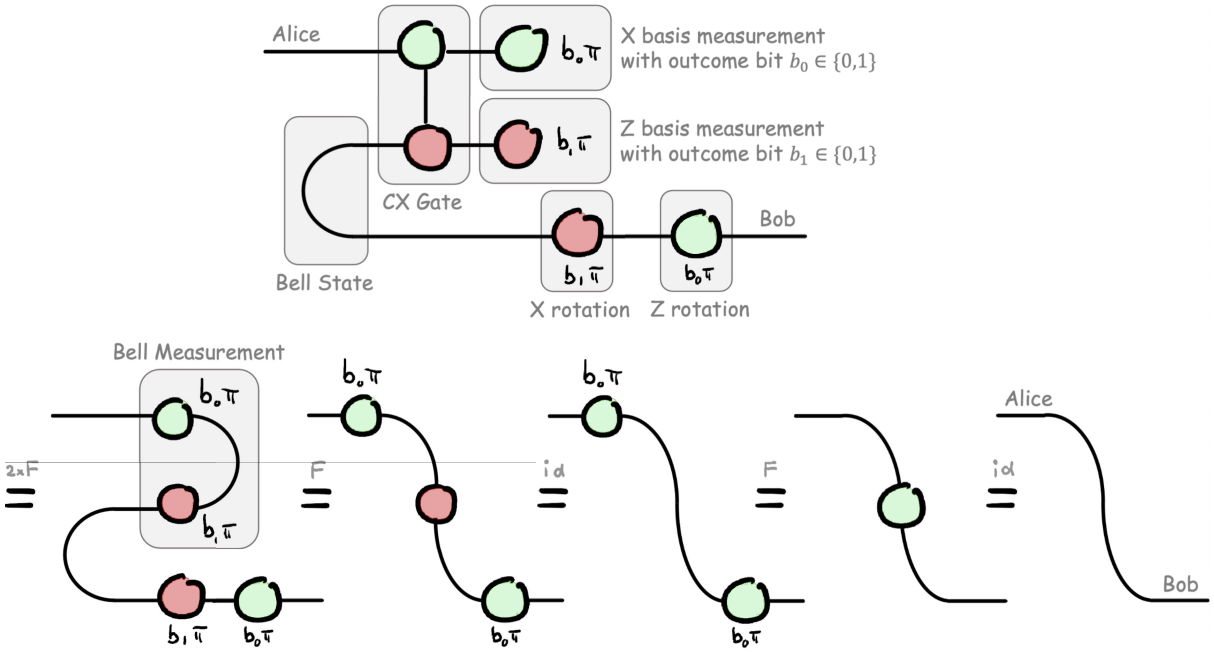


Figure 2.1: Figure reprinted from Ref. [DYP+23] proving correctness of the quantum teleportation protocol in the ZX-calculus.

Having covered sufficient ground to derive equalities of multi-qubit circuits with the rules so far, the question follows whether these rules are powerful enough to derive all other rules. This was answered in the affirmative in 2017-2018: a decade after it was first conceptualized, the ZX-calculus was proven to have the property of *completeness*. This means that any equality of the underlying linear maps can always be proven using only the rules of the calculus [HNW18; JPV18]—in other words, there are no "missing rules" that cannot be derived from a complete rule set. In 2018, this complete set of rules which was reduced to just 8 rules, each of which is conceptually simple, presented in Figure 2.2 [Vil19]. In other words, there are no missing rules that cannot be derived from these 8 rules.

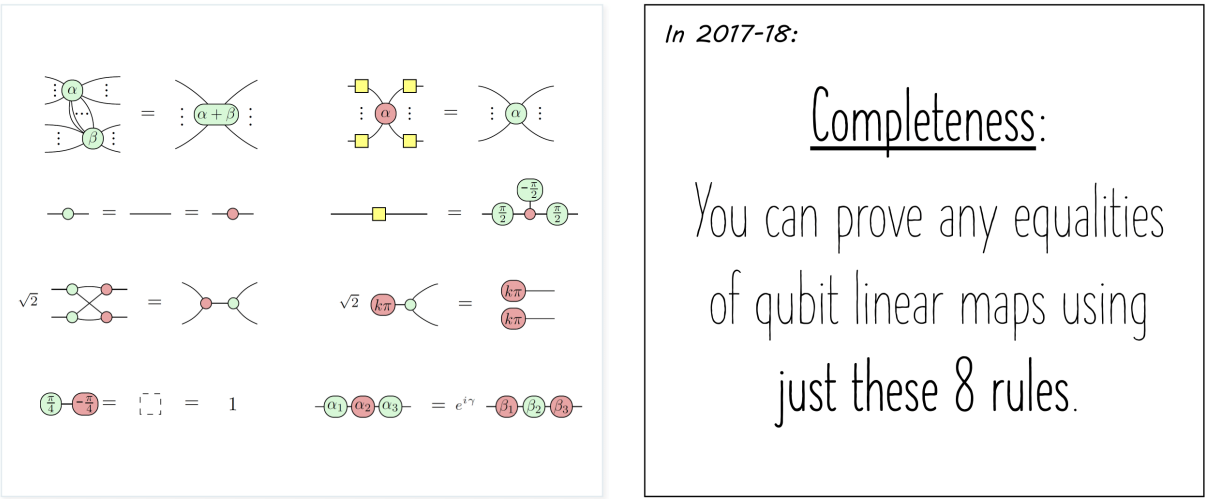


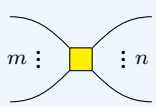
Figure 2.2: All equalities of qubit linear maps are derivable from these 8 rules of the ZX-calculus [Vil19].

In addition to the ZX-calculus, there are two other important calculi that are well-studied: the ZH-calculus and the ZW-calculus. These three calculi form a canonical trifecta of qubit graphical calculi, by the classification of qubit graphical calculi in Ref. [CJ20], where the main criterion is that Z and the other generator interact as a bialgebra.

All three of these calculi are sound, universal, and complete. In practice, which calculi to use differs depending on the situation. As a preview of types of circuits we will define in this thesis, the ZX-calculus is suited for the Clifford+Phases circuits, the ZH-calculus is suited for Toffoli+Hadamard circuits, and the ZW-calculus is suited for what we will call arithmetic circuits.

The ZH-calculus was first presented in Ref. [BK19]. In the ZH-calculus, in place of the X spider as a (non-derived) generator, we have the *H-box*. Although the ZH-calculus is also complete, in this thesis we focus on the phase-free (i.e. the H-box is unlabelled) fragment of the ZH-calculus.

Definition 2.2. The phase-free H-box is defined as:



$$= \sum_{\substack{x_1,\ldots,x_m,\\ y_1,\ldots,y_n=0}}^{1} (-1)^{x_1\cdot\ldots\cdot x_m\cdot y_1\cdot\ldots\cdot y_n} |y_1\ldots y_n\rangle \langle x_1\ldots x_m| \tag{2.3}$$

In the non-phase-free ZH-calculus, each H-box can be labelled with a complex number c, which replaces the (-1) in the above interpretation by c. In other words, when there is no phase label, the default is that $c = -1$.

When $m = n = 1$, the phase-free H-box is the H gate, as the interpretation above indeed

results in the matrix for the H gate. Conjugating any Z spider (with any phase) by the H gate on all wires, gives an X spider with the same number of wires and the same phase.

The phase-free qubit ZH-calculus corresponds precisely to (possibly post-selected) circuits over the Toffoli+Hadamard gate set. Like the ZX-calculus, the phase-free ZH-calculus has a complete rule set of 8 rules:

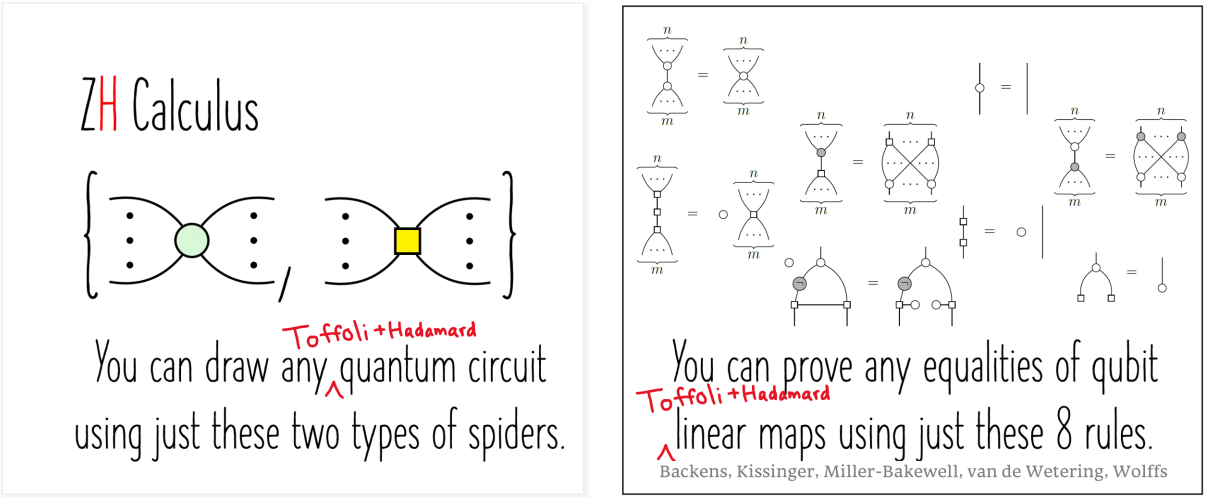


Figure 2.3: All equalities of qubit linear maps from the Toffoli+Hadamard gate set, are derivable from these 8 rules of the phase-free ZH-calculus [vdWW19; BKM+23].

The ZW-calculus was first presented in Ref. [CK10] and proven complete in Ref. [Had17]. It was conceptualized to study multipartite entanglement; the Z and W generators are respectively defined as generalizations of the GHZ and W states.

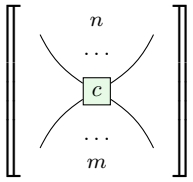
$$|GHZ\rangle = \frac{1}{\sqrt{2}}\left(|000\rangle + |111\rangle\right) \tag{2.4}$$

$$|W\rangle = \frac{1}{\sqrt{3}}\left(|001\rangle + |010\rangle + |100\rangle\right) \tag{2.5}$$

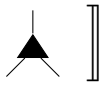
These two three-qubit states are respectively the representatives of the two tripartite entanglement equivalence classes under stochastic Local Operations and Classical Communication (sLOCC) [ABLS01].

In addition to the structural generators, the qubit ZXW-calculus (the qudit dimension $d = 2$

case of Ref. [PWS+23]) is built from the following generators:



$= |0^m\rangle\langle 0^n| + c|1^m\rangle\langle 1^n|, c \in \mathbb{C}$



$= |00\rangle\langle 0| + |01\rangle\langle 1| + |10\rangle\langle 1|$



$= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

We can use Z spiders (round) instead of Z boxes whenever c is a norm 1 complex number. Conjugating any Z spider by H on all wires gives an X spider of the same phase. While we do not use the W generator in this thesis, we do construct generalized qudit W states in Chapter 11.

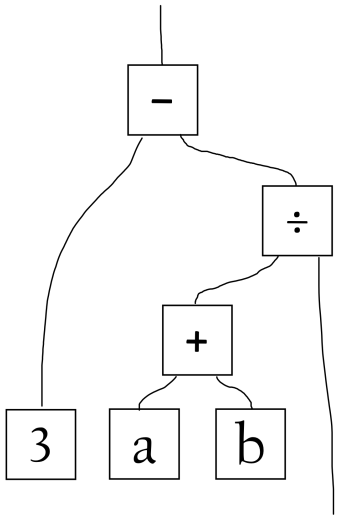
2.1 Classical functions in quantum algorithms

In this thesis, we will require a basic understanding of how classical reversible functions embed into quantum circuits.

We start with a useful observation, that arithmetic expressions can be drawn as trees. Wires carry information as inputs to the operands, whose output wires can then be input further.

Wires: numbers
Boxes: binary operations

$y = 3 - ((a + b) \div x)$ ⇝

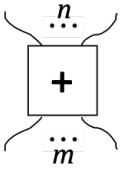


(2.6)

More generally, arithmetic equations can be drawn as graphs. As an important example, the equation below relating the input $\vec{x}$ and the output $\vec{y}$ can be drawn as the below operator,

generalizing addition:

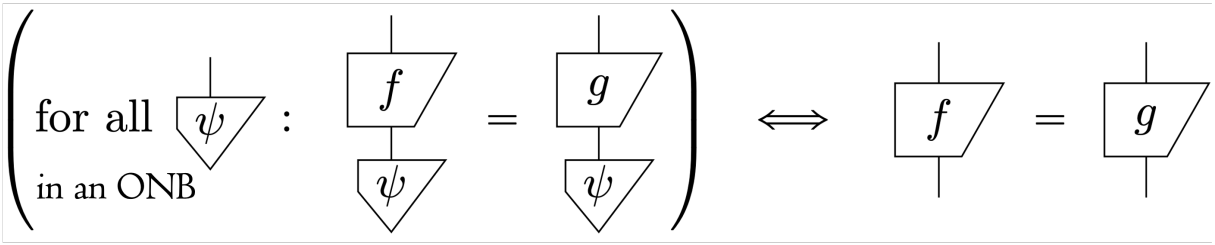
$$\sum_{j=1}^{n} y_j = \sum_{i=1}^{m} x_i \quad \rightsquigarrow$$



(2.7)

We didn't specify the types of x_i and y_j here—only that the equation type checks. If we take the wire types to be bits, then this addition is modulo 2. If we take the wire types to be qubits, and x_i and y_j to be bits identified with the Z-basis states, then this addition modulo 2 operator is exactly the m-input, n-output X spider.

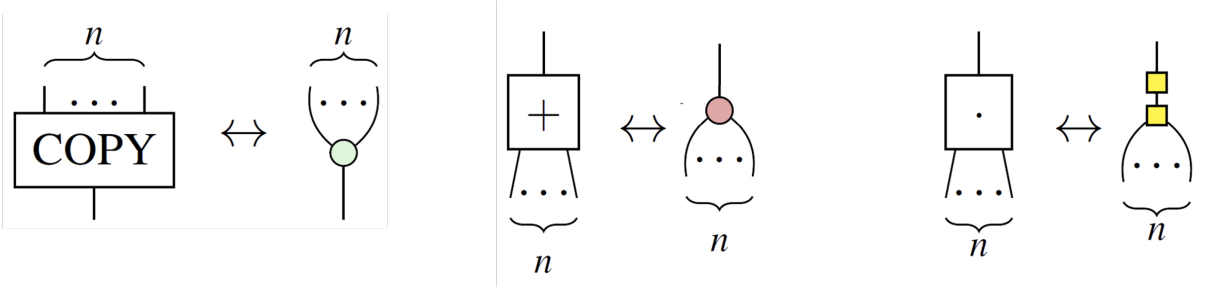
This definition is enough to fix the definition of the X spider, because it is enough to specify what it does to all Z-basis states. This is because every linear map is fixed by what it does on each state forming a basis for the space—what it does to any other pure state is fixed by pure states being linear combinations of basis states, and to any mixed state is fixed by mixed states being linear combinations of density operators of pure states. In quantum information, oftentimes it is most convenient to choose an orthonormal basis (ONB).



(2.8)

This lets us verify equalities of quantum processes, by simply checking that the equality holds on all elements of an ONB.

In the ZX- and ZH-calculi, we can define the (label-free) generators Z, X, and H based on their action on Z-basis states:

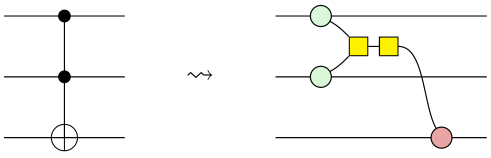


(2.9)

In fact, we can represent all computation on bits using just these three operations (modulo 2): copy, addition, and multiplication. For instance, multiplication modulo 2 is the binary AND

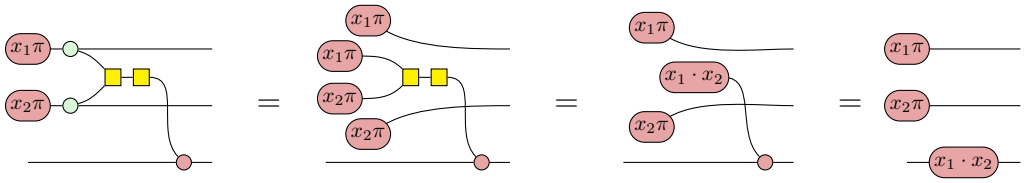
operation, addition modulo 2 is the binary OR operation, and addition modulo 2 where one of the inputs is 1 is the binary NOT operation.

The Toffoli (aka CCX) gate in the ZH-calculus is:



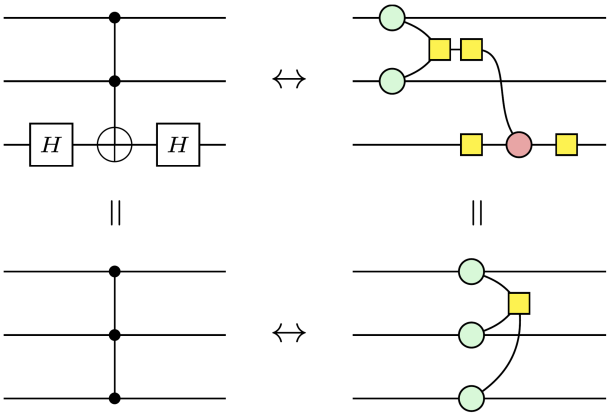
(2.10)

We can check that it does an X gate on the target qubit, if and only if both control qubits are in the state $|1\rangle$, i.e. their bits multiply to 1.



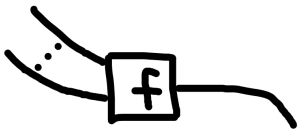
(2.11)

By conjugating CCX by H on the target qubit, we get the ZH diagram for the CCZ gate:



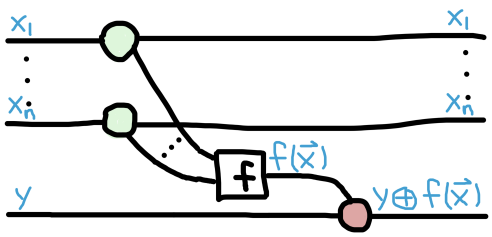
(2.12)

Any classical function on n bits $f : \{0,1\}^n \to \{0,1\}$ can be constructed from addition and multiplication modulo 2 operations, comprising a diagram of the shape:



(2.13)

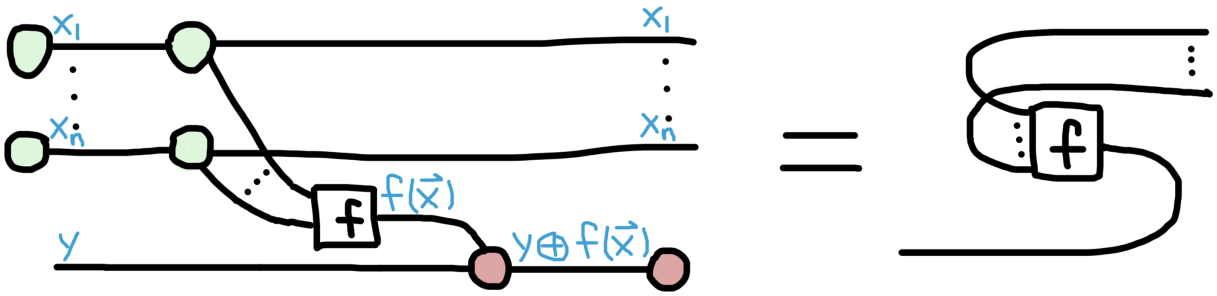
The ZH diagram for the quantum oracle U_f for the classical function f is:



(2.14)

U_f is a unitary which does an X gate on the target qubit if and only if f of the input bits is 1. Here the blue text is not needed in the diagram, but rather extra information to track how input bits $x_1, \ldots, x_n$ and the output bit y propagate through the circuit.

A helpful trick that quantum computers can do is transpose classical functions. For instance, one circuit that does this is to prepare all control qubits of the quantum oracle in the $|+\rangle$ state, and measure the target qubit in the computational basis. If $\langle 0|$ was measured, this results in:



(2.15)

As an equation on binary variables, this is $y = f(\vec{x})$, but now y is the input and $\vec{x}$ is the output. When $y = 0$ or $y = 1$ (or a superposition of these two computational basis states) is input to this transpose of f map, the output n qubits are in a superposition of all possible bitstrings for which $f(\vec{x})$ evaluates to (that superposition of) y.

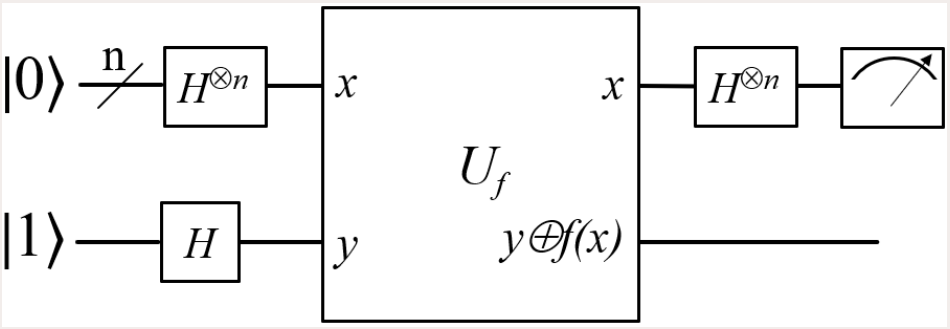
Below, we give an example of a quantum algorithm to show a use case of classical reversible circuits, which is a focus of this thesis. This is the only place in the thesis we will use the discard generator and compute specific probabilities, which we do not go into as this example is for illustrative purposes.

Example 2.16. The Deutsch-Jozsa algorithm [DJ92] is a staple textbook quantum algorithm, which for many students is the first one they learn. Despite not being practically useful, it has exponential speedup over the best possible deterministic classical algorithm.

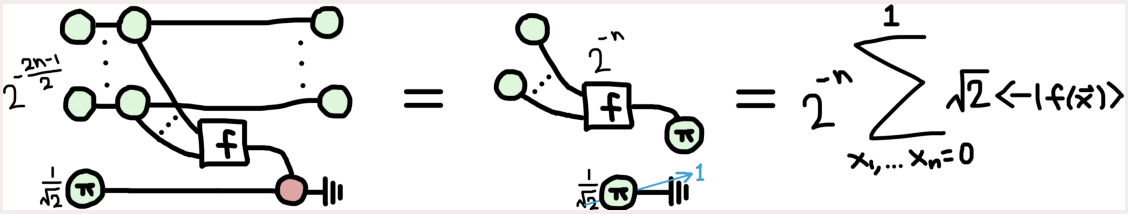
In the problem statement, you are promised that the function f that you have oracle access to, is either constant (the output bit is the same for all inputs) or balanced (the output bit is 1

for exactly half the inputs). The goal is to figure out which of these two cases it is.

The quantum circuit implementing the Deutsch-Jozsa algorithm (Reprinted from [Com19]) is:



For a quantum computation, the probability amplitude of a particular measurement outcome, is the scalar that the diagram evaluates to for that outcome, with all unmeasured qubits discarded (i.e. traced out). In this case, for the above circuit, we plug in $\langle 0|$ to each of the n control qubits and discard the target qubit of the quantum oracle.



(2.17)

In the final expression above, the summand equals 1 when $f(\vec{x}) = 0$ and -1 when $f(\vec{x}) = 1$. Summing over all 2^n possible input bitstrings $\vec{x}$, we see that the probability amplitude of measuring $|0\rangle^{\otimes n}$ is 1 when f is constant and 0 when f is balanced.

The Scalable ZX-calculus

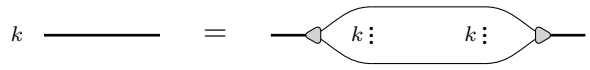
We now review the more compact notation of the scalable ZX-calculus (SZX) [CHP19], which we will briefly use for a succint proof for CSS codes. We do not need the more heavy lifting capabilities of the scalable ZX-calculus in this thesis, so we keep the presentation lightweight.

For any SZX-diagram $D : a \to b$, its interpretation $[\![D]\!]_s$ is a triplet (M, a, b) where $M \in \mathcal{M}_{2^{S(b)} \times 2^{S(a)}}(\mathbb{C})$. $[\![D]\!]_s$ is inductively defined as: $[\![D_1 \circ D_0]\!]_s = (M_1 \circ M_0, a, c)$, $[\![D_0 \otimes D_2]\!]_s = (M_0 \otimes M_2, a+c, b+d)$ where $[\![D_0]\!]_s = (M_0, a, b)$, $[\![D_1]\!]_s = (M_1, b, c)$, and $[\![D_2]\!]_s = (M_2, c, d)$.

$$
\begin{aligned}
\left[\!\left[\text{—}\blacksquare\text{—} \right]\!\right]_s &:= \left(\tfrac{1}{\sqrt{2}^n} \sum_{x,y\in\{0,1\}^n} (-1)^{x\bullet y} \left|y\right\rangle\!\left\langle x\right|, 1_n, 1_n\right) \\
\left[\!\left[\begin{array}{c}\vdots\\ \alpha \\ \vdots\end{array} \right]\!\right]_s &:= \left(\sum_{x\in\{0,1\}^n} e^{ix\bullet\alpha} \left|x^k\right\rangle\!\left\langle x^\ell\right|, k_n, \ell_n\right) \\
\left[\!\left[\begin{array}{c}\vdots\\ \alpha \\ \vdots\end{array} \right]\!\right]_s &:= \left[\!\left[\text{—}\blacksquare\text{—} \right]\!\right]_s^{\otimes \ell} \circ \left[\!\left[\begin{array}{c}\vdots\\ \alpha \\ \vdots\end{array} \right]\!\right]_s \circ \left[\!\left[\text{—}\blacksquare\text{—} \right]\!\right]_s^{\otimes k} \\
\left[\!\left[\text{—}\!\!\subset \right]\!\right]_s &:= \left(id_{n+1}, 1_{n+1}, 1{+}1_n\right) \\
\left[\!\left[\supset\!\!\text{—} \right]\!\right]_s &:= \left(id_{n+1}, 1{+}1_n, 1_{n+1}\right) \\
\left[\!\left[\subset \right]\!\right]_s &:= \left(\sum_{x\in\{0,1\}^n} \left|xx\right\rangle, 0, 2_n\right) \\
\left[\!\left[\supset \right]\!\right]_s &:= \left(\sum_{x\in\{0,1\}^n} \left\langle xx\right|, 2_n, 0\right) \\
\left[\!\left[\times \right]\!\right]_s &:= \left(\sum_{x\in\{0,1\}^n, y\in\{0,1\}^m} \left|yx\right\rangle\!\left\langle xy\right|, 1_n{+}1_m, 1_m{+}1_n\right) \\
\left[\!\left[\text{—} \right]\!\right]_s &:= \left(id_n, 1_n, 1_n\right) \\
\left[\!\left[\begin{array}{|c|}\hline\\ \\ \\ \hline\end{array} \right]\!\right]_s &:= 1
\end{aligned}
$$

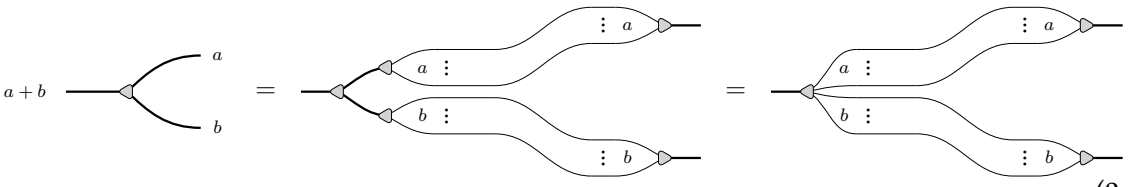
Where $\forall u, v \in \mathbb{R}^m$, $u \bullet v = \sum_{i=1}^m u_i v_i$, $M^{\otimes 0} = 1$, and $M^{\otimes k+1} = M \otimes M^{\otimes k}$.

Like the qubit ZX-calculus, the Only Connectivity Matters rule also applies to all scalable ZX-calculus diagrams. The scalable notation allows us to represent a register of qubits with a single wire. A bold wire labelled k is a register of k qubits.



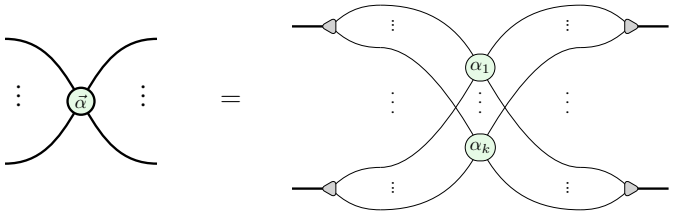
(2.18)

The divide node (below) and the gather node (transpose of the below) allow us to easily regroup the registers of qubits.

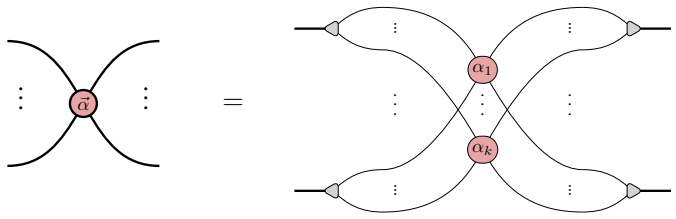


(2.19)

The spiders also have scalable versions, now labelled with a vector of phases $\vec{\alpha} = (\alpha_1, \cdots, \alpha_k) \in \mathbb{R}^k$.

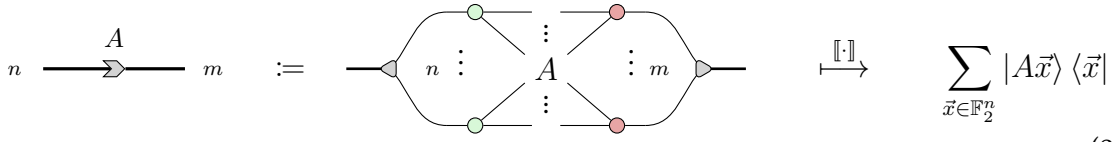


(2.20)



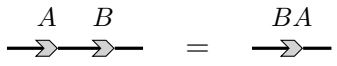
(2.21)

It is easy to see that all the rewrite rules of the ZX-calculus extend to the scalable setting. In addition, the scalable ZX-calculus introduces a generator called the matrix arrow. It allows us to compactly represent a bipartite graph of n input Z spiders connected to m output X spiders. Suppose $A \in \mathbb{F}_2^{m \times n}$ is the biadjacency matrix of such a graph, i.e. $A_{ij} = 1$ if and only if the ith Z spider is connected to the jth X spider. Matrix arrows in the scalable ZX-calculus are defined as



(2.22)

where $|\vec{x}\rangle$, $\vec{x} \in \mathbb{F}_2^n$ is a computational basis state. As a convention, we omit the matrix arrow label when it is the all-ones matrix. With the matrix arrows, we have additional rewrite rules. Composing matrix arrows corresponds to multiplying the matrices:



(2.23)

We can copy matrix arrows through the spiders as shown below:



(2.24)

This wraps up the crash course on the qubit ZX-calculus.

3

Quantum information theory

3.1 Stabilizer formalism

In this background section, we introduce the stabilizer formalism for qubit computation. In the next background section, we will revisit these definitions generalized to qudit computation. This way, readers interested in the qubit chapter of this thesis can read this background section on qubits, without needing to read background on qudits. Later in this thesis, but not here yet, we will have a light introduction to quantum error correcting codes in the ZX-calculus.

First, we define the Pauli group, whose elements are called Paulis.

Definition 3.1 (Pauli group)**.** The *Pauli group* $\mathcal{P}_n$ is the group of n-qubit operators generated by the *Pauli matrices*:

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

Sometimes, the Pauli matrices also includes the identity matrix

$$I = X^2 = Y^2 = Z^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Some definitions of the Pauli group also include arbitrary global phases $e^{i\theta}$ for any real θ.

The three Pauli matrices X, Y, and Z are also the rotations of angle π about the x, y, and z axes of the Bloch sphere, respectively.

Integer powers of i are always in the Pauli group because

$$I = -iXYZ = -iYZX = -iZXY$$

and so

$$iI = XYZ$$

Remark 3.2. When operators are written sequentially with symbols as done above, the intended order of composition matches that of matrix multiplication, from right to left.

In quantum information and computation, in many cases this way of writing operators with a space or nothing in between them actually denotes their tensor product, rather than matrix multiplication. Which of the two is being denoted is supposed to be clear from context—for example, when referring to quantum error correcting codes and stabilizer tableaus, tensor product is intended.

We seldom use a semicolon ; to denote composing operators from left to right, which for instances matches how quantum circuits are drawn. For example, for the composition of two operators A and B, B then A is written as

$$AB = B; A$$

As is conventionally defined in the literature, something is *generated by* a set of generators when every element of that something can be expressed by a finite composition of the generators. Generally, the allowed compositions are multiplication and tensor product. That something could be a group, a vector space, the quantum circuits possible over a gate set, a fragment of quantum computation, a fragment of a graphical calculus, or other things, where the or here is not mutually exclusive.

Next, we define the Clifford group, whose elements are called Cliffords.

Definition 3.3 (Clifford group)**.** The *Clifford group* $\mathcal{C}_n$ is the group of n-qubit operators

generated by the *Clifford gate set* [Got98b]

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad CX = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Sometimes, this definition of the Clifford group also includes arbitrary global phases.

The Pauli group is a subgroup of the Clifford group, which can be seen by observing that all generators of the Pauli group are elements of the Clifford group:

$$Z = S^2, X = HZH, iI = SXSX, Y = iXZ \tag{3.4}$$

Instead of defining the Clifford group by its generators, we could have also defined it based on the Paulis.

Definition 3.5 (Clifford unitary)**.** Let U be a unitary acting on n qubits. We say it is *Clifford* when every Pauli is mapped to another Pauli under conjugation by U, i.e. if $UPU^\dagger$ is in the n-qubit Pauli group for any Pauli P. All such unitaries form the Clifford group on n qubits.

A core concept to quantum information that is closely linked to the Paulis and Cliffords, is that of *stabilizer states*. Gottesman defined them as [Got98a]:

Definition 3.6 (Stabilizer state)**.** The stabilizer of a state is a group of *stabilizer generators*—each element of the Pauli group that the state is a $+1$ eigenstate of.

A state which can be completely described (up to global phase) by specifying the stabilizer is called a *stabilizer state*.

Definition 3.7. A state ψ is *stabilized by* an operation $\mathcal{O}$ if it leaves it unchanged:

$$\mathcal{O} |\psi\rangle = |\psi\rangle \tag{3.8}$$

Example 3.9. The GHZ state $|GHZ\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ is stabilized by $X \otimes X \otimes X$. A

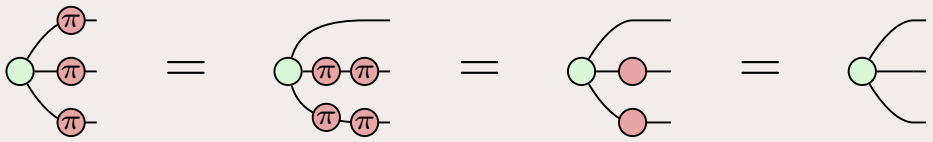
proof in bra-ket notation is

$$(X \otimes X \otimes X)\,|GHZ\rangle = \frac{1}{\sqrt{2}}\left((X\,|0\rangle) \otimes (X\,|0\rangle) \otimes (X\,|0\rangle) + (X\,|1\rangle) \otimes (X\,|1\rangle) \otimes (X\,|1\rangle)\right) \tag{3.10}$$

$$= \frac{1}{\sqrt{2}}(|111\rangle + |000\rangle) \tag{3.11}$$

$$= |GHZ\rangle \tag{3.12}$$

A proof in the ZX-calculus is



(3.13)

Campbell, Anwar, and Browne defined *stabilizer operations* as follows [CAB12]:

Definition 3.14. Consider a completely positive map $\mathbb{E}$: $\mathbb{B}(\mathbb{H}^{d^{n_{in}}}) \to \mathbb{B}(\mathbb{H}^{d^{n_{out}}})$. The map is a stabilizer operation if and only if it can be composed from the following elements:

1. Clifford unitaries,
2. measurements and subsequent projections on stabilizer subspaces,
3. preparation of fresh ancilla in a stabilizer state,
4. tracing out of unwanted qudits, and
5. adaptive decision making based both on measurement outcomes and on random coin tosses

When we introduce quantum error correcting codes in this thesis, we will also introduce how to represent stabilizer states with a *stabilizer tableau*—a binary matrix indicating the Pauli strings that stabilize that state. By representing the application of Clifford gates with tableau operations, any quantum computation in the stabilizer fragment can be efficiently simulated according to the Gottesman-Knill Theorem [Got98a]:

Theorem 3.15 (Gottesman-Knill)**.** *Any quantum computer performing only:*

a) Clifford group gates,

b) measurements of Pauli group operators, and

c) Clifford group operations conditioned on classical bits, which may be the results of earlier measurements,

can be perfectly simulated in polynomial time on a probabilistic classical computer.

Therefore, the Clifford group operations used in stabilizer quantum error correction codes, are also operations which are efficiently classically simulable.

However, gate sets consisting of only Clifford gates cannot be universal. To generate quantum circuits able to approximate any quantum computation, at least one non-Clifford gate must be added to the gate set.

Definition 3.16. For the purposes of this thesis, we will say that sufficient conditions to call an approximately universal gate set *fault-tolerant* are that:

- it contains gates which generate the Clifford group,
- it contains at least one non-Clifford gate
- this non-Clifford gate admits an injection (also referred to as gate teleportation) protocol, which consumes one or more magic states and whose probability of success if multiple rounds of the protocol are necessary converges quickly to one

As a gate set, note that this definition does not require specification (for instance, a given hardware implementation or noise model) of how these gates are implemented.

Remark 3.17. One notable gap in this definition is that for all non-Clifford gates in this thesis, we do not necessarily require that the magic state for the non-Clifford gate in question is at the endpoint of some magic state distillation protocol which supresses the error. I conjecture this to be the case for the two new non-Clifford gates defined in this thesis, the Toffoli and $\sqrt[d]{Z}$ gates for any prime qudit dimension. In any case the error of the quantum computation is bounded by the non-Clifford gate count, but the polynomial for how the error rates scale with the non-Clifford gate count is an important consideration outside the scope of this thesis.

There are several different proposals for how to implement a Clifford + non-Clifford gate set, such as *magic state distillation* [BK05], code switching [ADP14; Bom16; KB15], magic state cultivation [GSJ24], and more. Later in this thesis we will discuss code switching in more detail. For now, we introduce magic state distillation and injection, so that the concept of non-Clifford gates can be conveyed.

A *magic state* is a non-stabilizer state which can be continually refined to bound its error through a process called *magic state distillation* in a quantum error correcting code. Magic states are typically consumed in order to perform non-Clifford logical operations in another quantum error correcting code—notably, one that can do all logical Clifford operations in such

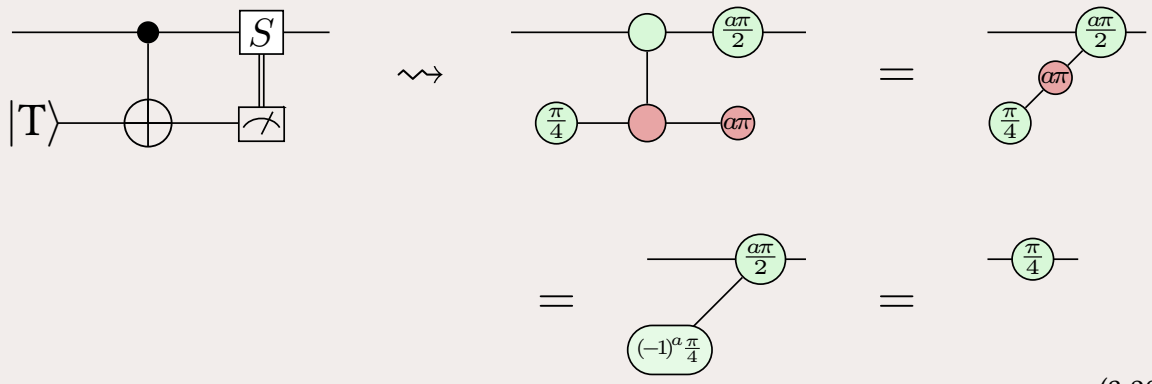
a way, that they don't propagate any single-qubit errors to multi-qubit errors in the same code block. Simple examples of small codes with this property which we encounter in this thesis are the $[\![4, 2, 2]\!]$ and the $[\![5, 1, 3]\!]$ code.

Later, when we introduce quantum error correcting codes, we will have a quick graphical explanation of magic state distillation protocols for the most well-studied magic state, the $|T\rangle$ state.

Example 3.18. The most well-studied magic state is the $|T\rangle$ state. Injecting one copy of the $|T\rangle$ state into a quantum circuit, lets you implement the most well-studied non-Clifford gate, the single-qubit T gate (also called the $\pi/8$ gate):

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \tag{3.19}$$

Translating the T magic state injection protocol into the ZX-calculus, we can verify that it implements the T gate deterministically (up to an irrelevant global phase), regardless of whether the measured bit a was 0 or 1.



(3.20)

This is also called a gate teleportation protocol because it teleports a resource state (the T state) in order to enable performing the T gate. In addition to this T state injection and the standard quantum teleportation protocol in the introduction section, many other quantum teleportation protocols admit a clean presentation in the ZX-calculus.

This has the effect of adding the T gate to the Clifford gate set; this combined gate set is called the *Clifford+T* gate set, and is the most studied gate set in quantum computation. It is (approximately) universal for quantum computation, meaning that any quantum circuit can be approximated to arbitrary precision by a circuit over the Clifford+T gate set.

Not all non-Clifford gates possess the same viability in a fault-tolerant setting. The Clifford

hierarchy is an important concept in characterizing which gates admit magic state injection protocols or transversal implementation on stabilizer codes.

Definition 3.21 (Clifford hierarchy)**.** The n^{th} level of the Clifford hierarchy C_n [Got99; PRTC20; RCP19; CGK17] is defined inductively as $C_1 \coloneqq \mathcal{P}$, the Pauli's and

$$C_n \coloneqq \{\text{unitary } U \mid \forall P \in \mathcal{P} : UPU^\dagger \in C_{n\text{-}1}\}. \tag{3.22}$$

The Pauli gates form C_1 and the Clifford gates form C_2, while certain non-Clifford gates belong to C_n for higher n. The Clifford hierarchy is significant because of its relation to the correction of Pauli errors. For instance, when implementing a gate via a state injection mechanism [BK05], a gate from C_i requires a correction from $C_{i\text{-}1}$. Thus, implementing a gate from the third level of the hierarchy using state injection requires a Clifford correction.

Important to the injectability of a gate is whether it is semi-Clifford, for which we follow the definition from Ref.[dSil21]:

Definition 3.23. A gate G is *semi-Clifford* if it admits a decomposition $G = C_2 D C_1$, as a diagonal gate in some level of the Clifford hierarchy D, pre- and post-composed by Clifford gates C_1 and C_2 respectively.

Any single-qubit or two-qubit gate in the k^{th} level of the Clifford hierarchy is *semi-Clifford*, and thereby implementable by a gate teleportation protocol [ZCC08]. This can be the protocol of Ref. [ZLC00] that recursively teleports diagonal gates in the Clifford hierarchy, to realize corrections that are successively one level lower in the Clifford hierarchy.

In Chapter 13, we will overview how to *synthesize* quantum circuits that realize certain matrices over a certain gate set.

4

Quantum error correcting codes

This section consists of a brief graphical introduction to quantum error correcting codes. We focus on the codes aspect of quantum error correction, meaning that we will not discuss noise channels nor the decoding process.

4.1 Classical error correcting codes

For those new to quantum error correction, the easiest starting point is likely to be with classical error correcting codes. Consider you have m classical bits, which each have a small but nonzero probability p of experiencing a bit flip error. Therefore, the probability of one of the bits not being flipped is $1-p$, and the total probability of no errors occurring is $(1-p)^m$.

In this section, let us consider the three-bit repetition code, which protects against bit flip errors. For each bit of information, encode three copies of it. There are two possible bitstrings that encoded bits can take: 000 and 111—these are called the *codewords* of this code. This code has a *code distance*, or commonly just called *distance*, of $d=3$ because the minimum Hamming distance between any two codewords is 3. This means that for one codeword to experience so much error that it becomes another codeword, there must have been a minimum of three bits flipping.

If all three of these bits are the same, then either all three bits flipped (which is highly unlikely,

with probability p^3), or none of them flipped (with probability $(1-p)^3$, which is the most likely outcome if $p < 0.25$). If two of the bits are the same, then so long as $p < 0.5$, it is more likely that only one of the bits flipped (with probability $3p(1-p)^2$), than two of them flipped (with probability $3p^2(1-p)$). Then we should presume that the more likely outcome was the one which occurred. We look at which value (0 or 1) was the majority of the three bits, and flip whichever of the three bits is not equal to that.

This value of $p = 0.5$ is the *threshold*. If the physical error rate p exceeds this threshold, then the probability that the decoding fails (does not correct the error that occurred) is higher than the probability that it succeeds. This value of 0.5 is the highest threshold possible for this error model—note this dependence of thresholds on the error model. However, this is not considered a good code because its *encoding rate* of $\frac{1}{3}$ is very low, of 1 logical bit (of encoded information) per every 3 physical bits (of bits you actually use to store and compute information).

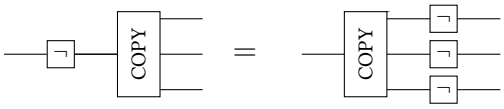
As the decoding takes place locally to each set of three bits which encode a logical bit, each *code block* consists of three physical bits and one logical bit. More generally, each code block consists of k logical bits and n physical bits, where $n \geq k$. Codes are commonly referred to by the three numbers $[\![n, k, d]\!]$, where d is the code distance—the minimum Hamming distance between codewords, i.e. the minimum number of bitflips to turn one codeword into another.

We can represent classical error correcting codes diagrammatically. This is because any computation on classical bits can be represented diagrammatically, using the generators {COPY, ADD, MULT}. The *encoder* map of the three-bit repetition code is the 1-input, 3-output COPY operator. Indeed, by the copy rule, inputting the bit 0 into the encoder gives the output 000, and inputting the bit 1 gives the output 111.



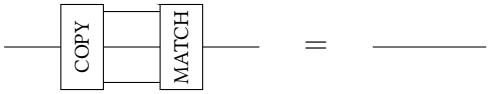
(4.1)

We can also see that the *logical operation* of doing a bit flip on the input bit, is implemented by the physical operation of doing a bit flip on all three physical bits.



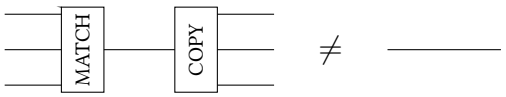
(4.2)

The encoder of any code is an isometry—it is reversible by doing its one-sided inverse, called the *ideal decoder* because it decodes all codewords and annihilates all other inputs to it. For this example code, the ideal decoder is the 3-input, 1-output MATCH operator which sends 000 to 0 and 111 to 1, and annihilates all other inputs.



(4.3)

Doing the ideal decoder, followed by the encoder, is notably not the identity operation. This follows from the fact that compressing n bits (or qubits) into $k < n$ bits (or qubits) necessarily loses information.

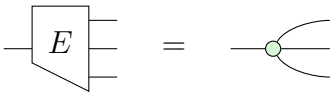


(4.4)

In fact, it is the projector onto the *code space* of the code, as it is identity on any codewords, and annihilates all three-bit states that are not codewords.

4.2 A quantum error correcting code

With quantum error correcting codes, there are two main differences with classical bit error correcting codes. First, rather than discrete codewords which are bitstrings, the space of encoded quantum states is continuous. For example, taking the three-bit repetition code from before, we can apply the trivial way to lift a classical code to a quantum code, by replacing each generator in the encoder with its quantum analogue. Replacing 0 with $|0\rangle$; 1 with $|1\rangle$; and the 1-input, 3-output COPY operation with the 1-input, 3-output Z spider: we get the three-qubit bit-flip code.



(4.5)

In addition to $|0\rangle$ and $|1\rangle$, we can encode any single-qubit state; this is determined by linearity of how the encoder maps $|0\rangle$ and $|1\rangle$. For an arbitrary single-qubit state determined by two complex coefficients c_0 and c_1 such that $|c_0|^2 + |c_1|^2 = 1$, the encoder maps this as:

$$c_0 |0\rangle + c_1 |1\rangle \mapsto c_0 |000\rangle + c_1 |111\rangle \tag{4.6}$$

The code space of this code is a vector space, which can be fixed by a basis for it—for instance $\{|000\rangle, |111\rangle\}$.

Note that here we adhere to the convention of omitting explicitly denoting the tensor product, e.g.

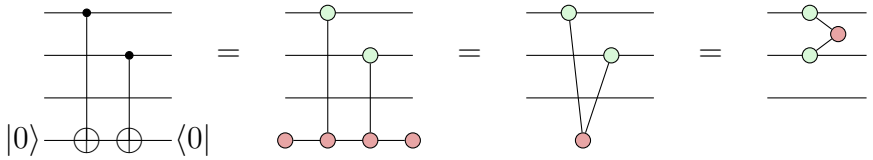
$$|000\rangle = |0\rangle \otimes |0\rangle \otimes |0\rangle = |0\rangle\,|0\rangle\,|0\rangle$$

and

$$IZZZZI = (I \otimes Z \otimes Z)(Z \otimes Z \otimes I) = (IZ) \otimes (ZZ) \otimes (ZI) = Z \otimes I \otimes Z = ZIZ$$

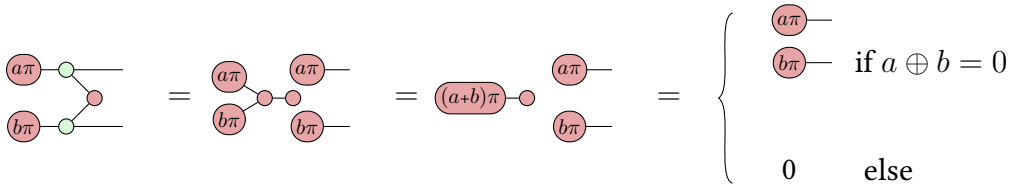
In the classical case we saw in the previous section, we could simply read the bits to see if any error occurred. The problem with doing this for quantum states, is that measuring whether a qubit is in either $|0\rangle$ or $|1\rangle$ necessarily collapses any qubit state that is a superposition of $|0\rangle$ and $|1\rangle$. The way that quantum errors are detectable without collapsing the encoded quantum

information, is by measuring the parity of two pairs of the three qubits. A fault-tolerant circuit for measuring one of these pairs is:



(4.7)

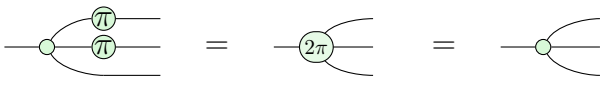
A proof that this measures the parities of the Z basis components of the input states is:



(4.8)

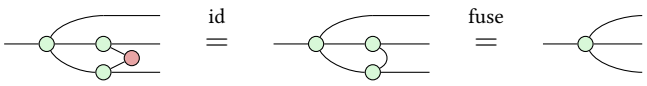
If no bit-flip error occurred, both two-qubit parity measurements should be 0. If any bit-flip error occurred, then at least one of these two parity measurements should be 1. These measurements are called *syndrome measurements*. The measurement outcomes, called *syndromes*, are information necessary for the decoder to determine which error most likely occurred, and hence what correction to apply.

Rather than by giving basis states for the code space, the other, more common way that the code space of a quantum error correcting code is defined, is by defining generators of the *stabilizers* of the code. The stabilizers of the three-qubit bit-flip code being measured in the above circuit are ZZI and IZZ. Note that X spiders (which are diagonal in the X basis) are stabilized by Pauli Z on every leg, while Z spiders (which are diagonal in the Z basis) are stabilized by Pauli X on every leg. This is simply a consequence of how $X^{\otimes n} |\text{GHZ}\rangle = |\text{GHZ}\rangle$, and $Z^{\otimes n} (H^{\otimes n} |\text{GHZ}\rangle) = (H^{\otimes n} |\text{GHZ}\rangle)$. We can see that these are indeed stabilizers for any codeword, by applying them to the encoder. This holds, whether we apply the Pauli operators (exemplified here for ZZI)



(4.9)

or their associated projectors (exemplified here for IZZ)



(4.10)

These two stabilizers, ZZI and IZZ, are (the only two, as $n - k = 3 - 1 = 2$) *stabilizer generators* of the code, because all other stabilizers of the code are a product of these.

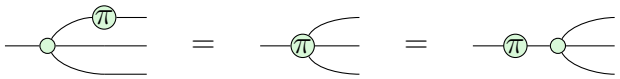
For this code, the only other non-identity stabilizer is their product, $ZIZ = ZZIIZZ = IZZZZI$. We do not need to measure this stabilizer, because (assuming no errors from the

measurements themselves) its measured bit is always the XOR of the other two bits which we have already measured.

We can write down these two stabilizer generators as a binary matrix, called the *parity check matrix* H_Z of the code. Each row of H_Z is a Z-type (i.e. over $\{Z, I\}$) stabilizer generator, with a 1 in the ith column if there is a Z on the ith qubit, and 0 otherwise.

$$H_Z = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{4.11}$$

The second main difference of quantum error correcting codes compared to classical bit error correcting codes, is that in addition to bit-flip errors, there are also *phase-flip* errors. A bit-flip error is a single-qubit Pauli X operator, which flips $|0\rangle$ to $|1\rangle$ and vice-versa. While a phase-flip error (a single-qubit Pauli Z operator) does not affect $|0\rangle$ or $|1\rangle$ other than by an irrelevant global phase, it is an error on superpositions of these states. For instance, it flips $|+\rangle$ to $|-\rangle$ and vice-versa. This code protects against bit-flip errors, but not phase-flip errors. We can see this from the fact that a physical phase-flip error always results in a logical phase-flip error.



(4.12)

This is not detectable, as any error which is equivalent to a logical operator is not detectable. This is because if enough error occurs that a codeword is transformed into another codeword, then the decoder cannot differentiate such error from a logical operation that did the same transformation.

This means that there is a geometric way to think about code distance, by plotting codewords in n-dimensional space. Consider a simple decoder which always maps a physical state to the nearest codeword. How to design a code maximizing the distance d between codewords is equivalent to a discretized sphere-packing problem with a sphere centered at each codeword, where physical states inside each sphere are corrected to the codeword at the center of the sphere.

For both classical and quantum error correcting codes, the code distance is the minimum number of different 'letters' between any two codewords. The distance of a quantum error correcting code is the minimum number of qubits which must experience a physical error (projected by the syndrome measurements to be Paulis), in order to change one codeword into another. For the reason above, this is equivalent to the minimum weight (number of non-identity Pauli operators) of all the logical operators of the code. For this code, the distance is $d = 1$, because the logical Z operator is $\overline{Z} = ZII = IZI = IIZ$, which is a weight 1 operator.

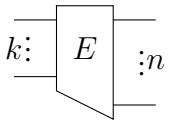
4.3 Stabilizer quantum error correcting codes

In the previous section, we gave a conceptual overview of quantum error correction (QEC) through toy classical and quantum examples. In this section, we go a step further and present preliminaries of stabilizer codes. This is an extremely general class of quantum error correcting codes; in fact, almost all of the well-studied quantum error correcting codes in the literature are stabilizer codes.

So far, we have discussed how a QEC code is often referred to by $[\![n, k, d]\!]$, where n is the number of physical qubits, k is the number of logical qubits, and d is the code distance. While the following concepts apply similarly when $n = k$, here for consistent terminology we assume $n > k$, as otherwise the code can neither detect nor correct errors. In the explanation below, the depictions are as if $k > 0$, but the definitions hold nonetheless for when $k = 0$ (which is simply a stabilizer state).

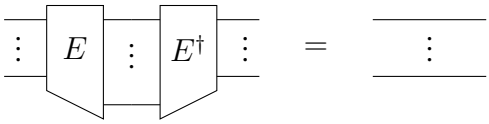
The encoder view

We discussed previously how to represent a QEC code by an encoder map $\mathsf{E} : \mathbb{C}^{\otimes k} \hookrightarrow \mathbb{C}^{\otimes n}$:



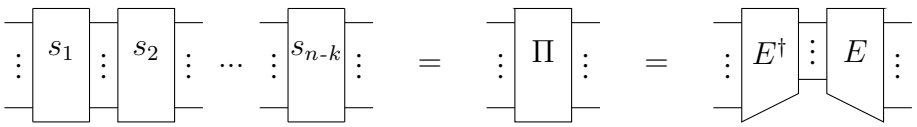
(4.13)

from k logical qubits to n physical qubits. E must be an isometry, meaning that it has a one-sided inverse $E^\dagger$. $E^\dagger$ is also called the *ideal decoder* because it decodes all codewords, and annihilates all physical states outside the code space.



(4.14)

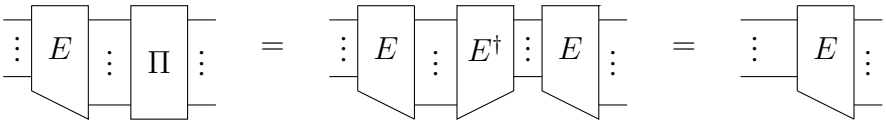
As $n > k$, applying $EE^\dagger$ is not the identity, but rather is the projector map $\Pi : \mathbb{C}^{\otimes n} \to \mathbb{C}^{\otimes n}$ onto the code space of the code. Π can be decomposed as the product of $n - k$ linearly independent projectors, each onto the $+1$ eigenspace of a Pauli string s_i. We will shortly define these Pauli strings as the *stabilizer generators* of the code.



(4.15)

The conventional terminology of a code would define the code by its code space, which is bijective with the projector Π. Comparing this to the encoder E in the above equation, we can

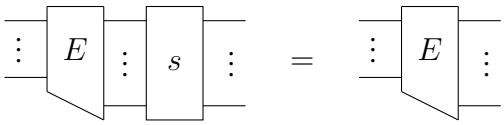
see that there are as many possible encoders for a given code, as there are unitaries on k qubits. The reason for this is that the code space is the image of the encoder map, i.e. the space of possible output states of the encoder. Information about the code space alone does not specify anything about which logical qubit states are encoded into which physical qubit states.



(4.16)

The code space is defined by *stabilizers* of the code. In the case of stabilizer codes, each stabilizer is a tensor product of Pauli operators, which we might refer to as a "Pauli string" for short. The default convention in the literature is that we take the sign, i.e. the eigenvalue upon applying the Pauli string to any codeword, to be $+1$.

Whether s below is measurement of any stabilizer(s) in the code (refer to the previous section for an example), or the tensor product of the single-qubit Pauli gates of the stabilizer(s), the following equation holds for all stabilizers s in the stabilizer group S:



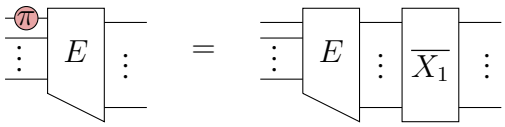
$\forall s \in S$ (4.17)

For any code, a non-unique choice of $n-k$ *stabilizer generators* is a set of linearly independent stabilizers, such that composing the projector for each stabilizer generator gives Π. This composition can be in any order because all stabilizers must commute with each other. If your measurement outcome of any stabilizer generator is $+1$, then the act of measurement implements the projector for that stabilizer corresponding to post-selecting that measurement; this means that if your physical state had a small amplitude of (continuous) error, the most probably outcome is that these measurements would project your physical state back into the code space. If you measure all stabilizer generators and observe at least one -1 outcome, this means that the measurement projected the state to outside the code space, and at least one error must have occurred. This discretization of continuous quantum errors due to measurement means that we can count the number of errors, as the number of undesirable single-qubit Pauli operators that occurred after measuring all stabilizer generators.

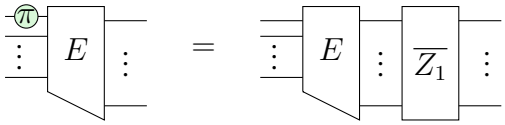
While many properties of codes can be derived without specifying how the logical states are mapped to the physical states, any implementation must make a choice on fixing the space of all *logical operators*. It suffices to specify $2k$ logical operators to fix any encoder map, which we will refer to as *the* logical operators. Conventionally, the easiest choice of a basis for the logical operators is to define physical implementations for Pauli X and Z on each logical qubit. Note that Pauli Y is not necessary, as it is a product of X and Z up to an irrelevant global

phase.

We depict the logical operators, drawn here on the first logical qubit, as [Kis22]:

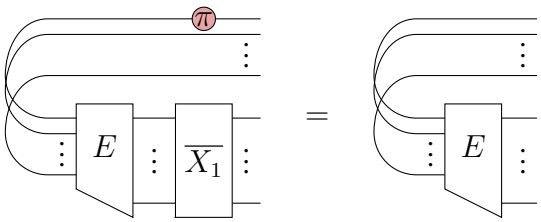


(4.18)

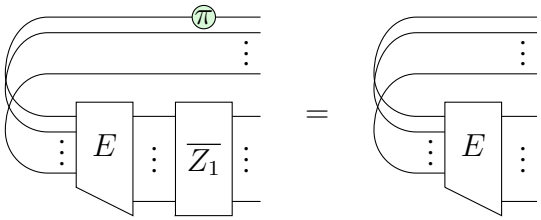


(4.19)

A helpful way to think about the logical operators is that they are stabilizers of the Choi state of the encoder map. If all of the logical operators are Pauli strings, then the encoder is a *Clifford* isometry (meaning that it can be decomposed as an n-qubit Clifford unitary acting on k identity wires tensored with $n-k$ $|0\rangle$ states); then, the logical operators of the $[\![n,k,d]\!]$ are stabilizer generators of the $[\![n+k,0,\infty]\!]$ "code", which is a stabilizer state of $n+k$ qubits and hence is fixed by $n+k=(n-k)+2k$ stabilizers.

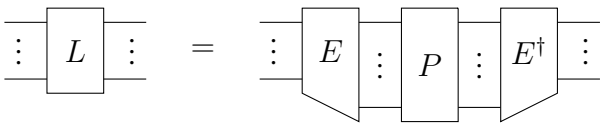


(4.20)



(4.21)

In general, any logical operation in the code is some map L on the logical qubits, which is physically implemented by a map P on the physical qubits, where

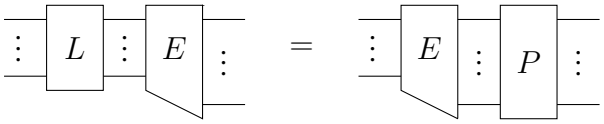


(4.22)

When $n>k$, there are many possible choices of P that implement the same L, because any operation in the stabilizer group of the code is identity on the logical qubits.

Equation (4.22) always holds if the following is satisfied, as $E^\dagger$ can be appended to both sides

of the equation below. We call this *pushing through the encoder* [HLY+23].



(4.23)

In the other direction,

Lemma 4.24. *Equation* (4.22) *implies Equation* (4.23) *only if* P *preserves the code space.*

Proof. We can see this by appending E to both sides of Equation (4.22):

$$L = E; P; E^\dagger \implies L; E = E; P; \Pi \tag{4.25}$$

Let us compare $E; P; \Pi$ to the right hand side of Equation (4.23), which is $E; P$. By pre-composing by $E^\dagger$, these match when $\Pi; P = \Pi; P; \Pi$; in other words, when P preserves the code space. □

Stabilizer codes admit a nice group-theoretic presentation. Any stabilizer code is equivalent to a *stabilizer group* $\mathcal{S}$, where $\mathcal{S} \subset \mathcal{P}_n$, the Pauli group on n qubits. The code space is the joint $+1$ eigenspace of $\mathcal{S}$.

A well-known lemma follows from the commutation relations of the stabilizer group:

Lemma 4.26. *Every operator in the stabilizer group must commute with all other elements in the stabilizer group.*

Proof. This follows from a proof by contradiction: Any two Pauli strings must either commute or anticommute. If two stabilizers $\vec{P}_1, \vec{P}_2$ anticommuted, then their product applied to any non-zero state in the code space $|\psi\rangle$ cannot be a stabilizer, as

$$\vec{P}_1\vec{P}_2 |\psi\rangle = -\vec{P}_2\vec{P}_1 |\psi\rangle = -\vec{P}_2 |\psi\rangle = - |\psi\rangle \tag{4.27}$$

□

Definition 4.28. The *centralizer* $\mathcal{C}(\mathcal{S})$ of a subset $\mathcal{S}$ in a group G is the set of elements in G which commute with every element of $\mathcal{S}$.

For any stabilizer code defined by the stabilizer group $\mathcal{S}$, all logical operators of the code make up $\mathcal{C}(\mathcal{S}) \setminus \mathcal{S}$, as all logical operators commute with all stabilizers of the code, but not with all other logical operators. For instance, the physical implementations of Pauli X and Z on the same logical qubit by definition must anticommute with each other.

All detectable errors of the code make up $\mathcal{P}_n \setminus \mathcal{C}(\mathcal{S})$, as these are all the Pauli operators which do not commute with at least one stabilizer of the code, and are hence detectable as at least one stabilizer generator must measure -1. All non-trivial undetectable errors of the code must be logical operators; trivial errors are physical errors that are equal to a stabilizer and hence does not cause a logical error.

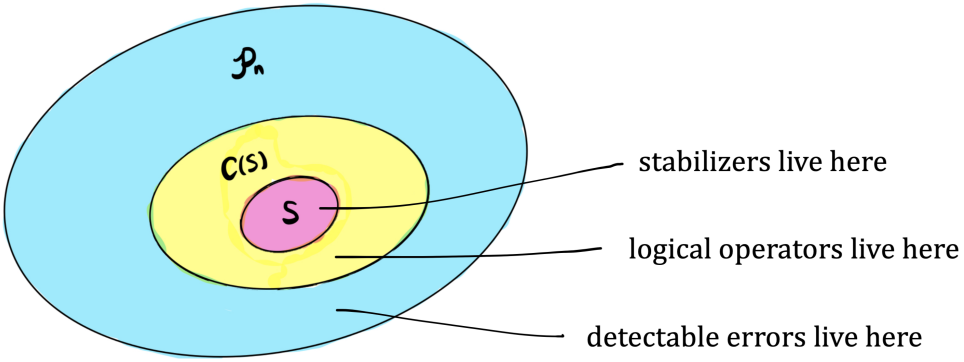


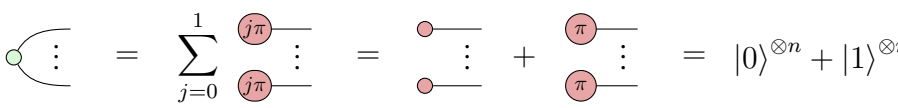
Figure 4.1: For any stabilizer code, the associated stabilizer group $\mathcal{S}$, its centralizer $\mathcal{C}(\mathcal{S})$, and the Pauli group on n qubits $\mathcal{P}_n$ satisfy the relation $\mathcal{S} \subset \mathcal{C}(\mathcal{S}) \subset \mathcal{P}_n$. The logical operators of the code make up $\mathcal{C}(\mathcal{S}) \setminus \mathcal{S}$, and the detectable errors of the code make up $\mathcal{P}_n \setminus \mathcal{C}(\mathcal{S})$.

4.4 CSS codes with the phase-free ZX-calculus

In Chapter 2, we discussed how the ZX-calculus is complete for the universal fragment of qubit quantum mechanics. Moreover, the ZX-calculus is complete for the *stabilizer fragment* (i.e. diagrams whose phases are all integer multiples of $\frac{\pi}{2}$) of qubit quantum mechanics [Bac14]. For now, we focus on CSS codes, a class of stabilizer codes with very nice properties named after Calderbank, Shor, and Steane [Ste96; CS96]. Precisely,

Definition 4.29. A *CSS code* is a stabilizer code whose stabilizer generators can be chosen in such a way, that all stabilizer generators are either over $\{X, I\}^{\otimes n}$ (called *X-type stabilizer generators*) or $\{Z, I\}^{\otimes n}$ (called *Z-type stabilizer generators*).

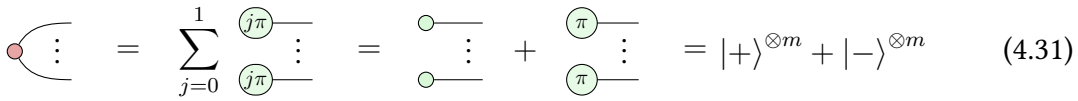
CSS codes are easy to represent in the ZX-calculus because their syndrome measurements are in the phase-free fragment of the ZX-calculus, i.e. diagrams whose phases are all zero [Kis22]. To start, we can observe that the m-qubit GHZ state (which is the Choi state of any phase-free Z spider) is a sum of two terms in the computational basis:



(4.30)

Flipping the colors, we can expand the $H^{\otimes m}|\text{GHZ}\rangle$ state (the Choi state of any phase-free X

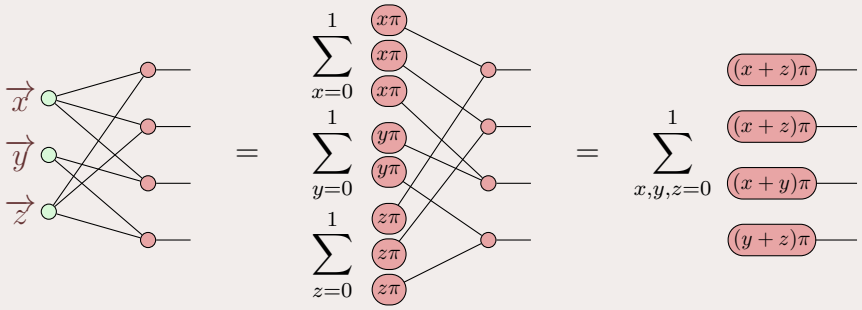
spider) as a sum of two terms in the X basis:



(4.31)

Example 4.32. Consider the leftmost ZX diagram below, whose three Z spiders are respectively identified with the binary vectors

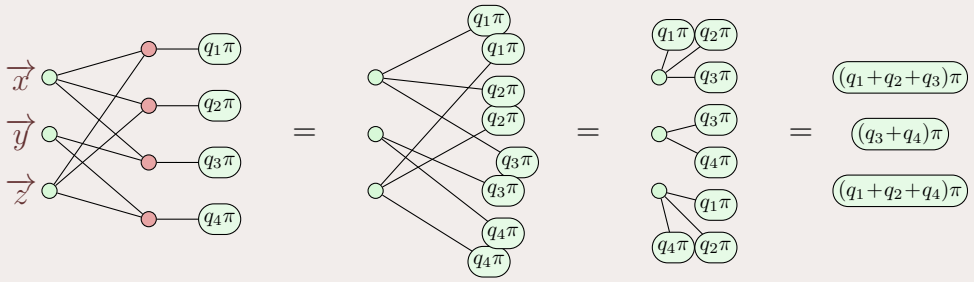
$$\vec{x} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \vec{z} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \tag{4.33}$$



(4.34)

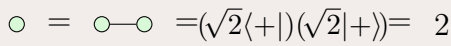
The $\vec{x}$, $\vec{y}$, and $\vec{z}$ labels are not necessary in the diagram, and exist simply to denote that the binary vector $\vec{x}$ is equal to the connectivity of that Z spider to the output qubits.

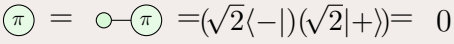
In addition to considering our original diagram expanded in the Z-basis, we can also consider how it acts as parity checks on X-basis states.



(4.35)

In the end, we get three scalars which are each nonzero only when they are of even parity. This is because 0-input, 0-output spiders are the scalar 0 if their phase (modulo 2π) is π, shown below for Z spiders, but the same holds for X spiders.



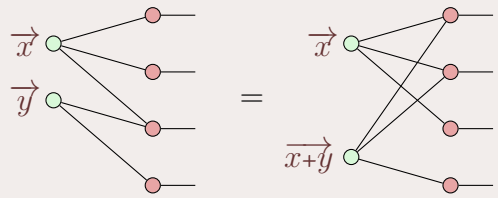


In other words, we have here a system of linear equations, the solutions to which are length 4 bitstrings $q_1q_2q_3q_4$. These correspond to those X-basis states $H^{\otimes 4}\left|q_1q_2q_3q_4\right\rangle$ which are not annihilated by the diagram.

$$H_X\vec{q} = \begin{bmatrix} \vec{x}^T \\ \vec{y}^T \\ \vec{z}^T \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \mod 2 \tag{4.36}$$

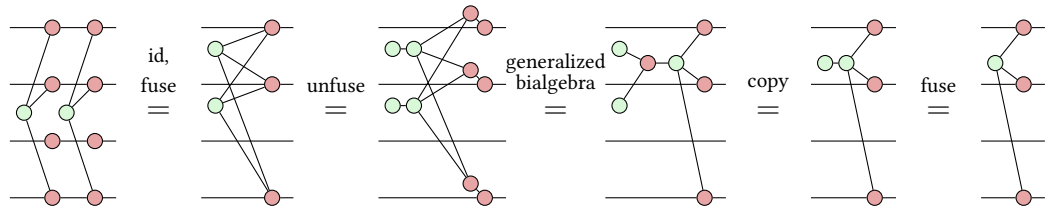
H_X is called the *parity check matrix*, where each row is a binary vector identified with an X-type stabilizer generator.

In fact, observe that the above system of equations is linearly dependent, and so only two of the equations are necessary; the third is redundant. Graphical proof of this is derivable by starting with the bialgebra rule [Kis22]. This enables replacing the biadjacency connectivity of a spider, by a sum of existing such connectivities, as one would do with systems of linear equations. For example, we can add the connectivities denoted by $\vec{x}$ and $\vec{y}$, to get a connectivity corresponding to $x \vec{+} y$.



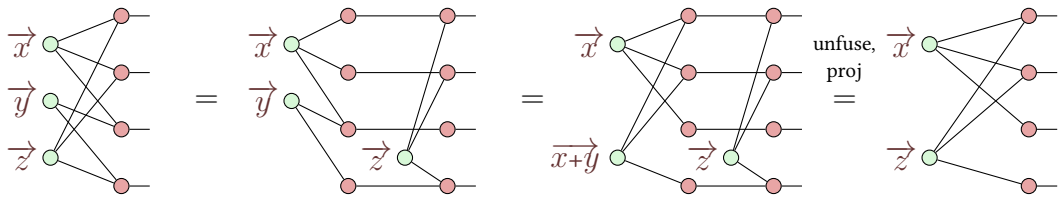
(4.37)

We can further show that consecutive diagrams representing the same stabilizer generator are redundant, which follows from $P^2 = P$ for any projector P.



(4.38)

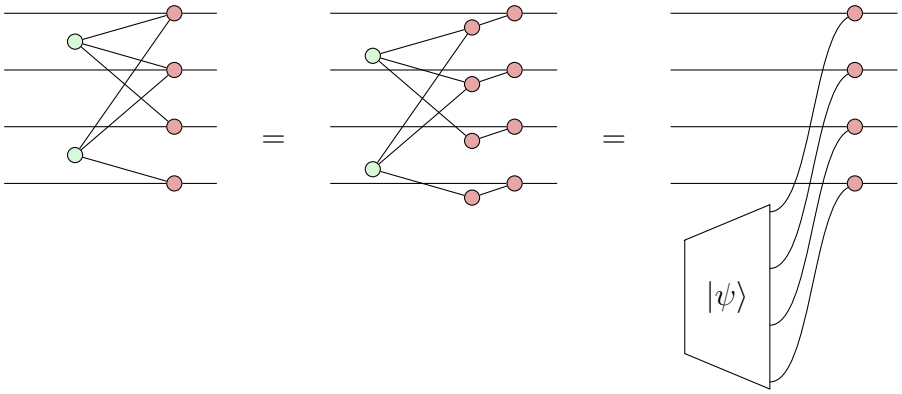
Substituting the above two equations, we can reduce our earlier system of equations.



(4.39)

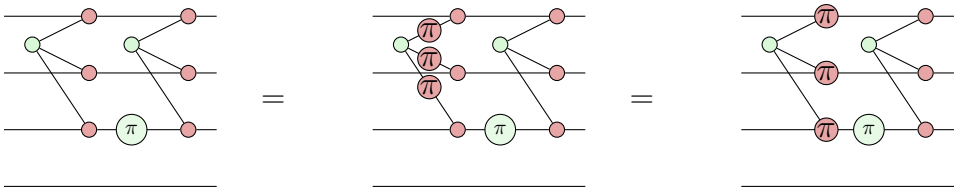
The code space of any CSS code expressed in either Z or X basis vectors, is each identifiable

with a state in the phase-free ZX-calculus. For instance, the state $|\psi\rangle$ above is identifiable with the code space spanned by $\{XXXI, IIXX\}$.



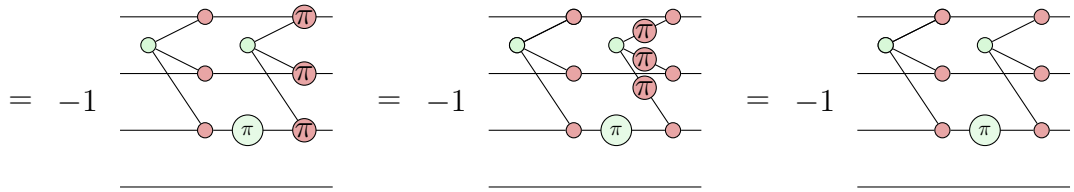
(4.40)

We can see that this (contrived toy example) code indeed detects phase-flip errors. Say that a phase-flip error occurs on the third qubit between two rounds of syndrome extraction. Presuming the syndrome measurements were not faulty, consider the two consecutive measurements of the $\{XXXI\}$ stabilizer.



(4.41)

Utilizing the fact that $ZX = -XZ$, we can proceed with the derivation:



(4.42)

What we found is that our original diagram, is equal to our original diagram times -1. How can this be? This can only be the case if the original diagram was equal to zero in the first place. Therefore, the probability of measuring +1 for the $\{XXXI\}$ stabilizer twice in a row, if between these measurements a phase-flip error occurred on the third qubit, is zero.

Remark 4.43. A helpful notation to streamline proofs like the above is to utilize Pauli webs and detecting regions. The basics are simple: For any spider, you can highlight all of its edges with the opposite color as that spider. You can also highlight any even number of its edges

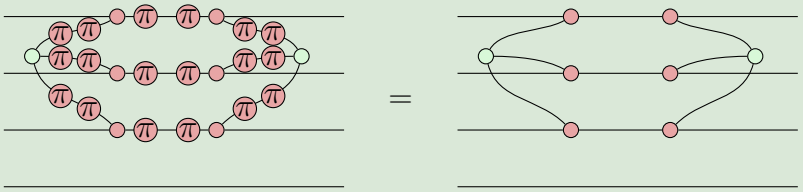
with the same color as that spider. To exemplify these two rules on a Z spider:



(4.44)

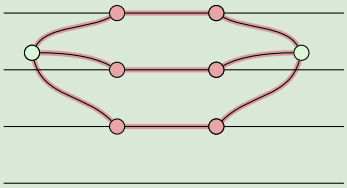
The name of the game is to find valid highlights, such that for every highlighted edge, both of their vertices (spiders) are consistent with these two rules.

Returning to our earlier example, we invoke these two rules for each spider of the diagram which is part of the $XXXI$ syndrome measurement.



(4.45)

The resulting highlighted detecting region is:



(4.46)

This is a detecting region because it is a Pauli web where no input nor output edges are highlighted.

The previous section covered an example of Z-type stabilizers, which protect against X (i.e. bit-flip) errors. In this section, we looked at X-type stabilizers, which protect against Z (i.e. phase-flip) errors.

CSS codes must protect against both Z and X errors in order to protect against errors projected to the Pauli group.

We can now introduce stabilizer tableaus for CSS codes represented by phase-free ZX diagrams. First, we give the general definition of stabilizer tableaus.

Definition 4.47. A *stabilizer tableau* of a stabilizer code is a binary matrix with m rows and $2n$ columns. Each row represents a stabilizer generator, where the left n columns denote which of the n qubits are acted on by Pauli X, and the right n columns denote which of the n qubits are acted on by Pauli Z. As $Y = XZ$ up to a global phase, qubits acted on by Pauli Y have 1s in that qubit's X position and Z position.

For example, the 4-qubit Pauli string $IXYZ$ is represented by $0110|0011$.

Remark 4.48. While the common convention is to take the sign of all stabilizers to be $+1$, more generally, if some of their eigenvalues are -1, these are indicated by 1s in an additional column of the stabilizer tableau.

Although not usually used in QEC, note that for some applications such as simulation, it is advantageous to also have twice as many rows to keep track of *destabilizer generators*—a basis of m Pauli strings which anticommute with the stabilizer group, marked by a 1 in an additional column of the tableau [AG04].

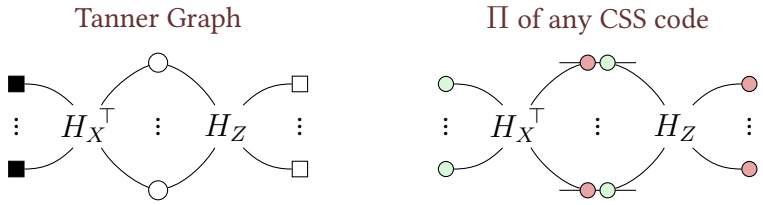
We can give another definition for CSS codes equivalent to Definition 4.29, in terms of their stabilizer tableaus.

Definition 4.49. CSS codes are stabilizer codes whose *stabilizer tableau* can be written in the following form, where H_X and H_Z are the parity check matrices for the X-type and Z-type stabilizer generators respectively.

$$T = \left(\begin{array}{c|c} H_X & 0 \\ 0 & H_Z \end{array}\right) \tag{4.50}$$

Equivalent to binary matrices, Tanner graphs are widely used for representing classical and CSS codes. A Tanner graph of a CSS code consists of two biadjacency graphs, whose connectivities are precisely $H_X^\top$ and H_Z. This is endowed with additional utility in the ZX-calculus because

Observation 4.51. The projector onto the code space Π is representable by a ZX diagram of precisely the same connectivity as the Tanner graph.



(4.52)

Definition 4.53. For any stabilizer code, its *complete stabilizer tableau* consists of a stabilizer tableau for the code, consisting of $n-k$ rows for the stabilizer generators, plus $2k$ more rows for the logical operators and k more columns (one for each logical qubit).

A helpful way to think about this is: If you take the Choi state of the encoder map uniquely specified by any complete stabilizer tableau, the stabilizer tableau of the resulting stabilizer state has precisely the same binary matrix as the complete stabilizer tableau, but the columns identified with the k inputs are instead identified with k additional outputs.

We will also need this definition of a transversal logical operation of a code.

Definition 4.54. A *transversal* logical operation in a code, is one that does not entangle any physical qubits in the same code block.

5

About qudits

The higher levels of many qubit devices are addressable and support operations making them viable as *qudits* — the generalization of qubits (for which $d = 2$) to d-dimensional Hilbert spaces. These enable applications including qudit algorithms [GSÇ+15; WHSK20], improved magic state distillation noise thresholds [Cam14], communication noise resilience [CDBO19], Bell inequality violation [VPB10], testing quantum gravity [TMMB21], and emulation of binary circuits [LBA+09a; GBD+19; NKF22]. Moreover, qubit circuit emulation of an arbitrary qudit unitary has been argued to be less asymptotically efficient than qudit circuit decomposition (except in the case where d is a power of 2) [BOB05].

Qudit computation has been proposed or experimentally demonstrated on a number of quantum processors [MRHW22] including cold atoms [ASR+15; KGH+21], nuclear magnetic resonance [GSÇ+15], nuclear spins [GFB+17], photonic quantum computing [RBC+18; HGL+18], Rydberg atoms [WUR+22], superconducting quantum computing [CBC+24a; BRS+21; YZZY18; YSK+20; HHDR21], and trapped ions [RMP+22a]. As of recent, error rates of universal qudit processors were shown to be competitive relative to qubit processors [RBC+18; CHZ+22], and maximally entangling qudit gates were physically realized [HWG+23].

Several concepts for qubits extend to qutrits, or more generally to qu*dits*, which are d-dimensional quantum systems — in particular, the concept of Pauli's and Cliffords. For a d-dimensional qudit, we follow the standard qudit definition of the Pauli Z and X gates, colloquially called

the clock and shift operators, as [Got99; HV12]

$$Z\,|k\rangle = \omega^k\,|k\rangle \qquad\qquad X\,|k\rangle = |k+1\rangle \tag{5.1}$$

where $k \in \mathbb{Z}_d$. Here $\omega := e^{2\pi i/d}$ is such that $\omega^d = 1$, and the addition $|k+1\rangle$ is taken modulo d. For this reason, in circuit notation we adopt the shorthand of $+1$ being the X gate and $+k$ being the X^k gate. The *Pauli group* is the set of unitaries generated by tensor products of the X and Z gate.

Another concept that readily generalizes to qudits is that of Clifford unitaries.

Definition 5.2. Let U be a qudit unitary acting on n qudits. We say it is *Clifford* when every Pauli is mapped to another Pauli under conjugation by U, i.e. if $UPU^\dagger$ is in the Pauli group for any Pauli P.

The set of n-qudit Cliffords forms a group under composition. For qubits, this group is generated by the S, H and CX gates. The same is true for prime-dimensional qudits, for the right generalization of these gates[1].

For qubits the X gate is the NOT gate, while $Z = \mathrm{diag}(1, -1)$. For a qubit there is only one non-trivial permutation of the standard basis states, implemented by the X gate. For qutrits, there are five ternary X gates, as there are five non-trivial permutations of the Z-basis states: 2 of which are Pauli X and $X^\dagger$, and the other 3 exchange exactly 2 of the 3 qutrit Z-basis states. As the prime-dimensional qudit dimension d increases, a growing number of permutations of single-prime-dimensional qudit Z-basis states fall outside the single-prime-dimensional qudit Clifford unitaries.

Just as the Z-basis states labelled by elements of $\mathbb{Z}_d$ are eigenstates of the Z operator, the X-basis states are eigenstates of the X operator. We denote the $|+\rangle$ state as the X-basis state with eigenvalue 1,

$$|+\rangle = \frac{1}{\sqrt{d}} \sum_{j=0}^{d\text{-}1} |j\rangle\,. \tag{5.3}$$

We now define the Clifford gate set, which is a set of three prime-dimensional qudit gates, $\{S, H, CX\}$, that generates all Clifford unitaries [Got99]. On Z-basis states $|k\rangle$, the prime-dimensional qudit S gate acts as

$$|k\rangle \mapsto \omega^{k(k\text{-}1)/2}\,|k\rangle\,, \tag{5.4}$$

[1]The gate definitions for various qudit Cliffords may vary across the literature up to a global phase. Indeed, by Definition 5.2, whether a gate is Clifford is invariant under changes in global phase.

the H gate acts as

$$|k\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{j=0}^{d\text{-}1} \omega^{jk} |j\rangle , \tag{5.5}$$

and the CX gate performs addition modulo d, acting as

$$|j\rangle |k\rangle \mapsto |j\rangle |(j+k) \bmod d\rangle . \tag{5.6}$$

A more helpful view of the H gate is that it sends each X-basis state to the Z-basis state with the same eigenvalue, hence it is also referred to as the discrete quantum Fourier transform.

To distinguish it from non-Clifford controlled X gates, we adopt the shorthand of labelling the control of the (Clifford) CX gate by Λ [BRS17] and the target by X or by $+1$.

Definition 5.7. Given a qudit unitary U we define

$$\Lambda(U) |c\rangle |t\rangle = |c\rangle \otimes (U^c |t\rangle), \tag{5.8}$$

i.e. we apply the unitary U a number of times equal to to the value of the control qudit.

We define the $\Lambda(+k) \coloneqq \Lambda(X^k)$ gate to be the repetition of the CX gate $k \in \mathbb{Z}_d$ times.

5.1 Diagonal gates in the Clifford hierarchy

Recall that we defined the Clifford hierarchy in Definition 3.21; the same definition holds irrespective of qudit dimension. Recently, it was proven by deSilva and Lautsch that for qudits of any prime dimension, any single-qudit unitary in any level of the Clifford hierarchy is semi-Clifford and therefore admits an efficient state injection protocol [dSL25].

Despite the third level of the Clifford hierarchy and higher not being groups (for instance they are not closed under composition), the Paulis, Cliffords, and the diagonal gates in the Clifford hierarchy are groups. This facilitated characterization of all diagonal gates in the Clifford hierarchy for arbitrary prime qudit dimension d by Cui, Gottesman, and Krishna [CGK17]. In particular, they identified that gates of the form

$$U_{m,a} = \sum_{j=0}^{d\text{-}1} \exp\left(\frac{2\pi i}{d^m} j^a\right) |j\rangle \langle j| \tag{5.9}$$

lie in the $((d\text{-}1)(m\text{-}1)+a)^{\text{th}}$ level of the Clifford hierarchy [CGK17, Equation 16].

All single-qudit operations in the third level of the Clifford hierarchy admit robust state injection protocols, as commutation with a Pauli gate requires a Clifford correction. The most well-

studied single-qubit non-Clifford gate, the T gate $\mathrm{diag}(1, \exp(i\frac{2\pi}{8}))$, rests in the third level of the Clifford hierarchy, and admits fault-tolerant implementation, such as through magic state distillation and state injection, or through transversal implementation on certain stabilizer codes. The T gate has been generalized to prime dimensions by means of explicitly solving for the diagonal single-qudit gates in the third level of the Clifford hierarchy [HV12]. This exact solution involved the multiplicative inverse of 12 modulo d, and thus yielded the same form for all prime dimensions with the exceptions of $d = 2$ and $d = 3$.

The qutrit T gate is defined as $\mathrm{diag}(1, \omega^{1/3}, \omega^{-1/3})$ [HV12], or a diagonal gate which is Clifford equivalent [BRS17]. Like the qubit T gate, the qutrit T gate belongs to the third level of the Clifford hierarchy, can be injected into a circuit using magic states, and its magic states can be distilled by magic state distillation [CAB12]. This means that we can fault-tolerantly implement this gate on many qutrit error correcting codes. Also, as is for qubits, the qutrit Clifford+T gate set is approximately universal, meaning that we can approximate any qutrit unitary using just Clifford gates and the T gate [CW15, Theorem 1]. More generally, adding a single-prime-dimensional qudit non-Clifford gate to the Clifford gate set yields approximately universal quantum computation.

The T gate is special for $d = 2$ and for $d = 3$. For all other prime dimensions, exactly solving for the single-qudit diagonal gates in the third level of the Clifford hierarchy yields a family of gates all satisfying the same explicit expression [HV12], of which a sensible choice is $U_{1,3}$ in Equation (5.9) [KT19].

5.2 Qutrit Hamiltonians and Gell-Mann matrices

In addition to being the bottom of the Clifford hierarchy, another feature of the qubit Pauli matrices is that they are traceless, Hermitian, and unitary; and form a basis for that which can be exponentiated to any two-qubit unitary matrix. Unfortunately, for qutrits (as is the case for qudit dimensions higher than 2), there are two different ways to generalize the qubit Paulis. One is the qudit Paulis (which are traceless and unitary, but not Hermitian), and the other is the Gell-Mann matrices (which are traceless and Hermitian, but not unitary).

Taking the convention of matrix elements left blank being zero, the Gell-Mann matrices are defined as follows.

Definition 5.10. The Gell-Mann matrices are

$$\lambda_1 = \begin{bmatrix} 0 & 1 & \cdot \\ 1 & 0 & \cdot \\ \cdot & \cdot & 0 \end{bmatrix}, \quad \lambda_2 = \begin{bmatrix} 0 & -i & \cdot \\ i & 0 & \cdot \\ \cdot & \cdot & 0 \end{bmatrix}, \quad \lambda_3 = \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & -1 & \cdot \\ \cdot & \cdot & 0 \end{bmatrix},$$

$$\lambda_4 = \begin{bmatrix} 0 & \cdot & 1 \\ \cdot & 0 & \cdot \\ 1 & \cdot & 0 \end{bmatrix}, \quad \lambda_5 = \begin{bmatrix} 0 & \cdot & -i \\ \cdot & 0 & \cdot \\ i & \cdot & 0 \end{bmatrix}, \quad \lambda_6 = \begin{bmatrix} 0 & \cdot & \cdot \\ \cdot & 0 & 1 \\ \cdot & 1 & 0 \end{bmatrix},$$

$$\lambda_7 = \begin{bmatrix} 0 & \cdot & \cdot \\ \cdot & 0 & -i \\ \cdot & i & 0 \end{bmatrix}, \quad \lambda_8 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \\ \cdot & \cdot & -2 \end{bmatrix}$$

In the qubit setting, exponentiating the complex number i times a qubit Pauli matrix P (i.e. e^{iP}) is equal to i times P:

$$e^{i\frac{\pi}{2}P} = iP \tag{5.11}$$

In addition to forming a fundamental basis for the Lie algebra $\mathfrak{su}(2)$, the qubit Pauli matrices are a basis for the 2×2 Hermitian matrices.

Due to the difference between the qutrit Gell-Mann matrices and the qutrit Pauli matrices, you cannot exponentiate the former (a basis for the Hermitian operators) to get the latter (a basis for the single-qudit unitary operators). This is inconvenient because hardware realizations tune Hamiltonians defined on the former, while quantum circuits and channels are defined on the latter.

Any unitary matrix U satisfies $U = e^{i\mathcal{H}}$, where $\mathcal{H}$ is a Hermitian matrix. To find $\mathcal{H}$, decompose U as $U = V\mathcal{D}V^{-1}$ where V is the eigenvector matrix and D is the eigenvalue diagonal matrix. You can rewrite $\mathcal{D}$ in the form $\mathcal{D} = e^{iA}$ where A is also a diagonal matrix. So $U = e^{iVAV^{-1}} = e^{i\mathcal{H}}$ for $\mathcal{H}$ Hermitian. Any 2×2 Hermitian matrix can be decomposed as a real linear combination of Pauli matrices. Likewise, any 3×3 Hermitian matrix can be decomposed as a real linear combination of the 8 Gell-Mann matrices [Gel62].

Example 5.12 (Example: Exponential form of X_{+1})**.** $U = X_{+1}$ has the matrix

$$U = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{5.13}$$

$U = e^{i\mathcal{H}}$ for Hermitian matrix

$$\mathcal{H} = \begin{bmatrix} 0 & -\frac{2\pi}{3\sqrt{3}}i & \frac{2\pi}{3\sqrt{3}}i \\ \frac{2\pi}{3\sqrt{3}}i & 0 & -\frac{2\pi}{3\sqrt{3}}i \\ -\frac{2\pi}{3\sqrt{3}}i & \frac{2\pi}{3\sqrt{3}}i & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{\sqrt{3}}i & \frac{1}{\sqrt{3}}i \\ \frac{1}{\sqrt{3}}i & 0 & -\frac{1}{\sqrt{3}}i \\ -\frac{1}{\sqrt{3}}i & \frac{1}{\sqrt{3}}i & 0 \end{bmatrix} \tag{5.14}$$

$\mathcal{H}$ can be decomposed as a real linear combination of the Gell-Mann matrices

$$\mathcal{H} = \frac{1}{\sqrt{3}}(\lambda_2 - \lambda_5 + \lambda_7) \tag{5.15}$$

For reference, in Table 5.1 and Table 5.2 below are the coefficients of the Gell-Mann matrices for qutrit Clifford unitaries.

U	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
I	0	0	0	0	0	0	0	0
X_{+1}	0	$-\frac{1}{\sqrt{3}}$	0	0	$\frac{1}{\sqrt{3}}$	0	$-\frac{1}{\sqrt{3}}$	0
X_{-1}	0	$\frac{1}{\sqrt{3}}$	0	0	$-\frac{1}{\sqrt{3}}$	0	$\frac{1}{\sqrt{3}}$	0
Z_{+1}	0	0		0	0	0	0	$\frac{\sqrt{3}}{2}$
Z_{-1}	0	0		0	0	0	0	$-\frac{\sqrt{3}}{2}$
S	0	0		0	0	0	0	$\frac{1}{2\sqrt{3}}$
$S^\dagger$	0	0		0	0	0	0	$-\frac{1}{2\sqrt{3}}$
H	$-\frac{\sqrt{3}}{4}$	0	$-\frac{3\sqrt{3}}{16}$	$-\frac{\sqrt{3}}{4}$	0	$\frac{\sqrt{3}}{8}$	0	$-\frac{3}{16}$
$H^\dagger$	$\frac{\sqrt{3}}{4}$	0	$\frac{3\sqrt{3}}{16}$	$\frac{\sqrt{3}}{4}$	0	$-\frac{\sqrt{3}}{8}$	0	$\frac{3}{16}$

Table 5.1: Coefficients $c_1, \ldots c_8$ of the 8 Gell-Mann matrices in the exponentiation of qutrit unitary U. The convention of omitting a factor of $\frac{2\pi}{3}$ is taken.

Just as there is a group homomorphism from $SU(2)$ to $SO(3, \mathbb{R})$, there is a group homomorphism from $SU(3)$ to $SO(8, \mathbb{R})$. This generalizes to there being a group homomorphism from $SU(d)$ to $SO(d^2 - 1, \mathbb{R})$, for any dimension d. To put it another way, any single-qutrit unitary (specified by a 3×3 complex-valued matrix in $SU(3)$) has an alternate representation as an 8×8 real-valued matrix in $SO(8)$.

While the Gell-Mann basis is the most well-studied choice of basis for qutrit computation, a number of other choices are interesting and highly useful. Ref. [CLB+24] slightly modifies the Gell-Mann matrices by local phases, to correspond to the gates implementable natively on their superconducting transmon architecture. For single-qutrit exact circuit synthesis, Ref. [GRT19] defines an *adjoint representation* as the group homomorphism to a different choice of basis, by constructing Hermitian operators by symmetrizing the Paulis. We utilize this representation in Ref. [GRvdWY22a] to elucidate the relationship between two qutrit gate sets.

CZ_{+1}	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8
λ_1	0	0	0	0	0	0	0	0
λ_2	0	0	0	0	0	0	0	0
λ_3	0	0	$\frac{1}{4}$	0	0	0	0	$-\frac{\sqrt{3}}{4}$
λ_4	0	0	0	0	0	0	0	0
λ_5	0	0	0	0	0	0	0	0
λ_6	0	0	0	0	0	0	0	0
λ_7	0	0	0	0	0	0	0	0
λ_8	0	0	$-\frac{\sqrt{3}}{4}$	0	0	0	0	$\frac{3}{4}$

Table 5.2: Coefficients of each pairwise tensor product of the 8 Gell-Mann matrices in the exponentiation of 2-qutrit unitary CZ_{+1}. The convention of omitting a factor of $\frac{2\pi}{3}$ is taken.

5.3 Rings and things

Definition 5.16 (Exact representation of unitaries)**.** A unitary matrix U can be *exactly represented* by a circuit over a gate set G if there exists a circuit C over G such that $C = U$.

Definition 5.17 (Approximate representation of unitaries)**.** A unitary matrix U can be *approximately represented* by a circuit over a gate set G if for any fixed precision δ, there exists a circuit C over G such that the operator norm $\|C - U\| \leq \delta$.

It turns out that the size of this approximation is not unfeasibly large: polynomial in $\log(\frac{1}{\delta})$. It also turns out that if you have an m-gate circuit (where the number of qubits each gate acts on is bounded by a constant) which exactly represents this unitary, the size of the approximate circuit can be guaranteed to be $O\left(m\log^c(\frac{m}{\delta})\right)$ for some constant c [NC10].

Definition 5.18. We say that (the set of unitaries over) a gate set $\mathcal{G}_1$ is a *subset* of (that of) another gate set $\mathcal{G}_2$, if for every circuit over $\mathcal{G}_1$, there exists a circuit over $\mathcal{G}_2$ implementing the exact same unitary. In other words, $\mathcal{G}_1 \subset \mathcal{G}_2$ means that the set of matrices that are exactly synthesizable from $\mathcal{G}_1$ is a subset of that from $\mathcal{G}_2$. When there exist matrices exactly synthesizable from $\mathcal{G}_2$ but impossible from $\mathcal{G}_1$, we say that this subset relation $\mathcal{G}_1 \subsetneq \mathcal{G}_2$ is a *strict subset*.

Definition 5.19. Two gate sets $\mathcal{G}_1$ and $\mathcal{G}_2$ are *equivalent* if $\mathcal{G}_1 \subset \mathcal{G}_2$ and $\mathcal{G}_2 \subset \mathcal{G}_1$.

Definition 5.20 (Identifying rings with gate sets)**.** We say a number ring R is *identified with* a gate set G if:

- every circuit over G has all its matrix elements in R, and
- every unitary matrix M with all its matrix elements in R is exactly representable by a circuit over G.

Typically, this is constrained to unitary matrices corresponding to possible sizes of circuits in your gate set. For instance, n-qubit circuits correspond to unitary matrices of size $2^n \times 2^n$, and n-qudit circuits to size $d^n \times d^n$.

Regarding the first bullet point: By looking at the generating gates of G, it is often not difficult to ascertain the smallest ring R which contains all the matrix elements of M. Since matrix multiplication only adds and multiplies elements of R, which are operations the ring is closed under, any circuit over G will also have all its matrix elements in R [KMM13]. For this reason, the green arrow in Figure 5.1 is marked as 'easy'.

The more difficult question is the reverse, the second bullet point, marked as 'hard'. The usual way of proving this is by giving an algorithm called the *exact synthesis algorithm*, which takes in a matrix and outputs a circuit over G which implements it. Finding such an algorithm is the *exact synthesis problem*. Note that this is first concerned with the existence of such an algorithm, which verifies a gate set and a ring can be identified with each other. The efficiency of the algorithm is a separate subsequent question of practical importance, which in the qubit case has seen much improvement since the first multiqubit exact synthesis result [Kli13; NWD20; KS24].

Definition 5.21 (The Exact Synthesis Problem)**.** The *exact synthesis problem* is: Given a unitary matrix M where all elements of the matrix are in a certain number ring R, find a quantum circuit C over a certain gate set G such that C implements M.

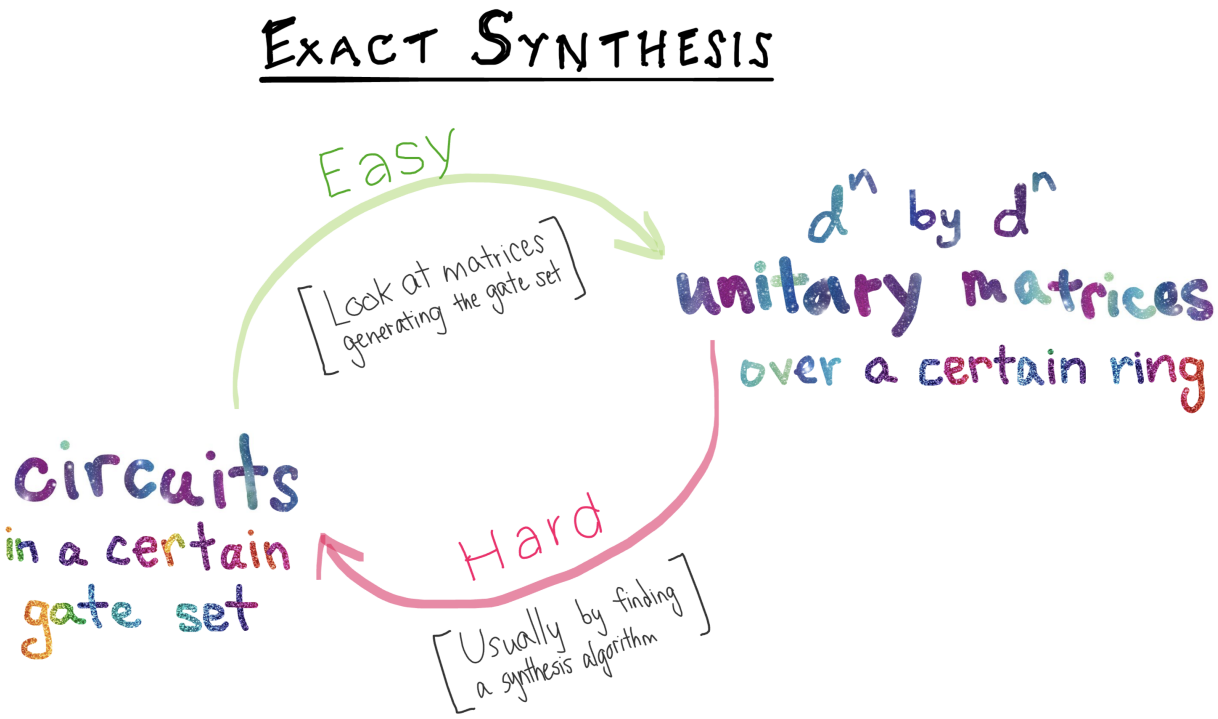


Figure 5.1: Depiction of the exact synthesis problem for quantum computation on d-dimensional qudits.

The exact synthesis problem was famously solved for multiqubit Clifford+T circuits for the first time in 2013 by Giles and Selinger [GS13]. However, prior to around a year and a half ago, it was unsolved for multiqudit circuits over a universal gate set for any qudit dimension greater than 2. Part IV of this thesis culminates in a solution to this problem for any qutrit Toffoli+Hadamard, any qutrit Clifford+T circuit, and any gate set situated above these two in the qutrit Clifford-cyclotomic hierarchy.

In this section, we set out the necessary prerequisites of ring theory for the exact synthesis problem.

Definition 5.22 (Extending a ring)**.** A ring R is *extended* by α, for some α in a larger ring containing R, to the ring

$$R[\alpha] = \left\{ \sum_j r_j \alpha^j | r_j \in R \right\} \tag{5.23}$$

For these additions and multiplication by α to be well-defined, α must be an element of a field containing the ring R.

Definition 5.24 (Localizing a ring)**.** A ring R is *localized* by μ, for μ a multiplicatively-closed

subset of R containing 1 but not 0, to the ring

$$\mu^{-1}R = \left\{ \frac{r}{m} | r \in R \text{ and } m \in \mu \right\} \tag{5.25}$$

Let $\mu_\beta \coloneqq \{\beta^k | k \in \mathbb{N}\}$ for some number β. The ring identified with the qubit Clifford+T gate set can be obtained by first extending the integers by i and $\sqrt{2}$, then localizing by powers of $\sqrt{2}$:

$$\mu_{\sqrt{2}}^{-1}\left(\mathbb{Z}[i, \sqrt{2}]\right) = \left\{ \frac{a + bi + c\sqrt{2} + di\sqrt{2}}{2^n} | a, b, c, d \in \mathbb{Z} \text{ and } n \in \mathbb{N} \right\} \tag{5.26}$$

This same ring can also be obtained by first localizing the integers by powers of 2, then extending by $\sigma = e^{i\frac{\pi}{4}} = \frac{1+i}{\sqrt{2}}$:

$$\left(\mu_2^{-1}\mathbb{Z}\right)[\sigma] = \left\{ a\sigma^3 + b\sigma^2 + c\sigma + d | a, b, c, d \in \mu_2^{-1}\mathbb{Z} \right\} \tag{5.27}$$

The identification of this ring with qubit Clifford+T was conjectured by Kliuchnikov, Maslov, and Mosca [KMM13] and proven by Giles and Selinger [GS13].

A high-level overview of the approach Giles and Selinger took to solve the multiqubit Clifford+T exact circuit synthesis problem is given by the steps below. For any element u of the ring defined (twice) above, applying the *residue map*

$$\rho : \mathbb{Z}[\sigma] \to \mathbb{Z}_2[\sigma] \tag{5.28}$$

to u, gives a length 4 bitstring $x = \rho(u)$, where each bit is respectively a, b, c, and d of Equation (5.27) taken modulo 2. Complex conjugation is denoted as $x^\dagger = \rho(u^\dagger)$, scalar multiplication by a scalar s is denoted as $sx = \rho(su)$, and the squared norm is denoted as $x^\dagger x = \rho(u^\dagger u)$. We match the notation of the source paper, except denote $e^{i\frac{\pi}{4}}$ by σ instead of ω because in this thesis, we reserve ω to denote $e^{i\frac{2\pi}{3}}$.

Procedure 1: Steps to achieving the original multiqubit Clifford+T exact synthesis result

Input: A unitary matrix of size $2^n \times 2^n$ over the ring
$\left(\mu_2^{-1}\mathbb{Z}\right)[\sigma] = \left\{a\sigma^3 + b\sigma^2 + c\sigma + d | a, b, c, d \in \mu_2^{-1}\mathbb{Z}\right\}$

Output: An n-qubit circuit over the Clifford+T gate set

1. Residue operations: For all possible residues $x \coloneqq \rho(u)$, represent $\{x^\dagger, \sigma x, \sqrt{2}x, x^\dagger x\}$ as operations on residues.
2. x is reducible (i.e. divisible by $\sqrt{2}$) iff $\sqrt{2}x = 0000$ iff $x^\dagger x = 0000$.
3. k is the least denominator exponent (LDE) of x iff $\rho(\sqrt{2}^k u)$ is irreducible.
4. Row operation: Determine how to find 2 entries in the same column vector, such that applying the two-level H operator to them reduces both their LDEs by 1.
5. Column lemma: For each column, do induction on LDE($\vec{u}$), until $k = 0$ and $|\vec{u}\rangle \in \mathbb{Z}[\sigma]^n$. Then $1 = \langle\vec{u}|\vec{u}\rangle = \sum_j u_j^\dagger u_j$, where all $u_j^\dagger u_j \in \{0, 1\}$.

Definition 5.29 (Cyclotomic polynomials)**.** The nth cyclotomic polynomial $\Phi_n(x)$ is

$$\Phi_n(x) = \prod_{\substack{1 \le k \le n-1, k \text{ coprime to } n \\ \gcd(k,n)=1}} \left(x - e^{i\frac{2\pi k}{n}}\right) \tag{5.30}$$

Definition 5.31 (Clifford-cyclotomic)**.** Let d be a prime number and let $k \in \mathbb{N}$. For qudits of dimension d, the Clifford-cyclotomic gate set of degree d^k is the universal gate set obtained by extending the Clifford gates with the gate $T_k = \text{diag}(1, \omega_k, {\omega_k}^2, ...{\omega_k}^{d-1})$, where ω_k is a primitive (d^k)th root of unity.

We have that for all $k \in \mathbb{N}$, $\mathcal{G}_k \subset \mathcal{G}_{k+1}$.

Exact synthesis for the multiqubit Clifford-cyclotomic gate set of arbitrary n, was proven in Ref. [AGK+24].

Observation 5.32. The qutrit Paulis are over the ring

$$\mathbb{Z}[\omega] = \{a + b\omega | a, b \in \mathbb{Z}\} \tag{5.33}$$

where $\omega = e^{i\frac{2\pi}{3}}$ is the third root of unity.

Observation 5.34. Let $\mu_3 \coloneqq \{3^k | k \in \mathbb{N}\}$. The qutrit Cliffords are over the ring

$$\mathbb{T}[\omega] = \{a + b\omega | a, b \in \mathbb{T}\} \tag{5.35}$$

where

$$\mathbb{T} = {\mu_3}^{-1}\mathbb{Z}\left\{\frac{a}{3^k} | a \in \mathbb{Z}, k \in \mathbb{N}\right\} \tag{5.36}$$

is called the *ring of triadic fractions.*

Observation 5.37. The qutrit Clifford+T gate set is over the ring

$$\mathbb{T}[\omega, \zeta] \approx \mathbb{T}[\zeta] = \left\{a + b\zeta + c\zeta^2 + d\zeta^3 + e\zeta^4 + f\zeta^5 | a, b, c, d, e, f \in \mathbb{T}\right\} \tag{5.38}$$

where $\zeta = e^{i\frac{2\pi}{9}}$ is the ninth root of unity.

The qutrit Clifford and Clifford+T gate sets are respectively identified with the cyclotomic polynomials $\Phi_3(x) = x^2 + x + 1$ and $\Phi_9(x) = x^6 + x^3 + 1$. For $k \geq 1$, as the roots of Φ_{3^k} are (3^k)th roots of unity, the matrix elements of each Clifford-cyclotomic gate set $\mathcal{G}_k$ are in the ring:

$$({\mu_3}^{-1}\mathbb{Z})[e^{i\frac{2\pi}{3^k}}] = \left\{a + b\zeta + c\zeta^2 + d\zeta^3 + e\zeta^4 + f\zeta^5 | a, b, c, d, e, f \in {\mu_3}^{-1}\mathbb{Z}\right\} \tag{5.39}$$

Unitaries with matrix elements outside $({\mu_3}^{-1}\mathbb{Z})[e^{i\frac{2\pi}{3^k}}]$ necessarily cannot be exactly synthesized from $\mathcal{G}_k$. However, this does not preclude the existence of unitary matrices with all elements in $({\mu_3}^{-1}\mathbb{Z})[e^{i\frac{2\pi}{3^k}}]$ that are impossible to exactly synthesize from $\mathcal{G}_k$. Proving that no such unitaries exist, i.e. all unitary matrices over $({\mu_3}^{-1}\mathbb{Z})[e^{i\frac{2\pi}{3^k}}]$ are exactly synthesizable from $\mathcal{G}_k$, is to solve the exact synthesis problem for all qutrit Clifford-cyclotomic gate sets of degree 3^k. We prove this result in Chapter 13 of this thesis.

Part III

Qubitland

Chapter Contents

6 Computational representations of stabilizer theory **81**

6.1 Quantum error correcting codes 81

6.2 The Phase-free Z-X Generalized Parity Form (GPF) 81

6.3 The RREF_X Normal Form for Complete Stabilizer Tableau Rows 85

6.4 Circuit Extraction 88

6.5 Transversal Gates and Magic State Distillation 93

6.6 Sample: Code switching and morphing 100

7 Isometries in the stabilizer fragment **103**

7.1 Graph State with Local Cliffords Normal Form for Graph Codes 104

7.2 Affine with Phases Normal Form for stabilizer codes 106

7.3 The stabilizer tableau equivalent of AP normal form 108

7.4 Adapting AP normal form to isometries 110

6

Computational representations of stabilizer theory

circa 2025, colorized

6.1 Quantum error correcting codes

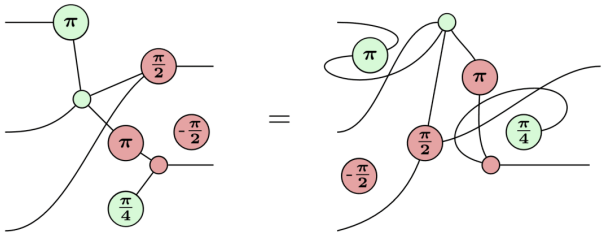
Remark 6.1. This section is based on the manuscript in preparation "A Three-Way Normal Form for Stabiliser Codes across ZX Diagrams, Circuits, and Tableaus" [HKL+26]. This is joint work with Jiaxin Huang, Aleks Kissinger, Sarah Meng Li, and John van de Wetering.

6.2 The Phase-free Z-X Generalized Parity Form (GPF)

In this section, we will introduce the Z-X Generalized Parity Forms. They are the normal forms for phase-free ZX calculus diagrams for an encoder of any CSS code. X-Z Generalized Parity Forms are defined by exchanging the roles of X and Z spiders.

Definition 6.2. We say a spider is an input (output) spider when it is connected to at least one input (output).

For instance, the following ZX-diagrams both have 3 inputs (open wires to the left) and 2 outputs (open wires to the right). The $\frac{\pi}{2}$ phase X spider is the only spider which is both an input and an output spider. The π phase X spider, the $-\frac{\pi}{2}$ phase X spider, and the $\frac{\pi}{4}$ phase Z spider are the only spiders which are neither input nor output spiders.

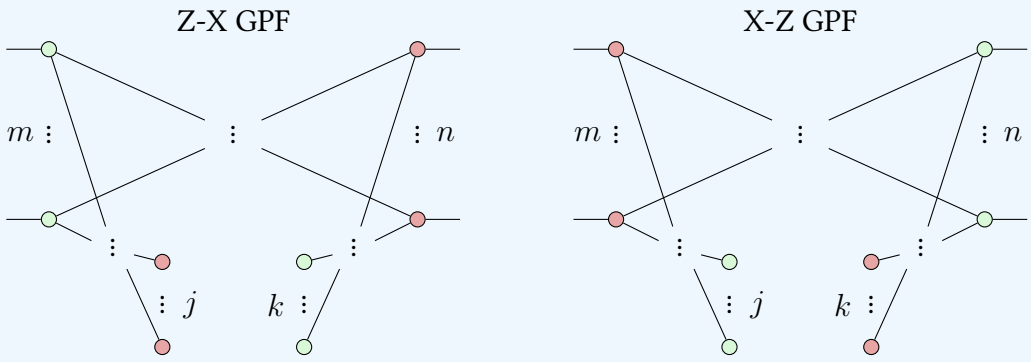


We start with the definition of the Z-X and X-Z Generalized Parity Forms from Ref. [KvdW24a]. Ref. [KvdW24a] also gives an efficient algorithm to rewrite any phase-free ZX-diagram into either of these forms, which we will not need the specifics of here other than to note its existence.

Definition 6.3. We say a phase-free ZX-diagram is in Z-X generalized parity form (GPF) when it is two-coloured and

1. every input is connected to a Z-spider and every output to an X-spider,
2. no two spiders of the same colour are connected,
3. there are no zero-arity (i.e. scalar) spiders, and
4. no non-input Z-spider is connected to a non-output X-spider.

Likewise, a phase-free ZX-diagram is in X-Z generalized parity form when, with the letters Z and X are exchanged, all above conditions are satisfied. Drawing the above conditions, we get that any phase-free ZX diagram in GPF must look like:



When we have subdiagrams which are bipartite graphs, one option is to label the wires across the bipartition with its biadjacency matrix: a matrix over $\mathbb{F}_2$ where each column corresponds

to a spider on the input side of the bipartition, and each row to a spider on the output side. A matrix entry is 1 if there is an edge between the corresponding spiders, and 0 otherwise.

Remark 6.4. A special case of GPF is when the linear map is unitary, i.e. $m = n$, $j = k = 0$, and the biadjacency matrix B is invertible. This corresponds to CNOT circuits, which we will later show can always be extracted from any GPF diagram which is an isometry.

Lemma 6.5. *All Z-X (or X-Z) GPF phase-free ZX diagrams which are isometries, must have* $j = 0$.

Proof. This can be concluded through proof by contradiction: Assume that $j \neq 0$. Each of these j red spiders is a post-selection, which annihilates all computational basis state inputs for which that parity is nonzero. However, this contradicts the diagram being an isometry, as isometries cannot annihilate any input states. Therefore, $j = 0$. □

Theorem 6.6. *For any CSS code, there exists a bijection between the complete stabilizer tableau of an encoder map for it, and a pair of phase-free Z-X and X-Z GPF diagrams both equal to that encoder map.*

Note without loss of generality that we are making an (initial) choice of logical operators to be either Z-type or X-type; we know from the encoder from stabilizer tableau extraction algorithm that we will present later, that this is always possible. To generalize to any choice of encoder, we can then allow any unitary to be applied on the logical qubits of the chosen phase-free encoder.

Proof. To go from a pair of Z-X and X-Z GPF diagrams for the same isometry to a complete stabilizer tableau, we can simply read off the connectivities. To prove that this indeed is the correct stabilizer tableau, recall from earlier that the logical operators are determined by the connectivity of the input spiders by π-copy, and likewise the stabilizer generators are also determined by their connectivity to the output spiders. Either diagram of the pair fixes the encoder map, so there cannot be more rows of the complete stabilizer tableau than can be written down from these two diagrams' connectivities.

To go from a complete stabilizer tableau of a CSS code where all rows are either Z-type or X-type, to a pair of Z-X and X-Z GPF diagrams for the same isometry, we directly write down the encoder map. How to do this for Z-X GPF (color change for X-Z GPF) is:

1. Draw $k + m_X$ Z spiders on the left, and n X spiders on the right, of a bipartite graph.
2. Draw an output wire for all n X spiders.

3. Draw an input wire to the first k Z spiders, and connect each of them to the X spiders which are supported by $|\overline{Z_i}\rangle$ for $i \in [1, k]$.

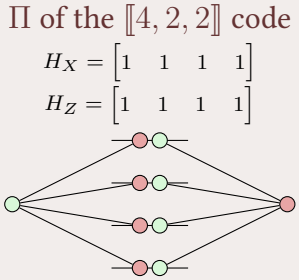
4. Connect each of the remaining m_X Z spiders to the X spiders which are supported by $|S_i^X\rangle$ for $i \in [1, m_X]$.

We know that a complete stabilizer tableau fixes an encoder map which must be an isometry, and that every isometry can be put into both Z-X and X-Z GPF. The logical operators and stabilizer generators are fixed by the connectivities and are indeed correct. We also know we have the correct number of logical operators, which is k Z-type plus k X-type. We also know that no X-type stabilizer generators can be produced from linear combinations of the Z half of the tableau (as all their X columns are zero), so there are no missing X-type stabilizer generators in the resulting Z-X GPF diagram, and vice versa. □

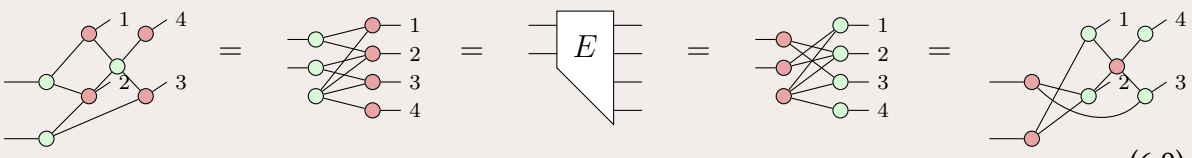
This brings us to a definition of a pair of Z-X and X-Z GPF diagrams, the combination of which enables one to read off a complete stabilizer tableau:

Definition 6.7. We call a pair of Z-X and X-Z GPF diagrams for the same encoder map, a *presentation* of the code, corresponding to a choice of complete stabilizer tableau in accordance with Theorem 6.6.

Example 6.8 (The $[\![4, 2, 2]\!]$ code)**.** The Tanner graph of the $[\![4, 2, 2]\!]$ code, drawn as a ZX-diagram for the projector onto the codespace Π, is



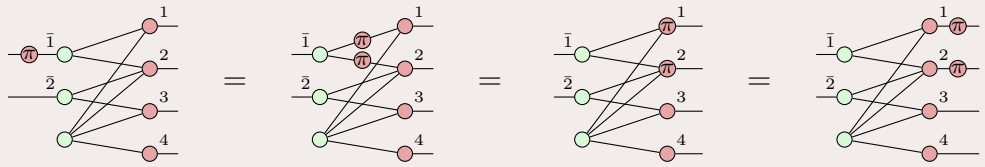
An encoder for this code is



(6.9)

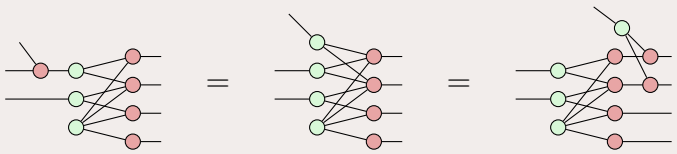
For any phase-free ZX-diagram which is an isometry, we can show that the physical implementation of Pauli X (or Z) on any logical qubit, is X (or Z) on every physical qubit that that logical qubit's Z (or X) spider in Z-X (or X-Z) GPF is connected to. As an example, we

can show that $\bar{X}_1 = XXII$ by applying the π-copy rule, which states that Pauli X on any one wire of a Z spider equals Pauli X on all its other wires simultaneously.



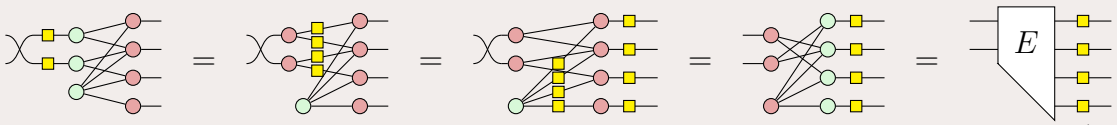
(6.10)

More generally, any ZX-diagram can be *pushed through the encoder* [HLY+23]. This means that given a phase-free ZX-diagram encoder for any CSS code, and any logical map L as a ZX-diagram, we can find always find a ZX-diagram for the physical map P which implements L in that CSS code. Essentially, we can push through each Z spider by the below steps (exemplified for the $[\![4, 2, 2]\!]$ code), and each X spider by applying the same rules but color-reversed. Note that this only finds a valid diagram for P, but optimizing the implementation is a separate matter.



(6.11)

As a self-dual code, the $[\![4, 2, 2]\!]$ code has a logical operation whose physical implementation is a transversal Hadamard gate on all physical qubits simultaneously. For this choice of encoder, this logical operation is SWAP; $H \otimes H$:



(6.12)

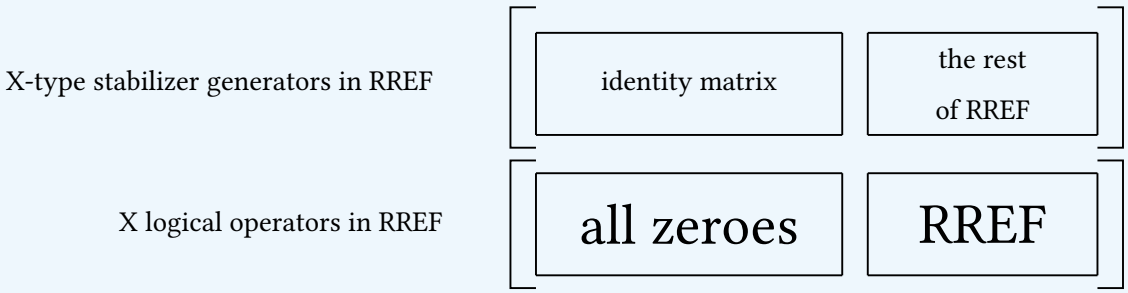
6.3 The RREF$_X$ Normal Form for Complete Stabilizer Tableau Rows

Definition 6.13. The RREF$_X$ form writes all X stabilizer generator rows ($\{S_X\}$) and X logical operator rows ($\{\overline{X}\}$) of the complete stabilizer tableau of a CSS code (presuming some choice of logical operators such that all $\overline{\mathrm{X}}$ are over $\{X, I\}$ and all $\overline{\mathrm{Z}}$ are over $\{Z, I\}$) as a binary matrix through the following steps:

1. Reduce $\{S_X\}$ to be in Reduced Row Echelon Form (RREF).

2. Subtract rows of $\{S_X\}$ from $\{\overline{X}\}$, to zero in $\{\overline{X}\}$ all columns which are pivots of $\{S_X\}$.
3. Reduce $\{\overline{X}\}$ to be in RREF.

The resulting binary matrix is of the form below, where the horizontal lines can be any bitstring.



Likewise, the RREF_Z form can be obtained by applying the above procedure to the Z stabilizer generator rows ($\{S_Z\}$) and Z logical operator rows ($\{\overline{Z}\}$).

Proposition 6.14. *For any CSS code, we can write down an encoder map for it in phase-free Z-X (or X-Z) GPF, such that its biadjacency matrix B is in RREF_X (or RREF_Z) form.*

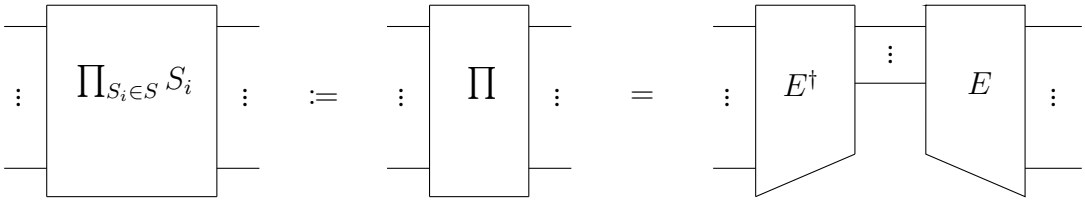
Remark 6.15. This restriction to only X logical operators over $\{X, I\}$ and Z logical operators over $\{Z, I\}$, is applicable to any CSS code. This is because a CSS code fixes the *image* of the encoder map, but not the encoder map itself. In fact, changing the choice of logical operators does not change the code distance.

In choosing to presume X logical operators over $\{X, I\}$ and Z logical operators over $\{Z, I\}$, CSS code encoder maps correspond precisely to isometries in the phase-free ZX calculus. Moreover, writing such an encoder map as a state by Choi-Jamiolkowski isomorphism preserves it being a CSS code — to be exact, a $k = 0$ CSS code. This perspective on CSS codes lends itself naturally to techniques developed in the study of the phase-free fragment of the ZX calculus.

We can apply this reasoning to phase-free encoders, to all encoders for CSS codes, by virtue of the fact that any choice of logical operator is always unitarily equal to a phase-free choice of logical operators of the form above.

This is enforced by the property that any quantum error correcting code fixes the image of any encoder map E, such that $E^\dagger$ then E must equal the projector collectively defined by each

stabilizer generator.

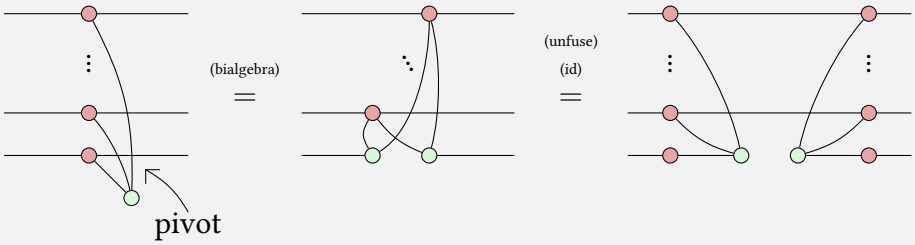


(6.16)

Therefore,

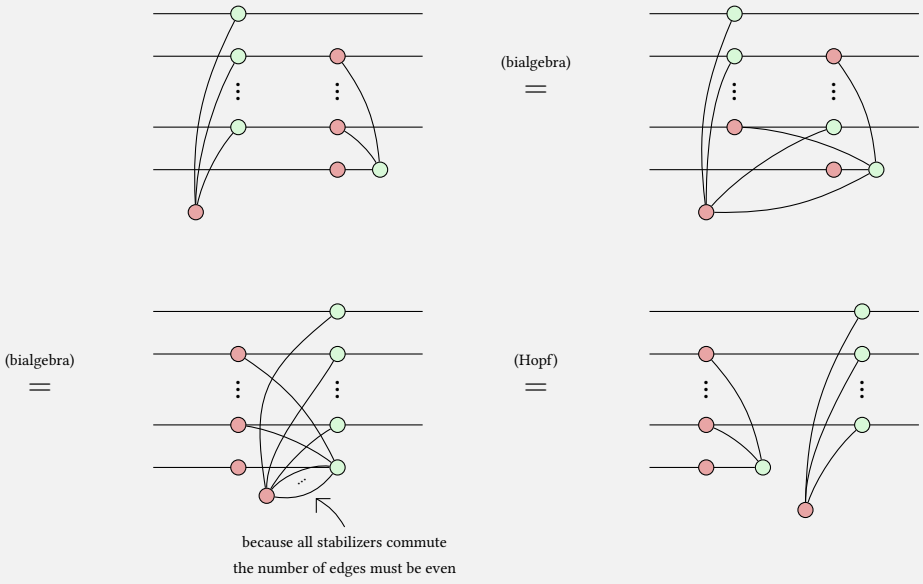
Proposition 6.17. *Given an incomplete stabilizer tableau for any CSS code, i.e. a complete set of stabilizer generators without specifying the logical operators, we can extract a phase-free encoder map in Z-X GPF. This implies that there always exists a choice of logical operators for any CSS code, such that all X logical operators are over $\{X, I\}$ and all Z logical operators are over $\{Z, I\}$.*

Proof. 1. For each X-type stabilizer generator, apply the bialgebra rule about the spider corresponding to its pivot column in the RREF_X stabilizer tableau.



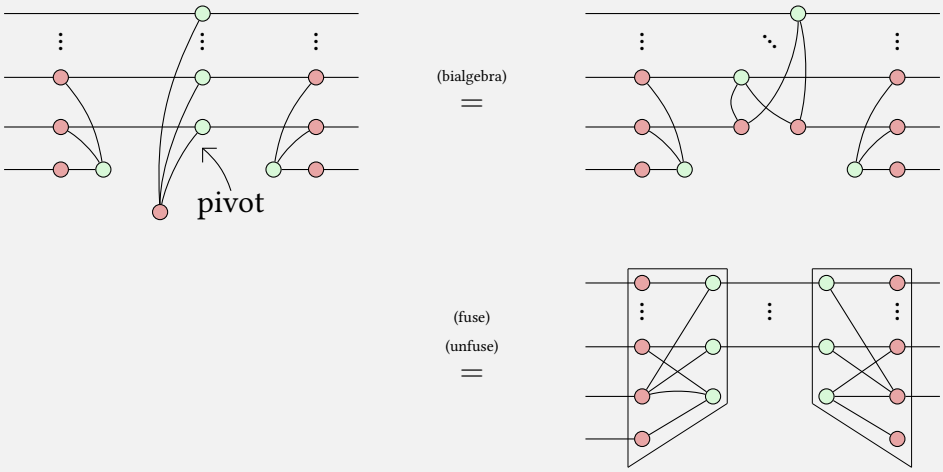
(6.18)

2. Commute all Z-type stabilizer generators to the middle, past all the X-type stabilizer generators.



(6.19)

3. For each Z-type stabilizer generators, apply the bialgebra rule about its pivot in RREF_Z. At the end, fuse all X and Z spiders, then unfuse all middle Z spiders to form $E^\dagger$ on the left and E on the right.



(6.20)

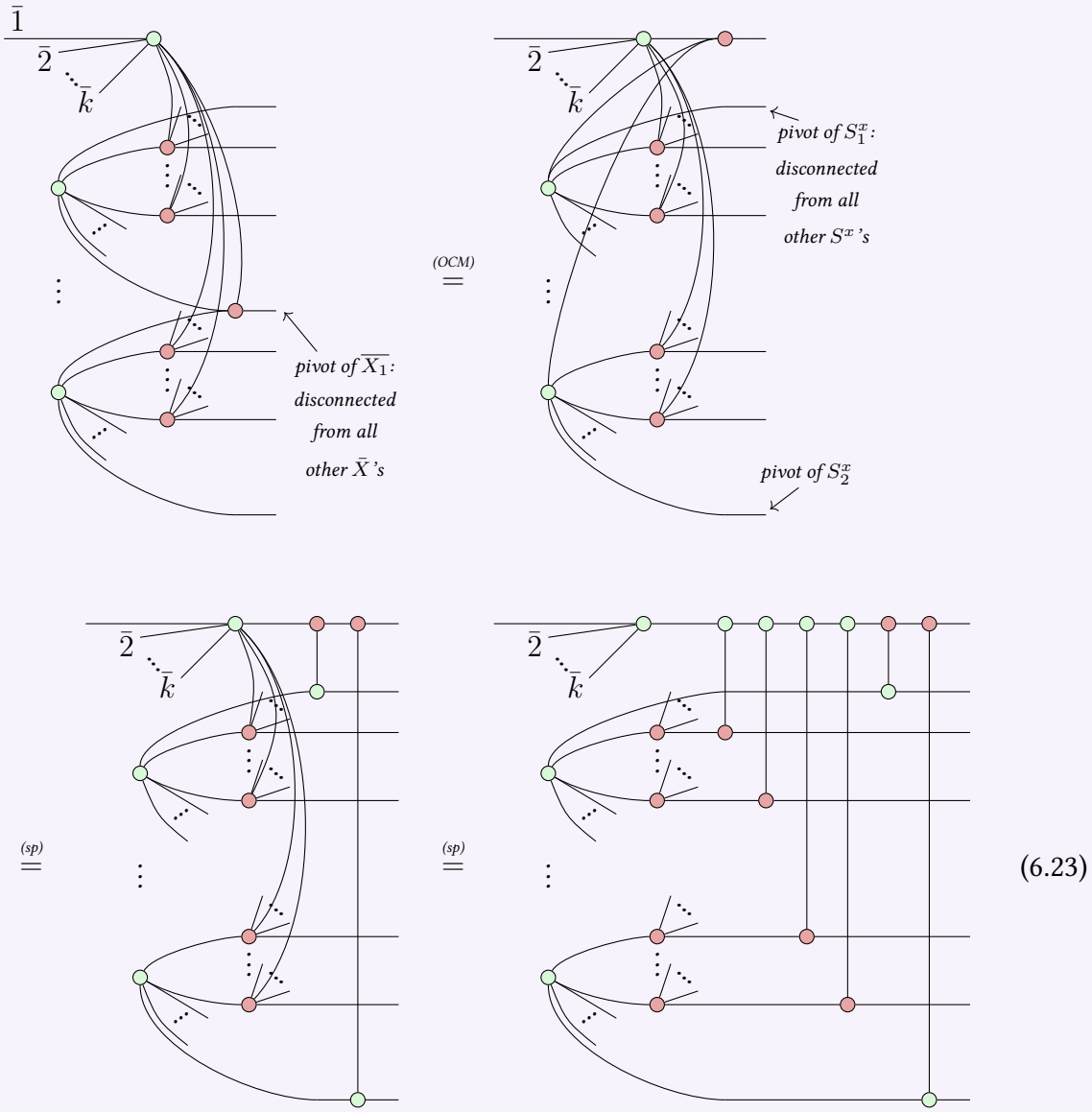
The total number of applications of the bialgebra rule, each of which reduces the number of wires in the middle by one, is equal to the number of stabilizer generators, which is $n - k$. Therefore, the total number of wires in the middle at the end is $n - (n - k) = k$, which is indeed the number of logical qubits. □

Corollary 6.21. *In the diagrams above, it can be seen that the stabilizer generators fix the image of E, and moreover that there are $k = n - s$ wires in the middle, for k logical qubits. To preserve the equality of Equation 6.16, no modifications can be done to the input n wires nor the output n wires of $EE^\dagger$, because doing so would make it no longer equal to $\prod$. The only modification which preserves this equality and the number of logical qubits k, is prepending the phase-free E by a unitary U which defines a new encoder, whilst appending the phase-free $E^\dagger$ by $U^\dagger$ which equals the dagger of the new encoder.*

6.4 Circuit Extraction

CSS code encoder maps must be *unitary embeddings*. This means that we can always write them as a phase-free ZX unitary applied to a tensor product of identity wires, $|0\rangle$ states, and $|+\rangle$ states. To derive this, we present an explicit *circuit extraction* algorithm below. Note that this is not of direct usefulness, because in practice encoder maps are not directly implemented; rather, in practice one would implement stabilizer measurements using fault-tolerant syndrome extraction circuits. Having said this, there are practical reasons to implement parts of encoder maps as circuits, such as for magic state distillation or code switching, as well as theoretical reasons for studying quantum error correcting codes.

Proposition 6.22. *For any CSS code encoder in phase-free Z-X (or X-Z) GPF, we can always extract a CNOT circuit from it. First, the Z-X GPF must be in RREF$_X$ (or X-Z GPF in RREF$_Z$) form; if not, it can be efficiently rewritten into this form by Gaussian elimination as described in the previous subsection. The key step of the circuit extraction is given below.*



(6.23)

Here, the pivot of each X logical operator is moved as to be on the same horizontal qubit wire as the Z spider for that logical qubit, by permuting the order the outputs are drawn. Following that, CX gates can be unfused for all other X spiders connecting to that Z spider.

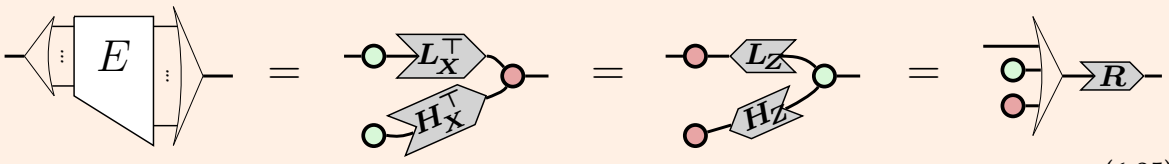
The same can be done to unfuse CX gates from each X-type stabilizer generator, such that the control of each new CX gate is on the pivot wire of that stabilizer generator.

The full algorithm iterates the above for each logical operator and for each stabilizer generator. The end result is a CNOT circuit acting upon identity wires, and ancillae initialized to $|0\rangle$ and

$|+\rangle$.

Regarding the complexity of this algorithm, reducing the stabilizer tableau rows to RREF_X can be done efficiently, such as by Gaussian elimination. The extraction itself is also efficient because is consists of k initial Only-Connectivity-Matters rewrites, followed by $b-k-g$ CX gate unfusions. Here k is the number of logical qubits of the code, b is the number of edges in the Z-X GPF biadjacency graph, and g is the number of X-type stabilizer generators.

Corollary 6.24. *As a result, we can leverage the Scalable ZX calculus to concisely represent phase-free isometries. Any CNOT circuit can be represented as a map in $\mathbb{F}_2$ by a reversible binary matrix. Therefore, any CSS code encoder can be drawn in the* scalable *ZX calculus in two different ways—as its Z-X or X-Z GPF biadjacency connectivity [KvdW24b], or as a CNOT circuit unitary embedding:*



(6.25)

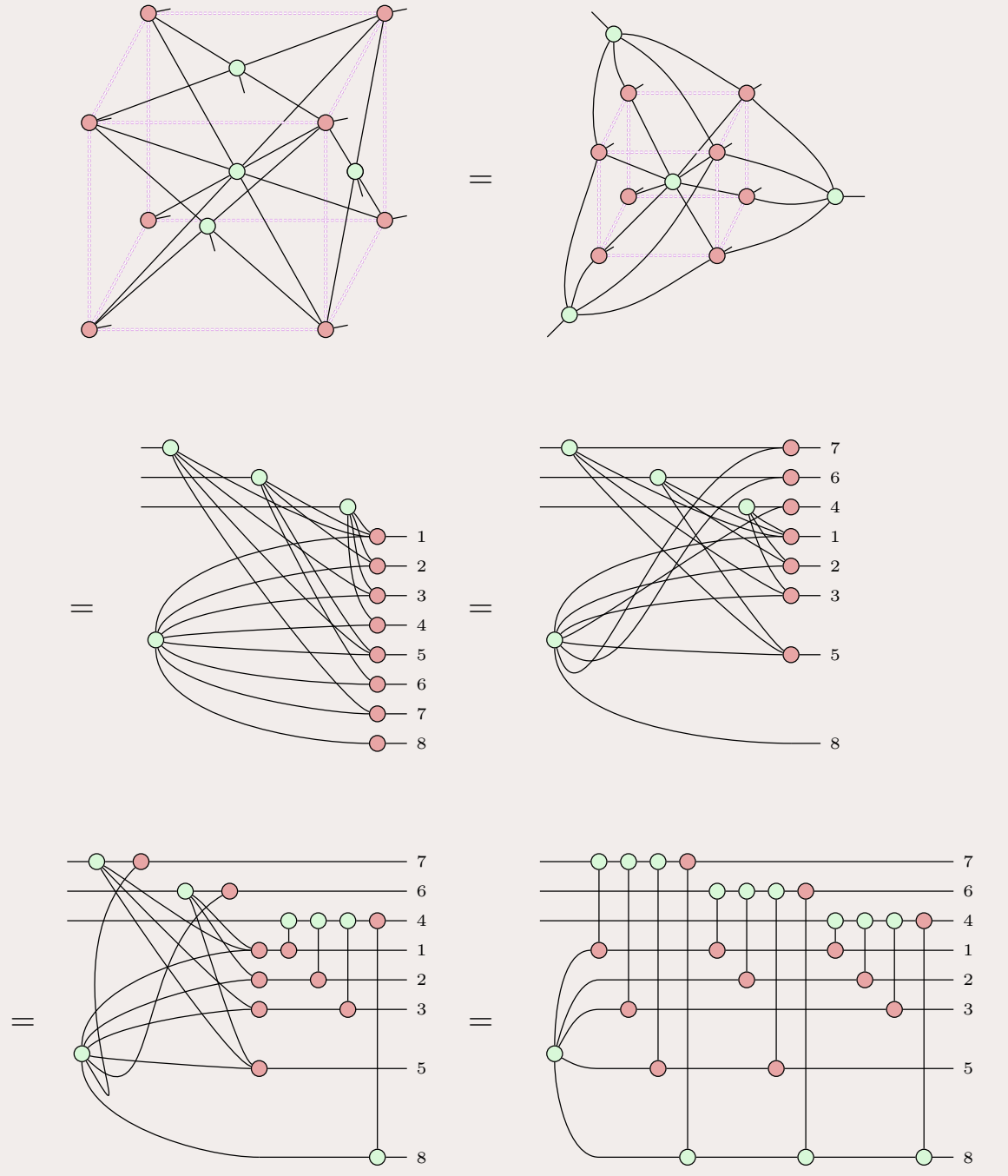
This uses the fact that an arbitrary CNOT circuit can be drawn in the Scalable ZX calculus, as a matrix arrow representing the phase-free ZX parity map, here labelled by the reversible binary function R.

We will shortly find certain scalable ZX calculus rewrite rules essential to succinctly deriving transversality of CX gates for all CSS codes.

First, for an example with more than one logical qubit:

Example 6.26. Let us apply the above procedure to the $[\![8,3,2]\!]$ code encoder. This is commonly drawn as a cube; to illustrate this cube shape, we have superimposed a cube in pink on top of the ZX diagram of its encoder map. Then the X logical operators are on three independent faces of the cube, while the weight 8 X-type stabilizer generator is in the volume

of the cube.



(6.27)

Note that following the last step here, the 5-qubit GHZ state can be further decomposed into 5 CNOT gates acting on one $|+\rangle$ state and four $|0\rangle$ states, as we do much later in Equation (12.41).

Remark 6.28. Circuit extraction to a unitary embedding of the $[\![8, 3, 2]\!]$ entirely by ZX calculus rewrites was previously done in Ref. [GD18, Proposition 5.3], and moreover believed to be the first time a circuit for this code was given. They achieve this using 56 rewrite rules

in the ZX calculus proof assistant Quantomatic: The first half of this is automated, and the latter half required substantial human intervention.

To explain how our procedure improves upon this, the primary contribution is that this is an algorithm applicable to all CSS codes, and as we will later explain, to all stabilizer codes with negligible modification. Additionally, this is both efficiently automated, and easy to do by hand so long as the code itself is small enough to work with.

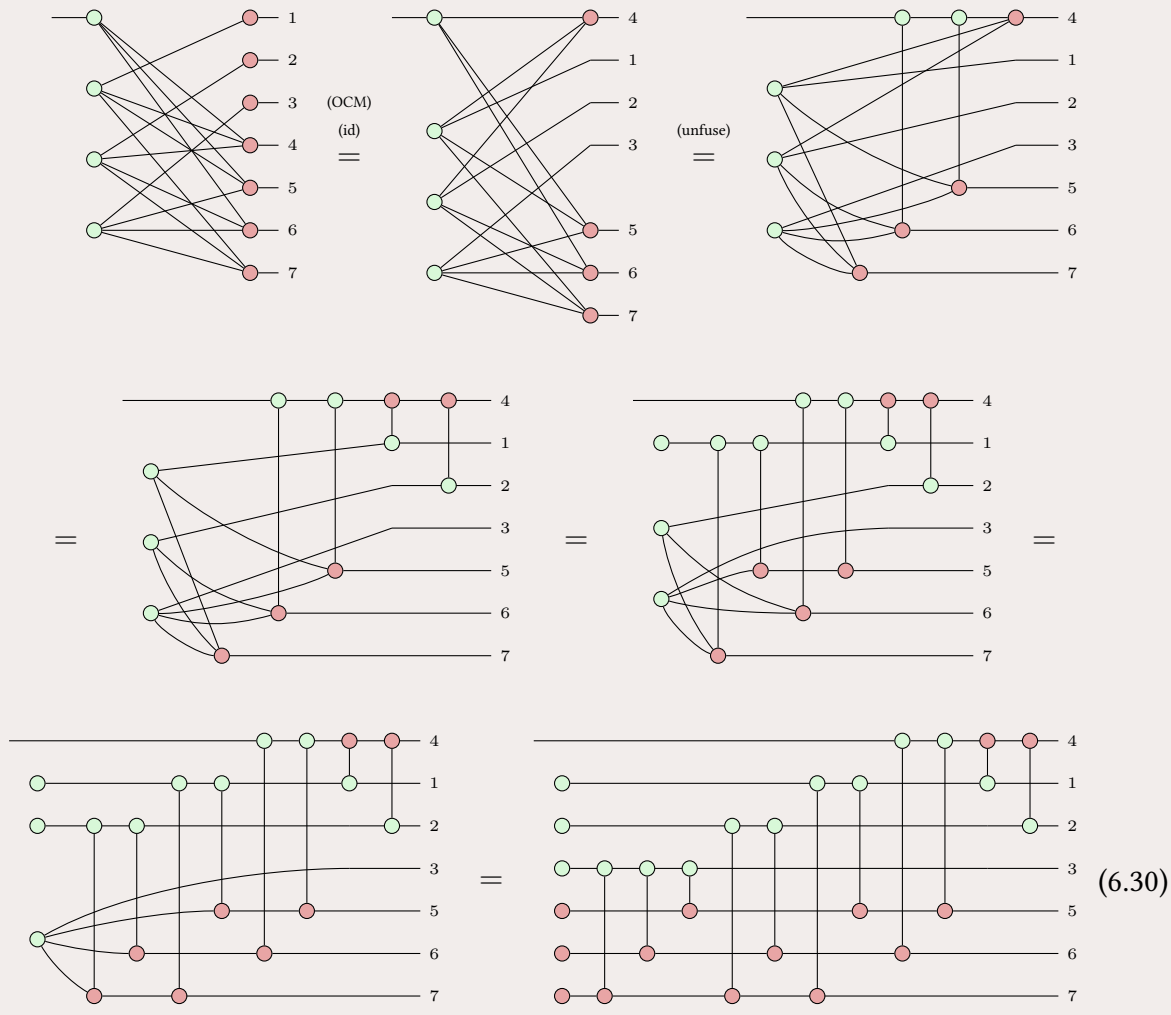
To do an example with more than one X-type stabilizer generator:

Example 6.29. Let us apply the above ciruit extraction procedure to the $[\![7, 1, 3]\!]$ Steane code encoder. This circuit extraction of the $[\![7, 1, 3]\!]$ from what we call Z-X GPF, has been previously done through a combination of automated rewrites in the Quantomatic software and human guided proof [DL14]. Its RREF_X matrix is

$$\left[\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]$$

where the the first row is a choice of indices of the physical qubits, the last row is for the X logical operator, and the middle 3 rows are the X-type stabilizer generators. Starting with

the Z-X GPF of the above matrix, the circuit extracts to:



(6.30)

6.5 Transversal Gates and Magic State Distillation

H gate

We previously mentioned that

Proposition 6.31. *Any self-dual CSS code (i.e. $H_X = H_Z$) has a logical gate whose physical implementation is a transversal Hadamard gate on all physical qubits simultaneously.*

Proof. Recall a fact we used in Lemma 4.24, that $\Pi; P = \Pi; P; \Pi$ means that the physical operation P preserves the code space defined by the projector Π. We can see that $P = H^{\otimes n}$ indeed preserves the code space, because

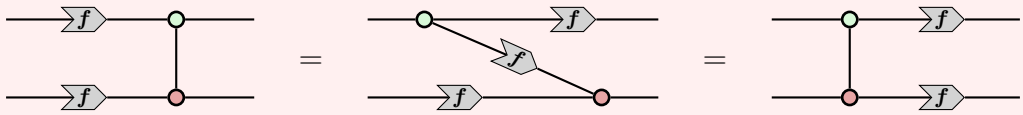
$$\Pi; H^{\otimes n}; \Pi = \Pi; \Pi; H^{\otimes n} = \Pi; H^{\otimes n} \tag{6.32}$$

The first equality follows from Observation 4.51, the color change rule, and the fact that $H_X = H_Z$. The second equality follows from $\Pi^2 = \Pi$ for any projector Π. □

CX gate

We would like to next prove transversality of any CX gate on any CSS code. In the scalable ZX-calculus, we can commute a pair of the same bijective matrix arrow i.e. equal to the same CNOT circuit, through the controls and the targets respectively of a ladder of transversal CX gates.

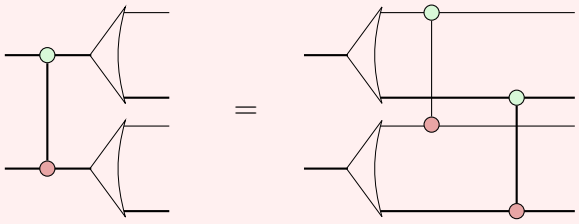
Lemma 6.33. *For any matrix arrow $f \in \mathbb{F}_2^{m \times m}$ that is both injective and surjective, i.e. is bijective,*



(6.34)

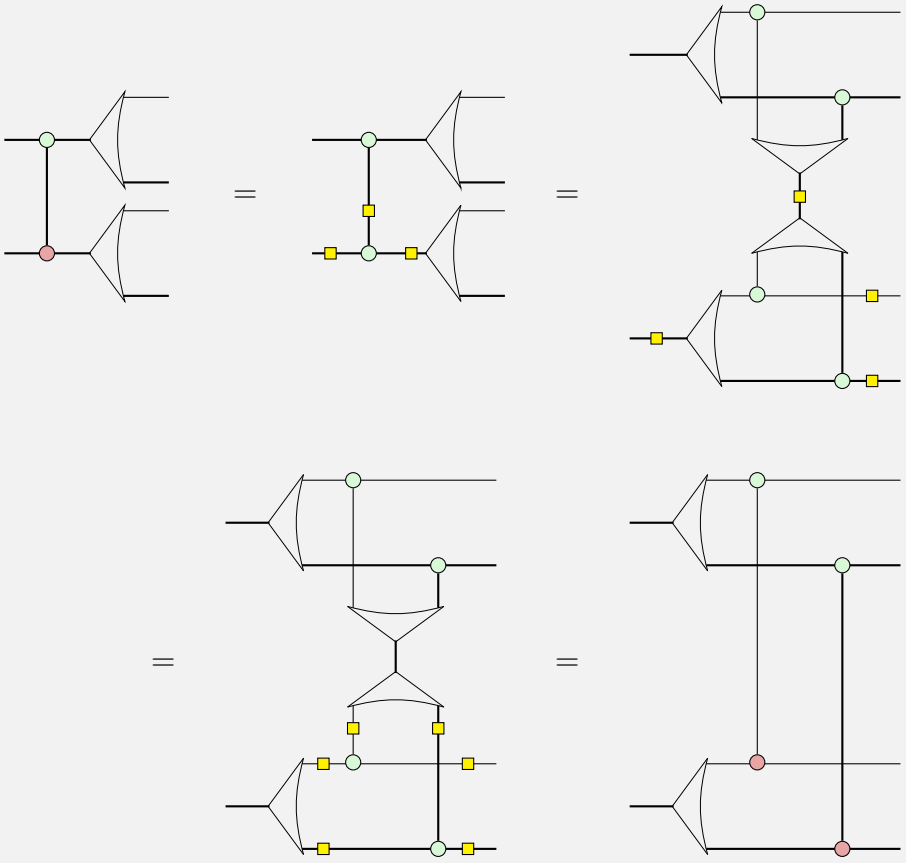
We also need to commute CX gates with dividers and gatherers, to verify that they act as one would expect.

Lemma 6.35.



(6.36)

Proof.



(6.37)

This can be applied iteratively, to show that this indeed implements a CX gate from the ith qubit on the top half to the ith qubit on the bottom half. □

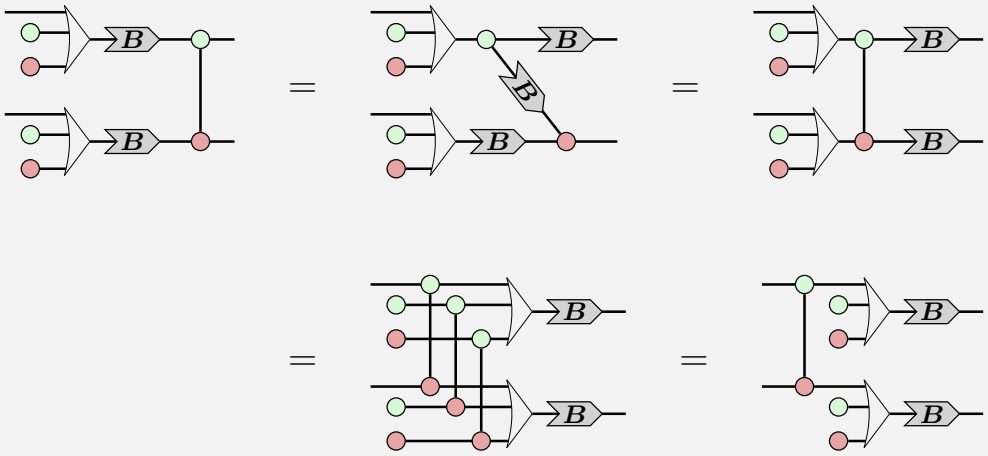
Lemma 6.38. *For any CSS code, we can write transversal CX gates between two code blocks physically, is equal to transversal CX gates between two code blocks logically, i.e.*

$$\overline{\Pi_{i=1}^{k} CX_i} = \Pi_{j=1}^{n} CX_j \tag{6.39}$$

where CX_i has control on the ith logical qubit of the top block and target on the ith logical qubit of the bottom block, and CX_j has control on the jth physical qubit of the top block and target on the jth physical qubit of the bottom block.

Proof. We apply Lemma 6.33 to an arbitrary CSS code encoder in the Scalable ZX calculus

form of Equation 6.25.



(6.40)

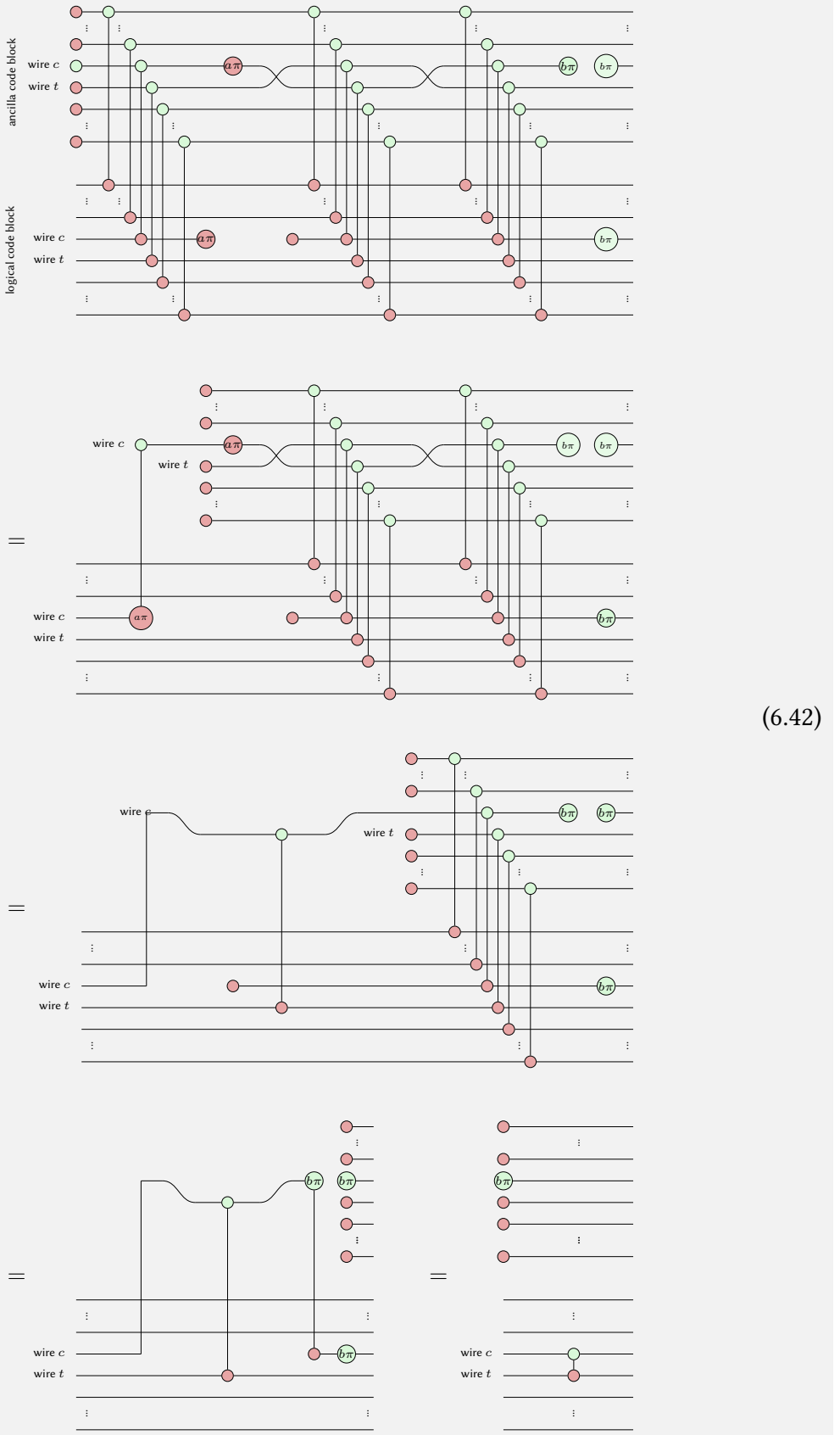
□

Although the above proof suffices to do any logical CX gate transversally when $k = 1$, we furthermore present a graphical proof to extend this to $k > 1$.

Proposition 6.41. *Any logical CX gate on any CSS code can be implemented transversally.*

Proof. When $k = 1$, Lemma 6.38 suffices. For when $k > 1$, we present below a proof entirely in the ZX calculus, of Gottesman's gate teleportation protocol to implement a transversal logical CX gate between two logical qubits in the same code block [Got97]. This protocol uses an ancilla code block prepared in the logical state where all logical qubits are initialized to $|0\rangle$, except that the logical qubit having the desired CX gate's control index c is initialized to $|+\rangle$. Note that below, classical feed-forward of measurement outcomes labelled by $a, b \in \{0, 1\}$ is

necessary for correction after each teleportation.



(6.42)

The proof proceeds near identically as above for the case where the transversal logical CX gate is between two different code blocks — the only difference, is that the middle ladder of CX gates instead goes from the ancilla code block to the desired CX gate's target qubit's code block. The protocol still works because it essentially teleports the control qubit from its control code block to the ancilla code block, swaps it to the same index within the code block as that of the target qubit within its own code block, does the CX, then undoes the swap and the teleportation. □

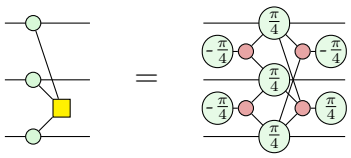
Diagonal gates

CSS codes with transversal diagonal gates in the third level of the Clifford hierarchy are called *triorthogonal* codes [BH12]. For such codes, H_X is a *triorthogonal* binary matrix, meaning that the Hamming weight of the product of any two rows, as well as the product of any three rows, is $0 \bmod 2$. All such gates admitted by CSS codes were characterized in the scalable ZX-calculus in Ref. [KvdW24b], which found that they correspond to *spider nest identities* [dBBW20a].

Here, we will demonstrate the well-known example of the transversal CCZ gate of the $[\![8, 3, 2]\!]$ code [Cam16] (also called the smallest interesting color code, an instance of transversal CCZ gates described in Ref. KubicaA2015colorcode), in the ZX-calculus. In addition to serving as an example to gain some intuition before we next introduce magic state distillation, the reason for being interested in this code specifically is that its connection to circuit synthesis will be the inspiration for the proposal at the end of the next chapter, for which we will revisit this example.

First, we will use the following fact about the phase polynomial representation of the CCZ gate as the map that acts as $|x, y, z\rangle \mapsto (-1)^{x \cdot y \cdot z} |x, y, z\rangle$, where x, y, z are bits denoting computational basis states. As we later explain and compute for qutrits in Equation (10.31), the qubit CCZ gate is identified by the exponent of (-1) being a multiplicative expression $x \cdot y \cdot z$; this can be replaced by an equivalent expression summing over modular addition. For this reason,

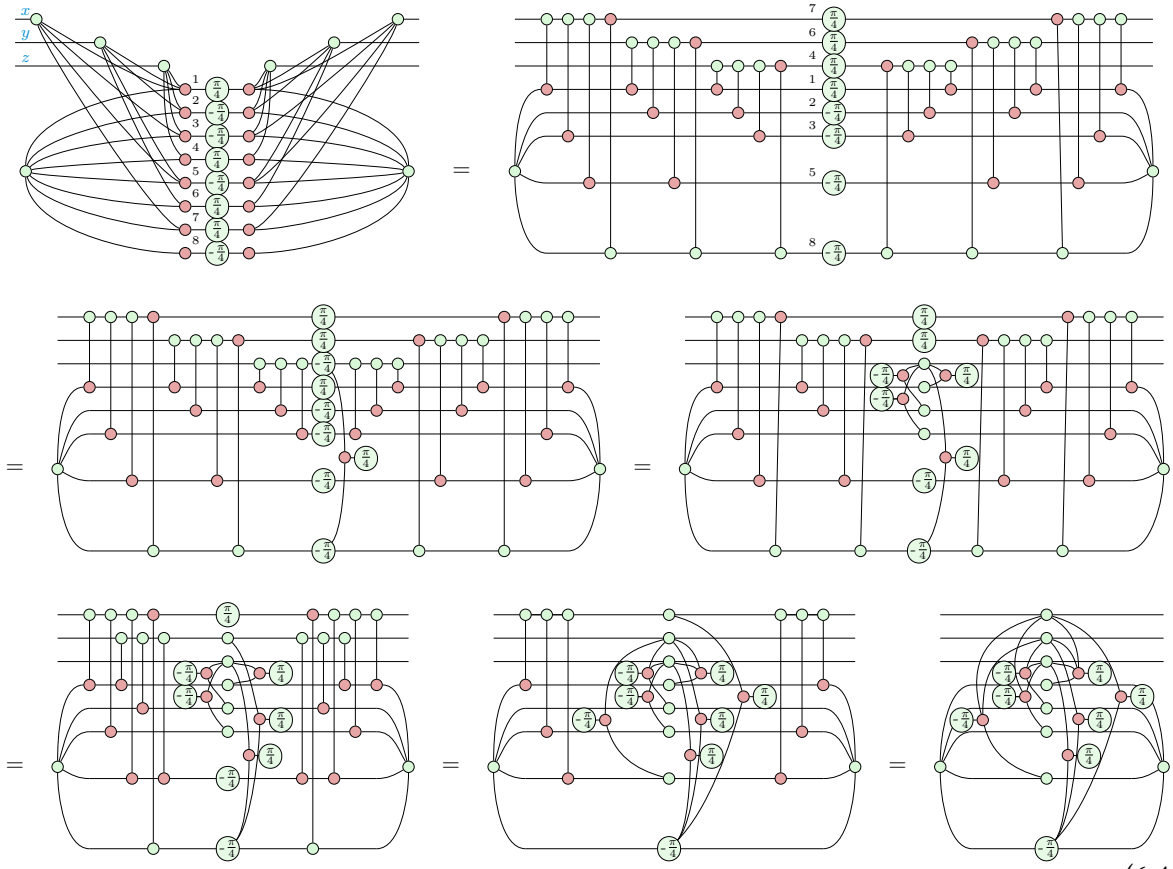
Observation 6.43. The qubit CCZ gate is equal to the following spider nest of phase gadgets:



(6.44)

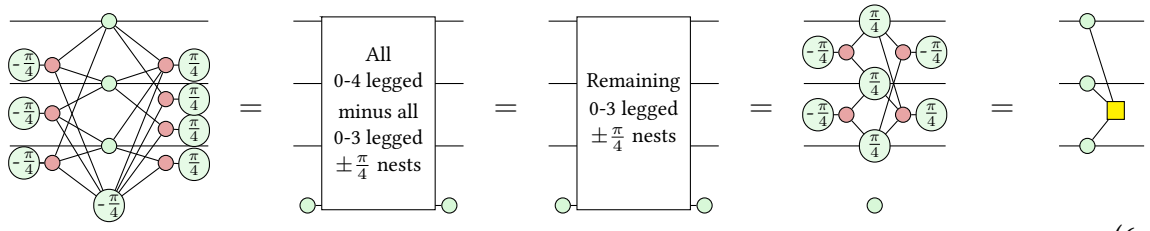
If instead all phases here are $\pm\frac{\pi}{2}$, then instead of the CCZ gate, this is the identity map on three qubits. More generally, for n qubits, having one of all possible connectivity phase gadgets of 0-n legs, of phases $\pm\frac{\pi}{2^{n-2}}$ where the sign matches the parity of the number of legs, is equal to the n-qubit identity map.

We can substitute the encoder E of the $[\![8,3,2]\!]$ code from Equation (6.27), to verify that $L = E;P;E^\dagger$ where L is the CCZ gate, and $P = TT^\dagger T^\dagger TT^\dagger TTT^\dagger$.



(6.45)

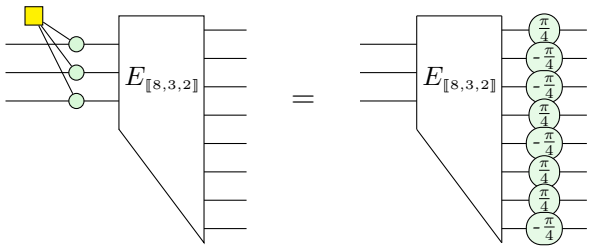
Continuing from the last step, we now utilize the above observation to recognize that what we have is the spider nest of all 0-4 legged $\mp\frac{\pi}{4}$ phase gadgets, conjugated by all 0-3 legged $\pm\frac{\pi}{4}$ phase gadgets on the upper three qubits. Therefore, we can then apply all 0-4 legged $\mp\frac{\pi}{4}$ phase gadgets, which equals the identity. The end result is precisely the CCZ gate.



(6.46)

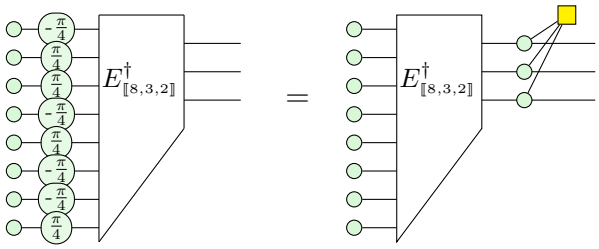
Magic state distillation

Here, we give a graphical portrayal of magic state distillation. We have just derived that the logical CCZ gate in the $[\![8,3,2]\!]$ code is physically implemented by 8 T and $T^\dagger$ gates:



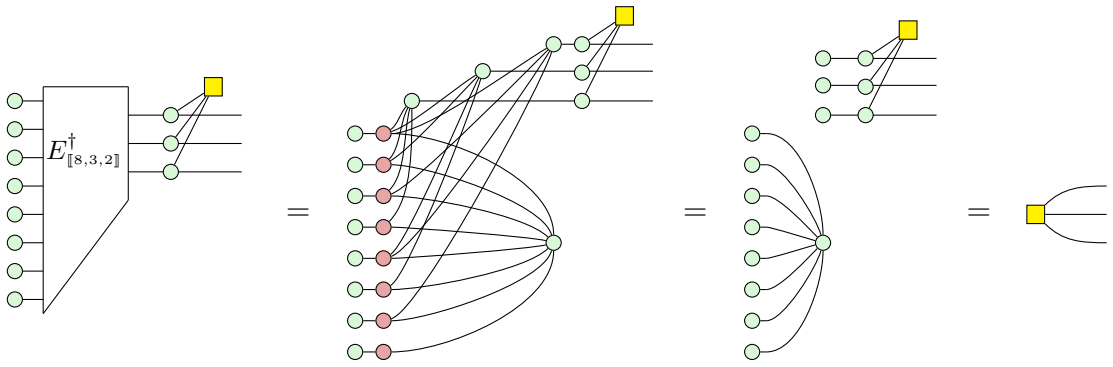
(6.47)

Taking the conjugate transpose of the above equation and inputting $|+\rangle^{\otimes 8}$, the ideal decoder acts on 8 T and $T^\dagger$ magic states as:



(6.48)

Substituting in the Z-X GPF of the encoder, the $|+\rangle$ states copy and fuse, leaving just the three-legged H-box.



(6.49)

In other words, the decoding of 8 T and $T^\dagger$ magic states results in a less noisy CCZ magic state.

6.6 Sample: Code switching and morphing

As a sample, here we showcase a connection between code switching and code morphing pointed out in our paper [HLY+23]. This serves primarily as a motivating application of the formalism thus far; refer to the paper and Sarah Meng Li's Master's thesis [Li23] for more specifics. In the paper, we focused on the example of code switching between the $[\![7,1,3]\!]$ code

and the $[\![15, 1, 3]\!]$ code, and code morphing to obtain a new code from the $[\![15, 1, 3]\!]$ code and the $[\![8, 3, 2]\!]$ code. This observation can be generalized to the class of all CSS codes, but this is ongoing work out of the scope of this thesis.

First, we give an illustrative high-level overview of this code switching protocol.

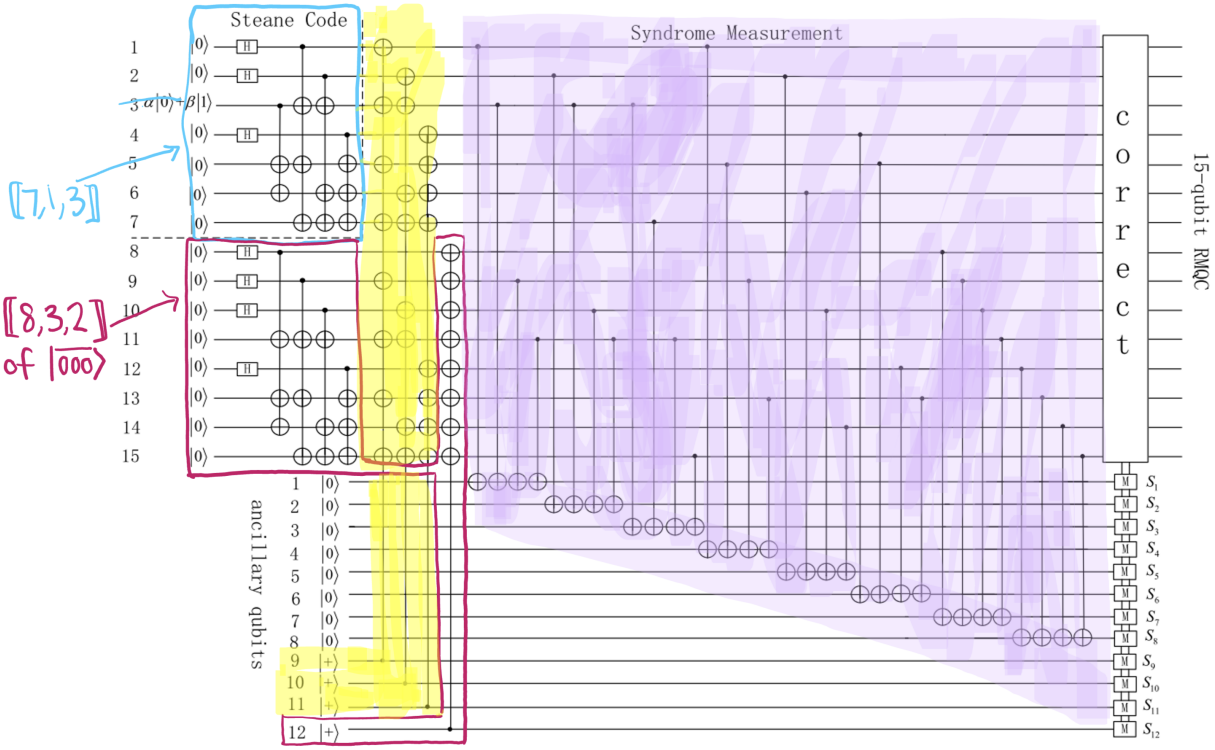


Figure 6.1: This circuit diagram reprinted from Ref. [QZPS18] is a non-fault-tolerant protocol for code switching from the $[\![7, 1, 3]\!]$ code to the $[\![15, 1, 3]\!]$ code. We have added highlighting and annotations to indicate the presence of smaller quantum error correcting codes that play a role in this code switching protocol. Note that the four measurements to ancilla qubits 9 to 12 mutually commute, and so the three measurements highlighted in yellow can also be scheduled after the preparation of $|0\bar{0}0\rangle$ in the $[\![8, 3, 2]\!]$ code.

A geometric description of this observed in Ref. [WCAB15], is that this code switching protocol adds 3 X-type stabilizer generators, which stitch thirds of the $[\![7, 1, 3]\!]$ triangle onto 3 faces of the $[\![8, 3, 2]\!]$ cube. This description can be made more mathematically concrete by drawing the encoders for both of these codes, then adding three weight 8 Z-type stabilizer generators by measuring them as in the yellow highlighted subcircuit of Figure 6.1. These three new stabilizers fuse, as in Z spider fusion, each Z-type stabilizer generator of the $[\![7, 1, 3]\!]$ with a Z logical operator of the $[\![8, 3, 2]\!]$ code.

Yet we can also connect this to *code morphing*, a protocol which resulted in the discovery of the smallest known code that can distill T magic states, in the original paper by Vasmer and Kubica [VK22].

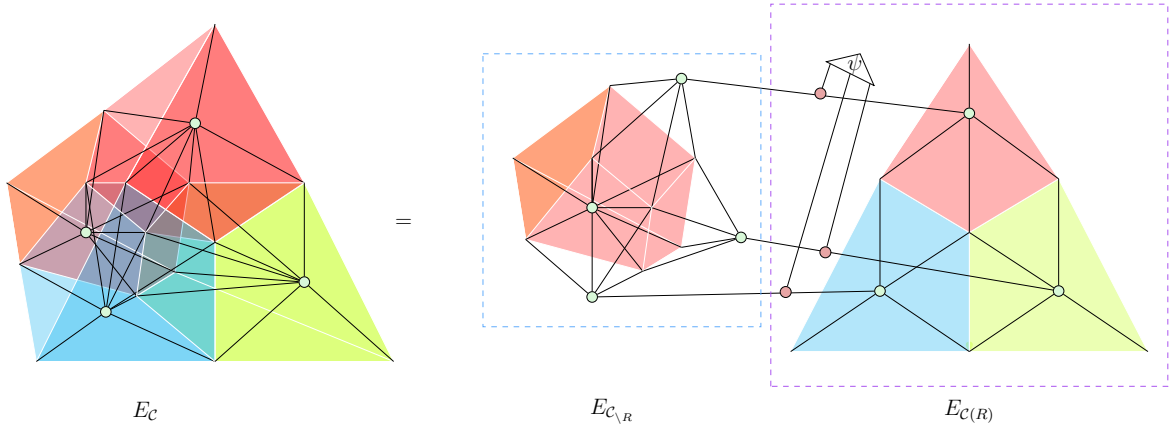


Figure 6.2: Figure with a minor modification from our paper [HLY+23, Example 4.2]. This depicts the code morphing protocol from Ref. [VK22] of the $[\![15, 1, 3]\!]$ code as the parent code C and the $[\![8, 3, 2]\!]$ code as the child code $C(R)$, to obtain the morphed code $C \setminus R = [\![10, 1, 2]\!]$. For brevity, the X spiders representing physical qubits and the logical qubit wires inputting to the Z spiders are omitted.

The modification to the figure from our paper is to replace the three identity X spiders between the two codes, with three 3-legged X spiders connected to the state $|\psi\rangle$. This draws a connection between this code morphing protocol [VK22], and code switching as an instance of gauge fixing a subsystem code [VLC+19]. This state $|\psi\rangle$ is of the three *gauge logical qubits* of the $[\![15, 1, 3, 3]\!]$ CSS subsystem code, the larger code encompassing both the $[\![15, 1, 3]\!]$ code (when $|\psi\rangle = |000\rangle$) and the $[\![7, 1, 3]\!]$ code (when $|\psi\rangle = |+++\rangle$). Substituting these two states into the above diagram, the former indeed fuses so the X spiders are identities, resulting in the $[\![15, 1, 3]\!]$ code encoder; the latter indeed copies across the X spiders and fuses into the $[\![10, 1, 2]\!]$ code encoder, resulting in the $[\![7, 1, 3]\!]$ code encoder.

7

Isometries in the stabilizer fragment

tableau-certified

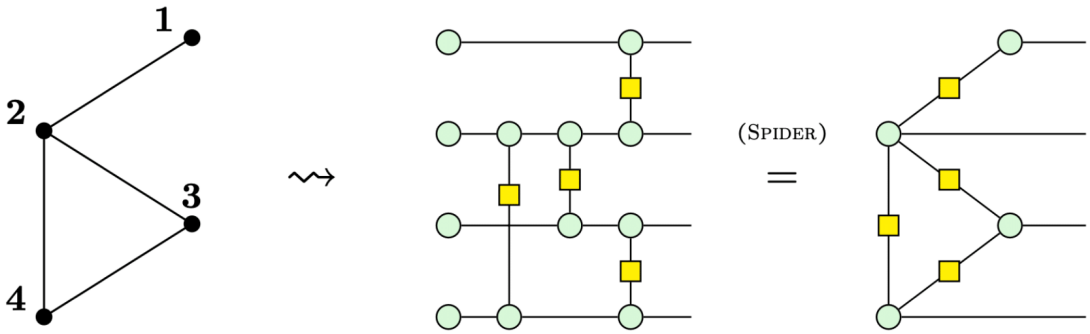
In this section, we present an on-the-nose translation between any $k = 0$ stabilizer code and its ZX normal form. These techniques will then be used to establish the one-to-one correspondence between any stabilizer code and a new normal form in the ZX-calculus.

We start by explaining for any CSS code, how to interconvert between stabilizer tableau and phase-free ZX-calculus representations through the Generalized Parity Form (GPF). Ensuing this, we remark that while the more general case of any graph code is suited to Graph State with Local Cliffords (GSLC) normal forms, this is identifiable with a complete stabilizer tableau, but not with the usual (i.e. incomplete) stabilizer tableau.

To bridge this gap, we adapt the reduced Affine with Phases (AP) normal form to enable reading off both complete and incomplete stabilizer tableaus for the most general setting of any stabilizer code. In turn, given any stabilizer tableau, we give an entirely graphical procedure to produce a valid encoder diagram. We can further conceptualize our adapted AP normal forms as a generalization of both the GPF and GSLC forms. This thereby sets up a unified framework to extend graphical results for CSS and graph codes to all stabilizer codes, explored in the next section.

7.1 Graph State with Local Cliffords Normal Form for Graph Codes

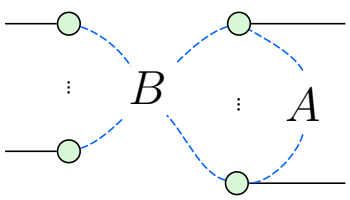
As a prelude to the case of all stabilizer codes, we briefly discuss another well-studied class of stabilizer codes: graph codes [Sch01]. We can look at graph codes as codes that admit an encoder whose Choi state is a *graph state*. Any graph state is identified with a simple graph $G(V, E)$. Graph states can be constructed in the circuit model by initializing all $\|V\|$ qubits to $|+\rangle$, then applying a CZ gate for each edge in the graph.



Graph states and graph codes are identified by a simple graph with two types of vertices, for input and output qubits respectively. Any stabilizer code is equivalent to a graph code, up to local (i.e. single-qubit) Clifford gates.

In the ZX-calculus, $k = 0$ graph codes (corresponding to stabilizer states) can be written as a Graph State with Local Cliffords (GSLC), as introduced by Backens [Bac14]. They gave an efficient algorithm to convert any stabilizer state to GSLC form together with a way to compare graph states representing identical states, resulting in the original proof of completeness of the qubit ZX-calculus for stabilizer quantum mechanics. A more practical algorithm for reducing an arbitrary stabilizer ZX-diagram, including those with inputs, to GSLC form was given in Ref. [BMdF+21].

Up to local Cliffords on the output qubits[1], graph codes admit encoders of the form

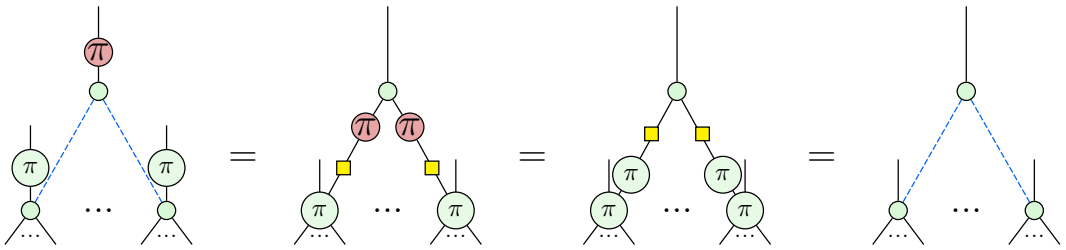


(7.1)

where B is a biadjacency matrix and A is an adjacency matrix.

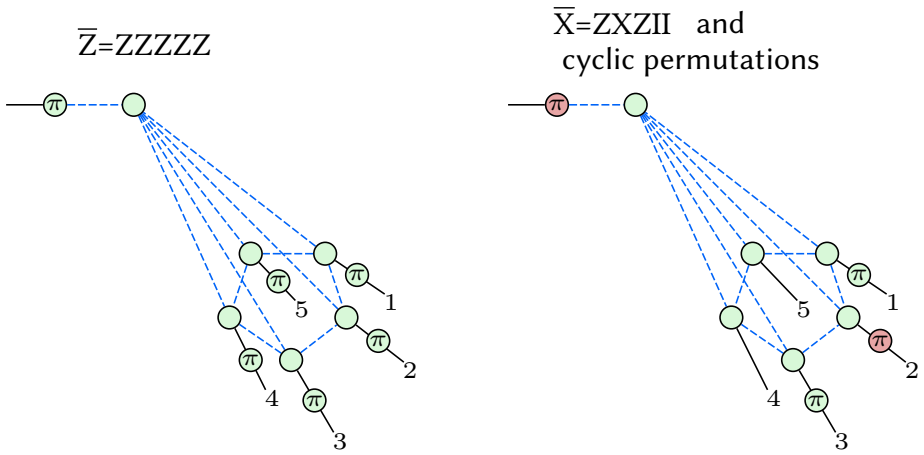
[1]Local Cliffords on the input qubits can always be disregarded for either of two reasons. One, any local Cliffords on the inputs can be relegated to the outputs, by operations on the graph such as local complementations. Two, stabilizer codes are defined by stabilizer groups; all encoder maps henceforth are equivalent up to unitaries on the logical qubits.

Ref.[GMV25] gave a stabilizer tableau normal form for any Clifford isometry in the circuit model, via the quantum circuit equivalent of GSLC form. This leverages a desirable property of GSLC form: It is possible to directly read off enough linearly dependent stabilizers from the graph state, to fix the diagram. This is due to the well-known property of graph states that one stabilizer can be written down for any vertex of the graph: X on that vertex, and Z on all its neighbours. Next to this, commuting through the local Cliffords is straightforward. In the ZX-calculus, the proof of this property on a subgraph consisting of a vertex and all its neighbors is as follows:



(7.2)

However, except for the case of $k = 0$ codes, GSLC form is not the right representation for reading off a stabilizer tableau. It does not provide information on the stabilizer group — which is precisely what defines the code. A toy example through which to illustrate this is the 5-qubit code [LMPZ96]. From examining its encoder in GSLC form, we can read off six linearly independent Pauli strings: one that is a logical Z operator, and five that are logical X operators.



(7.3)

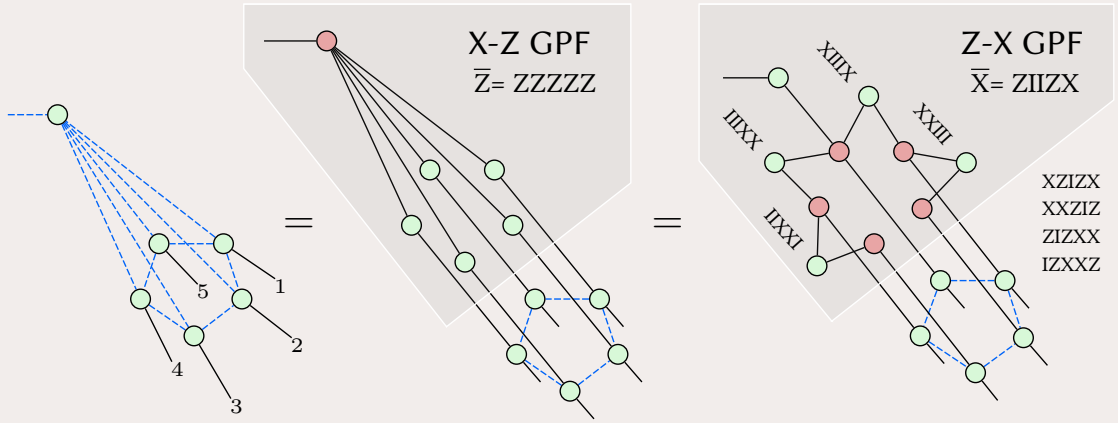
In other words, it is not the case that four ($= n - k = 5 - 1$) of these are stabilizer generators, and the other two fix logical X and Z operators respectively. To derive a stabilizer tableau from these Pauli strings, more steps are necessary in order to isolate stabilizer generators from any dependence on the logical qubits.

This corresponds to rewrites which bring us outside the simple graph formalism, but are still well captured in the ZX-calculus. The following example of the 5-qubit code is instructive for

demonstrating the steps involved, before we proceed to the general case of all stabilizer codes in the next section.

Example 7.4. At the start of Equation 7.5, we have the GSLC form of an encoder for the 5-qubit code, a graph code. By unfusion, this splits into an X-Z GPF diagram identified with the biadjacency matrix B, followed by a CZ circuit identified with the adjacency matrix A. The connectivity of the X-Z GPF diagram directly allows reading out the logical Z operator $\overline{Z} = ZZZZZ$. Note that Z-type logicals and stabilizers (i.e. Pauli strings over $\{Z, I\}$) always commute with CZ gates.

Next, we make use of the phase-free GPF algorithm from the previous section to rewrite that X-Z GPF diagram to Z-X GPF. Reading off from it a logical X operator and four linearly independent X-type stabilizers, we can then commute these Paulis through the CZ circuit to obtain the rest of the complete stabilizer tableau.



(7.5)

In brief, although we cannot read off stabilizer generators directly from any graph code in GSLC form, we showed an example where we rewrote it to a form where we can. In the next section, we show this for any stabilizer code, how to ensure there are enough linearly independent rows for the complete stabilizer tableau.

7.2 Affine with Phases Normal Form for stabilizer codes

Up to now, we can read off stabilizer tableaus for any CSS code and any graph code. As any stabilizer code is equivalent to a graph code up to local Cliffords, for many intents and purposes, the earlier algorithms will suffice. For instance, in searching for stabilizer codes with a good distance, it is only necessary to search up to the equivalence class defined by each graph code, as local Cliffords do not alter the code distance.

With this said, there are circumstances where tracking the precise local Clifford is important. A notable example is in determining the threshold based on an error model, local Cliffords

transform the type of error relative to what the code corrects. For quantum error correcting codes to be effective, they must account for the dominant sources of physical noise, decoding and correcting based on the likelihood of certain errors occurring. Better yet, codes can be designed for biased noise models, in order to detect certain types of errors more effectively than other, less probable types of errors.

In this section, we introduce a normal form for the encoder of any stabilizer code in the ZX-calculus. It is on-the-dot, meaning that there is no longer the limitation of being up to local Cliffords. Moreover, it enables reading off rows of the stabilizer tableau directly from the diagram. It is derived from the Affine with Phases (AP) normal form, which we will first review. This will allow us to directly read off rows of the stabilizer tableau from it. After that, we use a *reduced* AP normal form to study the case of isometries, in order to capture the complete stabilizer tableau for any stabilizer code on-the-dot.

The AP normal form is based on the stabilizer state normal form first introduced in Ref. [DD03], which represents a state as an equal weight superposition over an affine subspace of computational basis states. Its utility and representation in the ZX-calculus was independently discovered by Backens and McElvanney [MB23] and Kissinger and van de Wetering [KvdW24a]. In Ref. [PBC+23] we adapted it to arbitrary odd prime qudit dimension to give a completeness proof of stabilizer ZX-calculus with it as a unique normal form. A correspondence between AP normal form and stabilizer tableaus in graphical calculi for symplectic algebra was subsequently established by Booth, Carette, and Comfort [BCC24]; however, while this applies to any odd prime qudit dimension, unfortunately it does not apply to qubits as for two dimensions it is isomorphic to Spekken's toy model [CK22].

The algorithm for rewriting any qubit stabilizer state (or the Choi state of any diagram in the stabilizer fragment) to AP normal form is detailed in Ref. [KvdW24a].

Figure 7.1 visualizes AP normal form diagrams consistent with the notation in this text.

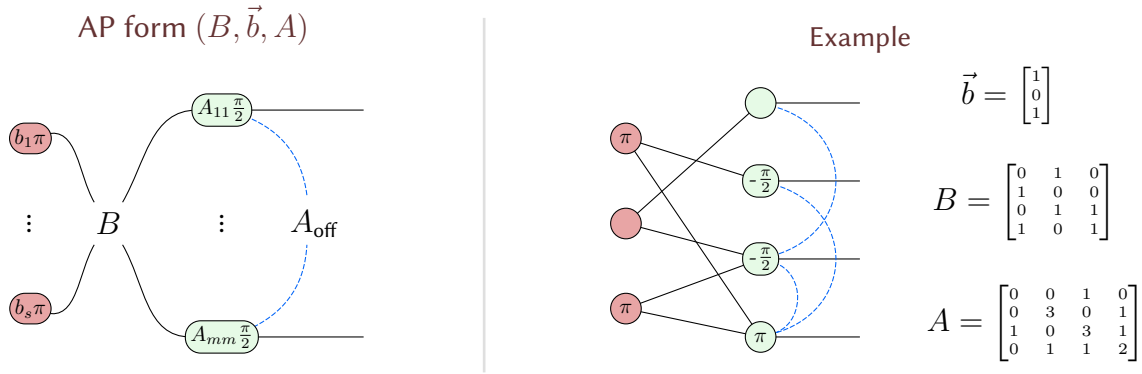


Figure 7.1: Left: General diagram for a stabilizer state in AP normal form. Right: An example stabilizer state in AP normal form, alongside its parameters $(B, \vec{b}, A)$.

From any stabilizer state in AP normal form, we can further refine the diagram to *reduced* AP (rAP) normal form. For reduced AP normal form, the three binary matrices that describe it $(B, \vec{b}, A)$ have the property of being *unique* for any stabilizer state, on-the-dot. This is accomplished by putting $(B, \vec{b})$ into Reduced Row Echelon Form (RREF), and thereafter reparametrizing A to be dependent only on the free variables, i.e. qubits that are not the pivots of $(B, \vec{b})$. The algorithm for rewriting an AP normal form diagram to reduced AP normal form is given in the Picturing Quantum Software textbook preprint [KvdW24a]. These two algorithms run in polynomial time of $O(n^3)$, performing the graphical equivalent of Gaussian elimination. Therefore, applying them to any two stabilizer diagrams achieves efficient determination of whether they are equal. If equal, this thus finds a series of rewrites that transform one diagram into the other. The AP normal form lets us establish a connection between ZX-calculus diagrams and tableaus, for any stabilizer states and hence any $k = 0$ stabilizer code.

7.3 The stabilizer tableau equivalent of AP normal form

In this section, we show how to read off any stabilizer tableau of any stabilizer code. In addition to corresponding with the AP normal form for stabilizer states of the previous section, this will also correspond to the next section, where we define and present a reduction algorithm to a normal form for any Clifford isometry.

First, consider a stabilizer tableau of the following form:

$$T = \left(\begin{array}{c|c} M_X & N_X \\ 0 & M_Z \end{array}\right)$$

Note that we adhere to the convention of the joint +1 eigenspace and thereby do not need an additional column for the sign of each stabilizer generators.

Up to re-ordering rows, this is just an arbitrary stabilizer tableau, to which we have chosen to make the rows of M_X linearly independent. We can always do this. Suppose row r of M_X is a linear combination of some other rows. Then we can simply add those rows to row r and replace row r with a pure-Z stabilizer, which we incorporate into M_Z. The number of rows of M_Z is the maximum number of pure-Z stabilizer generators supported in the codespace, which can be zero.

Since the rows of M_X are linearly independent, it has a right inverse R, i.e. a matrix such that $M_X R = I$. This implies that we can write $N_Z = M_X G$ for some matrix G (namely, $G := RN_Z$).

This gives us an alternative way to write the tableau.

Proposition 7.6. *The stabilizer tableau for any code can be written into the normal form*

$$T = \left(\begin{array}{c|c} M_X & M_X G \\ 0 & M_Z \end{array}\right) \tag{7.7}$$

where G is a symmetric matrix, and all rows are linearly independent.

We can in fact always choose G to be symmetric, i.e. $G = G^T$. One way to see this, is that this form of tableau corresponds to AP normal form, which any stabilizer state can be rewritten to. M_Z and M_X specific the connectivity of the affine part of the ZX diagram (in X-Z form and Z-X form, respectively). Accordingly, G defines the diagonal Clifford part consisting of S and CZ gates. If $G_{ij} = G_{ji} = 1$, introduce an S gate if $i = j$ and a CZ gate otherwise. Therefore, we can always find a G which is symmetric, namely the symplectic representation of the diagonal Clifford part of the AP normal form ZX diagram.

Conversely, given any CSS code (which is the special case of $G = 0$), we can always "twist" it with an adjacency matrix G to get a new code:

Lemma 7.8. *If the following matrix:*

$$S = \left(\begin{array}{c|c} M_X & 0 \\ 0 & M_Z \end{array}\right)$$

is symplectic, then so too is

$$T = \left(\begin{array}{c|c} M_X & M_X G \\ 0 & M_Z \end{array}\right)$$

for any symmetric G.

Proof. We already know the M_Z part "commutes" with everything, so we only need to check the lower half of the matrix is symplectic. This gives us:

$$(M_X G)M_X^T + M_X(M_X G)^T = M_X G M_X^T + M_X G^T M_X^T = M_X G M_X^T + M_X G M_X^T = 0$$

□

This gives us a recipe for directly writing down stabilizer tableaus of any $k = 0$ stabilizer code, through procedures entirely within the ZX-calculus. From AP normal form which is determined by $(B, \vec{b}, A)$, we can write down $M_Z = B$ directly. As the codespace of a stabilizer code is the joint $+1$ eigenspace of all stabilizer generators, it must be the case that $\vec{b} = 0$ and A is binary. This is because the presence of any X or Z gate changes at least one $+1$ eigenvalue of the constituent CSS code to -1. This lets us write down $G = A$. Finally, we can convert the

X-Z GPF subdiagram to Z-X GPF, enabling us to read off M_X as its biadjacency matrix.

This also gives us a recipe in the other direction, for directly writing down an AP normal form ZX-diagram given M_Z and G. Note that M_X is not necessary in this direction, but only because the bottom left quadrant (the quadrants can be different sizes) of the tableau is known to be zero. This is because it takes less information to fix a state, than to reconstruct a basis for all its stabilizer generators.

One might hope that there is a general way to write down a ZX-diagram in normal form from an arbitrary stabilizer tableau without having to do this 'preprocessing' of the tableau. However, this does not seem likely considering the analogy to CSS codes. If we are given a CSS code tableau where the generators are neatly separate into the Z and X parts, then we can directly read off the phase-free ZX-diagram that implements it. However, if we are given a 'scrambled' version of the tableau where the Z and X part are mixed, then it is no longer obvious that it is in fact a CSS code. The only way we can recover the knowledge that it was a CSS code (and hence should correspond to a phase-free diagram), is by unscrambling it, which corresponds to solving a set of linear equations similar to what we described above.

7.4 Adapting AP normal form to isometries

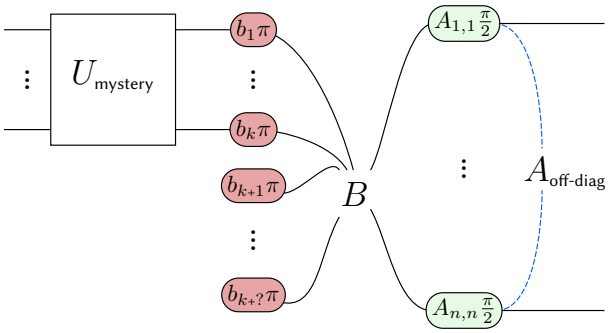
These algorithms were developed for states, but in the context of quantum error correction, we are interested in encoder maps and measurement circuits. By the Choi-Jamiolkowski isomorphism, for any $k > 1$ stabilizer code, we can convert any encoder map (which is an isometry) to a state, and apply the above algorithms. At this point, to recover an isometry, what we can do is "un-Choi", i.e. bend the output qubits that were originally inputs back into inputs.

We extend rAPNF for any stabilizer state, to a normal form for any isometry in the stabilizer fragment of the ZX-calculus, i.e. any Clifford isometry.

Definition 7.9. We will use the notation {G_{11}, G_{12}, ...}; {G_{21}, G_{22}, ...}; ... to denote layers in circuit decompositions. Here {G_{11}, G_{12}, ...} is the leftmost layer, {G_{21}, G_{22}, ...} is the second layer, and so on. Each layer consists of one or more gates, which mutually commute. The identity on any qubit(s) is always implicitly allowed in each layer. Each circuit decomposition means that for the class of circuits in question, there always exists a way to decompose into a circuit consisting of only gates which are allowed in the first layer, followed by a circuit of only gates allowed in the second layer, and so on.

Observation 7.10. Before we begin, we know that the decomposition we are looking for has

to be of the form

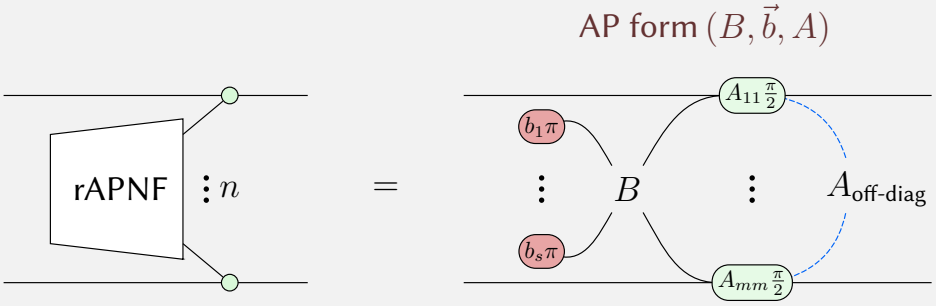


(7.11)

where we don't know yet what goes in the mystery box, other than that it's a Clifford unitary. We also don't know how many internal X spiders there are, denoted by a question mark.

Proof. We know this must be the case as a consequence of three facts.

First, we know that we can always choose the rightmost layer of this decomposition to be a projector to an APNF or rAPNF diagram of the form:



(7.12)

This is possible because the image of any stabilizer code encoder map, must be the code space defined by the stabilizer group of the code. This is described by a stabilizer tableau T, which can always be written into the form in Proposition 7.6, which is in one-to-one correspondence with an APNF diagram. In the case that T has no purely Z-type stabilizer generators, this projector is the identity.

Second, we know from Ref. [WQB23] that for any stabilizer code, its codespace is decomposable into three layers: the codespace for some CSS code; {X}; diagonal Clifford unitary. It follows from this that for any stabilizer code, there exists a choice of encoder that decomposes as: X-Z GPF; {X}; {S, CZ}. This is because for any CSS code, there exists a choice of encoder that is in X-Z GPF. Making this choice fixes the logical operators of the encoder of the stabilizer code.

Third, we know that for any code, any two choices of encoders are equivalent up to a unitary on the logical qubits. Since we are comparing an arbitrary Clifford isometry with a specific choice of Clifford isometry for the same code, the unitary that relates them must be Clifford.

We conclude the proof by propagating all X gates to the left to merge with all the other X

spiders, after which we can fuse the Z spiders in the X-Z GPF with those in the diagonal Clifford unitary. □

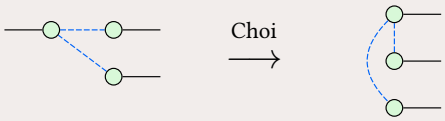
While this confirms the existence of such a normal form, we prove in the remainder of this section how to get to such a normal form. This also deduces what's in the mystery box.

This procedure starts by taking the Choi state of the isometry, then rewriting to rAPNF. We first discuss a few counterexamples to narrow down the cases that we need to consider.

It would be convenient if we had $k + m_z$ total X spiders in the rAPNF, where k is the number of logical qubits and m_z is the number of Z-type stabilizer generators, as we do in the CSS case. Then in the rAPNF algorithm, we could designate one pivot qubit for each X spider: For k of the X spiders we choose input qubits; for the other m_z we choose output qubits. This corresponds to Gaussian elimination of the biadjacency matrix. We would terminate the algorithm by 'un-Choi-ing' the map, i.e. transposing the k qubits to be inputs again.

However, we might not even have k X spiders in the rAPNF, much less $k + m_z$.

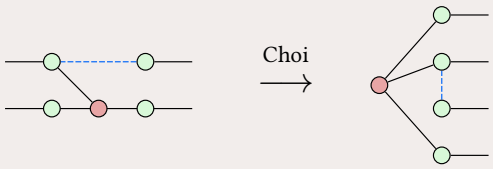
Example 7.13. Here is a toy counterexample where $k = 1$, but the number of X spiders in rAPNF is zero.



(7.14)

A natural question is how much we can modify the original rAPNF algorithm for states. While we can save some time by choosing pivots of X spiders to be inputs (instead of outputs) when possible, we cannot guarantee that every input that is connected to one X spider can be made the pivot of that X spider due to the following.

Example 7.15. If an input is connected to a X spider, it is not necessarily the only input connected to that X spider. An example showing this is a CX gate followed by an H gate:



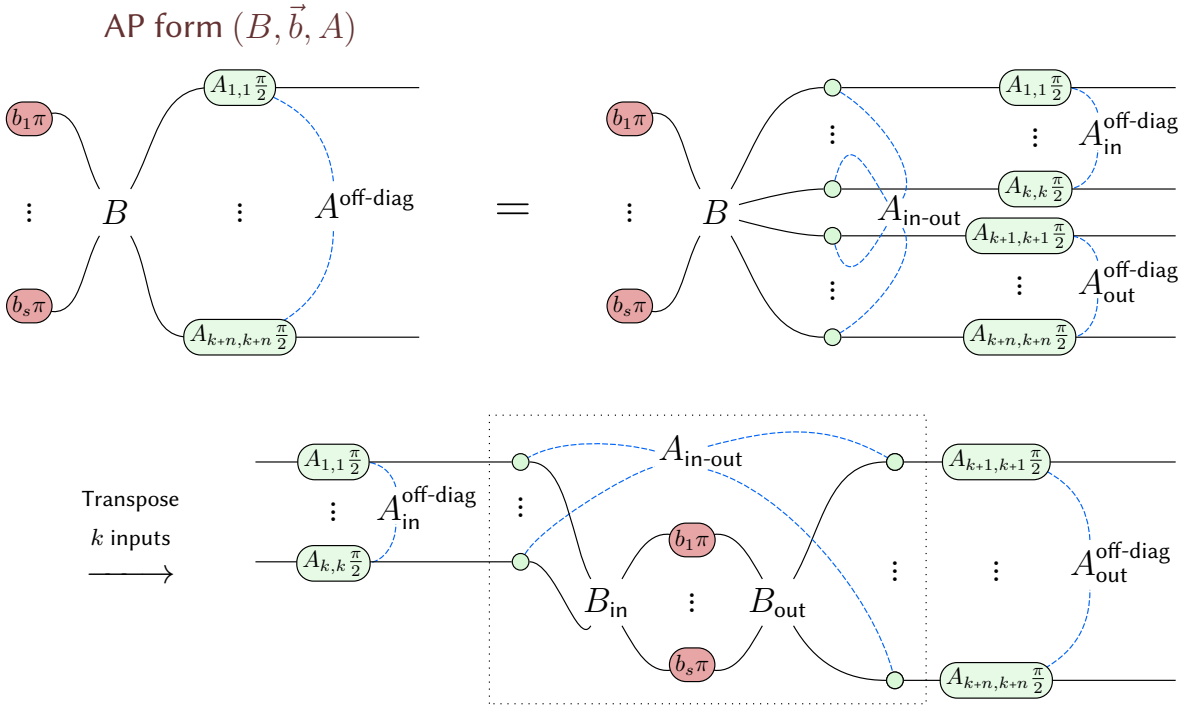
(7.16)

These examples tell us about what more we need to do to get to the normal form. After returning the rAPNF state back into the original isometry shape, we next unfuse two diagonal Clifford unitaries: all components of A that act only on inputs or only on outputs respectively.

Procedure 2: Choi-Jamiolkowski isomorphism on the encoder state in rAPNF

Input: A stabilizer state in rAPNF and a subset of its qubits to become inputs

Output: An intermediate form between rAPNF and iAPNF



From here, we work with the isometry subdiagram in the dotted box above. In addition to $(B, \vec{b})$, it has the remainder of A, consisting of CZ edges between an input and an output.

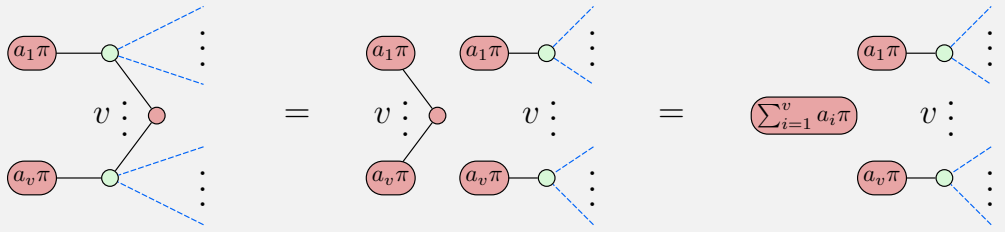
Compare our objective, a diagram satisfying the more general form of Observation 7.10, to the result of Procedure 2 which we have thus far. To achieve the desired transformation, we will iterate through all possible case distinctions of inputs, to make them pivots of X spiders in the biadjacency matrix, hence without CZ edges to the outputs.

We just need all inputs to be addressed; the order affects the final diagram, but is arbitrary. Since we are starting from rAPNF, a unique normal form, any deterministic algorithm will preserve uniqueness of the final isometry normal form. To fix an order here, we address each input from 1 to k, drawn from top to bottom, according to which of the below cases it satisfies.

Before going through the cases, we introduce a definition and preliminary lemmas that we will require.

Lemma 7.17. *Due to all encoders being isometries, it is not possible to have any X spider of the APNF be connected to only input qubits.*

Proof. If this were the case, half of all possible input Z-basis states would be annihilated, hence contradicting the isometry condition of the encoder.



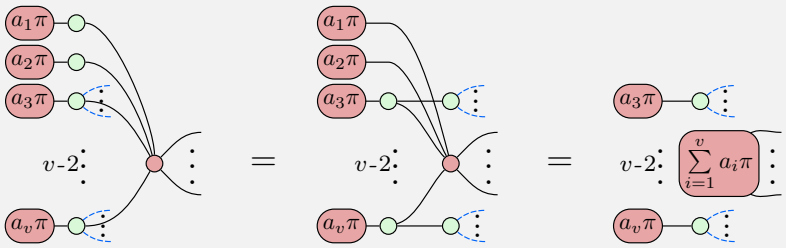
(7.18)

□

We introduce the next lemma to guarantee that CZ edges must exist in some of the subsequent cases.

Lemma 7.19. *Whenever v inputs to an isometry in rAPNF are each connected to the same X spider and no other X spiders, at most one of the v inputs has zero CZ edges to the outputs.*

Proof. We do this proof by contradiction. Assume that for v inputs connected to the same X spider and no other spiders, w inputs have zero CZ edges to the outputs for some $w \geq 2$. Consider the below subdiagram of these v inputs, prior to which the encoder is unitary thus far and therefore does not result in any redundant information of the input qubits. Inputting Z-basis states to this subdiagram denoted by v independent bits, we can see that the subdiagram takes this to $v - w + 1$ linearly independent binary variables.



(7.20)

As $w \geq 2$, $v - w + 1 < v$. As this makes it impossible to recover the input bits, this contradicts the fact that the encoder is an isometry. □

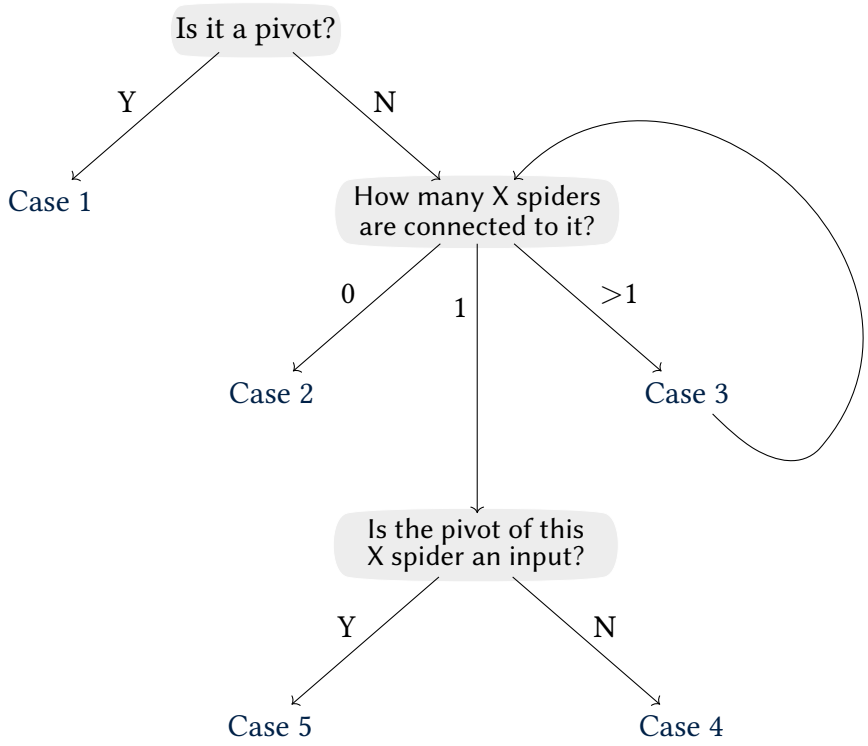
The procedure to derive a ZX diagram for the encoder in iAPNF proceeds with going through the below cases for each input. The control flow is specified in Algorithm 3.

Algorithm 3: Reducing any Clifford isometry to isometry AP normal form

Input: A Clifford isometry in rAPNF that has then undergone Procedure 2

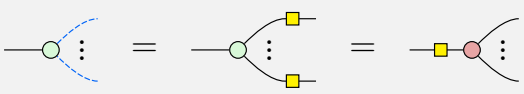
Output: The Clifford isometry in iAPNF

For each input of the dotted subdiagram from Procedure 2, traverse the flowchart below and resolve the cases accordingly.



Case 1. The simplest case is when there is nothing to do: the input was already the pivot of an X spider in the rAPNF. This implies it is connected to no CZ edges to the outputs and to no other X spiders.

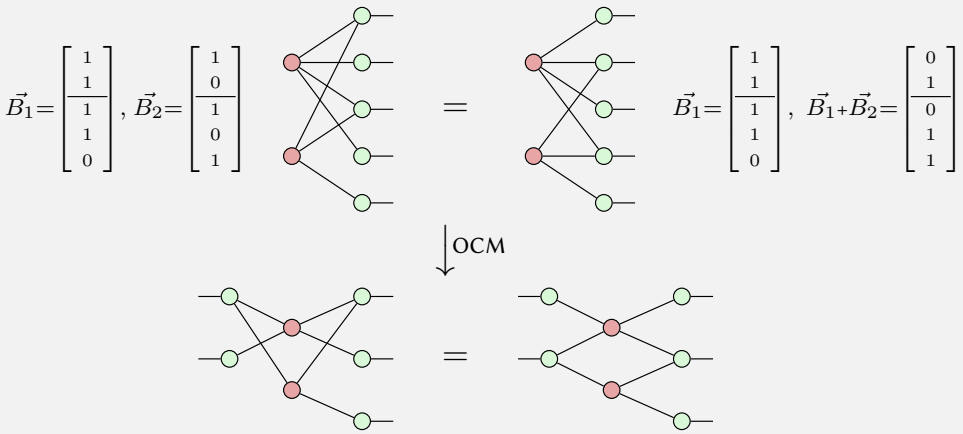
Case 2. The second simplest case is when the input is connected to no X spiders. By Lemma 7.19, there has to be at least one CZ edge to the outputs. Therefore, we can transform the Z spider to an X spider, with the same connectivity as all its CZ edges. This produces an H gate that is added to U_{in}.



(7.21)

Case 3. When the input is connected to two or more X spiders, we can add the connectivity of one (say, the topmost one) to all of the others [Kis22]. This corresponds precisely to row

addition in the biadjacency matrix B. An example is depicted below.



(7.22)

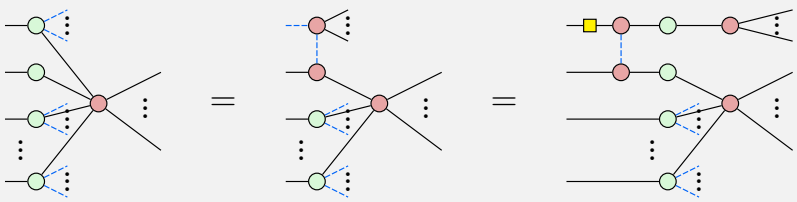
As a result, the input is connected to only one X spider, and will match one of the subsequent cases. Any existing pivots of these X spiders is preserved by this row addition in the biadjacency matrix B, as by definition of being a pivot it is guaranteed to be the only 1 in its column.

The remaining cases are of the input being connected to exactly one X spider, which it is not the pivot of.

Case 4. If that X spider does not already have an input as a pivot, make this input its pivot. As each X spider already has a pivot due to the earlier algorithm rAPNF, this means that its pivot was an output rather than an input. Note that this procedure can create CZ edges from other inputs of that X spider to the outputs. This causes no issues with existing inputs which are pivots, because we already asserted that none of them are connected to this X spider.

For subsequent cases, the input is connected to exactly one X spider, which already has a pivot which is another input. We have from Lemma 7.17 that the X spider must be connected to at least one output, and from Lemma 7.19 that this input must have at least one CZ gate to the outputs.

Case 5. Since the X spider already has a pivot which is another input, we can unfuse a CX gate with the control being the input in question and the target being the pivot. We can then transform the Z spider to an X spider, adding H; DX to U_{in}.



(7.23)

Theorem 7.24. *Every Clifford isometry can be written to a normal form composed of the following layers:* $\{S, CZ\}$; $\{H\}$; $\{DX\}$; E_{CSS}; $\{X\}$; $\{S, CZ\}$ *We call this* isometry AP normal form (iAPNF). *For stabilizer codes, we can directly read off the pure Z-type stabilizer generators* M_Z. *Likewise, by efficiently applying the algorithm to rewrite the inner CSS code phase-free encoder from X-Z to Z-X GPF, we can read off X-type stabilizer generators* M_X *which can then be pushed through the rightmost diagonal Clifford unitary to obtain the mixed stabilizer generators* $M_X G$. *We can furthermore read off the logical operators of the inner CSS code, which if conjugated through the logical unitary on the inputs, results in the ability to read off the complete stabilizer tableau. If taking the joint +1 eigenspace convention, there is no* $\{X\}$ *layer, and the only allowed powers of* S *in the final layer are* 0 *and* 1.

It is interesting that we are able to recover iAPNF without needing more than three layers of unitary gates on the logical qubits, only two of which are entangling. This number of distinct circuits possible with three layers, is certainly fewer than the number of all possible Clifford unitaries on k qubits. The number is even lower because the H and DX layers are not independent from each other: In Case 5 the original pivot can never have an H gate before its DX gate(s). Indeed, it is apparent some unitaries that are missing. Another way to look at this is that unitaries not generatable by these three layers must satisfy a symmetry property with respect to all possible iAPNF parameters, as we know that iAPNF holds for all Clifford isometries. For example, any SWAP of inputs has the same action as changing columns of the parameters $(B, \vec{b})$ of iAPNF. Another missing unitary, CX, has the same action as a change of B, specifically adding the column of the target to the column of the control. As for $X(\frac{\pi}{2})$, it has the same action as changing (B, A) of Z-X iAPNF, which can be seen by doing local complementation.

Chapter summary

In this chapter, we focused on the stabilizer fragment of the qubit ZX-calculus. We gave an unconventional fully graphical presentation of quantum error correcting codes, and connected this to graph-based, projector, circuit, and tableau representations of stabilizer codes. We gave a new unique normal form for any Clifford isometry generalizing the AP normal form for any stabilizer state, and related this to a tableau normal form for any stabilizer code. As sample applications, we looked at switching between QEC codes to achieve a universal fault-tolerant gate set, and mapping local fermionic operators to local qubit operators.

The immediate next steps would be to write down the normal form derivation for any odd prime qudit dimensional Clifford isometry, generalizing the AP normal form for states we achieved in Ref. [PBC+23]. While here we can decompose arbitrary stabilizer code encoders to reveal a compatible inner CSS code encoder, the aspiration is to flip this in the opposite

direction: Define an efficient (to the extent possible for an exponential size search space) search procedure to find a good stabilizer code which preserves the transversal diagonal operators of the inner CSS code, and hopefully other good code parameters. In sample Chapter 10.5, we outline the problem statement and a few ideas, of searching for a good qutrit stabilizer code with an interesting transversal emulated binary gate.

Part IV

Qutritland

Chapter Contents

8 The metaplectic gate set is a subset of Clifford+T **125**
8.1 The qutrit Clifford+T gates . . . 127
8.2 Finding the metaplectic gate in qutrit Clifford+T . . . 132

9 Controlling all of qutrit Clifford+T **137**
9.1 Controlled unitaries . . . 138
9.2 Adding controls to Clifford+T gates . . . 141
9.3 Building multiple-controlled Clifford+T gates . . . 145

10 Building diagonal gates with phase gadgets **151**
10.1 The qutrit ZX-calculus . . . 152
10.2 Constructing phase gadgets, controlled phase gates, and phase multipliers . . . 156
10.3 Emulating diagonal qubit unitaries with qutrit phase gadgets . . . 165
10.4 Sample: Superconducting implementation of qutrit phase multipliers . . . 167
10.5 Sample: Emulating qubits with qutrits . . . 172

While we overviewed qudit gates in Chapter 5, those definitions were given for all qudit dimensions or for all prime qudit dimensions. As this chapter focuses on qutrits ($d = 3$), we first review the matrices which generate the qutrit Pauli, Clifford, and Clifford+T gates in Figure. 7.2.

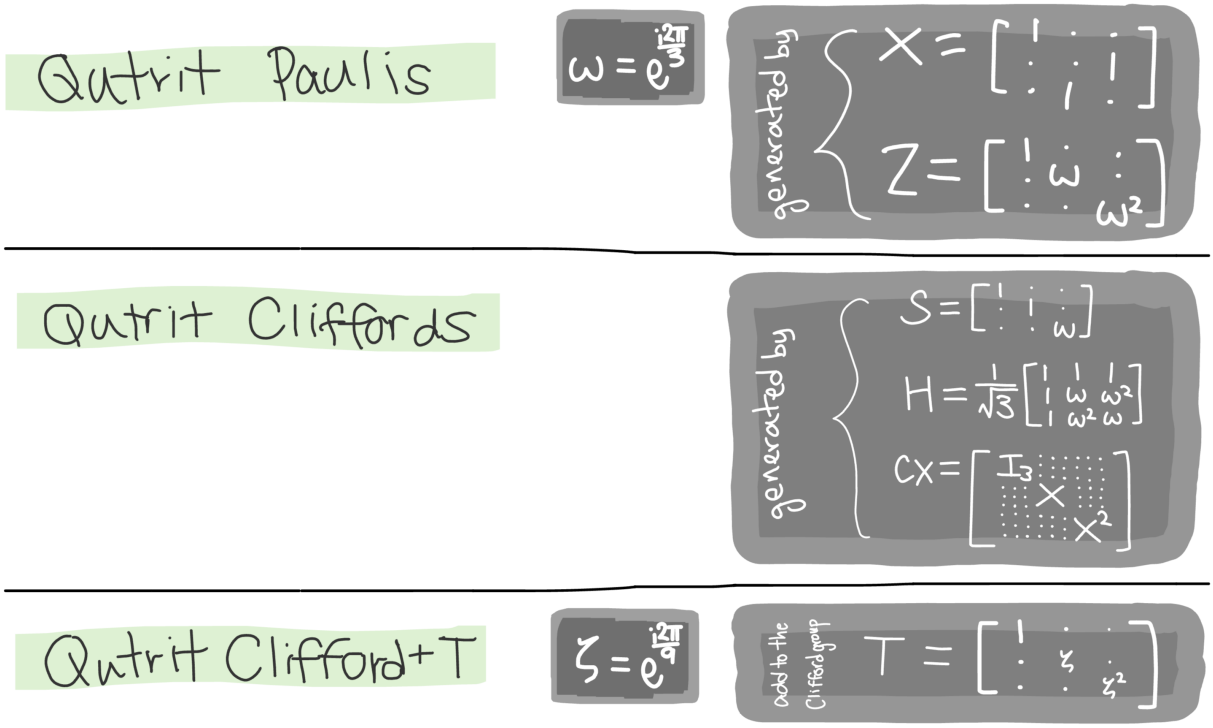


Figure 7.2: Here are the qutrit-specific matrices generating the Pauli, Clifford, and Clifford+T gates.

Classical computing technology works with bits, where the state of the fundamental information unit can be in one of two states. It is then not surprising that quantum computing researchers have mostly studied *qu*bits, where the fundamental unit of information can be in a superposition of two states. However, there are several benefits we can get by working instead with qu*dits*, where we work with higher-dimensional systems. One such benefit is that many proposed physical types of qubits are actually restricted subspaces of higher-dimensional systems, where the natural dimension can be much higher. By working with qudits we can exploit the additional degrees of freedom present in the system. Qudit quantum processors based on ion traps [RMP+22b] and superconducting devices [BRS+21; YZZY18; YSK+20] have already been demonstrated. By using the otherwise wasted dimensions accessible in a qudit we increases the device's information density, which leads to advantages in for instance runtime efficiency, resource requirements, magic state distillation noise thresholds, and noise resilience in communication [WHSK20; CDBO19; Cam14; Pra20]. These were just a few, but we will touch upon advantages of qudits in more detail in the next chapter.

For qudits to make a good foundation for a quantum computer, we need techniques to do

fault-tolerant computation with them. A well-studied approach for fault-tolerant computation with qubits is based on the observation that many quantum error correcting codes can natively implement Clifford gates, so that we only need to realize a fault-tolerant implementation of some non-Clifford gate. A popular choice for this gate is the T gate, a single-qubit gate that can be implemented by injecting its magic state into a circuit. As these magic states can be distilled to a desired level of fidelity, we can then implement approximately universal quantum computation fault-tolerantly [BK05]. Qudit analogues of this Clifford+T gate set have been developed, so that this approach of magic state distillation and injection can be used to do fault-tolerant computation for qudits of any dimension [CAB12].

In this chapter we will work with qu*trits*, three-dimensional systems. These are the most well-studied higher-dimensional qudit. Qutrits have been used to improve qubit readout [MOP+09] and reduce the circuit complexity of implementing multi-controlled qubit gates [GBD+19; KNX+20; ISM09; RRG07; LBA+09b]. By replacing some or all of the information carriers to be qutrits instead of qubits, the $|2\rangle$ energy level of the qutrit can be utilized to reduce resource requirements in terms of number of ancillae, entanglement complexity, gate depth and gate count, and non-Clifford gate count.

8

The metaplectic gate set is a subset of Clifford+T

Remark 8.1. This section is based on the paper "Qutrit Metaplectic Gates Are a Subset of Clifford+T" [GRvdWY22a], in collaboration with Andrew Glaudell, Neil J. Ross, and John van de Wetering.

Like the qubit setting, the qutrit Clifford+T gate set can be implemented fault-tolerantly on qutrit error correcting codes [CW15]. Having said this, for qutrits, another fault-tolerant gate set has garnered considerable study in the literature. Instead of the T gate, the R gate is another single-qutrit non-Clifford gate that can be added to the Clifford group to obtain a universal gate set. The R gate is also known as the *metaplectic* gate, the *reflection* gate, and the FLIP gate. It is defined as

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{8.2}$$

The Clifford+R gate set, also called the qutrit metaplectic gate set, has been studied in many papers prior to this work [ACB12a; CW15; CHW15; BCKW16; BCRS16; Boc16; BRS17]. It was first defined in Ref. [ACB12a], where it was shown to admit a magic state distillation and injection protocol. As it is a non-Clifford gate, it enables approximately universal quantum

computation when added to the Clifford gate set [Got99], as explicitly proven in Ref. [CW15, Theorem 2]. A proposed implementation for it uses certain weakly-integral non-abelian anyons with braiding and topological measurement [CHW15; CW15]. A recent preprint exhaustively searched the whole Clifford+R group as well as just Householder reflections, and found the asymptotic costs for this approximate synthesis to be better than that of qubits [GLL+25].

The definition of the R gate is deceptively simple, containing only 1's, 0's and a -1. However, it is in fact nowhere in the qutrit Clifford hierarchy [CGK17]. This is because for qutrits, Clifford gates are based on the third root of unity $\omega = e^{i2\pi/3}$. Despite this fact, the R gate can still be injected into a qutrit circuit using a repeat-until-success procedure of an R magic state which also allows a distillation protocol [ACB12a]. The R gate can hence also be realized fault-tolerantly [ACB12a]. Another construction of R is by a measurement-assisted repeat-until-success protocol requiring two ancillary qutrits to probabilistically realize it out of Clifford gates [CHW15]. The R gate has been suggested to be "more powerful in practice" than the T gate, in a study comparing Clifford+R to Clifford+T circuit widths and depths for compiling Shor's algorithm [BRS17]. Further nuances of the cost-effectiveness of these two gate sets under different resource estimation assumptions remains an open question.

Ref. [BRS17] computed the cost of approximating the third level of the Clifford hierarchy in the Clifford+R (which they refer to as the *metaplectic*) gate set, and claimed that constructing the R gate in the Clifford+T gate set requires multiple ancillae and repeat-until-success circuits. In Ref. [GRvdWY22a], we disproved this claim. We show that while no single-qutrit Clifford+T circuit composes to an R gate unitarily[1], rather unexpectedly, the R gate is exactly constructible through a unitary two-qutrit Clifford+T circuit with T-count 39, which we next construct. This demonstrates that $R \in$ Clifford+T.

We further prove that the converse is not true, i.e. that $T \notin$ Clifford+R, and hence Clifford+R $\subsetneq$ Clifford+T. As a consequence, any Clifford+R computation can be exactly implemented through Clifford+T gates with constant overhead, whereas there exist Clifford+T circuits whose implementation via Clifford+R must strictly increase in overhead with the desired precision—by the Solovay-Kitaev theorem, this overhead is expected to be polylog in the gate count divided by the precision [NC10]. We have reason to believe that the additional gates Clifford+T can exactly represent, which Clifford+R cannot, are important in practice. It is conjectured that not all ternary classical reversible gates can be exactly represented in Clifford+R [BCRS16], whereas in the next section we show they can all be exactly represented in Clifford+T [YvdW22].

The result that unitaries over Clifford+R are a strict subset of unitaries over Clifford+T may seem to be a contradiction, due to the fact that R is nowhere in the Clifford hierarchy, whereas

[1]Unless stated otherwise, we take "single-qutrit" to mean ancilla-free.

every Clifford gate and the T gate is in the third level C_3. However, while C_1 and C_2 are closed under composition, C_3 and higher levels of the Clifford hierarchy are not. Notably, it is not true that Clifford+T circuits implement unitaries in C_3.

8.1 The qutrit Clifford+T gates

and the ninth root of unity

Recall from Chapter 5 that in any prime qudit dimension, the set of n-qudit Cliffords forms a group under composition. This group is generated by the S, Hadamard, and CX gates.

In this section, in addition to introducing the Clifford+T gates, we leverage techniques from exact circuit synthesis. The following statement is attributable to many authors in the field, most notably Kliuchnikov, Maslov, and Mosca [KMM13]:

Lemma 8.3. *Let $\mathcal{G} = \{G_1, \cdots, G_k\}$ be a quantum gate set. For all $j \in \{1, \ldots, k\}$, let each G_j have the computational basis matrix representation M_j up to a complex global phase such that M_j is a matrix with entries in the number ring R. Then, up to a global phase, the matrix representation of any circuit over $\mathcal{G}$ only has entries in the number ring R.*

To better describe single-qutrit Clifford gates, we first define a notation for Z and X phase gates which are Clifford.

Definition 8.4. We write $Z(a, b)$ for the *phase gate* that acts as $Z(a, b)\,|0\rangle = |0\rangle$, $Z(a, b)\,|1\rangle = \omega^a\,|1\rangle$ and $Z(a, b)\,|2\rangle = \omega^b\,|2\rangle$ where we take $a, b \in \mathbb{R}$.

The collection of all $Z(a, b)$ operators constitutes the group of diagonal single-qutrit unitaries modded out by a global phase. Composition of these operations is given by $Z(a, b) \cdot Z(c, d) = Z(a + c, b + d)$. The reason we take the phases modulo $\frac{2\pi}{3}$ is to have the convenient property that $Z(a, b)$ is Clifford iff a and b are integers.

Definition 8.5. The qutrit S gate is $S \coloneqq Z(0, 1)$. I.e., it multiplies the $|2\rangle$ state by ω.

For qubits, the Hadamard interchanges the Z eigenbasis $\{|0\rangle, |1\rangle\}$, and the X basis consisting of the states $|\pm\rangle \coloneqq \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$. The same holds for the qutrit Hadamard[2]. In this case the X

[2]Strictly speaking, it is incorrect to call the higher-dimensional single-qudit quantum Fourier transform a Hadamard gate as the matrix is not made of 1's and -1's. We call it Hadamard to be consistent with prior literature on qudit computation.

basis consists of the following states (where we recall from above that $\frac{1}{\omega^2-\omega} = i/\sqrt{3}$):

$$|+\rangle \coloneqq \frac{1}{\omega^2-\omega}(|0\rangle + |1\rangle + |2\rangle) \tag{8.6}$$

$$|\omega\rangle \coloneqq \frac{1}{\omega^2-\omega}(|0\rangle + \omega\,|1\rangle + \omega^2\,|2\rangle) \tag{8.7}$$

$$|\omega^2\rangle \coloneqq \frac{1}{\omega^2-\omega}(|0\rangle + \omega^2\,|1\rangle + \omega\,|2\rangle) \tag{8.8}$$

Definition 8.9. The *qutrit Hadamard gate* H is the gate that maps $|0\rangle \mapsto |+\rangle$, $|1\rangle \mapsto |\omega\rangle$ and $|2\rangle \mapsto |\omega^2\rangle$. As a matrix:

$$H \coloneqq \frac{1}{\omega^2-\omega}\begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{pmatrix} \tag{8.10}$$

Unlike the qubit Hadamard, the qutrit Hadamard is *not* self-inverse. In fact, $H^2 = -X_{12}$, so $H^4 = \mathbb{I}$. In particular, $H^\dagger = H^3$. Furthermore, we note that just as the Clifford group in qubits generates certain global phases, the relation $(SH)^3 = -\omega$ implies that global phases of ± 1, $\pm\omega$, and $\pm\omega^2$ naturally appear in the qutrit Clifford group.

The Paulis and S gates have matrix representations with entries over $\mathbb{Z}[\omega]$. While H inherently introduces a denominator, rather than the conventional factor of $\frac{1}{\sqrt{3}}$, we chose $\frac{1}{\omega^2-\omega}$ to have a different global phase. This is because we can localize $\mathbb{Z}[\omega]$ to ensure we can characterize circuits containing H gates. Since

$$\frac{\omega^k}{\omega^2-\omega} = \frac{\omega^k(\omega-\omega^2)}{3}$$

we can introduce the appropriate denominators by localizing at $\mu = \{3^k \mid k \in \mathbb{N}\}$ to get the number ring $\mu^{-1}\mathbb{Z}[\omega]$. Note that this is equivalent to the number ring $\mathbb{T}[\omega]$ which consists of elements $a + b\omega$ where $a, b \in \mathbb{T}$ are triadic fractions:

Definition 8.11. The ring of *triadic fractions* is the number ring defined by localizing $\mathbb{Z}$ at the set $\mu = \{3^k \mid k \in \mathbb{N}\}$, which we denote as $\mathbb{T} \coloneqq \mu^{-1}\mathbb{Z} = \{a/3^k \mid a \in \mathbb{Z}, k \in \mathbb{N}\}$.

In Definition 8.4 we defined the Z phase gate. Similarly, we can define the X phase gates, that give a phase to the X basis states.

Definition 8.12. We define the *X phase gates* to be $X(a,b) \coloneqq HZ(a,b)H^\dagger$ where $a, b \in \mathbb{R}$.

We already saw examples of such X phase gates: $X = X(2,1)$ and $X^\dagger = X(1,2)$.

For a qubit there is only one non-trivial permutation of the standard basis states, which is implemented by the X gate. For qutrits there are five non-trivial permutations of the basis states, which all happen to be Clifford, which is not the case for higher qudit dimensions. We call these X gates and we specify them as X_L where L is a permutation of the elements $\{0, 1, 2\}$ written in cycle notation, omitting elements which are mapped to themselves. For example, X_{02} is the permutation which maps $|0\rangle \mapsto |2\rangle$, $|1\rangle \mapsto |1\rangle$, and $|2\rangle \mapsto |0\rangle$. The five non-trivial permutations are then X_{01}, X_{12}, X_{02}, X_{012}, and X_{021} along with the trivial identity permutation $\mathbb{I} = X_{\emptyset}$. Compositions of these operators are given by $X_L \cdot X_M = X_{L \cdot M}$ with $L \cdot M$ the composition of permutations. Note that $X_{012} = X$ and $X_{021} = X^\dagger$.

Any single-qutrit Clifford can be represented (up to global phase) as a composition of Clifford Z and X phase gates. In particular, we can represent the qutrit Hadamard as follows [GW17]:

$$H = -Z(2,2)X(2,2)Z(2,2) = -X(2,2)Z(2,2)X(2,2) \tag{8.13}$$

$$H^\dagger = -Z(1,1)X(1,1)Z(1,1) = -X(1,1)Z(1,1)X(1,1) \tag{8.14}$$

The third and final Clifford gate needed to generate the Clifford group is the CX gate.

Definition 8.15. The qutrit CX gate is the two-qutrit gate defined by $\text{CX}\,|i, j\rangle = |i, i + j\rangle$ where the addition is taken modulo 3.

Proposition 8.16. *Let U be a qutrit Clifford unitary. Then up to global phase U can be written as a composition of the S, H and CX gates [Got99].*

Corollary 8.17. *Let U be a qutrit Clifford unitary. Then up to a global phase U has a matrix representation in the computational basis with entries in the number ring $\mathbb{T}[\omega]$.*

Proof. This follows from Proposition 8.16, the definitions of S, H, and CX, and Lemma 8.3. □

Definition 8.18. The qutrit T gate is the Z phase gate defined as $T := Z(1/3, -1/3) = \text{diag}(1, \zeta, \zeta^8)$, where $\zeta = e^{i\frac{2\pi}{9}}$ is a ninth root of unity [PJKS18; CAB12; HV12].

Remark 8.19. Throughout this thesis, definitions of gates may vary, in order to be consistent with the paper in which these results appear, and the code base used to verify them. The two sources of these differences can both be attributed to choices of convention which do not alter the essential properties of the gate with respect to the Clifford group:

1. The choice of the global phase of the gate, a degree of freedom inherent to defining

the Clifford hierarchy based on conjugating the Pauli group (Definition 3.21)

2. Differences in convention across the literature of third-level gates in the Clifford hierarchy being up to Clifford equivalence, i.e. equal modulo a Clifford unitary

How to resolve these differences is typically straightforward, by direct substitution of the difference in global phase and/or Clifford gate. The main situation where more care must be taken with this substitution is when the gate is being controlled, and thus instead of a global phase, this is a *controlled global phase*—a concept which we define shortly.

Like the qubit T gate, the qutrit T gate belongs to the third level of the Clifford hierarchy, can be injected into a circuit using magic states, and its magic states can be distilled by magic state distillation [CAB12]. It can be fault-tolerantly implemented in a number of qutrit error correcting codes. The qutrit Clifford+T gate set is also approximately universal [CW15, Theorem 1].

The T gate introduces the phase ζ into matrix representations of circuits and thus we should consider extending the previously-defined $\mathbb{T}[\omega]$ by ζ. Note that ζ is a ninth root of unity which solves the cubic polynomial

$$\zeta^3 - \omega = 0 \tag{8.20}$$

over $\mathbb{T}[\omega]$. In fact, this polynomial has no solutions over $\mathbb{T}[\omega]$, implying that $\zeta \notin \mathbb{T}[\omega]$ (for more details, refer to the Appendix of the associated paper [GRvdWY22a]). We thus define the number ring $\mathbb{T}[\zeta]$:

Definition 8.21. The extension of $\mathbb{T}[\omega]$ by ζ is the number ring $\mathbb{T}[\omega][\zeta] \cong \mathbb{T}[\zeta]$ defined by

$$\mathbb{T}[\omega][\zeta] \cong \mathbb{T}[\zeta] \coloneqq \{a + b\zeta + c\zeta^2 + d\zeta^3 + e\zeta^4 + f\zeta^5 \mid a, b, c, d, e, f \in \mathbb{T}\}.$$

Any higher powers of ζ that might appear in an expression for an element of $\mathbb{T}[\zeta]$ can be reduced using for instance Equation (8.20).

Lemma 8.22. *Let U be a qutrit Clifford+T unitary. Then up to a global phase U has a matrix representation in the computational basis with entries in the number ring $\mathbb{T}[\zeta]$.*

Proof. By the definitions of S, H, T, and CX and Lemma 8.3. □

Using T gates, we can construct certain controlled unitaries. When we have an n-qu*bit* unitary U, we can speak of the controlled gate that implements U. This is the $(n+1)$-qubit gate that acts as the identity when the first qubit is in the $|0\rangle$ state, and implements U on the last n qubits if the first qubit is in the $|1\rangle$ state.

For qutrits there are however multiple notions of control.

Definition 8.23. Let U be a qutrit unitary. Then the $|2\rangle$-*controlled* U is the unitary $|2\rangle$-U that acts as

$$|0\rangle \otimes |\psi\rangle \mapsto |0\rangle \otimes |\psi\rangle \qquad |1\rangle \otimes |\psi\rangle \mapsto |1\rangle \otimes |\psi\rangle \qquad |2\rangle \otimes |\psi\rangle \mapsto |2\rangle \otimes U|\psi\rangle$$

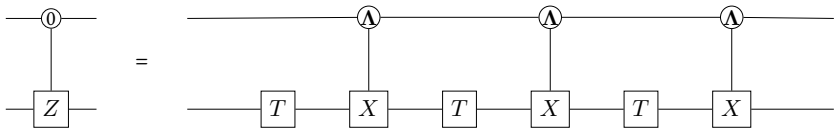
I.e., it implements U on the last qutrits if and only if the first qutrit is in the $|2\rangle$ state.

Note that by conjugating the first qutrit with X or $X^\dagger$ gates we can make the gate also be controlled on the $|1\rangle$ or $|0\rangle$ state.

A different notion of qutrit control was introduced by Bocharov, Roetteler, and Svore [BRS17]: Given a qutrit unitary U they define $\Lambda(U)|c\rangle|t\rangle = |c\rangle \otimes (U^c|t\rangle)$. I.e., apply the unitary U a number of times equal to to the value of the control qutrit, so that if the control qutrit is $|2\rangle$ we apply U^2 to the target qutrits. Note that we can get this notion of control from the former one: just apply a $|1\rangle$-controlled U, followed by a $|2\rangle$-controlled U^2. The Clifford CX gate defined earlier is in this notation equal to $\Lambda(X)$.

Adding controls to a Clifford gate generally makes it non-Clifford. In the case of the CX gate, which is $\Lambda(X)$, it is still Clifford, but the $|2\rangle$-controlled X is *not*.

As shown by Bocharov, Roetteler, and Svore, the $|0\rangle$-controlled Z gate can be constructed by the following 3 T gate circuit [BRS17, Figure 6]:

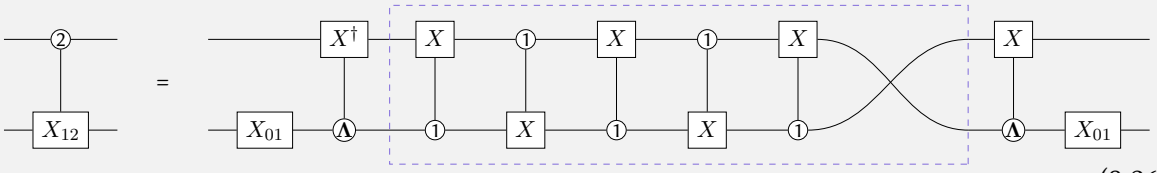


(8.24)

By conjugating the control qutrit by either $X^\dagger$ or X, the $|1\rangle$- and $|2\rangle$-controlled versions are respectively obtained. Taking the adjoint of Equation (8.24) has the effect of changing the target operation from Z to $Z^\dagger$. Finally, note that we can also use this construction for controlled X and $X^\dagger$ gates by conjugating the target qutrit of Equation (8.24) by an H or $H^\dagger$ gate. By adapting a different circuit from Ref. [BRS17] we can also construct the other controlled X permutation gates.

Lemma 8.25. *The $|2\rangle$-controlled versions of the X_{01}, X_{02}, and X_{12} gates can be implemented unitarily using Clifford+T gates without ancillae, with a T-count of 15.*

Proof. The $|2\rangle$-controlled X_{12} gate can be constructed as follows:



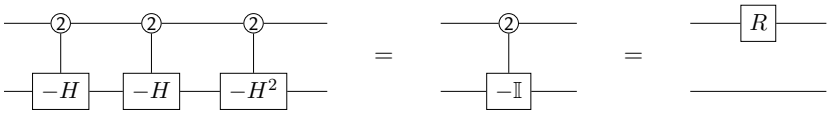
(8.26)

The dashed box shows the Clifford equivalent gate given by Bocharov, Roetteler, and Svore [BRS17; BCRS16] upon which the construction is based. The $|2\rangle$-controlled X_{01} and $|2\rangle$-controlled X_{02} gates follow from Clifford equivalence. □

8.2 Finding the metaplectic gate in qutrit Clifford+T

Remark 8.27. The code repository verifying the qutrit circuit constructions in this work is at https://github.com/lia-approves/qudit-circuits/tree/main/qutrit_R_from_T. Following the initial code base by the author of this thesis in Matlab, Alex Christopher Lim and Anikait Mundhra developed a Python implementation.

Previous implementations of the R gate require either distillation [ACB12a] or probabilistic creation of the diag$(1, 1, -1)$ state [CHW15; BRS17]; both approaches then necessitate injection by a repeat-until-success protocol. Here we present a new approach, which implements R unitarily over the qutrit Clifford+T gate set. As we will discuss later, it is actually impossible to exactly build the R gate from only single-qutrit Clifford+T gates. However, we *can* construct the two-qutrit $R \otimes \mathbb{I}$ unitarily using Clifford+T gates. We will do this by showing how to construct certain $|2\rangle$-controlled gates and then using the following observation:



(8.28)

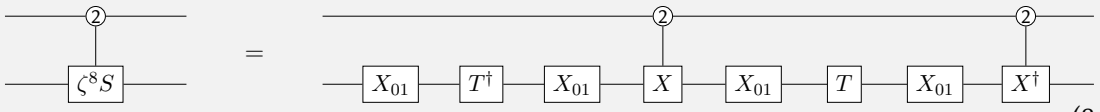
This works because $H^4 = \mathbb{I}$, and the fact that global phases become *local* phases when adding control wires to them. Here we have a *controlled* global phase because the global phase of -1 is applied to the target if and only if the control is in the $|2\rangle$ state. Therefore, this is an instance of phase kickback: The action of the $|2\rangle$-controlled $-\mathbb{I}$ gate is identical to applying $R \otimes \mathbb{I}$, i.e. the R gate to the control qutrit and identity on the target.

The $|2\rangle$-control of the gate $-H^2 = X_{12}$ was constructed as a Clifford+T circuit in Equation (8.26). It hence remains to show how we can construct the $|2\rangle$-controlled $-H$ gate in Clifford+T. We can build this using the $|2\rangle$-controlled S gate. Note that due to our particular choice of global phase convention for the Clifford gates, our constructions of controlled Clif-

fords are up to a controlled global phase. In work later in this thesis [YvdW22], we chose a different convention in order to do away with the controlled global phases here and exactly construct all multiple-controlled Clifford+T gates in Clifford+T.

Lemma 8.29. *The $|2\rangle$-controlled S gate can be constructed unitarily without ancillae, up to a controlled global phase of ζ^8, using only Clifford+T gates, with T-count 8.*

Proof. The correctness can be verified by direct computation of the following circuit.

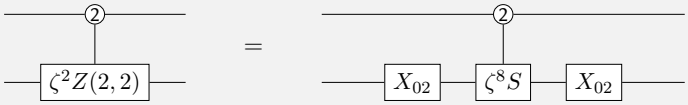


(8.30)

Alternatively, it is easy to see that this circuit does nothing if the first qutrit is in the $|0\rangle$ or $|1\rangle$ state, as the X_{01} gates cancel, so that the T can cancel with the $T^\dagger$. Otherwise, if the first qutrit is $|2\rangle$, then the middle permutations combine to $X_{01}XX_{01} = X_{012} = X^\dagger$. When the T is pushed through this, the phases get permuted, and when combined with the $T^\dagger$ gives an S gate up to global phase. □

Corollary 8.31. *The $|2\rangle$-controlled $Z(2,2) = diag(1,\omega^2,\omega^2)$ gate can be constructed unitarily without ancillae, up to a controlled global phase of ζ^2, using only Clifford+T gates, with T-count 8.*

Proof. Use the following circuit:



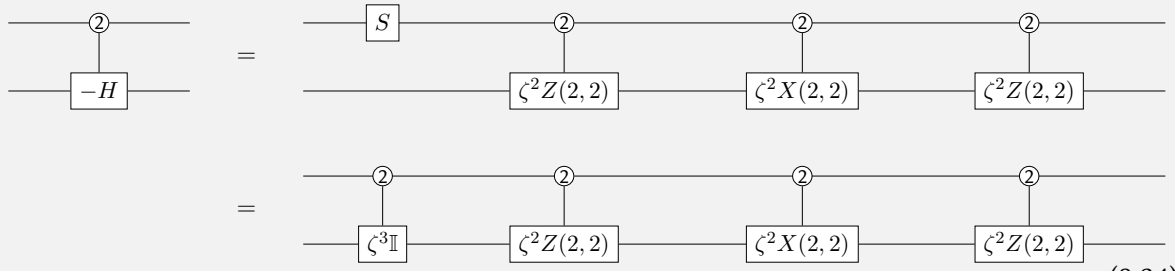
(8.32)

Its correctness can be verified by direct computation, or by commuting S and X_{02}. □

Lemma 8.33. *The $|2\rangle$-controlled H gate can be constructed unitarily without ancillae, up to a controlled global phase of -1, using Clifford+T gates with T-count 24.*

Proof. In the construction given below, we use the decomposition of H into alternating Z

and X Clifford rotations of Equation (8.13).



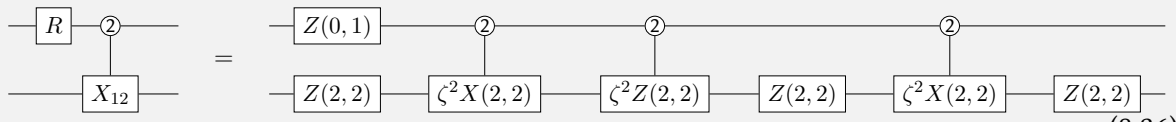
(8.34)

To construct the controlled H up to a controlled global phase, we apply Equation (8.32), conjugating the target by Hadamards for the X rotations per Definition 8.12. As we require three such gates, their combined $|2\rangle$-controlled global phase becomes $\zeta^{2\cdot 3} = \zeta^6$. As $\zeta^9 = 1$, the necessary correction is to apply the $|2\rangle$-controlled global phase of ζ^3 gate, i.e. the $Z(0,1) \otimes \mathbb{I}$ gate. □

We can now construct the R gate in Clifford+T. However, direct substitution of the 24 T-count $|2\rangle$-controlled H gate of Lemma 8.33 into Equation (8.28) yields a T-count 63 construction. We can do better by combining the two iterations of the controlled H in a smarter way.

Theorem 8.35. *The qutrit R gate can be constructed unitarily in Clifford+T with T-count 39, provided there is a borrowed (i.e. returned to its starting state) ancilla available.*

Proof. The equality of the circuits below can be verified by direct computation or by noting that it applies $Z(3,3) = \mathbb{I}$ to the target when the control is $|0\rangle$ or $|1\rangle$, and $H^2 = -X_{12}$ otherwise.



(8.36)

To get a circuit for the R gate we simply bring the $|2\rangle$-controlled X_{12} to the other side (as it is its own inverse). The total T-count of the resulting circuit is then $15 + 3 \cdot 8 = 39$. □

As we can construct the R gate as a Clifford+T circuit, any unitary that can be exactly constructed in the Clifford+R gate set can then be exactly (as opposed to approximately) constructed in the Clifford+T gate set. Although do note that our conversion presently seems rather inefficient, as the circuit in Equation (8.28) requires 39 T gates.

Corollary 8.37. *The gates implementable over the Clifford+R gate set is a subset of the gates implementable over the Clifford+T gate set.*

A natural question to ask now is whether we can do better. Do we really need two qutrits to

write the R gate as a Clifford+T unitary? The answer is *yes*: it is not possible to construct the R gate using just single-qutrit Clifford+T gates. This follows from the normal form that was found for single-qutrit Clifford+T unitaries in Ref. [GRT19]. Since the proof of this is rather technical we give a sketch here; refer to the Appendix of the associated paper for the full proof [GRvdWY22a].

The group of 3×3 unitary matrices acts on the 8-dimensional real vector space of traceless Hermitian matrices. This action defines, for each 3×3 unitary matrix U, an 8×8 real matrix $\overline{U}$ known as the *adjoint representation* of U. One can then gather information about U by studying its adjoint representation $\overline{U}$. In particular, it is a consequence of the normal forms for single-qutrit Clifford+T circuits introduced in Ref. [GRT19] that the adjoint representation of a single-qutrit Clifford+T operator has a very specific block matrix form. It can then be shown by computation that $\overline{R}$ is not of the appropriate form and therefore not Clifford+T .

Another natural question is the converse to Corollary 8.37: is the T gate included in Clifford+R? I.e., is the inclusion of Clifford+R within Clifford+T strict? We will show that this is indeed the case. We begin by considering matrix representations of circuits over Clifford+R.

Lemma 8.38. *Let U be a qutrit Clifford+R unitary. Then up to a global phase U has a matrix representation in the computational basis with entries in the number ring $\mathbb{T}[\omega]$.*

Proof. By the definitions of S, H, R, and CX and Lemma 8.3. □

Proposition 8.39. *$T \notin$ Clifford+R.*

Proof. We have $T \in$ Clifford+R if there exists a unitary circuit over Clifford+R which performs the operation $T \otimes \mathbb{I}_n$ up to a global phase for some $n \in \mathbb{N}$ where $\mathbb{I}_n$ is the n-qutrit identity. In the computational basis, $T \otimes \mathbb{I}$ has a matrix representation with entries from the set $\{0, 1, \zeta, \zeta^8\}$. By Lemma 8.38, we know that if $T \otimes \mathbb{I}$ permits an exact circuit over Clifford+R we must have $\{0, c, c\zeta, c\zeta^8\} \subset \mathbb{T}[\omega]$ for at least some global phase $c \in \mathbb{C}$ which satisfies $c^*c = 1$. As $\mathbb{T}[\omega]$ is closed under conjugation, we then also have $c^* \in \mathbb{T}[\omega]$, and as it is closed under multiplication we then have $c^*c\zeta = \zeta \in \mathbb{T}[\omega]$. However, it is well-known that $\zeta \notin \mathbb{T}[\omega]$, and so there exists no such global phase c. Hence, no such suitable c exists. As n was arbitrary, we conclude that no Clifford+R circuit exactly implements T in the computational basis. □

Corollary 8.40. *Clifford+$R \subsetneq$ Clifford+T.*

This means that contrary to previous claims that Clifford+R is more powerful than Clifford+T,

Clifford+T is strictly more powerful (at least asymptotically and for exact synthesis) than Clifford+R. Clifford+T can exactly synthesise every gate in Clifford+R up to a constant factor of overhead, while the converse is not true.

These constructions enable circuit synthesis results for Clifford+R to be directly translated, albeit inefficiently, to the Clifford+T setting. For example, the universal approximate synthesis algorithms of Refs. [BCKW16; Boc16] for Clifford+R, can be adapted to approximate synthesis with the Clifford+T gate set up to a (large, if you consider 39 a large number) constant factor overhead.

Our results demonstrate a way in which qutrit Clifford+T is different from that of qubit Clifford+T. While all the one-qubit Clifford+T circuits that can be constructed with and without ancillae coincide [KMM13], our result shows that this is not true for qutrits, as the single-qutrit R gate cannot be constructed in single-qutrit Clifford+T, but can be constructed using one borrowed ancilla.

9

Controlling all of qutrit Clifford+T

Remark 9.1. This section is based on the paper "Constructing All Qutrit Controlled Clifford+T gates in Clifford+T" [YvdW22], in collaboration with John van de Wetering. We thank Andrew Glaudell for pointing out Equation (9.34).

Our focus for this section is on what we will call $|2\rangle$-*controlled gates* — controlled unitaries that only fire when the control qutrit is in the $|2\rangle$ state, so that this gate acts as the identity on the target if the control is in the $|0\rangle$ or $|1\rangle$ state. In fact, many qutrit-native algorithms use $|2\cdots2\rangle$-controlled logic gates, including various ternary adders and incrementers [KP07; HWM17; GBD+19].

Given that it is natural for qutrit algorithms to use these types of controlled gates as building blocks, it is crucial to understand how we can implement these fault-tolerantly if we wish to use them for practical purposes. Unfortunately, while there is a 'naive' decomposition into qutrit Clifford+T gates of for instance the $|2\cdots2\rangle$-controlled X gate which uses $O(n)$ clean ancilla for n ternary controls [KP07], at the time of writing the paper this section is about, ancilla-free implementations required either uniformly controlled Givens rotations [KP06] or qutrit-controlled qubit gates [DW13], both of which are not fault-tolerant (at least as stated). The construction of Ref. [Mor16] is conceivably fault-tolerant, but it utilizes an exponential number of gates.

Ref. [BCRS16] describes an approach which could be applied to constructing the $|2\rangle^{\otimes k}$-controlled Z_{+1} gate for any k, and hence also the $|2\rangle^{\otimes k}$-controlled Z_{-1}, X_{+1} and X_{-1} gates for any k, by solving a system of linear equations modulo 3 where the number of equations is exponential in k. They present the explicit circuit for this for $k = 1$, but not for any $k > 1$. Their method does not suffice to construct the k-controlled X_{01} gate due to the X_{01} gate not being diagonal when conjugated by Hadamards.

This raises the question of how we can implement these $|2 \cdots 2\rangle$-controlled gates efficiently using more primitive and fault-tolerant gates.

In this section, we show that when we have any qutrit Clifford+T unitary U, we can construct an ancilla-free exact Clifford+T implementation of the $|2 \cdots 2\rangle$-controlled U unitary which uses a number of gates polynomial in the number of controls. Specifically, for k controls we require $O(k^{3.585})$ gates (this number comes from $\log_2 6 \approx 3.585$). Our work means in particular that we have fault-tolerant and ancilla-free implementations of all the constructions mentioned above. Note that after this work, we improved this to $O(k)$ gate count, which we present as Lemma 12.20 in the next chapter because it holds for all odd qudit dimensions [RvdWY23].

Note that our result, constructing controlled Clifford+T unitaries for any Clifford+T unitary, is not possible with qubits without allowing ancillae. For instance, it is not possible to construct a qubit controlled-T gate [KMM13] or a three-controlled Toffoli [GS13] using just Clifford+T gates and no ancillae.

As an application of our construction we give an algorithm for implementing any reversible classical trit function $f\colon \{0,1,2\}^n \to \{0,1,2\}^n$ as a unitary n-qutrit Clifford+T circuit using at most $O(3^n n^{3.585})$ Clifford+T gates—improved to $O(n3^n)$ after the discovery of Lemma 12.20. We find a lower bound for this problem of $O(3^n \cdot n/\log n)$ so that our result here is within a polynomial factor of optimal.

A software implementation of some of the key constructions is at https://github.com/lia-approves/qudit-circuits/tree/main/qutrit_control_Clifford_T.

9.1 Controlled unitaries

First, rather than re-define the qutrit gates of the previous section, we note that there are only two differences in convention between these two works: the global phase of the S and the H gates. Recall that the definitions of the Clifford gates (based on conjugation of the Pauli gates) are inherently up to global phase. The conventions here are chosen to make some of our results more elegant to state, whereas the previous section invoked properties of the number rings that more suited the other global phase convention.

Definition 9.2. The qutrit S gate is $S \coloneqq \zeta^8 \mathrm{diag}(1, 1, \omega)$. I.e. it multiplies the $|2\rangle$ state by the phase ω (up to a global phase).

The X basis consists of the following states:

$$|+\rangle \coloneqq \frac{-i}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle) \tag{9.3}$$

$$|\omega\rangle \coloneqq \frac{-i}{\sqrt{3}}(|0\rangle + \omega\,|1\rangle + \omega^2\,|2\rangle) \tag{9.4}$$

$$|\omega^2\rangle \coloneqq \frac{-i}{\sqrt{3}}(|0\rangle + \omega^2\,|1\rangle + \omega\,|2\rangle) \tag{9.5}$$

Definition 9.6. The *qutrit Hadamard gate* H is the gate that maps $|0\rangle \mapsto |+\rangle$, $|1\rangle \mapsto |\omega\rangle$ and $|2\rangle \mapsto |\omega^2\rangle$. As a matrix:

$$H \coloneqq \frac{-i}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{pmatrix} \tag{9.7}$$

We chose this global phase of the H gate to be in line with Ref. [GRT19].

In the previous section, we defined the Paulis, CX, T, $Z(a, b)$, and $X(a, b)$ gates, and the five single-qutrit X permutation gates: $X = X_{+1} = X_{012}$, $X^\dagger = X_{-1} = X_{021}$, X_{01}, X_{02}, and $X_{12} = -H^2$. We also defined the notions of Λ, $|0\rangle$, $|1\rangle$, and $|2\rangle$-controlled gates. In Equation (8.24) we showed a T-count 3 decomposition for the $|0\rangle$-controlled Z gate from [BRS17, Figure 6], and in Lemma 8.25 we showed a T-count 15 decomposition of the $|2\rangle$-controlled X_{01}, X_{02}, and X_{12} gates adapted from Ref. [BCRS16].

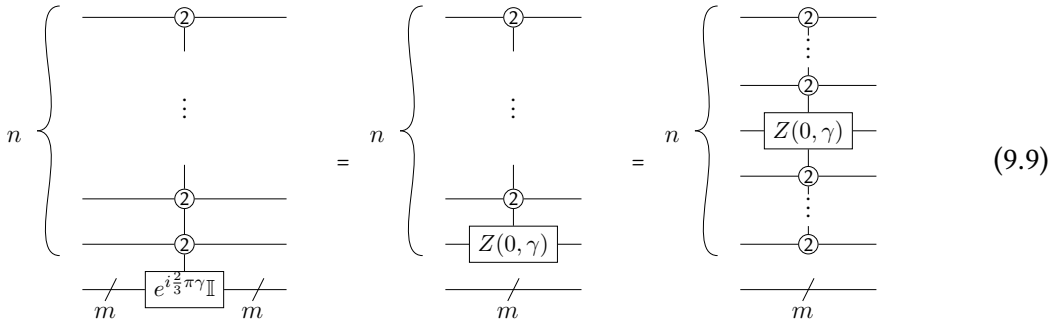
Note that the Λ notion of control is reconstructable from the singly-controlled (i.e. $|0\rangle$, $|1\rangle$, and $|2\rangle$-controlled) one: just apply a $|1\rangle$-controlled U, followed by a $|2\rangle$-controlled U^2. Adding controls to a Clifford gate generally makes it non-Clifford. In the case of the CX gate, which is $\Lambda(X_{+1})$, it is still Clifford, while the $|2\rangle$-controlled X_{+1} gate is *not* Clifford. This contrasts to the qubit case, as there are no non-Clifford two-qubit binary classical reversible gates.

Conjugating the two aforementioned non-Clifford two-qutrit gates by the appropriate single-qutrit Clifford gates, suffices to construct any singly-controlled permutation X or Z gate. By conjugating any $|0\rangle$-control qutrit by either $X^\dagger$ before and X after, or X before and $X^\dagger$ after, the $|1\rangle$- and $|2\rangle$-controlled versions are respectively obtained. As the control qutrit is always diagonal in the Z basis (which is self-adjoint), taking the adjoint of any controlled circuit decomposition, has the effect of changing the target operation to its adjoint. The target of any controlled Z gate can be changed to X or to $X^\dagger$, by conjugating by a H and $H^\dagger$ pair.

Note that we usually don't care about global phases in the definition of unitaries. However, when adding controls, the global phase becomes 'local' and hence is relevant [BBC+95, Lemma 5.2].

Definition 9.8. A *controlled global phase gate* is a controlled unitary where the unitary is $e^{i\phi}\mathbb{I}$, for an identity matrix I and some phase ϕ.

The number of qutrits the identity matrix acts on in this definition is irrelevant as the phase factor can be "factored out" from the tensor product of the controlled and target qutrits:

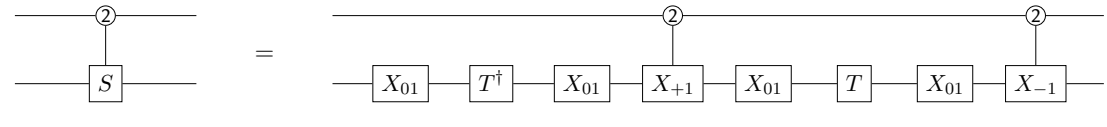


(9.9)

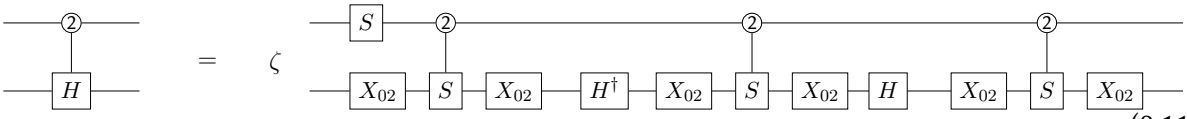
Here we wrote the global phase ϕ as $\phi = \gamma \cdot 2\pi/3$ so that we can represent the phase gate as a multiple of ω.

We will see that it can be easier or more cost effective to construct a controlled unitary 'up to a controlled global phase', and that to implement the unitary exactly, additional work must be done. A generalization of this idea was used for qubit controlled gates in Ref. [Mas16] to find more efficient decompositions of controlled gates.

In previous work [GRvdWY22a] we also found a construction of the $|2\rangle$-controlled S gate and the $|2\rangle$-controlled Hadamard gate:



(9.10)



(9.11)

Note that the circuit for the controlled-S gate that Equation (9.10) is based on in Ref. [GRvdWY22a] is only correct up to a controlled global phase. In this chapter we defined our S gate to include this global phase, so that it doesn't appear here. Due to this redefinition, the circuit Equation (9.11) acquires a (non-controlled) global phase of ζ. This is a problem if we wish to use the gates in this circuit as a base for adding more controls, as this global phase then becomes a *controlled* global phase. We will see later that there are however ways around this.

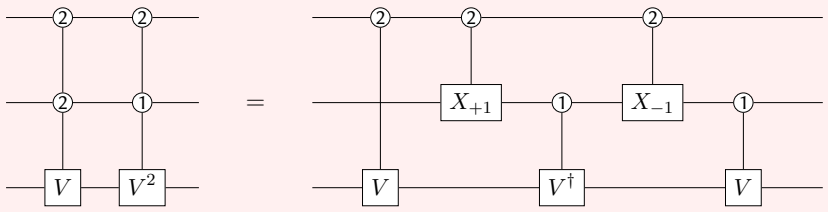
9.2 Adding controls to Clifford+T gates

In this section we will implement each the $|2\rangle^{\otimes k}$-controlled versions of every qutrit Clifford+T unitary. As Clifford+T unitaries are built out of Hadamard, CX, S and T gates, it suffices to show how we can construct k-controlled versions of each of these gates.

We will do this in stages, first showing how to construct the X permutation gates with two controls, and then any number of controls, before moving on to the other Clifford+T gates.

Before we can make the step to having an arbitrary number of controls, we first need to construct the permutation gates with two controls. First, we construct the $|22\rangle$-controlled X_{01} gate. To do this we use the following lemma that allows us to add more controls to a unitary, once we know how to construct it with just one control.

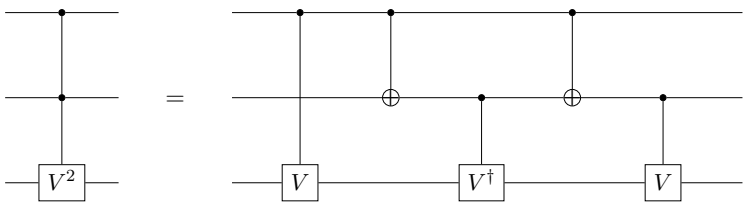
Lemma 9.12. *For any qutrit unitary V with a construction for $|2\rangle$-controlled V and $|2\rangle$-controlled $V^\dagger$, we can build the circuit consisting of the $|22\rangle$-controlled V multiplied by the $|21\rangle$-controlled V^2 unitary:*



(9.13)

Corollary 9.14. *If we pick $V = X_{01}$, then we have $V^2 = \mathbb{I}$, so that this construction gives us a way to construct the $|22\rangle$-controlled X_{01} using just $|2\rangle$-controlled X_{01} and X_{+1} and X_{-1} gates, which we already know how to construct in Clifford+T.*

Lemma 9.12 is easily shown to be correct by doing case distinctions on the control wires. While we believe this construction to be new, it is based on the qubit Sleator-Weinfurter decomposition [BBC+95, Lemma 6.1]:



(9.15)

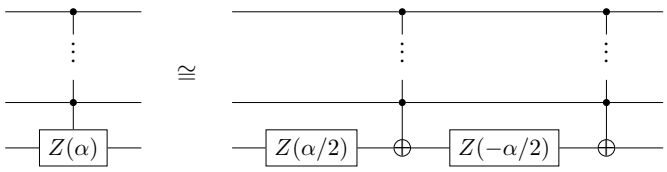
We note that Corollary 9.14 contradicts the statement of Ref. [Mor14] that "a 5-gates Barenco et al. type of realization of a ternary (generalized) Toffoli gate without adding an ancillary line is simply not possible"; their analysis of ternary generalizations of the Sleator-Weinfurter decomposition did not account for decompositions of the form of Equation (9.13).

Lemma 9.16. *We can construct the $|22\rangle$-controlled X_{01} gate in Clifford+T without using any ancilla.*

Additionally, by conjugating by the appropriate single-qutrit Cliffords on the bottom qutrit, we can also construct the $|22\rangle$-controlled $X_{12,}$ and X_{02} gates.

Remark 9.17. The $|22\rangle$-controlled X_{01} gate was decomposed by Di and Wei [DW13] in terms of qubit Clifford+T operations, where the CX gate could be performed pairwise on any two of the three qutrit Z-basis states. As far as we are aware, there does not exist an error correction protocol that can, without code switching, correct qubit Clifford operations on all three pairs of Z-basis states. The construction above is the first qutrit Clifford+T, and therefore suitable for fault-tolerant computation, decomposition of the $|22\rangle$-controlled X_{01} gate.

We can use this gate in turn to build the $|22\rangle$-controlled X_{+1} gate. To do this we will adapt a well-known construction for qubit controlled phases. Recall that if we can implement k-controlled Toffoli gates and the square root of a phase gate, that we can then also implement the k-controlled phase gate [BBC+95]:

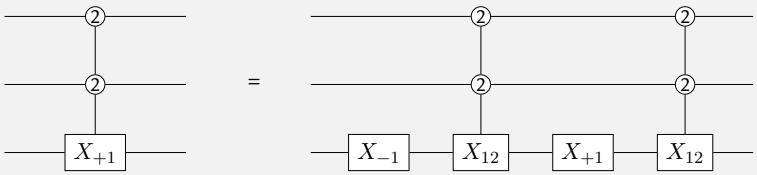


(9.18)

This works because when the Toffoli 'fires' the X gate is applied, and we have $XR_Z(\alpha)X \propto Z(-\alpha)$.

Lemma 9.19. *We can construct the $|22\rangle$-controlled X_{+1} gate in Clifford+T without using any ancillae.*

Proof. Consider the following circuit:



(9.20)

This works because $X_{12}X_{+1}X_{12} = X_{-1}$ and $X_{-1}^2 = X_{+1}$. □

Corollary 9.21. *By conjugating by appropriate Clifford gates, we can use the $|22\rangle$-controlled X_{+1} and X_{01} gates to construct any $|xy\rangle$-controlled permutation gate where $x, y \in \{0, 1, 2\}$ of*

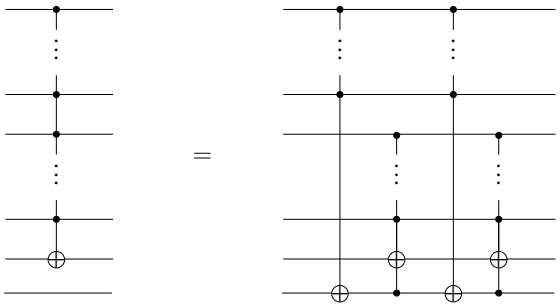
our choosing.

Now let's see how we can generalize these constructions to have any number of controls. Here instead of building the controlled X_{+1} gate out of the controlled X_{01}, we will go in the opposite direction. In order to efficiently build the $|2\rangle^{\otimes k}$-controlled X_{+1} gate, we will adapt a construction for multiple-controlled Toffolis for qubits that requires one *borrowed ancilla.*

Definition 9.22. A borrowed ancilla is an ancilla that can be in any state, and that is returned to the same state after the operation is finished.

Having a borrowed ancilla just means that in the circuit we are considering there is at least one other qutrit that is not directly involved with our construction.

We base our construction on the following qubit identity for Toffolis [BBC+95, Lemma 7.3]:

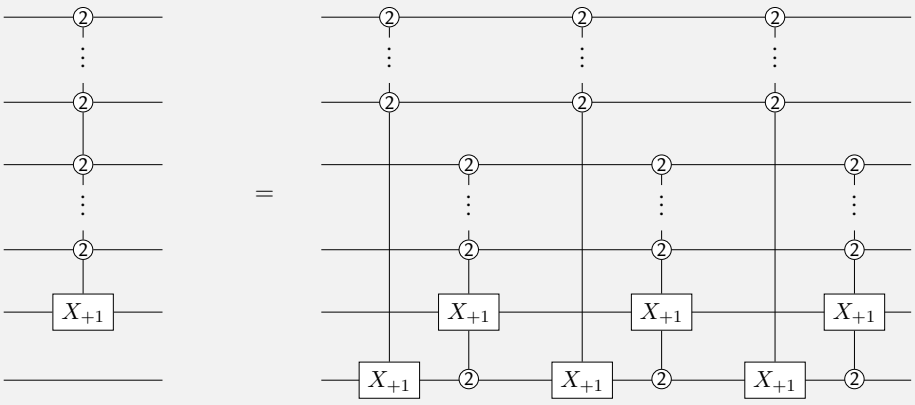


(9.23)

Using this construction, a k-controlled Toffoli where $k = 2^m$ can be decomposed into 4 Toffoli's with $k/2 = 2^{m-1}$ controls. Iterating this procedure then requires $O(k^2)$ 'standard' Toffolis with 2 controls each.

Lemma 9.24. *We can construct the $|2\rangle^{\otimes k}$-controlled X_{+1} gate using $O(k^{2.585})$ Clifford+T gates and using one borrowed ancilla.*

Proof. Consider the following identity:



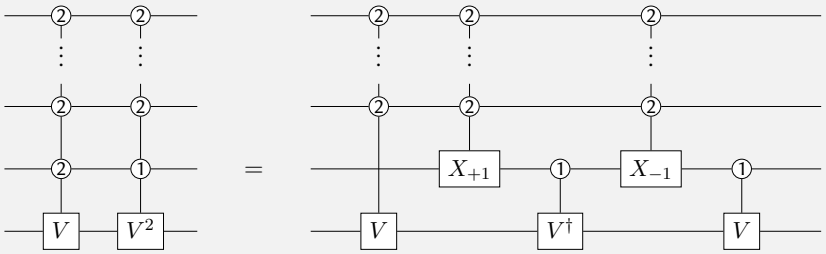
(9.25)

To see this is correct first note that it is the identity when any of the top control wires is not in the $|2\rangle$ state (as doing X_{+1} three times is just the identity). As such, let us assume they are all in the $|2\rangle$ state so that we can ignore these control wires. We are cycling the value of the ancilla three times, which means that the controlled X_{+1} gate on the target qutrit fires exactly once, when the ancilla is put into the $|2\rangle$ state. As we cycle it three times, the ancilla is put into the same state that it started in.

We can use this to reduce the implementation of a $|2\rangle^{\otimes k}$-controlled X_{+1} gate where $k = 2^m$ to a sequence of 6 $|2\rangle^{\otimes k/2}$-controlled X_{+1} gates. Letting $C(m)$ denote the number of $|22\rangle$-controlled X_{+1} gates needed to write the 2^m-controlled X_{+1} gate, we then get the relation $C(m) = 6C(m-1)$. As $C(1) = 1$ we calculate $C(m) = 6^{m-1} = \frac{1}{6}2^{m \log_2 6}$. Substituting $k = 2^m$ we then see we require $O(k^{\log_2 6})$ of the base gate, which we can round up to $O(k^{2.585})$. □

Lemma 9.26. *We can construct the $|2\rangle^{\otimes k}$-controlled X_{01} gate using $O(k^{3.585})$ Clifford+T gates without using any ancillae.*

Proof. Consider the following generalized version of Equation (9.13), for which the qubit analogue is [BBC+95, Lemma 7.5]:



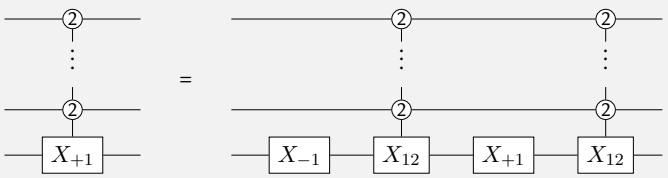
(9.27)

Again taking $V = X_{01}$, we see that we can reduce the construction of the $|2\rangle^{\otimes k}$-controlled X_{01} gate to the construction of the $|2\rangle^{\otimes k-1}$-controlled X_{01} at the cost of introducing two $|2\rangle^{\otimes k-1}$-controlled X_{+1} gates and two $|2\rangle$-controlled X_{01} gates. Iterating the procedure we see then that the full construction requires $2k$ $|2\rangle$-controlled X_{01} gates and $2k$ $|2\rangle^{\otimes m}$-controlled X_{+1} gates where $m \leq k-1$. The cost of this is dominated by the controlled X_{+1} gates, which cost $O(k^{2.585})$ gates each. As we have $O(k)$ of these, we require $O(k^{3.585})$ Clifford+T gates to build the $|2\rangle^{\otimes k}$-controlled X_{01} gate. Note that as the controlled X_{+1} gates are not controlled on all the wires that we have access to at least one borrowed ancilla, so that we can in fact use the construction of Lemma 9.24. □

This construction doesn't require any borrowed ancillae. We can now use a generalized version of Equation (9.20) to complete the circle and construct a many-controlled version of the X_{+1} gate that does not require any borrowed ancillae (at the cost of worse polynomial scaling).

Lemma 9.28. *We can construct the $|2\rangle^{\otimes k}$-controlled X_{+1} gate using $O(k^{3.585})$ Clifford+T gates without using any ancillae.*

Proof. Follows from the following circuit that generalizes Equation (9.20):



(9.29)

As this requires two copies of $|2\rangle^{\otimes k}$-controlled X_{01}, the cost is asymptotically the same as in Lemma 9.26. □

Note that we can change what the unitary is controlled on by conjugating any number of control qutrits by the appropriate qutrit X gate in $\{X_{+1}, X_{-1}, X_{01}, X_{02}, X_{12}\}$. Successive application of tritstring-controlled X gates leads to selective control on all possible combinations of tritstrings. We hence have the following.

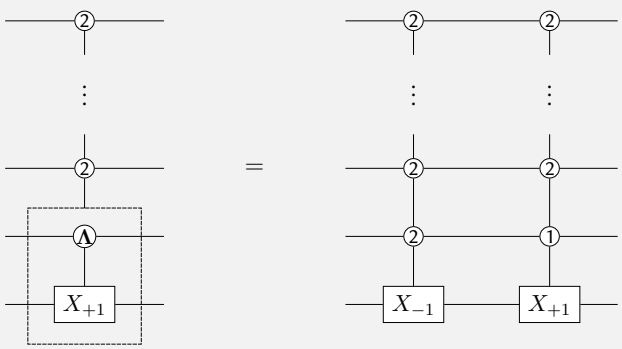
Corollary 9.30. *We can construct any generalization of a multiple-controlled Toffoli to the qutrit setting in Clifford+T unitarily without ancillae.*

9.3 Building multiple-controlled Clifford+T gates

Now we have all the tools to build the remaining controlled Clifford+T gates: CX, Hadamard, S and T.

Lemma 9.31. *The $|2\rangle^{\otimes k}$-controlled CX gate can be constructed unitarily without ancillae using a polynomial number of Clifford+T gates in k.*

Proof. Just consider the following construction, which can be obtained from applying Lemma 9.12 with $V = X_{-1}$:



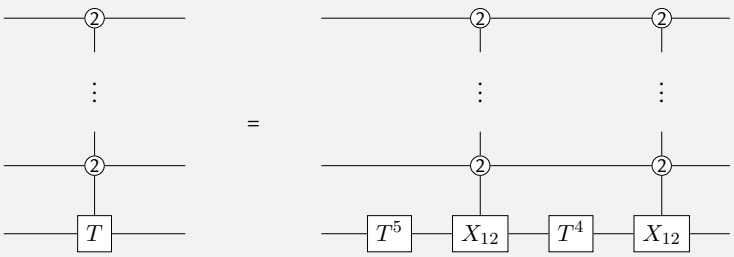
(9.32)

□

For the many-controlled T gate we use the 'square-root trick' of Equation (9.18). Here to find the square root we can use the fact $(T^5)^2 = T$ as $T^9 = \mathbb{I}$, and hence T^5 acts like a $\sqrt{T}$ gate. Similarly, T^4 is like $\sqrt{T}^\dagger$. It is then easily verified that we have the following construction for the controlled T gate.

Lemma 9.33. *We can build the $|2\rangle^{\otimes k}$-controlled T gate unitarily without ancillae using a polynomial number of Clifford+T gates in k.*

Proof. Consider the following circuit identity:



(9.34)

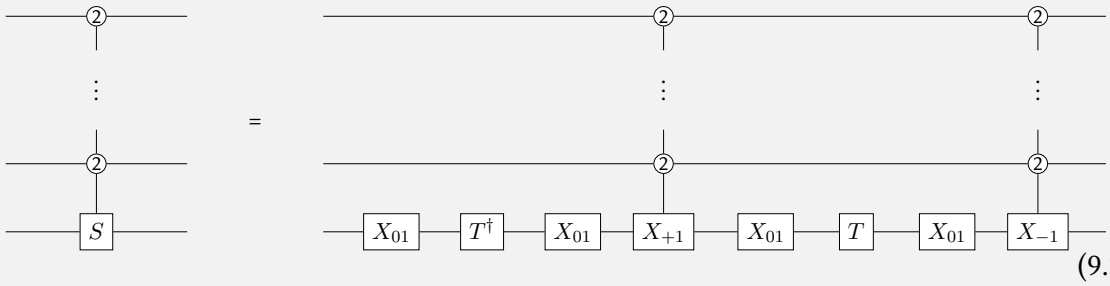
Its correctness follows because $X_{12}T^4X_{12} = T^5$. □

That we can implement the $|2\rangle$-controlled T in Clifford+T without ancillae is interesting as this is not possible in the qubit setting. With qubits, to construct the controlled T gate we have to either employ the $\sqrt{T}$ gate, or a clean ancilla [BBC+95; Kit96; AMMR13; KMM13; Sel13; KC18]. Indeed, Kliuchnikov, Maslov, and Mosca showed that for qubit Clifford+T a single ancilla is in fact necessary [KMM13]. This constructions generalizes odd qudit dimension 3.

To add controls to the S gate, we extend our construction from Ref. [GRvdWY22a]:

Lemma 9.35. *The $|2\rangle^{\otimes k}$-controlled S gate can be constructed unitarily without ancillae using a polynomial number of Clifford+T gates in k.*

Proof. We straightforwardly generalize Equation (9.10):



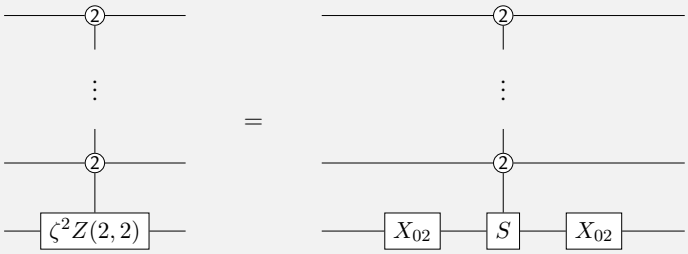
(9.36)

□

We will shortly show how to control the H gate. To do this we will need to handle some controlled global phase for which we need the following results.

Lemma 9.37. *The $|2\rangle^{\otimes k}$-controlled $Z(2,2) = diag(1,\omega^2,\omega^2)$ gate can be constructed up to a controlled global phase of ζ^2. This Clifford+T construction is unitary and ancilla-free, with T-count polynomial in k.*

Proof. Use the following circuit:



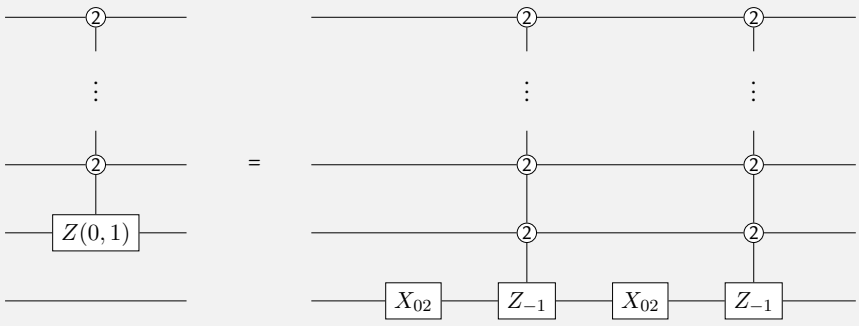
(9.38)

Its correctness can be verified by direct computation, or by commuting S and X_{02}. □

With one borrowed ancilla, we can construct the $|2\rangle^{\otimes k}$-controlled $\zeta S = Z(0,1)$ gate:

Lemma 9.39. *The $|2\rangle^{\otimes k}$-controlled $Z(0,1)$ gate can be constructed unitarily with one borrowed ancilla using only Clifford+T gates, with T-count polynomial in k.*

Proof. Use the following circuit:



(9.40)

The correctness follows from the fact that $Z_{-1}X_{02}Z_{-1}X_{02} = \omega\mathbb{I}$, and from Equation (9.9). □

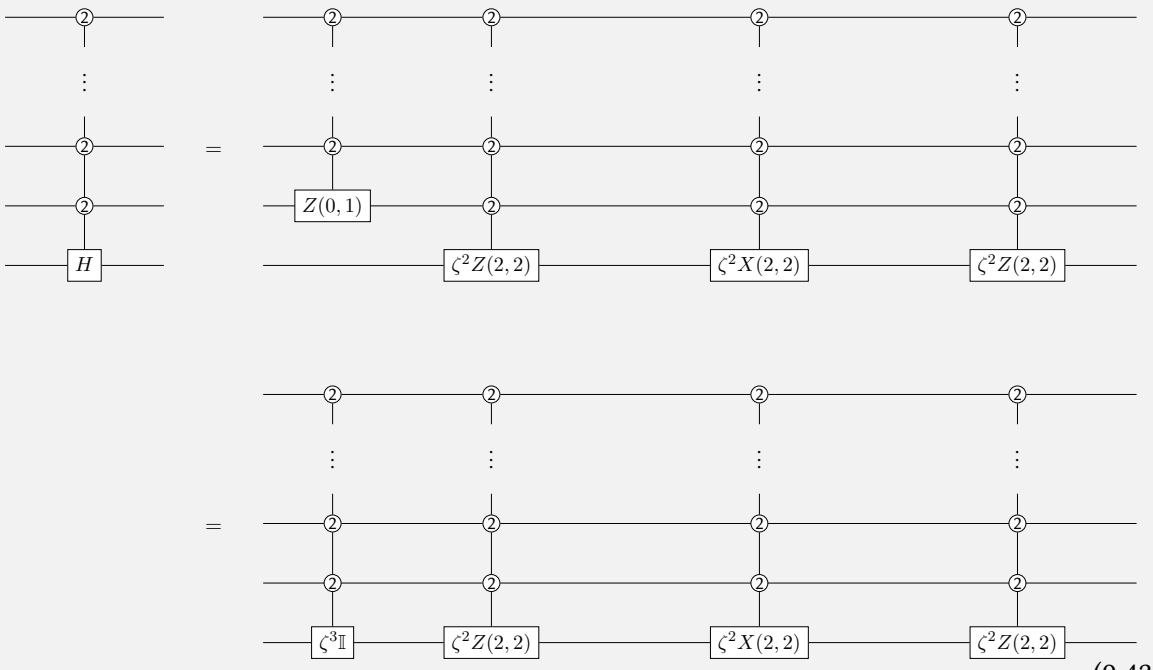
This construction of the $|2\rangle^{\otimes k}$-controlled ζS gate can be composed with a $|2\rangle^{\otimes k}$-controlled $S^\dagger$ gate, which we constructed in Lemma 9.35. The resulting unitary is then the controlled global phase of ζ gate;

Corollary 9.41. *We can construct the $|2\rangle^{\otimes k}$-controlled $\zeta\mathbb{I}$ gate unitarily without ancillae in Clifford+T. Equivalently, this gate is equal to the $|2\rangle^{\otimes(k-1)}$-controlled $Z(0,1/3)$ gate by*

Equation (9.9).

Lemma 9.42. *The $|2\rangle^{\otimes k}$-controlled Hadamard gate can be implemented unitarily without ancillae using a polynomial number of Clifford+T gates in k.*

Proof. Consider the following circuit:



(9.43)

Each of these gates can be constructed unitarily in Clifford+T without ancillaes and with a polynomial number of gates using Lemmas 9.37 and 9.39. To see why it is correct, note that when the gates fire it implements a $\zeta^9 Z(2,2)X(2,2)Z(2,2)$ gate, and as $\zeta^9 = 1$ and $Z(2,2)X(2,2)Z(2,2) = H$ by Equation (8.13) the construction is indeed correct. □

This leads to the main theorem of this work. We have now seen how to exactly construct unitarily and without ancillae the $|2\rangle^{\otimes k}$-controlled CX, S, H, T and permutation gates. Each of our constructions used at most $O(k^{3.585})$ Clifford+T gates.

Theorem 9.44. *Let U be any n-qutrit Clifford+T unitary consisting of N CX, S, H and T gates. Then there is a $n+k$-qutrit Clifford+T circuit implementing the $|2\rangle^{\otimes k}$-controlled U unitary using $O(Nk^{3.585})$ gates.*

As noted in Corollaries 9.21 and 9.30, by conjugating the control qutrits by certain Clifford operations, we can make the unitary controlled on every possible tritstring in the Z- or X-basis. Furthermore, by repeated application of the decomposition on different control values, more specific definitions of ternary control, including that of Def. 5.7 may be realized.

At this point in the paper [YvdW22], we then presented a synthesis algorithm for implementing any reversible classical trit function $f\colon \{0,1,2\}^n \to \{0,1,2\}^n$ as a unitary n-qutrit Clifford+T circuit using at most $O(3^n n^{3.585})$ Clifford+T gates. This factor of $O(n^{3.585})$ is the cost of constructing n-qutrit controlled Toffoli gates, which we subsequently improved to $O(n)$ in Lemma 12.20 which we present in the next chapter [RvdWY23].

We also proved a lower bound for this problem of synthesizing any reversible classical trit function to be $\Omega(3^n \cdot n/\log n)$. We will state this result in the next chapter instead, to prove it more generally for any qudit dimension. Comparing this lower bound to our synthesis algorithm which is $O(3^n n)$ with the asymptotically improved Toffoli decomposition, this means that our algorithm is within a logarithmic factor of the minimum gate count required.

10

Building diagonal gates with phase gadgets

Remark 10.1. This section is based on the paper [vdWY23], in collaboration with John van de Wetering.

An application of qutrits that has been a subject of much interest and potential is *emulation*: where qubit computation is emulated inside a subspace of the qudits to enable more resource-efficient gate implementations. In contrast, it has been argued that qubits cannot simulate qudit (where $d > 2$ and d is not a power of 2) computation efficiently [BOB05]. Much work on qutrits and emulation has focused on *classical* functions, which map tritstrings to tritstrings. As an example, using qutrits we can build logarithmic-depth Toffolis [GBD+19; NKF22] and binary AND gates on superconducting qutrits [CHZ+23]. Of much less study has been whether there are advantages to emulation using more inherently quantum gates such as diagonal unitaries.

For qubits a useful tool for understanding diagonal unitaries has been the concept of a *phase gadget* [KvdW20b]. This is a type of symmetric multi-qubit interaction that occurs naturally in many hardware architectures [PDF+21; PRE+22; SMCG16], and serves as a good basis for optimising quantum circuits [CDD+20; CSD20; dBBW20c; dBBW20b; vdWet21; BMdF+21]. Any diagonal qubit unitary can be expressed as a product of phase gadgets by writing the unitary as a *phase polynomial* [Amy19; MD23].

In this chapter we study the generalization of phase gadgets to the qutrit setting. We do this by adapting the qutrit ZX-calculus of Refs. [GW17; Wan18a] and transforming it into a *flexsymmetric* calculus [Car21] where the spiders have more desirable symmetry properties. We find this calculus has a simple set of rules for the Clifford fragment. We define phase gadgets analogously to the qubit case, meaning that as diagrams they look nearly identical. There are however significant differences between the qubit and qutrit gadgets. We will show that we can nevertheless use qutrit phase gadgets to construct some useful qutrit diagonal unitaries, such as controlled phase gates, and a type of gate we dub a *phase multiplier*[1]. This last one is possible by generalizing the formula that leads to the *graphical Fourier theory* for qubit diagonal unitaries [KvdWK19].

As an application of our results we show how we can emulate an arbitrary qubit diagonal unitary using qutrit phase gadgets. This leads us to a construction of the emulated qubit CCZ gate that requires only three non-Clifford qutrit R gates [GRvdWY22a]. This is surprising because using just qubits, we would require at least four T gates to implement the CCZ [HC17].

10.1 The flexsymmetric qutrit ZX-calculus

First, we must once again re-define the qutrit Clifford gate global phases. This time, the reason is to match conventions of the qutrit ZX-calculus conceived prior to this work.

Definition 10.2. The qutrit S gate is $S \coloneqq \mathrm{diag}(1, 1, \omega)$. I.e. it multiplies the $|2\rangle$ state by the phase ω.

The X basis consists of the following states:

$$\begin{aligned} |+\rangle &\coloneqq \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle) \\ |\omega\rangle &\coloneqq \frac{1}{\sqrt{3}}(|0\rangle + \omega\,|1\rangle + \bar{\omega}\,|2\rangle) \\ |\bar{\omega}\rangle &\coloneqq \frac{1}{\sqrt{3}}(|0\rangle + \bar{\omega}\,|1\rangle + \omega\,|2\rangle) \end{aligned}$$

Definition 10.3. The *qutrit Hadamard gate* H is the unitary mapping $|0\rangle \mapsto |+\rangle$, $|1\rangle \mapsto |\omega\rangle$ and $|2\rangle \mapsto |\bar{\omega}\rangle$.

$$H \coloneqq \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \bar{\omega} \\ 1 & \bar{\omega} & \omega \end{pmatrix} \tag{10.4}$$

[1] Not to be confused with multipliers, which are a different defined map in graphical calculi.

This implies that $H^2 = X_{12}$, $H^4 = \mathbb{I}$, and $H^\dagger = H^3$.

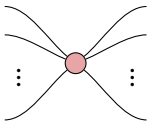
All other qutrit gates are defined consistently in this section thus far.

A helpful observation is an equivalent notation for the R gate:

Observation 10.5. The qutrit *reflection* gate is defined as $R \coloneqq Z(0, 3/2) = \text{diag}(1, 1, -1)$.

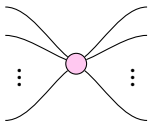
While quite similar to the qubit one, it loses some of the properties that make the original easy to work with. In particular, for each X-spider, the distinction between its input wires and output wires becomes important. This means we can no longer treat qutrit ZX-diagrams as undirected graphs with the spiders as vertices. This makes intuitive reasoning about these diagrams harder, and also complicates the implementation of software for dealing with these diagrams.

Here we will present a variation on the qutrit ZX-calculus of Refs. [WB14; GW17; Wan18a] where the spiders do enjoy this additional symmetry between inputs and outputs. The way we do this is by redefining the X-spider. In the original qutrit ZX-calculus we have



$$\propto \sum_{\substack{\vec{x},\vec{y} \\ x_1+\cdots+x_n=y_1+\cdots+y_m}} |\vec{y}\rangle\langle\vec{x}| \,. \tag{10.6}$$

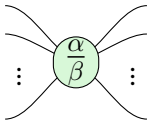
Here the sum $x_1 + \cdots + x_n = y_1 + \cdots + y_m$ is taken modulo 3. If we put a cup on one of the wires to turn an output into an input, then this has the effect of introducing a minus sign on that variable, changing for instance $x_1 + x_2 = y_1 + y_2$ into $x_1 + x_2 - y_2 = y_1$. For qubits this is not a problem since $-x = x$ modulo 2, but for qutrits this changes the map. We fix this by defining a new X-spider as



$$\propto \sum_{\substack{\vec{x},\vec{y} \\ x_1+\cdots+x_n+y_1+\cdots+y_m=0}} |\vec{y}\rangle\langle\vec{x}| \,. \tag{10.7}$$

We see that in this definition the inputs and outputs are treated on equal footing. In order to prevent confusion with earlier work, we will denote this new X-spider in pink, instead of in red[2].

Let's now give the full definition of the spiders. Following the convention first used in Ref. [WB14], we define the Z-spider as

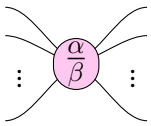


$$= |0\cdots0\rangle\langle0\cdots0| + \omega^\alpha |1\cdots1\rangle\langle1\cdots1| + \omega^\beta |2\cdots2\rangle\langle2\cdots2| \,.$$

[2]We have checked the accessibility of this color scheme; in fact, a red-green colorblind person greatly preferred this pink to the default ZX red.

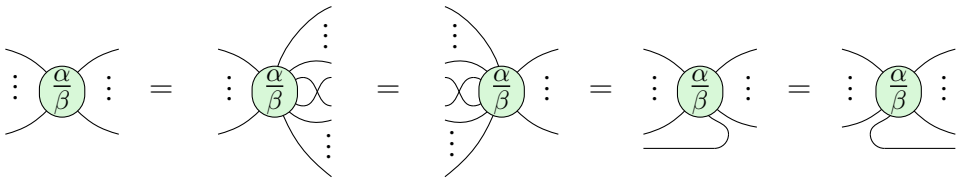
Here we have two phase angles α and β, as opposed to just the one angle in qubit ZX. Generalizing this to d-dimensional spiders, you would need to specify $d-1$ phase angles. In particular, when written in a spider $\frac{\alpha}{\beta}$ should be interpreted as two different phases and not as the fraction α/β. The one-input, one-output Z spider with arbitrary α and β is the $Z(\alpha,\beta)$ single-qutrit phase gate of Definition 8.4. Note that we define the phase angles as ω^α and ω^β so that these correspond to the complex phases $e^{i\frac{2\pi}{3}\alpha}$ and $e^{i\frac{2\pi}{3}\beta}$. This means that α and β are integers iff the spider is in the Clifford fragment of the calculus.

We define the X-spider similarly, but with respect to the X-basis:

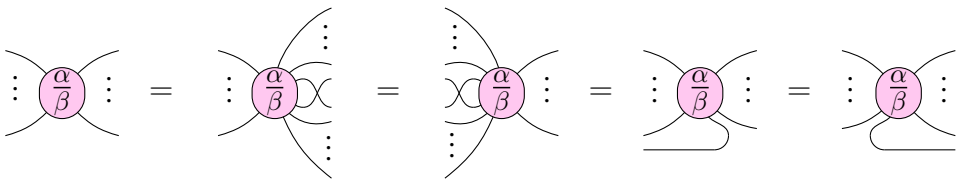


$$= |+\cdots+\rangle\langle+\cdots+| \;+\; \omega^\alpha\,|\bar{\omega}\cdots\bar{\omega}\rangle\langle\omega\cdots\omega| \;+\; \omega^\beta\,|\omega\cdots\omega\rangle\langle\bar{\omega}\cdots\bar{\omega}|$$

This requires some explanation, because this does not look symmetric in the inputs and outputs. However, note that $\langle\omega| = (|\omega\rangle)^\dagger = (|0\rangle + \omega\,|1\rangle + \omega\,|2\rangle)^\dagger = \langle 0| + \bar{\omega}\,\langle 1| + \omega\,\langle 2|$. Hence, if we take the transpose of $|\omega\rangle$ we actually get $\langle\bar{\omega}|$. It is straightforward to check that with $\alpha = \beta = 0$ we get back Equation (10.7). These definitions of the Z-spider and X-spider satisfy the symmetry properties we want, namely:



(10.8)



(10.9)

(10.10)

These symmetries mean our spiders are *flexsymmetric*, as defined by Carette [Car21], and as a result we may treat our ZX-diagrams as undirected graphs with the spiders as vertices. Note that here the cups and caps are defined with respect to the Z basis: $\subset = |00\rangle + |11\rangle + |22\rangle$. As usual, our calculus also formally has generators for the identity wire and the swap.

It will be useful to introduce an additional graphical generator for the Hadamard:



$$= |+\rangle\langle 0| + |\omega\rangle\langle 1| + |\bar{\omega}\rangle\langle 2| = |0\rangle\langle +| + |1\rangle\langle\bar{\omega}| + |2\rangle\langle\omega|\,. \tag{10.11}$$

We write the Hadamard as a slanted box, because it is self-transpose, but not self-adjoint, and thus is most accurately denoted in a way that is symmetric under a rotation, but not a reflection.

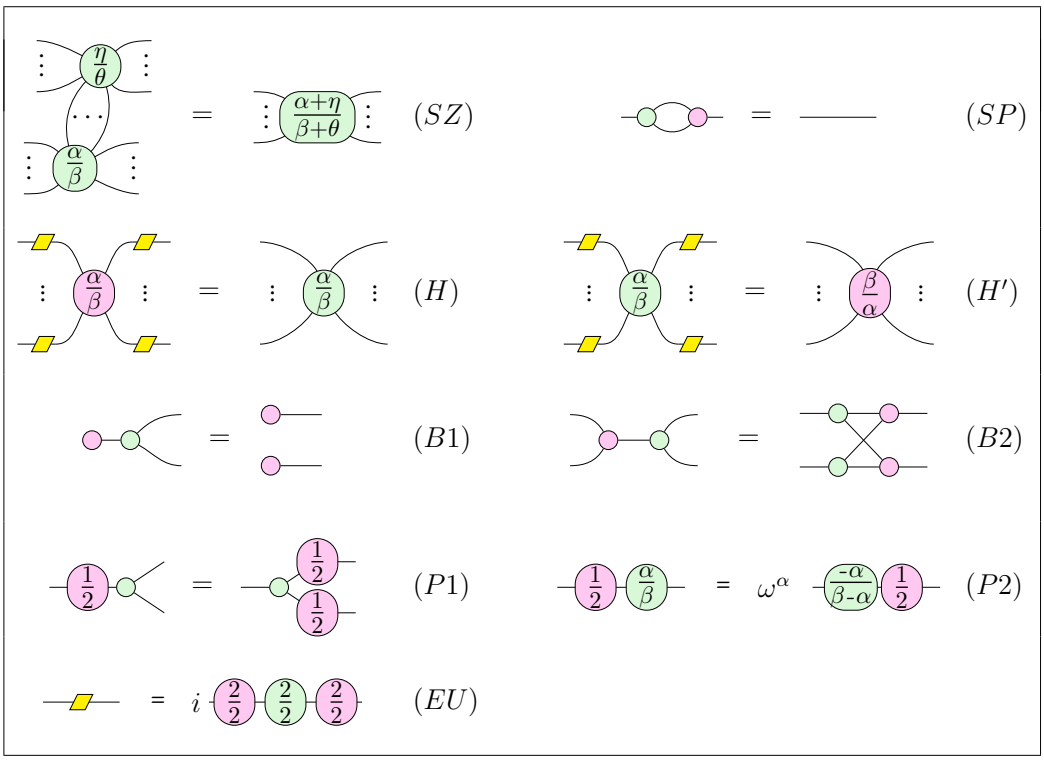
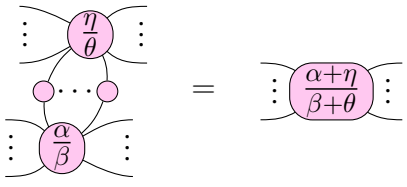


Figure 10.1: Rules for the flexsymmetric qutrit ZX-calculus. These hold for all $\alpha, \beta, \eta, \theta \in \mathbb{R}$, and for any permutation of the input and output wires. Additional useful derived rules are presented in Figure 10.2. The letters stand respectively for (S)pider-Z, (H)adamard, (B)ialgebra, (SP)ecial, (P)auli, and (EU)ler decomposition.

Our redefinition of the X-spider comes at a 'cost'. Namely, the 1-input, 1-output X-spider is no longer the identity: $\text{—○—} = |0\rangle\langle 0| + |2\rangle\langle 1| + |1\rangle\langle 2| = |+\rangle\langle +| + |\bar{\omega}\rangle\langle \omega| + |\omega\rangle\langle \bar{\omega}|$. This map is implementing $|x\rangle \mapsto |-x\rangle$ where $-x$ is taken modulo 3, and is equal to X_{12}. Additionally, the X-spider is not really a spider any more in the sense that it doesn't satisfy the standard spider-fusion equation. Instead it satisfies the 'harvestman equation' [Car21] that also holds for, for instance, the W-spider [Had15] and H-box [BK19]:



In Figure 10.1, we present a set of rewrite rules for this qutrit ZX-calculus. We have accounted for the global phase for each rule here as a complex number, as those will be relevant to us. Note however that the rewrite rules are not scalar-accurate as we are ignoring factors of $\sqrt{3}$.

Using these rules, other useful qutrit ZX-calculus rewrite rules may be derived. In particular, we can use these rules to prove the derived rules presented in Figure 10.2. As these rules are (a

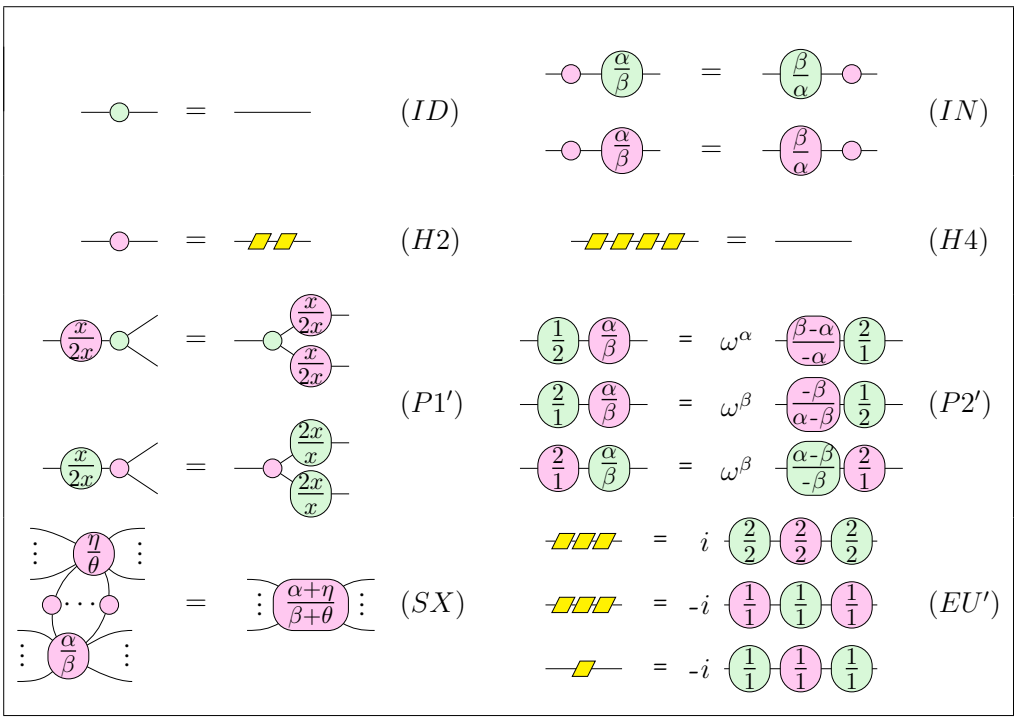


Figure 10.2: These rules are derivable from the rules of Figure 10.1 for any $\alpha, \beta, \eta, \theta \in \mathbb{R}$ and $x \in \{0, 1, 2\}$. The new letters stand respectively for (ID)entity, (IN)vert and (S)pider-X.

slight variation) on the non-flexsymmetric qutrit rules of Ref. [Wan18a], our calculus is also complete for the qutrit Clifford fragment (when ignoring non-zero scalars). The proofs of the derived rules of Figure 10.2, and explanation of why most rules of Figure 10.1 are necessary (i.e. not derivable from the others), can be found in the appendix of the paper [vdWY23].

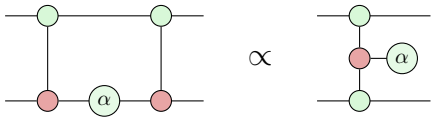
We see in these rules that there is a special role for phases of the form $\frac{x}{2x}$ where $x \in \{0, 1, 2\}$. This is because $\frac{x}{2x}$ $\propto |x\rangle$ and $\frac{x}{2x}$ $= Z^x$. These relations can be derived by using the identity $1 + \omega + \bar{\omega} = 0$ together with $\omega^2 = \omega^{-1} = \bar{\omega}$. In general we will see a lot of $\frac{\alpha}{2\alpha}$ phases because they implement the $|x\rangle \mapsto \omega^{\alpha x} |x\rangle$ phase gate. Additionally, note that the $(P2)$ rule on the qubit subspace is exactly the familiar qubit ZX rule π α $= e^{i\alpha}$ $-\alpha$ π, since the red π is the qubit Pauli X while the pink $\frac{1}{2}$ phase is the qutrit Pauli X.

10.2 Constructing phase gadgets, controlled phase gates, and phase multipliers

For qubits the concept of a *phase gadget* has proven very useful. There's several different ways we can define a qubit phase gadget. One way is to consider it as the diagonal gate $|x, y\rangle \mapsto e^{i\alpha(x \oplus y)} |x, y\rangle$ (for simplicity we are only considering a two-qubit phase gadget). This

applies a phase of $e^{i\alpha}$ when $x \oplus y = 1$. Here $\oplus$ is the XOR operation, which is the addition on $\mathbb{Z}_2$. This suggests that we should define the qutrit phase gadget as $|x, y\rangle \mapsto e^{i\alpha(x+y)} |x, y\rangle$ where now we take $x + y$ to be modulo 3.

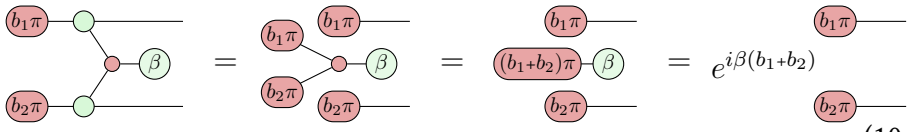
We could also define a phase gadget by its circuit realization or diagrammatic representation. For qubits [KvdW20b]:



(10.12)

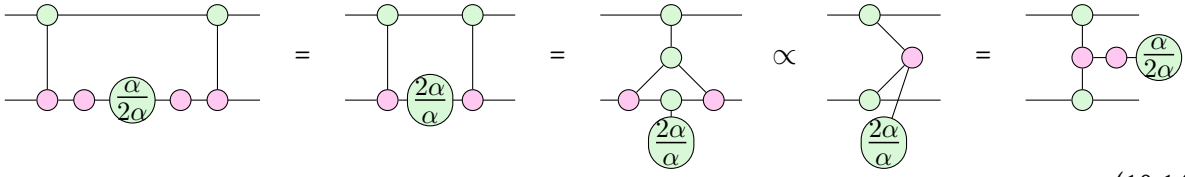
As a two-qubit diagonal gate, this has the action on computational basis states of:

$e^{i\beta Z\otimes Z}|b_1 b_2\rangle$ =



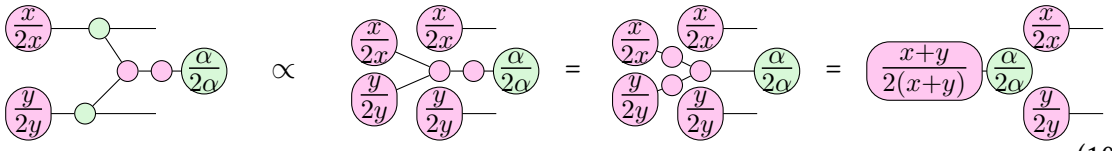
(10.13)

We claim the qutrit variant of this construction is given by the following circuit which can be simplified to a similar diagrammatic representation:



(10.14)

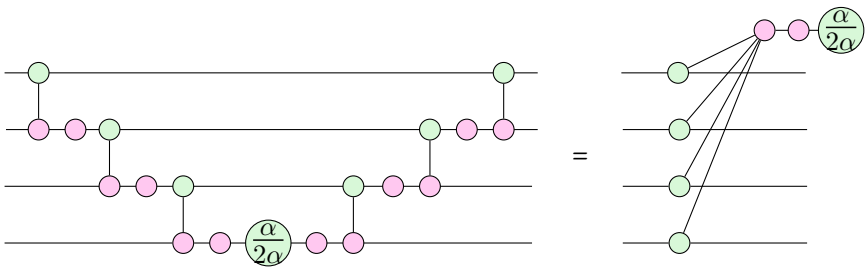
Indeed, inputting $|x, y\rangle$ into this diagram allows us to calculate its action:



(10.15)

This 'floating scalar' expression evaluates to $\sqrt{3}\omega^{\alpha(x+y \bmod 3)}$, so that this diagram indeed implements the operation we want, and we see that these three ways to define a qubit phase gadget—via the action, via the circuit, or via the diagrammatic representation—are also equal for qutrits.

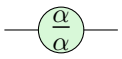
We can easily generalize this construction to an arbitrary number of qutrits:



$$:: \ |x,y,z,w\rangle \ \mapsto \ \omega^{\alpha(x+y+z+w \bmod 3)} |x,y,z,w\rangle \tag{10.16}$$

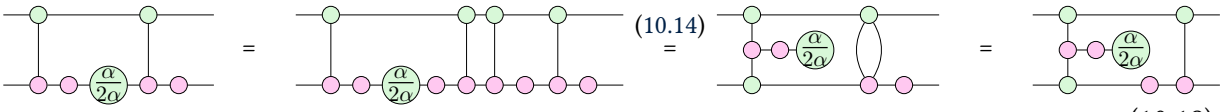
We can also define more general phase gadgets where the phases don't have to be related to each other, i.e. we can replace $Z(\alpha, 2\alpha)$ with $Z(\alpha, \beta)$. In this case we would still be calculating the value $x + y + z + w$ modulo 3, but then we apply a different phase depending on the value of this sum: if it is 0 we don't apply any phase; if it is 1 we apply ω^α; and if it is 2 we apply ω^β.

A particularly relevant choice of phases here is when $\alpha = \beta$. In this case, we apply the phase iff the sum value is not 0. For a trit x it turns out that $x^2 = 0$ if $x = 0$ and $x^2 = 1$ otherwise — this is actually a consequence of Fermat's little theorem and generalizes to $x^{p-1} = 1$ modulo p when $x \neq 0$ for p prime. Hence:



$$:: |x\rangle \mapsto \omega^{\alpha(x^2 \bmod 3)} |x\rangle \, . \tag{10.17}$$

There is a complication with the phase gadget circuit representation that doesn't arise in the qubit setting, which is that the CNOT gate is self-inverse while the CX qutrit gate is not. In Equation (10.14) we needed to pair a CX with a CX† to make the construction work. If we instead have a pair of CX gates, we get something a bit more complicated:



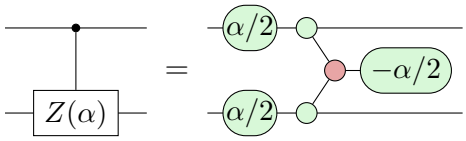
(10.18)

Remark 10.19. Another way to view qubit phase gadgets is as an exponentiated Pauli $e^{i\alpha Z\otimes Z}$ [CDD+20; CSD20]. This however does *not* generalize to qutrits, as the qutrit Pauli Z is not self-adjoint, and hence cannot be exponentiated to give a unitary. In fact, a qutrit phase gadget cannot be represented as the exponential of a 'pure tensor' like $e^{i\alpha A\otimes B}$. This does suggest that there could be another suitable generalization of a phase gadget that is the exponential of a tensor of *Gell-Mann matrices*, a qutrit basis of self-adjoint matrices we introduced in Chapter 5.2.

Controlled phase gates

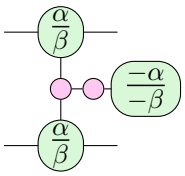
The other type of useful diagonal gate for qubits is the *controlled phase gate*. Such a gate applies a $Z(\alpha)$ gate on a qubit controlled on the value of a control. There are multiple ways in which we can generalize these to the qutrit setting. We consider first the $|2\rangle$-control. To see how we can build a $|2\rangle$-controlled Z phase gate, we will take inspiration from the qubit construction.

Recall that there we have:



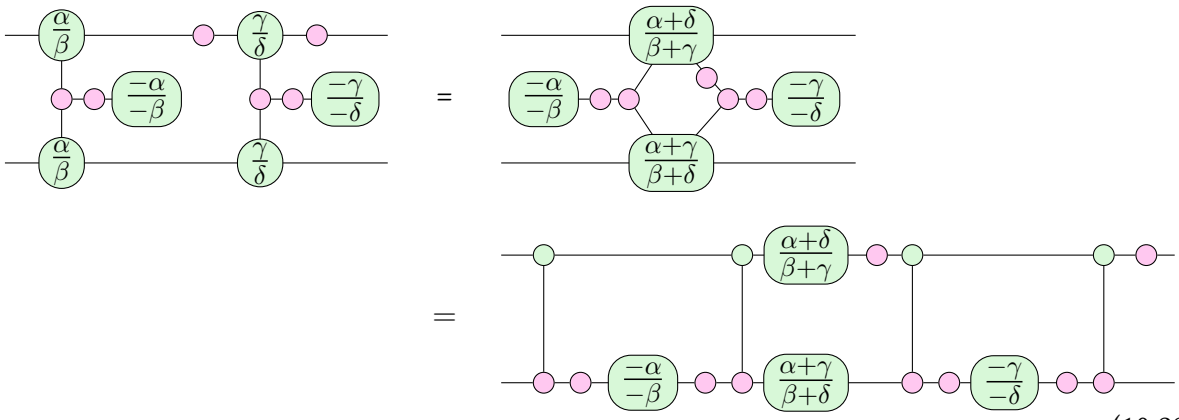
(10.20)

We can 'port' the right-hand side to the qutrit setting, by taking each of the phases to be a $Z(\alpha, \beta)$. However, we then run into some problems. It is easy to check that when the top qutrit (the control) is $|0\rangle$ that the diagram indeed acts as the identity on the bottom qutrit (the target). However, it implements a different phase gate on the target depending on whether the control is in $|1\rangle$ or $|2\rangle$:



$$\rightsquigarrow \begin{cases} Z(0,0) & \text{if control is } |0\rangle \\ Z(2\alpha-\beta, \alpha+\beta) & \text{if control is } |1\rangle \\ Z(\alpha+\beta, 2\beta-\alpha) & \text{if control is } |2\rangle \end{cases} \tag{10.21}$$

Seeing as we want to construct the $|2\rangle$-controlled gate that should act as the identity when the control is $|1\rangle$ this is a problem. We solve this issue by 'doubling up' the construction, with the second construction being conjugated by X_{12} on the control in order to interchange the role of $|1\rangle$ and $|2\rangle$:



(10.22)

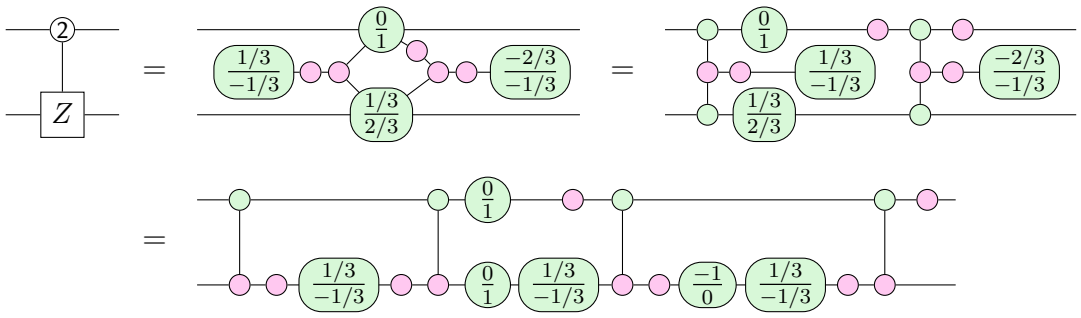
By referring to Equation (10.21) we see then that in order for Equation (10.22) to be equal to the $|2\rangle$-controlled $Z(\theta, \phi)$ gate it needs to satisfy a set of linear equations. We can solve these to get a (unique up to some Clifford phases) solution:

$$\alpha = \frac{\theta - \phi}{3} \qquad \beta = \frac{\theta}{3} \qquad \gamma = \frac{\phi}{3} \qquad \delta = \frac{\phi - \theta}{3} \tag{10.23}$$

This solution moreover has the desirable property that the phases in the decomposition have a denominator of three relative to the starting phase. This is natural for the qutrit Clifford-cyclotomic hierarchy of gate sets which we will discuss in the next chapter, where each

subsequent gate set in the hierarchy increments the power of 3 root of unity: $Z = Z(1, -1)$ gate, $T = Z(1/3, -1/3)$ gate, $\sqrt[3]{T} = Z(1/9, -1/9)$ gate, and so on.

We can hence write any $|2\rangle$-controlled phase gate using at most four CX gates and four phase gates. For example, if we pick $\theta = 1$ and $\phi = 2$ (so that we are constructing the controlled Z gate) we get:

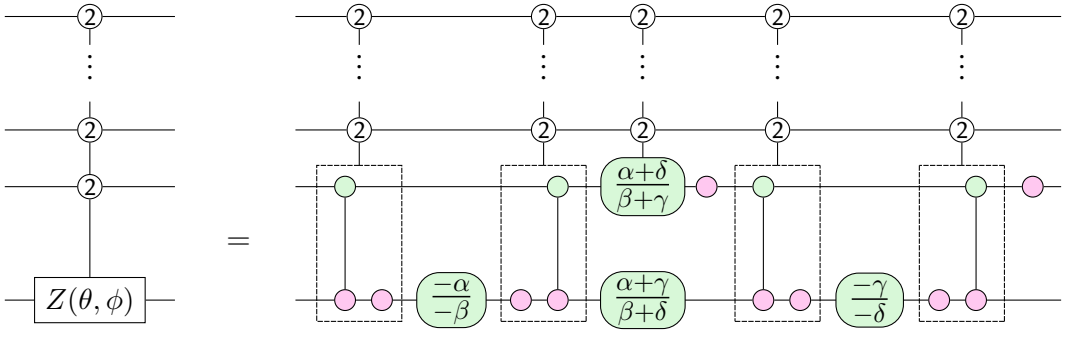


(10.24)

Here we write this blue dot with a 2 in it to denote a $|2\rangle$-control. We see then that our construction in the special case of $|2\rangle$-controlled Paulis indeed achieves the lowest known T-count of 3 [BCRS16]. By conjugating the control wire by either X_{+1} or X_{-1} we can make the gate instead be controlled on either $|1\rangle$ or $|0\rangle$.

We can add any number of controls to our construction in Equation (10.22) to make it controlled on any tritstring. Without loss of generality, let us say the tritstring in question is $|2\rangle^{\otimes n}$ (by conjugating with X gates, we can make this into a control on any tritstring of length n). The naïve way to construct this controlled gate is to inductively add controls to each gate in the decomposition: controlled constructions for the X or CX gate are described in, for instance, Ref. [YvdW22], while controlled Z phase gates can be constructed by recursively applying Equation (10.22). However, this method is not efficient as it requires an exponential number of gates as the number of controls increases.

We can do better by not adding controls to all the gates in the decomposition:



(10.25)

In the case where all the controlled gates fire, this indeed implements any desired Z phase gate on the target qutrit. Otherwise, none of the controlled gates fire, and then the bottom two qutrits becomes identity (use $(H2)$ and $(H4)$ on the top qutrit and (SZ) and (ID) on the

bottom qutrit). We hence get the following proposition. Actually, due to the gate count of any tritstring-controlled Toffoli improving from $O(n^{2.585})$ in Ref. [YvdW22] (the previous section) to $O(n)$ in Ref. [RvdWY23] (in the next chapter), the original cost of $O(n^{3.585})$ in [vdWY23, Proposition 4.2] of the paper has now been improved here to $O(n^2)$. The updated proposition is as follows:

Proposition 10.26. *Any tritstring-controlled qutrit Z or X phase gate can be constructed without ancillae and with $O(n^2)$ Clifford and phase gates.*

Proof. The X phase gates can be constructed from the Z phase gates by conjugating by Hadamards, so we only need to describe how to construct tritstring-controlled Z phase gates. Suppose we wish to construct a phase gate with n controls. By our work in Ref. [RvdWY23], each $|2\rangle^{\otimes(n-1)}$-controlled CX gate can be built ancilla-free using $O(n)$ qutrit Clifford+T gates. It then remains to show how to construct the $|2\rangle^{\otimes(n-1)}$-controlled $Z(\theta,\phi)$ phase gate in Equation (10.25). We do this recursively. To construct the gate with k controls, we need four controlled CX gates with $k-1$ controls and a $k-1$ controlled Z phase gate, which then needs four controlled CX gates with $k-2$ controls, and so on. The total asymptotic gate count is then $4O(n)+4O(n-1)+4O(n-2)+\ldots$ which gives us a gate count of $O(n^2)$. □

Note that in this construction, the size of the phases involved becomes exponentially smaller in the number of controls. We will next see that there is a notion of control which circumvents this issue.

Phase multipliers

The $|2\rangle$-controlled phase gate is just one possible way to extend the idea of a controlled-phase gate from qubits. Another way is to realize that for qubits we can describe the action of a controlled phase gate as $|x,y\rangle \mapsto e^{i\alpha\, x\cdot y}\,|x,y\rangle$. Indeed, if the control qubit is in the state $x=0$, then this is just the identity, while if $x=1$, we apply $e^{i\alpha y}$ which corresponds to a $Z(\alpha)$ gate on the $|y\rangle$ qubit.

We see then that while a phase gadget is based on the addition operation of $\mathbb{Z}_2$, controlled phase gates are based on the multiplication operation of $\mathbb{Z}_2$. This suggests that the controlled phase gate equivalent for qutrits should be $|x,y\rangle \mapsto e^{i\alpha x\cdot y}\,|x,y\rangle$ where now we take $x\cdot y$ modulo 3. We will show how we can construct this operation using phase gadgets. In order to distinguish this type of gate from the previously considered controlled phase gates, we will refer to a gate where the phase depends on $x\cdot y$ as a *phase multiplier*. Before we show how to build phase multipliers for qutrits, we first need to understand how to build them for qubits. For bits x and

y we have the relation

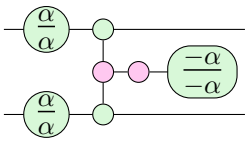
$$x \cdot y = \frac{1}{2}(x + y - (x \oplus y)). \tag{10.27}$$

Importantly, we are considering the $+$ operation here not modulo 2, but just as an action on real numbers, and we are writing $\oplus$ for addition modulo 2. Using this relation we can write $e^{i\alpha(x\cdot y)} = e^{i\frac{1}{2}\alpha(x+y-(x\oplus y))} = e^{i\frac{1}{2}\alpha x}e^{i\frac{1}{2}\alpha y}e^{-i\frac{1}{2}\alpha(x\oplus y)}$. This is where the circuit decomposition of Equation (10.20) comes from. This relation between additive and multiplicative phase gates follows from a Fourier-type duality that exists for semi-Boolean functions, which is explored in detail in Ref. [KvdWK19].

It turns out that a similar decomposition is possible for qutrits. Note that we can derive Equation (10.27) by starting with the equation $(x + y)^2 = x^2 + y^2 + 2x \cdot y$ and then realizing that $x^2 = x$ for $x \in \{0, 1\}$ so that this reduces to $x \oplus y = x + y + 2x \cdot y$ for bits. When working with trits we can't remove these squares, but we can still get a useful relation. Bring terms to the other side to get $-2x \cdot y = x^2 + y^2 - (x + y)^2$ and then use the fact that modulo 3 we have $-2 = 1$ to get $x \cdot y = x^2 + y^2 - (x + y)^2$. It is now straightforward to check that this continues to hold when we interpret the outer $+$ and $-$ here not modulo 3, but as operations on the real numbers, so that we get the relation:

$$x \cdot y \bmod 3 = (x^2 \bmod 3) + (y^2 \bmod 3) - ((x + y)^2 \bmod 3) \tag{10.28}$$

Hence, using Equation (10.17) we get the following decomposition:



$$:: \qquad |x, y\rangle \mapsto \omega^{\alpha(x\cdot y \bmod 3)} |x, y\rangle \tag{10.29}$$

We can easily generalize Equation (10.28) to as many variables as desired by iterating it. In fact, this n-qutrit phase multiplier defined by the action

$$|x_1, ..., x_n\rangle \mapsto \omega^{\alpha(x_1\cdot x_2\cdot ...\cdot x_n \bmod 3)} |x_1, ..., x_n\rangle \tag{10.30}$$

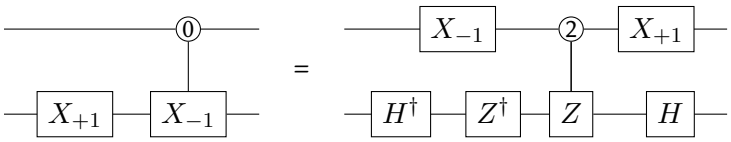
is, for the special case of $\alpha = 1$, an n-qutrit $C^{n-1}Z$ gate. This turns out to be a highly canonical for multiple reasons, as we will see in the next chapter when we define the qudit ZH-calculus and Toffoli+Hadamard gate set.

For three trits:

$$\begin{aligned}(x \cdot y) \cdot z &= x^2 \cdot z + y^2 \cdot z - (x + y)^2 \cdot z \\ &= x^4 + z^2 - (x^2 + z)^2 + y^4 + z^2 - (y^2 + z)^2 - (x + y)^4 - z^2 + ((x + y)^2 + z)^2 \\ &= x^2 + y^2 + z^2 - (x^2 + z)^2 - (y^2 + z)^2 - (x + y)^2 + ((x + y)^2 + z)^2 \end{aligned} \tag{10.31}$$

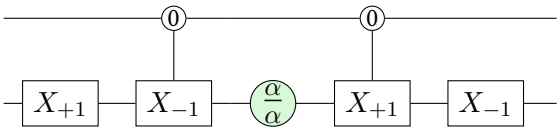
Here we used that $x^4 = x^2$ modulo 3.

Note that Equation (10.17) shows how to apply a phase proportional to the input trit squared modulo 3. However, in order to use this trick to apply a phase proportional to a higher order term such as $(y + x^2)^2$, we need a way to compute $y + x^2$ and store it "on the wire". In other words, we need to construct a circuit for the unitary defined by $|x, y\rangle \mapsto |x, y + x^2\rangle$. Because this simply adds 1 (modulo 3) to y iff $x \neq 0$, we construct it by adding 1 to the second qubit, and then applying a $|0\rangle$-controlled X_{-1} gate. To build this gate, we use the $|2\rangle$-controlled Z gate that we built from two phase gadgets in Equation (10.24):



$$:: \quad |x, y\rangle \mapsto |x, y + x^2\rangle \tag{10.32}$$

The above trick was also described in Ref. [BCRS16]. We further use this to build the type of phase gate below.



$$:: \quad |x, y\rangle \mapsto \omega^{\alpha(y+x^2)^2} |x, y\rangle \,. \tag{10.33}$$

We now have all the ingredients necessary to build the phase multiplier corresponding to the formula (10.31). What is interesting about this is that we do not have to use smaller factors of α. This is in contrast to the qubit counterpart of the formula (10.31) where we get a factor of $\frac{1}{4}$, due to the factor of $\frac{1}{2}$ in Equation (10.27). In fact, for qubits, the generalization to n variables will have a prefactor of $(1/2)^{n-1}$ so that for instance the three-qubit-controlled Z and controlled T gates cannot be constructed without ancillae in Clifford+T [GS13], as we need $\pi/8$ phase gates. Instead, no matter the number of qutrits, we do not get such a prefactor and can iteratively construct it as in the formula (10.31), as we did for qutrit Clifford+T in Ref. [YvdW22]. The circuit (10.33), alongside the square phase of Equation (10.17) suffices to generalize Equation (10.30) to any number of qutrits.

Proposition 10.34. *We can construct, without ancillae and using $O(2^n)$ Clifford+T, $Z(\alpha, \alpha)$, and $Z(-\alpha, -\alpha)$ gates, the n-qutrit phase multiplier gate defined by $|x_1, \ldots, x_n\rangle \mapsto \omega^{\alpha((x_1 \cdots x_n) \bmod 3)} |x_1, \ldots, x_n\rangle$.*

Proof. When we have two variables we use the formula

$$x \cdot y \bmod 3 = (x^2 \bmod 3) + (y^2 \bmod 3) - ((x + y)^2 \bmod 3) \tag{10.35}$$

to construct the two-qutrit phase multiplier. We generalized this to three variables in the

following way:

$$(x\cdot y)\cdot z \;=\; x^2\cdot z + y^2\cdot z - (x+y)^2\cdot z \;=\; x^2+y^2+z^2-(x^2+z)^2-(y^2+z)^2-(x+y)^2+((x+y)^2+z)^2. \tag{10.36}$$

To see how we go to 4 variables and beyond, we start with the expression $(x \cdot y \cdot z) \cdot w$ and decompose $x \cdot y \cdot z$ with the above formula resulting in terms $t_1^2, \ldots, t_n^2$. Each of these terms is a square, because that is the case for all the terms in Equation (10.35). Since we are working with qutrits we have $(t_j^2)^2 = t_j^2$. The terms in our formula are now of the form $t_j^2 \cdot w$. We apply Equation (10.35) to each of these. This gives us terms t_j^4, w^2 and $(t_j^2 + w)^2$. The first of these is just t_j^2, and by induction we already know how to construct the appropriate phase term on the circuit for this term. The second of these is w^2, and hence corresponds to a simple phase gate. Note that this is the same for each $t_j^2 \cdot w$ we are decomposing. Furthermore, the plus signs and minus signs on the terms are such that they almost all cancel, and we will have one copy of w^2.

The only 'interesting' new term we then get is hence $(t_j^2 + w)^2$. For instance, in Equation (10.36) the terms of this form are $(x^2 + z)^2$, $(y^2 + z)^2$ and $((x + y)^2 + z)^2$. The corresponding phase terms are constructed by using the gadget of Equation (10.32) to store $t_j^2 + w$ "on the wire" and then applying a $Z(\alpha, \alpha)$ phase gate.

Hence, if we go from 3 to 4 variables we get each of the original terms t_j^2, plus a w^2 term and a $(t_j^2 + w)^2$ term for each j. This straightforwardly generalizes to n variables, and it is then easy to check that we will have $2^n - 1$ terms.

We can build a circuit for the $n > 2$ qutrit phase multiplier by first building the circuit for $n - 1$ qutrits, and then inserting the gadget of Equation (10.33) after every application of a $Z(\alpha, \alpha)$ phase with as the target the nth qutrit. The reason this works is because the n-qutrit phase multiplier still contains every term of the $n - 1$ qutrit multiplier, but now also needs to combine those terms with the nth variable. In the four variable case, we would first store on a wire the value of the term t_j we need, and then apply a $Z(\alpha, \alpha)$ gate in order to get the phase $e^{i\alpha t_j^2}$. Then we would apply Equation (10.33) on the qutrit of w in order to get the phase $e^{i\alpha(t_j^2+w)^2}$. This construction involves temporarily storing $t_j^2 + w$ on the wire of w, so we can use this term if we want to construct the five-qutrit phase multiplier as well.

We then see that the cost of the n-qutrit phase multiplier in terms of (non-Clifford) gates is the cost of the $n - 1$ qutrit phase multiplier plus the cost of $2^{n-1} - 1$ applications of the Equation (10.33) gadget. In particular, each phase term requires precisely one of either a $Z(\alpha, \alpha)$ or a $Z(-\alpha, -\alpha)$ gate, so that we need $2^n - 1$ of them. The circuit of Equation (10.33) requires 6 T gates to construct, and hence the T-count of the n-qutrit phase multiplier is $6(2^{n-1} - 1) = 3 \cdot 2^n - 6$ (for $n > 2$). □

Remark 10.37. In Ref. [CGK17] the diagonal gates at all levels of the Clifford hierarchy are analysed for any qudit of prime dimension. They show for instance that the gate implementing $|x_1 \cdots x_n\rangle \mapsto \omega^{x_1 \cdots x_n} |x_1 \cdots x_n\rangle$ (which is the n-controlled $2\pi/3$ phase multiplier gate) is in the nth level of the Clifford hierarchy. This might be surprising as our construction shows how to build this gate, for any n, only using gates from the third level of the Clifford hierarchy (namely Clifford gates and the T gate). However, note that while the *diagonal* gates on a level of the hierarchy form a group, the full set of (not necessarily diagonal) gates is not closed under composition, and hence we can build higher-level unitaries using lower-level ones.

10.3 Emulating diagonal qubit unitaries with qutrit phase gadgets

We've now seen that we can use phase gadgets to build a number of useful diagonal unitaries. In this section we will see how we can build more general diagonal qutrit unitaries, and specifically those that *emulate* qubit operations. Qudit emulation of qubit operations can result in efficiency gains, by using higher level states rather than ancillae. While there has been significant work on emulating qubits using qutrits and qudits, much of this has been limited to realizing gates within classical reversible computing such as multiple-controlled Toffolis. In contrast, fewer works have addressed qutrit gate sets containing arbitrary phases. Examples include a $|2\rangle$-controlled $Z(0, \phi)$ decomposition in terms of qutrit-controlled qubit $\phi/3$ rotations [DW13] or quantum multiplexers and uniformly-controlled Givens rotations from the cosine-sine decomposition [KP06].

Definition 10.38 (Emulation)**.** Throughout this thesis, we will write $D_2 \overset{\text{e}}{\longmapsto} D_d$ to denote that a qubit map or diagram D_2 is emulated by a d-dimensional qudit map or diagram D_d. This choice of notation is because there is a surjective map from any n-qudit diagram to its qubit subspace on every qudit, in the computational basis. This map is the projection onto the space of computational basis states of the form $|\{0, 1\}^n\rangle$.

It is also helpful to define maps where some, but not all, of the qudits are emulating qubits. In those cases, we will note the subset of qudits that are emulating qubits with a subscript. For two-qutrit gates, we write $\overset{\text{e}}{\longmapsto}_{\text{bottom}}$ to denote that the bottom qutrit emulates a qubit, while the top qutrit remains a qutrit.

As this is a restriction, any given qudit diagram has a unique restriction to the qubit subspace, but any given qubit diagram has innumerably many qudit diagrams that emulate it.

We will first see how to emulate arbitrary qubit diagonal unitaries. Note that when we restrict to

the $\{|0\rangle, |1\rangle\}$ subspace, that a qutrit phase multiplier $|x_1, \ldots, x_n\rangle \mapsto e^{i\alpha((x_1 \cdots x_n) \bmod 3)} |x_1, \ldots, x_n\rangle$ only applies a phase α if and only if all n qubits are in the $|1\rangle$ state. Hence, for instance, the two-qubit $\mathrm{C}Z(\alpha)$ gate is directly emulated by its two-qutrit counterpart of Equation (10.30) with action $|x, y\rangle \mapsto e^{i\alpha x \cdot y} |x, y\rangle$. Consequently, by Proposition 10.34 we see that using qutrit Clifford+T gates along with $Z(\alpha, \alpha)$ and $Z(-\alpha, -\alpha)$ we can emulate the multiple-controlled $Z(\alpha)$ qubit gate without ancillae.

Now, by conjugating a multiple-controlled $\mathrm{C}Z(\alpha)$ gate by the appropriate X_{01} gates, we can decide on which input the $e^{i\alpha}$ phase is applied. Using multiple of these gates we can then arbitrarily decide for each input which phase should be applied to it. This then allows us to emulate an arbitrary diagonal qubit gate.

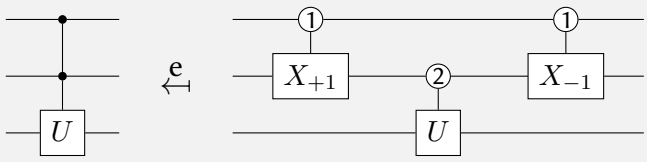
Proposition 10.39. *We can emulate the $\mathrm{diag}(\omega^{\alpha_1}, \ldots, \omega^{\alpha_{2^n}})$ qubit unitary using qutrit Clifford+T, $Z(\alpha_j, \alpha_j)$ and $Z(-\alpha_j, -\alpha_j)$ gates and without using ancillae.*

When using a standard qubit unitary synthesis algorithm, the desired phases $e^{i\alpha_j}$ would be implemented using many-controlled phase gates that require exponentially small angles $e^{i\alpha_j/2^n}$, which is problematic when the use-case is in fault-tolerant computing where non-Clifford phase gates must be constructed using magic state distillation and injection. Our construction could hence lead to some benefits in synthesising diagonal qubit unitaries using fewer non-Clifford resources.

It turns out that for the specific case of a qubit CCZ gate, that we can emulate it using qutrits in an even more efficient way. While we could use the emulation construction above, it turns out to be better to consider an altered construction.

Lemma 10.40. *Given a qutrit $|2\rangle$-controlled U gate for an emulated qubit unitary U, we can construct a qutrit emulation of the qubit CCU gate with the same non-Clifford cost as the $|2\rangle$-controlled U gate.*

Proof. One can readily verify, by initializing the top two qubits to $\{|00\rangle, |01\rangle, |10\rangle, \text{and } |11\rangle\}$, that the below qutrit decomposition from Ref. [GBD+19] emulates the qubit CCU gate.

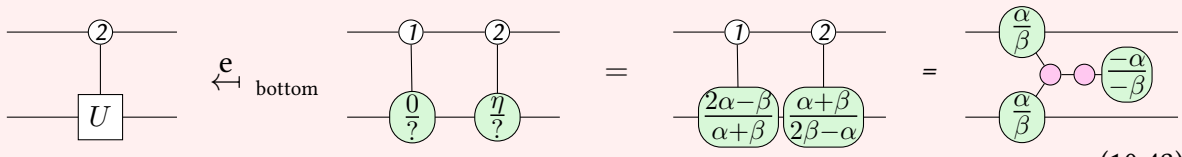


(10.41)

We can then replace the two non-Clifford $|1\rangle$-controlled X_{+1} and $|1\rangle$-controlled X_{-1} by CX and $\mathrm{CX}^\dagger$, preserving correctness of the emulation as the action on the $\{|0\rangle, |1\rangle\}$ subspace is unchanged [BRS17]. □

Using this lemma we see that to get an efficient emulation of the qubit CCZ, it remains to find an efficient qutrit emulation of the $|2\rangle$-controlled qubit Z gate.

Lemma 10.42. *Let $U = diag(1, \omega^\eta)$ be an arbitrary qubit Z phase gate. Then we can build the $|2\rangle$-controlled emulated U using the controlled phase gate of Equation (10.21):*



(10.43)

Here α and β satisfy, for some $k \in \mathbb{Z}$, $2\alpha - \beta = 3k$ and $\alpha + \beta = \eta$ and the question marks ? denote that these phases are irrelevant for the emulation.

If we choose $\alpha = 3/2$ and $\beta = 0$ in this construction we are emulating the $|2\rangle$-controlled qubit Z gate, because the phase of $\omega^{3/2} = -1$ applies iff the control qutrit is $|2\rangle$ and the target qubit is $|1\rangle$. Note that a $Z(3/2, 0)$ phase is equal to $X_{12}\ R\ X_{12}$, referring to the R gate from Observation 10.5. Hence:

Corollary 10.44. *The $|2\rangle$-controlled qubit Z gate can be emulated with R-count 3.*

Combining this corollary with Lemma 10.40 we arrive at our result.

Proposition 10.45. *The qubit CCZ gate can be emulated ancilla-free in qutrit Clifford+R with R-count 3.*

As shown in Ref. [HC17], any implementation of a CCZ gate requires at least four qubit T gates. Additionally any unitary implementation requires at least seven [GKMR14]. Here we see that surprisingly, by embedding the CCZ into qutrit space, we can construct it using just three non-Clifford single-qutrit gates and that moreover this is unitary and ancilla-free. This is also a new minimum amongst qudit emulations: for instance, in Ref. [HC19], they needed four qudit (for prime $d \geq 5$) T gates to emulate qubit CCZ.

10.4 Sample

how to implement phase multipliers on superconducting qutrits

This sample section describes a proposal for implementing qutrit phase multipliers of Equation (10.30) on superconducting transmon devices.

Remark 10.46. This section is based on joint work with Shuxiang Cao, Mustafa Bakr, Giulio Campanaro, Simone Fasciati, James Wills, Boris Shteynas, Vivek Chidambaram, John

van de Wetering, and Peter Leek. This was presented as the talk "Qutrit-ZX Calculus on superconducting transmon qutrits" by Shuxiang Cao at APS March Meeting in 2022 [CYB+22]. My contribution was to the theoretical aspects of synthesizing the pulse sequence jointly with Shuxiang Cao. The simulation of device parameters to search for pulse timings, in Figures 10.4 and 10.5, was done by Shuxiang Cao. For further information on superconducting transmon qudits, refer to Shuxiang Cao's thesis [Cao23] and papers on the subject [CBC+24b; CLB+24].

AC Stark shift induced ZZ interaction on superconducting qutrits was demonstrated for fluxonium qubits in Ref. [XFS+22] and for transmon qutrits in Ref. [GMM+22]. The latter work, put out since this work, engineers high-fidelity qutrit CZ and $\mathrm{CZ}^\dagger$ gates, largely superseding the work here. It is encouraging to know that the work here was on the right track, which I have included in hopes the reader might enjoy the learning experience, as I did.

In Figure 10.3, we see that the way transmon superconducting qubits are implemented, is by assigning $|0\rangle$ and $|1\rangle$ to the ground and first excited states respectively. In fact, transmon "qubits" are the restriction to the $\{|0\rangle, |1\rangle\}$ subspace of a quantum system with many energy levels. Therefore, superconducting quantum hardware is a natural platform for implementing qudits, as higher energy levels are already present but largely ignored in qubit devices.

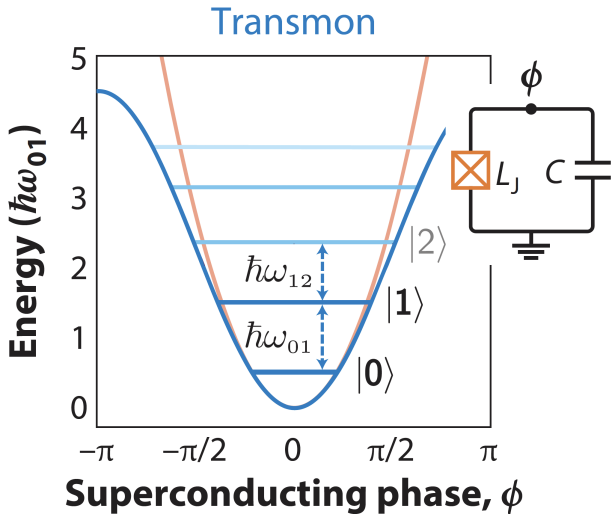


Figure 10.3: Energy spectrum of a transmon qubit reprinted from [KSB+20]. Due to the Josephson junction, this system exhibits a quantum anharmonic oscillator potential, which means that the energy levels have different transition frequencies between them. Therefore, unlike in a harmonic oscillator, driving one transition frequency should not simultaneously drive other transitions.

A system of two simultaneously-driven, coupled transmon devices indexed by $j \in \{c, t\}$, each with external drives, can be modelled by the following Hamiltonian in the frame of the

drive [MNM+21]:

$$H = \sum_{j=c,t} \Big[\underbrace{(\omega_j - \omega_d) a_j^\dagger a_j + \frac{\alpha_j}{2} a_j^\dagger a_j^\dagger a_j a_j}_{\text{Transmon}} + \underbrace{\epsilon\left(e^{i\phi_j} a_j + e^{-i\phi_j} a_j^\dagger\right)}_{\text{External drive}} \Big] + \underbrace{J\left(a_c^\dagger a_t + a_c a_t^\dagger\right)}_{\text{Capacitive coupling}} \tag{10.47}$$

in terms of the drive frequency ω_d, the bosonic annihilation operator a_j, the transition frequency ω_j between $|0\rangle$ and $|1\rangle$, the anharmonicity α_j, coupling strength J, and external drive amplitude $\epsilon e^{i\phi_j}$ for real ϵ and ϕ_j. Here $\hbar = 1$.

This proposed implementation of the parametrized ZZ interaction is by conditional Stark shifts. When both transmons are driven simultaneously, each two-qutrit Z basis state $|ct\rangle$ for $c, t \in \{0, 1, 2\}$, accumulates a phase difference at a speed δ_{ct}. $\delta_{ct} = 0$ whenever $c = 0$ or $t = 0$. For example, the action of this Stark effect-induced conditional phase shift on the below states is

$$|0\rangle (|0\rangle + |1\rangle + |2\rangle) \mapsto |0\rangle (|0\rangle + |1\rangle + |2\rangle) \tag{10.48}$$

$$|1\rangle (|0\rangle + |1\rangle + |2\rangle) \mapsto |1\rangle \left(|0\rangle + e^{i\delta_{11}t} |1\rangle + e^{i\delta_{12}t} |2\rangle\right) \tag{10.49}$$

$$|2\rangle (|0\rangle + |1\rangle + |2\rangle) \mapsto |2\rangle \left(|0\rangle + e^{i\delta_{21}t} |1\rangle + e^{i\delta_{22}t} |2\rangle\right) \tag{10.50}$$

The values of these δ_{ct} are a linear combination of the energy levels of the two-qutrit system:

$$-\delta_{ct} = E_{ct} + E_{00} - E_{c0} - E_{0t} \tag{10.51}$$

Through simulation using parameters close to that of superconducting qudit devices in the Department of Physics of the University of Oxford, the simulated δ_{ct} values are plotted against the external driving frequency and strength in Figure 10.4.

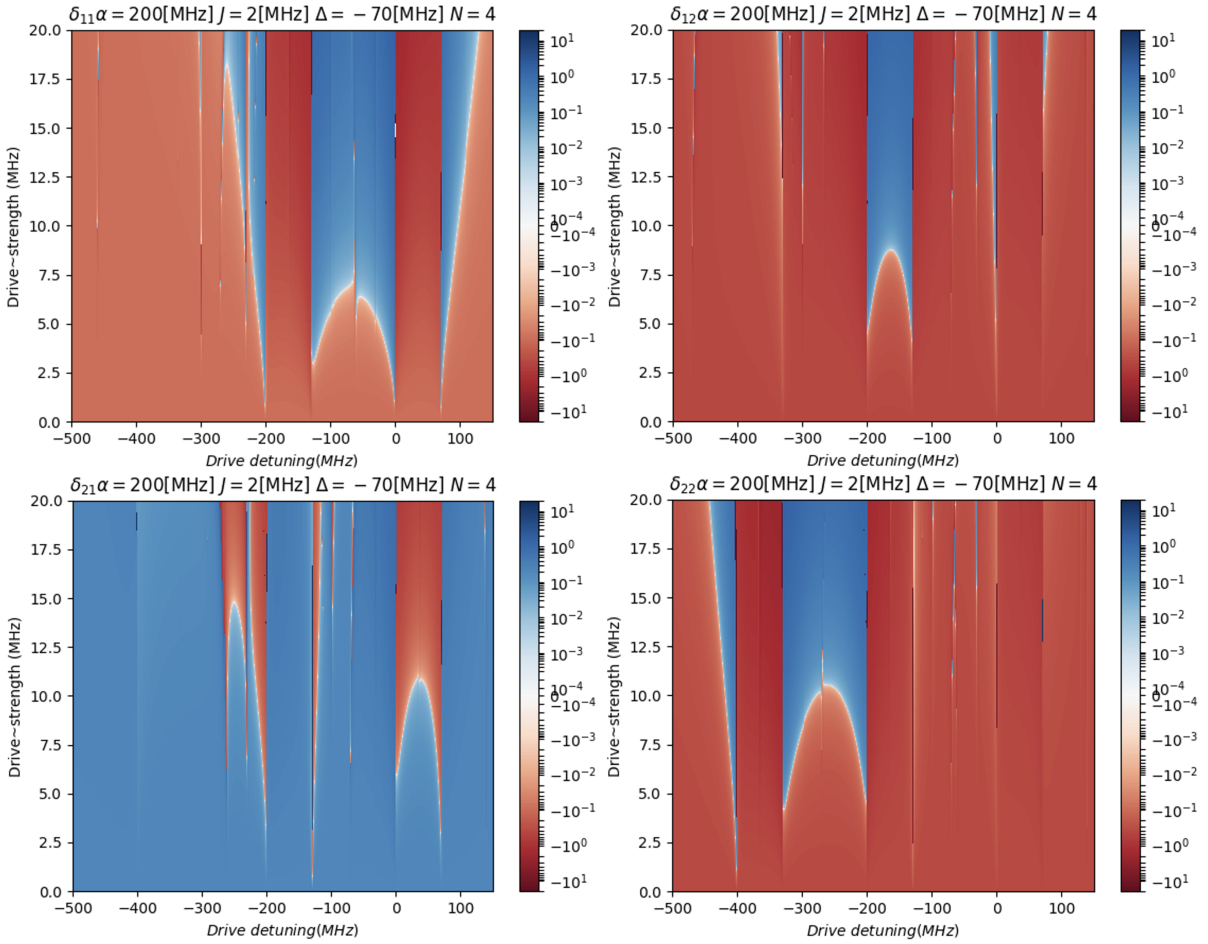
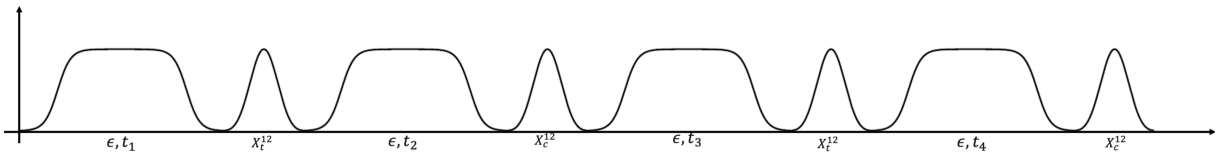


Figure 10.4: Simulation of phase difference accumulation speeds δ_{ct} in units of MHz, under the parameters: anharmonicity α of 200 MHz, coupling strength J of 2 MHz, detuning between the control and target qutrits Δ of -70 MHz, and truncation for second quantization to the lowest $N = 4$ energy levels.

Being a function of anharmonically spaced energy levels, a single Stark drive pulse is expected not to realize a two-qutrit ZZ phase multiplier requiring the total phase accumulated to satisfy

$$\delta_{12} = \delta_{21} = 2\delta_{11} = 2\delta_{22} \tag{10.52}$$

This is remedied by engineering a pulse sequence consisting of four Stark drive pulses, after each of which is exchange of population between $|1\rangle$ and $|2\rangle$.



To determine the durations t_1, t_2, t_3, t_4 of each of the four pulses implementing the two-qutrit

ZZ phase multiplier of Equation (10.30), solve the following linear system of equations:

$$\begin{bmatrix} \delta_{11} & \delta_{12} & \delta_{22} & \delta_{21} \\ \delta_{12} & \delta_{11} & \delta_{21} & \delta_{22} \\ \delta_{21} & \delta_{22} & \delta_{12} & \delta_{11} \\ \delta_{22} & \delta_{21} & \delta_{11} & \delta_{12} \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix} = \begin{bmatrix} \frac{2\pi}{3} + 2k_1\pi \\ \frac{4\pi}{3} + 2k_2\pi \\ \frac{4\pi}{3} + 2k_3\pi \\ \frac{2\pi}{3} + 2k_4\pi \end{bmatrix} \tag{10.53}$$

where k_i can be any integer number, i.e. the phases are modulo 2π. By modifying the target total accumulated phases on the right hand side of each equation, this pulse sequence can be adapted to compile other gate operations.

Varying driving strength and driving frequency, Figure 10.5 plots solutions to the above linear system of equations in order to verify the practicality of the total gate times.

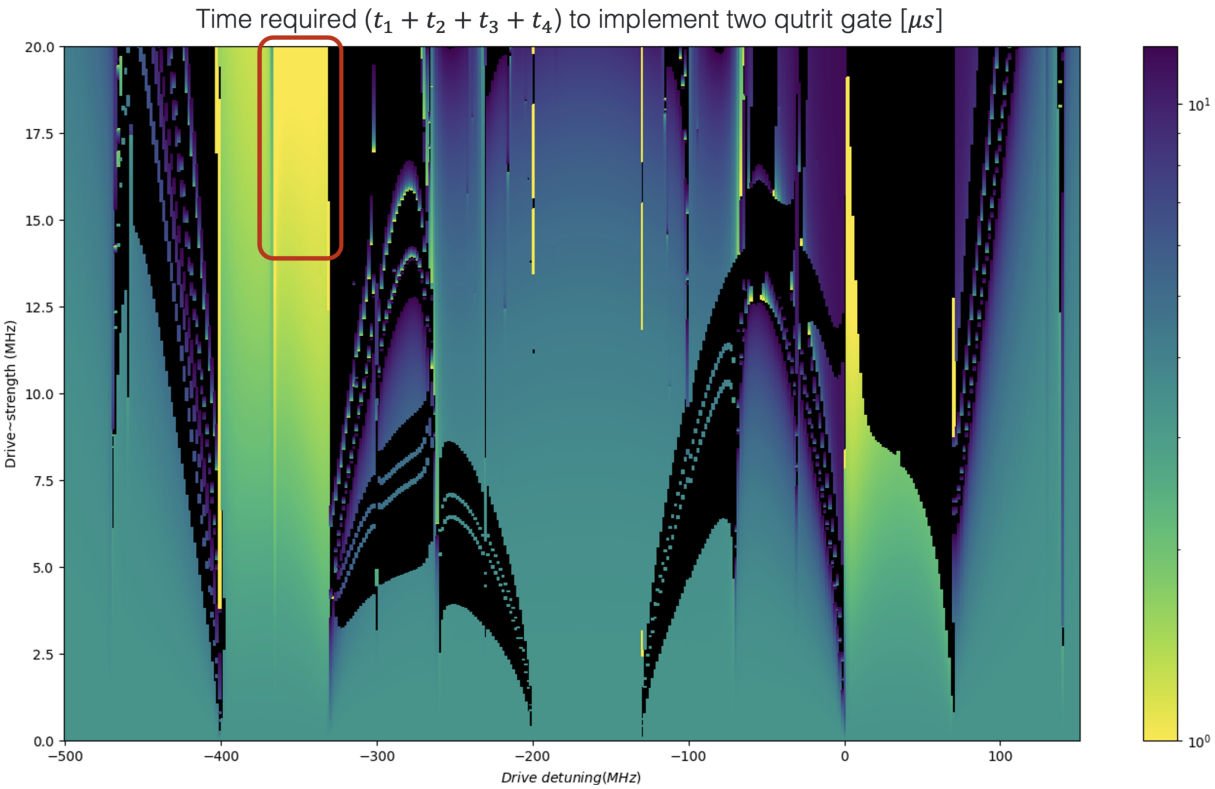


Figure 10.5: Black regions indicate where no solution for the linear system of equations can be found. Circled in red is a "sweet spot"—a region where the gate can be implemented in less than one microsecond.

10.5 Sample: Emulating qubits with qutrits

Remark 10.54. The work of this section has, since this thesis was written and revised, evolved into the paper "Transversal AND in Quantum Codes" [LY26], in collaboration with Christine Li.

This section presents further constructions for emulating qubit gates using qutrit gates. The motivation for this is to formalize a theory of fault-tolerance that is compatible with both qubit and qutrit stabilizer theory, in order to account for real-world systems where information is computed on in qubits which are a subspace of a larger qudit system. This section focuses on more concrete implementations to avoid underdeveloped speculation, and in hopes that these building blocks can generalize to more complex quantum protocols and higher-dimensional quantum systems. The section ends with a proposal to investigate the use of physical qutrits for logical qubit computation.

The first difficulty with developing a fault-tolerant theory of emulation is that qubit Cliffords are not a subgroup of qudit Cliffords.

Example 10.55. This is a small counterexample to show that doing a qubit Clifford gate on the qubit subspace of a qudit, with identity on all higher levels, is not necessarily a qudit Clifford operation. Recall the qutrit metaplectic gate, which is a single-qutrit non-Clifford gate:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{10.56}$$

Conjugating R by the qutrit Pauli X gate gives a gate which emulates the qubit Pauli Z gate on the qubit subspace of the qutrit.

$$X^\dagger R X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{10.57}$$

Example 10.58. We can also see that restricting a qutrit Clifford gate to its qubit subspace, is not necessarily a qubit Clifford operation. For example, the qutrit Pauli Z gate is

$$Z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \omega & 0 \\ 0 & 0 & \omega^2 \end{bmatrix} \tag{10.59}$$

where $\omega = e^{i\frac{2\pi}{3}}$ is not an element of the ring which all qubit Clifford matrices are required to be in.

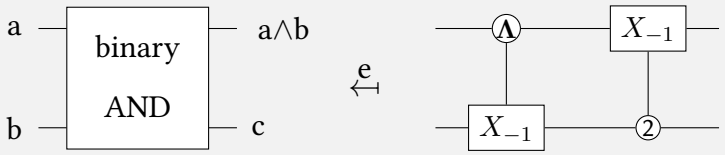
In this section, we investigate *emulation* of binary and qubit gates utilizing qutrits as an application of the qutrit controlled gates synthesized in Chapter 9.2. Emulating certain qubit gates using qudits can circumstantially be effective in conserving resources because it can be more efficient to use higher level states rather than ancillae; in fact, many of our constructions in this section are both unitary and require no ancillae.

As a first application of leveraging qutrit gates to accomplish a useful unitary otherwise impossible when restricted to qubit gates, we present two-qutrit unitary, fault-tolerant implementations of the binary AND and OR gates.

It is well-known that it is impossible to realize a two-qubit unitary for which one of the output qubits is the logical AND of the inputs — after all, the Toffoli gate is a three-qubit unitary. We show it is possible to perform a two-qutrit Clifford+T unitary that emulates the binary AND gate.

Lemma 10.60. *The binary AND gate can be emulated by a two-qutrit Clifford+T unitary with T-count 3.*

Proof. We will show that the below Clifford+T two-qutrit unitary emulates the binary AND gate:



(10.61)

Its cost is T-count 3 and Clifford CX-count 4. The cost calculation follows from the $|2\rangle$-controlled X_{+1} gate comprising of 3 Clifford CX gates and 3 qutrit T gates [BRS17]. The bottom output c is a qutrit. Throughout this section, the letter e above the equals sign indicates it is an emulation.

Its truth table is:

a	b	$a \wedge b$	c
0	0	0	0
0	1	0	1
0	2	2	2
1	0	0	2
1	1	1	0
1	2	1	1
2	0	2	1
2	1	1	2
2	2	2	0

Table 10.1: Truth table for a two-qutrit Clifford+T unitary emulation of the binary AND gate. The binary AND truth table is highlighted in blue.

On binary inputs, the left output bit is 1 if and only if both input bits are 1, and 0 otherwise. In fact, on binary inputs, it outputs the corresponding ternary representation. □

Interestingly, it requires 3 qutrit T gates and 4 qutrit Clifford CX gates, outperforming the Toffoli gate for which it is known that the optimal cost is 7 qubit T gates in the no-ancilla case [GKMR14], and 6 qubit Clifford CX gates independent of the number of ancillae [SM09]. When any two-qubit gates are permitted, the optimal two-qubit gate count is 5 for performing the Toffoli gate unitarily [YDY13]. Even when ancillae and operations conditioned on measurement are permitted, the best known qubit Clifford+T decomposition of the Toffoli gate is 4 qubit T gates [Jon13].

By applying the X_{01} gate (which emulates the binary NOT gate) to both inputs and to the AND'd output of Equation (10.61), in accordance with DeMorgan's law, the binary OR gate is emulated with the same T-count and Clifford CX-count.

This leads us to:

Proposition 10.62. *All of classical binary logic can be emulated using qutrit Clifford+T unitaries.*

Proof. Follows from the well-known fact that NAND (NOT AND) is a universal gate for classical binary logic. □

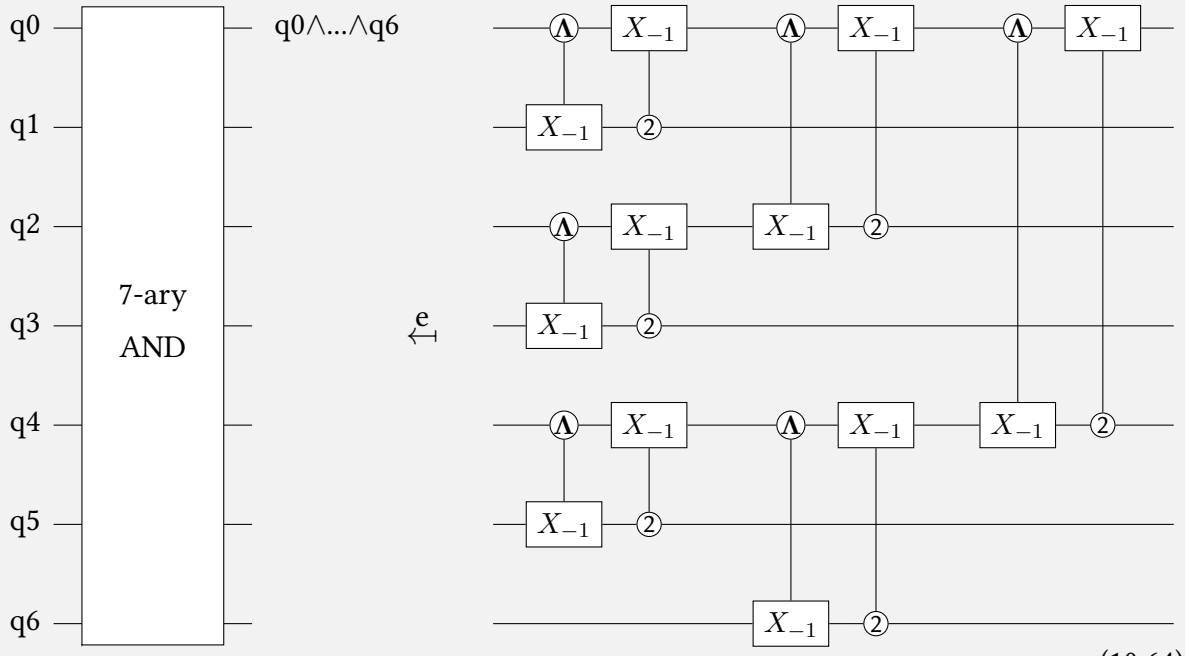
In particular, emulating NOT and XOR are easy (as this is with qutrit Cliffords), whereas emulating AND is not easy (as this is qutrit non-Clifford).

We can emulate a number of useful gates unitarily, and therefore reversibly, as a result. In

particular:

Proposition 10.63. *Consider the n-ary binary AND gate, which returns 1 iff all binary inputs are 1, and 0 otherwise. It can be emulated unitarily by Clifford+T gates without ancillae, in log(n) depth, with T-count $3n-3$ and Clifford CX-count $4n-4$.*

Proof. The idea is to apply the T-count 3 emulation of the binary AND gate from Lemma 10.60 $n-1$ times, as a binary tree structure. This generalizes to any positive integer n. The $n=7$ case is shown below:



(10.64)

□

At first glance, it may seem that with the exception of the AND'd output, the other outputs can be disregarded. However, they are necessary for the gate as a whole to be unitary and thus reversible. At the end of this section, we will demonstrate the usefulness of being able to compute and uncompute the n-ary binary AND gate.

Emulating the qubit X, Z, S, CX and CZ gates

We now proceed to investigate more generally the use cases of the ternary controlled gates we developed in Chapter 9.2 to fault-tolerantly emulate qubit gates. We begin by noting that emulation of qubit gates in general is an open problem. The qubit Hadamard and CNOT operations cannot be emulated utilizing qutrit Clifford gates [BRS17]. Let $\zeta = e^{i\frac{2\pi}{9}}$. Single-qutrit Clifford+T operators admit a normal form, which imposes the necessary but not sufficient condition that the matrix entries belong to the ring $\mathbb{Z}\left[\frac{1}{1-\zeta}\right]$ as independently determined by Prakash, Jain, Kapur, and Seth [PJKS18]; and Glaudell, Ross, and Taylor [GRT19]:

Definition 10.65 ($\mathbb{Z}\left[\frac{1}{1-\zeta}\right]$)**.** The matrix elements of all single-qutrit Clifford+T unitaries must be elements of the ring

$$\mathbb{Z}\left[\frac{1}{1-\zeta}\right] = \left\{ \frac{A}{(1-\zeta)^k} \middle| A \in \mathbb{Z}\left[\zeta\right], k \in \mathbb{N} \right\} \tag{10.66}$$

where

$$\mathbb{Z}\left[\zeta\right] = \left\{ \sum_{i=0}^{5} a_i (\zeta)^i \middle| \forall i, a_i \in \mathbb{Z} \right\} \tag{10.67}$$

Because $\frac{1}{\sqrt{2}}$ is not in the ring in Definition 10.65, it is impossible to exactly and deterministically emulate the qubit Hadamard gate utilizing single-qutrit Clifford+T gates. In general, the qubit Clifford operations do not neatly embed in the qutrit Clifford fragment.

In spite of this, we can still emulate quite a few qubit gates using qutrits. The qubit X gate can be emulated by the qutrit Clifford X_{01} gate. The qubit Z gate can be emulated by Clifford equivalence to the qutrit R gate:

$$\text{qubit } Z \overset{\text{e}}{\hookleftarrow} X_{12}\,\mathrm{R}X_{12} \tag{10.68}$$

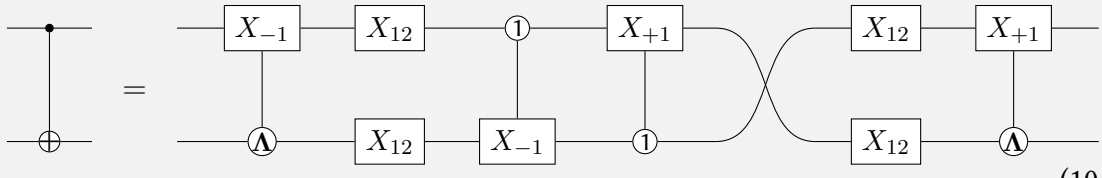
Likewise, the qubit S gate can be emulated by Clifford equivalence to the qutrit $\sqrt{\mathrm{R}}$ gate:

$$\text{qubit } S \overset{\text{e}}{\hookleftarrow} X_{12}\sqrt{\mathrm{R}}X_{12} \tag{10.69}$$

Extending magic state injection of the R gate [ACB12a], despite being nowhere in the Clifford hierarchy, it can be shown that the $\sqrt{\mathrm{R}} = \mathrm{diag}(1, 1, i)$ gate admits implementation by a repeat-until-success injection protocol of the $(1, 1, i)^\top$ state, where the probability of success rapidly converges to 1 with the number of rounds.

Lemma 10.70 ([BRS17])**.** *The qubit CX gate can be emulated by qutrit Clifford+T gates unitarily without ancillae, with T-count 6 and Clifford CX-count 8.*

Proof. For reference, the decomposition by Bocharov, Roetteler, and Svor [BRS17, Proposition 4] is given as a circuit diagram below:



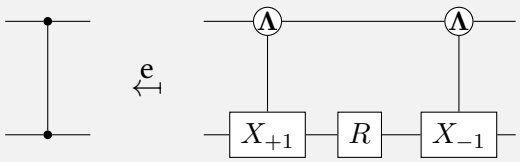
(10.71)

Like the qubit SWAP gate, the qutrit SWAP gate can be decomposed as three of the same

two-qutrit gate applied successively in alternating directions. By optimizing thereafter, the Clifford CX-count can be further reduced to 8. □

Proposition 10.72. *Utilizing one R gate, we can exactly emulate the qubit CZ gate:*

Proof. Although we can achieve a three R-count decomposition by conjugating the control of the $|2\rangle$-controlled qubit Z gate in Corollary 10.44 by X_{12}, we can do better as follows:



(10.73)

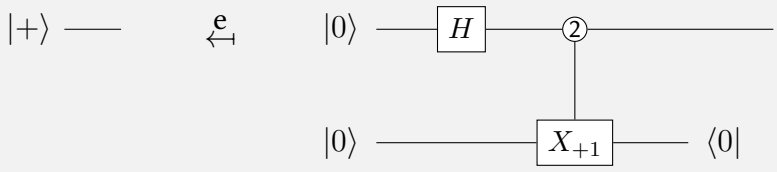
On qubit input states $|00\rangle$, $|01\rangle$, and $|10\rangle$, the states' sum modulo 3 is 0 or 1, and so no phase factor is applied. When the input state is $|11\rangle$, the states' sum modulo 3 is 2, and so a phase factor of -1 is applied. □

The qubit $|+\rangle$ and $|-\rangle$ states

The qubit $|+\rangle$ state is qutrit Clifford equivalent to the Norell state, for which two magic state distillation protocols have been developed by [DH15; Pra20]. Below, we present a new protocol for synthesizing the qubit $|+\rangle$ state which requires only Clifford+T gates, and computational basis state preparation and measurement. By applying one qutrit Clifford+T gate Clifford equivalent to the R gate before measurement, this protocol can be adapted to synthesize the qubit $|-\rangle$ state, which is Clifford equivalent to the i eigenstate of the qutrit Hadamard gate.

Lemma 10.74. *The qubit $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ state can be synthesized with 2/3 probability of success using 3 qutrit T gates and qutrit stabilizer states, gates, and measurements, with one ancilla qutrit.*

Proof. The qutrit Clifford+T circuit below produces the qubit $|+\rangle$ state with 2/3 probability.



(10.75)

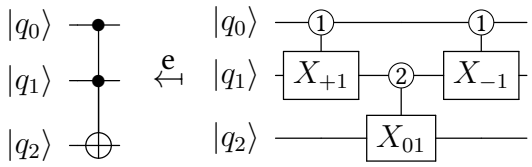
Before measurement, the state is in uniform superposition of $|00\rangle$, $|10\rangle$, and $|21\rangle$; measuring the bottom qubit to be the $|0\rangle$ state then produces the $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ state, equal to the qubit $|+\rangle$ state. □

Corollary 10.76. *By application of the $X_{12}RX_{12} = diag(1, -1, 1)$ gate before measurement, the qubit $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$ state can be subsequently attained requiring only Clifford+T gates, and computational basis state preparation and measurement, with one ancilla qutrit.*

Corollary 10.77. *By Clifford equivalence, the above proposition and corollary respectively construct the Norell state $\frac{|1\rangle+|2\rangle}{\sqrt{2}}$, and the i eigenstate of the qutrit Hadamard gate $|H, i\rangle = \frac{|1\rangle-|2\rangle}{2}$.*

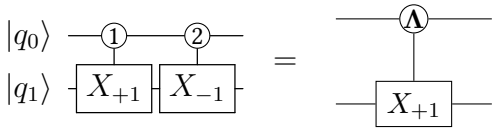
Qutrit decomposition of qubit Toffoli gate, linear depth

A qubit Toffoli gate can be built from qutrit gates using the decomposition by Gokhale et. al. [GBD+19]:

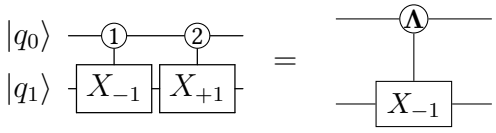


(10.78)

To exactly emulate the qubit CCX gate, the X gate on the target qutrit must be X_{01}. In their presentation of this gate they simply denoted the gate on the target qutrit as X, referring to any of the five single-qutrit computational basis permutation gates. The action of the gate is irrelevant for the cases where any of the input qutrits are in the $|2\rangle$ state, making it more efficient to use Clifford CX gates to replace the top two controlled gates:

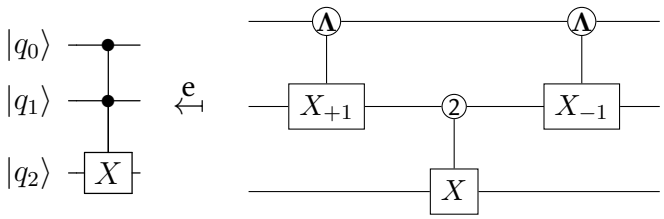


(10.79)



(10.80)

Without compromising the correctness of the qubit Toffoli gate, the above decomposition in Equation (10.78) was modified by Bocharov, Roetteler, and Svore as follows [BRS17, Proposition 6]:



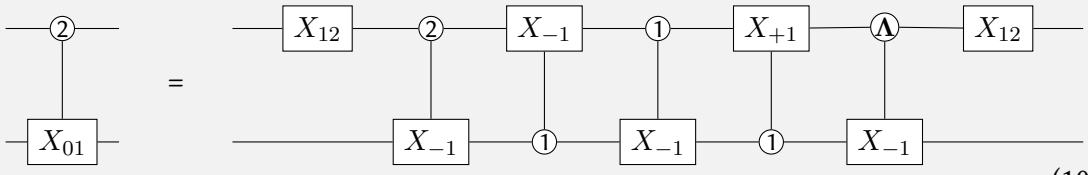
(10.81)

They concluded that the qubit CCX gate takes 15 T-gates to emulate with qutrit Clifford+T gates, taking the bottom gate to be the $|2\rangle$-controlled X_{01} gate, as the X_{01} gate emulates the qubit X gate. They lower the T-count to 12 by adding one clean ancilla.

We show that the T-count can be 12 without ancillae, by $|2\rangle$-controlling an emulation of the qubit X gate instead of the X_{01} gate.

Proposition 10.82. *The $|2\rangle$-controlled qubit X gate, which executes the qubit X gate if and only if the control qutrit is in the $|2\rangle$ state, can be emulated fault-tolerantly by a two-qutrit Clifford+T unitary without ancillae, with a T-count of 12.*

Proof. The decomposition is as follows. Recall that each of the 4 $|2\rangle$-controlled $X_{\pm 1}$ gates has a T-count of 3.



(10.83)

□

Corollary 10.84. *By Equation (10.81), the qubit CCX gate can be emulated unitarily without ancillae using Clifford+T gates, with T-count 12.*

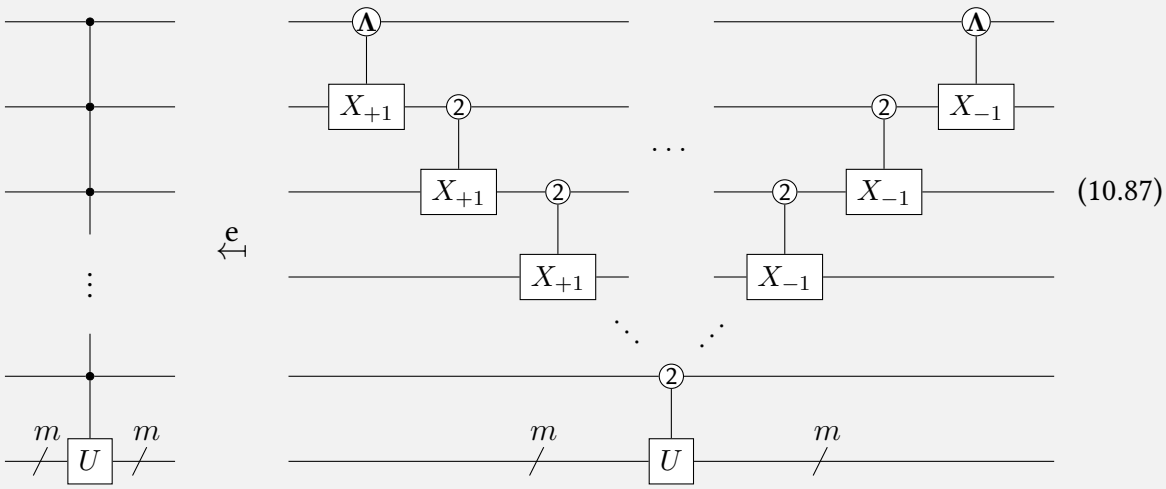
Proposition 10.85. *The qubit CCZ gate can be emulated unitarily without ancillae using Clifford+R gates, with R-count 3 and Clifford CX-count 5.*

Proof. Consider the qubit CCX gate emulation in Equation (10.81). If instead the controlled gate on the target is the $|2\rangle$-controlled qubit Z gate in Corollary 10.44, which has R-count 3 and Clifford CX-count 3, the emulated gate is the qubit CCZ gate. □

This is the first unitary decomposition with a non-stabilizer gate count of 3 for qutrit emulation of the qubit CCZ gate, albeit using R gates instead of T gates. Previously, Heyfron and Campbell gave a unitary CCZ decomposition with T-count 4 in terms of qudit Clifford + T where the qudits are of prime dimension greater than 3 [HC19, Figure 1].

Proposition 10.86. *The technique of emulating the qubit Toffoli can be generalized to n controls, and for any unitary U which can be $|2\rangle$-controlled.*

Proof.



(10.87)

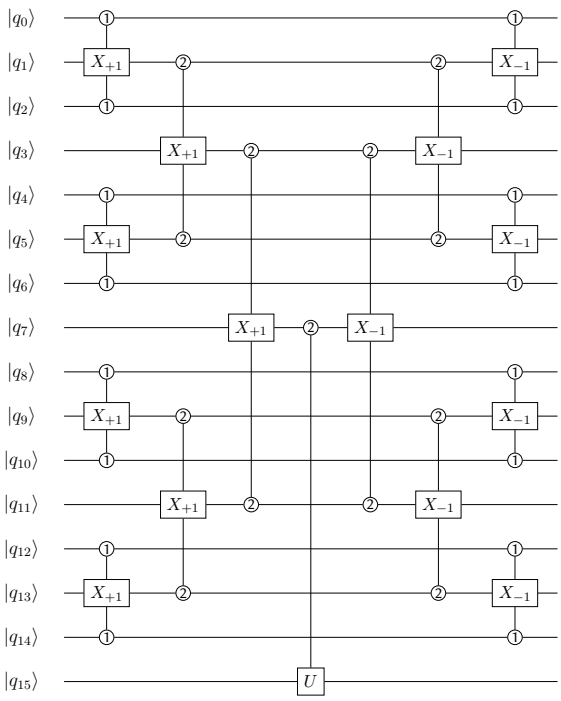
As each $|2\rangle$-controlled $X_{\pm 1}$ gate is comprised of three qutrit T gates, the T-count is $6(n-2)$ plus the cost of implementing the $|2\rangle$-controlled unitary. □

Corollary 10.88. *As the qubit Toffoli gate $|2\rangle$-controlled qubit X gate can be constructed with T-count 12 according to Proposition 10.82, the qubit generalized Toffoli with n controls can be unitarily emulated without ancillae using qutrit Clifford+T gates, with T-count of exactly $6n$.*

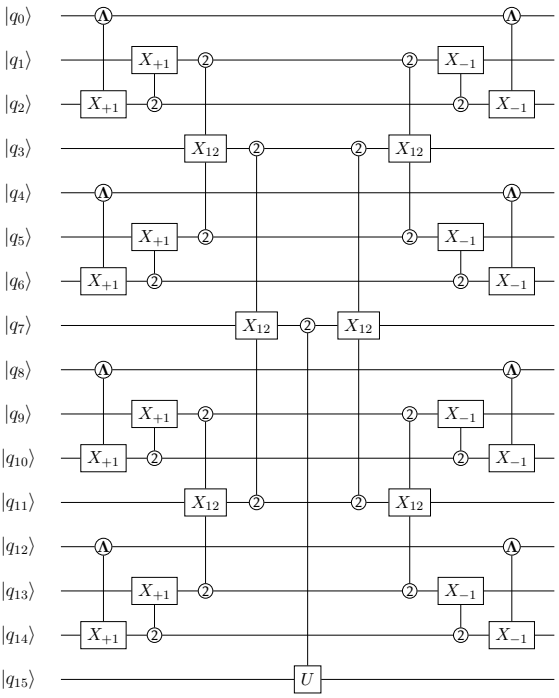
This is an improvement over [BRS17, Corollary 9], which emulates the qubit CCCNOT gate (i.e. $n = 3$) with T-count 18 using two clean ancillas, or T-count 21 using one clean ancilla.

Qutrit decomposition of qubit Toffoli gate, logarithmic depth

Gokhale et. al. [GBD+19] leveraged the $|22\rangle$-controlled gate to construct a qutrit emulation of the qubit Toffoli with n controls, with circuit depth logarithmic in n. We reprint their decomposition in Figure 10.6a. They cited the decomposition for the $|22\rangle$-controlled X gates by Di and Wei, in terms of qubit Clifford+T gates pairwise between the three qutrit computational basis states. As far as we know, a protocol to simultaneously error correct for these three gate sets has not been formulated.



(a) Qubit Generalized Toffoli gate emulation using intermediate qutrits by Gokhale et. al. [GBD+19, Figure 5].



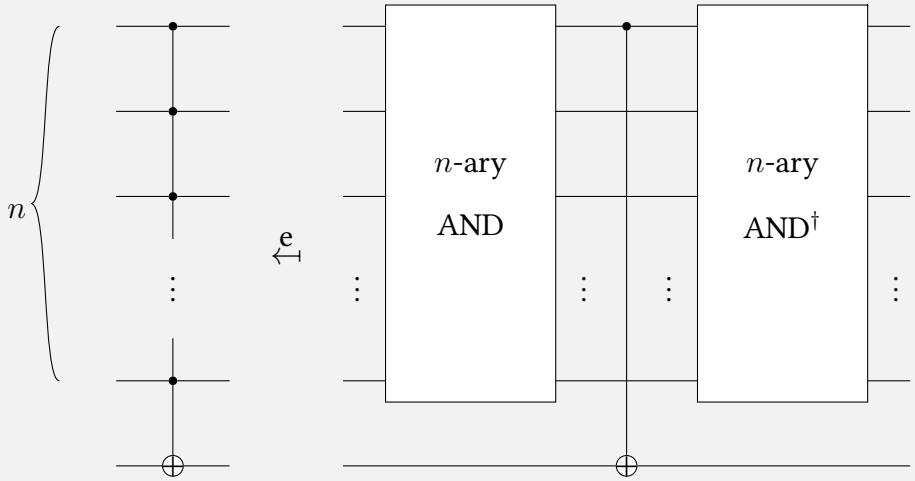
(b) We modified the left circuit to optimize T-count, given that we presently have a lower T-count decomposition for the $|22\rangle$-controlled X_{12} gate than the $|22\rangle$-controlled X_{+1} gate.

Using our decomposition for the $|22\rangle$-controlled dualizer gate in Equation (9.13), we adapt their decomposition in Figure 10.6a to the qutrit Clifford+T gateset. We thereby achieve linear qutrit T-count for the qubit generalized Toffoli gate, whilst retaining the desirable properties of logarithmic circuit depth and no logical ancillae needed (i.e. only using those necessary for magic state injection). We find it to be more efficient in qutrit T-count to modify each of their $|22\rangle$-controlled X_{+1} gates to $|22\rangle$-controlled X_{12} gates, for a constant factor improvement in lowering the T-count. To further decrease the resource requirements, we note that all boundary (either the inputs or outputs are qubits) $|22\rangle$-controlled gates in this decomposition are overconstrained; substituting them with our CCX_{+1} unitary decomposition in Equation (10.78) with T-count 3 suffices.

Despite this decomposition being linear in T-count, the coefficient of linearity is quite high, given that our decomposition for the $|22\rangle$-controlled X_{12} gate has T-count 51. It is the first fault-tolerant decomposition for it and we have not yet attempted to optimize this, reason being that we discovered a far more efficient construction:

Theorem 10.89. *The qubit C^n X gate can be emulated unitarily without ancillae in qutrit Clifford+T gates in O(log(n)) depth, with T-count exactly $6n$ and Clifford CX-count $8n$.*

Proof. We first apply the n-ary binary AND gate emulation of Proposition 10.63 with T-count $3(n-1)$ and Clifford CX-count $4(n-1)$, where the top output wire is the AND'd output. Then, we apply the qubit CX gate emulation of Lemma 10.70 controlled on the AND'd output, targeting the target of the C^n X gate, with T-count 6 and Clifford CX-count 8. Finally, we uncompute the n-ary binary AND gate by applying its inverse, also with T-count $3(n-1)$ and Clifford CX-count $4(n-1)$.



(10.90)

The T-count totals exactly $6n$ and the Clifford CX-count totals exactly $8n$. □

Corollary 10.91. *This construction can be generalized to gates of the form C^n U, controlled on $|1\rangle^{\otimes n}$ for qubit inputs, where U is any target unitary acting on m qudits for which a controlled version, for some type of control, can be built.*

Efficiency gains in emulating qubit logic on qutrits

We present the summary table of the most efficient qutrit emulation of each binary or qubit gate that we have found. In addition to giving the non-Clifford gate count, for near-term applications we give the number of two-qutrit operations required, as on present-day qubit quantum computers two-qubit operations are the dominant noise source. We also give the qutrit Clifford CX count where all other operations are single-qutrit Clifford+T gates.

Gate emulated	Non-Clifford gates	Clifford CX's	Two-qutrit gates
Binary AND	3 T	4	1
Binary OR	3 T	4	1
n-ary Binary AND	$(3n-3)$ T	4n-4	$n-1$
Qubit X (i.e. Binary NOT)	0	0	0
Qubit Z	1 R	0	0
Qubit CX	6 T	8	1
Qubit CZ	3 R	3	1
Qubit CCX	12 T	15	3
Qubit CCZ	3 R	5	3
Qubit C^{n}X, linear depth	$6n$ T	$6n+3$	$2n-1$
Qubit C^{n}Z, linear depth	$(6n-12)$ T + 3 R	$6n-7$	$2n-1$
Qubit C^{n}X, log depth (AND)	$6n$ T	$8n$	$2n-1$
Qubit C^{n}Z, log depth	$(6n-12)$ T + 3 R	$8n-5$	$2n-1$
Qubit C^{n}X, log depth [GBD+19]	$27n+12$ T	$64n+13$	$3.5n+1$

Table 10.2: Counts for decompositions of binary and qubit gates emulated unitarily without ancillae by qutrit gates in this thesis. The qubit CX gate decomposition is by [BRS17], which we then optimized to attain a qutrit Clifford CX-count of 8.

In this section, we present discussion of the results in Table 10.2. We start with the binary AND gate, which is Clifford equivalent to the binary OR gate. We conjecture that our T-count of 3 to emulate the binary AND gate is optimal for any prime qudit dimension, for the reason that the qudit T gate has phase $2\pi/(d^2)$, which yields $2\pi/d$ (and such a rotation in the X basis corresponds to the X_{+1} operation) on thrice application when $d=3$. To summarize our comparison to known optimums in the qubit case, the binary AND gate, which is impossible unitarily without ancillae on qubits, is more efficient in every metric in the table than the qubit alternative, the Toffoli gate. Moreover, computing and uncomputing the n-ary binary AND gate with a qubit CX in-between is an effective means of emulating the qubit n-controlled Toffoli gate (Qubit C^{n}X gate, log depth in Table 10.2) using exactly $6n$ T gates; we would like to know whether this is optimal. More generally, emulation using qutrits warrants further study; the existence of all binary logic gates unitarily (and thus reversibly) in the qutrit Clifford+T gateset implies that all classical computation built out of these gates can be transferred to the qutrit setting. For instance, it would be fascinating if NAND gates could be leveraged for constructs for quantum memory akin to the D Flip Flop or D Latch in classical computing.

To comment on the efficiency of the qubit CCZ gate emulation in Proposition 10.45, we remarked that we believe this to be a new lower bound of 3 on the required number of one non-stabilizer gate supplementing the Clifford gateset admitting a magic state distillation and injection protocol. The qubit case where ancillae and operations conditioned on measurement are permitted has been proven to require 4 T gates, whereas for the qudit case for any prime $d>3$ the best known decomposition to date requires 4 T gates. We conjecture that our lower

bound of 3 is optimal for any prime qudit dimension, considering that the controlled phase gate from phase gadgets paradigm requires three Clifford equivalent phase gates, which to get a phase of π when the qudit dimension is any prime d cannot be Clifford phase gates for which phases are multiples of $2\pi/d$.

We present a baseline comparison between the linear and log depth emulations of the qubit C^nX gate. Both are fault-tolerant, unitary implementations without ancillae which consume exactly $6n$ T gates. The log depth circuit is desirable for the obvious reason of runtime efficiency. With that said, the linear depth circuit wins in Clifford CX-count: $8n$ vs $6n$ scaling. Both require $2n-1$ two-qutrit operations. Considering NISQ architectures, the linear depth circuit may be preferable for devices with more linear connectivity such as trapped ions, whereas the log depth circuit may be preferable for devices with more 2D connectivity such as square or hexagonal lattices. Any two-qudit gate between qudits a path length p apart in the connectivity graph requires O(p) swaps to become adjacent. Near-term devices typically have low connectivity; as a consequence, the theoretical lower bound of the number of two-qubit gates necessary for state-of-the-art qubit Toffoli (C^nX) gate decompositions is in practice underestimated by a factor of n, as O(n) swap gates are necessary whenever a two-qubit operation is executed on non-adjacent qubits. In our log depth circuit, the maximum degree of connectivity required for no swaps to be necessary is at most $\log_2(n)$. The log depth, unitary, and ancillae-free proposal by Gokhale et. al. [GBD+19] requires a max degree of connectivity of 3, *independent* of n, which we have enabled to become fault-tolerant, and in this section we modified to optimize to $27n+12$ T gates. Another proposal which requires a max degree of connectivity of 3 is the non-fault-tolerant proposal by Kiktenko et. al. [KNX+20] which couples a qudit of degree $d \geq n+1$ to n control qubits. The linear depth circuit is optimal in terms of routing because it requires a max degree of connectivity of 2; in NISQ computation, qubit Hadamard gates can be subsequently applied around both the emulated target at the end of the line, and the desired target of the C^nX gate, for the target to be any of the qutrits.

Overall, utilizing the qutrit Clifford + T gateset to emulate a qubit NOT gate with any number of controls offers the advantage that no ancilla bits are needed; moreover, both the CX and T gate counts are linear in the number of controls. This is in contrast to the qubit case; Gidney concluded in his blog post that "with zero ancilla bits, it's impossible to build NOTs with lots of controls out of smaller operations", with the exception of allowing bootstrapping an ancilla bit out of small phase shifts exponentially smaller in the number of controls [Gid15].

Emulating qubit gates in a qutrit code

While there has been work on qudit circuit constructions implementing qubit operations, there is a dearth of quantum error correcting codes with more than one qudit dimension in mind. Ref. [Gun25] presents methods for working with composite dimensional stabilizer codes and

suggests that "this relative absence is due to the underlying ring theoretic tools required for analyzing such systems".

Indeed, recall our two definitions of the Clifford group while holds for any prime qudit dimension: One by preservation of the Pauli group under conjugation, and the other by a finite set of (three) gates which generate the group. In composite qudit dimensions, the former definition is used, while what the equivalent latter definition should be is an open question[3]—save the special case of the group of diagonal Cliffords, which was recently characterized in Ref. [FL24].

While we do not poke that bear of a question here, we do present an intermediate step from qudit circuit constructions emulating qubits, to qudit quantum error correcting codes with interesting logical operations on the qubit subspace. Bear in mind that this is preliminary work, but I hope that some of these ideas are interesting in their own right.

Ref. [CH17] found quadratic error suppression for combining magic state distillation and gate synthesis, as opposed to first the former then the latter.

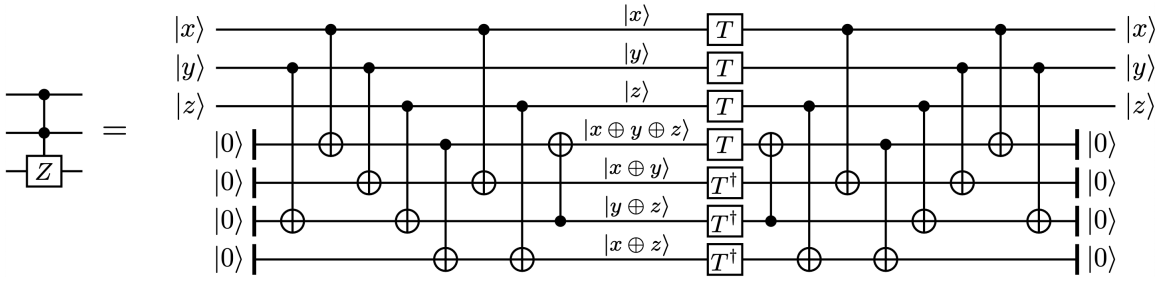


Figure 10.7: Figure for a T-depth one decomposition of the qubit CCZ gate, reprinted from Ref. [Sel13, Figure 1].

Building on this is the observation credited to Campbell [Cam16].

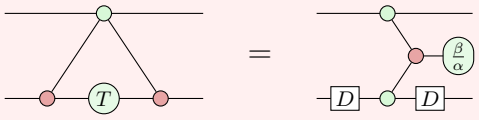
Observation 10.92. The above circuit consists of a CNOT circuit unitary embedding E, a layer of T and $T^\dagger$ gates, and $E^\dagger$. This therefore defines a $[\![7, 3, 1]\!]$ code; being distance 1, it has no error-correcting capabilities. To improve the code distance, it suffices to add one physical qubit, and a weight 8 X-type stabilizer generator connected to all 8 physical qubits. This construction preserves the transversal CCZ gate, resulting in the $[\![8, 3, 2]\!]$ code.

The plan for this project is to connect the dots between qudit circuit synthesis and error correcting codes with good logical operations for qubits.

We will work with qutrit phase gadgets. We opt to use the non-flexsymmetric qutrit ZX-calculus this time because it corresponds better to the Clifford CX operations in the circuit setting, where the directionality is always left to right.

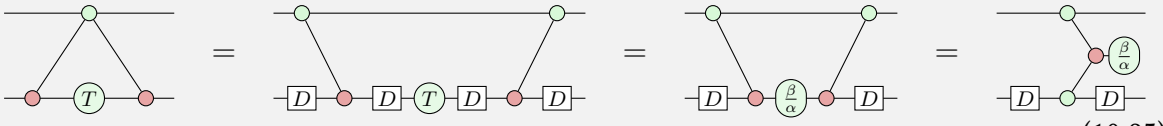
[3]Discussion with the authors of Refs. [WP24] and [PW24]

Lemma 10.93.



(10.94)

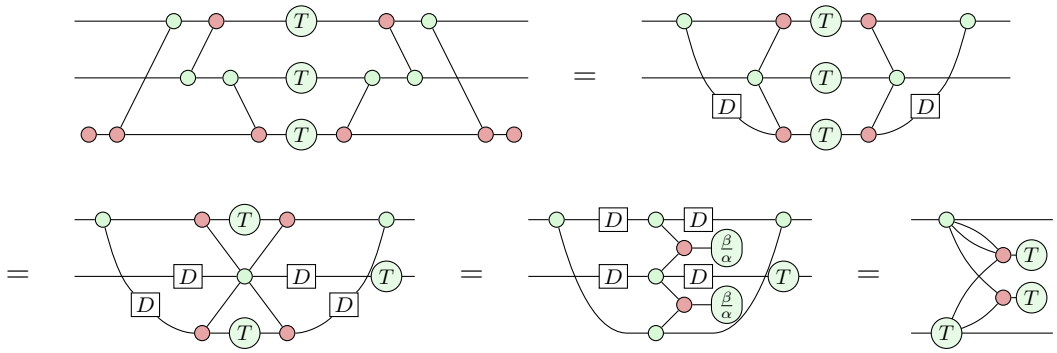
Proof.



(10.95)

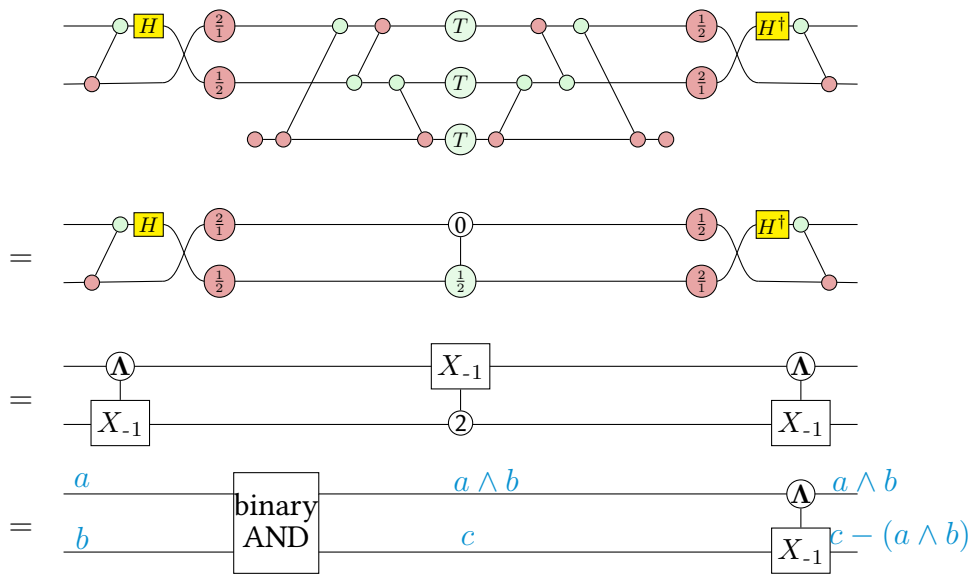
□

From the qutrit T-depth one circuit for the $|0\rangle$-controlled Z gate in Ref. [BRS17], we derive its phase gadget representation as $L = E; P; E^\dagger$.



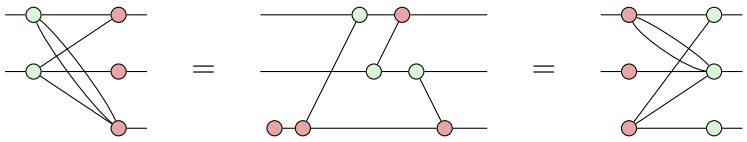
(10.96)

We found a Clifford unitary which we can bootstrap around it, to change the logical operation to a two-qutrit emulation of the binary AND gate. Observe that this matches the form in Ref. 10.60 up to an additional CX gate at the end, which only changes the garbage output trit c.



(10.97)

We can now ignore the outer logical unitary, and focus just on the phase-free ZX-diagram of the inner CSS code encoder. Writing it into Z-X and X-Z GPF.



(10.98)

This lets us read off the X-type rows of the complete stabilizer tableau:

$$\begin{aligned}
\bar{X}_1 &= X_1 X_3^2 \\
\bar{X}_2 &= X_1 X_2 X_3 \\
\bar{Z}_1 &= Z_1 Z_2^2 \\
\bar{Z}_2 &= Z_2 \\
S_1^Z &= Z_1 Z_2 Z_3
\end{aligned}$$

We immediately see that this is distance one code. This comes as no surprise because from the original unitary circuit, only one ancilla was added for it to be T-depth one, and hence only one stabilizer generator (Z-type) was added. In the absence of X-type stabilizer generators for Z errors to anticommute with, there is no protection against Z errors.

This sets up the problem. At the time of writing, we have succeeded at the next step: to first increase the code distance, while preserving the logical gate—up to disregarding the irrelevant throwaway output. We describe this in work with Christine Li, done since this thesis was written and revised, in the paper "Transversal AND in Quantum Codes" [LY26]. After that, the question is whether the code can at the same time, convert leakage errors from the qubit subspace to the $|2\rangle$ state to erasure errors, by observing and correcting them.

Chapter summary

In this chapter, we developed building blocks of fault-tolerant qutrit computation in the circuit model. We presented techniques for synthesizing multiple-controlled qutrit Toffolis, all controlled Clifford+T gates, controlled phase gates, arbitrary diagonal gates, and superconducting qutrit pulse sequences. Furthermore, we identified interesting connections between gate sets and number rings. These lead into the natural question of whether, like for qubits, the equivalence between a gate set and a number ring can be found for qutrits—and if so, what is the exact circuit synthesis algorithm to prove this. This question will be answered in the affirmative by the end of the next chapter. Before that, we will foray into two new qudit gate sets first introduced as part of this thesis work: Clifford+$\sqrt[d]{Z}$ and Toffoli+Hadamard.

Part V

Quditland

Chapter Contents

11 Scaling W state circuits in the Clifford hierarchy **193**

11.1 The $\sqrt[d]{Z}$ gate . . . 196

11.2 A fault-tolerant circuit for the $|0\rangle$-controlled prime-dimensional qudit X gate . 197

11.3 The d-qubit W state . . . 199

11.4 The controlled H gate with phase gadgets . . . 201

11.5 Building any size W states . . . 205

12 Qudit Toffoli+Hadamard and the ZH-calculus **211**

12.1 All odd prime-dimensional classical reversible gates can be built with just the $|0\rangle$-controlled X gate . . . 214

12.2 The qudit Toffoli+Hadamard gate set . . . 220

12.3 The qudit ZH-Calculus . . . 221

12.4 Translating ZH-diagrams to ZX-diagrams . . . 225

13 Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits **229**

13.1 Multiqutrit Exact Synthesis . . . 232

13.2 Gate Set Equivalences . . . 236

13.3 Rings and Groups . . . 240

13.4 Generators for $\mathrm{U}_n(\mathbb{Z}[1/3, \omega])$ . . . 243

13.5 Circuit Representations for the Elements of $\mathcal{S}_{3^n}$ . . . 245

13.6 Exact Synthesis of Toffoli+Hadamard Circuits . . . 250

13.7 Catalytic Embeddings . . . 251

13.8 Exact Synthesis of Clifford-Cyclotomic Circuits . . . 254

13.9 Circuit Complexity . . . 255

My interest in qudits (and predominantly in qutrits) came from a quantum computer architecture perspective. It is wasteful for systems which are physically qudits to pretend they are only two-dimensional. My view of this expanded as I learned about higher-dimensional quantum computation, and gained appreciation for many forms that quantum information can take beyond two-dimensional. In retrospect, I would say that I have continually been discovering how

vast an area of research finite-dimensional quantum information is. To me, qudits are a highly fundamental concept to this, as this knowledge can lay a foundation for people coming from qubit computation, and help with gaining intuition for understanding quantum computation with quantum systems which can also be fermionic, bosonic, spin, optical, continuous-variable, or combinations of these.

In this chapter, we broadly work towards a theory of quantum computation and quantum information for arbitrary finite dimension. Rather than having results for just qubits or qutrits, we can also prove a result for all prime dimensions or all finite dimensions at once. This furthermore ventures into mixed-dimensional quantum computation, which is closer to the true nature of quantum systems. A case can be made that more powerful graphical calculi make for more efficient and robust software, but at its core this is about a more accurate and expressive language for quantum computation. We pick up from where the previous chapter left off on building blocks of fault-tolerant qutrit computation in the circuit model. We start this chapter with work that is likely chronologically the first, of my work on qudit dimensions higher than three.

11

Scaling W state circuits in the Clifford hierarchy

This section is based on the paper "Scaling W State Circuits in the qudit Clifford Hierarchy" [Yeh23], which is about scalable constructions of qubit and qudit W states. This also serves as a good introduction to subsequent work in the rest of the chapter, as it features the qudit Clifford hierarchy, a new gate set, and the qudit ZX-calculus.

Understanding the structure of multipartite entanglement has long been of fundamental interest. The two multipartite entangled states most important in the literature are the GHZ state and the W state:

Definition 11.1. The N-qubit W state is

$$|W_N\rangle = \frac{1}{\sqrt{N}} \left(|10...0\rangle + |010...0\rangle + ... + |0...01\rangle\right). \tag{11.2}$$

Characterization of tripartite entangled states by Dür, Vidal, and Cirac found there to be only two distinct SLOCC-equivalence classes (equivalent up to stochastic local operations and classical communication): the GHZ-class and the W-class [DVC00]. As three-qubit states, GHZ states and the W states correspond respectively to a special and an anti-special Frobenius algebra, defining a compositional structure which can be used to compose arbitrarily large

multipartite qubit states from smaller ones [CK10].

In contrast to the GHZ state, the entanglement of the generalized W state is highly robust against decoherence and particle loss [ZFW10]. Robust and scalable multipartite entanglement is essential to many applications of quantum technology, including quantum communication, encryption, and distributed quantum computing. The W state in particular has been employed in protocols such as anonymous transmission in a quantum network [LMW18], photonic error detection [VLR20], quantum memory [LJW+20], quantum secret sharing [TYL19], and deterministic quantum communication [TH13]. A notable example is the classical distributed computing task of leader election, which is commonly prerequisite for graph-theoretic tasks such as computing minimum spanning trees. No known classical algorithm satisfies the same conditions achievable by quantum leader election algorithms for anonymous networks [Van14], for which (unitary transforms of) W states are not only sufficient, but necessary for [DP06].

A number of proposals exist in the literature for building larger (more than three qubits) W states. Some architectures are more suited for generating large W states directly, with proposed experimental schemes including single photon 8-qubit W state [HGNS15] and arbitrary size perfect W states (asymmetric as to be better suited to teleportation and superdense coding) [SSRP22]. One approach [GHP+14; Dik16] is analogous to reservoir sampling, where a series of different sized rotations distributes the entanglement uniformly with quadratic gate count, which could be made asymptotically linear with a rather involved binary-to-unary conversion [httb]. In another vein, two approaches to generate larger W states from smaller W states, are by fusing two smaller W states together [OMT+11; BYO13; LKY+16] for instance in a cavity QED architecture [ZYO+15; JSY17], or by expanding a W state with ancillae [LKY+16]. In addition, a technique compatible with the above methods is entanglement concentration protocols [BBPS96] which concentrate a less entangled W state on an arbitrary number of qubits into a maximally entangled W state [SPG+15].

Although these techniques can generate arbitrarily large (maximally entangled) W states in principle, their scalability has a practical limitation due to physical errors. These protocols were not intended for compatibility with quantum error correction codes and therefore are prone to propagation of errors, an issue that leads to their performance falling off with increasingly large system sizes. This can be especially concerning if the use case pertains to fairness (for instance quantum algorithms for voting [BH17]) or cryptography (for instance quantum key distribution [WTLY22]).

As a result, there is a need to investigate improved fault-tolerant protocols to prepare these highly entangled resource states. In this setting, the most probable errors on a subset of operations can be efficiently correctable. This transition necessitates a paradigm shift in the requirements and cost model that must be considered, to minimize non-correctable errors. To the best of our knowledge, there has been limited investigation into fault-tolerant W state

generation beyond a StackExchange post in 2018 by Gidney [htta].

In this work, we augment fault-tolerant protocols to construct large W states deterministically (i.e. with 100% success probability of the protocol in theory, independent of physical errors) in linear gate count and logarithmic depth. We do this by leveraging the higher levels of qudits as a computational resource, thus removing the need for ancillae and post-selection for constructing certain size W states.

Compared to the qubit case, the entanglement properties of qudits are less understood. Experimentally, qudit entanglement has demonstrated improved noise robustness compared to qubits [**SrivastavV2022noiserobustquditsteering**], and leveraging qutrit states can improve fidelity of qubit operations [BKL+19; BDR+22]. With regards to theory, although the W class has been generalized to qudits with similar properties of monogamy relations [KS08], entanglement classification of multiple qudit states is a hard task made more difficult by the existence of a form of entanglement that does not exist for qubits [PH23].

Definition 11.3. The N-qudit W state is often defined as [KS08]

$$|W_N\rangle = \frac{1}{\sqrt{(d-1)N}} \sum_{j=1}^{d-1} \left(|j0...0\rangle + |0j0...0\rangle + ... + |0...0j\rangle\right). \tag{11.4}$$

We focus on this definition. Other generalizations of the W state have been studied, in particular qudit Dicke state construction [Nep23].

The gate set this work is set in is what we call the $\sqrt[d]{Z}$ (can also be denoted $Z^{(1/d)}$ or dZ) gate. This is a family of gates, one for each prime qudit dimension d, in the d^{th} level of the Clifford hierarchy. For $d = 3$ this coincides with the qutrit T gate. Indeed, we show that this can be seen as an alternate generalization of the qutrit T gate to all odd prime dimensions, to the characterization of single-prime-dimensional qudit diagonal gates in the third level of the Clifford hierarchy by Howard and Vala [HV12].

Despite the open question of the practicality of fault-tolerant implementation of the $\sqrt[d]{Z}$ gate, we claim that it is important to study it given that it enables certain logical constructions of computational importance. A number of essential constructions in the qubit setting have no known implementation applicable in arbitrary prime dimension. In this work we focus on constructing W states considering the asymptotic gate count of a specific non-Clifford gate. In forthcoming work we will demonstrate that this gate set enables exact synthesis of any classical reversible logic gate (including any prime-dimensional qudit multiple-controlled Toffoli). In prior work with van de Wetering, we presented circuit constructions in qutrit Clifford+T for not only any ternary classical reversible gate, but also any multiple-controlled qutrit Clifford+T gate [YvdW22].

This can also facilitate future decompositions in other gate sets by establishing a pathway to realizing these computationally useful gates, through showing that direct implementation of the $\sqrt[d]{Z}$ gate is a sufficient condition. We note that proving whether a gate is impossible to exactly synthesize in a certain gate set is difficult, and is oftentimes preceded by the generally difficult problem of full characterization of all gates exactly synthesizable in that gate set.

11.1 The $\sqrt[d]{Z}$ gate

Refer to Chapter 5 for the relevant background on prime-dimensional qudits: the Clifford gates, diagonal gates in the Clifford hierarchy, and definitions of the prime-dimensional qudit T gate as a single-prime-dimensional qudit diagonal gate in the third level of the Clifford hierarchy.

In this text we investigate a different class of gates, by a different generalization of the qutrit T gate to all odd prime dimensions. Explored here are non-Clifford gates which can be used to construct important logical gates and resource states for which no fault-tolerant exact synthesis is presently known. We apply these gates to constructing qubit and prime-dimensional qudit generalized W states, for which, for most sizes of the W state, no construction which is both deterministic and potentially fault-tolerant was previously known.

We now introduce the sole single-qudit non-Clifford gate we will need for our constructions. To the best of our knowledge, this gate has not been studied in the context of gate synthesis before for $d > 3$. Let $\zeta := e^{\frac{2\pi i}{d^2}} = \sqrt[d]{\omega}$ denote the $d^{2^{\text{th}}}$ root of unity. We denote $\sqrt[d]{Z} = U_{2,1}$ in Equation (5.9) to be the gate acting as

$$|k\rangle \mapsto \zeta^k |k\rangle \tag{11.5}$$

for k in $\mathbb{Z}_d$. Indeed $(\sqrt[d]{Z})^d = Z$, justifying it being a d^{th} root of Z gate, albeit not the only gate to satisfy this condition.

According to Equation (5.9), the $\sqrt[d]{Z}$ gate is in the d^{th} level of the Clifford hierarchy. This implies that for qubits it is the Clifford $S = \text{diag}(1, i)$ gate. For qutrits it is $\text{diag}(1, \omega^{1/3}, \omega^{2/3}) = U_{2,1}$ in Equation (5.9), joining the other definitions of the qutrit T gate in the literature which are all Clifford equivalent to one another.

For all other d, this gate is above the third level of the Clifford hierarchy. As a single-qudit diagonal gate in the Clifford hierarchy, it admits an efficient fault-tolerant implementation [dSil21] by the recursive state injection protocol described for single-qubit diagonal gates in the Clifford hierarchy in Refs. [ZLC00].

Another gate we will refer to is the single-qudit gate $P_1(k)$:

Definition 11.6. In the $(d-1)^{\text{th}}$ level of the Clifford hierarchy [CGK17]:

$$P_1(k) := \sum_{j=0, j\neq k}^{d-1} |j\rangle\langle j| + \omega |k\rangle\langle k| \tag{11.7}$$

Next, in addition to the CX gate, we will need two more controlled gates in this text: the $|0\rangle$-controlled X gate and the controlled H gate.

11.2 A fault-tolerant circuit for the $|0\rangle$-controlled prime-dimensional qudit X gate

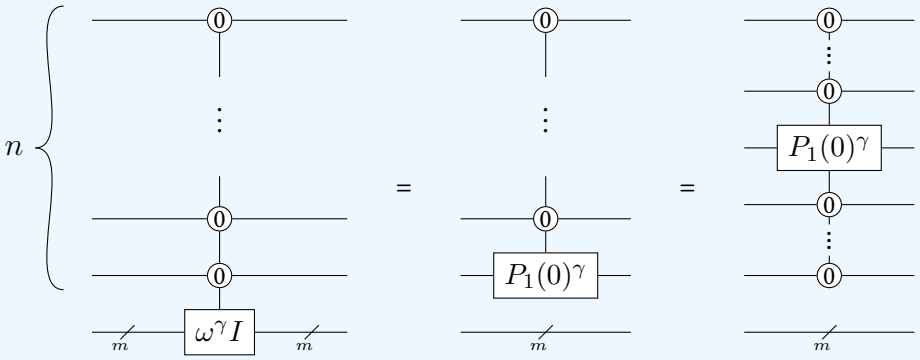
In retrospect, the next circuit construction turned out to be highly useful. We use it extensively in the next section, where after defining the qudit ZH-calculus, we can translate any ZH-diagram to ZX-calculus. Crucially, we can finally connect the dots between the Clifford group in any odd prime dimension d, and d-ary classical reversible circuits plus the Hadamard gate.

The $|0\rangle$-controlled X gate acts on the target as

$$\begin{cases} X & \text{if control is } |0\rangle \\ I & \text{if control is } |k\rangle\,,\ k \in \mathbb{Z}_d, k \neq 0 \end{cases} \tag{11.8}$$

Notably, it is non-Clifford except in the qubit case. Before we present its construction, we define *controlled global phases* [YvdW22]. A complication when trying to construct controlled unitaries, is that usually irrelevant global phases becomes 'local' and hence must be dealt with accordingly [BBC+95, Lemma 5.2].

Definition 11.9. A *controlled global phase gate* is a controlled unitary where the unitary is $e^{i\phi}\mathbb{I}$, for identity matrix I and phase ϕ:



(11.10)

The number of qudits I acts on is irrelevant as the phase factor can be "factored out". Here we substituted the global phase by γ where $\phi = \gamma \cdot \frac{2\pi}{d}$ so that the phases are powers of ω.

Our circuit decomposition of the $|0\rangle$-controlled X gate will be off by a certain controlled global phase, which we later calculate in Equation (11.20) to be $\omega^{(d-1)/2}$. If it is desired to correct this controlled global phase, we want to implement a diagonal gate CGP which (up to a global phase we can ignore) acts as

$$|0\rangle \mapsto \omega^{(d-1)/2} |0\rangle \,, \; |k\rangle \mapsto |k\rangle \,, \; k \in \mathbb{Z}_d, k \neq 0. \tag{11.11}$$

Lemma 11.12. *For any odd prime qudit dimension d, a two-qudit circuit can be constructed to correct a controlled global phase of $\omega^{(d-1)/2}$ on the target when the control is $|0\rangle$, with $(d-1)$ $\sqrt[d]{Z}$ gates.*

Proof. We implement this by applying $(d-1)/2$ times the gate $P_1(k)$ from Definition 11.6 where $k = 0$. Up to a global phase of ζ, for $k \in \mathbb{Z}_d$, the $P_1(k)$ gates are also exactly synthesizable in the Clifford+$\sqrt[d]{Z}$ gate set by conjugating $\sqrt[d]{Z}$ by Pauli X:

$$P_1(k) = \zeta \; X^k \, \sqrt[d]{Z}^\dagger \, X \, \sqrt[d]{Z} \, X^{\dagger(k+1)} \tag{11.13}$$

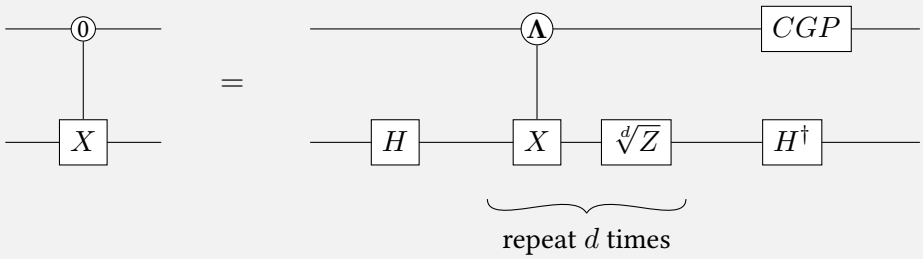
This follows due to $X \sqrt[d]{Z} X^\dagger = \text{diag}(\zeta^{d-1}, 1, \zeta, \zeta^2, ..., \zeta^{d-2})$, and $\sqrt[d]{Z}^\dagger = \text{diag}(1, \zeta^{-1}, \zeta^{-2}, ..., \zeta^{-(d-1)})$. For all other $k \in \mathbb{Z}_d$, conjugate by X k times. □

Remark 11.14. As $P_1(0)$ is one level lower in the Clifford hierarchy than $\sqrt[d]{Z}$, it could be sensible to add it to the Clifford+$\sqrt[d]{Z}$ gate set, instead of decomposing it into $\sqrt[d]{Z}$ gates.

The following proposition essentially generalizes the construction for the $|2\rangle$-controlled qutrit Z gate by Bocharov, Cui, Roetteler, and Svore [BCRS16] to arbitrary prime qudit dimension.

Proposition 11.15. *The $|0\rangle$-controlled X gate can be constructed with $\sqrt[d]{Z}$-count d (up to a controlled global phase that is a power of ω), or with $\sqrt[d]{Z}$-count $2d-1$ (exact implementation).*

Proof.



(11.16)

To derive that the above circuit implements the $|0\rangle$-controlled X gate, start by considering only the action of the subcircuit which is repeated d times. Observe that when the control

and target are Z-basis states $|x\rangle$ and $|y\rangle$ for $x, y \in \mathbb{Z}_d$, at the time slice after the first CX gate the state is $|x, (x+y) \bmod d\rangle$. The $\sqrt[d]{Z}$ gate on the target then contributes a factor of $\zeta^{(x+y) \bmod d}$. The $\sqrt[d]{Z}$ gate after that contributes a factor of $\zeta^{(2x+y) \bmod d}$, and so on. Repetition of the CX and the $\sqrt[d]{Z}$ gate d times computes to the diagonal gate

$$|x, y\rangle \mapsto \prod_{j=0}^{d-1} \zeta^{(jx+y) \bmod d} |x, y\rangle . \tag{11.17}$$

This can be broken down into two cases: $x = 0$ and $x \neq 0$. When $x = 0$, the output of Equation (11.17) is

$$\zeta^{dy} |x, y\rangle . \tag{11.18}$$

As $\zeta^d = \omega$, this is simply the Z gate on the target prime-dimensional qudit (except for qubits where this is the S gate).

When $x \neq 0$, the output is

$$\zeta^{\sum_{j=0}^{d-1} (jx+y) \bmod d} |x, y\rangle . \tag{11.19}$$

Note that in the sum, $(jx + y) \bmod d$ takes on the value of each element in $\mathbb{Z}_d$ exactly once. Therefore, when $x \neq 0$, the output is

$$\zeta^{\sum_{j=0}^{d-1} j} |x, y\rangle = \zeta^{d(d-1)/2} |x, y\rangle . \tag{11.20}$$

As $\zeta^d = \omega$, this amounts to a controlled global phase of $\omega^{(d-1)/2}$ which applies iff $x \neq 0$, labelled CGP in Equation (11.16). This can be either corrected by applying Lemma 11.12, or omitted in situations where this phase discrepancy is tolerated to improve efficiency.

Finally, we conjugate the target qudit by H to obtain the $|0\rangle$-controlled X gate (up to a global phase that we can safely ignore). □

For $d = 3$ CGP is Clifford, so the $|0\rangle$-controlled X gate has T-count 3 [BCRS16]. For $d = 2$, while $(d-1)/2$ is not an integer, the construction remains valid by substituting $CGP = Z$.

Corollary 11.21. *For $k, p \in \mathbb{Z}_d$, conjugating the control qudit by X^k yields the $|k\rangle$-controlled X gate. From this, the $|k\rangle$-controlled X^p gate can be built naïvely by repeating the construction p times.*

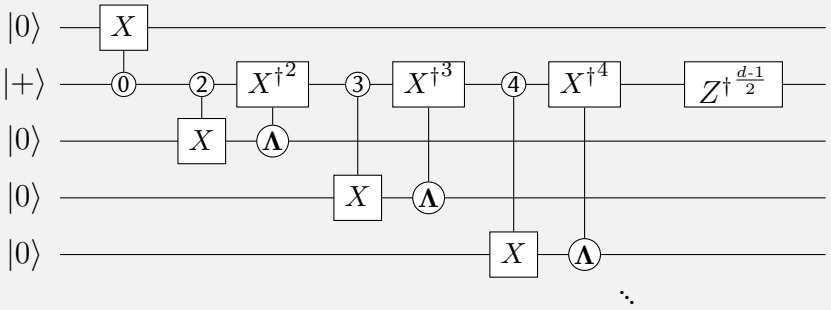
11.3 The d-qubit W state

We now introduce our first W state construction.

Proposition 11.22. *For any prime qudit dimension d, the d-qubit W state can be constructed deterministically in the Clifford+$\sqrt[d]{Z}$ gate set using $d^2 - d$ $\sqrt[d]{Z}$ gates.*

Proof.

$|W_d\rangle$ =



(11.23)

Correctness of the circuit is straightforward to verify by observing that the d Z-basis states which make up the $|+\rangle$ state in equal superposition each contribute to one of the d terms of the output W state. Consider the unitary circuit being applied to the state

$$|0\rangle |+\rangle |0\rangle^{\otimes(d-2)} = \frac{1}{\sqrt{d}} \sum_{j=0}^{d-1} |0\rangle |j\rangle |0\rangle^{\otimes(d-2)} . \tag{11.24}$$

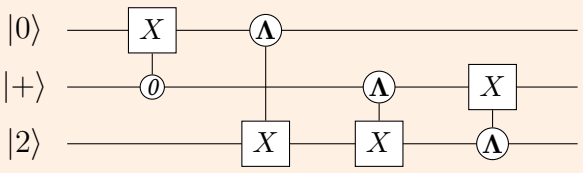
Observe that each subcircuit iterating over $k \in \mathbb{Z}_d$ where $k \geq 2$, consisting of the $|k\rangle$-controlled X gate followed by k $CX^\dagger$ gates, acts as

$$|0\rangle |k\rangle |0\rangle^{\otimes(d-2)} \mapsto |0\rangle^{\otimes k} |1\rangle |0\rangle^{\otimes(d-k-1)} \tag{11.25}$$

while being identity for all $j \neq k$ on $|0\rangle |j\rangle |0\rangle^{\otimes(d-2)}$ and on $|0\rangle^{\otimes j} |1\rangle |0\rangle^{\otimes(d-j-1)}$. For $d(d-1)$ $\sqrt[d]{Z}$ count, delay all controlled global phase corrections until applying $Z^\dagger \frac{d-1}{2}$ times at the end. □

Corollary 11.26. *As the qutrit CGP is Clifford, the three-qubit W state can be implemented with qutrit T-count 3:*

$|W_3\rangle$ =



(11.27)

This achieves a new minimum single-prime-dimensional qudit non-Clifford gate count to synthesize the W state, and moreover deterministically.

Remark 11.28. When $d = 2$, the construction in Proposition 11.22 does indeed produce $\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. However, as this is equivalent to the Bell state up to an X gate, this does not possess the usefulness of a W state of three or more qubits.

11.4 The controlled H gate

with phase gadgets

In the previous section, we showed how to construct the generalized qubit W state of any prime size. However, the scaling of $O(N^2)$ $\sqrt[d]{Z}$ gate count implies that this approach is less asymptotically efficient for large d. In this section, in order to improve this to asymptotically linear gate count, we present how to construct a controlled prime-dimensional qudit H gate, up to a controlled global phase not of practical consequence in this work. We do this by extending our controlled H for qutrits in Ref. [YvdW22] to any prime qudit dimension.

Lemma 11.29. *Up to a global phase, in any prime qudit dimension, the H gate can be decomposed into Clifford phase gates.*

Proof. For $d = 2$, this is the well-known decomposition of $H = R_z(\frac{\pi}{2})R_x(\frac{\pi}{2})R_z(\frac{\pi}{2})$.

For all odd prime qudit dimensions, let $2^{-1} \bmod d$ denote the multiplicative inverse of 2 in $\mathbb{Z}_d$. The qudit H gate can be decomposed into a Z rotation followed by an X rotation followed by a Z rotation as [Wan22; NW19]:

$$H = i^{(d-1)/2} \quad Z(\vec{\tau})X(\vec{\tau})Z(\vec{\tau}) \tag{11.30}$$

$$= i^{(d-1)/2}\ S\, Z^{(2^{-1} \bmod d)}\ H^\dagger\, S\, Z^{(2^{-1} \bmod d)}\ H\ S\, Z^{(2^{-1} \bmod d)} \tag{11.31}$$

where for k from 1 to $d-1$,

$$\tau(k) := 2^{-1}k^2 \bmod d, \tag{11.32}$$

$$Z(\vec{v}) := |0\rangle\langle 0| + \sum_{k=1}^{d-1} \omega^{v(k)} |k\rangle\langle k|, \tag{11.33}$$

and

$$X(\vec{v}) = H^\dagger Z(\vec{v}) H. \tag{11.34}$$

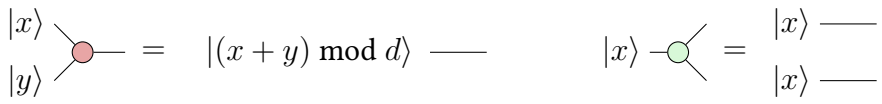
□

Our approach to controlling the H gate is to control each component Z and X phase gate in Equation (11.30). We apply strategies from past work: generalizing controlled phase gates from qubits to qutrits [vdWY23] and controlling the qutrit H gate [GRvdWY22b]. In future work we will consider generalizing to higher dimensions, adding any number of controls to qutrit

gates [YvdW22].

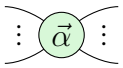
The key structural insight leverages the generalization to arbitrary dimensions of *phase gadgets*, a symmetric multi-qudit interaction which applies a phase dependent on the sum of all computational basis states modulo d. Phase gadgets have been characterized thus far in two graphical calculi: in the qubit ZX-calculus [CD11] in Ref. [KvdW20b; CDD+20], and in the qutrit ZX-calculus [WB14] in Ref. [vdWY23]. We remark that phase gadgets are broadly applicable to all qudit dimensions as a graphical construct to relate diagonal gates in the circuit model and polynomial functions.

Hence, this gives us an idea of what sort of circuit structure may be a good place to start. Keeping minimal on introducing formalism of the qudit ZX-calculus [Ran14; WB14; Wan22; IC22], we introduce non-unitary circuit elements which respectively act as addition modulo d and copying of basis states, labelled $|x\rangle$ and $|y\rangle$ below:



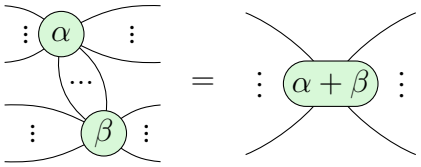
(11.35)

More generally, these are called spiders. They can have any number m and n of input and output wires, and can be labelled by a phase vector $\vec{\alpha}$ of length $d-1$. By convention, when $\vec{\alpha}$ is the zero vector, the label is omitted.



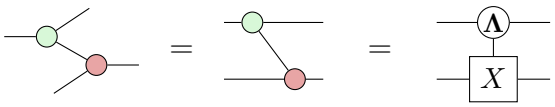
$$:= |0\rangle^{\otimes n} \langle 0|^{\otimes m} + \sum_{j=1}^{d-1} \omega^{\alpha(j)} |j\rangle^{\otimes n} \langle j|^{\otimes m} \tag{11.36}$$

The X spider is defined likewise, substituting X-basis states for the Z-basis states here. The Z and X spiders satisfy the fusion rule between same color spiders:

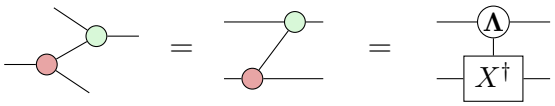


(11.37)

The CX gate is then representable as the composition of the two operations in Equation (11.35), whereas its inverse $CX^\dagger$ is its diagram flipped horizontally.

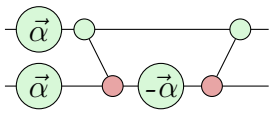


(11.38)



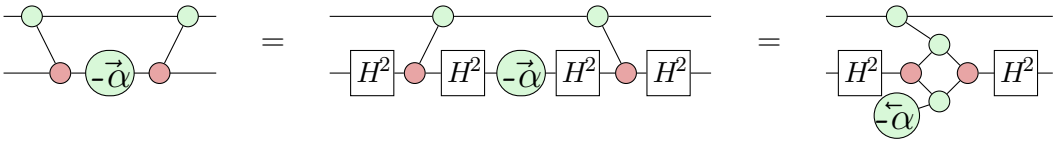
(11.39)

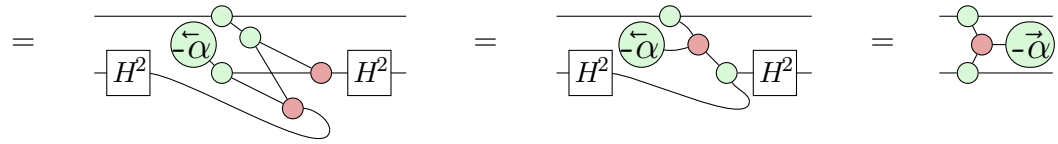
Generalizing the two-qubit circuit for any controlled rotation, to (certain controlled rotations in) any qudit dimension:



(11.40)

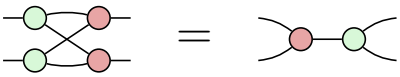
It has a *phase gadget*, a diagonal, symmetric two-qudit interaction:





(11.41)

where above we applied the qudit bialgebra rule:

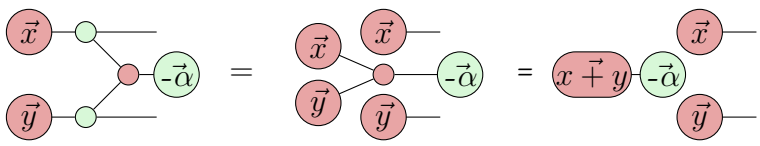


(11.42)

For qubits, any diagonal qubit unitary can be expressed as a product of phase gadgets by writing the unitary as a phase polynomial [Amy19; GD20], as it serves as a good basis for optimising quantum circuits [CDD+20; CSD20; dBBW20c; dBBW20b; vdWet21; BMdF+21]. It may thus inform how to optimize qudit computation, building upon the work for qutrits in Ref. [TM22].

There are several different ways to define a two-qudit phase gadget. One way is to consider it as the diagonal gate $|x, y\rangle \mapsto \omega^{\alpha(x \oplus y)} |x, y\rangle$. This applies a phase of $\omega^{\alpha(x \oplus y)}$ when $x \oplus y = 1$. Here $\oplus$ is the XOR operation, which is the addition on $\mathbb{Z}_d$.

Inputting to Equation (11.41) Z-basis states $|x, y\rangle$ (X spiders with labels $\vec{x} = [(d-1)x, ..., x]$ and $\vec{y} = [(d-1)y, ..., y]$) evaluates to:



(11.43)

Next, we interpret each spider in the rightmost diagram as its corresponding linear map: the identity on Z-basis states accompanied by a 'floating scalar' expression that evaluates to $\omega^{-\alpha(x+y \bmod d)}$. Hence this phase gadget indeed implements the operation we want. Therefore, we see that these three ways to define a qubit phase gadget—via the action, via the circuit, or

via the diagrammatic representation—are equal in all finite qudit dimensions.

For our first controlled H gate construction, we seek to implement a controlled phase gate such that the phase gate $\vec{\tau}$ applies to the target when $x = 1$, and identity applies to the target when $x = 0$. We can thus solve the below system of equations for $\vec{\alpha}$ to find when the circuit of Equation (11.40) implements $\sqrt[(2^{-1} \bmod d)]{Z(\vec{\tau})}$:

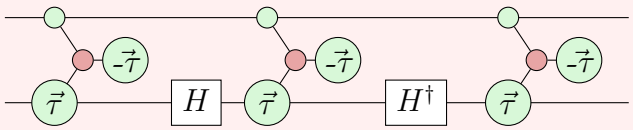
$$\begin{cases} \alpha(x) + \alpha(y) - \alpha(x + y \bmod d) = 0 & \text{if } x = 0 \\ \alpha(x) + \alpha(y) - \alpha(x + y \bmod d) = y^2 \bmod d & \text{if } x = 1 \end{cases} \tag{11.44}$$

Setting $\alpha(0) = 0$ and solving for the remaining elements of $\vec{\alpha}$:

$$\alpha(k) = -\sum_{j=0}^{k-1} j^2 \bmod d \tag{11.45}$$

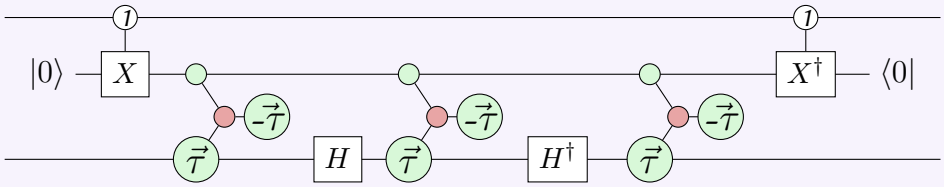
To implement $Z(\vec{\alpha})$, as a diagonal gate comprised solely of integer powers of ω, we can always construct it from $P_1(k)$ gates decomposed in Equation (11.13). Then we can build controlled H, as applying the $Z(\vec{\alpha})$ gate $2^{-1} \bmod d$ times gives $Z(\vec{\tau})$. Since $\alpha(1) = 0$, we can omit the $Z(\vec{\tau})$ gate on the control of the phase gadget:

Lemma 11.46. *The following circuit performs identity on the target when the control is $|0\rangle$, and the H gate up to a controlled global phase of $i^{(d-1)/2}$ when the control is $|1\rangle$:*



(11.47)

Proposition 11.48. *With a clean and uncomputed ancilla, we can adapt the construction of Lemma 11.46 to implement the $|1\rangle$-controlled H gate up to a controlled global phase:*



(11.49)

Remark 11.50. For all W states in this text, we conserve non-Clifford gate count by not correcting this controlled global phase; these controlled global phases collectively incur only a global phase for the final W state. Moreover, whenever the control qubit is known to be in the $\{|0\rangle, |1\rangle\}$ subspace, for instance in Equation (11.53), the ancilla-free implementation in Equation (11.47) suffices.

11.5 Building any size W states

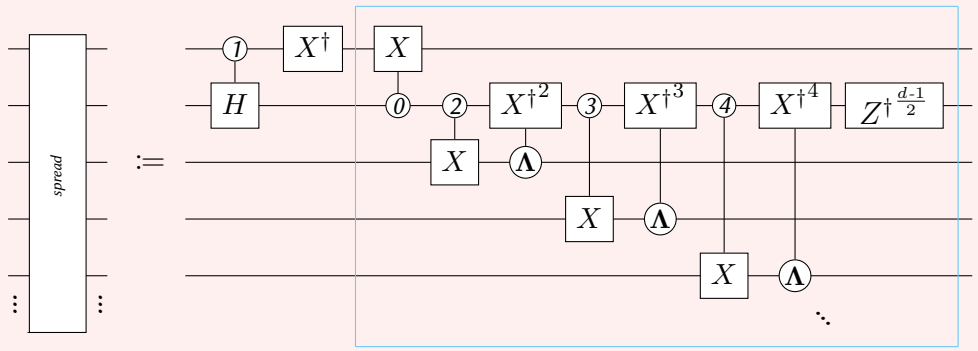
Prime power size W states

The goal is to now scale the size of the generalized qubit W state W_{d^n} synthesized using qudit gates, to any prime power size. This is accomplished by implementing a gate which has the action

$$|0...0\rangle \mapsto e^{i\theta} |0...0\rangle \quad \text{and} \quad |10...0\rangle \mapsto e^{i\phi} W_{d^n} \tag{11.51}$$

where $e^{i\theta}$ and $e^{i\phi}$ are any global phases.

Lemma 11.52. *The spread gate, a d-qudit gate which fulfills the criteria of Equation (11.51), is implementable by the circuit*



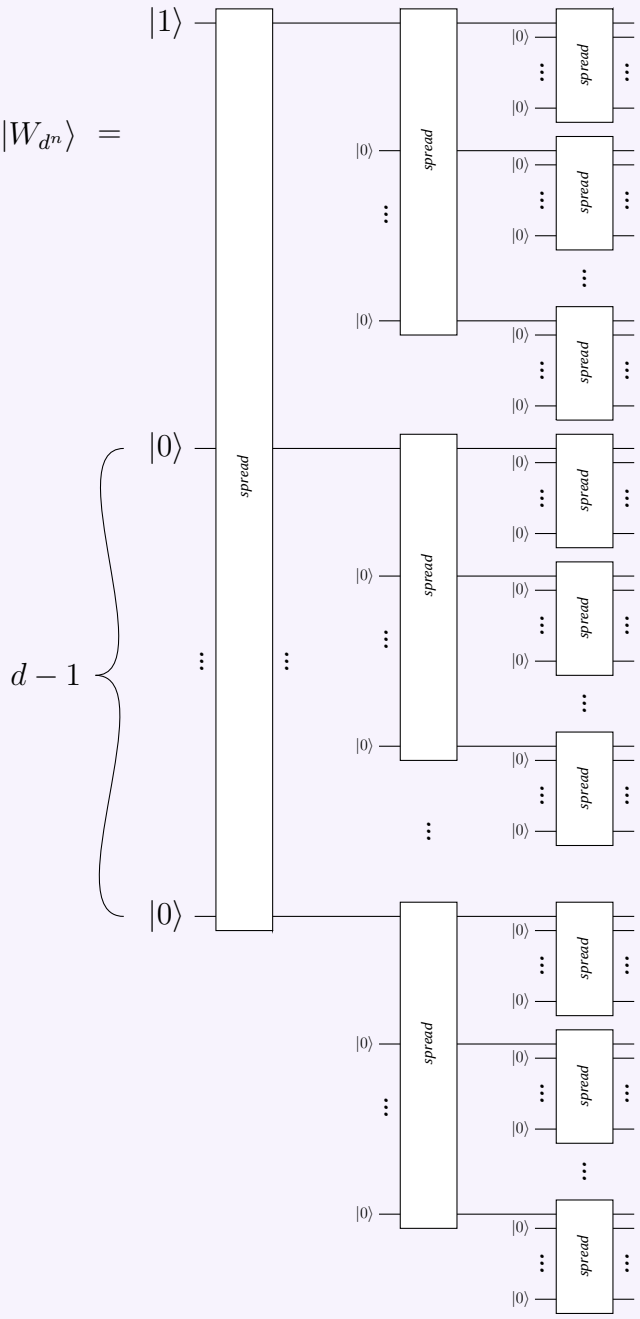
(11.53)

Note that the blue box contains the circuit in Equation (11.23).

Proposition 11.54. *By recursively applying the spread gate to each output prime-dimensional qudit tensored with $d-1$ $|0\rangle$ prime-dimensional qudits, for a total of n layers, the entanglement can be uniformly spread to build the $N = d^n$ size W state up to a global phase with $O(N)$ $\sqrt[d]{Z}$*

count.

$$|W_{d^n}\rangle \;=$$



(11.55)

Substitute the first spread gate by Equation (11.23) *to reduce gate count.*

Proof. The i^{th} layer of spread gates consists of d^{i-1} spread gates. Therefore, the total number of spread gates is

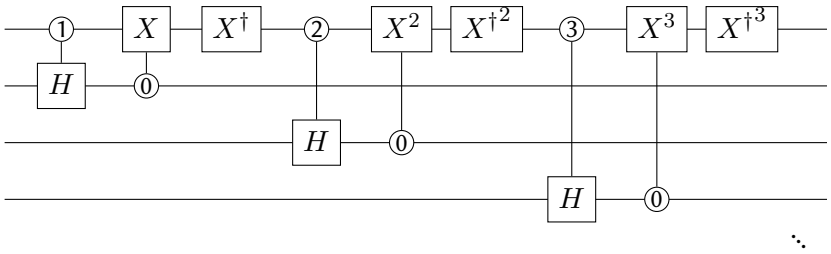
$$\sum_{j=0}^{n-1} d^j = \frac{d^n - 1}{d - 1} = O(N) \tag{11.56}$$

As each spread gate has gate count independent of N, the total $\sqrt[d]{Z}$ count is $O(N)$. □

Remark 11.57. The qubit version of the above method was discovered, albeit not well-known, in 2018 in a StackExchange post by Gidney [htta]. It achieves a T-count of $2N-4$, establishing an upper bound for the minimum number of T gates needed to implement the W state. Since then, the asymptotic lower bound has been found to match the upper bound: based on a reasonable complexity-theoretic conjecture, Arunachalam, Bravyi, Nirkhe, and O'Gorman found that any qubit Clifford+T circuit building an N-qubit W-state must contain at least a number of T gates linear in N [ABNO22]. It would be interesting to investigate the non-Clifford resource requirements of the generalized qubit and qudit W states in the prime-dimensional qudit setting.

Prime power size qudit W states

Furthermore, we generalize the constructions thus far from *qubit* W states to *qudit* W states. Consider this modification of the spread circuit in Equation (11.53):



(11.58)

For inputs of the form $|k0\ldots0\rangle$ for $k \in \mathbb{Z}_d$, up to phases we can safely ignore, it has action

$$\begin{cases} |k0\ldots0\rangle \mapsto |k0\ldots0\rangle & \text{if } k=0 \\ |k0\ldots0\rangle \mapsto \frac{1}{\sqrt{d}}\left(\sum_{j=1}^{d-1} |0\rangle^{\otimes k} |j\rangle |0\rangle^{\otimes(d-k-1)} + |k0\ldots0\rangle\right) & \text{if } k\neq 0 \end{cases} \tag{11.59}$$

The d-qudit W state is achieved by inputting to the top qudit of the above circuit the resource state

$$\frac{1}{\sqrt{d-1}}\sum_{j=1}^{d-1}|j\rangle \quad = \quad |+\rangle \text{ — } 0 \text{ —}$$
$$|0\rangle \text{ — } X \text{ — } \langle 0| \tag{11.60}$$

and inputting $|0\rangle$ for all other qudits.

Moreover, as we did for qubit W states, we can likewise exponentially magnify the size of this qudit W state. This is because the above circuit maps, for some irrelevant global phases $e^{i\theta}$ and $e^{i\phi}$,

$$|0\ldots0\rangle \mapsto e^{i\theta}|0\ldots0\rangle \quad \text{and} \quad \left(\frac{1}{\sqrt{d-1}}\sum_{j=1}^{d-1}|j\rangle\right)|0\ldots0\rangle \mapsto e^{i\phi}W_{d^n} \tag{11.61}$$

Therefore,

Proposition 11.62. *We can build the size $N = d^n$ qudit (of prime dimension d) W state by inputting $\left(\frac{1}{\sqrt{d-1}}\sum_{j=1}^{d-1}|j\rangle\right)|0...0\rangle$ into n layers of the circuit in Proposition 11.54, where the qudit W state spread gate gate is given in Equation (11.58). This has $O(N)$ $\sqrt[d]{Z}$ count because the $\sqrt[d]{Z}$ count for each of the $O(N)$ spread gates is independent of N for fixed d. This requires no further post-selection other than the initial copy of the above resource state.*

Building any size W states

In the fault-tolerant setting, protocols to generate large W states should be both scalable, robust against errors, and deterministic. The above protocols as-is generate only prime power size W states. We now present two ways to construct any size qubit W state.

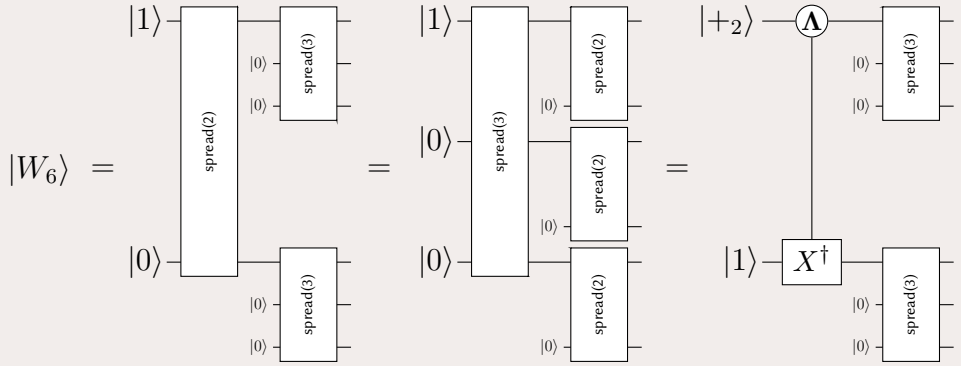
By switching dimensions mid-computation

Here, we modify earlier constructions to deterministically build any composite (i.e. non-prime) size N-qubit W states. This is by generalizing Proposition 11.54 to allow spread gates of differing qudit dimensions.

Proposition 11.63. *For any composite number N with prime factorization $N = N_1 \times N_2 \times ... \times N_f$ (which is unique up to permutation), a deterministic circuit to build the N-qubit W state is the circuit of Proposition 11.54, where the i^{th} layer of spread gates is of qudit dimension N_i. This has $O(N)$ non-Clifford gate count because the $\sqrt[N_i]{Z}$ count for each of the $O(N)$ spread(N_i) gates is independent of N.*

Remark 11.64. Even if for any prime qudit dimension d the Clifford+$\sqrt[d]{Z}$ gate set were efficiently implementable fault-tolerantly, for the circuit in Proposition 11.63 to be efficiently implementable fault-tolerantly, some mechanism to compose different qudit dimension states or gates is necessary. One solution would be to find a *code switching* protocol [Kub18] which can switch mid-computation from the quantum error correcting code in use to another of a different qudit dimension. A framework to consider for computations involving differing qudit dimensions is the *qufinite ZX-calculus* [Wan22].

Example 11.65. Three deterministic 6-qubit W state circuits:



(11.66)

The rightmost circuit above is moreover fault-tolerant: distill the qutrit magic state $|+_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ (a qubit X-basis state) in the 5-qutrit stabilizer code [ACB12b; DH15] or the 11-qutrit Golay code [Pra20], then do in a qutrit stabilizer code $\mathrm{CX}^\dagger$ and two spread(3) gates.

By post-selection of the W state

To attain other size N-qubit W states via post-selection, necessitates $d^{\lceil \log_d N \rceil} - N$ qubits to be successfully post-selected on the $|0\rangle$ state. If this fails, this protocol must be repeated until success; the expected number of attempts is

$$\left(\prod_{j=N+1}^{d^{\lceil \log_d N \rceil}} \left(1 - \frac{1}{j}\right)\right) \sum_{h=0}^{\infty} (h+1) \left(1 - \prod_{k=N+1}^{d^{\lceil \log_d N \rceil}} \left(1 - \frac{1}{k}\right)\right)^h = \frac{d^{\lceil \log_d N \rceil}}{N}. \tag{11.67}$$

This is a geometric series of the form $(1-r)\sum_{h=0}^{\infty}(h+1)r^h = \frac{1}{1-r}$, where $1-r$ is a recurrence relation that solves to $\frac{N}{d^{\lceil \log_d N \rceil}}$. Hence for large N, the expected number of attempts is at most d. In other words, the expected gate count to create an N-qubit W state is at most a factor of d greater than the optimal scenario where the post-selection succeeds on the first try. Thus, the expected gate count remains linear in N.

Considerations for network topologies

The protocols in this work only use one- and two-qudit gates. It is easy to see that, in order to perform a two-qudit gate on two qudits which have a path length of b nodes apart in the connectivity graph, an overhead of $O(b)$ swaps per two-qudit gate suffices to perform a two-qudit gate between them. Therefore, the gate count of the protocol adapted to any quantum network topology is $O(bN)$.

Moreover, within each spread gate, the two-qudit gates you do are always less than d distance apart. Therefore, assuming nearest-neighbor connectivity in the network of each node (i.e. each logical qudit) participating in the protocol, the overhead due to routing can be bounded

by a factor of $O(d)$. Thus, the best possible performance of these protocols is when the network topology graph contains as a subgraph, the tree whose shape is that of the circuit in Equation (11.55).

12

Qudit Toffoli+Hadamard and the ZH-calculus

Remark 12.1. This section is based on parts of the paper "The Qudit ZH-Calculus: Generalised Toffoli+Hadamard and Universality" [RvdWY23], in collaboration with Patrick Roy and John van de Wetering. In addition to introducing a qudit ZH-calculus, this paper contains two types of universality results. One, which I will not discuss as to not overlap with Patrick Roy's Master's thesis [Roy22] aside from the section here defining the calculus, is that for prime dimensions d, the phase-free qudit ZH-calculus is universal for matrices over the ring $\mathbb{Z}[e^{i\frac{2\pi}{d}}]$. This section is about the other type of universality result, defining a Toffoli+Hadamard gate set for any prime qudit dimension and justifying why it is natural to define it as *the* Toffoli+Hadamard gate set. This section moreover shows that the qudit ZH-calculus corresponds precisely to Toffoli+Hadamard circuits with ancillae and post-selections.

Remark 12.2. Regarding conventions, all gate definitions in this chapter match that of the background on qudits in Chapter 5. Note that results in this section will state whether they hold for any prime dimension, any odd prime dimension, or any odd dimension.

This is the section where we will finally connect the dots between the Clifford group in any odd prime dimension d, and d-ary classical reversible circuits plus the Hadamard gate. What we will show is that unlike for qubits, for odd prime-dimensional qudits the latter contains the former.

The trifecta of qubit graphical calculi is the ZX-, ZW-, and ZH-calculi: not only historically, but these three were found to be the only three non-trivial qubit graphical calculi, notably forming a bialgebra with Z algebras [CJ20].

The qubit ZX-calculus [CD08; CD11] is the most well-studied of these. Its generators are diagonal in the Z and X bases. It can represent any map in the Clifford+Phases fragment, which is exactly universal for any qubit linear maps. Its *phase-free* fragment corresponds to CNOT circuits (which can have $|0\rangle$ and $|+\rangle$ ancillae and post-selections), and admits an interpretation as linear relations [Kis22].

The qubit ZW-calculus [CK10; Had17; Wan21a], which are based on the GHZ and W states, is suited for reasoning with sums of linear maps. This has a range of applications, particularly bosonic and fermionic computations [Had15; dFHN19; dFSP+23; CMPV23], and Hamiltonians [SWY23], and differentiation [JPV24] and integration [WYK24]. The phase-free ZW-calculus is universal and complete for integer-valued matrices.

The qubit ZH-calculus [BK19; BKM+23], has as generators the Z spider and the H-box. As in Definition 2.2, the H-box is a generalization of the H gate to arbitrary number of inputs and outputs. The m-input, 1-output H-box, followed by the H gate, is the AND operation $|x\rangle \otimes |y\rangle \mapsto |x \wedge y\rangle$ on the input bits x, y corresponding to Z basis states. Therefore, in the ZH-calculus, it is easy to represent gates whose action is multilinear on the Z basis states, notably the Toffoli and other classical reversible gates. ZH-diagrams have been studied for reasoning about hyper-graph states [Kup19], the path-sum formalism [LvdWK21; Vil21b; Vil23], quantum binary decision diagrams [Vil21a] and more [EvdWCG22]. The phase-free ZH-calculus corresponds to Toffoli+Hadamard circuits (which can have single-qubit Z-basis state ancillae and post-selections), and is universal for matrices over $\mathbb{Z}$ [BKM+23].

Significant research progress has been made in recent years generalizing quantum graphical calculi originally for qubits, to higher-dimensional qudits. At the time of writing the paper, for the ZX-calculus there was work on qutrits [vdWY23; Wan18b; WB14], the prime-dimensional qudit stabilizer fragment [IC22; PBC+23], and the universal algebraic qudit ZX-calculus [Wan21b; Wan22]. For the ZW-calculus there were several different proposals for qudit generalizations [PWS+23; Wan21a]. Missing from these proposals is a qudit generalization of the ZH-calculus.

In the associated paper, for the first time, we presented a qudit ZH-calculus. The qubit ZH-calculus can represent arithmetic over $\mathbb{Z}_2$. We show that likewise, the prime d-dimensional qudit ZH-calculus can represent arithmetic over $\mathbb{Z}_d$. Operations in the ZH-calculus act in $\mathbb{Z}_d$ as follows:

- the Z-spider represents copy, e.g. the 1-input, 2-output case is $x \mapsto (x, x)$,

- the X-spider represents addition, e.g. the 2-input, 1-output case is $(x, y) \mapsto (x +_d y)$, and
- the 1-output H-box followed by $H^\dagger = H^3$ represents multiplication, e.g. the 2-input, 1-output case is $(x, y) \mapsto x \cdot_d y$.

Throughout this section, we denote addition and multiplication modulo d by $+_d$ and $\cdot_d$.

To motivate what we are going for, we start by defining the qudit generalization of the Toffoli+Hadamard gate set, which to our knowledge has not been investigated before. While anyone can define a gate set by listing a set of gates, the real test(s) for if this particular gate set is worthy of being the right, correct, or canonical generalization of the qubit Toffoli+Hadamard gate set is to show that it has exactly the kinds of properties to make a convincing argument.

First, we need to determine a set of one or more gates which suffice to construct all the d-ary classical reversible circuits. We achieve this following our work for the qutrit case in Ref. [YvdW22][1], but here we 1) generalize it to all odd qudit dimensions, 2) improve the asymptotic gate count to be within a log factor of what is required, and 3) were able to reduce the number of gates in this gate set to the minimum possible of *one*.

While all binary classical reversible circuits are constructable from just the qubit Toffoli gate, we show that all odd d-ary classical reversible circuits are constructable from just the odd-dimensional qudit $|0\rangle$-controlled X gate[2].

Second, we consider how powerful adding the Hadamard gate to all the d-ary classical reversible gates is. For qubits, the Toffoli+Hadamard gate set is known to be *computationally universal* for qubit computation [Shi03]—this is weaker than approximately universal because it means that any qubit computation can be simulated up to arbitrary precision error, as opposed to implementing any qubit computation up to arbitrary precision error. The short story for why this qubit gate set can't implement arbitrary qubit unitaries, is that the matrices for both the Toffoli and the Hadamard gates is real-valued, so it is impossible for a composition of these gates to generate any matrix with complex entries. However, this constraint does not apply in higher dimensions because the matrix for the qudit Hadamard gate[3] has complex entries. What we find is that the Toffoli+Hadamard gate set is *approximately universal* for qudits of any odd prime dimension d, and hence this result is stronger for odd prime dimension qudits than for qubits.

Third, we define the rules and generators of the first ZH-calculus for prime-dimensional

[1]For these three reasons, in this thesis we skipped presenting this result for the qutrit case in Chapter 5.1, in favor of presenting the best result to date in this section.

[2]In the next section, for qutrits, we also achieve this result for another gate, the Toffoli i.e. CCX gate

[3]Technically, in mathematics a Hadamard matrix is a ± 1 matrix of maximum possible determinant, named after Hadamard's 1893 article on the matter [Had93]. Although an accurate terminology would be the d-dimensional discrete Fourier transform, we adhere to the (admittedly incorrect) convention predominant in the literature.

qudits. We then give a translation in any odd prime qudit dimension, from the ZH-calculus to the ZX-calculus. This enables us to prove the precise fragment of computation that it corresponds to—the prime-dimensional qudit Toffoli+Hadamard gate set, allowing ancillae and post-selections. Coming full circle, this means that this calculus is approximately universal for quantum computation in any odd prime dimension.

12.1 All odd prime-dimensional classical reversible gates can be built

with just the $|0\rangle$-controlled X gate

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.

We define the *$|0\rangle$-controlled X gate* as acting on the computational basis as follows:

$$|c, t\rangle \mapsto \begin{cases} |c, t +_d 1\rangle, & \text{if } c = 0 \\ |c, t\rangle, & \text{else} \end{cases} \tag{12.3}$$

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 CX gate conjugated by X on the control).

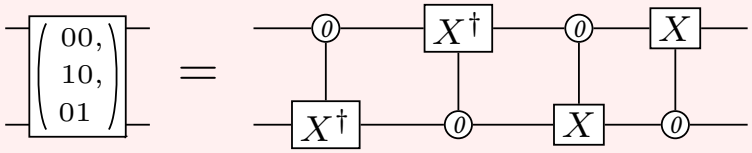
In the remainder of this section, we derive the following theorem:

Theorem 12.4. *For any odd qudit dimension d, any d-ary classical reversible function $f : \mathbb{Z}_d^n \to \mathbb{Z}_d^n$ 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.*

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 12.5.



(12.6)

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 $|x + (y + x^{d-1} - 1)^{d-1} - 1, y + x^{d-1} - \left(x + (y + x^{d-1} - 1)^{d-1} - 1\right)^{d-1}\rangle$. 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:

$$\begin{cases} |0\rangle, & \text{if } x = 0 \text{ and } y = 1 \\ |1\rangle, & \text{if } x = 1 \text{ and } y = 0 \\ |y\rangle, & \text{else} \end{cases} \tag{12.7}$$

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:

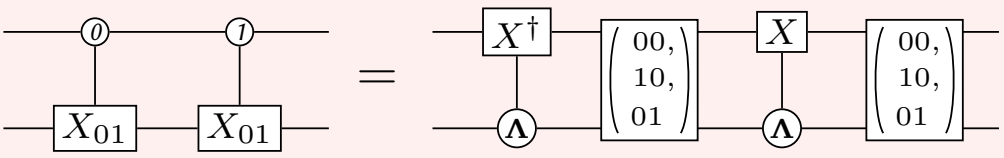
$$\begin{cases} |0\rangle, & \text{if } x = 1 \text{ and } y = 0 \\ |1\rangle, & \text{if } x = 0 \text{ and } y = 0 \\ |x\rangle, & \text{else} \end{cases} \tag{12.8}$$

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. □

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 12.9.



(12.10)

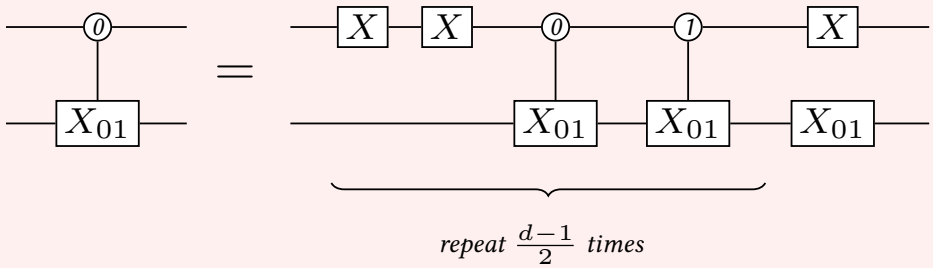
Note that here we have a Λ-controlled gate which for $\Lambda(U)$ implements $|x, y\rangle \mapsto U^x |y\rangle$. In

this particular case, $\Lambda(X)\,|x,y\rangle = |x,x+y\rangle$ is the CX gate, and is Clifford.

Corollary 12.11. *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.

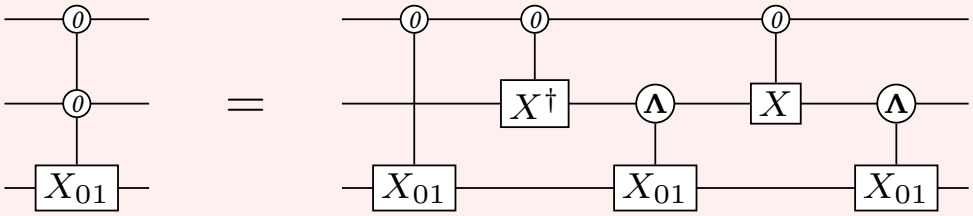
Lemma 12.12.



(12.13)

We then utilise the following generalisation to all odd qudit dimensions d, for which the qubit analogue is in Ref. [BBC+95, Lemma 7.5] and the qutrit analogue is in Ref. [YvdW22, Lemma 5]:

Lemma 12.14.

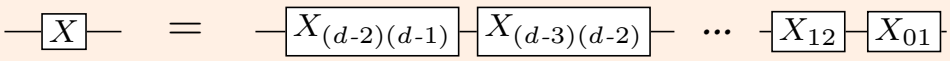


(12.15)

where $\Lambda(X_{01})$ can be further decomposed into controlling X_{01} on all odd computational basis states.

Corollary 12.16. *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.*

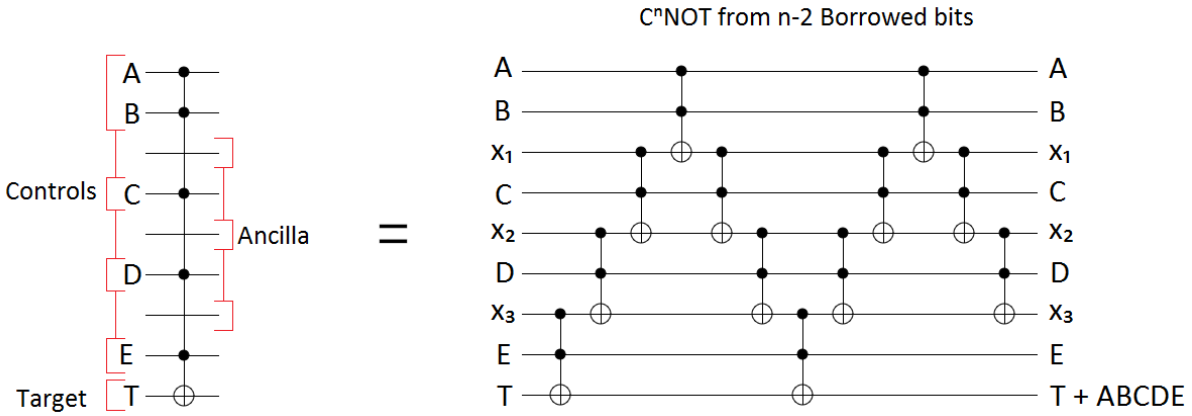
Corollary 12.17. *As we will shortly discuss in Proposition 12.22, any permutation can be generated by 2-cycles i.e. permutations exchanging only two elements. For example, the $|00\rangle$-controlled X gate can be obtained by $|00\rangle$-controlling each gate in the decomposition:*



(12.18)

In previous work [YvdW22], 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 [Gid15]):

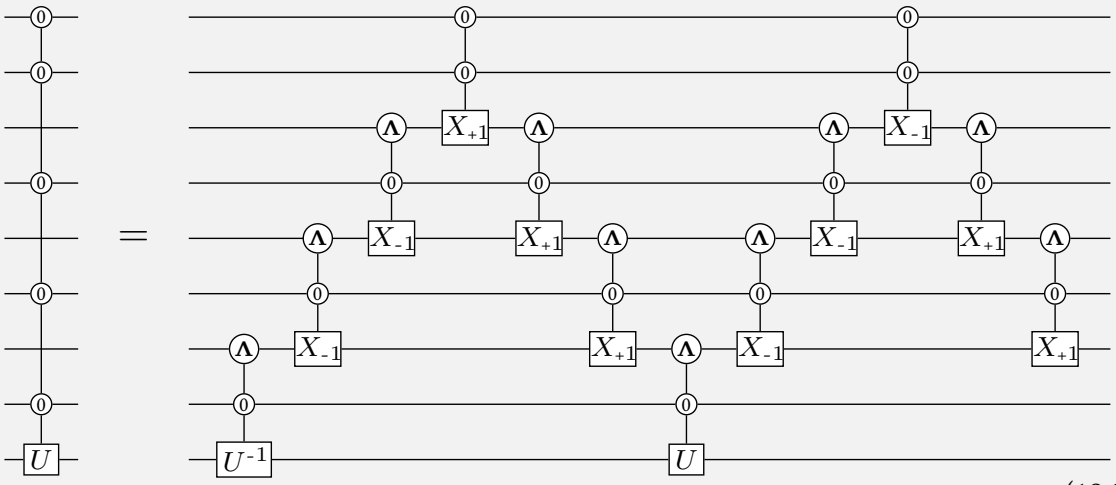


(12.19)

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 12.20. *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:



(12.21)

This approach is adaptable to arbitrary n. The control qudits can be conjugated by X gates to generalize this from $|0...0\rangle$-controlled, to control on any ditstring. This decomposition also applies if any of the control qudits are other types of controls, for instance Λ-controlled.

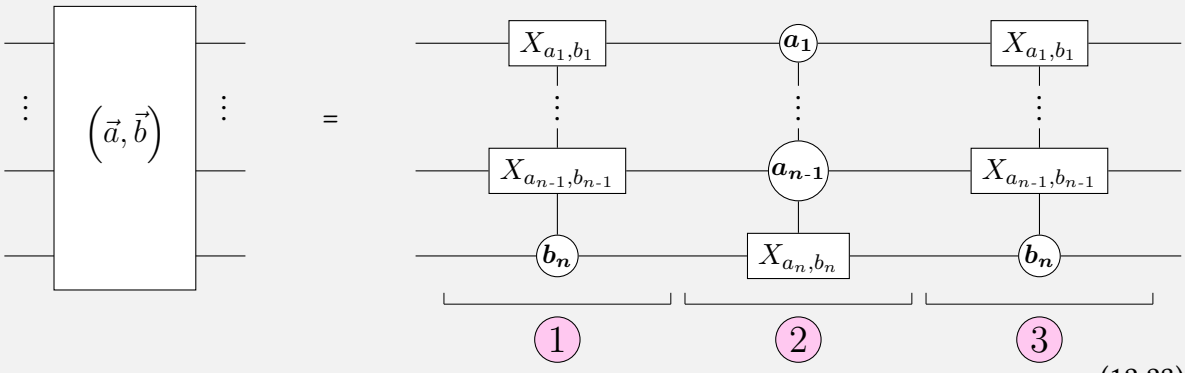
As an added note, if it is preferred to construct $\Lambda(U)$ instead of $\Lambda(|0\rangle$-controlled $U)$, the bottommost control qudit in the above decomposition can be omitted at the cost of adding

one borrowed ancilla. □

This lets us immediately apply the following proposition reprinted from Ref. [YvdW22] for qutrits, which holds for arbitrary qudit dimension.

Proposition 12.22. *Let $\vec{a} = (a_1, ..., a_n)$ and $\vec{b} = (b_1, ..., b_n)$ 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 12.20 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:



(12.23)

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 12.20 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 Equation (12.23) maps $\vec{a}$ to $\vec{b}$, $\vec{b}$ to $\vec{a}$, and is identity on all other ditstrings. □

As any permutation can be written as product of disjoint 2-cycles (this is well known, and explained in Ref. [YvdW22]), 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. [YvdW22]:

Theorem 12.24 (Restatement of Theorem 13.16)**.** *For any odd qudit dimension d, any d-ary classical reversible function $f : \mathbb{Z}_d^n \to \mathbb{Z}_d^n$ on n dits can be constructed by a circuit of $O(d^n n)$ $|0\rangle$-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 $O(d^n n)$. □

Remark 12.25. A preprint [ZLS23] appearing after submission of the associated paper independently discovered a version of Lemmas 12.14 and 12.20 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 generalization of Ref. [YvdW22], they independently derived our Proposition 12.26 and a version of our Theorem 13.16 which uses more types of gates than just the $|0\rangle$-controlled X, but which does work for all qudit dimensions.

To lower bound the gate complexity, we can generalize our proof from Ref. [YvdW22] to the qudit setting:

Proposition 12.26. *For any qudit dimension d, there exist d-ary classical reversible functions $f : \mathbb{Z}_d^n \to \mathbb{Z}_d^n$ 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 $N \geq \frac{\log d}{2} \frac{n d^n}{\log c + 2 \log n} \geq \frac{\log d}{6} \frac{n d^n}{\log n}$ for $n \geq c$, which shows that we must have $N = O(n d^n / \log n)$. □

Comparing the asymptotic gate counts of Theorem 13.16 and Proposition 12.26, we conclude that our algorithm is near-optimal (if not optimal), as it is within a $\log n$ factor of the gate count necessary.

12.2 The qudit Toffoli+Hadamard gate set

Interestingly, the previous section proved that 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 $f : \mathbb{Z}_d^n \to \mathbb{Z}_d^n$) with the help of $|0\rangle$ ancillae. Hence, the $|0\rangle$-controlled X gate is universal for all classical reversible logic—generalizing to all odd d what the three-qubit Toffoli gate does for $d = 2$. For this reason, it makes sense to consider the qudit gate set containing $|0\rangle$-controlled X and Hadamard, as a possible generalization of the qubit Toffoli+Hadamard gate set to arbitrary odd prime dimension.

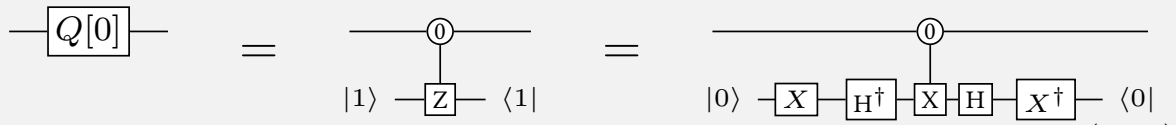
For qubits, adding the Hadamard gate to all the classical reversible gates (which is generated by the Toffoli gate and zeroed ancillae) suffices for approximately universal quantum computation [Shi03]. By combining Theorems 13.16 and 12.27, we find that this is in fact true in any prime qudit dimension.

Theorem 12.27. *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-qutrit non-Clifford gate) which we constructed in an appendix of the paper; as this is very similar to our construction of this gate in Chapter 8 [GRvdWY22c, Section 3], we do not repeat it here. The one noteworthy modification for this paper, was additionally constraining the allowed gate set to be {$|0\rangle$-controlled X, H} instead of all of qutrit Clifford+T. This thereby also proves that the gates implementable over the qutrit

metaplectic gate set is a subset of that over the qutrit Toffoli+Hadamard gate set.

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 Ref. [WHSK20] 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:



(12.28)

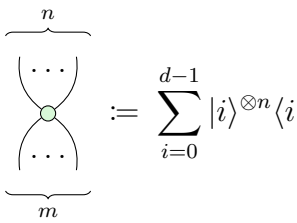
Conjugating by X gates then yields all $Q[i]$ gates. □

12.3 The qudit ZH-Calculus

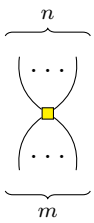
Remark 12.29. This section ending before the phase-free ZH- to ZX-calculus translation is verbatim from the paper, and jointly written with Patrick Roy and John van de Wetering. It was not reworded because as the section is solely the definition of the generators, interpretations, and rules of the calculus, there is little room to change anything wording wise.

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 [Car21] the orientation of diagrams in this chapter 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:



$$:= \sum_{i=0}^{d-1} |i\rangle^{\otimes n}\langle i|^{\otimes m}$$

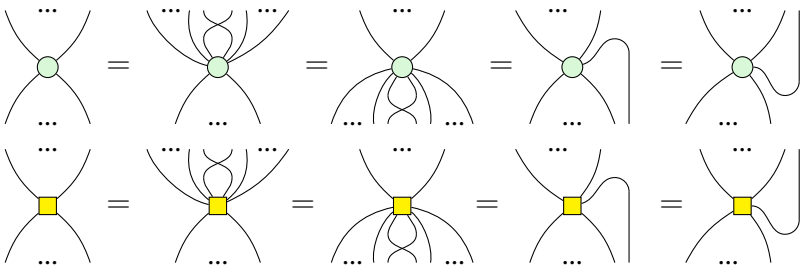


$$:= \frac{1}{\sqrt{d}} \sum_{i_1,\ldots,i_m,j_1,\ldots,j_n \in \mathbb{Z}_d} \omega^{i_1 \cdot \ldots \cdot i_m \cdot j_1 \cdot \ldots \cdot j_n} |j_1 \ldots j_n\rangle\langle i_1 \ldots i_m|$$

This matches the qubit-ZH definitions of [BK19], except that now the sums go from 0 to $d-1$ instead of from 0 to 1, and we use the dth 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

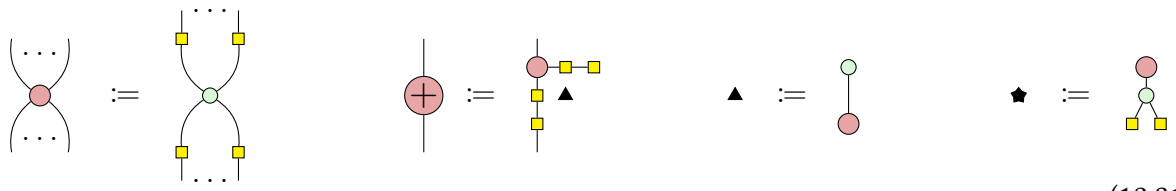
some tedious constants from appearing everywhere [dBea21, Ap. E]. As a consequence of this choice of normalisation, the 1-input, 1-output phase-free H-box corresponds exactly to the qudit Hadamard $|x\rangle \mapsto \frac{1}{\sqrt{d}} \sum_y \omega^{x \cdot y} |y\rangle$. Note that while the matrix of 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 [BKM+23].

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 [Car21]:



Remark 12.30. Note that the actual choice of dth root of unity $\omega = e^{2\pi i/d}$ is not important. We can choose any primitive dth 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.

There are a couple of useful derived generators we will need:

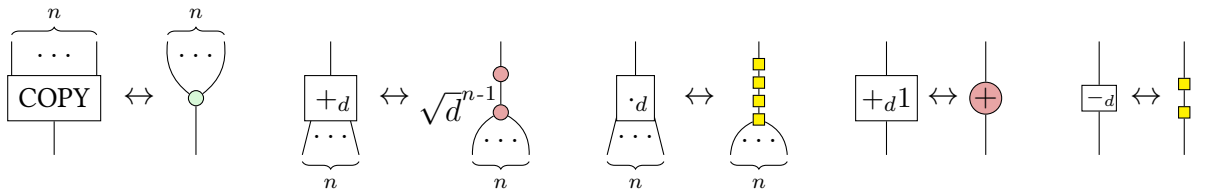


(12.31)

The first of these is the well-known X-spider. The second realizes the Pauli X gate, e.g. the map $X|i\rangle = |i +_d 1\rangle$. 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 [BKM+23, Eq. 5]. Similarly, our generators turn out to correspond with arithmetic

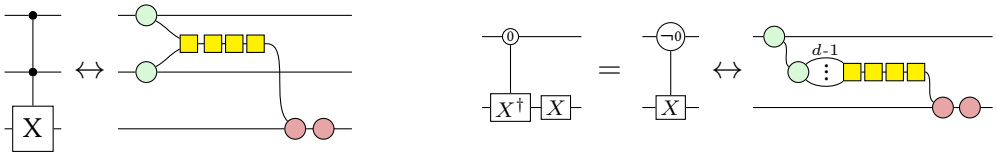
operations over $\mathbb{Z}_d$:



(12.32)

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* [CJ20], and we will use it throughout the diagrams in this chapter. 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):



(12.33)

The correctness of the Toffoli construction follows easily from the interpretation given in Equation (12.32). 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.

Rules of the qubit ZH-calculus generalize to qudits; see Figure 12.1. We prove soundness of these rules, along with additional rules, in the appendix of the associated paper.

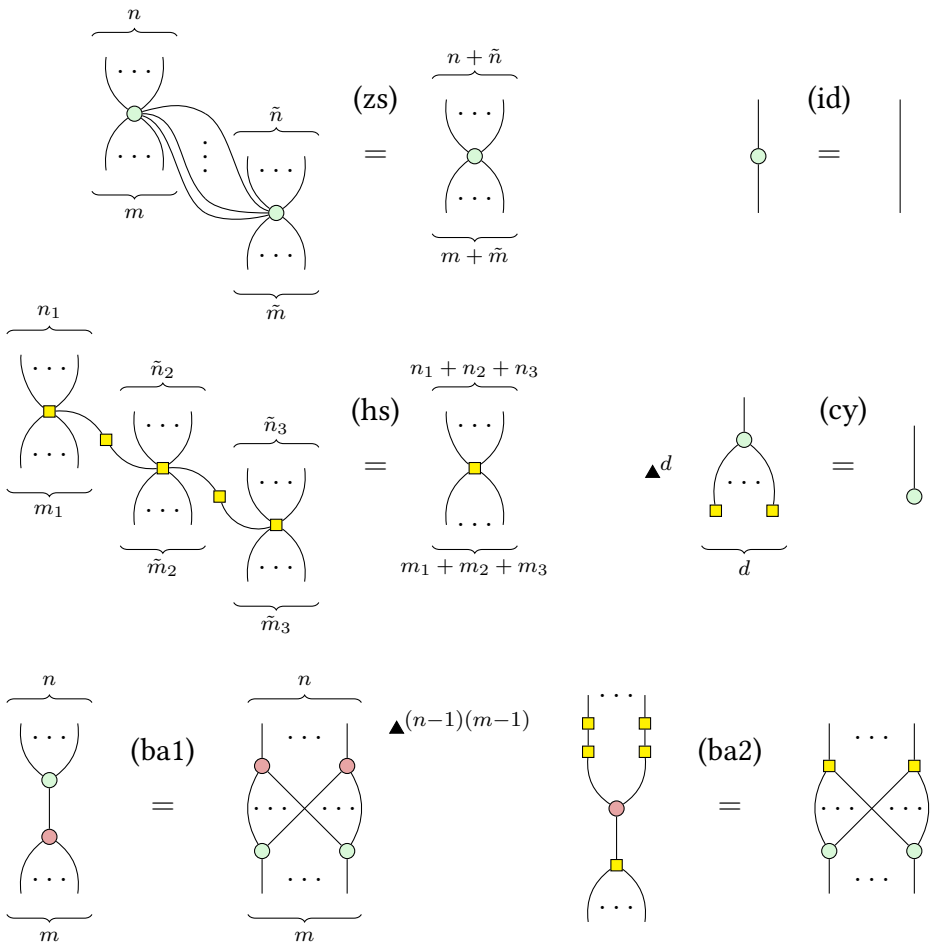
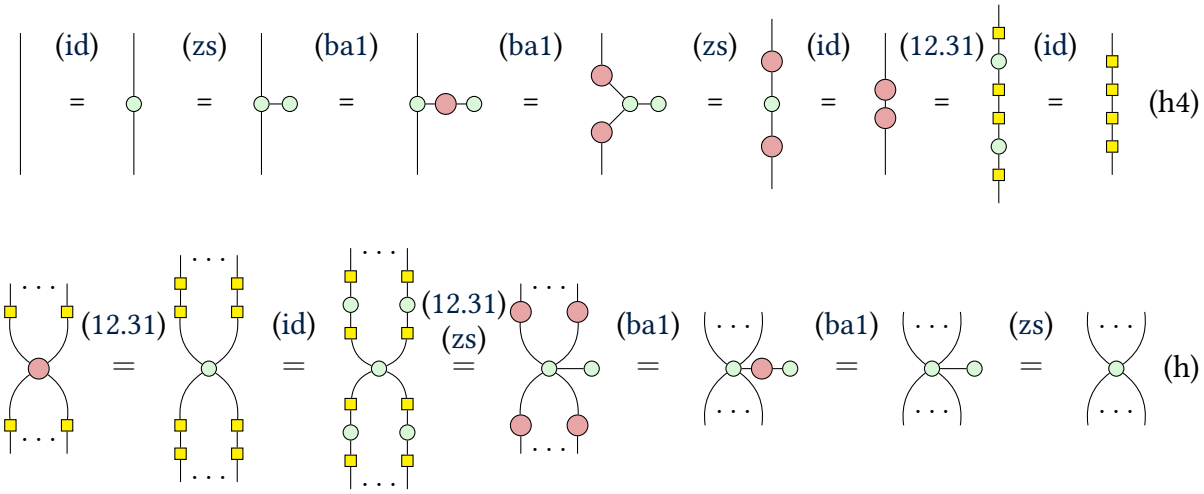


Figure 12.1: Basic rules of the phase-free qudit ZH-calculus. Some additional new and derived rules are presented in the appendices of the associated paper. The rules hold for all n and m. Here d is the dimension of the qudit.

The Z-spider fusion rule generalizes 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 generalizes 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 it the cyclic structure of the group $\mathbb{Z}_d$.

Note how the above ruleset neither contains a rule stating that $H^4 = \mathrm{id}$, nor an inverted color change rule. That is because we can derive them from the rules presented above:



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 [vdWY23, Fig. 1].

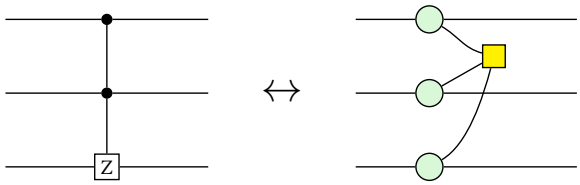
Since these derivations hold for arbitrary dimension d, they particularly hold for $d = 2$. This means that due to the qu*bit* H-box fusion rule, (h4) actually implies the self-inverseness of Hadamard gates, making the (hh) rules of Backens et al's ruleset redundant [BKM+23, Tab. 1].

In the appendix of the associated paper, we also present a generalization of the ortho rule from the phase-free qubit ZH-calculus [BKM+23]. Hence, we have a prime-dimensional qudit generalization of all the phase-free qubit ZH-calculus rewrite rules [BKM+23]. While those rules are complete for the qubit phase-free calculus, it is unknown whether this continues to hold for qudits.

12.4 Translating ZH-diagrams to ZX-diagrams

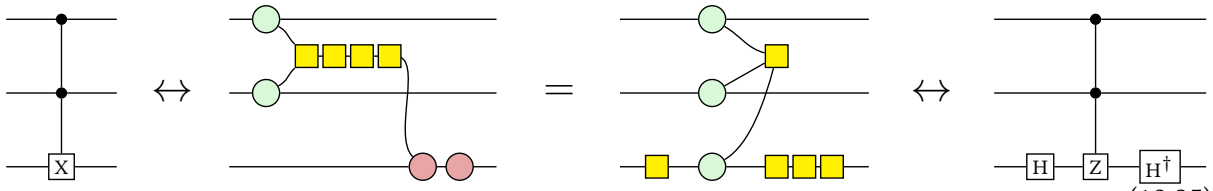
The qudit ZX-calculus is universal, and hence can represent any linear map between qudits [WB14]. So in particular, there must be some way to interpret ZH-diagrams into ZX-diagrams. As the only generator of ZH-diagrams that is different from the ZX-calculus is the H-box, this is the only one we will 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 earlier.

First, note that there is a close correspondence between an H-box and the qudit CCZ gate, which acts like $|x, y, z\rangle \mapsto \omega^{x \cdot y \cdot z} |x, y, z\rangle$:



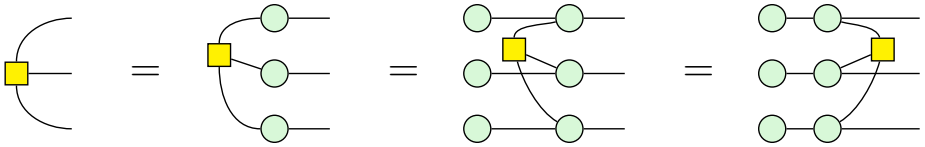
(12.34)

An additional check that this is the natural definition of the CCZ gate, is that it is the result of conjugating the target of the Toffoli gate by H gates.



(12.35)

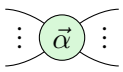
Hence, in particular the three-legged H-box is equal to one copy of the qudit CCZ gate acting on $|{+}{+}{+}\rangle$:



(12.36)

Since a CCZ gate is just the Toffoli from Equation (12.33) 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 13.16 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:

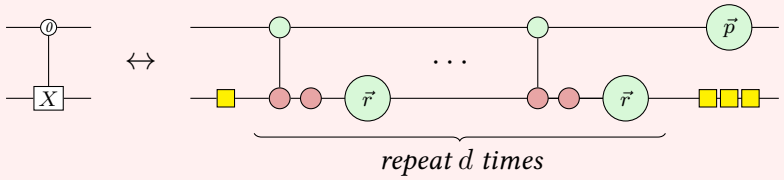


$$= |0...0\rangle\langle 0...0| + e^{i\alpha_1} |1...1\rangle\langle 1...1| + ... + e^{i\alpha_{d\text{-}1}} |(d\text{-}1)...(d\text{-}1)\rangle\langle (d\text{-}1)...(d\text{-}1)| \tag{12.37}$$

By translating the Clifford+Phases decomposition of Proposition 11.15 from the previous section, we have two equivalent diagrams for the $|0\rangle$-controlled X gate: one phase-free ZH-diagram, and one Clifford+Phases ZX-diagram.

Lemma 12.38 ([Yeh23]). *The prime-dimensional qudit $|0\rangle$-controlled X gate can be decomposed into the Clifford+Phases gate set (decomposing H as phase gates [Wan22, Remark 2.3]), and*

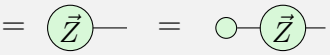
written as a ZX-diagram:



(12.39)

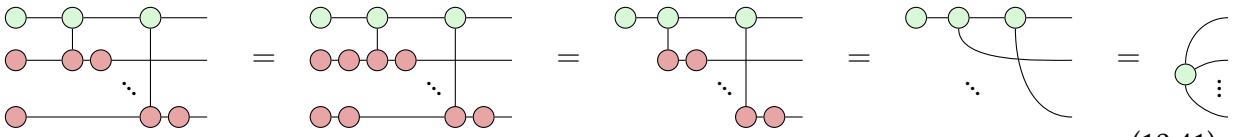
where $\vec{p} = \left(\omega^{\frac{-(d-1)}{2}}, \omega^{\frac{-(d-1)}{2}}, ..., \omega^{\frac{-(d-1)}{2}}\right)$ *and* $\vec{r} = \left(\omega^{\frac{1}{d}}, \omega^{\frac{2}{d}}, ..., \omega^{\frac{d-1}{d}}\right)$ *represents the* d*th root of* Z *gate from Ref. [Yeh23].*

Theorem 12.40. *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 $Z|+\rangle$:  where $\vec{Z} = \left(\omega, \omega^2, ..., \omega^{d\text{-}1}\right)$ indicates the Z gate.

Eq (12.36) shows how to relate the three-legged H-box to the CCZ gate. We can then invoke Theorem 13.16 to build the Toffoli gate from $|0\rangle$-controlled X gates, which is related to the CCZ gate by conjugating the target by Hadamards. Equation (12.39) 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). □

Note that phase-free Z-spiders are just GHZ states $|0...0\rangle + |1...1\rangle + ... + |(d\text{-}1)...(d\text{-}1)\rangle$, up to cups and caps. Hence, using a typical decomposition of any size qudit GHZ state into a CX circuit on $|+0...0\rangle$,



(12.41)

we can further decompose the Z-spiders in Theorem 12.40 into $\{|0\rangle$-controlled X, H, $|0\rangle\}$ as $|+\rangle = H|0\rangle$. This enables us to write any phase-free ZH-diagram as a $\{|0\rangle$ -controlled X, H$\}$ circuit where ancillae and postselections on $|0\rangle$ and $\langle 0|$ are allowed. By Equation (12.33), we can also write any circuit composed of $\{|0\rangle$ -controlled X, H, $|0\rangle$, $\langle 0|\}$ as a phase-free ZH-diagram. Prime-dimensional phase-free ZH-diagrams thus correspond to $\{|0\rangle$ -controlled X, H$\}$ circuits with ancillae and postselections. This therefore generalizes the same correspondence as in the qubit case, between the phase-free ZH-calculus and the Toffoli+Hadamard gate set.

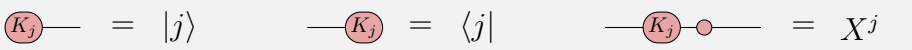
13

Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits

Before we dive into this section, we have a couple of points in retrospect, to round out the results up to now.

Proposition 13.1. *As is the case for qubits, for any odd prime qudit dimension, the Toffoli and CCZ gates are in the third level of the Clifford hierarchy.*

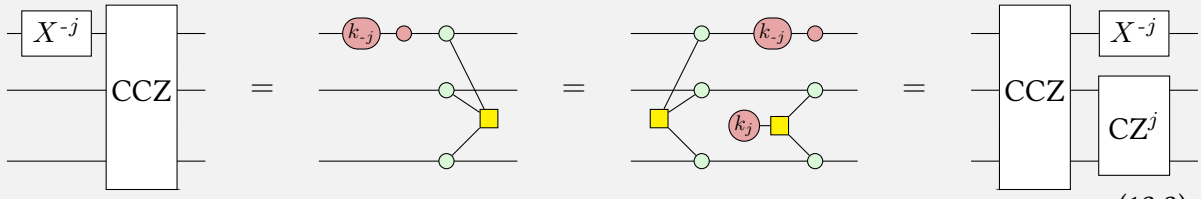
Proof. Noting the translations of the following diagrams from the flexsymmetric qudit calculus:



(13.2)

Pauli Z directly commutes through the CCZ gate because both are diagonal in the Z basis.

We can commute Pauli X through the CCZ gate as follows:



(13.3)

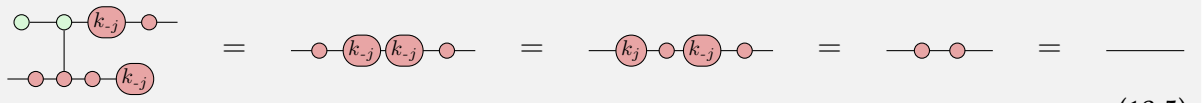
□

As it is in the third level of the Clifford hierarchy, the Toffoli and CCZ gates admit magic state injection for any prime qudit dimension; moreover, as both these gates are semi-Clifford (Definition 3.23), a more efficient (halving the number of ancillae qudits) gate teleportation can be utilized [dSil21].

Proposition 13.4. *For any prime qudit dimension, the CCZ gate is deterministically implementable by one round of the below magic state injection protocol, with one copy of the CCZ magic state*[a].

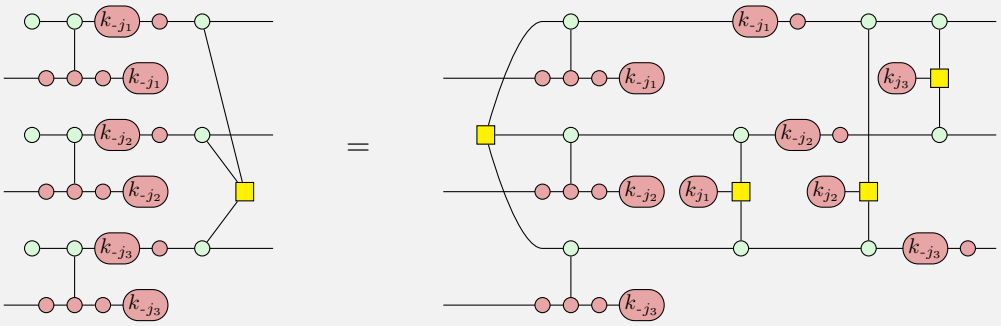
[a] Thank you to Nadish de Silva for the pointers explaining the approach for efficient multi-qudit gate teleportation to which he alluded in Ref. [dSil21]

Proof. The multiqudit case of the teleportation circuit follows from parallelism of the single-qudit case.



(13.5)

By commuting the intended gate to teleport (the CCZ gate) through the three-qubit teleportation circuit, we get that the CCZ magic state can be teleported by that circuit, with Pauli X and Clifford CZ gate corrections controlled on the measurement outcomes.



(13.6)

□

As for magic state distillation of CCZ magic states, while we have not looked into a specific

code, we can try the approach proposed earlier for synthesizing qutrit codes with a transversal binary AND gate. For qutrits, it should be possible to identify a T-depth 1, distance 1 code in CSS GPF. Equation (10.31) suggests the existence of qutrit error correcting codes admitting transversal implementation of the CCZ gate—and hence not the H gate, by the Eastin-Knill no-go theorem [EK09] against codes having an approximately universal transversal gate set.

Figure 13.1 shows relationships between the three qutrit gate sets we have been focusing on.

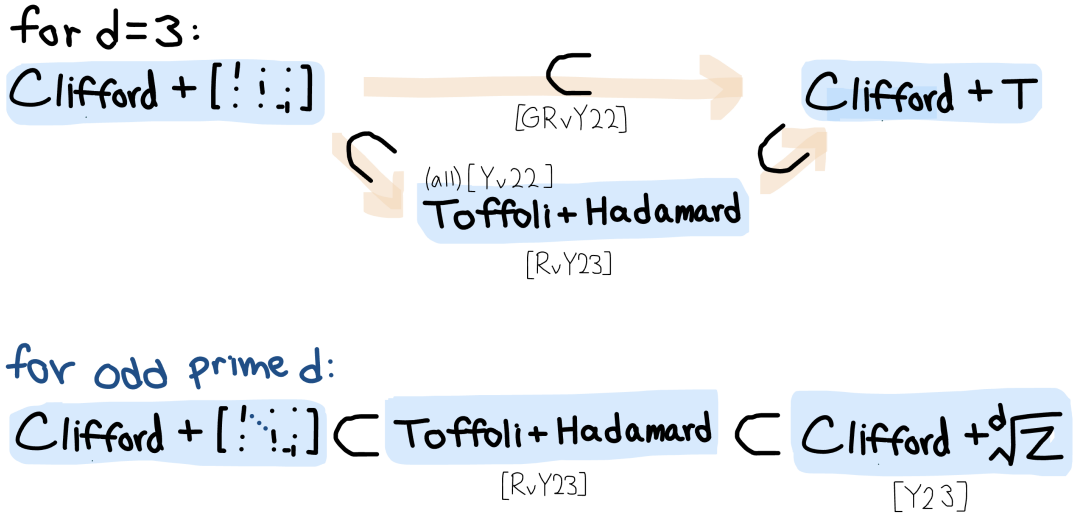


Figure 13.1: **Top:** The three qutrit gate sets we have been focusing on, and a recap of our results about relationships between them. **Bottom:** For generalizations of these three qutrit gate sets to arbitrary odd prime qudit dimension, these subset relations continue to hold.

Here we summarize our results thus far pertaining to the relations between these gate sets. In Theorem 8.35 we showed that Clifford+R is a subset of Clifford+T [GRvdWY22a]. In Ref. [YvdW22] we showed how to generate all ternary classical reversible gates; implicitly we constructed these from only the $|0\rangle$-controlled X gate, plus single-qutrit X gates. In Ref. [Yeh23] we defined the Clifford+$\sqrt[d]{Z}$ gate set for arbitrary prime qudit imension d. We showed that you could build the $|0\rangle$-controlled X gate in it. In Ref. [RvdWY23], we defined the Toffoli+Hadamard gate set for arbitrary odd prime qudit dimension d. We showed that it contains the $(-1)_{[x]}$ 1-level operator, the Cliffords, and all the d-ary classical reversible gates.

This section is the exception to the organization of this chapter being about qudits, the previous chapter being about qutrits, and the chapter before that being about qubits. This is for the order of this section relative to the previous sections to match the real-world timeline. The Toffoli+Hadamard gate set for any prime dimension higher than two, was first defined in the paper the previous section is about. After that, the qutrit Toffoli+Hadamard gate set plays a key role in the result of this section—the first exact synthesis of any approximately universal

gate set for multiple qudits of a dimension higher than two.

There are a number of reasons justifying the canonicity of this gate definition, summarizing from the previous chapter and the start of this chapter.

- Conjugating the target of the qutrit Toffoli gate by the Hadamard gate indeed gives the CCZ gate, a symmetric three-qutrit gate diagonal in the Z basis.
- The CCZ gate is easy and efficient to implement by one round of magic state injection, with one copy of the CCZ magic state.
- As is analogous for the qubit case, the qutrit Toffoli gate (when allowing $|0\rangle$ and $|1\rangle$ ancillae) suffices to generate all ternary classical reversible circuits.
- The fragment of computation represented by the qutrit Toffoli+Hadamard gate set corresponds precisely to that of phase-free ZH-calculus diagrams allowing ancillae and post-selections, and hence is generated by just two arbitrary arity tensors, Z and H.
- The qutrit Toffoli+Hadamard gate set is approximately universal, and contains the qutrit Cliffords as a subgroup.
- Recall that in Chapter 8 we proved that the gates implementable over the qutrit metaplectic gate set is a strict subset of that over the qutrit Clifford+T gate set. The qutrit Toffoli+Hadamard gate set sits right below the Clifford+T gate set in the qutrit Clifford-cyclotomic hierarchy.

Depending on purpose, with regards to exact circuit synthesis, in some settings the $|0\rangle$-controlled X gate and the CCX gate can be used interchangeably. While it is certainly easier to implement a two-qudit gate than a three-qudit gate, choosing the CCX gate has the benefit of generalizing from the qubit case, whereas the $|0\rangle$-controlled X gate does not have a qubit analog.

13.1 Multiqutrit Exact Synthesis

Remark 13.7. The remainder of this section will consist of the paper "Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits" [GRvdWY24], in collaboration with Andrew Glaudell, Neil J. Ross, and John van de Wetering. A software implementation verifying circuit constructions of the multiqutrit level operators is available at https://github.com/lia-approves/qutrit-Clifford-cyclotomic [GRvdWY24].

Despite the varied advantages of qudit computation, many of the advancements made in qubit exact circuit synthesis have not been solved for qutrits, much less for qudits. The handful of results available for qutrit exact circuit synthesis includes a normal form for single-qutrit

Clifford+T unitaries [GRT19; PJKS18], a proof that all classically reversible functions on trits can be implemented using Clifford+T circuits [YvdW22], and an exact synthesis result for single-qutrit Clifford+R unitaries [KMV25]. The most discernible gap, is that at the time of writing the paper, no multiqudit (for any $d > 2$) exact circuit synthesis result had ever been achieved before.

Before diving into the details, we can offer a brief perspective as to why the qutrit instance of this problem had additional technical challenges, compared to the qubit case. The most upfront obstacle is a matter of scale. After taking the residue map (Equation (5.28)), the qubit Clifford+T gate set only has 16 elements in total. The original qubit Clifford+T exact circuit synthesis paper simply enumerated a 16 row table of the needed operations on residues, for which the row operations turned out to only be three distinct cases.

On the other hand, the qutrit Clifford+T gate set has a whopping 729 elements in total after taking the residue map. Not only would it be inhuman to read a 729 row table, but brute force searching for case distinctions without a strategy would be computationally intractable.

Qubit Clifford+T

$$(\mu_2^{-1}\mathbb{Z})[\sigma] = \{a\sigma^3 + b\sigma^2 + c\sigma + d \mid a,b,c,d \in \mu_2^{-1}\mathbb{Z}\} \xrightarrow{\text{mod } 2} 2^4 = 16 \text{ elements}$$

Qutrit Clifford+T

$$(\mu_3^{-1}\mathbb{Z})[\zeta] = \{a + b\zeta + c\zeta^2 + d\zeta^3 + e\zeta^4 + f\zeta^5 \mid a,b,c,d,e,f \in \mu_3^{-1}\mathbb{Z}\} \xrightarrow{\text{mod } 3} 3^6 = 729 \text{ elements}$$

Thankfully, there are number-theoretic tools at our disposal to handle the situation more manageably.



In the order we will encounter them in this section, we preview a breakdown of three important differences between the qubit [GS13] and qutrit (this work) Clifford+T exact circuit synthesis algorithms. First, although the qubit least denominator exponent (LDE) did induction on powers of $\sqrt{2}$, for the qutrit case here we do induction on powers of $\lambda = 1 - \omega$ instead of $\sqrt{3}$, to leverage properties of λ proven in Section 13.3. Second, where the qubit algorithm applied the H two-level operator to reduce the LDE of two elements of a column vector at once, as

the qutrit H gate has a 3 by 3 matrix, we have to reduce the LDE of three elements at once (Section 13.4). Third, while the qubit algorithm directly synthesized Clifford+T circuits, we will first solve the exact synthesis problem for the qutrit Toffoli+Hadamard gate set, then utilize the technique of *catalytic embeddings* to lift this along with a single T magic state, to achieve exact synthesis of any qutrit Clifford+T unitary. On the plus side, the qubit approach (overviewed in Algorithm 1) provided a blueprint for how to tackle the problem.

Now consider the ring $\mathbb{Z}[1/3, \omega_k]$, which can be defined as the smallest unital subring of $\mathbb{C}$ containing $1/3$ and ω_k. Since $-\omega^2/\sqrt{-3} = \omega^2(1-\omega)/3$, the entries of X, CX, CCX, H, and T_k lie in $\mathbb{Z}[1/3, \omega_k]$. Hence, any n-qutrit circuit over $\mathcal{G}_k$ must represent a unitary matrix with entries in $\mathbb{Z}[1/3, \omega_k]$. The purpose of this paper is to show that the converse implication is also true.

Disclaimer: After the present work was completed, it was brought to our attention that related results were independently established in Ref. [KSV+24].

The qutrit Clifford-cyclotomic hierarchy of gate sets

First, we re-define the gates we will be using. We give them all here because with regards to the number-theoretic properties we will be proving, we need the precise definitions we work with.

Definition 13.8 (Qutrit Cliffords)**.** Let k be a positive integer and let $\omega_k \in \mathbb{C}$ be the **primitive 3^k-th root of unity** $\omega_k = e^{i\frac{2\pi}{3^k}}$. For simplicity, we write ω for ω_1. The single-qutrit **Pauli** X gate, **Pauli** Z gate, **phase** gate S, and **Hadamard** gate H are defined below.

$$X = \begin{bmatrix} \cdot & \cdot & 1 \\ 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & \omega & \cdot \\ \cdot & \cdot & \omega^2 \end{bmatrix} \qquad S = \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \\ \cdot & \cdot & \omega \end{bmatrix} \qquad H = \frac{-\omega^2}{\sqrt{-3}} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{bmatrix}$$

The two-qutrit **controlled-X** gate CX is the permutation matrix whose action on the computational basis is defined by $|i\rangle\,|j\rangle \mapsto |i\rangle\,|i+j\rangle$, with addition performed modulo 3.

The gate set $\{H, S, CX\}$ is the **Clifford** gate set.

Definition 13.9. The three-qutrit **doubly-controlled-X** gate CCX (or **Toffoli** gate) is defined by $|i\rangle\,|j\rangle\,|k\rangle \mapsto |i\rangle\,|j\rangle\,|k+ij\rangle$.

Definition 13.10. The single-qutrit T_k gate is defined as

$$T_k = \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & \omega_k & \cdot \\ \cdot & \cdot & \omega_k^2 \end{bmatrix}.$$

When $k = 2$, T_k is the qutrit T gate [HV12].

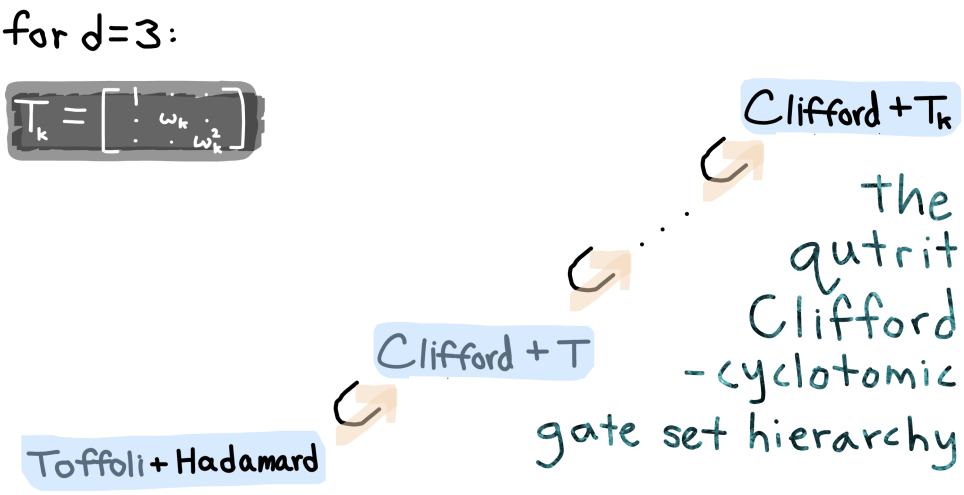
Remark 13.11. Another definition of the qutrit T gate that is also used in the literature and elsewhere in this thesis, is $T = \text{diag}(1, \omega_2, \omega_2^\dagger)$, where $\omega_2^\dagger$ is the complex conjugate of ω_2. Note that either definition is equally valid as they are both single-qutrit diagonal gates; these two definitions are Clifford equivalent, differing by only an S gate. We use this definition here because it is suitable for generalization to higher dimensions, such as how in Ref. [Yeh23] we used the generalization of T_2 to prove a decomposition of the $|0\rangle$-controlled X gate in any odd prime qudit dimension.

Definition 13.12. The **Clifford-cyclotomic gate set of degree** 3^k is the gate set $\mathcal{G}_k = \{X, CX, CCX, H, T_k\}$.

In the next subsection, we will show that

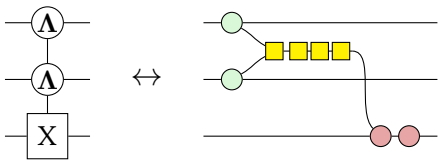
Lemma 13.13. *When $k = 1$, we have $T_1 = Z = HXH^\dagger$, so that the Clifford-cyclotomic gate set of degree 3 is equivalent to the qutrit **Toffoli+Hadamard** gate set [RvdWY23].*

We will also show that when $k \geq 2$, the gate set $\mathcal{G}_k$ is equivalent (up to a single ancillary qutrit) to the **Clifford+**T_k gate set $\{H, S, CX, T_k\}$. In particular, the Clifford-cyclotomic gate set of degree 9 is equivalent to the well-known qutrit **Clifford+**T gate set [GRT19; GRvdWY22a; PJKS18; YvdW22]. Because $T_{k+1}^3 = T_k$, the Clifford-cyclotomic gate sets form a hierarchy of universal gate sets whose first level is given by the Toffoli+Hadamard gate set, whose second level is given by the Clifford+T gate set, and whose subsequent levels are given by finer and finer extensions of the Clifford gate set.



13.2 Gate Set Equivalences

Recall from the previous section that the **Toffoli** gate can be represented in the prime-dimensional qudit ZH-calculus [RvdWY23] as below.



(13.14)

In Equation 13.14, Λ denotes the following type of control: if U is a unitary and $|c\rangle$ and $|t\rangle$ are computational basis states, then $\Lambda(U)\,|c\rangle\,|t\rangle = |c\rangle \otimes (U^c\,|t\rangle)$. In particular, $\Lambda(X)$ is the CX gate and $\Lambda(\Lambda(X)) = \Lambda(CX)$ is the CCX gate.

We now recall the definition of the $|0\rangle$**-controlled** X gate, which applies an X gate to its target if and only if its control is in the state $|0\rangle$ [RvdWY23].

Definition 13.15. Let d be a prime. The qudit $|0\rangle$**-controlled** X gate acts on computational basis states as

$$|c,t\rangle \mapsto \begin{cases} |c,t+1\rangle & \text{if } c=0, \text{ and} \\ |c,t\rangle & \text{otherwise,} \end{cases}$$

where arithmetic is performed modulo d.

Remarkably, when d is a prime greater than 2, the X gate and the $|0\rangle$-controlled X gate suffices to generate all of the d-ary classical reversible gates [RvdWY23]. Moreover, as was shown in [Sel16; YvdW22], when $d = 3$, no ancillary qutrits are needed for this purpose. In contrast, there is no collection of reversible one and two-qubit gates that suffices to generate all of the binary reversible gates.

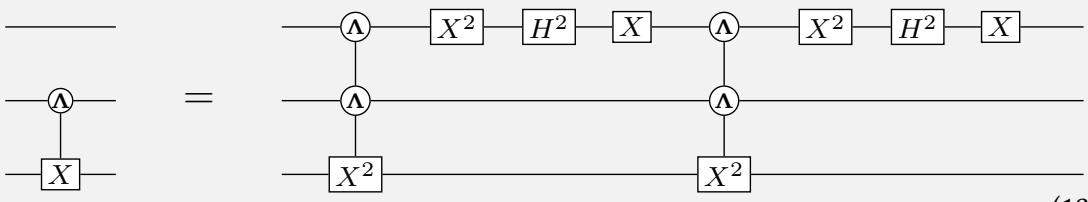
Theorem 13.16 ([YvdW22], Theorem 2). *Any ternary classical reversible function $f : \{0,1,2\}^n \to \{0,1,2\}^n$ can be represented by an ancilla-free circuit of X and $|0\rangle$-controlled X gates.*

Here, we only need multiply-controlled Toffoli gates, which can be built with a gate count linear in the number of controls, as in [RvdWY23; ZLS23]. The constructions of [RvdWY23; ZLS23] use no more borrowed ancillae than there are controls. They can be made into ancilla-free constructions by building Toffoli gates with $n/2$ controls using at most $n/2$ borrowed ancillae. Following [YvdW22], one can then combine six of these Toffoli gates with $n/2$ controls to construct a Toffoli gate with $n-1$ controls, and then combine 3 of these Toffoli gates with $n-1$ controls to add the final control.

We now show that the CX gate can be represented by a circuit over $\{X, CCX, H\}$.

Lemma 13.17. *The gate sets $\{X, CX, CCX, H\}$ and $\{X, CCX, H\}$ are equivalent up to a single borrowed ancilla.*

Proof. The circuit below represents the CX gate using a single borrowed ancilla.

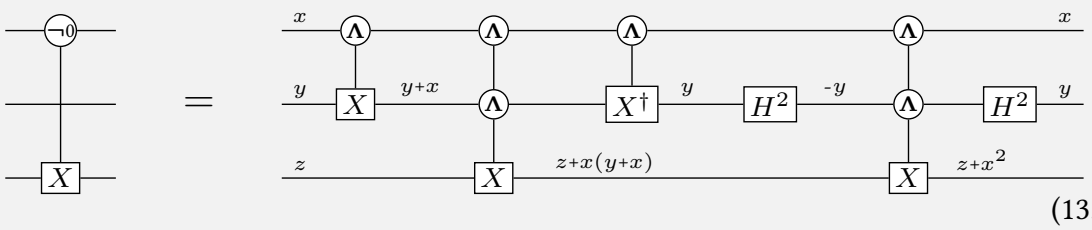


(13.18)

□

Proposition 13.19. *Let $C_{|0\rangle}X$ denote the qutrit $|0\rangle$-controlled X gate. Then the gate sets $\{X, CCX, H\}$ and $\{X, C_{|0\rangle}X, H\}$ are equivalent up to a single borrowed ancilla.*

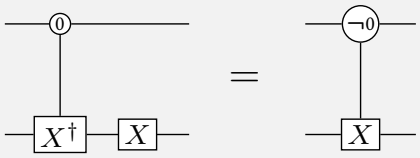
Proof. The gates X and CCX are ternary classical reversible functions. Hence, by Theorem 13.16, they can both be represented by a circuit over $\{X, C_{|0\rangle}X, H\}$. Thus, every matrix that can be represented by a circuit over $\{X, CCX, H\}$ can be represented by a circuit over $\{X, C_{|0\rangle}X, H\}$. Conversely, we have



(13.20)

where $x, y, z \in \{0,1,2\}$ are input qutrit computational basis states and the basis state on

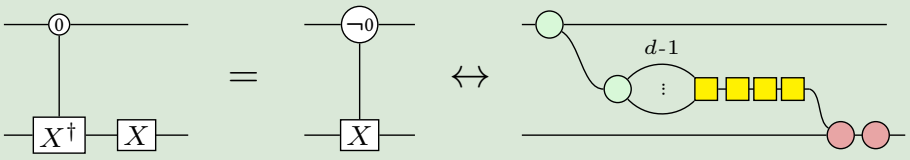
a wire is updated whenever it is changed by the circuit. The $\neg 0$ on the left-hand side of Equation 13.20 indicates that the X gate is applied when the control is not in the state $|0\rangle$. To see that Equation 13.20 holds, note that $x^2 = 1$ for $x \neq 0$ so that $z + x^2$ is indeed the desired state. Moreover, we have



(13.21)

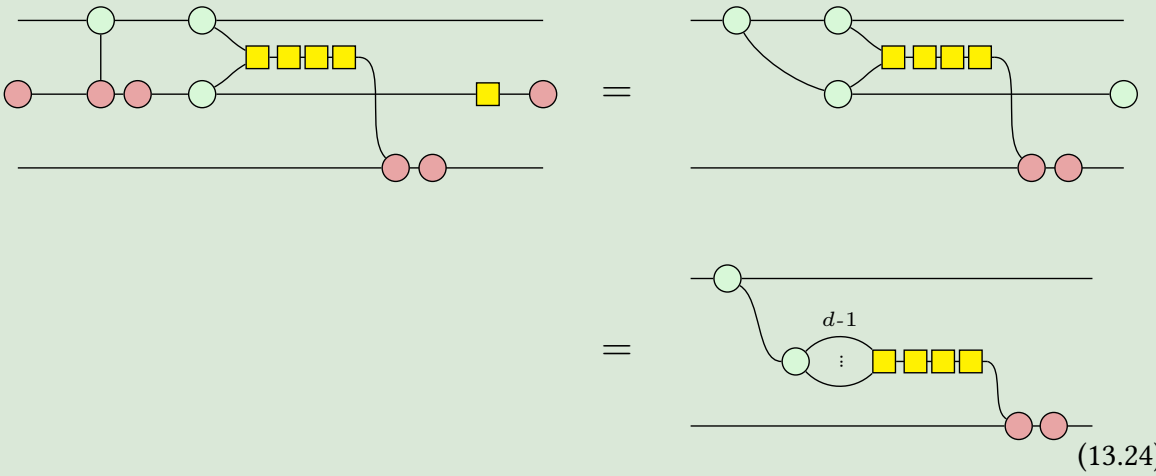
Therefore, multiplying the inverse of the circuit on the right-hand side of Equation 13.20 by an X gate yields a representation of the $C_{|0\rangle}X$ over the gate $\{X, CCX, H\}$ by Lemma 13.17. Hence, every matrix that can be represented by a circuit over $\{X, C_{|0\rangle}X, H\}$ can be represented by a circuit over $\{X, CCX, H\}$ using a single borrowed ancilla. □

Remark 13.22. The construction in Proposition 13.19 can be explained (and, in fact, was found) using the prime-dimensional qudit ZH-calculus [RvdWY23]. In the prime-dimensional qudit ZH-calculus, we have



(13.23)

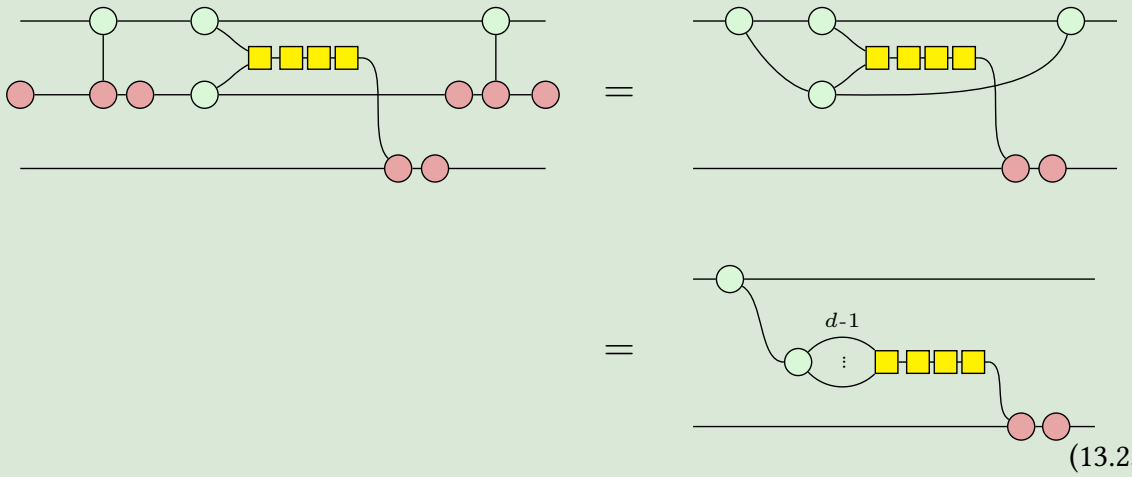
where the $d-1$ label indicates there are $d-1$ number of wires in parallel. We then get the following construction of the $|\neg 0\rangle$-controlled X gate:



(13.24)

The post-selected circuit in Equation 13.24 can be made deterministic by adding a $CX^\dagger$ gate

for uncomputation, which yields a construction requiring a fresh ancilla:



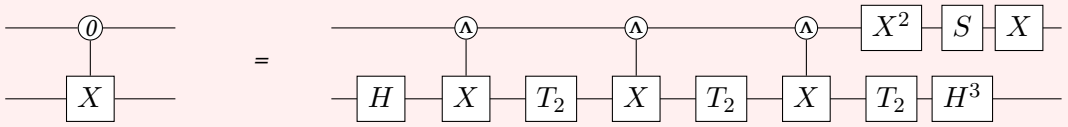
(13.25)

The construction is then modified in order to work with a borrowed ancilla, which yields the circuit in Equation 13.20.

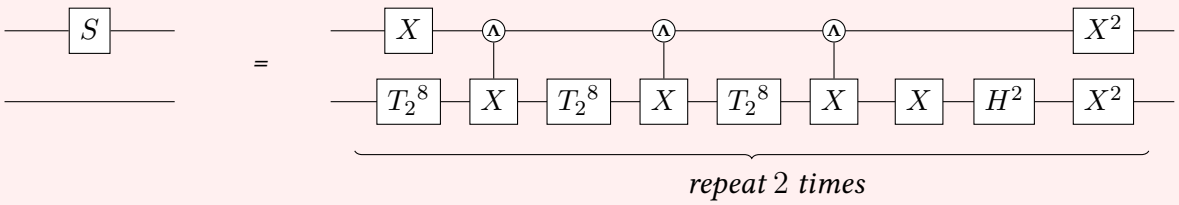
By Lemma 13.17 and Proposition 13.19, the gate set $\mathcal{G}_1$ is equivalent (up to a borrowed ancilla) to the gate set consisting of the X gate, the $|0\rangle$-controlled X gate, and the Hadamard gate. Hence, by Theorem 13.16, any ternary classical reversible function can be represented by a circuit over $\mathcal{G}_1$ using at most one borrowed ancilla.

We now show that, when $k \geq 2$, the Clifford-cyclotomic gate set of degree 3^k is equivalent, up to a borrowed ancilla, to the Clifford+T_k gate set. We take advantage of some constructions from [BRS17] (see, in particular, Figure 6 in [BRS17]).

Lemma 13.26. *We have:*



Lemma 13.27. *We have:*



Proposition 13.28. *When $k \geq 2$, the Clifford-cyclotomic gate set $\mathcal{G}_k$ is equivalent to the Clifford+T_k gate set up to a single borrowed ancilla.*

Proof. Recall that $\mathcal{G}_k = \{X, CX, CCX, H, T_k\}$ and that Clifford+$T_k = \{H, S, CX, T_k\}$. To prove the proposition, we therefore need to show that the S gate can be represented by a circuit over $\mathcal{G}_k$ and that the X and CCX gates can be represented by Clifford+T_k circuits. That the S gate can be represented by a circuit over $\mathcal{G}_k$ follows from Lemma 13.27 and the fact that $T_2 = T_k^{3^{k-2}}$. That the X can be represented by a Clifford+T_k circuit simply follows from the fact that $X = H^\dagger T_2^3 H$. That the CCX gate can be represented by a Clifford+T_k circuit follows from Lemma 13.26 and Theorem 13.16. □

The propositions above show that there is some flexibility in the definition of Clifford-cyclotomic gate sets and, in particular, that the gate set $\{X, CX, CCX, H, T_k\}$ is by no means minimal.

13.3 Rings and Groups

In this section, we discuss the rings and groups which will be important in the rest of the paper. In what follows, when u, u', and v are elements of a ring R, we write $u \equiv_v u'$ if u is congruent to u' modulo v, i.e., if $u - u' = rv$ for some $r \in R$.

The Ring $\mathbb{Z}[\omega_k]$

Definition 13.29. Let $k \geq 1$. The **primitive 3^k-th root of unity** $\omega_k \in \mathbb{C}$ is defined as $\omega_k = e^{2\pi i/3^k}$.

We have, for $k > 1$, $\omega_k^3 = \omega_{k-1}$, $\omega_k^{3^k} = 1$, $\omega_k^\dagger = \omega_k^{3^k-1}$, and $\omega_k^0 + \omega_k^1 + \ldots + \omega_k^{3^k-1} = 0$. We write ω for ω_1.

Definition 13.30. Let $k \geq 1$. The ring $\mathbb{Z}[\omega_k]$ of **cyclotomic integers of degree** 3^k is the smallest subring of $\mathbb{C}$ that contains ω_k.

The ring $\mathbb{Z}[\omega_k]$ can be defined in a variety of ways [Was82]. It will be useful for our purposes to note that $\mathbb{Z}[\omega] = \{a + b\omega \mid a, b \in \mathbb{Z}\}$, and that, for $k \geq 2$,

$$\mathbb{Z}[\omega_k] = \{a + b\omega_k + c\omega_k^2 \mid a, b, c \in \mathbb{Z}[\omega_{k-1}]\}.$$

Furthermore, the expression of an element of $\mathbb{Z}[\omega_k]$ as a linear combination of elements of $\mathbb{Z}[\omega_{k-1}]$ is unique. The ring $\mathbb{Z}[\omega_k]$ is closed under complex conjugation and, for $k \geq 2$, we have $\mathbb{Z}[\omega_{k-1}] \subseteq \mathbb{Z}[\omega_k]$.

Properties of $\mathbb{Z}[\omega]$

We now record some useful properties of $\mathbb{Z}[\omega]$. If $u = a + b\omega \in \mathbb{Z}[\omega]$, then

$$u^\dagger u = (a + b\omega)(a + b\omega^2) = a^2 + ab(\omega + \omega^2) + b^2 = a^2 - ab + b^2. \tag{13.31}$$

In particular, if $u \in \mathbb{Z}[\omega]$, then $u^\dagger u$ is a nonnegative integer, since the Euclidean norm of a complex number is always nonnegative.

Definition 13.32. We define $\lambda \in \mathbb{Z}[\omega]$ as $\lambda = 1 - \omega$.

By Equation 13.31, we have $\lambda^\dagger \lambda = 3$. Similarly, we have $\lambda^2 = 1 - 2\omega + \omega^2 = -3\omega$, so that $3 = -\lambda^2\omega^2$. Hence, $3 \equiv_\lambda 0$.

Proposition 13.33. *We have*

- $\mathbb{Z}[\omega]/(3) \cong \{0, 1, 2, \omega, 2\omega, 1 + \omega, 1 + 2\omega, 2 + \omega, 2 + 2\omega\} \cong \mathbb{Z}/(3) + \omega\mathbb{Z}/(3)$ *and*
- $\mathbb{Z}[\omega]/(\lambda) \cong \{0, 1, 2\} \cong \mathbb{Z}/(3)$.

Proof. The first item follows from the fact that $3 \equiv_3 0$. The second item follows from the fact that $3 \equiv_\lambda 0$ and the fact that $\omega \equiv_\lambda 1$. □

Proposition 13.34. *If $u \in \mathbb{Z}[\omega]$, then $u^\dagger u \equiv_\lambda 0$ or $u^\dagger u \equiv_\lambda 1$.*

Proof. Let $u = a + b\omega \in \mathbb{Z}[\omega]$. By Proposition 13.33, $\mathbb{Z}[\omega]/(\lambda) \cong \mathbb{Z}/(3)$. By Equation 13.31,

$$u^\dagger u = a^2 - ab + b^2 \equiv_\lambda a^2 + 2ab + b^2 = (a + b)^2.$$

Hence $u^\dagger u$ is a square modulo λ and therefore cannot be congruent to 2, since 0 and 1 are the only squares in $\mathbb{Z}/(3)$. □

Proposition 13.35. *If $u \in \mathbb{Z}[\omega]$, then $u \not\equiv_\lambda 0$ if and only if $u \equiv_3 \pm\omega^x$ for some $x \in \{0, 1, 2\}$.*

Proof. The table below lists the elements of $\mathbb{Z}[\omega]/(3)$ as given by Proposition 13.33, together with their residues modulo λ.

$\mathbb{Z}[\omega]/(3)$	$\mathbb{Z}[\omega]/(\lambda)$
0	0
1	1
2	2
ω	1
2ω	2
$1+\omega$	2
$1+2\omega$	0
$2+\omega$	0
$2+2\omega$	1

The statement then follows by inspection of the table, using the fact that $1+\omega = -\omega^2 \equiv_3 -\omega^2$ and $2 \equiv_3 -1$. □

Denominators

Definition 13.36. Let $k \geq 1$. The ring $\mathbb{Z}[1/3, \omega_k]$ is defined as $\mathbb{Z}[1/3, \omega_k] = \{u/3^\ell \mid u \in \mathbb{Z}[\omega_k] \text{ and } \ell \in \mathbb{N}\}$.

Because the elements of $\mathbb{Z}[\omega_k]$ can be expressed as linear combinations of elements of $\mathbb{Z}[\omega_{k-1}]$, the elements of $\mathbb{Z}[1/3, \omega_k]$ can similarly be expressed as linear combinations of elements of $\mathbb{Z}[1/3, \omega_{k-1}]$. In particular, for $k \geq 2$, every element u of $\mathbb{Z}[1/3, \omega_k]$ can be uniquely written as $u = a + b\omega_k + c\omega_k^2$ with $a, b, c \in \mathbb{Z}[1/3, \omega_{k-1}]$.

The ring $\mathbb{Z}[1/3, \omega_k]$ is the localization of $\mathbb{Z}[\omega_k]$ by the powers of 3. Alternatively, $\mathbb{Z}[1/3, \omega_k]$ can be thought of as the localization of $\mathbb{Z}[\omega_k]$ by the powers of λ. Indeed, since $3 = -\omega^2\lambda^2$, we have $3^{-\ell} = (-\omega^2\lambda^2)^{-\ell} = (-\omega)^\ell(\lambda)^{-2\ell}$. As a result, any element of $\mathbb{Z}[1/3, \omega_k]$ can be written as u/λ^ℓ for some $u \in \mathbb{Z}[\omega_k]$ and some $\ell \in \mathbb{N}$. We leverage this fact to define, in the usual way (see [AGR20; GS13; Gre14]), the notions of **λ-denominator exponent** and **least λ-denominator exponent**.

Definition 13.37. Any nonnegative integer ℓ such that $v \in \mathbb{Z}[1/3, \omega_k]$ can be written as $v = u/\lambda^\ell$ with $u \in \mathbb{Z}[\omega_k]$ is **λ-denominator exponent** of v. The smallest such ℓ is the **least λ-denominator exponent** of v and is denoted $\mathrm{lde}(v)$.

The notions of denominator exponent and least denominator exponent extend to matrices (and therefore to vectors) with entries in $\mathbb{Z}[1/3, \omega_k]$: an integer ℓ is a λ-denominator exponent of a matrix M if it is a λ-denominator exponent of all of the entries of M; the smallest such ℓ is the least λ-denominator exponent of M.

The Group $\mathrm{U}_n(\mathbb{Z}[1/3, \omega_k])$

Definition 13.38. We write $\mathrm{U}_n(\mathbb{Z}[1/3, \omega_k])$ for the group of n-dimensional unitary matrices with entries in $\mathbb{Z}[1/3, \omega_k]$ and $\mathrm{U}(\mathbb{Z}[1/3, \omega_k])$ for the collection of all unitary matrices with entries in $\mathbb{Z}[1/3, \omega_k]$.

13.4 Generators for $\mathrm{U}_n(\mathbb{Z}[1/3, \omega])$

Following [AGR20; GS13; Gre14; NC10], we use **m-level matrices** to define a subset of $\mathrm{U}_n(\mathbb{Z}[1/3, \omega])$, which we show is a generating set, and whose circuit representations we construct in Chapter 13.5.

Definition 13.39. The matrices (-1), (ω), X, and H are defined as follows:

$$(-1) = \begin{bmatrix} -1 \end{bmatrix}, \quad (\omega) = \begin{bmatrix} \omega \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \text{and} \quad H = \frac{-\omega^2}{\lambda} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{bmatrix}.$$

Definition 13.40. Let M be an $m \times m$ matrix, let $m \leq n$, and let $0 \leq x_1 < \ldots < x_m \leq n-1$. The **$m$-level matrix** $M_{[x_1,\ldots,x_m]}$ is the $n \times n$ matrix whose entries are given as follows

$$M_{[x_1,\ldots,x_m]_{i,j}} = \begin{cases} M_{i',j'} & \text{if } i = x_{i'} \text{ and } j = x_{j'}, \\ I_{i,j} & \text{otherwise.} \end{cases}$$

For example, for $n = 4$, we have

$$(\omega)_{[1]} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & \omega & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & 1 \end{bmatrix} \qquad \text{and} \qquad H_{[0,2,3]} = \frac{-\omega^2}{\lambda} \begin{bmatrix} 1 & \cdot & 1 & 1 \\ \cdot & \lambda/(-\omega^2) & \cdot & \cdot \\ 1 & \cdot & \omega & \omega^2 \\ 1 & \cdot & \omega^2 & \omega \end{bmatrix}.$$

When applied to a vector $|u\rangle$, the matrix $(\omega)_{[1]}$ acts as (ω) on the entry of index 1 and the matrix $H_{[0,2,3]}$ acts as H on the entries of index 0, 2, and 3.

Definition 13.41. We write $\mathcal{S}_n$ for the subset of $\mathrm{U}_n(\mathbb{Z}[1/3, \omega])$ defined as

$$\mathcal{S}_n = \{(-1)_{[x]}, (\omega)_{[x]}, X_{[x,y]}, H_{[x,y,z]} \mid 0 \leq x < y < z \leq n-1\}.$$

Lemma 13.42. *Let $u_0, u_1, u_2 \in \mathbb{Z}[\omega]$ be such that $u_0 \not\equiv_\lambda 0$, $u_1 \not\equiv_\lambda 0$, and $u_2 \not\equiv_\lambda 0$. Then there exists $x_0, x_1, x_2 \in \{0, 1, 2\}$ and $y_0, y_1, y_2 \in \{0, 1\}$ such that*

$$H(\omega)^{x_0}_{[0]}(\omega)^{x_1}_{[1]}(\omega)^{x_2}_{[2]}(-1)^{y_0}_{[0]}(-1)^{y_1}_{[1]}(-1)^{y_2}_{[2]} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} u'_0 \\ u'_1 \\ u'_2 \end{bmatrix}$$

for some $u'_0, u'_1, u'_2 \in \mathbb{Z}[\omega]$ such that $u'_0 \equiv_\lambda 0$, $u'_1 \equiv_\lambda 0$, and $u'_2 \equiv_\lambda 0$.

Proof. Let $j \in \{0, 1, 2\}$. Since $u_j \not\equiv_\lambda 0$, we have, by Proposition 13.35, $u_j \equiv_3 (-1)^{w_j}(\omega)^{z_j}$. Hence, setting $y_j = -w_j$ and $x_j = -z_j$, we get $(\omega)^{x_j}(-1)^{y_j} u_j \equiv_3 1$. Therefore,

$$(\omega)^{x_0}_{[0]}(\omega)^{x_1}_{[1]}(\omega)^{x_2}_{[2]}(-1)^{y_0}_{[0]}(-1)^{y_1}_{[1]}(-1)^{y_2}_{[2]} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} v_0 \\ v_1 \\ v_2 \end{bmatrix}$$

for some $v_0, v_1, v_2 \in \mathbb{Z}[\omega]$ such that $v_0 \equiv_3 v_1 \equiv_3 v_2 \equiv_3 1$. The result then follows by computation, since $v_0 + v_1 + v_2 \equiv_3 v_0 + \omega v_1 + \omega^2 v_2 \equiv_3 v_0 + \omega^2 v_1 + \omega v_2 \equiv_3 0$. □

Lemma 13.43. *Let $|u\rangle \in \mathbb{Z}[1/3, \omega]^n$ be a unit vector. If $\operatorname{lde}|u\rangle = 0$, then $|u\rangle = \pm\omega^x |j\rangle$ for some $0 \leq x \leq 2$ and some $0 \leq j \leq n - 1$.*

Proof. Let $|u\rangle \in \mathbb{Z}[1/3, \omega]^n$. Since $\operatorname{lde}|u\rangle = 0$, we have $|u\rangle \in \mathbb{Z}[\omega]^n$. Since $|u\rangle$ is a unit vector, we also have

$$1 = \langle u \rangle = \sum u_j^\dagger u_j$$

with $u_j \in \mathbb{Z}[\omega]$. Because each $u_j^\dagger u_j$ is a nonnegative integer, there must be exactly one j for which $u_j^\dagger u_j = 1$, while $u_{j'}^\dagger u_{j'} = 0$ for all $j' \neq j$. If $u_j^\dagger u_j = 1$ then $a_j^2 - a_j b_j + b_j^2 = 1$, and this equation can only be true if $a = \pm 1$ and $b = 0$, $a = 0$ and $b = \pm 1$, or $a = b = \pm 1$. In the first case, $|u\rangle = \pm |j\rangle$, in the second case, $|u\rangle = \pm\omega |j\rangle$, and in the third case, $|u\rangle = \pm\omega^2 |j\rangle$, □

Lemma 13.44. *Let $|u\rangle \in \mathbb{Z}[1/3, \omega]^n$ be a unit vector. If $\operatorname{lde}|u\rangle > 0$, then there exists $G_0, \ldots, G_q \in \mathcal{S}_n$ such that $\operatorname{lde}(G_q \cdots G_0 |u\rangle) < \operatorname{lde}|u\rangle$.*

Proof. Write $|u\rangle$ as $|v\rangle / \lambda^\ell$, with $\ell = \operatorname{lde}|u\rangle$. Since $\langle u \rangle = 1$ and $\lambda^\dagger \lambda = 3$, we get $3^\ell = \langle v \rangle = \sum v_j^\dagger v_j$. Hence, $\sum v_j^\dagger v_j \equiv_\lambda 0$. By Proposition 13.34, $v_j^\dagger v_j$ is either 0 or 1 modulo λ, and by Proposition 13.33, $\mathbb{Z}[\omega]/(\lambda) \cong \mathbb{Z}/(3)$. Thus, the number of v_j such that $v_j \not\equiv_\lambda 0$ must be a multiple of 3. Hence, we can group the entries of $|v\rangle$ into triples and apply Lemma 13.42 to each such triple. This maps $|u\rangle$ to some $|u\rangle'$ of lower least denominator exponent. □

Lemma 13.45. *Let $|u\rangle \in \mathbb{Z}[1/3,\omega]^n$ be a unit vector and let $0 \leq j \leq n-1$. Then there exists $G_0,\ldots,G_q \in \mathcal{S}_n$ such that $G_q \cdots G_0 |u\rangle = |j\rangle$.*

Proof. By induction on $\mathrm{lde}\,|u\rangle$. If $\mathrm{lde}(|u\rangle) = 0$, then, by Lemma 13.43, $|u\rangle = \pm\omega^x e_{j'}$ for some $0 \leq j' \leq n-1$ and some $0 \leq x \leq 2$. We can therefore reduce $|u\rangle$ to $|j\rangle$ by applying $(-1)_{[j']}$, $(\omega)_{[j']}$, and $X_{[j,j']}$ or $X_{[j',j]}$, as needed. If $\mathrm{lde}\,|u\rangle > 0$, then, by Lemma 13.44, there exists $G_p,\ldots,G_0 \in \mathcal{S}_n$ such that $\mathrm{lde}(G_p \cdots G_0 |u\rangle) < \mathrm{lde}(|u\rangle)$. We can then conclude by applying the induction hypothesis to $G_p \cdots G_0 |u\rangle$. □

Proposition 13.46. *Let U be an $n \times n$ matrix. Then $U \in \mathrm{U}_n(\mathbb{Z}[1/3,\omega])$ if and only if U can be written as a product of elements of $\mathcal{S}_n$.*

Proof. The right-to-left direction is immediate. For the left-to-right direction, consider the matrix $U^\dagger \in \mathrm{U}_n(\mathbb{Z}[1/3,\omega])$. Iteratively applying Lemma 13.45 to the columns of $U^\dagger$ yields a sequence $G_0,\ldots,G_q$ of elements of $\mathcal{S}_n$ such that

$$G_0 G_1 \cdots G q U^\dagger = I,$$

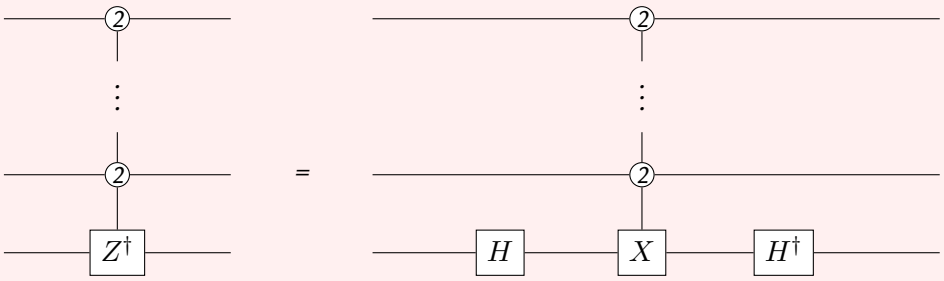
and we can therefore write U as $U = G_0 G_1 \cdots G_q$. □

13.5 Circuit Representations for the Elements of $\mathcal{S}_{3^n}$

We now provide explicit constructions for the elements of $\mathcal{S}_{3^n}$. We focus on the matrices in $\mathcal{S}_{3^n}$ where, writing each computational basis state on n qutrits as n trits, the levels are chosen to be those with the greatest value (taking the last qutrit to have the least significant trit). Indeed, these constructions can then be adapted to arbitrary levels by conjugating them by ternary classical reversible circuits using Theorem 13.16 and Proposition 13.19.

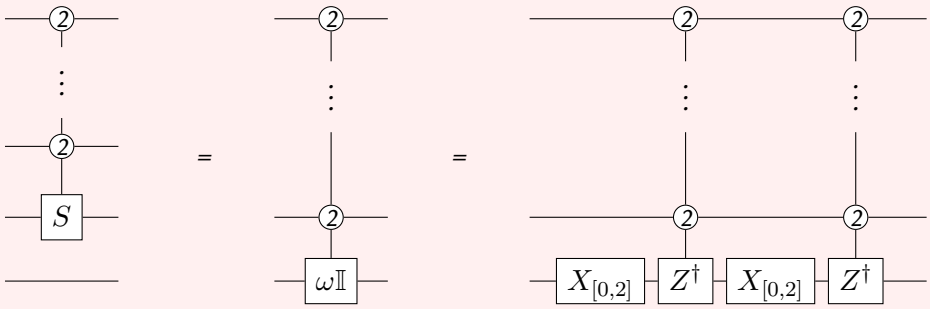
By Theorem 13.16 and Proposition 13.19, the multiply-controlled X gate can be expressed as a circuit over $\mathcal{G}_1$ using a single borrowed ancilla. We can therefore express the multiply-controlled Z gate as well, since $Z^\dagger = HXH^\dagger$.

Lemma 13.47. *We have:*



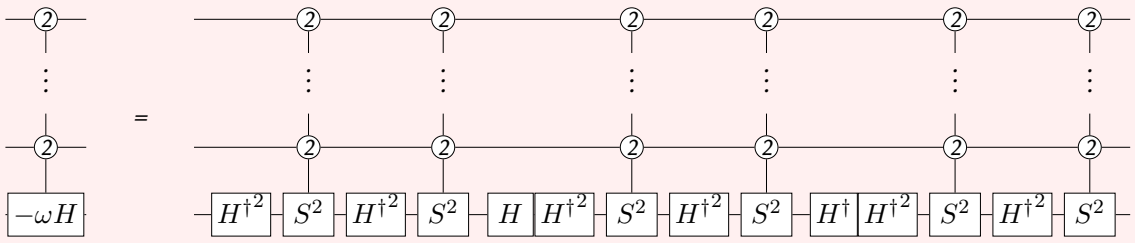
From this and the fact that $(\omega)_{[2]} = S$ when acting on a single qutrit, we can construct the 1-level matrix $(\omega)_{[x]}$ using a single borrowed ancilla.

Lemma 13.48. *We have:*

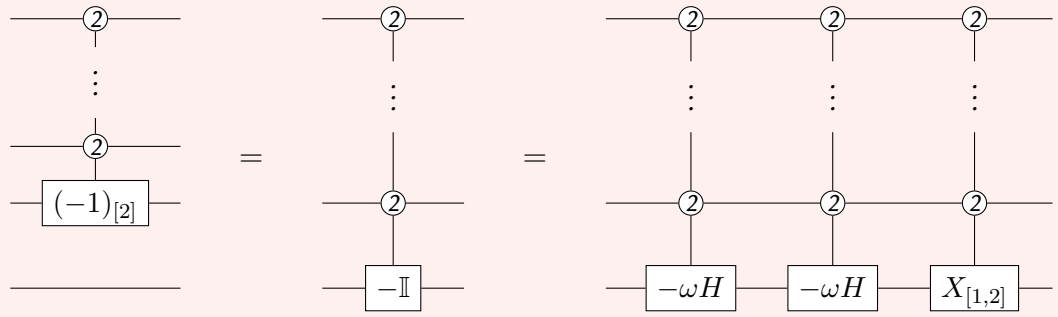


The next two lemmas let us construct the 1-level matrix $(-1)_{[x]}$. When acting on a single qutrit, this is the $(-1)_{[2]} = \mathrm{diag}(1, 1, -1)$ gate. This gate is also known as the **metaplectic** gate [Boc16; BRS17; CW15] and in earlier work, we referred to this gate as the R gate [GRvdWY22a].

Lemma 13.49. *We have:*

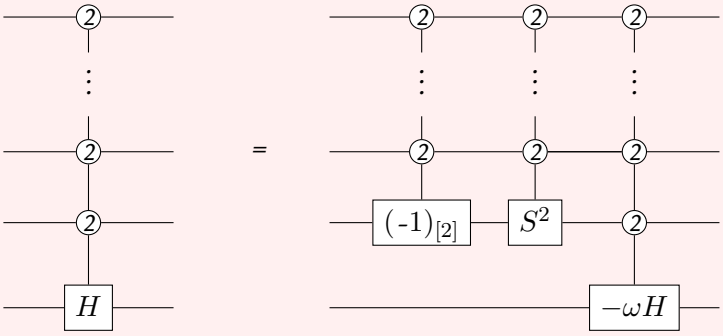


Lemma 13.50. *We have:*



We can now synthesize the 3-level matrix $H_{[x_1,x_2,x_3]}$ matrix over $\mathcal{G}_1$. To do this, apply Lemma 13.49 as well as the appropriate controlled global phase correction: a product of 1-level $\omega_{[x]}$ matrices and $(-1)_{[x]}$ matrices.

Lemma 13.51. *We have:*



We have now constructed all of the required 1-, 2-, and 3-level matrices (up to a permutation). We can therefore prove Proposition 13.60, which we restate below, making the ancilla requirements explicit.

Proposition 13.52. *If $U \in \mathcal{S}_{3^n}$, then U can be represented by a circuit over $\mathcal{G}_1$ using at most 2 borrowed ancillae. Explicitly,*

- $(-1)_{[x]}$ *requires 2 borrowed ancillae,*
- $(\omega)_{[x]}$ *requires 1 borrowed ancillae,*
- $X_{[x_1,x_2]}$ *requires 1 borrowed ancilla, and*
- $H_{[x_1,x_2,x_3]}$ *requires 1 borrowed ancillae.*

Proof. This follows from Lemmas 13.17, 13.48, 13.50, and 13.51, Proposition 13.19, , and Theorem 13.16. □

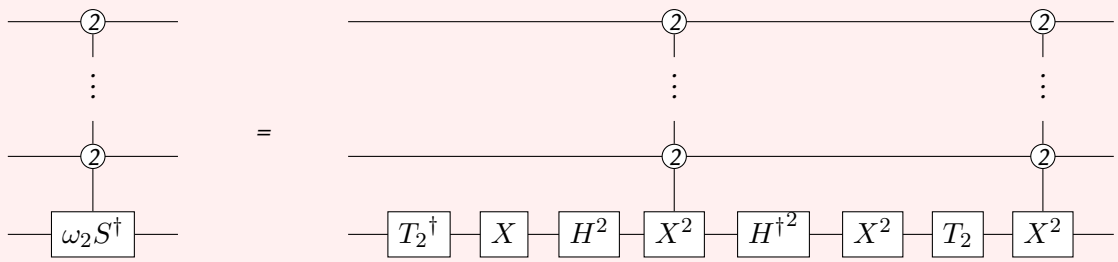
The number of ancillae required to represent the elements of $\mathcal{S}_{3^n}$ is, to a certain extent, an

artifact of the choice of gate set. For example, including the S gate or the $|0\rangle$-controlled X gate in the gate set would lower the ancilla-count for some of the elements of $\mathcal{S}_{3^n}$.

The proposition above shows that the matrices that can be represented by a multiqutrit circuit over the Clifford+$(-1)_{[2]}$ gate set (also known as the **Clifford+R** or the **metaplectic** gate set) are a subset of those representable by a circuit over $\mathcal{G}_1$. At the time of writing, we do not know whether this inclusion is strict, although the conjecture in [BCRS16] that not all ternary classical reversible gates can be exactly represented over the Clifford+$(-1)_{[2]}$ gate set lends credence to this idea.

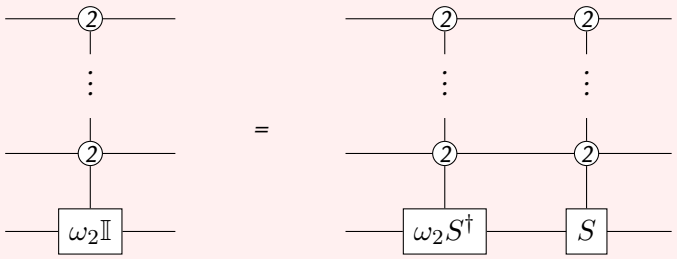
If a matrix can be represented by a circuit over $\mathcal{G}_k$, it can also be represented by a circuit over $\mathcal{G}_{k+1}$. It therefore follows from the proposition above that all of the elements of $\mathcal{S}_{3^n}$ can be represented by a circuit over $\mathcal{G}_2$. We close this section by showing that the 1-level matrix $(\omega_2)_{[x]}$ can be represented by a circuit over $\mathcal{G}_2$ and by providing further generalizations of the above constructions. This paves the way for a direct proof of exact synthesis for Clifford+T circuits (rather than the more indirect one using catalytic embeddings, as in Theorem 13.72). Over $\mathcal{G}_2$, the ancilla requirements are lowered, since the $|0\rangle$-controlled X gate can be represented by an ancilla-free circuit by Lemma 13.26. To construct $(\omega_2)_{[x]}$, we first build a modification of $(\omega)_{[x]}$ which differs by a controlled global phase of ω_2.

Lemma 13.53. *We have:*



We note that unlike the construction in Lemma 13.48 which required one (additional) borrowed ancilla, this construction requires no (additional) borrowed ancillae. By combining the construction of Lemma 13.53 and that of Lemma 13.51, we can therefore represent $H_{[x_1,x_2,x_3]}$ without ancillae. Similarly, by combining the construction of Lemma 13.53 and that of Lemma 13.50, we can represent $(-1)_{[x]}$ using a single borrowed ancillae. Finally, $(\omega_2)_{[x]}$ can be constructed as in the next lemma using 2 borrowed ancillae.

Lemma 13.54. *We have:*



Proposition 13.55. *The 1-, 2-, and 3-level matrices $(-1)_{[x]}$, $(\omega_2)_{[x]}$, $X_{[x_1,x_2]}$, and $H_{[x_1,x_2,x_3]}$ can be represented by a circuit over $\mathcal{G}_2$ using at most 2 borrowed ancillae. Explicitly,*

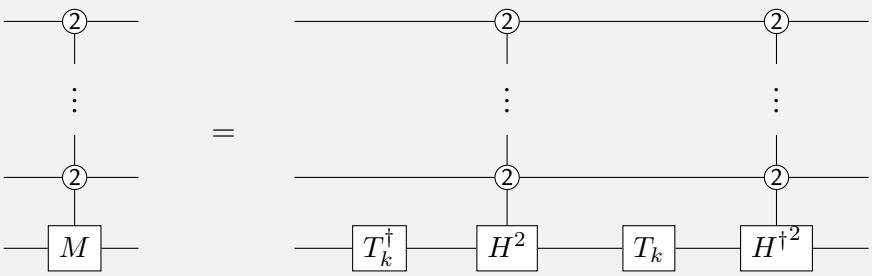
- *$(-1)_{[x]}$ requires 1 borrowed ancilla,*
- *$(\omega_2)_{[x]}$ requires 2 borrowed ancillae,*
- *$X_{[x_1,x_2]}$ requires 0 borrowed ancillae, and*
- *$H_{[x_1,x_2,x_3]}$ requires 0 borrowed ancillae.*

Proof. This follows from Lemmas 13.26, 13.50, 13.51, 13.53, and 13.54 and Theorem 13.16. □

We can generalize the above construction to Clifford-cyclotomic gate sets of higher degree.

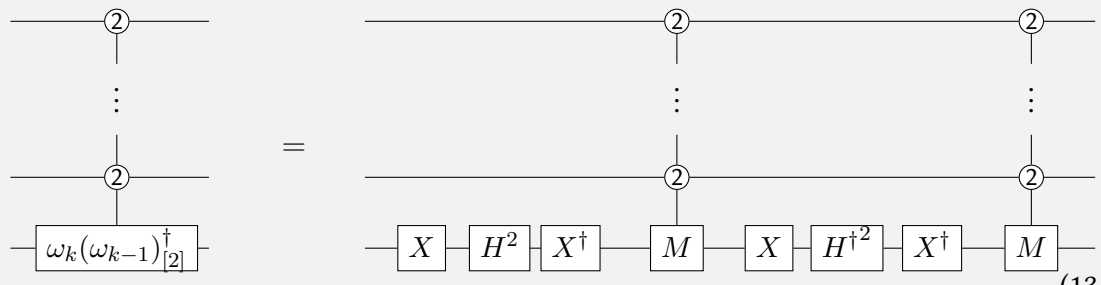
Proposition 13.56. *Let $k \geq 1$. The 1-level matrix $(\omega_k)_{[x]}$ can be represented by a circuit over $\mathcal{G}_k$ using k borrowed ancillae.*

Proof. First, we build the multiply-controlled M gate, where $M = \mathrm{diag}(1, \omega_k, \omega_k^\dagger)$.



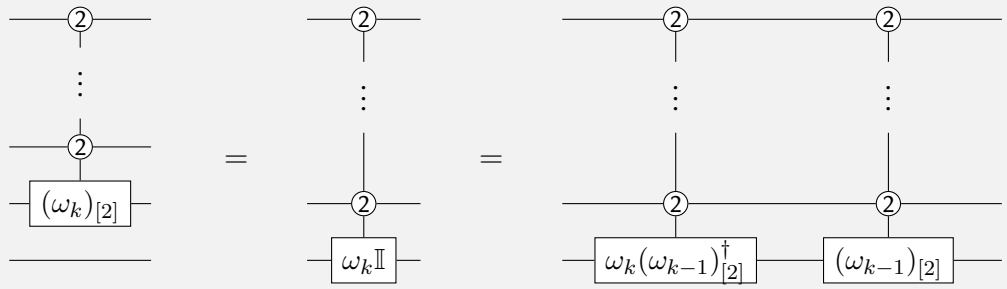
(13.57)

Then, we can build the multiply-controlled one-qutrit gate $\omega_k(\omega_{k-1})^\dagger_{[2]} = \omega_k \text{diag}(1, 1, \omega_{k-1}^\dagger)$.



(13.58)

Finally, we can combine this with the multiply-controlled one-qutrit gate $(\omega_{k-1})_{[2]} = \text{diag}(1, 1, \omega_{k-1})$ to get $(\omega_k)_{[2\ldots2]}$.



(13.59)

Since a single borrowed ancilla suffices to build (ω) and 2 borrowed ancillae suffice to build (ω_2), the above equation shows that k ancillae suffice to build (ω_k). □

13.6 Exact Synthesis of Toffoli+Hadamard Circuits

Let $\mathcal{G}$ be a set of quantum gates. A unitary matrix U can be **represented by a circuit over** $\mathcal{G}$ if there exists a circuit C over $\mathcal{G}$ such that, for any state $|u\rangle$, we have $C|u\rangle = U|u\rangle$. The circuit may use ancillary qutrits, but these must start and end the computation in the same state. If that state can be arbitrary, the ancillary qutrits are said to be **borrowed**; if that state is required to be $|0\rangle$, the ancillary qutrits are said to be **fresh**. Unless otherwise specified, ancillae are assumed to be fresh. Note that if a matrix can be represented by a circuit using m borrowed ancillae, then it can also be represented by a circuit using m fresh ancillae.

Recall that the Clifford-cyclotomic gate set $\mathcal{G}_k$ is defined as $\mathcal{G}_k = \{X, CX, CCX, H, T_k\}$. In Section 13.2 we prove that $\mathcal{G}_1$ is equivalent to the Toffoli+Hadamard gate set, up to two borrowed ancillae and that, when $k \geq 2$, $\mathcal{G}_k$ is equivalent to the Clifford+T_k gate set $\{H, S, CX, T_k\}$, up to a single borrowed ancilla. The next proposition shows that all of the elements of $\mathcal{S}_{3^n}$ can be represented by a circuit over $\mathcal{G}_1$ using no more than 2 borrowed ancillae. The proof of the proposition can be found in Section 13.5.

Proposition 13.60. *If $U \in \mathcal{S}_{3^n}$, then U can be represented by a circuit over $\mathcal{G}_1$ using at most 2 borrowed ancillae.*

Using Proposition 13.60 we are now in a position to define an exact synthesis algorithm for multiqutrit Toffoli+Hadamard circuits.

Theorem 13.61. *If $U \in \mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_1])$, then U can be represented by an n-qutrit circuit over $\mathcal{G}_1$ using at most 2 ancillae.*

Proof. By Proposition 13.46, $\mathcal{S}_{3^n}$ generates $\mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_1])$. Hence, it is sufficient to show that the elements of $\mathcal{S}_{3^n}$ can be represented by an n-qutrit circuit over $\mathcal{G}_1$. This follows from Proposition 13.60, since if 2 borrowed ancillae suffice to construct a circuit for U, then 2 fresh ancillae are also sufficient for this purpose. □

13.7 Catalytic Embeddings

Definition 13.62. Let $\mathcal{U}$ and $\mathcal{V}$ be collections of unitaries. An **m-dimensional catalytic embedding** of $\mathcal{U}$ into $\mathcal{V}$ is a pair $(\phi, |c\rangle)$ of a function $\phi : \mathcal{U} \to \mathcal{V}$ and a vector $|c\rangle \in \mathbb{C}^m$ such that if $U \in \mathcal{U}$ has dimension n then $\phi(U) \in \mathcal{V}$ has dimension nm and

$$\phi(U)(|u\rangle \otimes |c\rangle) = (U\,|u\rangle) \otimes |c\rangle$$

for every $|u\rangle \in \mathbb{C}^n$. The vector $|c\rangle$ is the **catalyst** of the catalytic embedding $(\phi, |c\rangle)$. We sometimes express the fact that $(\phi, |c\rangle)$ is a catalytic embedding of $\mathcal{U}$ into $\mathcal{V}$ by writing $(\phi, |c\rangle) : \mathcal{U} \to \mathcal{V}$.

Definition 13.63. Let $(\phi, |c\rangle) : \mathcal{U} \to \mathcal{V}$ and $(\phi', |c\rangle') : \mathcal{V} \to \mathcal{W}$ be catalytic embeddings of dimension m and m', respectively. The **concatenation** of $(\phi, |c\rangle)$ and $(\phi', |c\rangle')$ is the $m'm$-dimensional catalytic embedding $(\phi', |c\rangle') \circ (\phi, |c\rangle)$ defined by $(\phi', |c\rangle') \circ (\phi, |c\rangle) = (\phi' \circ \phi, |c\rangle \otimes |c\rangle')$.

The concatenation of catalytic embeddings is associative and the catalytic embedding $(1_{\mathcal{U}}, [1]) : \mathcal{U} \to \mathcal{U}$ acts as an identity for concatenation.

Definition 13.64. Let $k \geq 2$. We define Ω_k and $|c_k\rangle$ as

$$\Omega_k = \begin{bmatrix} \cdot & \cdot & \omega_{k-1} \\ 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \end{bmatrix} \qquad \text{and} \qquad |c_k\rangle = \frac{1}{\lambda} \begin{bmatrix} 1 \\ \omega_k^{-1} \\ \omega_k^{-2} \end{bmatrix}.$$

The matrix Ω_k is unitary and the vector $|c_k\rangle$ is an eigenvector of Ω_k for eigenvalue ω_k. Indeed, since $\omega_{k-1} = \omega_k^3$, we have

$$\Omega_k |c_k\rangle = \frac{1}{\lambda} \begin{bmatrix} \cdot & \cdot & \omega_{k-1} \\ 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \end{bmatrix} \begin{bmatrix} 1 \\ \omega_k^{-1} \\ \omega_k^{-2} \end{bmatrix} = \frac{1}{\lambda} \begin{bmatrix} \omega_k \\ 1 \\ \omega_k^{-1} \end{bmatrix} = \frac{\omega_k}{\lambda} \begin{bmatrix} 1 \\ \omega_k^{-1} \\ \omega_k^{-2} \end{bmatrix} = \omega_k |c_k\rangle . \tag{13.65}$$

Now recall from Section 13.3 that, for $k \geq 2$, every $u \in \mathbb{Z}[1/3, \omega_k]$ can be written uniquely as a linear combination of the form $u = a + b\omega_k + c\omega_k^2$, where $a, b, c \in \mathbb{Z}[1/3, \omega_{k-1}]$. Therefore, every matrix U over $\mathbb{Z}[1/3, \omega_k]$ can be uniquely written as $U = A + B\omega_k + C\omega_k^2$, where A, B, and C are matrices over $\mathbb{Z}[1/3, \omega_{k-1}]$. We use this fact below to define a function $\mathrm{U}(\mathbb{Z}[1/3, \omega_k]) \to \mathrm{U}(\mathbb{Z}[1/3, \omega_{k-1}])$.

Proposition 13.66. *Let $k \geq 2$. The assignment*

$$A + B\omega_k + C\omega_k^2 \longmapsto A \otimes I + B \otimes \Omega_k + C \otimes \Omega_k^2$$

defines a function $\phi_k : \mathrm{U}(\mathbb{Z}[1/3, \omega_k]) \to \mathrm{U}(\mathbb{Z}[1/3, \omega_{k-1}])$.

Proof. Let $U \in \mathrm{U}(\mathbb{Z}[1/3, \omega_k])$ and write U as $U = A + B\omega_k + C\omega_k^2$ for some matrices A, B, and C over $\mathbb{Z}[1/3, \omega_{k-1}]$. Now let $U' = A \otimes I + B \otimes \Omega_k + C \otimes \Omega_k^2$. It is clear that U' is a matrix with entries in $\mathbb{Z}[1/3, \omega_{k-1}]$. We now show that U' is unitary. Since U is unitary and since $U = A + B\omega_k + C\omega_k^2$, we can express the equation $U^\dagger U = I$ in terms of A, B, and C. Using $\omega_k^\dagger = \omega_{k-1}^\dagger \omega_k^2$, this yields

$$(A^\dagger A + B^\dagger B + C^\dagger C) + (A^\dagger B + B^\dagger C + C^\dagger A\omega_{k-1}^\dagger)\omega_k + (A^\dagger C + B^\dagger A\omega_{k-1}^\dagger + C^\dagger B\omega_{k-1}^\dagger)\omega_k^2 = I.$$

Hence, $A^\dagger A + B^\dagger B + C^\dagger C = I$ and $A^\dagger B + B^\dagger C + C^\dagger A\omega_{k-1}^\dagger = A^\dagger C + B^\dagger A\omega_{k-1}^\dagger + C^\dagger B\omega_{k-1}^\dagger = 0$. Now note that $\Omega_k^\dagger = \omega_{k-1}^\dagger \Omega_k^2$, so that $U'^\dagger U'$ is equal to

$$(A^\dagger A + B^\dagger B + C^\dagger C) \otimes I + (A^\dagger B + B^\dagger C + C^\dagger A\omega_{k-1}^\dagger) \otimes \Omega_k + (A^\dagger C + B^\dagger A\omega_{k-1}^\dagger + C^\dagger B\omega_{k-1}^\dagger) \otimes \Omega_k^2.$$

Hence, $U'^\dagger U' = I$. Reasoning analogously shows that $U' U'^\dagger = I$, so that U' is indeed

unitary. □

Proposition 13.67. *Let $k \geq 2$. The pair $(\phi_k, |c_k\rangle)$ is a 3-dimensional catalytic embedding of $\mathrm{U}(\mathbb{Z}[1/3, \omega_k])$ into $\mathrm{U}(\mathbb{Z}[1/3, \omega_{k-1}])$.*

Proof. By Proposition 13.66, $\phi_k : \mathrm{U}(\mathbb{Z}[1/3, \omega_k]) \to \mathrm{U}(\mathbb{Z}[1/3, \omega_{k-1}])$ is a function and, by construction, if $U \in \mathrm{U}(\mathbb{Z}[1/3, \omega_k])$ has dimension n, then $\phi_k(U)$ has dimension $3n$. Moreover, if $|u\rangle \in \mathbb{C}^n$, then

$$\begin{aligned}
\phi_k(U)(|u\rangle \otimes |c_k\rangle) &= (A \otimes I + B \otimes \Omega_k + C \otimes \Omega_k^2)(|u\rangle \otimes |c_k\rangle) \\
&= A \otimes I(|u\rangle \otimes |c_k\rangle) + B \otimes \Omega_k(|u\rangle \otimes |c_k\rangle) + C \otimes \Omega_k^2(|u\rangle \otimes |c_k\rangle) \\
&= A\,|u\rangle \otimes I\,|c_k\rangle + B\,|u\rangle \otimes \Omega_k\,|c_k\rangle + C\,|u\rangle \otimes \Omega_k^2\,|c_k\rangle \\
&= A\,|u\rangle \otimes |c_k\rangle + B\,|u\rangle \otimes \omega_k\,|c_k\rangle + C\,|u\rangle \otimes \omega_k^2\,|c_k\rangle \\
&= A\,|u\rangle \otimes |c_k\rangle + \omega_k B\,|u\rangle \otimes |c_k\rangle + \omega_k^2 C\,|u\rangle \otimes |c_k\rangle \\
&= (A\,|u\rangle + \omega_k B\,|u\rangle + \omega_k^2 C\,|u\rangle) \otimes |c_k\rangle \\
&= (U\,|u\rangle) \otimes |c_k\rangle\,.
\end{aligned}$$

Hence, $(\phi_k, |c_k\rangle)$ is a catalytic embedding. □

Remark 13.68. The catalytic embedding constructed in Proposition 13.66 and Proposition 13.67 takes advantage of the fact that the matrix Ω_k and the algebraic number ω_k have many properties in common. Importantly, the polynomial $x^3 - \omega_{k-1}$ is both the characteristic polynomial of Ω_k and the minimal polynomial of ω_k over the ring $\mathbb{Z}[1/3, \omega_{k-1}]$. This construction generalizes to many other rings of interest (see [ACG+23]).

Corollary 13.69. *Let $k \geq 2$. There is a 3^{k-1}-dimensional catalytic embedding $(\phi, |c\rangle) : \mathrm{U}(\mathbb{Z}[1/3, \omega_k]) \to \mathrm{U}(\mathbb{Z}[1/3, \omega])$.*

Proof. Applying Proposition 13.67 repeatedly yields a sequence of catalytic embeddings

$$\mathrm{U}(\mathbb{Z}[1/3, \omega_k]) \xrightarrow{(\phi_k, |c_k\rangle)} \cdots \xrightarrow{(\phi_3, |c_3\rangle)} \mathrm{U}(\mathbb{Z}[1/3, \omega_2]) \xrightarrow{(\phi_2, |c_2\rangle)} \mathrm{U}(\mathbb{Z}[1/3, \omega]).$$

Concatenating the catalytic embeddings in this sequence yields the desired result. □

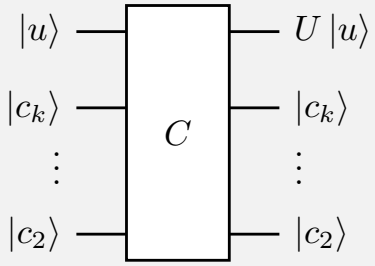
Note that the catalyst $|c\rangle$ in the catalytic embedding $(\phi, |c\rangle)$ of Corollary 13.69 is the product state $|c\rangle = |c_2\rangle \otimes \cdots \otimes |c_k\rangle$.

13.8 Exact Synthesis of Clifford-Cyclotomic Circuits

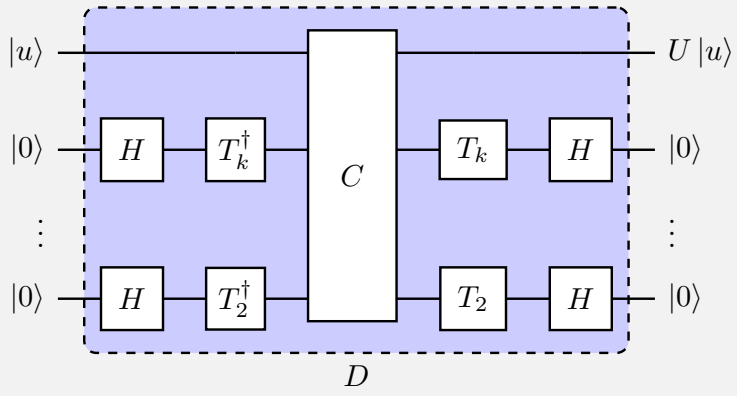
We can now prove our main result, which will follow straightforwardly from the results of Sections 13.5, 13.6, and 13.7.

Proposition 13.70. *Let $k \geq 2$. If $U \in \mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$, then U can be represented by an n-qutrit circuit over $\mathcal{G}_k$ using at most $k+1$ ancillae.*

Proof. Let $U \in \mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$ and let $(\phi, |c\rangle)$ be the catalytic embedding constructed in Corollary 13.69, with $|c\rangle = |c_2\rangle \otimes \cdots \otimes |c_k\rangle$. We then have $\phi(U) \in \mathrm{U}_{3^{n+k-1}}(\mathbb{Z}[1/3, \omega])$, so that, by Theorem 13.61, $\phi(U)$ can be represented by an $(n+k-1)$-qutrit circuit C over $\mathcal{G}_1$ using at most 2 fresh ancillae. By Definition 13.62, the action of $\phi(U)$ on an input of the form $|u\rangle \otimes |c_2\rangle \otimes \cdots \otimes |c_k\rangle$ can be depicted as below (where the ancillary qutrits used in C, if any, are omitted).



But, for $2 \leq \ell \leq k$, we have $|c_\ell\rangle = T_\ell^\dagger H |0\rangle$ and $T_\ell^\dagger = (T_k^\dagger)^{3^{k-\ell}}$. Hence, we can construct the following circuit over $\mathcal{G}_k$.



Since all of the ancillae in D (including the ancillae potentially present in C) start and end the computation in the $|0\rangle$ state, then D is a circuit over $\mathcal{G}_k$ which represents U and uses at most $k+1$ (fresh) ancillae, as desired. □

Remark 13.71. The circuit constructed in Proposition 13.70 actually use $k-1$ fresh ancillae and no more than 2 borrowed ancillae. For brevity, we simply stated the proposition in terms of fresh ancillae. One can amend the constructions in Section 13.5 to reduce the total ancilla-count from $k+1$ to k, at the cost of requiring all ancillae to be fresh.

Theorem 13.72. *Let $k \geq 1$ and let U be a $3^n \times 3^n$ unitary matrix. Then U can be represented by an n-qutrit circuit over $\mathcal{G}_k$ if and only if $U \in \mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$. Moreover, $k+1$ ancillae are always sufficient to construct a circuit for U.*

Proof. The left-to-right direction is a consequence of the fact that the entries of the elements of $\mathcal{G}_k$ lie in the ring $\mathbb{Z}[1/3, \omega_k]$. The right-to-left direction is given by Theorem 13.61 and Proposition 13.70. □

13.9 Circuit Complexity

The proof of Theorem 13.72 is constructive: it provides an algorithm to construct a circuit for a given matrix. In this section, we briefly discuss the complexity of the resulting circuit, reasoning as in [AGLR23; GS13]. We start by considering Proposition 13.46 before turning to Theorem 13.72.

Lemma 13.73. *Let $U \in \mathrm{U}_m(\mathbb{Z}[1/3, \omega])$ and let $\ell = \mathrm{lde}(U)$. The algorithm of Proposition 13.46 expresses U as a product of $O(2^m \ell)$ elements of $\mathcal{S}_m$ in the worst case.*

Proof. Consider the first column of U. In the worst case, its least denominator exponent is ℓ. To reduce this least denominator exponent by one requires $O(m)$ operations. Hence, reducing the first column of U completely requires $O(\ell m)$ operations in the worst case. The reduction of the first column may increase the least denominator exponent of the second column from ℓ to 2ℓ, since each entry of the second column may be affected by up to ℓ 3-level matrices in the course of this first reduction. Once the first column has been reduced, the second column may still have $m-1$ nonzero entries. Reducing the second column will hence require $O(2\ell(m-1))$ operations in the worst case. In general, reducing the j-th column will require $O(2^{j-1}\ell(m-j))$ operations in the worst case so that the overall reduction of U requires at most

$$O\left(\sum_{i=0}^{n-1} 2^i \ell(m-i)\right)$$

operations. Simplifying the resulting sum yields a total of $O(2^m \ell)$ operations. □

Theorem 13.74. *Let $U \in \mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$ and let $\ell = \mathrm{lde}(U)$. The algorithm of Theorem 13.72 represents U as a circuit of $O((n+k)2^{3^{n+k-1}}\ell)$ gates in the worst case.*

Proof. The algorithm of Theorem 13.72 uses the catalytic embedding $(\phi, |c\rangle)$ of Corollary 13.69 to construct a matrix $\phi(U)$ over $\mathbb{Z}[1/3, \omega]$. The dimension of $\phi(U)$ is 3^{n+k-1} and its least denominator exponent is no more than ℓ. Hence, by Lemma 13.73, the algorithm of Proposition 13.46 will express $\phi(U)$ as a product of no more than $O(2^{3^{n+k-1}}\ell)$ elements of $\mathcal{S}_{3^{n+k-1}}$. It follows from the circuit constructions given in Section 13.5, that each element of $\mathcal{S}_{3^{n+k-1}}$ can be represented by a circuit consisting of $O(n+k)$ gates. Hence, the circuit produced by Theorem 13.72 consists of no more than $O((n+k)2^{3^{n+k-1}}\ell)$ gates. □

Conclusion

We showed that the matrices that can be exactly represented by an n-qutrit circuit over the Clifford-cyclotomic gate set of degree 3^k are precisely the elements of $\mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$. Moreover, we showed that no more than $k+1$ ancillae are required to construct a circuit for an element of $\mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$.

Our proof contains an algorithm for synthesizing a circuit over $\mathcal{G}_k$, given a matrix in $\mathrm{U}_{3^n}(\mathbb{Z}[1/3, \omega_k])$. However, in this first attainment of exact circuit synthesis, there is significant room for improvement with the efficiency of the algorithm, if the qubit case is any indicator. The original n-qubit Clifford+T exact synthesis algorithm utilized $O(3^{2^n}nk)$ elementary Clifford+T gates [GS13], which was improved to $O(4^n nk)$ by Kliuchnikov [Kli13]. Amy, Glaudell, Li, and Ross then showed that Kliuchnikov's technique could also be applied to the qubit Toffoli+Hadamard gate set, improving the elementary Toffoli+Hadamard exact circuit synthesis gate count from $O(2^{2^n}nk)$ to $O(2^{2n}nk)$ [AGLR23]. These results for the qubit case suggest a promising outlook for exponential improvement in the efficiency of multiqutrit exact circuit synthesis algorithms.

The other area for improvement is to reduce the ancillae count. As shown in Sections 13.2 and 13.5, some of this is an artifact of choosing a more minimal gate set. Minimizing the number of generating gates is for instance conducive to implementation, as fewer elementary gates are needed to perform fault-tolerantly in order to achieve universality for all unitary matrices over the given ring. Having said this, with climbing higher in the Clifford-cyclotomic hierarchy, the bulk of the ancillae come from the use of catalytic embeddings, which would require new techniques to make more efficient. On this note, future work could be to characterize the matrices representable by ancilla-free circuits, as has been done for qubit matrices in Refs. [AGR20; GS13]. However, these should be different for qutrits. For one, while the qubit controlled T gate does not have an ancilla-free implementation in qubit Clifford+T, the qutrit

T gate does in qutrit Clifford+T [YvdW22].

The other natural direction to pursue would be extending these results to higher-dimensional qudits. Our findings thus far indicate that certain techniques here do not directly apply to dimension 5. Considering that the Clifford-cyclotomic hierarchy is consistently definable for all prime qudit dimensions, it is expected that a number of these results will generalize, but to prove this might require new ideas.

Chapter summary

We introduced two new qudit gate sets for any odd prime qudit dimension d: Clifford+$\sqrt[d]{Z}$ and Toffoli+Hadamard. These are both approximately universal for quantum computation, and exactly universal for any d-ary classical reversible circuit. The $\sqrt[d]{Z}$ gate is in the dth level of the Clifford hierarchy, and is a generalization of the qutrit T gate. The qudit Toffoli+Hadamard gate set seems to have all the properties you would want considering the qubit setting, and even more perks that the qubit version does not have. In particular, a two-qubit gate, the $|0\rangle$-controlled X gate, can take on the role that the three-qubit Toffoli gate does, in generating all the d-ary classical reversible circuits. Moreover, while the qubit Toffoli+Hadamard gate set is only computationally universal, the qudit Toffoli+Hadamard gate set is more powerful as it is approximately universal for quantum computation.

The other highlight of this chapter an exciting resolution to the multiqutrit Clifford+T exact synthesis problem, since the qubit version was solved in 2013 [GS13]. Perhaps the most novel idea that made this work, was to recognize that the Toffoli+Hadamard gate set was below (and a strict subset of) Clifford+T in the Clifford-cyclotomic hierarchy of gate sets. This enabled the use of catalytic embeddings to reduce to two subproblems: Toffoli+Hadamard exact synthesis, and how to utilize this result to synthesize circuits over gate sets higher in the Clifford-cyclotomic hierarchy. We ended the chapter by discussing future directions continuing these investigations.

Part VI

Thesis End

14

What didn't make it into the thesis

This brings the thesis to a close. In addition to the work in this thesis, this PhD witnessed a number of successes in generalizing quantum graphical calculi beyond qubits. A number of these I contributed to, opting not to include in this thesis, include:

- A unique normal form for the stabilizer fragment for prime-dimensional qudits [PBC+23]
- Completeness for arbitrary finite dimensions of the ZXW-calculus [PWS+23], a long-awaited result which led to subsequent completeness of the ZX-calculus for all of finite-dimensional Hilbert space [PSW25]
- A lifting theorem to prove equalities of linear operators in bosonic Fock space from their truncated equivalent in finite-dimensional ZXW-calculus [dFSP+23]
- The Focked-up ZX-calculus, a graphical calculus for continuous-variable quantum computation which also supports operations on the discrete Fock basis [SYG24]
- An end-to-end compilation framework for fusion-based photonic architectures, from optical circuits to mixed-state qubit ZX-diagrams to dataflow programming to universal measurement-based quantum computation [dFPYC24; dFPC+26]
- Completeness for qubit controlled states and operators, which go beyond concrete generators to higher-order maps [AYY26]
- A ZX-calculus perspective on fermion-to-qubit mappings [MSY25]
- A graphical framework for SU(2) representation theory [WES+25]

- Construction of QEC codes with built-in transversal AND gate [LY26]

The outlook looks bright for higher-dimensional, fault-tolerant, and mixed-state quantum computation adaptable to different computational models and architectures.

Bibliography

[ABLS01] Antonio Acín, Dagmar Bruß, Maciej Lewenstein, and Anna Sanpera. "Classification of mixed three-qubit states". In: *Physical Review Letters* 87.4 (2001), p. 040401.

[ABNO22] Srinivasan Arunachalam, Sergey Bravyi, Chinmay Nirkhe, and Bryan O'Gorman. "The Parametrized Complexity of Quantum Verification". en. In: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. DOI: 10.4230/LIPICS.TQC.2022.3.

[AC04] Samson Abramsky and Bob Coecke. "A categorical semantics of quantum protocols". In: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004.* IEEE. 2004, pp. 415–425.

[ACB12a] Hussain Anwar, Earl T Campbell, and Dan E Browne. "Qutrit magic state distillation". In: *New Journal of Physics* 14.6 (June 2012), p. 063006. DOI: 10.1088/1367-2630/14/6/063006.

[ACB12b] Hussain Anwar, Earl T Campbell, and Dan E Browne. "Qutrit magic state distillation". In: *New Journal of Physics* 14.6 (June 2012), p. 063006. DOI: 10.1088/1367-2630/14/6/063006.

[ACG+23] M. Amy, M. Crawford, A. N. Glaudell, M. L. Macasieb, S. S. Mendelson, and N. J. Ross. *Catalytic Embeddings of Quantum Circuits.* 2023. arXiv: 2305.07720 [quant-ph].

[ADP14] Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. "Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes". In: *Phys. Rev. Lett.* 113 (8 Aug. 2014), p. 080501. DOI: 10.1103/PhysRevLett.113.080501.

[AG04] Scott Aaronson and Daniel Gottesman.
"Improved simulation of stabilizer circuits".
In: *Physical Review A* 70.5 (2004), p. 052328.

[AGK+24] Matthew Amy, Andrew N. Glaudell, Shaun Kelso, William Maxwell, Samuel S. Mendelson, and Neil J. Ross.
Exact Synthesis of Multiqubit Clifford-Cyclotomic Circuits. 2024.
arXiv: 2311.07741 [quant-ph].

[AGLR23] Matthew Amy, Andrew N. Glaudell, Sarah Meng Li, and Neil J. Ross.
"Improved Synthesis of Toffoli-Hadamard Circuits".
In: *Reversible Computation*. Ed. by Martin Kutrib and Uwe Meyer.
Cham: Springer Nature Switzerland, 2023, pp. 169–209.
ISBN: 978-3-031-38100-3.

[AGR20] Matthew Amy, Andrew N. Glaudell, and Neil J. Ross.
"Number-theoretic characterizations of some restricted Clifford+T circuits".
In: *Quantum* 4 (2020), p. 252. ISSN: 2521-327X.
DOI: 10.22331/q-2020-04-06-252.

[AMMR13] Matthew Amy, Dmitri Maslov, Michele Mosca, and Martin Roetteler.
"A Meet-in-the-Middle Algorithm for Fast Synthesis of Depth-Optimal Quantum Circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32.6 (2013), pp. 818–830.
DOI: 10.1109/TCAD.2013.2244643.

[Amy19] Matthew Amy.
"Towards Large-scale Functional Verification of Universal Quantum Circuits".
In: *Proceedings of the 15th International Conference on Quantum Physics and Logic, Halifax, Canada, 3-7th June 2018*.
Ed. by Peter Selinger and Giulio Chiribella. Vol. 287.
Electronic Proceedings in Theoretical Computer Science.
Open Publishing Association, 2019, pp. 1–21. DOI: 10.4204/EPTCS.287.1.

[ASR+15] B. E. Anderson, H. Sosa-Martinez, C. A. Riofrío, Ivan H. Deutsch, and Poul S. Jessen. "Accurate and Robust Unitary Transformations of a High-Dimensional Quantum System".
In: *Phys. Rev. Lett.* 114 (24 June 2015), p. 240401.
DOI: 10.1103/PhysRevLett.114.240401.

[AYY26] Edwin Agnew, Lia Yeh, and Richie Yeung.
Algebraic Structure of Quantum Controlled States and Operators. 2026.
arXiv: 2603.13454 [quant-ph].

[Bac14] Miriam Backens.
"The ZX-calculus is complete for stabilizer quantum mechanics".

In: *New Journal of Physics* 16.9 (Sept. 2014), p. 093021.
DOI: 10.1088/1367-2630/16/9/093021.

[BBC+95] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, et al.
"Elementary gates for quantum computation".
In: *Physical Review A* 52.5 (Nov. 1995), pp. 3457–3467. ISSN: 1094-1622.
DOI: 10.1103/physreva.52.3457.

[BBPS96] Charles H. Bennett, Herbert J. Bernstein, Sandu Popescu, and Benjamin Schumacher.
"Concentrating partial entanglement by local operations".
In: *Physical Review A* 53.4 (Apr. 1996), pp. 2046–2052.
DOI: 10.1103/physreva.53.2046.

[BCC24] Robert I. Booth, Titouan Carette, and Cole Comfort.
Graphical Symplectic Algebra. 2024. arXiv: 2401.07914 [cs.LO].

[BCKW16] Alex Bocharov, Xingshan Cui, Vadym Kliuchnikov, and Zhenghan Wang.
"Efficient topological compilation for a weakly integral anyonic model".
In: *Physical Review A* 93.1 (Jan. 2016). ISSN: 2469-9934.
DOI: 10.1103/physreva.93.012313.

[BCRS16] Alex Bocharov, Shawn Cui, Martin Roetteler, and Krysta Svore.
"Improved Quantum Ternary Arithmetics".
In: *Quantum Information and Computation* 16 (July 2016), pp. 862–884.
DOI: 10.26421/QIC16.9-10-8.

[BDR+22] T. Brown, E. Doucet, D. Ristè, G. Ribeill, K. Cicak, J. Aumentado, R. Simmonds, et al. "Trade off-free entanglement stabilization in a superconducting qutrit-qubit system".
In: *Nature Communications* 13.1 (July 2022), p. 3994. ISSN: 2041-1723.
DOI: 10.1038/s41467-022-31638-0.

[BH12] Sergey Bravyi and Jeongwan Haah.
"Magic-state distillation with low overhead".
In: *Phys. Rev. A* 86 (5 Nov. 2012), p. 052329.
DOI: 10.1103/PhysRevA.86.052329.

[BH17] Ning Bao and Nicole Yunger Halpern.
"Quantum voting and violation of Arrow's impossibility theorem".
In: *Physical Review A* 95.6 (June 2017). DOI: 10.1103/physreva.95.062306.

[BK05] Sergey Bravyi and Alexei Kitaev.
"Universal quantum computation with ideal Clifford gates and noisy ancillas".
In: *Phys. Rev. A* 71 (2 Feb. 2005), p. 022316.
DOI: 10.1103/PhysRevA.71.022316.

[BK19] Miriam Backens and Aleks Kissinger. “ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity”. In: *Electronic Proceedings in Theoretical Computer Science* 287 (Jan. 2019), pp. 23–42. ISSN: 2075-2180. DOI: 10.4204/eptcs.287.2.

[BKL+19] T. Bækkegaard, L. B. Kristensen, N. J. S. Loft, C. K. Andersen, D. Petrosyan, and N. T. Zinner. “Realization of efficient quantum gates with a superconducting qubit-qutrit circuit”. In: *Scientific Reports* 9.1 (Sept. 2019), p. 13389. ISSN: 2045-2322. DOI: 10.1038/s41598-019-49657-1.

[BKM+23] Miriam Backens, Aleks Kissinger, Hector Miller-Bakewell, John van de Wetering, and Sal Wolffs. “Completeness of the ZH-calculus”. In: *Compositionality* 5 (July 2023), p. 5. ISSN: 2631-4444. DOI: 10.32408/compositionality-5-5.

[BLN+24] Hector Bombin, Daniel Litinski, Naomi Nickerson, Fernando Pastawski, and Sam Roberts. “Unifying flavors of fault tolerance with the ZX calculus”. In: *Quantum* 8 (June 2024), p. 1379. ISSN: 2521-327X. DOI: 10.22331/q-2024-06-18-1379.

[BMdF+21] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski, and John van de Wetering. “There and back again: A circuit extraction tale”. In: *Quantum* 5 (Mar. 2021), p. 421. ISSN: 2521-327X. DOI: 10.22331/q-2021-03-25-421.

[BOB05] Stephen Bullock, Dianne O’Leary, and Gavin Brennen. “Asymptotically Optimal Quantum Circuits for d-level Systems”. In: *Physical Review Letters* 94.23 (June 2005). DOI: 10.1103/physrevlett.94.230502.

[Boc16] Alex Bocharov. “A Note on Optimality of Quantum Circuits over Metaplectic Basis”. In: *Quantum Information and Computation* 18 (June 2016). DOI: 10.26421/QIC18.1-2-1.

[Bom16] Héctor Bombín. “Dimensional jump in quantum error correction”. In: *New Journal of Physics* 18.4 (Apr. 2016), p. 043038. DOI: 10.1088/1367-2630/18/4/043038.

[BPV21] Agustin Borgna, Simon Perdrix, and Benoit Valiron. “Hybrid Quantum-Classical Circuit Simplification with the ZX-Calculus”. In: *Programming Languages and Systems*. Springer International Publishing, 2021, pp. 121–139. DOI: 10.1007/978-3-030-89051-3_8.

[BRS+21] M. S. Blok, V. V. Ramasesh, T. Schuster, K. O'Brien, J. M. Kreikebaum, D. Dahlen, A. Morvan, et al.
"Quantum Information Scrambling on a Superconducting Qutrit Processor".
In: *Phys. Rev. X* 11 (2 Apr. 2021), p. 021010.
DOI: 10.1103/PhysRevX.11.021010.

[BRS17] Alex Bocharov, Martin Roetteler, and Krysta M. Svore. "Factoring with qutrits: Shor's algorithm on ternary and metaplectic quantum architectures".
In: *Phys. Rev. A* 96 (1 July 2017), p. 012306.
DOI: 10.1103/PhysRevA.96.012306.

[BYO13] Sinan Bugu, Can Yesilyurt, and Fatih Ozaydin. "Enhancing the W-state quantum-network-fusion process with a single Fredkin gate".
In: *Phys. Rev. A* 87 (3 Mar. 2013), p. 032331.
DOI: 10.1103/PhysRevA.87.032331.

[CAB12] Earl T. Campbell, Hussain Anwar, and Dan E. Browne. "Magic-State Distillation in All Prime Dimensions Using Quantum Reed-Muller Codes".
In: *Phys. Rev. X* 2 (4 Dec. 2012), p. 041021. DOI: 10.1103/PhysRevX.2.041021.

[Cam14] Earl T. Campbell.
"Enhanced Fault-Tolerant Quantum Computing in d-Level Systems".
In: *Phys. Rev. Lett.* 113 (23 Dec. 2014), p. 230501.
DOI: 10.1103/PhysRevLett.113.230501.

[Cam16] Earl T. Campbell. *The Smallest Interesting Colour Code.*
https://earltcampbell.com/2016/09/26/the-smallest-interesting-colour-code/.
Accessed: 2025-04-26. Aug. 2016.

[Cao23] Shuxiang Cao. "Implementation of variational quantum algorithms on superconducting qudits".
Doctoral Thesis. Oxford, UK: University of Oxford, 2023.
arXiv: 1709.08086 [math.CT].

[Car21] Titouan Carette. "Wielding the ZX-calculus, flexsymmetry, mixed states, and scalable notations".
Doctoral Thesis. Lorraine, France: Université de Lorraine, 2021.

[CBC+24a] Shuxiang Cao, Mustafa Bakr, Giulio Campanaro, Simone D Fasciati, James Wills, Deep Lall, Boris Shteynas, et al. "Emulating two qubits with a four-level transmon qudit for variational quantum algorithms".
In: *Quantum Science and Technology* 9.3 (Apr. 2024), p. 035003.
DOI: 10.1088/2058-9565/ad37d4.

[CBC+24b] Shuxiang Cao, Mustafa Bakr, Giulio Campanaro, Simone D Fasciati, James Wills, Deep Lall, Boris Shteynas, et al. "Emulating two qubits with a four-level transmon qudit for variational quantum algorithms".

In: *Quantum Science and Technology* 9.3 (Apr. 2024), p. 035003.
DOI: 10.1088/2058-9565/ad37d4.

[CD08] Bob Coecke and Ross Duncan. "Interacting Quantum Observables".
In: *Automata, Languages and Programming*.
Ed. by Luca Aceto, Ivan Damgård, Leslie Ann Goldberg,
Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz.
Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 298–310.
ISBN: 978-3-540-70583-3.

[CD11] Bob Coecke and Ross Duncan.
"Interacting quantum observables: categorical algebra and diagrammatics".
In: *New Journal of Physics* 13.4 (Apr. 2011), p. 043016. ISSN: 1367-2630.
DOI: 10.1088/1367-2630/13/4/043016.

[CDBO19] Daniele Cozzolino, Beatrice Da Lio, Davide Bacco, and Leif Katsuo Oxenløwe.
"High-Dimensional Quantum Communication: Benefits, Progress, and Future Challenges". In: *Advanced Quantum Technologies* 2.12 (2019), p. 1900038.
DOI: https://doi.org/10.1002/qute.201900038.
eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/qute.201900038.

[CDD+20] Alexander Cowtan, Silas Dilkes, Ross Duncan, Will Simmons, and
Seyon Sivarajah. "Phase Gadget Synthesis for Shallow Circuits".
In: *Proceedings 16th International Conference on Quantum Physics and Logic, Chapman University, Orange, CA, USA., 10-14 June 2019*.
Ed. by Bob Coecke and Matthew Leifer. Vol. 318.
Electronic Proceedings in Theoretical Computer Science.
Open Publishing Association, 2020, pp. 213–228.
DOI: 10.4204/EPTCS.318.13.

[CGK17] Shawn X. Cui, Daniel Gottesman, and Anirudh Krishna.
"Diagonal gates in the Clifford hierarchy".
In: *Phys. Rev. A* 95 (1 Jan. 2017), p. 012329.
DOI: 10.1103/PhysRevA.95.012329.

[CH17] Earl T. Campbell and Mark Howard. "Unified framework for magic state distillation and multiqubit gate synthesis with reduced resource cost".
In: *Physical Review A* 95.2 (Feb. 2017). ISSN: 2469-9934.
DOI: 10.1103/physreva.95.022316.

[CHM+22] Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix, and Benoit Valiron.
"LO_v-Calculus: A Graphical Language for Linear Optical Quantum Circuits".
In: *Leibniz International Proceedings in Informatics (LIPIcs)*. Vol. 241. Vienna,

Austria: Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Aug. 2022, 35:1–35:16. DOI: 10.4230/LIPIcs.MFCS.2022.35.

[CHP19] Titouan Carette, Dominic Horsman, and Simon Perdrix. "SZX-Calculus: Scalable Graphical Quantum Reasoning". In: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. DOI: 10.4230/LIPICS.MFCS.2019.55.

[CHW15] Shawn X. Cui, Seung-Moon Hong, and Zhenghan Wang. "Universal quantum computation with weakly integral anyons". In: *Quantum Information Processing* 14.8 (May 2015), pp. 2687–2727. ISSN: 1573-1332. DOI: 10.1007/s11128-015-1016-y.

[CHZ+22] Yulin Chi, Jieshan Huang, Zhanchuan Zhang, Jun Mao, Zinan Zhou, Xiaojiong Chen, Chonghao Zhai, et al. "A programmable qudit-based quantum processor". In: *Nature Communications* 13.1 (Mar. 2022), p. 1166. ISSN: 2041-1723. DOI: 10.1038/s41467-022-28767-x.

[CHZ+23] Ji Chu, Xiaoyu He, Yuxuan Zhou, Jiahao Yuan, Libo Zhang, Qihao Guo, Yongju Hai, et al. *Scalable algorithm simplification using quantum AND logic*. 2023.

[CJ20] Titouan Carette and Emmanuel Jeandel. "A Recipe for Quantum Graphical Languages". In: *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Ed. by Artur Czumaj, Anuj Dawar, and Emanuela Merelli. Vol. 168. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 118:1–118:17. ISBN: 978-3-95977-138-2. DOI: 10.4230/LIPIcs.ICALP.2020.118.

[CK10] Bob Coecke and Aleks Kissinger. "The Compositional Structure of Multipartite Quantum Entanglement". In: *Automata, Languages and Programming*. Ed. by Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 297–308. ISBN: 978-3-642-14162-1. DOI: 10.1007/978-3-642-14162-1_25.

[CK22] Cole Comfort and Aleks Kissinger. "A Graphical Calculus for Lagrangian Relations". In: *Electronic Proceedings in Theoretical Computer Science* 372 (Nov. 2022), pp. 338–351. ISSN: 2075-2180. DOI: 10.4204/eptcs.372.24.

[CKZ+23] Nicholas Chancellor, Aleks Kissinger, Stefan Zohren, Joschka Roffe, and Dominic Horsman.
"Graphical structures for design and verification of quantum error correction".
In: *Quantum Science and Technology* 8.4 (Sept. 2023), p. 045028.
ISSN: 2058-9565. DOI: 10.1088/2058-9565/acf157.

[CLB+24] Shuxiang Cao, Deep Lall, Mustafa Bakr, Giulio Campanaro, Simone D. Fasciati, James Wills, Vivek Chidambaram, et al.
"Efficient Characterization of Qudit Logical Gates with Gate Set Tomography Using an Error-Free Virtual Z Gate Model".
In: *Phys. Rev. Lett.* 133 (12 Sept. 2024), p. 120802.
DOI: 10.1103/PhysRevLett.133.120802.

[CMPV23] Titouan Carette, Etienne Moutot, Thomas Perez, and Renaud Vilmart.
"Compositionality of planar perfect matchings". In: *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*. Vol. 261. Paderborn, Germany, July 2023, 120:1–120:17.
DOI: 10.4230/LIPIcs.ICALP.2023.120.

[Com19] Wikimedia Commons. *File:Deutsch-Jozsa-algorithm-quantum-circuit.png*. 2019. URL: https://commons.wikimedia.org/wiki/File:Deutsch-Jozsa-algorithm-quantum-circuit.png.

[CS96] A. R. Calderbank and Peter W. Shor.
"Good quantum error-correcting codes exist".
In: *Physical Review A* 54.2 (Aug. 1996), pp. 1098–1105. ISSN: 1094-1622.
DOI: 10.1103/physreva.54.1098.

[CSD20] Alexander Cowtan, Will Simmons, and Ross Duncan.
A Generic Compilation Strategy for the Unitary Coupled Cluster Ansatz. 2020.
arXiv: 2007.10515 [quant-ph].

[CW15] Shawn X. Cui and Zhenghan Wang.
"Universal quantum computation with metaplectic anyons".
In: *Journal of Mathematical Physics* 56.3 (Mar. 2015), p. 032202.
ISSN: 1089-7658. DOI: 10.1063/1.4914941.

[CYB+22] Shuxiang Cao, Lia Yeh, Mustafa Bakr, Giulio Campanaro, Simone Fasciati, James Wills, Boris Shteynas, et al.
"Qutrit-ZX Calculus on superconducting transmon qutrits".
In: *APS March Meeting Abstracts*. Vol. 2022. 2022, F37–007.

[dBBW20a] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. "Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities".
In: *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*. Ed. by Steven T. Flammia. Vol. 158.

Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020, 11:1–11:23. ISBN: 978-3-95977-146-7. DOI: 10.4230/LIPIcs.TQC.2020.11.

[dBBW20b] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. "Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities". In: *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*. Ed. by Steven T. Flammia. Vol. 158. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 11:1–11:23. ISBN: 978-3-95977-146-7. DOI: 10.4230/LIPIcs.TQC.2020.11.

[dBBW20c] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. "Techniques to Reduce $\pi/4$-Parity-Phase Circuits, Motivated by the ZX Calculus". In: *Electronic Proceedings in Theoretical Computer Science* 318 (May 2020), pp. 131–149. DOI: 10.4204/eptcs.318.9.

[dBDHP20] Niel de Beaudrap, Ross Duncan, Dominic Horsman, and Simon Perdrix. "Pauli Fusion: a Computational Model to Realise Quantum Transformations from ZX Terms". In: *Electronic Proceedings in Theoretical Computer Science* 318 (May 2020), pp. 85–105. ISSN: 2075-2180. DOI: 10.4204/eptcs.318.6.

[dBea21] Niel de Beaudrap. "Well-tempered ZX and ZH Calculi". In: *Electronic Proceedings in Theoretical Computer Science* 340 (Sept. 2021), pp. 13–45. ISSN: 2075-2180. DOI: 10.4204/eptcs.340.2.

[dBH20] Niel de Beaudrap and Dominic Horsman. "The ZX calculus is a language for surface code lattice surgery". In: *Quantum* 4 (Jan. 2020), p. 218. ISSN: 2521-327X. DOI: 10.22331/q-2020-01-09-218.

[DD03] Jeroen Dehaene and Bart De Moor. "Clifford group, stabilizer states, and linear and quadratic operations over GF(2)". In: *Phys. Rev. A* 68 (4 Oct. 2003), p. 042318. DOI: 10.1103/PhysRevA.68.042318.

[dFC23] Giovanni de Felice and Bob Coecke. "Quantum Linear Optics via String Diagrams". In: *Electronic Proceedings in Theoretical Computer Science* 394 (Nov. 2023), pp. 83–100. ISSN: 2075-2180. DOI: 10.4204/eptcs.394.6.

[dFHN19] Giovanni de Felice, Amar Hadzihasanovic, and Kang Feng Ng. "A diagrammatic calculus of fermionic quantum circuits". In: *Logical Methods in Computer Science* Volume 15, Issue 3 (Sept. 2019). DOI: 10.23638/LMCS-15(3:26)2019.

[dFPC+26] Giovanni de Felice, Boldizsár Poór, Cole Comfort, Lia Yeh, Mateusz Kupper, William Cashman, and Bob Coecke. “A dataflow programming framework for linear optical distributed quantum computing”.
In: *Quantum* 10 (Jan. 2026), p. 1972. ISSN: 2521-327X.
DOI: 10.22331/q-2026-01-19-1972.

[dFPYC24] Giovanni de Felice, Boldizsár Poór, Lia Yeh, and William Cashman. *Fusion and flow: formal protocols to reliably build photonic graph states*. 2024.
arXiv: 2409.13541 [quant-ph].

[dFSP+23] Giovanni de Felice, Razin A. Shaikh, Boldizsár Poór, Lia Yeh, Quanlong Wang, and Bob Coecke. “Light-Matter Interaction in the ZXW Calculus”.
In: *Electronic Proceedings in Theoretical Computer Science* 384 (Aug. 2023), pp. 20–46. ISSN: 2075-2180. DOI: 10.4204/eptcs.384.2.

[DH15] Hillary Dawkins and Mark Howard.
“Qutrit Magic State Distillation Tight in Some Directions”.
In: *Phys. Rev. Lett.* 115 (3 July 2015), p. 030501.
DOI: 10.1103/PhysRevLett.115.030501.

[Dik16] Firat Diker. *Deterministic construction of arbitrary W states with quadratically increasing number of two-qubit gates*. 2016.
DOI: 10.48550/ARXIV.1606.09290.

[DJ92] David Deutsch and Richard Jozsa.
“Rapid solution of problems by quantum computation”.
In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (1992), pp. 553–558.

[DKPvdW20] Ross Duncan, Aleks Kissinger, Simon Perdrix, and John van de Wetering.
“Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus”.
In: *Quantum* 4 (June 2020), p. 279. ISSN: 2521-327X.
DOI: 10.22331/q-2020-06-04-279.

[DL14] Ross Duncan and Maxime Lucas.
“Verifying the Steane code with Quantomatic”. In: *Electronic Proceedings in Theoretical Computer Science* 171 (Dec. 2014), pp. 33–49. ISSN: 2075-2180.
DOI: 10.4204/eptcs.171.4.

[DP06] Ellie D’Hondt and Prakash Panangaden.
“The Computational Power of the W And GHZ States”.
In: *Quantum Info. Comput.* 6.2 (Mar. 2006), pp. 173–183. ISSN: 1533-7146.

[DP10] Ross Duncan and Simon Perdrix. “Rewriting measurement-based quantum computations with generalised flow”. In: *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming: Part II.*

ICALP'10. Bordeaux, France: Springer-Verlag, 2010, pp. 285–296.
ISBN: 3642141617.

[dSil21] Nadish de Silva. "Efficient quantum gate teleportation in higher dimensions". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 477.2251 (July 2021). ISSN: 1471-2946. DOI: 10.1098/rspa.2020.0865.

[dSL25] Nadish de Silva and Oscar Lautsch. *The Clifford hierarchy for one qubit or qudit*. 2025. arXiv: 2501.07939 [quant-ph].

[DVC00] W. Dür, G. Vidal, and J. I. Cirac. "Three qubits can be entangled in two inequivalent ways". In: *Physical Review A* 62.6 (Nov. 2000). ISSN: 1094-1622. DOI: 10.1103/physreva.62.062314.

[DW13] Yao-Min Di and Hai-Rui Wei. "Synthesis of multivalued quantum logic circuits by elementary gates". In: *Phys. Rev. A* 87 (1 Jan. 2013), p. 012325. DOI: 10.1103/PhysRevA.87.012325.

[DYP+23] Selma Dündar-Coecke, Lia Yeh, Caterina Puca, Sieglinde M.-L. Pfaendler, Muhammad Hamza Waseem, Thomas Cervoni, Aleks Kissinger, et al. "Quantum Picturalism: Learning Quantum Theory in High School". In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, Sept. 2023, pp. 21–32. DOI: 10.1109/qce57702.2023.20321.

[EK09] Bryan Eastin and Emanuel Knill. "Restrictions on Transversal Encoded Quantum Gate Sets". In: *Physical Review Letters* 102.11 (Mar. 2009). ISSN: 1079-7114. DOI: 10.1103/physrevlett.102.110502.

[EvdWCG22] Richard D.P. East, John van de Wetering, Nicholas Chancellor, and Adolfo G. Grushin. "AKLT-States as ZX-Diagrams: Diagrammatic Reasoning for Quantum States". In: *PRX Quantum* 3 (1 Jan. 2022), p. 010302. DOI: 10.1103/PRXQuantum.3.010302.

[FL24] Lingxuan Feng and Shunlong Luo. "Groups of diagonal gates in the Clifford hierarchy". In: *Theoretical and Mathematical Physics* 221.3 (Dec. 2024), pp. 2007–2021. ISSN: 1573-9333. DOI: 10.1134/S0040577924120018.

[GBD+19] Pranav Gokhale, Jonathan M. Baker, Casey Duckering, Natalie C. Brown, Kenneth R. Brown, and Frederic T. Chong. "Asymptotic improvements to quantum circuits via qutrits". In: *Proceedings of*

the 46th International Symposium on Computer Architecture (June 2019). DOI: 10.1145/3307650.3322253.

[GD18] Liam Garvie and Ross Duncan. "Verifying the Smallest Interesting Colour Code with Quantomatic". In: *Electronic Proceedings in Theoretical Computer Science* 266 (Feb. 2018), pp. 147–163. ISSN: 2075-2180. DOI: 10.4204/eptcs.266.10.

[GD20] Arianne Meijer-van de Griend and Ross Duncan. "Architecture-aware synthesis of phase polynomials for NISQ devices". English. In: *Electronic Proceedings in Theoretical Computer Science* (Apr. 2020). 17th International Conference on Quantum Physics and Logic , QPL 2020 ; Conference date: 02-06-2020 Through 06-06-2020. ISSN: 2075-2180.

[Gel62] Murray Gell-Mann. "Symmetries of Baryons and Mesons". In: *Phys. Rev.* 125 (3 Feb. 1962), pp. 1067–1084. DOI: 10.1103/PhysRev.125.1067.

[GFB+17] C. Godfrin, A. Ferhat, R. Ballou, S. Klyatskaya, M. Ruben, W. Wernsdorfer, and F. Balestro. "Operating Quantum States in Single Magnetic Molecules: Implementation of Grover's Quantum Algorithm". In: *Phys. Rev. Lett.* 119 (18 Nov. 2017), p. 187702. DOI: 10.1103/PhysRevLett.119.187702.

[GHP+14] Markus Gräfe, René Heilmann, Armando Perez-Leija, Robert Keil, Felix Dreisow, Matthias Heinrich, Hector Moya-Cessa, et al. "On-chip generation of high-order single-photon W-states". In: *Nature Photonics* 8.10 (Oct. 2014), pp. 791–795. ISSN: 1749-4893. DOI: 10.1038/nphoton.2014.204.

[Gid15] Craig Gidney. *Constructing Large Controlled Nots*. June 2015.

[Gid21] Craig Gidney. "Stim: a fast stabilizer circuit simulator". In: *Quantum* 5 (July 2021), p. 497. ISSN: 2521-327X. DOI: 10.22331/q-2021-07-06-497.

[GKMR14] David Gosset, Vadym Kliuchnikov, Michele Mosca, and Vincent Russo. "An Algorithm for the T-Count". In: *Quantum Info. Comput.* 14.15–16 (Nov. 2014), pp. 1261–1276. ISSN: 1533-7146.

[GLL+25] Erik J. Gustafson, Henry Lamm, Diyi Liu, Edison M. Murairi, and Shuchen Zhu. *Synthesis of Single Qutrit Circuits from Clifford+R*. 2025. arXiv: 2503.20203 [quant-ph].

[GMM+22] Noah Goss, Alexis Morvan, Brian Marinelli, Bradley K. Mitchell, Long B. Nguyen, Ravi K. Naik, Larry Chen, et al. "High-fidelity qutrit entangling gates for superconducting circuits".

In: *Nature Communications* 13.1 (Dec. 2022), p. 7481. ISSN: 2041-1723. DOI: 10.1038/s41467-022-34851-z.

[GMV25] Timothée Goubault de Brugière, Simon Martiel, and Christophe Vuillot. "A graph-state based synthesis framework for Clifford isometries". In: *Quantum* 9 (Jan. 2025), p. 1589. ISSN: 2521-327X. DOI: 10.22331/q-2025-01-14-1589.

[Got97] Daniel Gottesman. "Stabilizer codes and quantum error correction". PhD thesis. California Institute of Technology, 1997. DOI: 10.48550/arXiv.quant-ph/9705052.

[Got98a] Daniel Gottesman. "The Heisenberg Representation of Quantum Computers". In: *Proc. XXII International Colloquium on Group Theoretical Methods in Physics*. Ed. by Stuart P. Corney, Robert Delbourgo, and Peter D. Jarvis. GROUP22. Hobart, Australia: International Press of Boston, 1998, pp. 32–43. ISBN: 9781571460547.

[Got98b] Daniel Gottesman. "Theory of fault-tolerant quantum computation". In: *Phys. Rev. A* 57 (1 Jan. 1998), pp. 127–137. DOI: 10.1103/PhysRevA.57.127.

[Got99] Daniel Gottesman. "Fault-Tolerant Quantum Computation with Higher-Dimensional Systems". In: *Chaos, Solitons & Fractals* 10.10 (Sept. 1999), pp. 1749–1758. ISSN: 0960-0779. DOI: 10.1016/s0960-0779(98)00218-5.

[Gre14] Seth Evenson Murray Greylyn. "Generators and relations for the group $\mathrm{U}_4(\mathbb{Z}[1/\sqrt{2}, i])$". MA thesis. Dalhousie University, 2014.

[GRT19] Andrew N. Glaudell, Neil J. Ross, and Jacob M. Taylor. "Canonical forms for single-qutrit Clifford+T operators". In: *Annals of Physics* 406 (July 2019), pp. 54–70. ISSN: 00034916. DOI: 10.1016/j.aop.2019.04.001.

[GRvdWY22a] Andrew N. Glaudell, Neil J. Ross, John van de Wetering, and Lia Yeh. "Qutrit Metaplectic Gates Are a Subset of Clifford+T". en. In: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. DOI: 10.4230/LIPICS.TQC.2022.12.

[GRvdWY22b] Andrew N. Glaudell, Neil J. Ross, John van de Wetering, and Lia Yeh. "Qutrit Metaplectic Gates Are a Subset of Clifford+T". In: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*. Ed. by François Le Gall and Tomoyuki Morimae. Vol. 232. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany:

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 12:1–12:15. ISBN: 978-3-95977-237-2. DOI: 10.4230/LIPIcs.TQC.2022.12.

[GRvdWY22c] Andrew N. Glaudell, Neil J. Ross, John van de Wetering, and Lia Yeh. "Qutrit Metaplectic Gates Are a Subset of Clifford+T". In: *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*. Ed. by François Le Gall and Tomoyuki Morimae. Vol. 232. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 12:1–12:15. ISBN: 978-3-95977-237-2. DOI: 10.4230/LIPIcs.TQC.2022.12.

[GRvdWY24] Andrew N. Glaudell, Neil J. Ross, John van de Wetering, and Lia Yeh. "Exact Synthesis of Multiqutrit Clifford-Cyclotomic Circuits". In: *Electronic Proceedings in Theoretical Computer Science* 406 (Aug. 2024), pp. 44–62. ISSN: 2075-2180. DOI: 10.4204/eptcs.406.2.

[GS13] Brett Giles and Peter Selinger. "Exact synthesis of multiqubit Clifford+T circuits". In: *Physical Review A* 87.3 (Mar. 2013). ISSN: 1094-1622. DOI: 10.1103/physreva.87.032332.

[GSÇ+15] Z. Gedik, I. A. Silva, B. Çakmak, G. Karpat, E. L. G. Vidoto, D. O. Soares-Pinto, E. R. deAzevedo, and F. F. Fanchini. "Computational speed-up with a single qudit". In: *Scientific Reports* 5.1 (Oct. 2015), p. 14671. ISSN: 2045-2322. DOI: 10.1038/srep14671.

[GSJ24] Craig Gidney, Noah Shutty, and Cody Jones. *Magic state cultivation: growing T states as cheap as CNOT gates*. 2024. arXiv: 2409.17595 [quant-ph].

[Gun25] Lane G. Gunderman. *Beyond Integral-Domain Stabilizer Codes*. 2025. arXiv: 2501.04888 [quant-ph].

[GW17] Xiaoyan Gong and Quanlong Wang. *Equivalence of Local Complementation and Euler Decomposition in the Qutrit ZX-calculus*. 2017. arXiv: 1704.05955 [quant-ph].

[Had15] Amar Hadzihasanovic. "A Diagrammatic Axiomatisation for Qubit Entanglement". In: *Proceedings of the 2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. LICS '15. USA: IEEE Computer Society, 2015, pp. 573–584. ISBN: 9781479988754. DOI: 10.1109/LICS.2015.59.

[Had17] Amar Hadzihasanovic. "The algebra of entanglement and the geometry of composition".

Doctoral Thesis. Oxford, UK: University of Oxford, 2017.
arXiv: 1709.08086 [math.CT].

[Had93] Jacques Hadamard. "Resolution d'une question relative aux determinants". In: *Bull. des sciences math.* 2 (1893), pp. 240–246.

[HC17] Mark Howard and Earl Campbell. "Application of a Resource Theory for Magic States to Fault-Tolerant Quantum Computing". In: *Phys. Rev. Lett.* 118 (9 Mar. 2017), p. 090501. DOI: 10.1103/PhysRevLett.118.090501.

[HC19] Luke E. Heyfron and Earl Campbell. *A quantum compiler for qudits of prime dimension greater than 3*. 2019. arXiv: 1902.05634 [quant-ph].

[HEMN20] Michael Hanks, Marta P. Estarellas, William J. Munro, and Kae Nemoto. "Effective Compression of Quantum Braided Circuits Aided by ZX-Calculus". In: *Physical Review X* 10.4 (Nov. 2020). ISSN: 2160-3308. DOI: 10.1103/physrevx.10.041030.

[HGL+18] Xiao-Min Hu, Yu Guo, Bi-Heng Liu, Yun-Feng Huang, Chuan-Feng Li, and Guang-Can Guo. "Beating the channel capacity limit for superdense coding with entangled ququarts". In: *Science Advances* 4.7 (2018), eaat9304. DOI: 10.1126/sciadv.aat9304. eprint: https://www.science.org/doi/pdf/10.1126/sciadv.aat9304.

[HGNS15] René Heilmann, Markus Gräfe, Stefan Nolte, and Alexander Szameit. "A novel integrated quantum circuit for high-order W-state generation and its highly precise characterization". In: *Science Bulletin* 60.1 (2015), pp. 96–100. ISSN: 2095-9273. DOI: 10.1007/s11434-014-0688-5.

[HHDR21] Alexander D. Hill, Mark J. Hodson, Nicolas Didier, and Matthew J. Reagor. *Realization of arbitrary doubly-controlled quantum phase gates*. 2021. DOI: 10.48550/arXiv.2108.01652. arXiv: 2108.01652 [quant-ph].

[HKL+26] Jiaxin Huang, Aleks Kissinger, Sarah Meng Li, John van de Wetering, and Lia Yeh. *A Three-Way Normal Form for Stabiliser Codes across ZX Diagrams, Circuits, and Tableaus*. In preparation. 2026.

[HLY+23] Jiaxin Huang, Sarah Meng Li, Lia Yeh, Aleks Kissinger, Michele Mosca, and Michael Vasmer. "Graphical CSS Code Transformation Using ZX Calculus". In: *Electronic Proceedings in Theoretical Computer Science* 384 (Aug. 2023), pp. 1–19. ISSN: 2075-2180. DOI: 10.4204/eptcs.384.1.

[HNW18] Amar Hadzihasanovic, Kang Feng Ng, and Quanlong Wang. "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.

Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 502–511. ISBN: 9781450355834. DOI: 10.1145/3209108.3209128.

[htta] Craig Gidney (https://cstheory.stackexchange.com/users/4967/craig-gidney). *Computing Robustness of Magic of n-bit W states.* Theoretical Computer Science Stack Exchange. URL: https://cstheory.stackexchange.com/q/40272 (version: 2018-02-25). eprint: https://cstheory.stackexchange.com/q/40272.

[httb] Craig Gidney (https://quantumcomputing.stackexchange.com/users/119/craig-gidney). *General construction of Wn state.* Quantum Computing Stack Exchange. URL: https://quantumcomputing.stackexchange.com/a/4358 (version: 2018-10-05). eprint: https://cstheory.stackexchange.com/q/40272.

[HV12] Mark Howard and Jiri Vala. "Qudit versions of the qubit $\pi/8$ gate". In: *Phys. Rev. A* 86 (2 Aug. 2012), p. 022316. DOI: 10.1103/PhysRevA.86.022316.

[HWG+23] Pavel Hrmo, Benjamin Wilhelm, Lukas Gerster, Martin W. van Mourik, Marcus Huber, Rainer Blatt, Philipp Schindler, et al. "Native qudit entanglement in a trapped ion quantum processor". In: *Nature Communications* 14.1 (Apr. 2023), p. 2242. ISSN: 2041-1723. DOI: 10.1038/s41467-023-37375-2.

[HWM17] Majid Haghparast, Robert Wille, and Asma Taheri Monfared. "Towards quantum reversible ternary coded decimal adder". In: *Quantum Information Processing* 16.11 (Nov. 2017), p. 284. ISSN: 1570-0755, 1573-1332. DOI: 10.1007/s11128-017-1735-3.

[IC22] Robert I. Booth and Titouan Carette. "Complete ZX-calculi for the stabiliser fragment in odd prime dimensions". In: *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022).* Ed. by Stefan Szeider, Robert Ganian, and Alexandra Silva. Vol. 241. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, pp. 343–357. ISBN: 978-3-95977-256-3. DOI: 10.4230/LIPIcs.MFCS.2022.24.

[ISM09] Radu Ionicioiu, Timothy Spiller, and William Munro. "Generalized Toffoli gates using qudit catalysis". In: *Physical Review A* 80 (July 2009), p. 012312. DOI: 10.1103/PhysRevA.80.012312.

[Jon13] Cody Jones. “Low-overhead constructions for the fault-tolerant Toffoli gate”. In: *Phys. Rev. A* 87 (2 Feb. 2013), p. 022328. DOI: 10.1103/PhysRevA.87.022328.

[JPV18] Emmanuel Jeandel, Simon Perdrix, and Renaud Vilmart. “Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics”. In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS ’18. Oxford, United Kingdom: Association for Computing Machinery, 2018, pp. 569–578. ISBN: 9781450355834. DOI: 10.1145/3209108.3209139.

[JPV24] Emmanuel Jeandel, Simon Perdrix, and Margarita Veshchezerova. “Addition and Differentiation of ZX-diagrams”. In: *Logical Methods in Computer Science* Volume 20, Issue 2 (May 2024). DOI: 10.46298/lmcs-20(2:10)2024.

[JSY17] Y. Q. Ji, X. Q. Shao, and X. X. Yi. “Fusing atomic W states via quantum Zeno dynamics”. In: *Scientific Reports* 7.1 (May 2017), p. 1378. ISSN: 2045-2322. DOI: 10.1038/s41598-017-01499-5.

[KB15] Aleksander Kubica and Michael E. Beverland. “Universal transversal gates with color codes: A simplified approach”. In: *Phys. Rev. A* 91 (3 Mar. 2015), p. 032330. DOI: 10.1103/PhysRevA.91.032330.

[KC18] Taewan Kim and Byung-Soo Choi. “Efficient decomposition methods for controlled-Rn using a single ancillary qubit”. In: *Scientific Reports* 8.1 (Apr. 2018), p. 5445. ISSN: 2045-2322. DOI: 10.1038/s41598-018-23764-x.

[KGH+21] Valentin Kasper, Daniel González-Cuadra, Apoorva Hegde, Andy Xia, Alexandre Dauphin, Felix Huber, Eberhard Tiemann, et al. “Universal quantum computation and quantum error correction with ultracold atomic mixtures”. In: *Quantum Science and Technology* 7.1 (Nov. 2021), p. 015008. DOI: 10.1088/2058-9565/ac2d39.

[Kis22] Aleks Kissinger. *Phase-free ZX diagrams are CSS codes (...or how to graphically grok the surface code)*. 2022. arXiv: 2204.14038 [quant-ph].

[Kit96] Alexei Kitaev. “Quantum measurements and the Abelian stabilizer problem”. In: *Electronic Colloq. on Computational Complexity*. 1996.

[Kli13] Vadym Kliuchnikov. *Synthesis of unitaries with Clifford+T circuits*. 2013. arXiv: 1306.3200 [quant-ph].

[KMM13] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. “Fast and Efficient Exact Synthesis of Single-Qubit Unitaries Generated by Clifford and T Gates”. In: *Quantum Info. Comput.* 13.7–8 (July 2013), pp. 607–630. ISSN: 1533-7146.

[KMV25] Amolak Ratan Kalra, Michele Mosca, and Dinesh Valluri. “Synthesis and Arithmetic of Single Qutrit Circuits”. In: *Quantum* 9 (Feb. 2025), p. 1647. ISSN: 2521-327X. DOI: 10.22331/q-2025-02-26-1647.

[KNX+20] E. O. Kiktenko, A. S. Nikolaeva, Peng Xu, G. V. Shlyapnikov, and A. K. Fedorov. “Scalable quantum computing with qudits on a graph”. In: *Physical Review A* 101.2 (Feb. 2020). ISSN: 2469-9934. DOI: 10.1103/physreva.101.022304.

[KP06] Faisal Shah Khan and Marek Perkowski. “Synthesis of multi-qudit hybrid and d-valued quantum logic circuits by decomposition”. In: *Theoretical Computer Science* 367.3 (2006), pp. 336–346. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2006.09.006.

[KP07] Mozammel H. A. Khan and Marek A. Perkowski. “Quantum ternary parallel adder/subtractor with partially-look-ahead carry”. In: *Journal of Systems Architecture* 53.7 (July 2007), pp. 453–464. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2007.01.007.

[KS08] Jeong San Kim and Barry C Sanders. “Generalized W-class state and its monogamy relation”. In: *Journal of Physics A: Mathematical and Theoretical* 41.49 (Oct. 2008), p. 495301. DOI: 10.1088/1751-8113/41/49/495301.

[KS24] Vadym Kliuchnikov and Sebastian Schönnenbeck. *Multi-qubit circuit synthesis and Hermitian lattices*. 2024. arXiv: 2405.19302 [quant-ph].

[KSB+20] Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. “Superconducting Qubits: Current State of Play”. In: *Annual Review of Condensed Matter Physics* 11.Volume 11, 2020 (2020), pp. 369–395. ISSN: 1947-5462. DOI: https://doi.org/10.1146/annurev-conmatphys-031119-050605.

[KSV+24] Amolak Ratan Kalra, Manimugdha Saikia, Dinesh Valluri, Sam Winnick, and Jon Yard. *Multi-qutrit exact synthesis*. 2024. arXiv: 2405.08147 [quant-ph].

[KT19] Anirudh Krishna and Jean-Pierre Tillich. “Towards Low Overhead Magic State Distillation”. In: *Physical Review Letters* 123.7 (Aug. 2019). DOI: 10.1103/physrevlett.123.070507.

[Kub18] Aleksander Kubica. “The ABCs of the Color Code: A Study of Topological Quantum Codes as Toy Models for Fault-Tolerant Quantum Computation and Quantum Phases Of Matter”.
PhD thesis. California Institute of Technology, 2018.
DOI: 10.7907/059V-MG69.

[Kup19] Mateusz Kupper.
“Analysis of quantum hypergraph states in the ZH-calculus”.
MA thesis. University of Edinburgh, 2019.

[KvdW20a] Aleks Kissinger and John van de Wetering.
“PyZX: Large Scale Automated Diagrammatic Reasoning”. In: *Electronic Proceedings in Theoretical Computer Science* 318 (May 2020), pp. 229–241.
ISSN: 2075-2180. DOI: 10.4204/eptcs.318.14.

[KvdW20b] Aleks Kissinger and John van de Wetering.
“Reducing the number of non-Clifford gates in quantum circuits”.
In: *Physical Review A* 102.2 (Aug. 2020). ISSN: 2469-9934.
DOI: 10.1103/physreva.102.022406.

[KvdW24a] Aleks Kissinger and John van de Wetering. *Picturing Quantum Software: An Introduction to the ZX-Calculus and Quantum Compilation*. Preprint, 2024.

[KvdW24b] Aleks Kissinger and John van de Wetering. “Scalable Spider Nests (...Or How to Graphically Grok Transversal Non-Clifford Gates)”. In: *Electronic Proceedings in Theoretical Computer Science* 406 (Aug. 2024), pp. 79–95.
ISSN: 2075-2180. DOI: 10.4204/eptcs.406.4.

[KvdWK19] Stach Kuijpers, John van de Wetering, and Aleks Kissinger.
Graphical Fourier Theory and the Cost of Quantum Addition. 2019.
arXiv: 1904.07551 [quant-ph].

[KvdWV22] Aleks Kissinger, John van de Wetering, and Renaud Vilmart.
“Classical Simulation of Quantum Circuits with Partial and Graphical Stabiliser Decompositions”. In:
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
DOI: 10.4230/LIPICS.TQC.2022.5.

[KZ15] Aleks Kissinger and Vladimir Zamdzhiev.
“Quantomatic: A Proof Assistant for Diagrammatic Reasoning”. In:
Automated Deduction - CADE-25. Springer International Publishing, 2015, pp. 326–336. ISBN: 9783319214016. DOI: 10.1007/978-3-319-21401-6_22.

[LBA+09a] Benjamin P. Lanyon, Marco Barbieri, Marcelo P. Almeida, Thomas Jennewein, Timothy C. Ralph, Kevin J. Resch, Geoff J. Pryde, et al.
“Simplifying quantum logic using higher-dimensional Hilbert spaces”.

In: *Nature Physics* 5.2 (Feb. 2009), pp. 134–140. ISSN: 1745-2481.
DOI: 10.1038/nphys1150.

[LBA+09b] Benjamin P. Lanyon, Marco Barbieri, Marcelo P. Almeida, Thomas Jennewein, Timothy C. Ralph, Kevin J. Resch, Geoff J. Pryde, et al. "Simplifying quantum logic using higher-dimensional Hilbert spaces". In: *Nature Physics* 5.2 (Feb. 2009), pp. 134–140. ISSN: 1745-2481. DOI: 10.1038/nphys1150.

[LCR22] Adrian Lehmann, Ben Caldwell, and Robert Rand. *VyZX : A Vision for Verifying the ZX Calculus*. 2022. arXiv: 2205.05781 [quant-ph].

[Li23] Sarah Meng Li. "Graphical CSS Code Transformation Using ZX Calculus". Master's thesis. Waterloo, Ontario, Canada, Dec. 2023.

[LJW+20] C. Li, N. Jiang, Y.-K. Wu, W. Chang, Y.-F. Pu, S. Zhang, and L.-M. Duan. "Quantum Communication between Multiplexed Atomic Quantum Memories". In: *Phys. Rev. Lett.* 124 (24 June 2020), p. 240504. DOI: 10.1103/PhysRevLett.124.240504.

[LKY+16] Ke Li, Fan-Zhen Kong, Ming Yang, Fatih Ozaydin, Qing Yang, and Zhuo-Liang Cao. "Generating multi-photon W-like states for perfect quantum teleportation and superdense coding". In: *Quantum Information Processing* 15.8 (Aug. 2016), pp. 3137–3150. ISSN: 1573-1332. DOI: 10.1007/s11128-016-1332-x.

[LMPZ96] Raymond Laflamme, Cesar Miquel, Juan Pablo Paz, and Wojciech Hubert Zurek. *Perfect Quantum Error Correction Code*. 1996. arXiv: quant-ph/9602019 [quant-ph].

[LMW18] Victoria Lipinska, Gláucia Murta, and Stephanie Wehner. "Anonymous transmission in a noisy quantum network using the W state". In: *Phys. Rev. A* 98 (5 Nov. 2018), p. 052320. DOI: 10.1103/PhysRevA.98.052320.

[LvdWK21] Louis Lemonnier, John van de Wetering, and Aleks Kissinger. "Hypergraph Simplification: Linking the Path-sum Approach to the ZH-calculus". In: *Proceedings 17th International Conference on Quantum Physics and Logic, Paris, France, June 2 - 6, 2020*. Ed. by Benoit Valiron, Shane Mansfield, Pablo Arrighi, and Prakash Panangaden. Vol. 340. Electronic Proceedings in Theoretical Computer Science. Open Publishing Association, 2021, pp. 188–212. DOI: 10.4204/EPTCS.340.10.

[LY26] Christine Li and Lia Yeh. *Transversal AND in Quantum Codes*. 2026. arXiv: 2603.04548 [quant-ph].

[Mas16] Dmitri Maslov. “Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization”. In: *Phys. Rev. A* 93 (2 Feb. 2016), p. 022311. DOI: 10.1103/PhysRevA.93.022311.

[MB23] Tommy McElvanney and Miriam Backens. *Complete Flow-Preserving Rewrite Rules for MBQC Patterns with Pauli Measurements*. Nov. 2023. DOI: 10.4204/eptcs.394.5.

[MD23] Arianne Meijer-van de Griend and Ross Duncan. *Architecture-Aware Synthesis of Phase Polynomials for NISQ Devices*. Nov. 2023. DOI: 10.4204/eptcs.394.8.

[MNM+21] Bradley K. Mitchell, Ravi K. Naik, Alexis Morvan, Akel Hashim, John Mark Kreikebaum, Brian Marinelli, Wim Lavrijsen, et al. “Hardware-Efficient Microwave-Activated Tunable Coupling between Superconducting Qubits”. In: *Phys. Rev. Lett.* 127 (20 Nov. 2021), p. 200502. DOI: 10.1103/PhysRevLett.127.200502.

[MOP+09] François Mallet, Florian R. Ong, Agustin Palacios-Laloy, François Nguyen, Patrice Bertet, Denis Vion, and Daniel Esteve. “Single-shot qubit readout in circuit quantum electrodynamics”. In: *Nature Physics* 5.11 (Nov. 2009), pp. 791–795. ISSN: 1745-2481. DOI: 10.1038/nphys1400.

[Mor14] Claudio Moraga. “On Some Basic Aspects of Ternary Reversible and Quantum Computing”. In: *Proceedings of the 2014 IEEE 44th International Symposium on Multiple-Valued Logic*. ISMVL ’14. USA: IEEE Computer Society, 2014, pp. 178–183. ISBN: 9781479935352. DOI: 10.1109/ISMVL.2014.39.

[Mor16] Claudio Moraga. “Quantum p-Valued Toffoli and Deutsch Gates with Conjunctive or Disjunctive Mixed Polarity Control”. In: *2016 IEEE 46th International Symposium on Multiple-Valued Logic (ISMVL)*. 2016, pp. 241–246. DOI: 10.1109/ISMVL.2016.22.

[MRHW22] Kevin Mato, Martin Ringbauer, Stefan Hillmich, and Robert Wille. “Adaptive Compilation of Multi-Level Quantum Operations”. In: *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. 2022. ISBN: 978-1-6654-9113-6. DOI: 10.1109/QCE53715.2022.00070.

[MSY25] Haytham McDowall-Rose, Razin A. Shaikh, and Lia Yeh. *From Fermions to Qubits: A ZX-Calculus Perspective*. 2025. arXiv: 2505.06212 [quant-ph].

[NC10] Michael A Nielsen and Isaac L Chuang.
Quantum computation and quantum information.
Cambridge University Press, 2010.

[Nep23] Rafael I. Nepomechie. *Qudit Dicke state preparation.* 2023.
DOI: 10.48550/arXiv.2301.04989. arXiv: 2301.04989 [quant-ph].

[NKF22] A. S. Nikolaeva, E. O. Kiktenko, and A. K. Fedorov.
"Decomposing the generalized Toffoli gate with qutrits".
In: *Physical Review A* 105.3 (Mar. 2022). DOI: 10.1103/physreva.105.032621.

[NW19] Kang Feng Ng and Quanlong Wang. *Completeness of the ZX-calculus.*
Presentation at 16th International Conference on Quantum Physics and Logic (QPL), 2019.

[NWD20] Philipp Niemann, Robert Wille, and Rolf Drechsler.
"Advanced exact synthesis of Clifford+ T circuits".
In: *Quantum Information Processing* 19 (2020), pp. 1–23.

[OMT+11] Sahin Ozdemir, E Matsunaga, Toshiyuki Tashima, Takashi Yamamoto, M Koashi, and N Imoto. "An optical fusion gate for W-states".
In: *New Journal of Physics* 13 (Oct. 2011).
DOI: 10.1088/1367-2630/13/10/103003.

[PBC+23] Boldizsár Poór, Robert I. Booth, Titouan Carette, John van de Wetering, and Lia Yeh. "The Qupit Stabiliser ZX-travaganza: Simplified Axioms, Normal Forms and Graph-Theoretic Simplification". In: *Electronic Proceedings in Theoretical Computer Science* 384 (Aug. 2023), pp. 220–264. ISSN: 2075-2180.
DOI: 10.4204/eptcs.384.13.

[PDF+21] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, et al.
"Demonstration of the trapped-ion quantum CCD computer architecture".
In: *Nature* 592.7853 (Apr. 2021), pp. 209–213.
DOI: 10.1038/s41586-021-03318-4.

[PH23] Christopher Popp and Beatrix C. Hiesmayr. "Comparing bound entanglement of bell diagonal pairs of qutrits and ququarts".
In: *Scientific Reports* 13.1 (Feb. 2023), p. 2037. ISSN: 2045-2322.
DOI: 10.1038/s41598-023-29211-w.

[PJKS18] Shiroman Prakash, Akalank Jain, Bhakti Kapur, and Shubangi Seth.
"Normal form for single-qutrit Clifford+T operators and synthesis of single-qutrit gates". In: *Physical Review A* 98.3 (Sept. 2018). ISSN: 2469-9934.
DOI: 10.1103/physreva.98.032304.

[Pra20] Shiroman Prakash. "Magic state distillation with the ternary Golay code".
In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering*

Sciences 476.2241 (2020), p. 20200187. DOI: 10.1098/rspa.2020.0187.
eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2020.0187.

[PRE+22] Luca Petit, Maximilian Russ, Gertjan HGJ Eenink, William IL Lawrie, James S Clarke, Lieven MK Vandersypen, and Menno Veldhorst. *Design and integration of single-qubit rotations and two-qubit gates in silicon above one kelvin*. 2022.

[PRTC20] Tefjol Pllaha, Narayanan Rengaswamy, Olav Tirkkonen, and Robert Calderbank. "Un-Weyl-ing the Clifford Hierarchy". In: *Quantum* 4 (Dec. 2020), p. 370. ISSN: 2521-327X. DOI: 10.22331/q-2020-12-11-370.

[PSW25] Boldizsár Poór, Razin A. Shaikh, and Quanlong Wang. *ZX-calculus is Complete for Finite-Dimensional Hilbert Spaces*. 2025. arXiv: 2405.10896 [quant-ph].

[PW24] Jennifer Paykin and Sam Winnick. *Qudit Quantum Programming with Projective Cliffords*. 2024. arXiv: 2407.16801 [quant-ph].

[PWS+23] Boldizsar Poor, Quanlong Wang, Razin A. Shaikh, Lia Yeh, Richie Yeung, and Bob Coecke. " Completeness for arbitrary finite dimensions of ZXW-calculus, a unifying calculus ". In: *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2023, pp. 1–14. DOI: 10.1109/LICS56636.2023.10175672.

[QZPS18] Dong-Xiao Quan, Li-Li Zhu, Chang-Xing Pei, and Barry C Sanders. "Fault-tolerant conversion between adjacent Reed–Muller quantum codes based on gauge fixing". In: *Journal of Physics A: Mathematical and Theoretical* 51.11 (Feb. 2018), p. 115305. ISSN: 1751-8121. DOI: 10.1088/1751-8121/aaad13.

[Ran14] André Ranchin. "Depicting qudit quantum mechanics and mutually unbiased qudit theories". In: *Electronic Proceedings in Theoretical Computer Science* 172 (Dec. 2014), pp. 68–91. DOI: 10.4204/eptcs.172.6.

[RBC+18] Martin Ringbauer, Thomas R. Bromley, Marco Cianciaruso, Ludovico Lami, W. Y. Sarah Lau, Gerardo Adesso, Andrew G. White, et al. "Certification and Quantification of Multilevel Quantum Coherence". In: *Phys. Rev. X* 8 (4 Oct. 2018), p. 041007. DOI: 10.1103/PhysRevX.8.041007.

[RCP19] Narayanan Rengaswamy, Robert Calderbank, and Henry D. Pfister. "Unifying the Clifford hierarchy via symmetric matrices over rings".

In: *Phys. Rev. A* 100 (2 Aug. 2019), p. 022304.
DOI: 10.1103/PhysRevA.100.022304.

[RMP+22a] Martin Ringbauer, Michael Meth, Lukas Postler, Roman Stricker, Rainer Blatt, Philipp Schindler, and Thomas Monz.
"A universal qudit quantum processor with trapped ions".
In: *Nature Physics* 18.9 (July 2022), pp. 1053–1057. ISSN: 1745-2481.
DOI: 10.1038/s41567-022-01658-0.

[RMP+22b] Martin Ringbauer, Michael Meth, Lukas Postler, Roman Stricker, Rainer Blatt, Philipp Schindler, and Thomas Monz.
A universal qudit quantum processor with trapped ions. 2022.

[Roy22] Patrick Roy. "Qudit ZH-calculus".
Master's Thesis. Oxford, UK: University of Oxford, 2022.

[RRG07] T. C. Ralph, K. J. Resch, and A. Gilchrist. "Efficient Toffoli gates using qudits".
In: *Physical Review A* 75.2 (Feb. 2007). ISSN: 1094-1622.
DOI: 10.1103/physreva.75.022313.

[RvdWY23] Patrick Roy, John van de Wetering, and Lia Yeh.
"The Qudit ZH-Calculus: Generalised Toffoli+Hadamard and Universality".
In: *Electronic Proceedings in Theoretical Computer Science* 384 (Aug. 2023), pp. 142–170. ISSN: 2075-2180. DOI: 10.4204/eptcs.384.9.

[Sch01] D. Schlingemann. *Stabilizer codes can be realized as graph codes*. 2001.
arXiv: quant-ph/0111080 [quant-ph].

[Sel13] Peter Selinger. "Quantum circuits of T-depth one".
In: *Physical Review A* 87.4 (Apr. 2013), p. 042302. ISSN: 1050-2947, 1094-1622.
DOI: 10.1103/PhysRevA.87.042302.

[Sel16] Peter Selinger. *Reversible k-valued logic circuits are finitely generated for odd k*. 2016. arXiv: 1604.01646 [cs.ET].

[Shi03] Yaoyun Shi. "Both Toffoli and Controlled-NOT Need Little Help to Do Universal Quantum Computing".
In: *Quantum Info. Comput.* 3.1 (Jan. 2003), pp. 84–92. ISSN: 1533-7146.
DOI: 10.5555/2011508.2011515.

[SK25] Matthew Sutcliffe and Aleks Kissinger. "Fast classical simulation of quantum circuits via parametric rewriting in the ZX-calculus". In: *To appear in proceedings of the 2025 International Conference on Quantum Physics and Logic*. Open Publishing Association, 2025. arXiv: 2403.06777 [quant-ph].

[SM09] Vivek V. Shende and Igor L. Markov. "On the CNOT-Cost of TOFFOLI Gates".
In: *Quantum Info. Comput.* 9.5 (May 2009), pp. 461–486. ISSN: 1533-7146.

[SMCG16] Sarah Sheldon, Easwar Magesan, Jerry M. Chow, and Jay M. Gambetta.
"Procedure for systematically tuning up cross-talk in the cross-resonance

gate". In: *Phys. Rev. A* 93 (6 June 2016), p. 060302.
DOI: 10.1103/PhysRevA.93.060302.

[SPG+15] YuBo Sheng, Jun Pan, Rui Guo, Lan Zhou, and Lei Wang.
"Efficient N-particle W state concentration with different parity check gates".
In: *Science China Physics, Mechanics & Astronomy* 58.6 (June 2015), pp. 1–11.
ISSN: 1869-1927. DOI: 10.1007/s11433-015-5672-9.

[SSRP22] Manoranjan Swain, M. Karthick Selvan, Amit Rai, and Prasanta K. Panigrahi.
Generation and entanglement study of generalized N-mode single photon perfect W-states. 2022. DOI: 10.48550/arXiv.2112.05145.
arXiv: 2112.05145 [quant-ph].

[Ste96] A. M. Steane. "Error Correcting Codes in Quantum Theory".
In: *Phys. Rev. Lett.* 77 (5 July 1996), pp. 793–797.
DOI: 10.1103/PhysRevLett.77.793.

[SWY23] Razin A. Shaikh, Quanlong Wang, and Richie Yeung.
"How to Sum and Exponentiate Hamiltonians in ZXW Calculus".
In: *Electronic Proceedings in Theoretical Computer Science* 394 (Nov. 2023),
pp. 236–261. ISSN: 2075-2180. DOI: 10.4204/eptcs.394.14.

[SYG24] Razin A. Shaikh, Lia Yeh, and Stefano Gogioso. *The Focked-up ZX Calculus: Picturing Continuous-Variable Quantum Computation*. 2024.
arXiv: 2406.02905 [quant-ph].

[TH13] Chia-Wei Tsai and Tzonelih Hwang.
"Deterministic quantum communication using the symmetric W state".
In: *Science China Physics, Mechanics and Astronomy* 56.10 (Oct. 2013),
pp. 1903–1908. ISSN: 1869-1927. DOI: 10.1007/s11433-013-5191-5.

[TM22] Alex Townsend-Teague and Konstantinos Meichanetzidis.
Simplification Strategies for the Qutrit ZX-Calculus. 2022.
arXiv: 2103.06914 [cs.CC].

[TMMB21] Jules Tilly, Ryan J. Marshman, Anupam Mazumdar, and Sougato Bose.
"Qudits for witnessing quantum-gravity-induced entanglement of masses under decoherence". In: *Physical Review A* 104.5 (Nov. 2021).
DOI: 10.1103/physreva.104.052416.

[TYL19] Chia-Wei Tsai, Chun-Wei Yang, and Narn-Yih Lee.
"Semi-quantum secret sharing protocol using W-state".
In: *Modern Physics Letters A* 34.27 (2019), p. 1950213.
DOI: 10.1142/S0217732319502134.
eprint: https://doi.org/10.1142/S0217732319502134.

[Van14] Rodney Van Meter. *Quantum networking*. John Wiley & Sons, 2014.

[vdWet21] John van de Wetering. “Constructing quantum circuits with global gates”. In: *New Journal of Physics* 23.4 (Apr. 2021), p. 043015. DOI: 10.1088/1367-2630/abf1b3.

[vdWW19] John van de Wetering and Sal Wolffs. *Completeness of the Phase-free ZH-calculus*. 2019. arXiv: 1904.07545 [quant-ph].

[vdWY23] John van de Wetering and Lia Yeh. “Building Qutrit Diagonal Gates from Phase Gadgets”. In: *Electronic Proceedings in Theoretical Computer Science* 394 (Nov. 2023), pp. 46–65. ISSN: 2075-2180. DOI: 10.4204/eptcs.394.4.

[Vil19] Renaud Vilmart. “A near-minimal axiomatisation of ZX-calculus for pure qubit quantum mechanics”. In: *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE. 2019, pp. 1–10.

[Vil21a] Renaud Vilmart. “Quantum Multiple-Valued Decision Diagrams in Graphical Calculi”. In: *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*. Ed. by Filippo Bonchi and Simon J. Puglisi. Vol. 202. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 89:1–89:15. ISBN: 978-3-95977-201-3. DOI: 10.4230/LIPIcs.MFCS.2021.89.

[Vil21b] Renaud Vilmart. “The Structure of Sum-Over-Paths, its Consequences, and Completeness for Clifford”. In: *Foundations of Software Science and Computation Structures*. Ed. by Stefan Kiefer and Christine Tasson. Cham: Springer International Publishing, 2021, pp. 531–550. ISBN: 978-3-030-71995-1. DOI: 10.1007/978-3-030-71995-1_27.

[Vil23] Renaud Vilmart. “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 2023)*. Ed. by Bartek Klin and Elaine Pimentel. Vol. 252. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, 36:1–36:17. ISBN: 978-3-95977-264-8. DOI: 10.4230/LIPIcs.CSL.2023.36.

[VK22] Michael Vasmer and Aleksander Kubica. “Morphing Quantum Codes”. In: *PRX Quantum* 3.3 (Aug. 2022). ISSN: 2691-3399. DOI: 10.1103/prxquantum.3.030319.

[VLC+19] Christophe Vuillot, Lingling Lao, Ben Criger, Carmen García Almudéver, Koen Bertels, and Barbara M Terhal.

"Code deformation and lattice surgery are gauge fixing".
In: *New Journal of Physics* 21.3 (Mar. 2019), p. 033028. ISSN: 1367-2630.
DOI: 10.1088/1367-2630/ab0199.

[VLR20] Madhav Krishnan Vijayan, Austin P. Lund, and Peter P. Rohde.
"A robust W-state encoding for linear quantum optics".
In: *Quantum* 4 (Aug. 2020), p. 303. DOI: 10.22331/q-2020-08-03-303.

[VPB10] Tamás Vértesi, Stefano Pironio, and Nicolas Brunner.
"Closing the Detection Loophole in Bell Experiments Using Qudits".
In: *Phys. Rev. Lett.* 104 (6 Feb. 2010), p. 060401.
DOI: 10.1103/PhysRevLett.104.060401.

[Wan18a] Quanlong Wang.
"Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics".
In: *Electronic Proceedings in Theoretical Computer Science* 266 (Feb. 2018),
pp. 58–70. ISSN: 2075-2180. DOI: 10.4204/eptcs.266.3.

[Wan18b] Quanlong Wang.
"Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics".
In: *Proceedings 14th International Conference on Quantum Physics and Logic, Nijmegen, The Netherlands, 3-7 July 2017.*
Ed. by Bob Coecke and Aleks Kissinger. Vol. 266.
Electronic Proceedings in Theoretical Computer Science.
Open Publishing Association, 2018, pp. 58–70. DOI: 10.4204/EPTCS.266.3.

[Wan21a] Quanlong Wang. *A non-anyonic qudit ZW-calculus*. 2021.
arXiv: 2109.11285 [quant-ph].

[Wan21b] Quanlong Wang. "An Algebraic Axiomatisation of ZX-calculus".
In: *Proceedings 17th International Conference on Quantum Physics and Logic, Paris, France, June 2 - 6, 2020*. Ed. by Benoît Valiron, Shane Mansfield,
Pablo Arrighi, and Prakash Panangaden. Vol. 340.
Electronic Proceedings in Theoretical Computer Science.
Open Publishing Association, 2021, pp. 303–332.
DOI: 10.4204/EPTCS.340.16.

[Wan22] Quanlong Wang. *Qufinite ZX-calculus: a unified framework of qudit ZX-calculi*.
2022. arXiv: 2104.06429 [quant-ph].

[Was82] L. C. Washington. *Introduction to Cyclotomic Fields*.
NY: Springer New York, 1982. ISBN: 978-1-4684-0133-2.
DOI: 10.1007/978-1-4612-1934-7.

[WB14] Quanlong Wang and Xiaoning Bian.
"Qutrit Dichromatic Calculus and Its Universality". In: *Electronic Proceedings*

in Theoretical Computer Science 172 (Dec. 2014), pp. 92–101. ISSN: 2075-2180. DOI: 10.4204/eptcs.172.7.

[WCAB15] Fern H. E. Watson, Earl T. Campbell, Hussain Anwar, and Dan E. Browne. "Qudit color codes and gauge color codes in all spatial dimensions". In: *Phys. Rev. A* 92 (2 Aug. 2015), p. 022312. DOI: 10.1103/PhysRevA.92.022312.

[WES+25] Quanlong Wang, Richard D. P. East, Razin A. Shaikh, Lia Yeh, Boldizsár Poór, and Bob Coecke. *Beyond Penrose tensor diagrams with the ZX calculus: Applications to quantum computing, quantum machine learning, condensed matter physics, and quantum gravity*. 2025. arXiv: 2511.06012 [quant-ph].

[WHSK20] Yuchen Wang, Zixuan Hu, Barry C. Sanders, and Sabre Kais. "Qudits and High-Dimensional Quantum Computing". In: *Frontiers in Physics* 8 (2020), p. 479. ISSN: 2296-424X. DOI: 10.3389/fphy.2020.589504.

[WP24] Sam Winnick and Jennifer Paykin. *Condensed Encodings of Projective Clifford Operations in Arbitrary Dimension*. 2024. arXiv: 2407.16861 [quant-ph].

[WQB23] Mark A Webster, Armanda O Quintavalle, and Stephen D Bartlett. "Transversal diagonal logical operators for stabiliser codes". In: *New Journal of Physics* 25.10 (Oct. 2023), p. 103018. ISSN: 1367-2630. DOI: 10.1088/1367-2630/acfc5f.

[WTLY22] Hung-Wen Wang, Chia-Wei Tsai, Jason Lin, and Chun-Wei Yang. "Authenticated Semi-Quantum Key Distribution Protocol Based on W States". In: *Sensors* 22.13 (2022). ISSN: 1424-8220. DOI: 10.3390/s22134998.

[WUR+22] Jordi R. Weggemans, Alexander Urech, Alexander Rausch, Robert Spreeuw, Richard Boucherie, Florian Schreck, Kareljan Schoutens, et al. "Solving correlation clustering with QAOA and a Rydberg qudit system: a full-stack approach". In: *Quantum* 6 (Apr. 2022), p. 687. DOI: 10.22331/q-2022-04-13-687.

[WYK24] Quanlong Wang, Richie Yeung, and Mark Koch. "Differentiating and Integrating ZX Diagrams with Applications to Quantum Machine Learning". In: *Quantum* 8 (Oct. 2024), p. 1491. ISSN: 2521-327X. DOI: 10.22331/q-2024-10-04-1491.

[XFS+22] Haonan Xiong, Quentin Ficheux, Aaron Somoroff, Long B. Nguyen, Ebru Dogan, Dario Rosenstock, Chen Wang, et al. "Arbitrary controlled-phase gate on fluxonium qubits using differential ac Stark shifts". In: *Phys. Rev. Res.* 4 (2 Apr. 2022), p. 023040. DOI: 10.1103/PhysRevResearch.4.023040.

[YDY13] Nengkun Yu, Runyao Duan, and Mingsheng Ying.
"Five two-qubit gates are necessary for implementing the Toffoli gate".
In: *Phys. Rev. A* 88 (1 July 2013), p. 010304.
DOI: 10.1103/PhysRevA.88.010304.

[Yeh23] Lia Yeh. "Scaling W State Circuits in the qudit Clifford Hierarchy".
In: *Companion Proceedings of the 7th International Conference on the Art, Science, and Engineering of Programming*. <Programming> '23 Companion.
ACM, Mar. 2023, pp. 90–100. DOI: 10.1145/3594671.3594687.

[YSK+20] M. A. Yurtalan, J. Shi, M. Kononenko, A. Lupascu, and S. Ashhab.
"Implementation of a Walsh-Hadamard Gate in a Superconducting Qutrit".
In: *Phys. Rev. Lett.* 125 (18 Oct. 2020), p. 180504.
DOI: 10.1103/PhysRevLett.125.180504.

[YvdW22] Lia Yeh and John van de Wetering.
"Constructing All Qutrit Controlled Clifford+T gates in Clifford+T".
In: *Reversible Computation*.
Ed. by Claudio Antares Mezzina and Krzysztof Podlaski.
Cham: Springer International Publishing, 2022, pp. 28–50.
ISBN: 978-3-031-09005-9. DOI: 10.1007/978-3-031-09005-9_3.

[YZZY18] Biaoliang Ye, Zhen-Fei Zheng, Yu Zhang, and Chui-Ping Yang.
"Circuit QED: single-step realization of a multiqubit controlled phase gate with one microwave photonic qubit simultaneously controlling n − 1 microwave photonic qubits". In: *Optics Express* 26.23 (Nov. 2018), p. 30689.
ISSN: 1094-4087. DOI: 10.1364/oe.26.030689.

[ZCC08] Bei Zeng, Xie Chen, and Isaac L. Chuang.
"Semi-Clifford operations, structure of C_k hierarchy, and gate complexity for fault-tolerant quantum computation".
In: *Phys. Rev. A* 77 (4 Apr. 2008), p. 042313.
DOI: 10.1103/PhysRevA.77.042313.

[ZFW10] Ming-Jing Zhao, Shao-Ming Fei, and Zhi-Xi Wang. "Entanglement Property and Monogamy Relation of Generalized Mixed W States".
In: *International Journal of Quantum Information* 08.06 (2010), pp. 905–912.
DOI: 10.1142/S0219749910006216.

[ZLC00] Xinlan Zhou, Debbie W. Leung, and Isaac L. Chuang.
"Methodology for quantum logic gate construction".
In: *Physical Review A* 62.5 (Oct. 2000). ISSN: 1094-1622.
DOI: 10.1103/physreva.62.052316.

[ZLS23] Wei Zi, Qian Li, and Xiaoming Sun.
"Optimal Synthesis of Multi-Controlled Qudit Gates".

In: *2023 60th ACM/IEEE Design Automation Conference (DAC)*. 2023, pp. 1–6. DOI: 10.1109/DAC56929.2023.10247925.

[ZYO+15] Xue-Ping Zang, Ming Yang, Fatih Ozaydin, Wei Song, and Zhuo-Liang Cao. "Generating multi-atom entangled W states via light-matter interface based fusion mechanism". In: *Scientific Reports* 5.1 (Nov. 2015), p. 16245. ISSN: 2045-2322. DOI: 10.1038/srep16245.

Acknowledgements

I didn't learn physics until my last year of high school, so I hadn't considered pursuing it. This all changed thanks to a professor at my undergraduate university who encouraged me to study physics and do research when I hadn't even thought of these as possibilities. I got the idea in my head that I wanted to do quantum computing. I started with going to any academic event with the word "quantum". I watched so many talks that I didn't understand at all. I asked a lot of ~~dumb~~ ignorant questions and read books and papers. Slowly the percentage of what I could follow rose from close to nothing to 10% to higher. So to celebrate the start of what became the present journey, I cannot thank enough Sathya Guruswamy, Phill Conrad, Sara Sterphone, Dave Patterson, Thomas Schibler, Murat Karaorman, Ömer Eğecioğlu, Samantha Davis, Wim van Dam, Margaret Martonosi, and Yipeng Huang for mentoring me during my undergraduate years at the College of Creative Studies, University of California Santa Barbara where I learned many things, most importantly what I wanted to aspire to.

... and then there was crossing an ocean to start a PhD during a pandemic. I couldn't believe that the dream to become a quantum computer scientist could become real. Thank you to Aleks Kissinger and John van de Wetering for supervising this PhD, and for going above and beyond to support my journey. Thank you to the members of the Quantum Group in the Department of Computer Science, University of Oxford for being in this together: Razin Shaikh, Benjamin Rodatz, Richie Yeung, Boldizsár Poór, Shengqiao Luo, Alexander Cowtan, Will Simmons, Amin Karamlou, Matthew Sutcliffe, Nick Ormrod, Luca Mondada, Tiffany Duneau, Nihil Shah, Isaac Friend, Tein van der Lugt, David Gao, Alex Koziell-Pipe, Nicola Pinzani, Hlér Kristjánsson, Augustin Vanrietvelde, Cole Comfort, James Hefford, Matt Wilson, Nick Hu, Vincent Wang-Maścianica, Jonathan Barrett, Matty Hoban, Stefano Gogioso, Sergii Strelchuk, Sam Staton, and more who have joined in the past two years. Thank you to Shuxiang Cao, Maria Violaris, Zev Shirazi, Jack Liell-Cock, Jessica Richards, and Timothy Chan for your friendship sharing the journey. Thank you to Bob Coecke, Quanlong Wang, Sara Sabrina Zemljič, Caterina Puca, Muhammad Hamza Waseem, Selma Dündar-Coecke, and more from the Quantinuum Oxford office for doing good teamwork together. Thank you to Patrick Roy, Edwin Agnew, Haytham McDowall-Rose, Christine Li, Cella Kove, and Maria Dincă for inspiring me through watching

your evolution. Thank you to Oriel College and the Clarendon Scholars community for the Oriel Basil-Reeve Scholarship that funded my first three years of this PhD, and to the Google PhD Fellowship for funding the last two years. Thank you to Cristiana Sessini, Albert Sun, Hannah Jauncey, Pan Ei Ei Phyoe, Su Mon Latt, Tanya Wilson, Marcus Williamson, Maria Fransiska Pudjohartono, Luigi Muci, and Kimberly Wilson for being part of my precious memories at Oriel, through the pandemic and after. I'm holding onto these precious memories made with you all here forever.

Thank you to James Weaver for guiding my education work at IBM Quantum, and to Qubit by Qubit and The Coding School nonprofit for the chance to make a difference teaching quantum computing to hundreds of high school students from around the world. Thank you to Mio Murao, Juani Bermejo-Vega, Yu-Fang Chen, and QuSoft for the cherished experience of visiting your group, marking formative highlights of learning and happiness. Thank you to my thesis committee, Simon Perdrix and Bálint Koczor, for the in-depth reading of this thesis, for the defense, and for your valuable and much appreciated feedback.

Thank you to Quantum Universal Education, Alberto Maldonado-Romo, Samanvay Sharma, Kevin Joven Jofroit Noriega, Jayanti Singh, Praveen J, Ricky Kienhoefer, Bavithra Govintharajah, Mariana Filipova, Brian La Cour, Marek Osiński, and Megan Ivory for inspiring me and for everything we have been building for quantum education. Thank you to Zeki Seskir, İlke Ercan, Angelina Frank, Clara Yun Fontaine, and to all whom I met through Strengthening and Entangling Global QUantum ROots, for being the people behind the grassroots initiatives making the quantum world a lovelier and more welcoming place.

Thank you to friends made through quantum adventures: Sarah Meng Li, Krystal Maughan, Kevin Mato, Maria Gragera Garcés, Jiaxin Huang, Waheeda Saib, Rhea Alexander, Filipa Peres, Giulio Camillo, Arianne Meijer-van de Griend, Gregory Rosenthal, Sahar Ben Rached, Shwetha Jayaraj, Adam Wesolowski, and many more. Here's to future adventures together.

Thank you most of all to my childhood and adulthood friends who bring a lot of joy to my life, and to my family for supporting me and cheering me on. Lastly, tongue in cheek, I thank you, the reader, in advance because your plausible existence stirs me to write this.



Figure 1: Kudos to the cat who mashed a few random letters of this thesis.