Preservation of determinism in MBQC under ZX-calculus rewrites

by Tommy McElvanney

UNIVERSITY OF BIRMINGHAM

A thesis submitted to the University of Birmingham
for the degree of DOCTOR OF PHILOSOPHY

School of Computer Science,
College of Engineering and Physical Sciences
University of Birmingham
November 2024

UNIVERSITY OF BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence.

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

Acknowledgements

Firstly, I would like to thank my supervisor Miriam Backens for all of their support and for their patience with me throughout the whole of my PhD - I am greatly appreciative of everything you have done for me. I would also like to thank my co-supervisor, Martin Escardo, along with the rest of my Thesis group, Paul Levy and Steve Vickers, for their helpful discussions throughout my PhD.

Thank you to Will Simmons for spending time helping me get a deeper understanding of Pauli flow and the circuit extraction routine, as well as for sending me the unpublished version of their circuit extraction paper containing the ZX-calculus version of the circuit extraction - this was incredibly helpful for me and really helped me wrap my head around exactly how the circuit extraction worked. Thank you also to John Van de Wetering for useful conversations about the ZX-calculus and for giving me the opportunity to manage the ZX-calculus YouTube channel.

Thank you to Korbinian Staudacher for plenty of interesting discussions about our research and for giving me ideas on applications of my theoretical work - your visit to Birmingham really helped me to understand the topic area in much greater depth. Thank you also to Piotr Mitosek for joining in on these discussions and for being a great colleague and friend throughout my PhD.

Thank you to my parents and the rest of my family for being supportive and encouraging throughout all of my studies.

Thank you to the EPSRC for funding my studies.

Finally, the biggest thanks to my partner Trisha for always being there for me and encouraging me to do my best. You have always supported me through thick and thin and pushed me to do better than I ever thought I could have and I can't thank you enough. I feel so lucky to have you in my life every single day and I can't wait to see what the future holds for us.

Abstract

The ZX-calculus is a powerful graphical language for reasoning about and optimizing quantum computations, being able to represent and rewrite both quantum circuits and measurement-based quantum computations (MBQCs) in a natural way, along with acting as a tool to translate between the two aforementioned types of quantum computation. Finding a circuit that is equivalent to a ZX-diagram can be #P hard [11] - it is therefore important to look at the classes of ZX-diagrams that we have efficient circuit extraction algorithms for. The most general of these classes are ZX-diagrams with a property known as Pauli flow - this is a property from measurement-based quantum computing which ensures that computations remain deterministic despite quantum measurements being inherently probabilistic. The standard ZX-calculus rules are not particularly well suited to‘ measurement-based quantum computing as they often do not preserve the existence of Pauli flow nor preserve the MBQC structure itself. Thus by developing our understanding of ZX-calculus rewrite rules which preserve the existence of Pauli flow, we can better understand which ZX diagrams we are able to efficiently extract a circuit from as well as which Measurement-based computations we can perform deterministically.

In this thesis, I introduce new ZX-calculus rewrite rules and prove that these rules preserve the existence of Pauli flow. These rules have a variety of applications across many areas of quantum computation which will also be explored - to start, I show that these rules give us a procedure for rewriting any measurement-based quantum computation with arbitrary planar measurements along with X, Y and Z Pauli measurements into a diagram containing only XY-planar measurements and X and Y Pauli measurements (that is, only containing green spiders in the ZX-calculus). I then show that when we restrict to the stabilizer fragment of quantum computing - where we only have Pauli-measurements in measurement-based quantum computations - we are able to find a complete set of rewrite rules which preserve the existence of Pauli flow. To do this, I introduce a procedure for rewriting any stabilizer ZX-diagram into a unique normal form using only flow-preserving rewrite rules.

Finally, I examine the flow preservation of the "neighbour unfusion" rule of [60]. This rule was used in an optimization procedure for reducing 2-qubit gate count but the authors did not know in what cases this rule preserved gflow which was important for their algorithm to function. Neighbour unfusion is a special case of one of the rules which I proved to preserve Pauli flow and so it also preserves Pauli flow, but it is not as simple in the case of gflow. I find a condition that is both sufficient and necessary for neighbour unfusion to preserve the existence of gflow, along with providing less strict conditions for gflow to be preserved that can function as useful heuristics in their algorithm.

Contents

Chapter 1

Introduction

Quantum computing is a powerful and impactful field of research which is currently going through a boom - investment in quantum computing from governments around the world is at an all time high [56] and many companies and countri,ies are racing one another to have the most powerful quantum computer (currently being dominated by Quantinuum with their quantum computers H1-1 and H2) [54]. There are various different experimental methods that are being worked on to build quantum computers, including trapped ions, superconducting qubits and photonics. Among the companies working on quantum computers, IBM, Google and Rigetti are all working on superconducting qubits, IonQ and Quantinuum are working on trapped ions, while Xanadu, PsiQuantum and Quandela are working on photonic quantum computers.

Alongside the experimental work done to build quantum computers, there is much theoretical work being done on many areas important to quantum computing, including quantum algorithms, quantum error correction and quantum circuit optimization. Quantum algorithms research involves finding new algorithms that can be run on quantum computers to solve problems faster than classical computers, with the most famous examples being Shor's algorithm [57] for factoring large numbers. Quantum error correction focuses on finding ways to correct for errors that occur in quantum computations as quantum computers are inherently noisy and errors can occur during the computation. Quantum

circuit optimization is a field of research which focuses on finding ways to reduce the number of gates in a quantum circuit, as well as reducing the depth of the circuit (the number of sequential gates that need to be applied) and the number of qubits used in the circuit.

As quantum computaions are easily impacted by noise and we currently are not easily able to perform wide-scale error correction, it is said that we are in the Noisy Intermediate scale Quantum (NISQ) era [12] - it is therefore especially important to focus on optimization of quantum circuits while we are not fully able to perform error correction as a more optimized computation requiring less gates leads to less errors caused by the imperfect implementation of said gates.

There have been many different approaches to optimizing quantum circuits, from directly re-writing circuit diagrams [20], using path-sum techniques to using machine learning techniques to brute-force the optimization [19, 32]. One particularly interesting tool for optimization of quantum circuits is the ZX-calculus, a graphical language for reasoning about (and rewriting) quantum computations. The ZX-calculus is universal, meaning that any linear map between Hilbert spaces can be represented by a ZX-diagram, as well as complete, which means that any two diagrams that represent the same linear map are able to be rewritten into one another using the ZX-calculus' rewrite rules. These qualities have led to the ZX-calculus being used for a wide variety of applications in quantum computing, including circuit optimization [27, 43, 60], error-correction [23, 41] and classical simulation [44, 61] among other applications.

The ZX-calculus is inspired by Roger Penrose's original graphical calculus for reasoning about tensor networks and was developed initially by Bob Coecke and Ross Duncan [21] (based on Coecke's earlier work on categorical quantum mechanics with Samson Abramsky [1]), with large contributions by many others including Aleks Kissinger and Miriam Backens. Of paricular importance to this thesis is the work done by Backens [6] on the completeness of the ZX-calculus for stabilizer quantum mechanics, which is a class of quantum mechanics that is efficiently simulable on a classical computer. After this,

completeness results were found for other fragments of the ZX-calculus, including the Clifford+T fragment [35] and the full universal ZX-calculus [52]. Other graphical calculi have also been developed for reasoning about quantum computations, including the ZW-calculus [34] and the ZH-calculus [8], both of which are also universal and complete - these calculi are not as widely used as the ZX-calculus and I will not be discussing them in this thesis.

One particularly interesting feature of the ZX-calculus is that it is also a natural model for measurement-based quantum computation (MBQC) [9, 29], a different paradigm of quantum computing where one prepares a large entangled "resource state" and performs the computation by sequential measurements of different qubits in said resource state. Instead of sequentially applying quantum gates to qubits as in the quantum circuit model, the measurement of physical qubits essentially applies the unitary maps to logical qubits carrying through the measurement-based computation.

Measurement-based quantum computations are inherently non-deterministic as each measurement has two possible outcomes - we therefore require extra conditions to be satisfied for measurement-based computations to have a deterministic outcome. In particular, we need to be able to correct for any undesired measurement outcomes by changing the measurement angles of qubits to be measured in the future. Conditions which ensure these corrections are possible are known as flow conditions [15, 24], with Pauli flow [15, 58] being the most general of these flow conditions.

To be able to optimize quantum circuits using the ZX-calculus, it is important to be able to effecively translate between quantum circuits and ZX-diagrams. While it is easy to convert any quantum circuit into a ZX-diagram (as each quantum gate has a simple translation to a ZX-diagram), the reverse problem of finding a circuit that is equivalent as a linear map to an arbitrary ZX-diagram is in general #P hard [11]. Diagrams with Pauli flow are the most general class of ZX-diagrams for which we have a polynomial time circuit extraction algorithm [58], thus if we are able to apply Pauli-flow-preserving rewrite rules (rules for which if the left side of the equality has Pauli flow, the right side of the equality

also has Pauli flow) to a ZX-diagram with Pauli flow, we know we will be able to easily extract a circuit back out at the end. As ZX-diagrams corresponding to circuits (those which can be constructed from only rotations, Hadamards and CNOT gates) naturally have Pauli flow [58], flow-preserving rewrite rules provide us with a powerful tool for performing circuit optimization tasks.

Another important area of quantum computing research is that of blind quantum computation [31, 45]. As quantum computers are expensive and difficult to produce and maintain, it is likely that most people using quantum computers will be doing so by remotely accessing a quantum server. This leaves room for interference from the owner of said server, or for a bad actor to spy on the computation and discover the results. Blind quantum computation protocols restrict the amount of information, whether it be inputs, outputs or the algorithm itself, able to be obtained by a bad actor trying to interfere with the computation. Flow-preserving rewrite rules have also found applications here, allowing the user to add new qubits to be used as "traps" or to disguise the connectivity of the algorithm they are trying to implement while maintaining determinism and still implementing the same linear map [17].

For a graphical language like the ZX-calculus to be used optimally, we need to be able to rewrite any two equivalent diagrams into one another using only the rules of the language - this is a property known as completeness. Since Backens' original stabilizer completeness proof [6], there have been a multitude of completeness results for different fragments of the ZX-calculus [39] [35], [52]. These results are not so applicable in the context of MBQC as the rewrite rules used often break the MBQC-form structure of a diagram or do not preserve any flow conditions and thus can lead to non-determistic computations when applied to a diagram with flow. It is therefore interesting to consider whether it is possible to find a set of MBQC-form-preserving and flow-preserving rewrite rules that are complete for different fragments of the ZX-calculus.

Chapter 2 introduces quantum circuits and the ZX-calculus, explaining the rewrite rules which make up the Stabilizer ZX-calculus' ruleset and covering Backens' original com-

pleteness result for the Stabilizer ZX-calculus [6]. Chapter 3 covers Measurement-based quantum computing (MBQC) as well as each of the flow conditions, from least to most general, then discusses flow-preserving rewrite rules that have been used previously in the literature [27]. Chapter 4 contains all of the ZX-calculus rewrite rules which we proved to preserve the existence of Pauli flow in both of our published papers [[46] and [46]]. Chapter 5 contains our work on complete flow-preserving ZX-calculus rewrite rules published in QPL2022 [46], while Chapter 6 contains our work on neighbour unfusion published in QPL2023 [47].

All of the work that was published (i.e. chapters 4, 5 and 6) is co-written by Miriam Backens. They contributed largely to the sections on complete flow preserving rewrite rules, specifically the parts involving the holant literature, as well as contributing to many of the proofs throughout the work. They also provided me with the idea of looking further into Pauli-flow preserving ZX-calculus rewrite rules.

Chapter 2

Quantum Circuits and the ZX-calculus

In this section we shall introduce quantum circuits, give an overview of the ZX-calculus and show that the ZX-calculus is complete for the stabilizer fragment of quantum mechanics.

2.1 Qubits

Quantum computation has traditionally used a circuit model similar to the logic-gate model for classical computations. New developments in diagrammatic reasoning have allowed us to find a much more powerful language - known as the ZX-calculus - in which we can represent quantum algorithms in a similar way to circuits, but are also able to easily manipulate the diagrams to perform optimization tasks - we are able to rewrite/optimize circuits directly, but due to the rigid structure of quantum circuits and the complexity of the rewrite ruels, this is not ideal. The ZX-calculus is more flexible than quantum circuits and the rewrite rules are intuitive, often having some algebraic meaning. We will first introduce the concept of quantum circuits and use them to motivate the ZX-calculus.

The circuit model of classical computation involves applying logic gates to bits. In much the same way, the circuit model of quantum computation involves applying quantum gates to qubits. We shall now define these terms.

Definition 2.1.1. A qubit is an element of $\mathbb{C}^2$, that is to say that it takes the following form, where $a, b \in \mathbb{C}$.

$$\begin{bmatrix} a \\ b \end{bmatrix}$$

A qubit is normalised if it has norm 1, i.e. $|a|^2 + |b|^2 = 1$.

Using only normalised qubits removes a lot of un-needed complication from calculations without losing anything of importance, thus henceforth we shall implicitly mean "normalised qubit" whenever we say qubit.

Much like how bits can be in the state 0 or 1, qubits can be in the following two important states, known as the standard, canonical or Z basis.

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

The key difference between classical and quantum computing is that qubits can exist in any superposition of $|0\rangle$ and $|1\rangle$ - in general, we may write the state of any qubit as $|\psi\rangle = a\,|0\rangle + b\,|1\rangle$ with $a, b \in \mathbb{C}$. Qubits 'live' in a Hilbert space, which is a vector space with a distance function such that the space is a compete metric space with respect to said distance function. Given two Hilbert spaces H and H', we can form the combined space $H \otimes H'$ where $\otimes$ is called the tensor or monoidal product. The combined state of two independent quantum systems is equal to the tensor product of the states of the individual systems.

$|0\rangle$ and $|1\rangle$ do not form the only basis over $\mathbb{C}^2$. We will also regularly use the following basis, known as the Hadamard or X basis.

$$|+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

and

$$|-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Here the $\frac{1}{\sqrt{2}}$ term is needed for normalisation.

The two aforementioned bases are known as the Z and X bases as these states (vectors) are the eigenstates (eigenvectors) of the Pauli Z and Pauli X operators respectively. The Pauli operators commonly appear throughout all of quantum mechanics and computing, and are defined as follows.

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

The eigenstates of a matrix/operator are the states for which the operator only acts as a scalar when applied to said states - that is, for a matrix M, its eigenstates $|\psi\rangle$ are those that satisfy $M|\psi\rangle = \lambda|\psi\rangle$ for some $\lambda \in \mathbb{C}$. We can then see that $|0\rangle$ and $|1\rangle$ are eigenstates of Pauli-Z and that $|+\rangle$ and $|-\rangle$ are eigenstates of Pauli-X.

$$Z|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$Z|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} = -|0\rangle$$

$$X|+\rangle = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle$$

$$X|-\rangle = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \tfrac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = -|-\rangle$$

Another important property of qubits that distinguishes quantum computing from regular computing is entanglement. We say that a product state of two (or more) qubits is *separable* if it can be written as the tensor product of two single qubit states. A multi-

qubit state that is not separable is said to be entangled. The most commonly used entangled states are the Bell states (or Bell basis), given below.

$$|\phi^+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle)$$

$$|\phi^-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle - |1\rangle \otimes |1\rangle)$$

$$|\psi^+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle)$$

$$|\psi^+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle \otimes |1\rangle - |1\rangle \otimes |0\rangle)$$

We can now see how measuring one of the entangled qubits tells us the state of the other - say we measured the first qubit of a pair in the state $|\phi^+\rangle$ and found it was in the state $|0\rangle$. We then know that the second qubit is also in the state $|0\rangle$ by the following;

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

2.2 Quantum circuits

2.2.1 Single Qubit gates

The most commonly used model of quantum computing is the quantum circuit model. This involves applying a sequence of quantum gates to qubits and then performing measurements to read out data at the end of the computation. We shall define quantum gates and give examples of them here.

Definition 2.2.1. A unitary map is a linear map $U : H \to H$ on a Hilbert space H satisfying $U^\dagger U = UU^\dagger = I$, where $I : H \to H$ is the identity map.

A quantum gate on d-dimensional qudits is a unitary map $U : \mathbb{C}^{d^n} \to \mathbb{C}^{d^m}$ between two Hilbert spaces corresponding to the state spaces of qudits. As we shall be working with only qubits, quantum gates will be unitary maps from $\mathbb{C}^{2^n}$ to $\mathbb{C}^{2^m}$.

Examples of single qubit gates include the Pauli matrices (X,Y and Z) defined in the previous section, along with the following three important gates.

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \; S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \text{ and } T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}$$

Note that $T^2 = S$ and $S^2 = Z$. Moreover, we can define arbitrary Z and X 'rotations' as follows.

$$R_Z(\alpha) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

$$R_X(\alpha) = \begin{bmatrix} \cos(\frac{\alpha}{2}) & -i\sin(\frac{\alpha}{2}) \\ -i\sin(\frac{\alpha}{2}) & \cos(\frac{\alpha}{2}) \end{bmatrix}$$

We can then see that $Z = R_Z(\pi)$, $S = R_Z(\frac{\pi}{2})$, $T = R_Z(\frac{\pi}{4})$ and $X \cong R_X(\pi)$, where $\cong$ means equality up to some scalar s with $|s| = 1$ (there are several valid definitions of R_X - with this definition of $R_X(\alpha)$ we have $X = iR_X(\pi)$). Phases (scalars with modulus 1) have no effect on the overall computation and can be corrected for classically, thus we are able to ignore these without any impact on the validity of our calculations. This is also the reason why there are several valid definitions for R_X (and also R_Z) - $e^{i\alpha}R_X$ is a valid definition for all α due to phases not changing anything.

A Z-rotation of angle α sends a state $a\,|0\rangle + b\,|1\rangle$ to $a\,|0\rangle + e^{i\alpha}b\,|1\rangle$, introducing a phase of $e^{i\alpha}$ between the two Z-basis states. Similarly, an X-rotation of angle α sends a state $a\,|+\rangle + b\,|-\rangle$ to $a\,|+\rangle + e^{i\alpha}b\,|-\rangle$, introducing a phase of $e^{i\alpha}$ between the two X-basis states.

The aforementioned H-gate, known as the Hadamard gate, is the basis-change matrix taking the Z-basis to the X-basis and vice versa. This will appear frequently going forward and thus it is important to develop a good understanding of its uses. We have the following.

$$H\,|0\rangle = |+\rangle \text{ and } H\,|1\rangle = |-\rangle.$$

$$H\,|+\rangle = |0\rangle \text{ and } H\,|-\rangle = |1\rangle.$$

$$H \circ R_Z(\alpha) \circ H = R_X(\alpha) \text{ and } H \circ R_X(\alpha) \circ H = R_Z(\alpha)$$

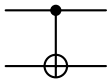
$H \circ H = id$, where id is the identity.

2.2.2 Two qubit gates and universality

All of the quantum gates we have introduced so far have only applied to one qubit - to be able to perform any useful computations, we must also be able to have different qubits interact with each other in some way. The most commonly used two qubit gate is the $CNOT$ or Controlled-NOT gate - this requires us to have one qubit be the 'control' and the other the 'target'. If the control qubit is in the state $|1\rangle$, the $CNOT$ gate acts by applying a NOT (Pauli-X) gate to the target, and acts as the identity if the control is in the state $|0\rangle$ (extending to all other states by linearity). Explicitly, we have the following.

$$CNOT = |0\rangle\langle 0| \otimes id + |1\rangle\langle 1| \otimes X$$

In quantum circuit notation, $CNOT$ gates are represented as follows.

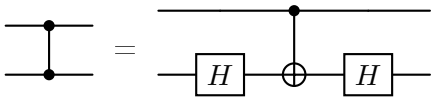


The two-qubit $CNOT$ gate along with single qubit $R_Z(\alpha)$ and H gates are sufficient to represent all quantum circuits [53]. Alternatively, the $CNOT$ gate along with single qubit S, T, Z and H gates are sufficient to approximate any quantum circuit to arbitrary precision - this gate set is known as Clifford+T [14].

It is often not practical to only use the aforementioned gates, and can be useful to define some other quantum gates even if they are simply composites of CNOTs, Z rotations and Hadamard gates - in particular, the Controlled-Z (CZ) gate will be of much use to us later on. The CZ gate again has a control and a target and applies a Pauli-Z to the target if the control is in the state $|1\rangle$, however this gate is symmetrical and so it does not actually matter which qubit we choose to be the control and which is the target. Again, explicitly we have the following.

$$CZ = |0\rangle\langle 0| \otimes id + |1\rangle\langle 1| \otimes Z$$

The CZ-gate can be written in terms of $CNOT$ gates and H-gates as follows.



2.3 The ZX-calculus

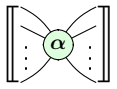
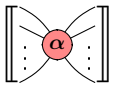
Quantum circuit diagrams are useful for graphically representing quantum computations, but are not so suited for use in optimization or reasoning about quantum computations in general due to their rigid structure and due to the circuit rewrite rules being unintuitive and confusing. The ZX-calculus is an alternative to quantum circuit diagrams that allows us to both represent and reason about quantum computations easily and will be the focus of this thesis. We will provide a brief introduction to the ZX-calculus here; for a more thorough overview, see [22, 64].

2.3.1 ZX-diagrams

A ZX-diagram consists of *spiders* and *wires*. There are different conventions for how to read the diagrams - we will use the convention that diagrams are read from left to right: wires entering a diagram from the left are inputs while wires exiting the diagram on the right are outputs, like in the quantum circuit model. ZX-diagrams compose in two distinct ways: *sequential composition*, which involves connecting the output wires of one diagram to the input wires of another, and *parallel composition* (or the tensor product), which just involves drawing one diagram vertically above the other. The linear map corresponding to a ZX-diagram D is denoted by $[\![D]\!]$.

ZX-diagrams are generated by two families of spiders which may have any number of inputs or outputs, corresponding to the Z and X bases respectively. Z-spiders are drawn as green dots and X-spiders as red dots; with m inputs, n outputs, and using $(\cdot)^{\otimes k}$ to

denote a k-fold tensor power, we have:

 $= |0\rangle^{\otimes n} \langle 0|^{\otimes m} + e^{i\alpha} |1\rangle^{\otimes n} \langle 1|^{\otimes m}$  $= |+\rangle^{\otimes n} \langle +|^{\otimes m} + e^{i\alpha} |-\rangle^{\otimes n} \langle -|^{\otimes m}$

Spiders with exactly one input and output are unitary, in particular $[\![-\alpha-]\!] = |0\rangle\langle 0| + e^{i\alpha}|1\rangle\langle 1| = R_Z(\alpha)$ and $[\![-\alpha-]\!] = |+\rangle\langle +| + e^{i\alpha}|-\rangle\langle -| \cong R_X(\alpha)$.

Two diagrams D and D' are said to be equivalent if $[\![D]\!] = z\,[\![D']\!]$ for some non-zero complex number z. For the rest of the paper, whenever we write a diagram equality we will mean equality up to some global scalar in this way - as long as there is no non-zero scalar, we know that we could have used the ZX-calculus with scalars and tracked the scalars if needed, and so we can take equality up to any non-zero scalar when using the scalar-free ZX-calculus. For treatments of the ZX-calculus which do not ignore scalars see [7] for the stabilizer fragment, [39] for the Clifford+T fragment and [35, 40] for the full ZX-calculus.

The Hadamard gate $H = |+\rangle\langle 0| + |-\rangle\langle 1| \cong Z_{\frac{\pi}{2}} \circ X_{\frac{\pi}{2}} \circ Z_{\frac{\pi}{2}}$ will be used frequently. The bra-ket definition of H is somewhat obvious as it is defined as the gate which sends $|0\rangle$ to $|+\rangle$ and $|1\rangle$ to $|-\rangle$ - the decomposition as Z and X rotations is less obvious and we shall show that this holds now.

$$Z_{\frac{\pi}{2}} \circ X_{\frac{\pi}{2}} \circ Z_{\frac{\pi}{2}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{bmatrix} \begin{bmatrix} \cos(\frac{\alpha}{2}) & -i\sin(\frac{\alpha}{2}) \\ -i\sin(\frac{\alpha}{2}) & \cos(\frac{\alpha}{2}) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i \\ 1 & i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

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

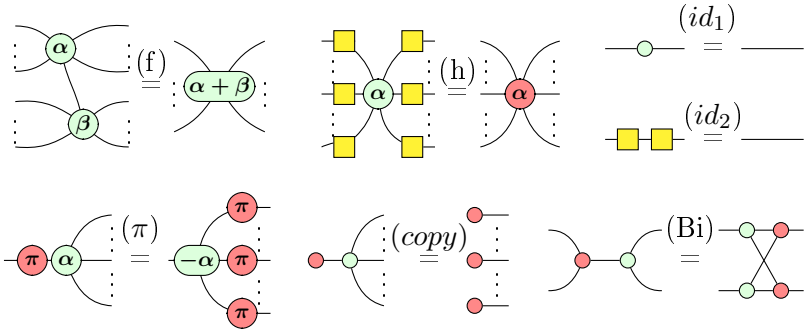


Figure 2.1: A complete set of rewrite rules for the scalar-free stabilizer ZX-calculus. Each rule also holds with the colours or the directions reversed.

The Hadamard gate has two common graphical representations – a yellow square, or a blue dotted line – with the latter only used between green spiders:

2.3.2 ZX-calculus rewrite rules

The ZX-calculus is equipped with a set of rewrite rules which can be used to transform a ZX-diagram into another diagram representing the same linear map. We give a rule set for the stabilizer ZX-calculus in Figure 2.3.2 - we shall give a formal definition of what "stabilizer" means in section 2.4. Note that each rule also holds with the colours swapped from green to red and vice-versa and also hold with directions reversed.

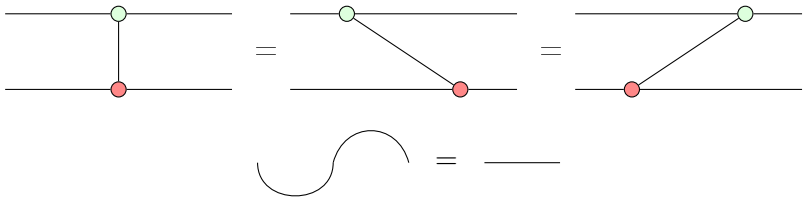
Together with the definition of —■— and the meta-rule "only topology matters", this set of rewrite rules is complete: any two stabilizer ZX-diagrams which correspond (up to non-zero scalar factor) to the same linear map can be rewritten into one another using these rules [6].

The stabilizer ZX-calculus is only complete for diagrams where each phase is a multiple of $\frac{\pi}{2}$ – there exist complete sets of rewrite rules for the Clifford+T fragment [35, 39] (where each phase is a multiple of $\frac{\pi}{4}$) as well as the ZX-calculus with arbitrary phases [40, 52], but these just add extra (complicated and less intuitive) rules onto the stabilizer ZX-calculus' set of rewrite rules, and so we shall focus on the Stabilizer rules here. In the following sections we shall explain each of the Stabilizer ZX-calculus rewrite rules in more

detail.

2.3.3 Only Connectivity Matters

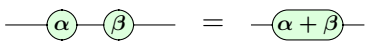
ZX-diagrams form a dagger compact category (more specifically a PROP - we shall not be covering category theory in detail here as it will not be relevant for the rest of our work; for more on this see [36]). Additionally, ZX-diagrams satisfy a property known as flex-symmetry which allows legs of spiders to be swapped and bent around arbitrarily [18]. As a result of this, we are able to treat ZX-diagrams as graphs - only the connectivity of the diagram matters and not how the diagram is embedded in the plane. This allows us to move around spiders at will and arbitrarily stretch, bend and twist wires without affecting the interpretation of the diagram. Some examples of this meta-rule in action are given below.



2.3.4 Spider Fusion

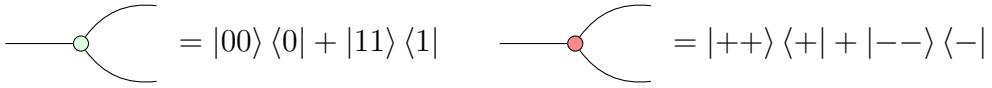
The first rule in 2.3.2 is known as spider fusion and it says that connected spiders of the same colour can be merged together, summing their phases. We can also read this rule from right to left and arbitrarily decompose a spider, splitting off any subset of the wires entering and leaving the spider as well as splitting off whatever phase we like onto a new spider (providing the phases still sum to the original phase).

Spider fusion is easily seen to hold when each spider has only two legs - this then says that doing a $R_Z(\alpha)$ rotation then a $R_Z(\beta)$ rotation is the same as doing a single $R_Z(\alpha + \beta)$ rotation, as seen below.

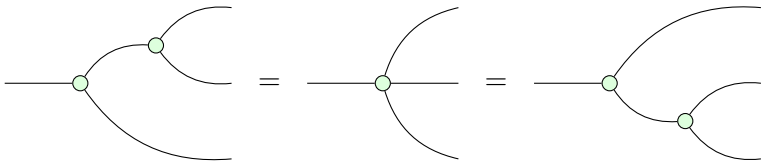


When the spiders have multiple legs, we can see that spider fusion holds from the definition

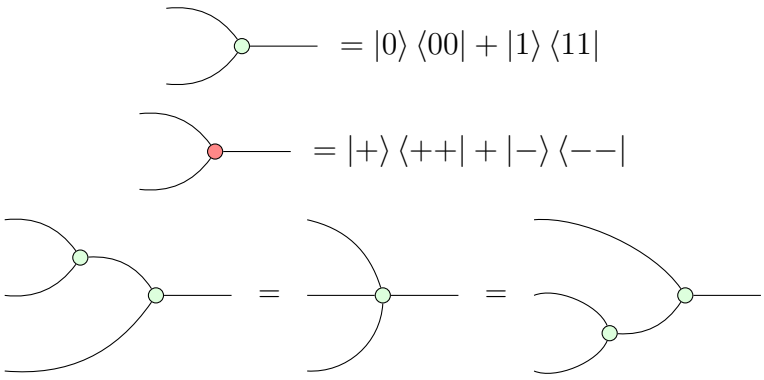
of the spiders and by thinking of the spiders as copy maps for their corresponding bases. Consider the one-input, two-output phaseless spiders.



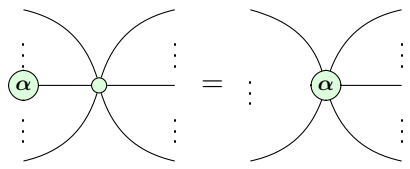
If we input $|0\rangle$ or $|1\rangle$ into the green spider, we get two $|0\rangle$s or two $|1\rangle$s output (similarly with $|0\rangle$ and $|1\rangle$ replaced with $|+\rangle$ and $|-\rangle$ in the case of the red spider). We can therefore see the one-input, two-output spiders as copy maps for the basis elements. We can generalise this to copying a basis element more than once by connecting many of these one-input, two-output spiders together – we then see that this operation being associative is a consequence of the spider fusion rule.



Similarly, if we look at the adjoint of the copy maps, that is the 2-input, 1-output phaseless spiders, we see that we can generalise these to n-input, 1 output maps in a similar way.



We can therefore think of more complex spiders with multiple inputs and outputs as compositions of the copy maps and their adjoints via the spider fusion rule. We can introduce phases by connecting a 1-arity spider with a phase onto one of the legs of our spider which through spider fusion adds the phase to the spider as a whole.

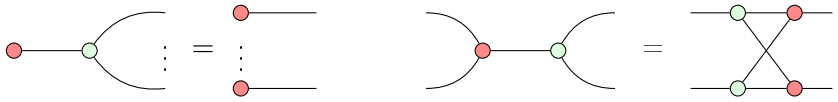


In Dirac notation we have $(|0^n\rangle\langle 0^m| + |1^n\rangle\langle 1^m|)(|0\rangle + e^{i\alpha}|1\rangle) = |0^n\rangle\langle 0^{m-1}| + e^{i\alpha}|1^n\rangle\langle 1^{m-1}|$.

We can see that spider fusion holds here simply through the Dirac notation interpretation of the diagrams. We can therefore think of generalised spiders with phases as a composition of copy maps, their adjoints and single-legged phase spiders.

2.3.5 Strong Complementarity

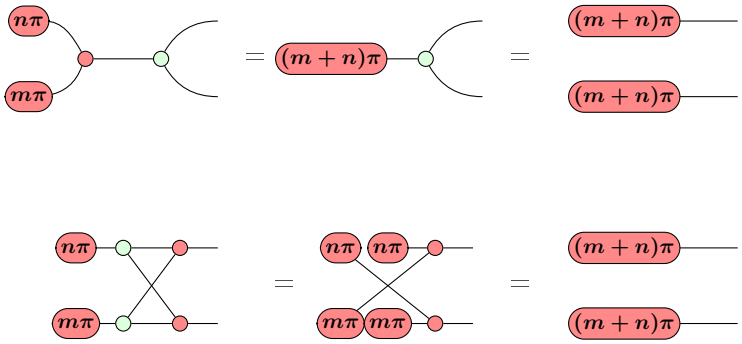
The families of green and red spiders satisfy a property known as strong complementarity. In the language of the ZX-calculus, this means the following rules hold (as well as with directions and colours reversed).



The first rule, known as the copy rule, holds as 0-phase 1-legged spiders are the basis elements that are copied by spiders of the other colour - in Dirac notation we have $(|0^n\rangle\langle 0| + |1^n\rangle\langle 1|)|0\rangle = |0^n\rangle$ (and similarly with the X-basis when the colours are reversed).

Example 2.3.1. The second rule is not quite as easy to see where it comes from. This is known as the bialgebra rule as it comes from the copy maps of one colour and the adjoint of the copy map of the other colour form an algebraic structure known as a bialgebra. A simplified way of thinking of this is that for Z-basis elements, the 1-input, 2 output phaseless green spider acts as the copy map and the 2-input, 1-output red spider acts as an XOR (again with the same holding for the X-basis with the colours reversed).

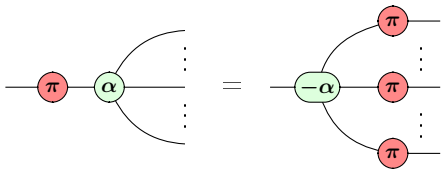
We then see that on basis elements, applying an XOR to two inputs and copying the result is the same as copying both inputs and applying two XOR gates, each with one copy of each of the inputs, seen diagrammatically as follows.



For more on the bialgebra rule and to see a full proof of why this holds, see [22].

2.3.6 π-commutation

The final stabilizer ZX rewrite rule to explain is π-commutation, given diagrammatically as follows.



This essentially boils down to $R_X(\pi)$ gates acting as NOT gates for the Z-basis (and similarly $R_Z(\pi)$ gates acting as NOT gates for the X-basis). We can see this holds in Dirac notation.

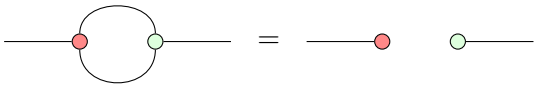
$$\begin{aligned}(|0^n\rangle\langle 0| + e^{i\alpha}|1^n\rangle\langle 1|)(|0\rangle\langle 1| + |1\rangle\langle 0|) &= |0^n\rangle\langle 1| + e^{i\alpha}|1\rangle^n\langle 0| \\ &= (|0\rangle\langle 1| + |1\rangle\langle 0|)^{\otimes n}(|1^n\rangle\langle 1| + e^{i\alpha}|0^n\rangle\langle 0|) \\ &= e^{i\alpha}(|0\rangle\langle 1| + |1\rangle\langle 0|)^{\otimes n}(|0^n\rangle\langle 0| + e^{-i\alpha}|1^n\rangle\langle 1|)\end{aligned}$$

Here the first map corresponds to the diagram on the left hand side of the equality in the π-commutation rule and the final map is the same as the diagram on the right up to a global phase of $e^{i\alpha}$. Viewing the green spider as a generalised copy map, this essentially says that applying a NOT before copying is the same as applying a NOT to each copy (as long as we correct for the phase change).

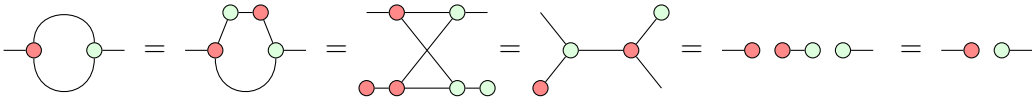
2.3.7 Other useful derivable rewrite rules

From the rules given in 2.3.2, we are able to derive many other useful rules. We shall introduce a few of the more important derivable rules here =- to read more on this see [10]. We shall begin by deriving the Hopf law which is used to disconnect two opposite coloured spiders with two legs connecting them - this rule is often given as an axiom of the calculus but is derivable from the other rules we have given.

Proposition 2.3.2. *The "Hopf law" given below is derivable in the ZX-calculus.*

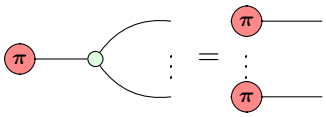


Proof. Consider the following sequence of rewrites.

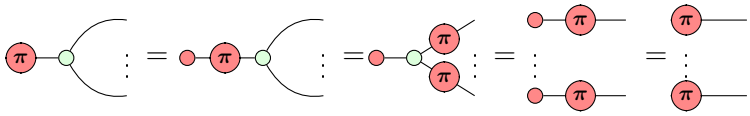


Here the first equality uses the identity rule for two-legged phaseless spiders, the second uses spider unfusion and "only connectivity matters", the third uses the bialgebra rule and the forth uses the copy rule. The final equality is because diagrams with no inputs or outputs correspond to scalars (moreover this scalar is equal to $(|+\rangle + |-\rangle)(\langle 0| + \langle 1|) = |+\rangle\langle 0| + |-\rangle\langle 0| + |+\rangle\langle 1| + |-\rangle\langle 1| = |+\rangle\langle 0| + |-\rangle\langle 0|$ which is non-zero) and we take equality to be up to some scalar factor. □

Proposition 2.3.3. *The "π-copy" rule given below is derivable in the Zx-calculus.*



Proof. Consider the following sequence of re-writes.



Here the first equality uses spider unfusion, the second uses π-commutation, the third uses the copy rule and the last uses spider fusion. □

2.4 Stabilizers

Stabilizers are an important concept in group theory that also appear heavily in the Quantum computing literature and will be of importance going forward. We have already mentioned the stabilizer fragment of quantum computing and in this section we shall properly define what this means. We begin by introducing how we can act upon elements of a set with elements of a group.

Definition 2.4.1. If G is a group with identity id and X is a set, a (left) group action α of G on X is a function $\alpha : G \times X \to X$ satisfying the following axioms for all $g, h \in G$ and all $x \in X$:

1. $\alpha(id, x) = x$
2. $\alpha(g, \alpha(h, x)) = \alpha(gh, x)$

An example of a group action could be applying elements from the group of unitary maps on a single qubit to the set of all ZX-diagrams. The maps of particular interest to us here are those which shall act as the identity on certain ZX-diagrams - this is exactly what a stabilizer is, as we shall define below.

Definition 2.4.2. Suppose we are given a group G and a set X. For every $x \in X$, the stabilizer subgroup of G with respect to x is the set of all elements of G that act as the identity map on x:

$$G_x = \{g \in G | gx = x\}$$

Going forward the stabilizers of quantum states will be of particular use to us - the reason for this is most easily demonstrated using an example.

Example 2.4.3. Consider the following Bell state on two qubits:

$$|\phi^-\rangle = \tfrac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$

Notice that the following equalities hold:

$$Z_1 Z_2 |\phi^-\rangle = -Z_1 |\phi^-\rangle = |\phi^-\rangle$$

$$-X_1X_2\left|\phi^-\right\rangle = -X_1(\tfrac{1}{\sqrt{2}}(|01\rangle - |10\rangle)) = -\tfrac{1}{\sqrt{2}}(|11\rangle - |00\rangle) = \left|\phi^-\right\rangle$$

Therefore Z_1Z_2 and $-X_1X_2$ are stabilizers of $\left|\phi^-\right\rangle$ (alternatively we say that $\left|\phi^-\right\rangle$ is stabilized by Z_1Z_2 and $-X_1X_2$). More importantly (and less obviously), $\left|\phi^-\right\rangle$ is the *unique* two-qubit quantum state that is stabilized by Z_1Z_2 and $-X_1X_2$ - sometimes quantum states can be more easily described by their stabilizers, which is why we care so much about them [53].

In the previous section, we introduced the stabilizer ZX-calculus rewrite rule set, without fully defining what this means - we shall introduce this properly now. First, recall that the Pauli operators are given by the following matrices;

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

Definition 2.4.4. The Pauli group on n qubits, P_n, consists of all tensor products of Pauli and identity matrices with phase factors in $\{\pm 1, \pm i\}$, i.e.

$$P_n = \{\alpha g_1 \otimes g_2 \otimes \ldots \otimes g_n | \alpha \in \{\pm 1, \pm i\} \text{ and } g_i \in \{I, X, Y, Z\} \text{ for } i = 1, 2, \ldots, n\}$$

An n-qubit quantum state is called a *stabilizer state* if it is stabilized by P_n [6].

Of particular importance are the Clifford operators, which always map stabilizer states to other stabilizer states. We shall also give the formal definition for Clifford operators here.

Definition 2.4.5. The Clifford group on n-qubits, denoted C_n, is the group of operators which normalize the Pauli group;

$$C_n = \{U | \forall g \in P_n : UgU^\dagger \in P_n\}$$

Importantly, it can be shown that the Clifford group is generated by S, H and $CNOT$ [53].

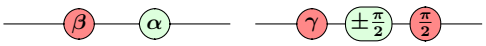
2.5 Completeness of the ZX-calculus for the Stabilizer fragment

In this section we shall introduce everything necessary to show that the ZX-calculus is complete for the stabilizer fragment. Completeness means that any two diagrams representing the same linear map can be rewritten into one another using the rules of the calculus - we focus on the stabilizer fragment here as it is most relevant to our work later in this thesis and also is the most simple to understand.

Definition 2.5.1 ([28])**.** A *graph state diagram* is a ZX-diagram where each vertex is a (phase- free) green spider, each edge connecting spiders has a Hadamard gate on it, and there is a single output wire incident on each vertex. A ZX-diagram is in *graph state with local Clifford (GS-LC) form* if it is a graph state up to single qubit Clifford operators on the input and output wires.

As we are working in the stabilizer fragment, there are a finite number of possible single qubit Clifford operators - it is useful to classify these somehow.

Lemma 2.5.2. *Any single qubit Clifford operator can be written uniquely in one of the two following forms:*

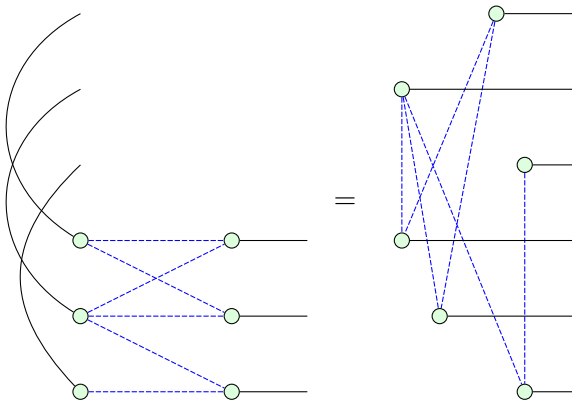


with $\alpha, \beta, \gamma \in \{0, \frac{\pi}{2}, \pi, -\frac{\pi}{2}\}$.

Proof. The fact that any Clifford operator can be rewritten into these forms follows from applying the spider rule, the π-commutation rule and the Euler decomposition of the Hadamard, noting that $S = -SZ$. The uniqueness comes from that there are 24 distinct normal forms shown here which is equal to the size of the single qubit Clifford group [6]. □

All stabilizer diagrams correspond to a stabilizer state diagram under map-state duality [22] (also known as the Choi-Jamiolkowski isomorphism) - essentially, we can attach

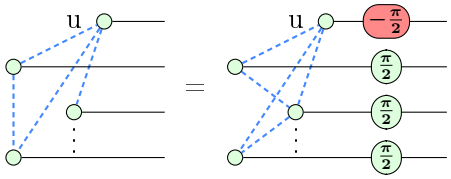
caps to each input of a stabilizer diagram to turn these inputs into outputs, turning the process into a state as seen below.



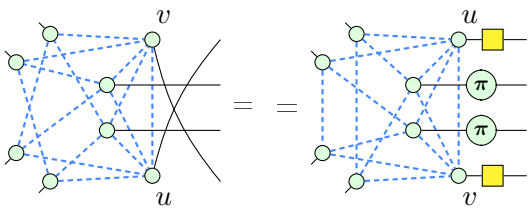
All stabilizer states are local-Clifford equivalent to some graph-state [51] and thus every stabilizer state can be rewritten into GS-LC form. The proof that each stabilizer state can be written in GS-LC form is covered in detail in [6].

The stabilizer completeness result largely depends on the following rewrite rule which will be used frequently throughout the rest of our work.

Definition 2.5.3. The following rewrite rule, known as Local Complementation, holds in the ZX-calculus [28] and is derivable in the stabilizer ZX-calculus.



We can use local complementation to change the Clifford operators applied on the outputs of spiders as well as to change the connectivity of the diagram. Given two connected vertices u and v, doing a local complementation on u, then another on v, then a third on u is known as a pivot and has some important properties that will be useful throughout this thesis. Here it is important as it applies Hadamard gates to the outputs of both u and v as seen below, allowing us to turn two connected red vertices into two green vertices.



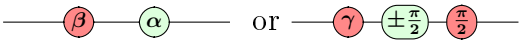
We have now defined the tools needed to show that the stabilizer ZX-calculus is complete. We shall do so by rewriting diagrams to a (pseudo) normal form and then showing that any two diagrams in this form that are equivalent can be rewritten into one another. In the following, a "vertex operator" means the local Clifford operators that are applied on the output of a green spider in a graph state.

Definition 2.5.4. A ZX diagram is in *reduced GS-LC (rGS-LC) form* if it is a graph-state diagram where the single-qubit Clifford operators are all in the set $\{$  ,  $\}$ for some $k \in \mathbb{Z}_4$ and if no two qubits with red phases in their vertex operator are connected to each other.

Using local complementation and pivoting, we are able to rewrite any diagram into rGS-LC form.

Theorem 2.5.5. *Any stabilizer state is equal to some rGS-LC diagram in the ZX-calculus.*

Proof. As previously mentioned, each stabilizer state can be rewritten into GS-LC form using stabilizer ZX-calculus rewrite rules. By 2.5.2, each vertex operator can be brought into one of the following two forms



A local complementation about a vertex v pre-multiplies the vertex operator of v with a red $\frac{-\pi}{2}$ phase spider, allowing any vertex operator to be brought into one of the forms of 2.5.4 by applying some number of local complementations. Applying a local complementation to a vertex v also toggles the connectivity of its neighbours along with applying a green $\frac{\pi}{2}$ phase spider to each neighbour - premultiplication by a green $\frac{\pi}{2}$ does not map the vertex operators in 2.5.4 to itself, but does not increase the number of red nodes in vertex operators. Therefore, the process of removing red nodes with local complementations must terminate after at most 2n steps for an n-qubit diagram, at which point all vertex operators will be in the set defined in 2.5.4.

Once all vertex operators are in the right form, any two neighbouring vertices with red

nodes in their vertex operator can have their red nodes removed using a pivot and potentially extra local complementations, leaving the diagram in rGS-LC form. □

To obtain completeness, it remains to show that any two equivalent rGS-LC form diagrams can be rewritten into the same diagram. The proof of this is long and will not be covered in extensive detail here, but we shall describe the process - for the full proof, see [10].

The proof begins with the following definition;

Definition 2.5.6. A pair of rGS-LC diagrams on the same number of qubits is called *simplified* if there are no pairs of qubits p, q such that p has a red node in its vertex operator in the first diagram but not the second, q has a red node in the second but not the first, and p and q are adjacent in at least one of the diagrams.

It is then shown that any pair of rGS-LC diagrams can be simplified. The proof concludes by proving the following theorem.

Theorem 2.5.7. *The two diagrams making up a simplified pair of rGS-LC diagrams are equivalent (correspond to the same quantum state) if and only if they are identical.*

Therefore the process of simplification of a pair of rGS-LC diagrams is sufficient to rewrite any two equivalent rGS-LC diagrams into the same diagram. We can therefore rewrite any two equivalent stabilizer states into one another using only the stabilizer ZX-calculus rules (along with other rules derivable from the standard stabilizer ZX-calculus rules).

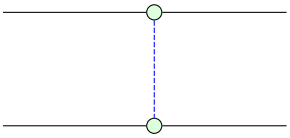
Chapter 3

Determinism in Measurement-based Quantum computing

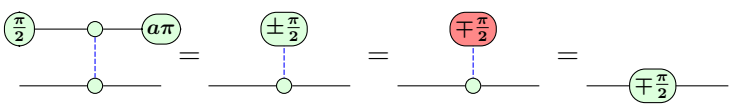
3.1 Quantum Measurements

One of the more confusing concepts in quantum computing is the concept of measurements. Consider some qubit in an arbitrary state $|\psi\rangle = a|0\rangle + b|1\rangle$. If we were to perform a Z-basis measurement on this state, we could obtain either $|0\rangle$ or $|1\rangle$ as the result, with probabilities a^2 and b^2 respectively.

Example 3.1.1. Consider the following diagram, consisting of two qubits with a CZ-gate applied between them.



If we were to prepare the top qubit in the positive eigenstate of the Pauli-Y operator, then measure it in the Pauli-Z basis, we would obtain the following.



This is then equivalent to a circuit with just one qubit where we have applied an S-gate. We can now see that measurements can also be used to implement unitary maps, which

will be an important concept in the rest of this thesis.

When using measurements to implement linear maps, we need specific measurement outcomes to occur to obtain the exact linear map that we want. As we don't know that each measurement will give the desired result, we need to know that we can correct for an error by adapting future measurements in order for things to be deterministic.

3.2 Measurement-based Quantum computation

Measurement-based Quantum computation (MBQC) is a particularly interesting model of quantum computation with no classical analogue. In MBQC, one first constructs a highly entangled resource state that can be independent of the specific computation that one wants to perform (only depending on the 'size' of the computation) by preparing qubits in the $|+\rangle$ state and applying CZ-gates to certain pairs of qubits. The computation then proceeds by performing single qubit measurements in a specified order. MBQC is a universal model for quantum computation – any computation can be performed by choosing an appropriate resource state and then performing a certain combination of measurements on said state.

MBQC restricts the allowed single-qubit measurements to three planes of the Bloch sphere: those spanned by the eigenstates of two Pauli matrices, called the XY, YZ and XZ planes. Each time a qubit u is measured in a plane $\lambda(u)$ at an angle α, one may obtain either the desired outcome, denoted $\langle +_{\lambda(u),\alpha}|$, or the undesired outcome $\langle -_{\lambda(u),\alpha}| = \langle +_{\lambda(u),\alpha+\pi}|$. The allowed measurements are described by the following for $\alpha \in [0, 2\pi]$[[24] p.292]:

$$|+_{XY,\alpha}\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle) \qquad |-_{XY,\alpha}\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - e^{i\alpha}|1\rangle)$$

$$|+_{XZ,\alpha}\rangle = \cos(\tfrac{\alpha}{2})|0\rangle + \sin(\tfrac{\alpha}{2})|1\rangle \qquad |-_{XZ,\alpha}\rangle = \sin(\tfrac{\alpha}{2})|0\rangle - \cos(\tfrac{\alpha}{2})|1\rangle$$

$$|+_{YZ,\alpha}\rangle = \cos(\tfrac{\alpha}{2})|0\rangle + i\sin(\tfrac{\alpha}{2})|1\rangle \qquad |-_{YZ,\alpha}\rangle = \sin(\tfrac{\alpha}{2})|0\rangle - i\cos(\tfrac{\alpha}{2})|1\rangle$$

Measurements where the angle is an integer multiple of $\frac{\pi}{2}$ are Pauli measurements; the corresponding measurement type is denoted by simply X, Y, or Z. The ZX-diagram

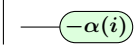
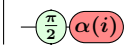
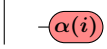
operator	$\langle +_{XY,\alpha}\vert_i$	$\langle +_{XZ,\alpha}\vert_i$	$\langle +_{YZ,\alpha}\vert_i$	$\langle +_{X,0}\vert_i$	$\langle +_{Y,0}\vert_i$	$\langle +_{Z,0}\vert_i$	$\langle +_{X,\pi}\vert_i$	$\langle +_{Y,\pi}\vert_i$	$\langle +_{Z,\pi}\vert_i$
diagram									

Table 3.1: MBQC measurement effects in Dirac notation and their corresponding ZX-diagrams.

corresponding to each (desired) measurement outcome is given in Table 3.1.

Remark 3.2.1. Throughout this thesis, we will use green spiders with an angle α to represent XY measurements - this is due to it being inconvenient to add a minus sign every time we talk about XY measurement. If one was to try implement any of the measurement based computations used throughout the paper, they should note that a green measurement with a phase of α corresponds to a measurement in the XY-plane at an angle of $-\alpha$.

Measurement-based computations are traditionally expressed as *measurement patterns*, which use a sequence of commands to describe how the resource state is constructed and how the computation proceeds [25].

Definition 3.2.2. A *measurement pattern* consists of a collection V of qubits with distinguished subsets $I, O \subseteq V$ of inputs and outputs, and a sequence of commands from;

- Preparations N_u initialising qubit $u \notin I$ to $|+\rangle$;
- Entangling operators E_{uv}, applying a CZ gate between distinct qubits u and v;
- Destructive measurements $M_u^{\lambda,\alpha}$, projecting qubit u onto either $|+_{\lambda,\alpha}\rangle$ with outcome 0 or $|-_{\lambda,\alpha}\rangle$ with outcome 1;
- Corrections $[X_u]^v$ or $[Z_u]^v$, conditionally applying an X or Z gate to qubit $u \in V$ if the outcome of measurement v was 1.

A measurement pattern is *runnable* if additionally:

- All non-input qubits are prepared exactly once;
- A non-input qubit is not acted on by any other command before its preparation;

- All non-output qubits are measured exactly once;
- A non-output qubit is not acted on by any other command after its measurement
- No correction depends on an outcome not yet measured.

As the resource states are graph states, a graphical representation of MBQC protocols can be more intuitive; we shall therefore focus on the representation of MBQC with ZX-diagrams.

Definition 3.2.3. [9, Definitions 2.18, 2.23] A ZX-diagram is in *MBQC-form* if it consists of a graph state diagram in which each vertex of the graph may furthermore be connected to an input (in addition to its output), and a measurement effect instead of its output. A ZX-diagram is in *MBQC+LC-form* if it is in MBQC-form up to single qubit Clifford operators on the input and output wires.

Much of the literature on MBQC focuses on labelled open graphs which are a graph theoretical description of measurement patterns that are equivalent to MBQC-form ZX diagrams. We shall introduce labelled open graphs briefly below and may use MBQC-form diagram and labelled open graph interchangeably going forward.

Definition 3.2.4. A *labelled open graph* is a tuple $\Gamma = (G, I, O, \lambda)$, where $G = (V, E)$ is a simple undirected graph, $I \subseteq V$ is a set of input vertices, $O \subseteq V$ is a set of output vertices, and $\lambda : V \setminus O \to \{X, Y, Z, XY, XZ, YZ\}$ assigns a measurement plane or Pauli measurement to each non-output vertex.

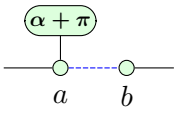
3.3 Corrections in MBQC

3.3.1 Causal flow

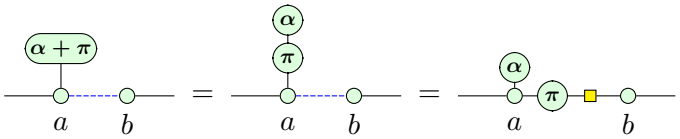
When performing a measurement-based quantum computation, the outcome of the computation depends on the outcomes of each individual measurement. To ensure that the computation gives a deterministic outcome (i.e. gives the same result every time we perform said computation), we need to be able to apply corrections each time we obtain

the undesired outcome from a measurement. When measuring a qubit at an angle α, the undesired outcome is equivalent to measuring said qubit at an angle of $\alpha + \pi$. The following examples illustrate how we can correct for an undesired measurement using the ZX-calculus' rewrite rules - we shall begin with diagrams only containing XY-measured qubits and generalise this to allow all planes along with Pauli measurements in the following sections.

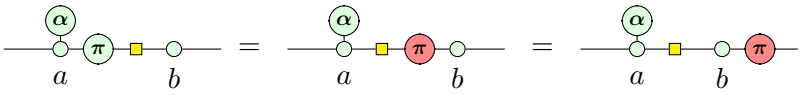
Example 3.3.1. Consider the following MBQC-form diagram.



Here, we consider a to be measured at an angle α, but we have obtained the undesired outcome of said measurement. Using spider fusion and unfusion, we can move the π error as follows.



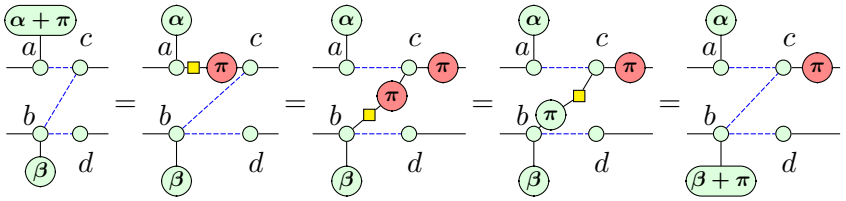
We can then apply the colour change rule to pass the green π through the Hadamard gate, followed by applying π-copy to obtain a red π on the output wire.



Clifford gates on output wires can be corrected for by re-interpreting the measurement outcome, thus we have found a way to correct the error obtained from measuring a. Here we say that we can correct a on output b.

This example illustrates that it is possible to correct for undesired measurement outcomes but is an extremely basic example of this. We shall now consider a slightly more complex case.

Example 3.3.2. In the following diagram, we shall consider measuring a first and obtaining the undesired measurement outcome. We can correct the error on a as follows.



It now looks like b has an error on it, however if we measure b after a then we can just change the angle we measure b at from β to $\beta + \pi$. We thus see that we need a (partial) order on the vertices to be able to perform corrections. If we then measure b and obtain the undesired result, we can correct for this on output vertex d.

Generalising the result from the previous example, if we correct an error by applying a red π to a vertex v, we must apply green πs to each of v's neighbours due to the π-copy rule.

We can now define a condition on diagrams which ensures they can be corrected in this way. In the following, f assigns a corrector to each measured vertex.

Definition 3.3.3 ([15], Definition rephrased)**.** An MBQC-form diagram $\Gamma = (G, I, O, \lambda)$ has causal flow $(f, \prec)$ if, for all measured vertices $v \in V \setminus O$, we have the following.

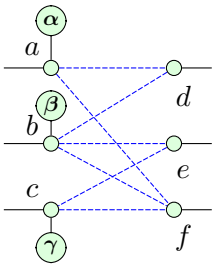
1. v and $f(v)$ are neighbours..
2. $v \prec f(v)$
3. If $w \in N_G(f(v))$ and $w \neq v$ then $v \prec w$.

Causal flow is a sufficient condition for ensuring strong, stepwise and uniform determinism of MBQC-form diagrams with only XY-measured vertices. This means that all branches of the computation should implement the same linear operator up to a phase, any interval of the computation (along with the associated corrections) should be deterministic on its own, and the computation should be deterministic for all choices of measurement angles. We shall say deterministic to mean strong, stepwise and uniformly deterministic from henceforth. In the next subsection we shall develop a condition that is both sufficient and necessary for this.

3.3.2 gflow

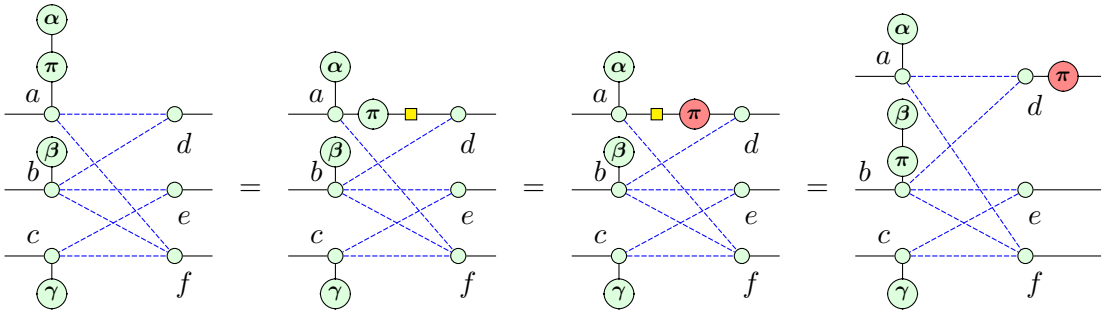
For causal flow, we only allowed ourselves to 'push forward the π-error' once. Generalised flow (or gflow) is essentially what we obtain when we allow ourselves to push forward errors as many times as we like.

Example 3.3.4. Consider the following diagram [15].



This diagram can never have a causal flow as each output is connected to two or more of the measured vertices - no matter which order we measure the qubits in, an error on the last measurement would require us to apply a green π correction to an already measured vertex which is not allowed.

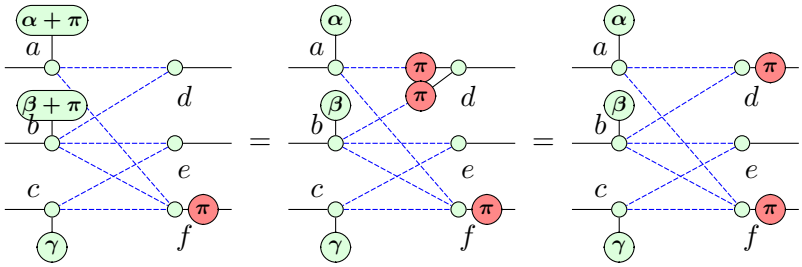
For example, suppose we tried to correct a with vertex d, b with vertex e and c with vertex f. Correcting a with d would put a green π on b as seen in the diagram below, thus a must be measured before b.



Similarly, correcting b with e would put a green π on c, thus b must be measured before c. Then, correcting c with f would put a green π on both a and b, meaning that c must be measured before both a and b, giving a contradiction.

Nonetheless, we are still able to correct for errors in this diagram. By pushing the aforementioned green πs that would appear on a and b (when correcting for an error on c) onto output d, as seen in the following diagram, we find a way to correct for errors on c using

only future vertices.

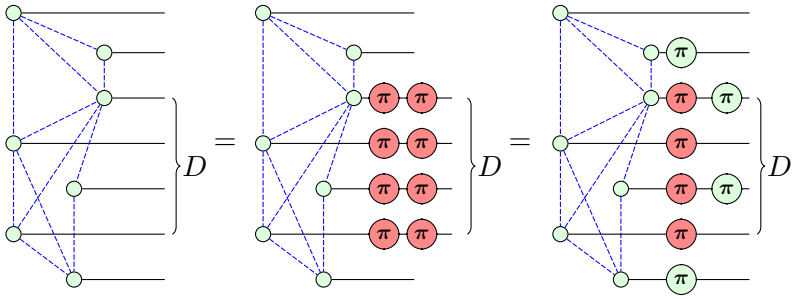


We call $\{d, f\}$ the 'correction set' of c here.

Pushing errors around the diagram as done previously can get complicated - often it is easier to use another property of graph states to find correction sets. Given some subset D of the vertices of a graph G, we denote the set of vertices which have an odd number of neighbours in D by $\mathrm{Odd}_G(D)$. Using this, we can define the 'fixed-point property' which allows us to perform more general corrections for undesired measurement outcomes.

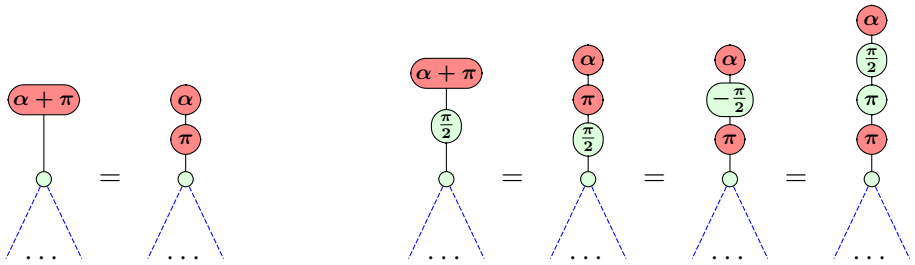
Proposition 3.3.5. *Given some graph state G and any subset D of the vertices, applying $X_D Z_{\mathrm{Odd}_G(D)}$ leaves the state invariant.*

We shall demonstrate this fixed-point property with a graphical example.



Now, suppose we obtain a green π error on the qubit second from the top. We then know that if we apply a red π to the third, forth, fifth and sixth qubit along with a green π to the third, fifth and seventh qubit, this is the same as 'undoing' the green π error on the second qubit. We should therefore think of the elements of D as the correction set (those vertices that obtain Pauli-X (red π) corrections), while the elements of $Odd_G(D)$ should be thought of as the vertices which obtain Pauli-Z (green π) corrections.

Up until this point, we have only considered XY-plane measurements. The fixed point property also tells us how to correct for errors from YZ and XZ measurements.



Obtaining the undesired result from a YZ-measurement is the same as obtaining a red π error, and thus we require that YZ-measured vertices appear in their own correction sets and not their odd neighbourhood so that they only obtain a red π when applying the fixed point property. Similarly, obtaining the undesired outcome from a XZ-measurement is the same as obtaining both a red π and a green π error, thus we require XZ-measurements to appear in both their correction set and its odd neighbourhood so that they obtain both a red π and a green π when applying the fixed point property.

We are now able to define extended gflow.

Definition 3.3.6 ([15], Definition 3 rephrased)**.** A labelled open graph (G, I, O, λ) has (extended) generalized flow (or gflow) if there exists a map $g : V \setminus O \to \mathcal{P}(V \setminus I)$ and a partial order $\prec$ over V such that for all $v \in V \setminus O$,

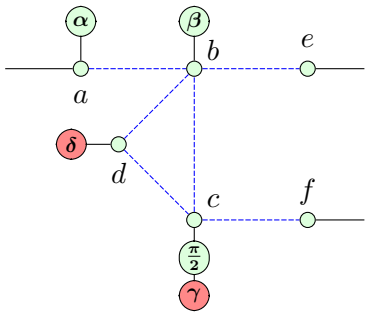
1. If $w \in g(v)$ and $v \neq w$ then $v \prec w$,
2. If $w \in \mathsf{Odd}_G(g(v))$ and $v \neq w$ then $v \prec w$,
3. If $\lambda(v) = XY$ then $v \notin g(v)$ and $v \in \mathsf{Odd}_G(g(v))$
4. If $\lambda(v) = XZ$ then $v \in g(v)$ and $v \in \mathsf{Odd}_G(g(v))$
5. If $\lambda(v) = YZ$ then $v \in g(v)$ and $v \notin \mathsf{Odd}_G(g(v))$

The partial order restricts the time order in which the qubits need to be measured. The set $g(u)$ denotes qubits that are modified by Pauli-X to compensate for an undesired measurement outcome on u and $\mathrm{Odd}_G(g(u))$ denotes the set of vertices that are modified by Pauli-Z.

In simple terms, an MBQC-form diagram has extended gflow if each qubit comes before every other qubit that obtains a red or green π when correcting for an error on said

qubit, along with each qubit receiving the right correction depending on which plane it is measured in.

Example 3.3.7. Consider the following diagram which has measurements in all three planes [9].



This has an extended gflow with partial order $a \prec b \prec c \prec d \prec e, f$ and correction set function defined as:

$$g(a) = \{b\}$$

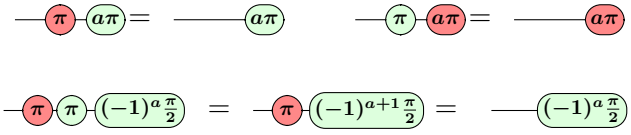
$$g(b) = \{c\}$$

$$g(c) = \{c, d\}$$

$$g(d) = \{d, e, f\}$$

3.3.3 Pauli flow

In the case of gflow, we require every measurement to be arbitrary planar measurements. However, some MBQC-form diagrams are only deterministic if we force some of the measurements to be at specific angles (specifically Pauli measurements with angles restricted to integer multiples of $\frac{\pi}{2}$). This is due to the following equations holding for Pauli measurements.



In words, X-corrections do nothing to Pauli-X measured vertices, Z-corrections do nothing to Pauli-Z measured vertices, and an X and Z correction together does nothing to

Pauli-Y measured vertices. This means that already measured Pauli-X vertices can appear in the correction set of other vertices as this does not change the interpretation of the diagram (similarly, Pauli-Z vertices can appear in odd neighbourhoods and Pauli-Y vertices can appear in correction sets if they also appear in the odd neighbourhood of said correction set).

Combining this with the definition of gflow gives us Pauli flow, the flow condition we will be focussing on for the majority of this thesis.

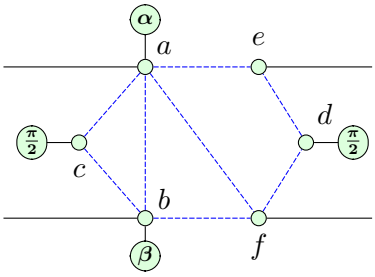
Definition 3.3.8 ([15, Definition 5])**.** A labelled open graph (G, I, O, λ) has Pauli flow if there exists a map $p : V \setminus O \to \mathcal{P}(V \setminus I)$ and a partial order $\prec$ over V such that for all $u \in V \setminus O$,

1. if $v \in p(u)$, $v \neq u$ and $\lambda(v) \notin \{X, Y\}$, then $u \prec v$.
2. if $v \in \mathrm{Odd}_G(p(u))$, $v \neq u$ and $\lambda(v) \notin \{Y, Z\}$, then $u \prec v$.
3. if $\neg(u \prec v)$ and $\lambda(v) = Y$, then $v \notin p(u) \bigtriangleup \mathrm{Odd}_G(p(u))$.
4. if $\lambda(u) = XY$, then $u \notin p(u)$ and $u \in \mathrm{Odd}_G(p(u))$.
5. if $\lambda(u) = XZ$, then $u \in p(u)$ and $u \in \mathrm{Odd}_G(p(u))$.
6. if $\lambda(u) = YZ$, then $u \in p(u)$ and $u \notin \mathrm{Odd}_G(p(u))$.
7. if $\lambda(u) = X$, then $u \in \mathrm{Odd}_G(p(u))$.
8. if $\lambda(u) = Z$, then $u \in p(u)$.
9. if $\lambda(u) = Y$ then either $u \in p(u)$ and $u \notin \mathrm{Odd}_G(p(u))$ or $u \notin p(u)$ and $u \in \mathrm{Odd}_G(p(u))$.

Pauli flow is a sufficient condition for measurement patterns corresponding to MBQC-form diagrams to be deterministic [15, p. 5]. Recently, a new condition known as Shadow Pauli flow has been found that is both sufficient and necessary for patterns with both arbitrary planar measurements and Pauli measurements to be deterministic [49]. Every diagram that has a Shadow Pauli flow has a Pauli flow (when all of the Pauli measurements are

done first), thus our work on diagrams with Pauli flow largely also applies to the case of Shadow Pauli flow.

Example 3.3.9. Consider the following diagram Pauli-Y measurements on qubits c and d and XY-planar measurements on qubits a and b.



This has a Pauli flow with $p(d) = \{d\}, p(c) = \{c, d, f\}, p(b) = \{e, f\} and p(a) = \{d, e\}$ and partial order $a, b, c, d \prec e, f$. We can have $d \in p(c)$ and $d \in p(a)$ with $a, c \not\prec d$ as $d \in \mathsf{Odd}_G(p(c))$ and $d \in \mathsf{Odd}_G(p(a))$ and thus an error on a or c leads to both an X and a Z gate appearing on d - then as d is Y-measured, $X \circ Z$ acts as the identity on d as in the beginning of this subsection.

3.3.4 Focused Flows

Given some diagram with gflow or Pauli flow, there may be several different correction set functions and partial orders that satisfy the flow conditions. We shall introduce some special types of flow that have important properties - for example, it can often be useful to 'maximally delay' the measurements, allowing us to keep as many qubits around to correct for errors on for as long as possible. We shall formalise this below - as every gflow is also a Pauli flow, we shall just refer to Pauli flows, though all of this applied to gflow also.

Definition 3.3.10 (Definition A.1 [58])**.** For a given labelled open graph (G, I, O, λ) and a given Pauli flow $(p, \prec)$ of (G, I, O, λ), let

$$V_k^{\prec} = \begin{cases} \max_{\prec}(V) & \text{if } k = 0 \\ \max_{\prec}(V \setminus \cup_{i<k} V_i) & \text{if } k \geq 0 \end{cases}$$

where $\max_{\prec}(X) := \{u \in X \text{ s.t. } \forall v \in X, \neg(u \prec v)\}$ is the set of maximal elements of X.

Definition 3.3.11 (Definition A.2 [58])**.** For a given labelled open graph (G, I, O, λ) and two given Pauli flows $(p, \prec)$ and $(p', \prec')$ of (G, I, O, λ), $(p, \prec)$ is more delayed than $(p', \prec')$ if for all k,

$$\left|\bigcup_{i=0}^{k} V_i^{\prec}\right| \geq \left|\bigcup_{i=0}^{k} V_i^{\prec'}\right|$$

and there exists a k where this inequality is strict. A Pauli flow $(p, \prec)$ is maximally delayed if there exists no Pauli flow on the same open graph that is more delayed.

Maximally delayed gflows are particularly useful due to the following theorem.

Theorem 3.3.12 ([58], Theorem 4.5)**.** *There exists an algorithm that decides whether a given labelled open graph has a Pauli flow, and that outputs such a Pauli flow if it exists. Moreover, this output is maximally delayed, and the algorithm completes deterministically in time that grows polynomially with the number of vertices in the graph.*

Another interesting type of flow is known as a focused Pauli flow - with only XY, X and Y measurements (those corresponding to green spiders), this corresponds to pushing forward each green π correction as much as possible. Generalised to all possible measurements, we get the following.

Definition 3.3.13 ([58], Definition 4.3 reformulated)**.** A Pauli flow $(p, \prec)$ on a labelled open graph $\Gamma = (G, I, O, \lambda)$ is focused if the following is satisfied for all $v \in V \setminus O$;

1. $w \in p(v) \setminus \{v\}$ implies $w \in O$ or $\lambda(w) \in \{XY, X, Y\}$
2. $w \in \mathsf{Odd}_G(p(v)) \setminus \{v\}$ implies $w \in O$ or $\lambda(w) \in \{XZ, YZ, Y, Z\}$
3. $w \in p(v) \bigtriangleup \mathsf{Odd}_G(p(v))$ implies $w \in O$ or $\lambda(w) \neq Y$

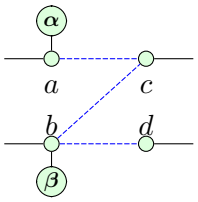
Lemma 3.3.14 ([58], Lemma 4.6)**.** *If a labelled open graph has a Pauli flow, then it has a maximally delayed, focused Pauli flow.*

This allows us to always use a maximally delayed, focused Pauli flow in proofs about

Pauli flows if the properties of being maximally delayed and/or focused are of use in said proof. One of said useful properties is the following.

Proposition 3.3.15 ([48], also [9] Corollary 2.47)**.** *Suppose we have a labelled open graph* $\Gamma = (G, I, O, \lambda)$ *with* $\lambda(v) = XY$ *for all* $v \in V \setminus O$, $|I| = |O|$ *and suppose* Γ *has a focused gflow* $(g, \prec)$*. Then this gflow can be reversed in a very strict sense - let* $\Gamma' = (G, O, I, \lambda')$ *be the reversed pattern with the roles of inputs and outputs swapped and* λ' *mapping all non-outputs to* XY*. Then there exists a focused gflow* $(g_{rev}, \prec_{rev})$ *for* Γ' *where* $\prec_{rev}$ *is the reverse of* $\prec$ *and* $u \in g_{rev}(v)$ *if and only if* $v \in g(a)$*.*

Example 3.3.16. We shall now provide an example of a focused and reversed flow. Consider the following MBQC-form diagram;



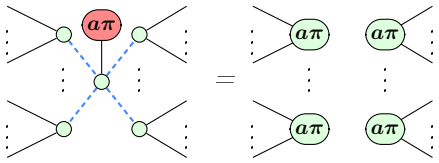
This has an obvious Pauli flow with $p(a) = \{c\}$, $p(b) = \{d\}$ and $a \prec b \prec c, d$, however this Pauli flow is not focused - we have $\text{Odd}(p(a)) = \{b\}$ which is not allowed by the focussing conditions. If we instead take $p(a) = \{c, d\}$ (keeping $p(b)$ the same), we get a focused Pauli flow with $a, b \prec c, d$ - this is then reversible, with the reversed flow having $p_{rev}(c) = \{a, b\}$, $p_{rev}(d) = \{b\}$ and partial order $c, d \prec_{rev} a, b$.

3.4 Existing flow-preserving rewrite rules

The basic ZX-calculus rewrite rules in Figure 2.3.2 do not generally preserve even the MBQC-form structure of a ZX-calculus diagram. Yet there are some more complex derived rewrite rules that are known to preserve both the MBQC-form structure and the existence of a flow. These rules were previously considered in the context of gflow [27] and extended gflow [9]; the Pauli-flow preservation proofs are due to [58]. The simplest of these rules is Z-deletion:

Lemma 3.4.1 ([58, Lemma D.6])**.** *Deleting a* Z*-measured vertex preserves the existence*

of Pauli flow.



Other rewrite rules are based around quantum generalisations of two graph-theoretic operations.

Definition 3.4.2. Let $G = (V, E)$ be a graph and $u \in V$. The *local complementation of* G *about* u is the operation which maps G to $G \star u := (V, E \bigtriangleup \{(b, c) | (b, u), (c, u) \in E \text{ and } b \neq c\})$, where $\bigtriangleup$ is the symmetric difference operator given by $A \bigtriangleup B = (A \cup B) \setminus (A \cap B)$. The *pivot of* G *about the edge* (u, v) is the operation mapping G to the graph $G \wedge uv := G \star u \star v \star u$.

Local complementation keeps the vertices of the graph the same but toggles some edges: for each pair of neighbours of u, i.e. $v, v' \in N_G(u)$, there is an edge connecting v and v' in $G \star u$ if and only if there is no edge connecting v and v' in G. Pivoting is a series of three local complementations, but has some special properties which make it worth distinguishing. It interchanges the vertices u and v and complements (or 'toggles') the connectivity between the following three subsets of vertices [13, Section 8]:

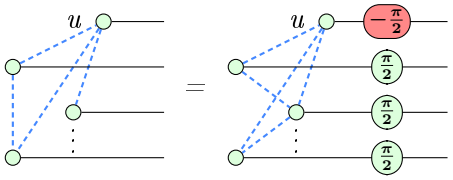
- $N_G(u) \setminus (\{v\} \cup N_G(v))$, the neighbours of u that are neither neighbours of v nor v itself.
- $N_G(v) \setminus (\{u\} \cup N_G(u))$, the neighbours of v that are neither neighbours of u nor u itself.
- $N_G(u) \cap N_G(v)$, the common neighbours of u and v.

From the above characterisation we see that pivoting is symmetric, i.e. $G \wedge uv = G \wedge vu$.

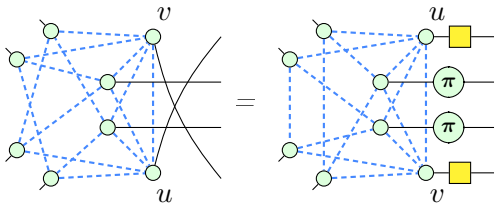
Both local complementation and pivoting give rise to operations on MBQC-form diagrams which preserve the MBQC form as well as the existence of Pauli flow (after some simple merging of single-qubit Cliffords into measurement effects, cf. [9, Section 4.2]). We

illustrate the operations with examples as they are difficult to express in ZX-calculus in generality.

Lemma 3.4.3 ([58, Lemma D.12]). *A local complementation about a vertex u preserves the existence of Pauli flow.*



Lemma 3.4.4 ([58, Lemma D.21]). *A pivot about an edge (u, v) preserves the existence of Pauli flow.*



Observation 3.4.5. *Lemmas 3.4.3 and 3.4.4 provide their own inverses since four successive local complementations about the same vertex, or two successive pivots about the same edge, leave the diagram invariant. Two successive local complementations correspond to the π-copy rule.*

Local complementation and pivoting have a wide range of uses, most importantly in circuit extraction and optimization. One particular use is rewriting MBQC-form diagrams into particular pseudo-normal forms (which we can then perform optimization procedures on). The following form only has XY-measured vertices along with 'phase gadgets', which are tools used in the ZX-calculus that correspond to YZ-measured vertices that are only connected to XY-measured vertices in the setting of MBQC.

Definition 3.4.6. An MBQC-form diagram is in *phase-gadget form* if

- there does not exist any $v \in V \setminus O$ such that $\lambda(v) = XZ$, and
- there does not exist any pair of neighbours $v, w \in V \setminus O$ such that $\lambda(v) = \lambda(w) = YZ$.

Given some MBQC-form diagram, we can rewrite it into phase-gadget form by performing local complementation on each XZ-measured vertex to change its plane, then pivoting

along each edge between two YZ-measured vertices to turn them into two XY-measured vertices.

3.5 Circuit extraction

Previously we found that it is easy to construct a ZX-diagram from a circuit and it is easy to turn any ZX-diagram into an MBQC-form diagram. It is then natural to consider the following: given some arbitrary ZX-diagram, are we able to find a circuit that implements the same linear operator? Clearly we would need the ZX-diagram to have the same number of inputs and outputs as well as being unitary for it to be equivalent to a circuit. However, even knowing that it is unitary is not enough – this problem has been shown to be #P-hard[65].

Flows are particularly interesting as polynomial-time circuit extraction procedures exist for each type of (causal, generalised or Pauli) flow [[9],[58]].

The gflow circuit extraction algorithm begins by rewriting the diagram into phase-gadget form, then starting from the right hand side of the diagram it makes sequential changes to the diagram to make it look more like a circuit. We shall introduce the Pauli flow circuit extraction algorithm of [58] in more depth here.

3.5.1 Phase Polynomials, Phase Gadgets and Pauli gadgets

Phase polynomials are the class of unitary maps generated by circuits containing just $CNOT$ gates and Z-rotations and have found many uses across circuit optimization and verification [2, 3, 5, 37, 64]. A unitary constructed in this way acts on computational basis states as follows:

$$|\overrightarrow{x}\rangle = |x_1 x_2 \dots x_n\rangle \to e^{if(x_1,x_2,\dots,x_n)} |A\overrightarrow{x}\rangle$$

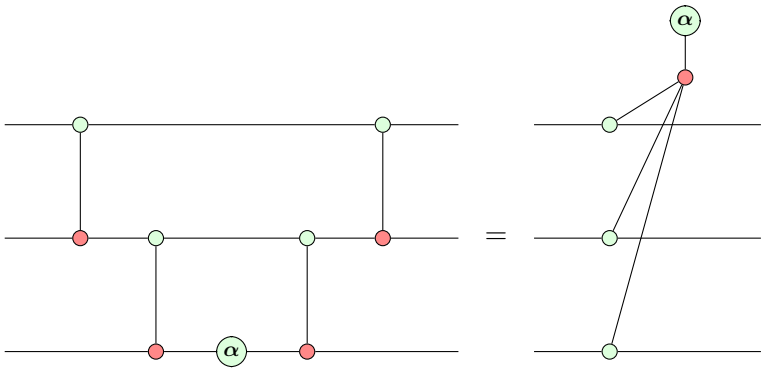
where $A : \mathbb{B}^n \to \mathbb{B}^n$ is a linear boolean function (constructed from just $CNOT$ gates) and $f : \mathbb{B}^n \to \mathbb{R}$. Note that exponentials of operators are defined similarly to regular

exponentials; we have

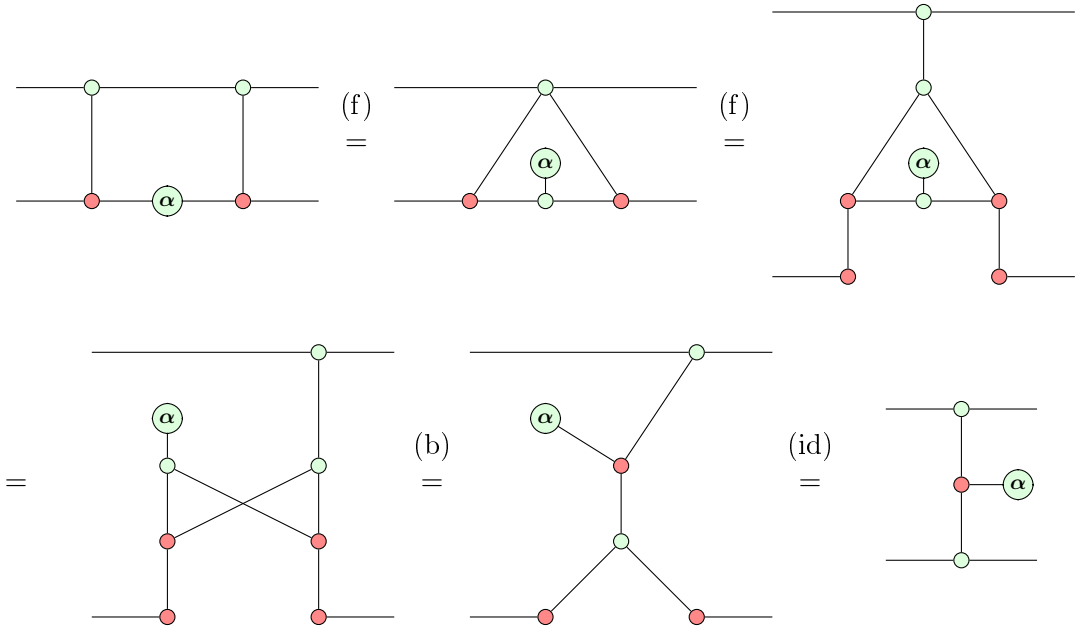
$$e^F = \sum_{n=0}^{\infty} F^n$$

where F^0 is considered to be the identity matrix with the same dimensions as F.

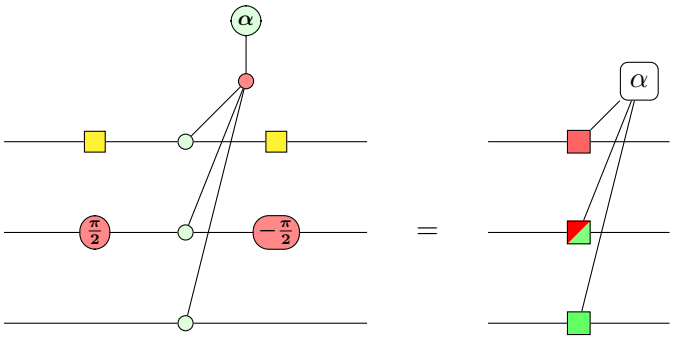
If f contains only a single XOR term, then we call the unitary implementing this map a Phase gadget. A phase gadget adds a phase to the state if the number of qubits in the $|1\rangle$ state is odd, demonstrated by the XOR term in the exponent. Phase gadgets have a natural representation as a ZX-diagram, an example of a which is given below, corresponding to the implementation of $e^{-i\frac{\alpha}{2}Z_1Z_2Z_3}$.



We can show that the form on the left-hand side is equivalent to the right hand side easily when there are just two qubits and extend to larger numbers of qubits by induction [64].



We can similarly define gadgets where the rotations are in any Pauli basis, not just Z. These are known as Pauli gadgets and an example (the implementation of $e^{-i\frac{\alpha}{2}X_1Y_2Z_3}$) is shown below - a red square means the gadget is acting on the qubit in the X-basis, a green square corresponds to the Z-basis and a half-red, half-green square corresponds to the Y-basis. This notation for Pauli gadgets is used to remove the need for the changes of basis that appear on the left-hand side of the diagram below.



Pauli exponentials (unitaries of the form $e^{i\frac{\alpha}{2}P}$ for some $P/in\{I, X, , Z\}^{\otimes n}$) benefit from elegant relations with stabilizers.

Lemma 3.5.1 (Product Rotation Lemma, [58] Lemma 3.1)**.** *Let A and B be commuting operators such that $B\mathcal{C} = \mathcal{C}$ for some linear map $\mathcal{C}$. Then $e^{i\alpha A}\mathcal{C} = e^{i\alpha AB}\mathcal{C}$.*

Lemma 3.5.2 (Reorder rules, [58] Lemma 3.2)**.** *For any Pauli Strings $A, B \in \{I, X, Y, Z\}^{\otimes n}$ and angles α, β, if A and B commute, then*

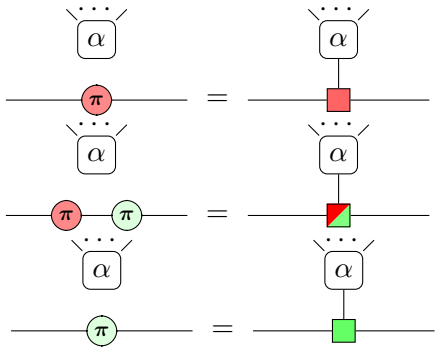
$$e^{i\alpha A}B = Be^{i\alpha A}$$

$$e^{i\alpha A}e^{i\beta B} = e^{i\beta B}e^{i\alpha A}$$

Lemma 3.5.3 (Reorder rules, [58] Lemma 3.2)**.** *For any Pauli Strings $A, B \in \{I, X, Y, Z\}^{\otimes n}$ and angle β, if A and B anticommute, then*

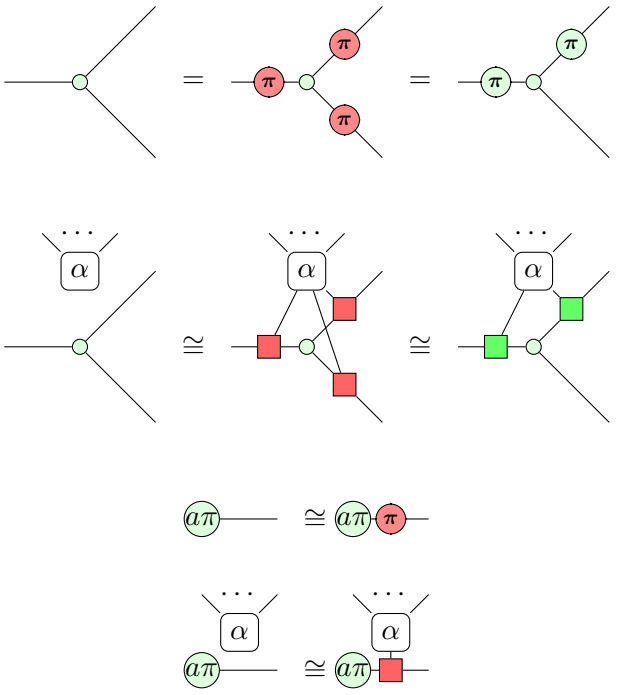
$$e^{i\frac{\pi}{4}A}B = (iAB)e^{i\frac{\pi}{4}A}$$

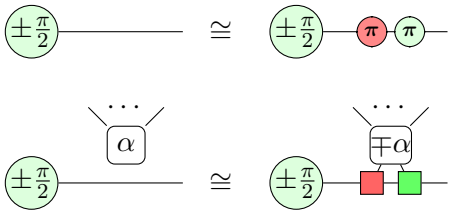
$$e^{i\frac{\pi}{4}A}e^{i\beta B} = e^{i\beta(iAB)}e^{i\frac{\pi}{4}A}$$



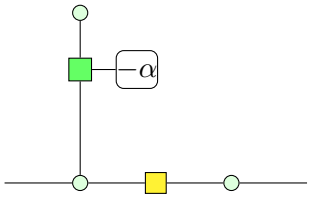
One key property of Pauli gadgets (which shall be referred to as "Pauli Gadget Teleportation") is that anywhere one sees a Pauli-X or Pauli-Z in a stabilizer, one can instead connect a Pauli gadget where this Pauli rotation was with the corresponding coloured square. Explicitly, the following holds (where each of the Z and X rotations come from a state's stabilizer);

To prove this holds, one can take each generator from the stabilizer fragment of the ZX-calculus, find a generating set of stabilizers and show that, in each case, one can add/remove legs of a Pauli gadget exactly where π phases show up [59].

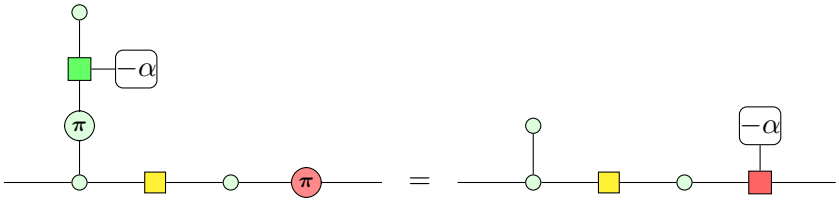




An example of this in use, which will be important in the following section on circuit extraction, can be seen below. Consider the following diagram;



Applying a Pauli-Z to the input and a Pauli-X to the output is a stabilizer of this diagram (as one can colour change through the Hadamard gate and cancel out the other). This can be used to rewrite the diagram using the aforementioned Pauli gadget rules as follows (also noting that two green legs of a Pauli gadget meeting cancel each other out by spider fusion and the Hopf law);



3.5.2 Circuit extraction for ZX-diagrams with Pauli flow

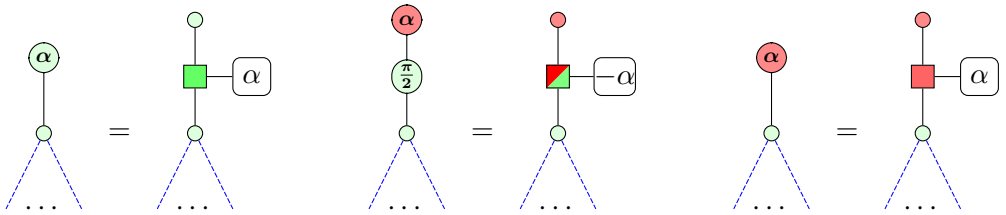
To motivate the following, note that each planar basis measurement can be constructed as a basic rotation of some Pauli basis [58].

$$\langle \pm_{XY,\alpha}| \cong \langle \pm_{X,0}| \, e^{i\frac{\alpha}{2}Z}$$

$$\langle \pm_{XZ,\alpha}| \cong \langle \pm_{Z,0}| \, e^{i\frac{\alpha}{2}Y}$$

$$\langle \pm_{YZ,\alpha}| \cong \langle \pm_{Z,0}| \, e^{-i\frac{\alpha}{2}X}$$

We are thus able to write the different planar measurements in terms of Pauli gadgets as follows.

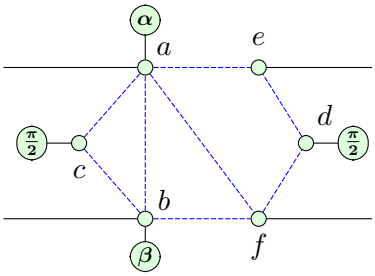


In the Pauli flow circuit extraction algorithm, one represents all planar measurements by Pauli gadgets and then use the rule which lets us add or remove legs of a Pauli gadget wherever we see Z or X rotations in stabilizers to teleport the Pauli gadgets onto outputs. This is done by considering an error occuring when measuring the qubit one wants to extract, as well as the corrections on vertices it is corrected on - this adds several Pauli-X and Pauli-Z gates which one can use to attach or remove legs from the Pauli gadgets.

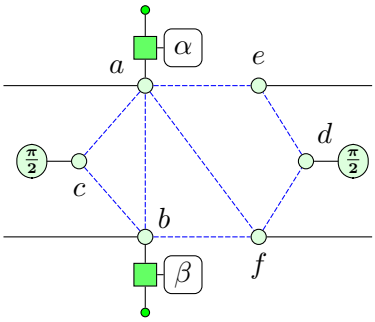
Theorem 3.5.4 ([58], Theorem 4.7)**.** *Let (Γ, α) describe a measurement pattern where Γ has a Pauli flow. Then there is an algorithm that identifies an equivalent circuit requiring no ancillae which completes in time polynomial in the number of vertices in Γ.*

Proof. For details on the proof of this, see [58]. To summarise how this works, the Pauli flow gives us the order to extract vertices and the specific stabilizer needed to use the rules of 3.5.1 to teleport all planar measurements onto the outputs, leaving us with a stabilizer diagram (to the left of the outputs) which can always simply be converted into a circuit. We shall give more details on how this circuit extraction algorithm works with an example. □

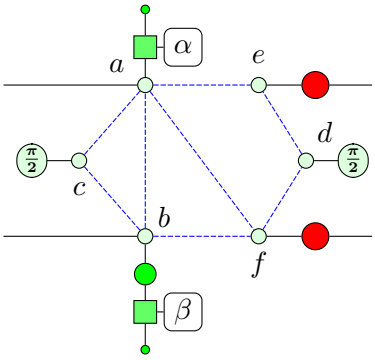
Example 3.5.5. Consider the diagram from Example 3.3.9.



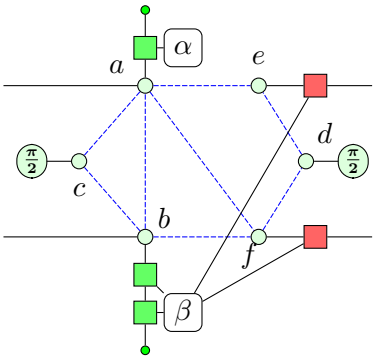
This had a Pauli flow $(p, \prec)$ with $p(d) = \{d\}, p(c) = \{c, d, f\}, p(b) = \{e, f\}$ and $p(a) = \{d, e\}$ and partial order $a, b, c, d \prec e, f$. Moreover, this flow is focused - for all vertices $v \in V$, we have $w \in p(v) \setminus \{v\}$ implies $w \in O$ or $\lambda(w) = Y$ and $w \in \mathsf{Odd}_G(p(v))$. Similarly, for all $v \in V$, $w \in \mathsf{Odd}_G(p(v))$ implies $w \in O$ or $\lambda(w) = Y$ and $w \in p(v)$, satisfying the focussing conditions. One can replace the planar measurements with Pauli gadgets to obtain the following diagram.



Due to the partial order given by the Pauli flow, either a or b can be extracted first - we will start with b. Consider an error occuring upon the measurement of b. As $p(b) = \{e, f\}$ and $\mathsf{Odd}_G(p(b)) = \{b\}$, we can correct the error on b using the following stabilizer:

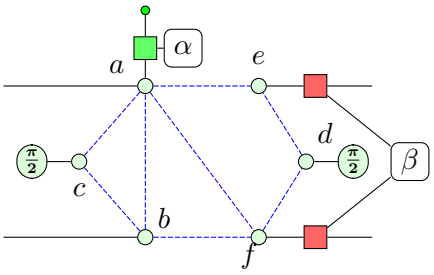


Then, using the "gadget teleportation" rules of 3.5.1 one can rewrite all of the π phase spiders introduced by the stabilizer into Pauli gadget legs, obtaining the following.

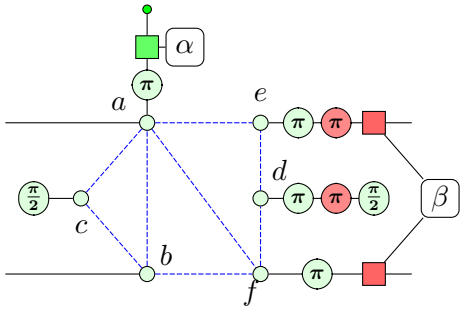


Then, by the definition of Pauli gadgets and using the Hopf rule, two connected Pauli

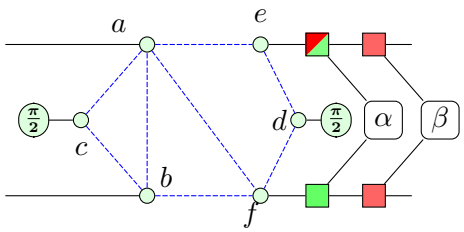
gadget legs of the same colour causes both of the legs to be deleted, allowing us to extract b's measurement onto the outputs.



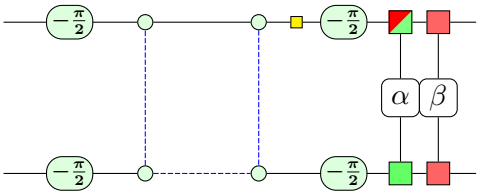
One can then begin to extract a using the same method - we had $p(a) = \{d, e\}$ and $\mathsf{Odd}_G(p(a)) = \{a, d, e, f\}$, therefore the following stabilizer corrects for an error on a;



As d is Y-measured, by 3.3.3 we know that this acts as the identity on d and therefore we can remove the red and green πs from the measurement of d. Replacing the remaining π phase spiders with Pauli gadget legs and deleting the doubled leg on a as we did when extracting b, we obtain the following:



Now everything to the left of the Pauli gadgets is just a stabilizer diagram which can always be simply brought into the form of a circuit [58]. In the particular case here, we can perform local-complementation about vertices c and d and then Z-delete these vertices - after finally unfusing a spider from a connected to vertex f we obtain the following diagram that is equivalent to a circuit after decomposing the Pauli gadgets into $CNOT$ and phase gates.


$-\frac{\pi}{2}$
$-\frac{\pi}{2}$
$-\frac{\pi}{2}$
$-\frac{\pi}{2}$
α
β

Chapter 4

Rewrite rules which preserve the existence of Pauli flow

The content in this chapter can be found in the papers [46] and [47].

Much research has been done previously on ZX-calculus rewrite rules which preserve flow conditions [[9], [27] etc.], but aside from the work of Simmons in [58] these have all focused on preserving gflow. As Pauli flow is a generalisation of gflow, it is natural to question which rewrite rules preserve the existence of Pauli flow; this forms the core question we address in our research, along with what applications said rewrite rules may have in different areas of quantum computing.

Furthermore, while there had been previous research on rewrite rules which reduce the number of spiders in a ZX-diagram while preserving flow conditions, rewrite rules which increase the number of spiders had not been studied beyond introducing new degree-2 vertices along input or output wires (e.g. [9, Lemma 4.1]). Here we discuss several rewrite rules which preserve the existence of Pauli flow while increasing the number of qubits.

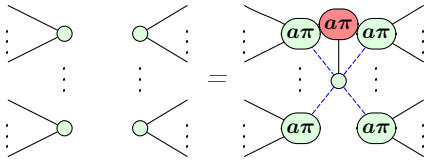
The regularly used ZX-calculus rewrite rules generally do not preserve the MBQC-form structure of diagrams. We thus must look for different rules which do preserve this

structure, as well as the existence of Pauli flow, for us to be able to effectively optimize and reason about measurement-based computations in the ZX-calculus. Some of these new rules will come from directly converting the standard ZX-calculus rewrite rules into MBQC-form-preserving versions of themselves while others will be completely new.

4.1 Inserting new Z-measured qubits

By the π-copy rule and strong complementarity, measuring a qubit in the Z-basis corresponds to deleting said qubit. It therefore stands to reason that we should be able to arbitrarily insert extra Z-measured qubits into a ZX-diagram without changing whether the computation is deterministic or not. Here we prove that inserting Z-measured qubits into a MBQC+LC form diagram preserves the existence of Pauli flow [46].

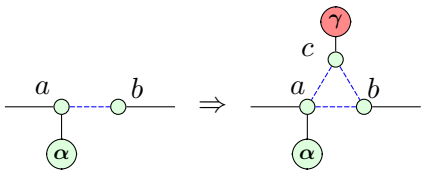
Proposition 4.1.1. *Let $G = (V, E, I, O, \lambda)$ be a labelled open graph with Pauli flow and let $W \subseteq V$ be some arbitrary subset of the vertices. Then $G' = (V', E', I, O, \lambda')$ has a Pauli flow, where $V' = V \cup \{x\}$, $E' = E \cup \{(x, w) \mid w \in W\}$ with $\lambda'(v) = \lambda(v)$ if $v \neq x$ and $\lambda'(x) = Z$.*



Proof. Let $(p, \prec)$ be a Pauli flow for G and define $p' : V' \setminus O \to \mathcal{P}(V' \setminus I)$ by $p'(v) := p(v)$ if $v \neq x$ and $p'(x) := \{x\}$. For vertices from the original graph, measurement planes and correction sets remain the same while the only change to odd neighbourhoods is that x may be added. Thus conditions 4–7 and 9 remain trivially satisfied. Condition 8 holds for x as $x \in p'(x)$, and for all other Z-measured vertices because $(p, \prec)$ is a Pauli flow.

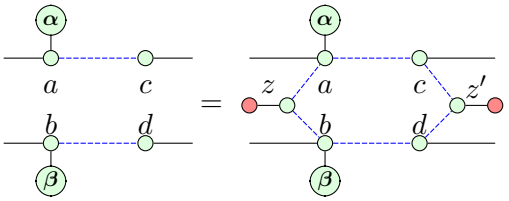
Let $\prec'$ be the transitive closure of $\prec \cup \{(x, v) | v \in N_{G'}(x)\}$. Then $\prec'$ is a partial order because $\prec$ is a partial order and we only add successors for x. Now, condition 1 of Pauli flow is inherited from $(p, \prec)$ for all $u \in V \setminus O$ because $u \notin p'(x)$. Condition 2 is satisfied for all $u \in V \setminus O$ because $\lambda(x) = Z$ and $(p, \prec)$ is a Pauli flow. Condition 3 is inherited because the new vertex has only successors. □

Remark 4.1.2. Note that it is important that we are using Pauli flow here; this rule may not preserve gflow if we take the new vertex to be an arbitrary YZ-measurement. Consider the following rewrite.



The diagram on the left has a gflow with $g(a) = \{b\}$. The diagram on the right cannot have a gflow; as c is YZ-measured, we have that $g(c) = \{c\}$ and $\text{Odd}(g(c)) = \{a, b\}$ or that $g(c) = \{a, b, c\}$ and $\text{Odd}(g(c)) = \emptyset$, thus $c \prec a$ and $c \prec b$. Then, as a is XY-measured we must have $g(a) = \{b\}$ - however, then $Odd(g(a)) = \{a, c\}$ and thus a must precede c for $(g, \prec)$ to be a gflow, giving a contradiction.

Example 4.1.3. A basic example of Z-insertion in use is given below - this easily generalises to more complex examples.

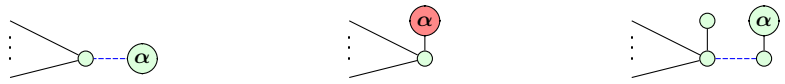


The left-hand diagram has a Pauli flow with $p(a) = \{c\}$, $p(b) = \{d\}$ and $a, b \prec c, d$. Inserting the new Z-measured qubits preserves the existence of Pauli flow - the right hand diagram also has a Pauli flow with $p(a) = \{c\}$, $p(b) = \{d\}$, $p(z) = \{z\}$ and $p(z') = \{z'\}$, with partial order $z, z' \prec a, b \prec c, d$.

4.2 Converting planar measurements to XY-measurements

In the *graph-like diagrams* [27] used in PyZX, all spiders are green and all edges are Hadamard edges. 'Phase gadgets' consist of a degree-1 green spider connected to a phase-free green spider by a Hadamard edge as in the left-most diagram of (4.1). When converting graph-like diagrams to MBQC-form, it is difficult to know whether to interpret phase gadgets as a single YZ-measured vertex (middle diagram) or as an X-measured vertex

connected to a degree-1 XY-measured vertex (right-most diagram):

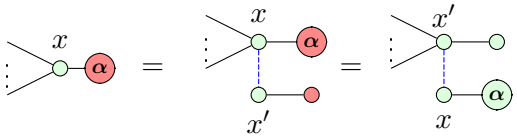


(4.1)

The following proposition shows that both interpretations are valid and can be interconverted.

Proposition 4.2.1. *Let (G, I, O, λ) be a labelled open graph with Pauli flow where $G = (V, E)$, and suppose there exists $x \in V$ with $\lambda(x) = \mathrm{YZ}$. Then (G', I, O, λ') has Pauli flow, where $V' = V \cup \{x'\}$, $E' = E \cup \{\{x, x'\}\}$, and $\lambda'(x) = X$, $\lambda'(x') = \mathrm{XY}$, and $\lambda'(v) = \lambda(v)$ otherwise.*

Proof. Consider the following sequence of rewrites.

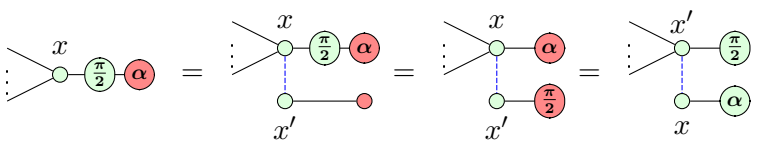


Here we insert the Z-measured vertex x' connected only to x, then pivot along the edge (x, x'). Both Z-insertion and pivoting preserve the existence of Pauli flow, thus our new rewrite rule also preserves the existence of Pauli flow. □

A similar sequence of rewrites allows us to rewrite XZ-measurements in terms of just Y-measurements and XY-measurements.

Proposition 4.2.2. *Let (G, I, O, λ) be a labelled open graph with Pauli flow where $G = (V, E)$, and suppose there exists $x \in V$ with $\lambda(x) = \mathrm{XZ}$. Then (G', I, O, λ') has Pauli flow, where $V' = V \cup \{x'\}$, $E' = E \cup \{\{x, x'\}\}$, and $\lambda'(x) = Y$, $\lambda'(x') = \mathrm{XY}$ and $\lambda'(v) = \lambda(v)$ otherwise.*

Proof. Consider the following sequence of rewrites.



Here we insert a Z-measured vertex x' connected to the XZ-measured vertex x, perform local complementation about x', then pivot along the edge (x, x'). Each of these rewrites preserves the existence of Pauli flow, thus the resulting pattern has Pauli flow. □

Using the two previous propositions, we are able to re-write any XZ- and YZ-planar measurements into a Pauli measurement plus an XY-measurement. This implies the following.

Proposition 4.2.3. *Let (G, I, O, λ) be an arbitrary MBQC-form diagram with Pauli flow. Then there exists an equivalent diagram (G', I', O', λ') with Pauli flow where $\lambda'(v) \in \{X, Y, XY\}$ for all $v \in V'$.*

Proof. We begin by applying Z-deletion (Lemma 3.4.1) to all Z-measured vertices, leaving us with only X, Y, XY, XZ and YZ vertices. It remains to remove all XZ and YZ measurements.

By Proposition 4.2.1, we can convert every YZ-measured vertex into an X-measured vertex connected to an XY-measured vertex while preserving the existence of Pauli flow. Then, by Proposition 4.2.2 we can convert every XZ-measured vertex into a Y-measured vertex connected to an XY-measured vertex. We now only have X, Y and XY measured vertices remaining, and each rewrite rule used to get here preserves the existence of Pauli flow, thus the resulting graph has Pauli flow. □

Remark 4.2.4. Note that Pauli flow is important here: the gflow conditions need not be satisfied if the newly-introduced Pauli measurements were taken to be arbitrary XY-measurements instead.

For example, the first of the following two diagrams has a gflow $(g, \prec)$ with $g(a) = \{c\}$, $g(b) = \{d\}$, $g(x) = \{c, d, x\}$ and $a, b, x \prec c, d$. The second diagram has Pauli flow by Proposition 4.2.1, but it does not have gflow: any flow $(p, \prec')$ on the second diagram must have $x \in p(x')$ to satisfy $x' \in \mathsf{Odd}\,(p(x'))$, as inputs a, b do not appear in correction sets. Similarly, $x' \in p(x)$ as it is the only neighbour. Thus the gflow conditions would

require $x \prec' x'$ and $x' \prec' x$ simultaneously, which is not possible.

4.3 Subdividing an edge

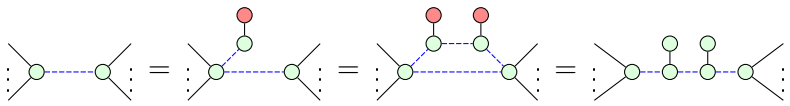
The obfuscation protocol for blind quantum computing of [17] used an unpublished rewrite rule which Backens had proven to preserve the existence of Pauli flow with certain restrictions to measurement planes. The following proposition generalises this to cover all possible measurement planes.

Proposition 4.3.1. *Let $G = (V, E)$ be a graph with vertices V and edges E. Suppose the labelled open graph (G, I, O, λ) has Pauli flow. Pick an edge $\{v, w\} \in E$ and subdivide it twice, i.e. let $G' := (V', E')$, where $V' := V \cup \{v', w'\}$ contains two new vertices v', w', and*

$$E' := (E \setminus \{\{v, w\}\}) \cup \{\{v, w'\}, \{w', v'\}, \{v', w\}\}.$$

Then (G', I, O, λ') has Pauli flow, where $\lambda'(v') = \lambda'(w') = X$ and $\lambda'(u) = \lambda(u)$ for all $u \in V \setminus O$.

Proof. We may subdivide an edge by inserting two Z-measured vertices as shown in the following diagram, then pivoting about these two Z-measured vertices.



As inserting Z-measured vertices and pivoting both preserve the existence of Pauli flow, subdividing an edge also preserves the existence of Pauli flow. □

4.4 Splitting a vertex

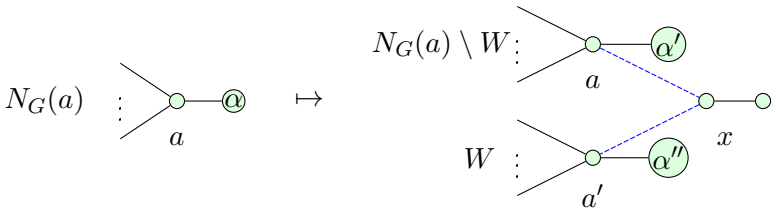
Each of the previously mentioned Pauli-flow preserving rewrite rules only changes measurement angles by integer multiples of $\frac{\pi}{2}$. Here we introduce the first Pauli-flow preserving rewrite rule which allows us to change measurement angles arbitrarily.

To simplify the proof, the proposition requires that all measurements in the pattern are XY, X or Y; by Proposition 4.2.3 this is without loss of generality.

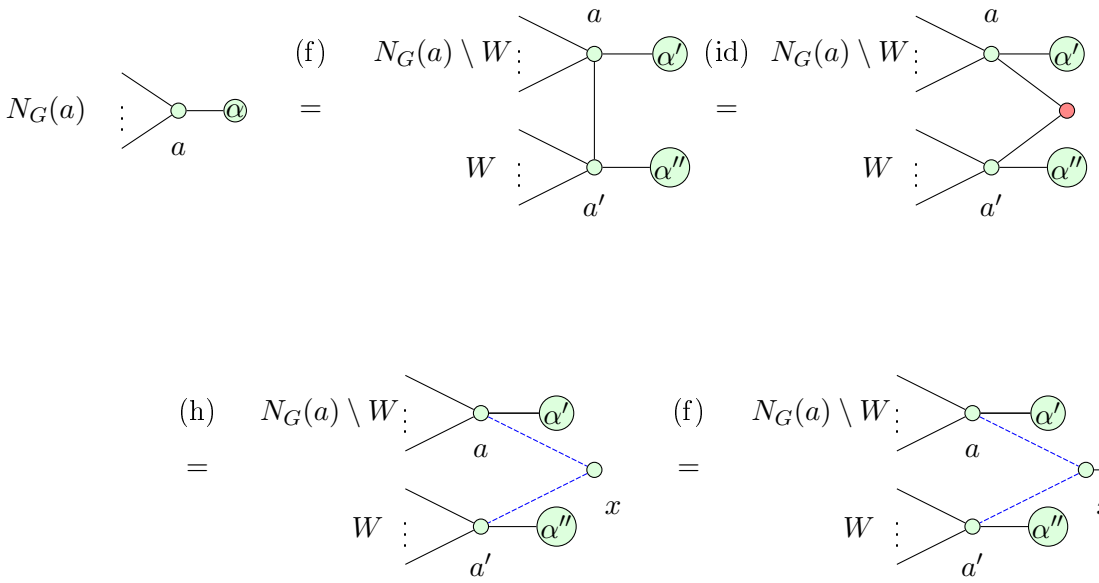
Proposition 4.4.1. *Let $G = (V, E)$ be a graph with vertices V and edges E. Suppose the labelled open graph (G, I, O, λ) has Pauli flow and satisfies $\lambda(u) \in \{\mathrm{XY}, X, Y\}$ for all $u \in O^c$. Pick a vertex $a \in O^c$ such that $\lambda(a) = \mathrm{XY}$ and split it, i.e. let $G' := (V', E')$, where $V' := V \cup \{x, a'\}$ contains two new vertices x, a', and choose some (possibly empty) subset $W \subseteq N(x)$ such that*

$$E' := (E \setminus \{\{a, w\} \mid w \in W\}) \cup \{\{a', w\} \mid w \in W\} \cup \{\{a, x\}, \{x, a'\}\}.$$

Then (G', I, O, λ') has Pauli flow, where $\lambda'(x) = X$, $\lambda'(a') = \mathrm{XY}$, and $\lambda'(u) = \lambda(u)$ for all $u \in V \setminus O$.



Proof. We begin by showing that this rule is valid in the ZX-calculus. Here the first rewrite uses spider unfusion, the second uses the identity rule for 2-arity spiders, the third uses the colour change rule and the final step uses spider unfusion again.



Let $(p, \prec)$ be a focused Pauli flow for (G, I, O, λ); this exists as the labelled open graph has Pauli flow. Since all measurements are XY, X or Y, the focusing conditions from Definition 3.3.13 reduce to:

- For all $u \in V \setminus O$, if $v \in \mathsf{Odd}_G(p(u)) \setminus (O \cup \{u\})$ then $\lambda(v) = Y$.
- For all $u \in V \setminus O$ and all $v \in V \setminus (O \cup \{u\})$ such that $\lambda(v) = Y$, we have $v \in p(u) \leftrightarrow v \in \mathsf{Odd}_G(p(u))$.

Now, for all $u \in V \setminus O$, define

$$p'(u) := \begin{cases} p(u) \cup \{x, a'\} & \text{if } a \in p(u) \text{ and } |p(u) \cap W| \equiv 1 \pmod 2 \\ p(u) \cup \{a'\} & \text{if } a \in p(u) \text{ and } |p(u) \cap W| \equiv 0 \pmod 2 \\ p(u) \cup \{x\} & \text{if } a \notin p(u) \text{ and } |p(u) \cap W| \equiv 1 \pmod 2 \\ p(u) & \text{if } a \notin p(u) \text{ and } |p(u) \cap W| \equiv 0 \pmod 2, \end{cases}$$

then it is straightforward to check that $\mathsf{Odd}_{G'}(p'(u)) = \mathsf{Odd}_G(p(u))$. For example, in the first case, note that $\cup$ can be replaced by Δ since x, a' are new vertices that cannot appear

in $p(u)$. Thus

$$\begin{aligned}
\mathsf{Odd}_{G'}\left(p'(u)\right) &= \mathsf{Odd}_{G'}\left(p(u) \,\Delta\, \{x, a'\}\right) \\
&= \mathsf{Odd}_{G'}\left(p(u)\right) \Delta\, \mathsf{Odd}_{G'}\left(\{x, a'\}\right) \\
&= \mathsf{Odd}_{G'}\left(a\right) \Delta \left(\bigtriangleup_{w \in p(u) \setminus (W \cup \{a\} \cup O)} \mathsf{Odd}_{G'}\left(w\right) \right) \Delta \left(\bigtriangleup_{w \in (p(u) \cap W) \setminus O} \mathsf{Odd}_{G'}\left(w\right) \right) \\
&\Delta\, \{a, x, a'\} \,\Delta\, W \\
&= \mathsf{Odd}_{G}\left(a\right) \Delta \left(\bigtriangleup_{w \in p(u) \setminus (W \cup \{a\} \cup O)} \mathsf{Odd}_{G}\left(w\right) \right) \Delta \left(\bigtriangleup_{w \in (p(u) \cap W) \setminus O} \mathsf{Odd}_{G}\left(w\right) \Delta\, \{a, a'\} \right) \\
&\Delta\, \{a, a'\} \\
&= \mathsf{Odd}_{G}\left(p(u)\right),
\end{aligned}$$

where the third step uses $\mathsf{Odd}_{G'}\left(a\right) = \mathsf{Odd}_{G}\left(a\right) \Delta\, W \,\Delta\, \{x\}$, and the final step uses $|p(u) \cap W| \equiv 1 \pmod 2$.

If a is an input in G, then it remains an input in G' (the 'input' is not a neighbour so cannot be transferred to a' during the splitting process). This is without loss of generality: if a' is desired to be an input, replace W by $N_G(a) \setminus W$ and swap the labels α' and α'' to get a labelled open graph that is equivalent to the desired one up to relabelling $a \leftrightarrow a'$. Having a be an input is compatible with the correction set for x in the next step below. If a is not an input, there is actually a choice of whether to correct x via a or a'; we shall always choose the latter for some slight notational convenience.

Let $p'(x) := \{a'\}$, and let $p'(a') := p'(a) \,\Delta\, \{x\}$, resulting in the following odd neighbourhoods: [1]

$$\mathsf{Odd}_{G'}\left(p'(x)\right) = \mathsf{Odd}_{G'}\left(\{a'\}\right) = W \cup \{x\} \tag{4.2}$$

[1] Note there is a choice for $p'(x)$, since x can be corrected either via a or via a'; we pick the latter for some slight notational convenience.

and

$$\begin{aligned}\mathsf{Odd}_{G'}\left(p'(a')\right) &= \mathsf{Odd}_{G'}\left(p'(a)\right) \Delta\, \mathsf{Odd}_{G'}\left(\{x\}\right) \\ &= \mathsf{Odd}_{G}\left(p(a)\right) \Delta\, \{a, a'\} \\ &= \left(\mathsf{Odd}_{G}\left(p(a)\right) \cup \{a'\}\right) \setminus \{a\} \end{aligned} \tag{4.3}$$

where the final step uses the fact that $a \in \mathsf{Odd}_G\left(p(a)\right)$ and $a' \notin \mathsf{Odd}_G\left(p(a)\right)$ (since a' is not even in G).

Let $\prec'$ be the transitive closure of

$$\prec \cup \Big\{(w, a') \mid w \prec a\Big\} \cup \Big\{(a', w) \mid a \prec w\Big\} \cup \Big\{(x, w) \mid w \in W\Big\} \cup \Big\{(x, a')\Big\}.$$

This is a partial order since a' has the same relationships as a (except for being a successor of x) and x only has successors.

We shall now show that $(p', \prec')$ satisfy the nine conditions of Pauli flow.

Claim 1: For all $u \in O^c$, if $v \in p'(u)$ and $u \neq v$ and $\lambda'(v) \notin \{X, Y\}$, then $u \prec' v$.

- For original vertices $u \in V \setminus O$, $v \in p'(u)$ implies $v \in p(u)$ or $v \in \{x, a'\}$. If $v \in p(u)$ then either $u \prec v$ and thus $u \prec' v$ by the definition of $\prec'$ or $\lambda(u) = \lambda'(u) \in \{X, Y\}$. If $v = x$ then $\lambda'(v) = X$ and thus we don't need to consider this case. Finally, if $v = a'$ then $a \in p(u)$, thus $u \prec a$ which gives us $u \prec' v = a'$ by the definition of $\prec'$.
- For $u = x$, the only element of $p(x)$ is a' and we have $x \prec' a'$.
- For $u = a'$, $v \in p'(a')$ implies $v \in p(a)$ or $v = x$. For the latter case, $\lambda'(x) = X$ and thus we do not need to consider this. In the former case, $v \in p(a)$ gives us that $a \prec v$ or $\lambda(v) = \lambda'(v) \in \{X, Y\}$ as $(p, \prec)$ is a Pauli flow. So either $a' \prec v$ by the definition of $\prec'$ or the property is trivial anyway.

Claim 2: For all $u \in O^c$, if $v \in \mathsf{Odd}_{G'}\left(p'(u)\right)$ and $u \neq v$ and $\lambda'(v) \notin \{Y, Z\}$, then $u \prec' v$.

- For original vertices $u \in V \setminus O$, we have defined p' in such a way that $\mathsf{Odd}_{G'}(p'(u)) = \mathsf{Odd}_G(p(u))$, thus this property is inherited from $(p, \prec)$ being a Pauli flow.
- For $u = x$, we have $\mathsf{Odd}_{G'}(p'(x)) = W \cup \{x\}$, and $x \prec' w$ for any $w \in W$ by the definition of $\prec'$.
- For $u = a'$, $v \in \mathsf{Odd}_{G'}(p'(a'))$ and $v \neq a'$ implies $v \in \mathsf{Odd}_G(p(a))$ by (4.3). As a' has the same successors as a, we get that $a' \prec' v$ as desired.

Claim 3: For all $u \in O^c$, if $\neg(u \prec' v)$ and $u \neq v$ and $\lambda'(v) = Y$, then $v \in p'(u) \longleftrightarrow v \in \mathsf{Odd}_{G'}(p'(u))$.

- For original vertices $u \in V \setminus O$, this is inherited from $(p, \prec)$, as the only changes to correction sets and odd neighbourhoods involve adding or removing x or a', which are not Y-measured.
- For $u = x$, we have $p'(x) \cup \mathsf{Odd}_{G'}(p'(x)) = W \cup \{x, a'\}$. By the definition of the partial order, $x \prec a'$ and $x \prec' w$ for all $w \in W$, so the claim is trivially true.
- For $u = a'$,

  $$\lambda'(v) = Y \text{ and } v \in p'(a') \text{ implies } v \in p(a).$$

  As $(p, \prec)$ is a focused Pauli flow, we must have $v \in \mathsf{Odd}_G(p(a))$, thus $v \in \mathsf{Odd}_{G'}(p'(a))$ and finally $v \in \mathsf{Odd}_{G'}(p'(a'))$ by (4.3).

  For the other direction,

  $$\lambda'(v) = Y \text{ and } v \in \mathsf{Odd}_{G'}(p'(a')) \text{ implies } v \in \mathsf{Odd}_G(p(a)).$$

  As $(p, \prec)$ is a Pauli flow, $v \in p(a)$ thus $v \in p'(a)$ and finally $v \in p'(a')$, as desired.

Claim 4: For all $u \in O^c$, if $\lambda'(u) = \mathrm{XY}$, then $u \notin p'(u)$ and $u \in \mathsf{Odd}_{G'}(p(u))$.

- If $u = a'$, then this is true by inspection.

- If $u = x$, then the claim is true trivially since $\lambda'(x) \neq \mathrm{XY}$.
- If $u \in V \setminus O$, then this property is inherited from $(p, \prec)$.

Claim 5: For all $u \in O^c$, if $\lambda'(u) = \mathrm{XZ}$, then $u \in p'(u)$ and $u \in \mathsf{Odd}_{G'}(p'(u))$.

- This is trivially true as we have no XZ-measured vertices.

Claim 6: For all $u \in O^c$, if $\lambda'(u) = \mathrm{YZ}$, then $u \in p'(u)$ and $u \notin \mathsf{Odd}_{G'}(p'(u))$.

- This is trivially true as we have no YZ-measured vertices.

Claim 7: For all $u \in O^c$, if $\lambda'(u) = X$, then $u \in \mathsf{Odd}_{G'}(p'(u))$.

- if $u = a'$, then this is true trivially since $\lambda'(a') \neq X$.
- If $u = x$, then the claim is true by (4.2).
- If $u \in V \setminus O$, then this property is inherited from $(p, \prec)$.

Claim 8: For all $u \in O^c$, if $\lambda'(u) = Z$, then $u \in p'(u)$.

- This is trivially true as we have no Z-measured vertices.

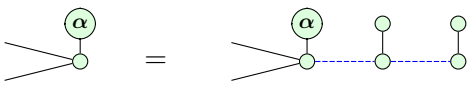
Claim 9: For all $u \in O^c$, if $\lambda'(u) = Y$, then $u \in p'(u)$ and $u \notin \mathsf{Odd}_G(p'(u))$, or $u \notin p'(u)$ and $u \in \mathsf{Odd}_G(p'(u))$.

- This is true trivially for a, x and a', and inherited for all other vertices.

All nine properties are satisfied, therefore $(p', \prec')$ is a Pauli flow for G'. □

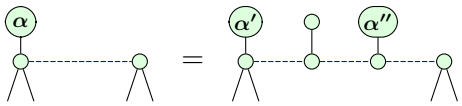
We are able to obtain other useful rewrite rules as immediate corollaries of this.

Corollary 4.4.2. *Using Proposition 4.4.1 with $W = \emptyset$ and $\alpha'' = 0$, we obtain the following rule used in [17].*



This rule can alternatively be derived in a more round-about way from Z-insertion and pivoting, but we next prove a rule that truly requires vertex splitting.

Corollary 4.4.3. *By applying vertex splitting with* $|W| = 1$*, we obtain the following 'neighbour unfusion' rule, where* $\alpha = \alpha' + \alpha''$ *(the measurement for the right-most vertex is not drawn as it can be measured in any plane, or even be an output).*



Chapter 5

Completeness with flow-preserving rewrite rules for Stabilizer MBQC diagrams

The content in this chapter can be found in the paper [46].

In Backens' original proof that the Stabilizer ZX-calculus is complete [6], local complementation and pivoting are used to rewrite any two GS-LC form diagrams that represent the same linear map into the same diagram. Using local complementation and Z-deletion allows us to rewrite any diagram into GS-LC form (up to map-state duality) - as local complementation is its own inverse and Z-insertion is the inverse to Z-deletion, using the Z-insert rule, local complementation, pivoting and Z-deletion rules allows us to rewrite any two equivalent stabilizer MBQC-form diagrams with Pauli flow into one another. We shall do this by rewriting each diagram into a unique normal form using these rules - this section will introduce the first unique normal form for stabilizer ZX-diagrams, then introduce the aforementioned flow-preserving rewrite procedure.

5.1 A canonical form for stabilizer state diagrams

Stabilizer state diagrams in the ZX-calculus have a pseudo-normal form: the rGS-LC form, which arises from the representation of a stabilizer state in terms of a graph state and local Clifford operators [6].

Here, we propose a new pseudo-normal form, based on the representation of a stabilizer state in terms of its affine support and a phase polynomial [4]. Like the rGS-LC form, this is closely related to the stabilizer graphs of Elliott et al. [30] but it translates them into the ZX-calculus differently. The new normal form allows (and in most cases requires) both green and red spiders, meaning it is not strictly 'graph-like'.

Based on a recent proposal by Hu and Khesin [38], we then show how to make this new pseudo-normal form unique, yielding a canonical form for stabilizer state diagrams in the ZX-calculus[1]. In the process, we simplify the uniqueness proof of Hu and Khesin by making use of formalisms and results from the literature about holant problems.

We first prove some lemmas about the algebraic representation of stabilizer states which will be useful in proving uniqueness of the canonical form. Next we introduce to the new pseudo-normal 'phase polynomial form' and show how it corresponds to stabilizer states in phase-polynomial representation. Finally, we define the canonical form, prove its uniqueness, and give an algorithm for rewriting diagrams into canonical form. Throughout this section, diagrams contain red spiders and thus are not in MBQC+LC-form; yet by colour changing all of the red vertices and unfusing phases these can straightforwardly be transformed into MBQC+LC-form diagrams.

[1] At QCTIP 2022, we learned that an analogous result was independently derived by John van de Wetering [65].

5.2 Stabilizer states in terms of affine support and phase polynomial

It has long been known [26, 62] that an n-qubit stabilizer state can be written (up to normalisation) as

$$\sum_{x \in A} i^{l(x)} (-1)^{q(x)} |x\rangle \,, \tag{5.1}$$

where A is an affine subspace of $\mathbb{Z}_2^n$, $l(x) = \sum_j d_j x_j$ for some fixed $d_j \in \mathbb{Z}_2$ is a linear function computed modulo 2, and $q(x) = \sum_{j<k} c_{jk} x_j x_k + \sum_j c_j x_j$ for some fixed $c_{jk}, c_j \in \mathbb{Z}_2$ is a quadratic function. The functions l and q together form a phase polynomial for the state, while A determines the support.

Assuming $\dim(A) = n - m$, the elements of the affine space A are the solutions to a set of linear equations $Rx = b$, where R is an $m \times n$ binary matrix of rank m (with $0 \leq m \leq n$) and $b \in \mathbb{Z}_2^m$. Each component of x is considered a variable. With respect to this linear system, the variables $x_1, \ldots, x_n$ can be partitioned (not generally uniquely) into a set of $(n - m)$ *free* variables and a set of m *dependent* variables such that every assignment of values to the free variables induces exactly one assignment of values to the dependent variables which satisfies all the linear equations. This follows from a standard process of solving the system of linear equations, which also yields a linear equation in terms of the free variables for each dependent variable. In the following, we will denote the set of indices by $[n] := \{1, 2, \ldots, n\}$ and the free variables by a subset $F \subseteq [n]$ of the indices, and write the dependent variables as $x_j = a_j \oplus \bigoplus_{k \in F} a_{jk} x_k$, where $a_j, a_{jk} \in \mathbb{Z}_2$ and the sum is modulo 2. If $a_{jk} = 1$, we say the variable x_j depends on x_k.

It will be useful to give a canonical choice of free variables, this is inspired by Hu and Khesin's normal form for stabilizer states [38], and will lead us to an analogous normal form for stabilizer diagrams.

Definition 5.2.1. We call the result of the following procedure the canonical set of free variables. Start with x_1 and consider the variables in ascending order. For each j, if the value of x_j is fixed by the requirement to satisfy $Rx = b$ given values for all free variables

among $x_1, \ldots, x_{j-1}$ then we say that x_j is dependent. Otherwise we say that x_j is free.

Lemma 5.2.2. *Given an affine space A, the canonical set F is the unique set of free variables with the following property: if x_j depends on the free variable x_k, then $k < j$.*

Proof. Let F' be another set of free variables for A which also has the property that if x_j is a dependent variable and depends on the free variable x_k, then $k < j$. In other words, for each $j \in [n] \setminus F'$, there is an equation $x_j = a_j + \sum_{k<j} a_{jk} x_k$, where furthermore $a_{jk} = 0$ if $k \notin F'$.

Now suppose for a contradiction that $F \neq F'$. The two sets must have the same size $|F| = |F'| = \dim(A)$. Thus, there must be a smallest element $j \in F$ such that $j \notin F'$. Then F' induces an equation

$$x_j = a_j \oplus \bigoplus_{k \in F',\, k<j} a_{jk} x_k. \tag{5.2}$$

Suppose $a_{jk} = 1$ only if $k \in F$. Then the value of x_j is fixed by the free variables of lower index in F, so j should not be free according to Definition 5.2.1, a contradiction.

Otherwise, there exists some $k' \notin F$ such that $a_{jk'} = 1$. But then by the definition of F, there exists some equation $x_{k'} = b_{k'} \oplus \bigoplus_{\ell \in F,\, \ell < k'} b_{k'\ell}$. Thus we can substitute for $x_{k'}$ in (5.2) while preserving the property that x_j only depends on variables of lower index. The process eliminates one variable which is not in F from the decomposition and does not introduce any new variables which are not in F. Hence repeated application will terminate, at which point we have an equation that fixes x_j from only variables in F of index less than j. Again, this means j should not be in F, a contradiction.

Hence we must have $F = F'$. □

As pointed out in the holant literature, it is possible to express the functions l and q solely in terms of the free variables, while keeping their other properties the same [16, Definition 8].

Lemma 5.2.3. *Suppose F denotes a set of free variables for the affine space A, and*

$|\psi\rangle$ is some stabilizer state with support on A. Then there exists a linear function l and a quadratic function q, both depending only on the free variables, as well as a scalar $\lambda \in \mathbb{C} \setminus \{0\}$, such that:

$$|\psi\rangle = \lambda \sum_{x \in A} i^{l(x)} (-1)^{q(x)} |x\rangle .$$

Proof. In (5.1), the functions l and q are allowed to depend on all components of the bit string x, i.e. $l(x) = \bigoplus_j d_j x_j$ for some fixed $d_j \in \mathbb{Z}_2$ and $q(x) = \bigoplus_{j<k} c_{jk} x_j x_k \oplus \bigoplus_j c_j x_j$ for some fixed $c_{jk}, c_j \in \mathbb{Z}_2$.

Given the set of free variables F, solving the defining system of linear equations for A yields linear equations $x_j = a_j \oplus \bigoplus_{k \in F} a_{jk} x_k$ for every $j \in [n] \setminus F$, where $a_j, a_{jk} \in \mathbb{Z}_2$.

Now suppose $d_j \neq 0$ for some $j \notin F$. Then we can substitute

$$l(x) = \bigoplus_{j \in [n]} d_j x_j = \left(\bigoplus_{j \in [n] \setminus \{s\}} d_j x_j \right) \oplus a_s \oplus \bigoplus_{t \in F} a_{st} x_t = a_s \oplus \bigoplus_{j \in [n] \setminus \{s\}} (d_j \oplus a_{sj}) x_j,$$

where we define $a_{sj} = 0$ if $j \notin F$. The a_s is constant and the factor i^{a_s} can be absorbed into the overall scalar λ. Since l is computed modulo 2, the new function satisfies the same properties as the original one but no longer depends on x_s. Furthermore, as $a_{sj} = 0$ for all $j \notin F$, this process does not introduce any new dependencies on dependent variables.

Therefore, the substitution process strictly decreases the number of dependent variables that l depends on and successive applications will eventually yield a function that depends only on free variables. An analogous argument holds for q. □

There are generally multiple ways of expressing the same state in the form of (5.1). Yet if we pick a set of free variables F and require l and q to depend only on free variables, the representation becomes unique. Moreover, we can even give a unique representation in terms of a phase polynomial (evaluated modulo 4, rather than 2).

Lemma 5.2.4. *Given an n-qubit stabilizer state $|\psi\rangle$ and a set $F \subseteq [n]$, there exists a unique polynomial $p(x) = \sum_{j \in F} r_j x_j + 2 \sum_{j,k \in F,\, j<k} s_{jk} x_j x_k$ with $r_j \in \mathbb{Z}_4$ and $s_{jk} \in \mathbb{Z}_2$ and scalar $\lambda \in \mathbb{C} \setminus \{0\}$ such that $|\psi\rangle = \lambda \sum_{x \in A} i^{p(x)} |x\rangle$.*

To prove this, we first need to prove the following lemma.

Lemma 5.2.5. *Let $|\psi\rangle$ and $|\phi\rangle$ be two stabilizer states with the same support A, and let F be a set of free variables for A. Suppose there exists $\lambda, \mu \in \mathbb{C} \setminus \{0\}$ such that*

$$|\psi\rangle = \lambda \sum_{x \in A} i^{l(x)} (-1)^{q(x)} |x\rangle \qquad \textit{and} \qquad |\phi\rangle = \mu \sum_{x \in A} i^{l'(x)} (-1)^{q'(x)} |x\rangle$$

where for some $d_j, d'_j, c_{jk}, c_j, c'_{jk}, c'_j \in \mathbb{Z}_2$,

$$l(x) = \bigoplus_{j \in F} d_j x_j \qquad q(x) = \bigoplus_{j,k \in F,\, j<k} c_{jk} x_j x_k \oplus \bigoplus_j c_j x_j$$

$$l'(x) = \bigoplus_{j \in F} d'_j x_j \qquad q'(x) = \bigoplus_{j,k \in F,\, j<k} c'_{jk} x_j x_k \oplus \bigoplus_j c'_j x_j.$$

Then $|\psi\rangle$ and $|\phi\rangle$ are linearly dependent if and only if for all $j, k \in F$ we have $d_j = d'_j$, $c_{jk} = c'_{jk}$, and $c_j = c'_j$.

Proof. The 'if' direction is straightforward: if $d_j = d'_j$, $c_{jk} = c'_{jk}$, and $c_j = c'_j$ for all $j, k \in F$, then $\mu |\psi\rangle = \lambda |\phi\rangle$.

For the 'only if' direction, note that $l(x) = l'(x) = q(x) = q'(x) = 0$ if all variables in F are assigned 0, so by rescaling such that $\lambda = \mu$, we get $|\psi\rangle = |\phi\rangle$ if and only if they are linearly dependent.

By definition, each assignment of values to the free variables in F induces one assignment of values to all the variables that is in A. Suppose there exists a $j \in F$ such that $d_j \neq d'_j$, wlog assume $d_j = 1$ and $d'_j = 0$ (otherwise the argument is symmetric). Let ξ be the bit string in A that has every free variable set to 0 except the one with index j. Then $\langle \xi \mid \psi \rangle$ is imaginary while $\langle \xi \mid \phi \rangle$ is real, so since the two states have the same non-zero amplitude for the assignment induced by setting all free variables to 0, they cannot be linearly dependent.

Similarly, suppose there exists $j \in F$ such that $c_j \neq c'_j$, then for the same ξ we have $\langle \xi \mid \psi \rangle = - \langle \xi \mid \phi \rangle$, so again the two states cannot be linearly dependent.

So without loss of generality, assume that $d_j = d'_j$ and $c_j = c'_j$ for all $j \in F$. Now suppose

there are $j, k \in F$ such that $c_{jk} \neq c'_{jk}$. Let ζ be the bit string induced by the assignment where $x_j = x_k = 1$ and all other free variables are 0. Then again, $\langle \zeta \mid \psi \rangle = -\langle \zeta \mid \phi \rangle$ so the two states cannot be linearly dependent.

Therefore, linear dependence implies that for all $j, k \in F$ we have $d_j = d'_j$, $c_{jk} = c'_{jk}$, and $c_j = c'_j$. □

We are now able to prove Lemma 5.2.4.

Proof of Lemma 5.2.4. Via Lemmas 5.2.3 and 5.2.5, we can uniquely write

$$|\psi\rangle = \lambda \sum_{x \in A} i^{l(x)} (-1)^{q(x)} |x\rangle$$

where $l(x) = \bigoplus_{j \in F} d_j x_j$ and $q(x) = \bigoplus_{j,k \in F,\, j<k} c_{jk} x_j x_k \oplus \bigoplus_j c_j x_j$ with all coefficients taking values in $\mathbb{Z}_2$.

As $y \bmod 2 = y^2 \bmod 4$ for all $y \in \mathbb{Z}$, we have

$$\bigoplus_{j \in F} d_j x_j = \left(\sum_{j \in F} d_j x_j \right)^2 \bmod 4 = \left(\sum_{j \in F} d_j x_j + 2 \sum_{j,k \in F,\, j<k} d_j d_k x_j x_k \right) \bmod 4,$$

where we have used the fact that $d_j, x_j \in \mathbb{Z}_2$ for all j and hence $(d_j x_j)^2 = d_j x_j$. We can thus write

$$\sum_{x \in A} i^{l(x)} (-1)^{q(x)} |x\rangle = \sum_{x \in A} i^{p(x)} |x\rangle ,$$

where

$$\begin{aligned} p(x) &= \sum_{j \in F} d_j x_j + 2 \left(\sum_{j,k \in F,\, j<k} c_{jk} x_j x_k + \sum_j c_j x_j \right) + 2 \sum_{j,k \in F,\, j<k} d_j d_k x_j x_k \\ &= \sum_{j \in F} (d_j + 2c_j) x_j + 2 \sum_{j,k \in F,\, j<k} (c_{jk} + d_j d_k) x_j x_k \end{aligned}$$

Now, $r_j := (d_j + 2c_j) \in \mathbb{Z}_4$. The coefficient $s_{jk} := c_{jk} + d_j d_k$ could take value 2, but as p is in the exponent of i and s_{jk} is multiplied by 2, we may without loss of generality replace it with $s_{jk} := c_{jk} \oplus d_j d_k$ so that $s_{jk} \in \mathbb{Z}_2$.

Conversely, we can find functions l and q from p by setting $d_j := r_j \bmod 2$, $c_j := \frac{1}{2}(r_j - d_j)$,

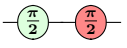
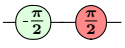
and $c_{jk} := s_{jk} \oplus d_j d_k$. Thus, by uniqueness of l and q, the phase polynomial expression is also unique. □

5.3 A new pseudo-normal form related to phase polynomials

In the rGS-LC form for stabilizer state diagrams, local Clifford operators on the graph state are expressed in terms of green and red spiders. Alternatively, it is also possible to express local Clifford operators in terms of green spiders and Hadamards (and this is what is done in the stabilizer graph formalism of [30]). In ZX-terms, this means the allowed local Clifford operators are  and  , where $k \in \mathbb{Z}_4$ and $a \in \mathbb{Z}_2$. As for red nodes in rGS-LC diagrams, qubits whose local Clifford operator contains an H are not allowed to be connected to each other; therefore we can 'push' the Hadamards through and get the following pseudo-normal form. It is possible to convert between the two kinds of local Clifford operators via local complementations on the qubits that have red nodes or Hadamards.

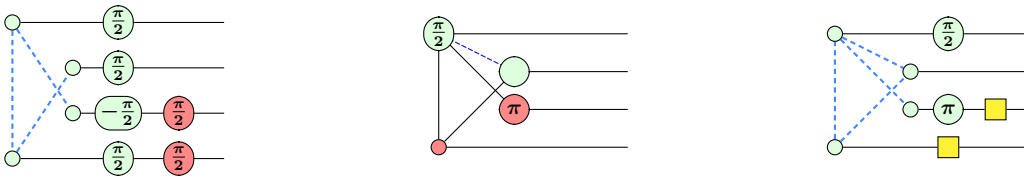
Definition 5.3.1. A stabilizer ZX-calculus diagram is in phase-polynomial form if the following hold:

- Each dangling edge is connected to a unique red or green spider.
- Red spiders have phases that are 0 or π.
- Green spiders have phases that are integer multiples of $\pi/2$.
- There may be edges connecting spiders of different colours.
- Furthermore, green spiders may be connected to other green spiders via Hadamard nodes.

Observation 5.3.2. *An rGS-LC diagram can be brought into phase-polynomial form via the following process. First, apply local complementations to all qubits that have red nodes in their local Cliffords. This maps*  *to*  *and*  *to*  .

Then, change the colour of all spiders which now have Hadamards as part of their vertex operators and merge adjacent spiders of the same colour.

Example 5.3.3. Applying this procedure to the rGS-LC diagram on the left yields the phase polynomial-form diagram in the middle. Colour-changing each red spider and unfusing the phases leads to an equivalent GS-LC form diagram which we will say is in phase-polynomial form up to colour changing the spiders with Hadamard gates in their vertex operators.



Diagrams in phase-polynomial form correspond directly to pairs of a state and a set of free variables for the underlying affine support. We shall prove that this holds and then give an example illustrating this correspondence.

Lemma 5.3.4. *Ignoring scaling, there is a bijection between phase-polynomial form diagrams and pairs* $(|\psi\rangle, F)$*, where* $|\psi\rangle$ *is an* n*-qubit stabilizer state and* $F \subseteq [n]$ *indicates a set of free variables for the affine space* A *which is the support of* $|\psi\rangle$*.*

Proof. By Lemma 5.2.4, there exists a unique function $p(x) = \sum_{j\in F} r_j x_j + 2\sum_{j,k\in F,\, j<k} s_{jk} x_j x_k$ with $r_j \in \mathbb{Z}_4$ and $s_{jk} \in \mathbb{Z}_2$ such that $|\psi\rangle \cong \sum_{x\in A} i^{p(x)} |x\rangle$. To construct a diagram from a state and a set of free variables from this, proceed as follows:

- For each dependent variable x_k with $k \in [n] \setminus F$, find the unique linear expression $x_k = a_k \oplus \bigoplus_{j\in F} a_{kj} x_j$ which satisfies the defining linear equations $Rx = b$ of the affine space A.
- For each $j \in F$, place a green spider with an output wire. The phase of this spider is $r_j \frac{\pi}{2}$.
- For each $k \in [n] \setminus F$, place a red spider with an output wire. The phase of this spider is $a_j \pi$.
- Draw a (plain) edge connecting the green spider j to the red spider k whenever

$a_{kj} = 1$.

- Draw a Hadamard edge connecting the green spiders j and j' whenever $s_{jj'} = 1$.

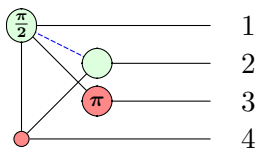
Conversely, given a diagram in phase-polynomial form, construct the corresponding state as below:

- The set F of free variables consists of the indices of the green spiders.
- The affine space A is defined by the set of equations $\left\{x_j = a_j \oplus \bigoplus_{k \in N(j)} x_k\right\}_{j \in [n] \setminus F}$, where $a_j = 0$ if the phase of the red spider with index j is 0, and 1 otherwise.
- For each $j \in F$ such that the phase of the green spider j is α_j, define r_j to be the value in $\mathbb{Z}_4$ that is equivalent to $\frac{2\alpha_j}{\pi} \bmod 4$.
- For each $j, k \in F$ with $j < k$, define $s_{jk} = 1$ if there exists a Hadamard edge between spiders j and k, and $s_{jk} = 0$ otherwise.

Let $p(x) := \sum_{j \in F} r_j x_j + 2 \sum_{j,k \in F,\, j<k} s_{jk} x_j x_k$, then the desired state is $\sum_{x \in A} i^{p(x)} |x\rangle$. The two procedures are inverses of each other (noting that $\frac{3\pi}{2} \equiv -\frac{\pi}{2} \bmod 2\pi$).

Suppose D is the ZX-diagram corresponding to some stabilizer state $|\psi\rangle$ according to the above translation. Then it is straightforward to see that the support of $[\![D]\!]$ and the support of $|\psi\rangle$ are equal. Thus, by phase-polynomial techniques, it is quick to check that $[\![D]\!]$ equals $|\psi\rangle$ up to scalar factor. □

Example 5.3.5. Consider the following phase-polynomial form diagram from Example 5.3.3, where we have numbered the qubits from top to bottom.



Following the procedure from Lemma 5.3.4, we construct the state corresponding to this diagram. The state will be expressed as $\sum_{x \in A} i^{p(x)} |x\rangle$, where $p(x) = \sum_{j \in F} r_j x_j + 2 \sum_{j,k \in F,\, j<k} s_{jk} x_j x_k$. Here, F is the set of free variables, A is the affine space on which the state has support, and $p(x)$ is the phase polynomial with $r_j \in \mathbb{Z}_4$ and $s_{jk} \in \mathbb{Z}_2$ for all $j, k \in F$.

- The set F of free variables corresponding to this diagram is $F = \{x_1, x_2\}$ since qubits 1 and 2 are denoted by green spiders.
- The affine space A is defined by the following set of equations arising from the red spiders:

$$x_3 = 1 \oplus x_1 \qquad\qquad x_4 = x_1 \oplus x_2 \tag{5.3}$$

  since qubit 3 has phase π (giving the constant 1 on the right-hand side) and is connected to qubit 1, while qubit 4 has phase 0 and is connected to both 1 and 2.
- For the linear terms in the phase polynomial, we get that $r_1 = 1$ and $r_2 = 0$ as the phase of x_1 is $\frac{\pi}{2}$ and the phase of x_2 is 0.
- For the quadratic terms in the phase polynomial, we have $s_{12} = 1$ as there is a Hadamard edge connecting x_1 and x_2.

Combining these, the phase polynomial is $p(x) = x_1 + 2x_1x_2$. The state corresponding to the diagram is therefore given by:

$$\begin{aligned}\sum_{x \in A} i^{x_1 + 2x_1x_2} |x\rangle &= \sum_{x_1, x_2 \in \mathbb{Z}_2} i^{x_1} (-1)^{x_1x_2} |x_1 x_2 (1 \oplus x_1)(x_1 \oplus x_2)\rangle \\ &= |0010\rangle + |0111\rangle + i|1001\rangle - i|1100\rangle\end{aligned}$$

It is then quick to check that applying the procedure in Lemma 5.3.4 for constructing a diagram from a state and a set of free variables gives back the original diagram.

Instead, we will show how to construct the diagram corresponding to the same state with a different set of free variables $F = \{x_2, x_3\}$. To do this, we first rewrite the affine space and the phase polynomial in terms of the new free variables x_2 and x_3, and then apply the procedure for obtaining diagrams.

Choosing x_3 to be free instead of x_1, we rearrange the first equation of (5.3) and then substitute it into the second to get:

$$x_1 = 1 \oplus x_3 \qquad\qquad x_4 = 1 \oplus x_2 \oplus x_3 \tag{5.4}$$

Substituting into the phase polynomial yields $p(x) = (1 \oplus x_3) + 2(1 \oplus x_3)x_2$ where $\oplus$ denotes addition modulo 2. Yet we want the phase polynomial to be computed modulo 4, since $i^4 = 1$. Now, as $y \bmod 2 = y^2 \bmod 4$ for all $y \in \mathbb{Z}$, and $b^2 = b$ for all $b \in \mathbb{Z}_2$, this can be rewritten to:

$$p(x) = (1 \oplus x_3) + 2(1 \oplus x_3)x_2 = (1 + x_3)^2 + 2(1 + x_3)^2 x_2 = 1 + 2x_2 + 3x_3 + 2x_2x_3 \pmod 4$$

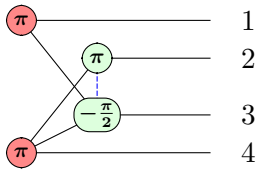
We thus have $r_2 = 2$, $r_3 = 3$, and $s_{23} = 1$. The constant term in the phase polynomial is irrelevant since we are ignoring global scalars. Up to scalar factor, the full state is

$$\sum_{x_2, x_3 \in \mathbb{Z}_2} i^{2x_2 + 3x_3 + 2x_2x_3} \left|(1 \oplus x_3)x_2x_3(1 \oplus x_2 \oplus x_3)\right\rangle .$$

To construct the diagram corresponding to this state and set of free variables:

- We already have the equations for the dependent variables in terms of $F = \{x_2, x_3\}$ in (5.4).
- Place a green spider with phase $r_2 \frac{\pi}{2} = \pi$ for qubit 2 and a green spider with phase $r_3 \frac{\pi}{2} = \frac{3\pi}{2}$ (or, equivalently, $-\frac{\pi}{2}$) for qubit 3. Each of the spiders is connected to one output wire.
- Place a red spider with phase π for qubit 1 and a red spider with phase π for qubit 4 since the equations for both x_1 and x_4 contain a constant term. Again, each of the spiders is connected to one output wire.
- Variable x_1 depends on x_3, so draw a plain wire between the spiders for qubits 1 and 3. Variable x_4 depends on both x_2 and x_3, so draw plain wires between the spiders for qubits 2 and 4, as well as between 3 and 4.
- As $s_{23} = 1$, draw a Hadamard edge connecting the green spiders corresponding to x_2 and x_3.

This yields the following diagram:



5.4 The canonical phase-polynomial diagram

Using the bijection between phase-polynomial form diagrams and pairs of a state and a set of free variables, we can now define a unique canonical diagram for any stabilizer state.

Definition 5.4.1. Let $|\psi\rangle$ be a stabilizer state, then its canonical diagram is the one translated from $(|\psi\rangle, F)$ by Lemma 5.3.4, where F is the canonical set of free variables according to Definition 5.2.1.

Apart from the translation into our terminology, this differs from the normal form definition of Hu and Khesin [38] only by reversing the order: we ask for free variables to come first whereas they put them last. Our uniqueness proof, making use of the properties of the affine support of a stabilizer state is shorter and simpler than that in [38].

Theorem 5.4.2. *The canonical form is unique.*

Proof. This follows from the uniqueness of the canonical set of free variables proved in Lemma 5.2.2 and from the bijection between pairs consisting of a state and a set of free variables in Lemma 5.3.4. □

Proposition 5.4.3. *Every phase-polynomial form diagram can be re-written into canonical form using only local complementation and pivoting.*

Proof. Pick some order $<$ on the spiders, say from top to bottom. We want each red spider to only be connected to spiders that appear earlier in $<$. While this does not hold, repeat the following procedure:

1. Let d_k be the minimal red spider under $<$ such that there exists some green spider f_j connected to d_k with $d_k < f_j$.

2. Let f_h be the maximal green spider under $<$ such that d_k is connected to f_h.

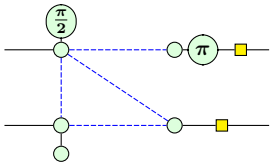
3. If f_h has a phase of $\pm\frac{\pi}{2}$, perform local complementation about f_h and then about d_k. Otherwise, pivot about the edge connecting f_h and d_k. After applying either of these equivalence transformations, f_h is now red and d_k is now green and the diagram is still in phase-polynomial form.

4. By maximality of f_h, we have that f_h is only connected to green spiders f_n with $f_n < f_h$. By minimality of d_k, we have that d_k is only connected to red spiders d_m with $d_k < d_m$.

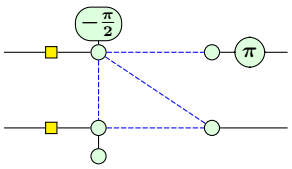
This procedure strictly reduces the number of connections between red spiders and green spiders that appear later in the order. Hence repeating it will eventually terminate, transforming any phase-polynomial form diagram into canonical form. □

Remark 5.4.4. The canonical form is unique only up to the choice of order on the qubits; different orders may yield different 'canonical forms'. Thus the choice of order is arbitrary (but needs to happen in advance, independently of the diagram considered) – we have chosen top-to-bottom for simplicity.

Example 5.4.5. Later, we shall come across the following diagram in canonical form.



This is in canonical form when we take the input (left-hand) qubits to come before the output qubits in the chosen order - if we instead reverse the order, the canonical form of the same state would instead be the following diagram (obtained by pivoting on the top and bottom edges of the previous diagram).



5.5 Complete flow-preserving rewrite rules

We are now able to assemble the main proof. In the following, we will say an MBQC+LC-form diagram has *no interior spiders* if the MBQC-form part of the diagram (i.e. ignoring the local Cliffords) has no interior vertices ($V \setminus (I \cup O) = \emptyset$). Additionally, we say an MBQC+LC-form diagram has Pauli flow if its MBQC-form part has Pauli flow (analogous to gflow in [9, Section 4.1]).

Theorem 5.5.1. *Given two equivalent stabilizer MBQC+LC-form diagrams D and D' with Pauli flow and satisfying $[\![D]\!] \cong [\![D']\!]$, there exists a sequence of rewrite rules – each preserving the existence of Pauli flow and preserving the MBQC+LC-form – transforming D into D'.*

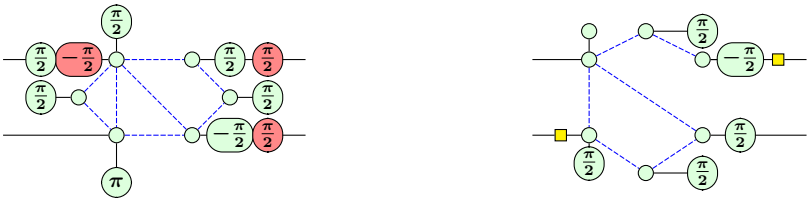
Proof. We begin by deleting all Z-measured vertices from both diagrams, keeping track of which vertices we delete and their set of neighbours when deleted. The resulting diagrams has Pauli flow by Lemma 3.4.1. After all Z-measured vertices are removed, the MBQC-form parts of the diagrams (ignoring the local Cliffords) only have X and Y measurements and are thus of the kind considered in [27]. Then, there exists a terminating procedure (consisting of a sequence of local complementations, pivots and Z-deletions) rewriting the two diagrams into MBQC+LC-form diagrams N and N' which contain no interior spiders [27, Theorem 5.4]. Since local complementation and pivoting also preserve the existence of Pauli flow (Lemmas 3.4.3 and 3.4.4), N and N' will also have Pauli flow.

As only X and Y measurements remain, they can be spider-merged and unmerged through each qubit to become local Cliffords on the outputs, thus N and N' are equivalent to GS-LC form diagrams. By [6, Theorem 13], every GS-LC form diagram can be rewritten into rGS-LC form using a sequence of local complementations, thus this step preserves Pauli flow. By Observation 5.3.2, we can then rewrite each diagram into phase polynomial form, again using only local complementations (along with some operations on the local Cliffords that do not alter the flow), thus preserving Pauli flow. Finally, by Proposition 5.4.3, we can rewrite each diagram into canonical form[1]. The rewrite steps use only local

complementations and pivoting, so they preserve Pauli flow. The resulting diagrams are equivalent and the canonical form is unique, so we have found a sequence of local complementations, pivots and Z-deletions rewriting D and D' into the same canonical form diagram C.

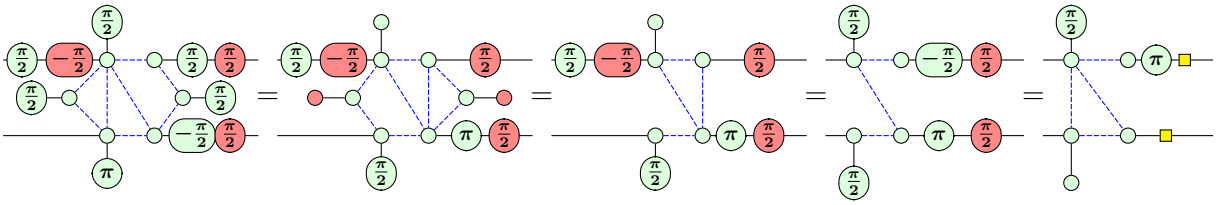
By Observation 3.4.5, local complementation and pivot can be inverted. Furthermore, Z-insert is a Pauli-flow preserving inverse to Z-delete. Thus the sequence of rewrites from D' to C can be inverted while still preserving Pauli-flow. By rewriting D to C, then rewriting C to D', we obtain a sequence of flow-preserving rewrite rules transforming D into D'. This completes the proof. □

Example 5.5.2. We shall give a short example of this rewrite procedure in action. Consider the following two MBQC+LC-form diagrams, which we will call D and D', and which satisfy $[\![D]\!] \cong [\![D']\!]$ by (non-flow preserving) diagram simplification techniques.

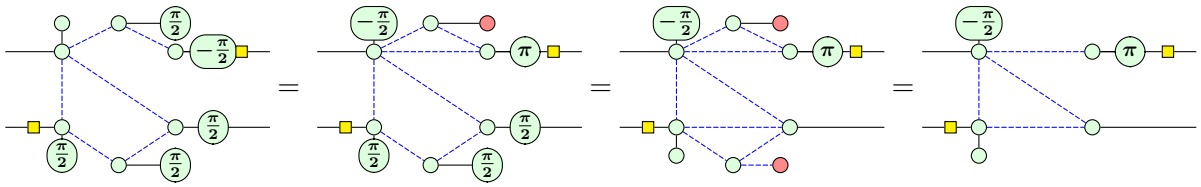


Using the procedure from the proof of Theorem 5.5.1, we first rewrite D to phase polynomial form. Perform triple local complementations (i.e. 'inverse local complementations') about both the left-most and right-most qubits in the MBQC-form part, then apply Z-deletion to these qubits. A local complementation about the top left qubit gives us the fourth diagram, which is in rGS-LC form and in fact is equivalent to the left-most diagram in Example 5.3.3 up to map-state duality. We then obtain the final diagram by following the procedure in Observation 5.3.2; note that this diagram is already in canonical form (up to map-state duality and colour changing spiders with Hadamard gates in their vertex operators) assuming that the input qubits have lower indices than the output qubits.

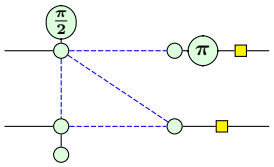
[1] Up to map-state duality and colour changing vertices with Hadamard operators.



For D', we perform local complementation about the two interior qubits of the MBQC-form part (here we have done this about the top qubit first, then the bottom qubit), and Z-delete both qubits.



This final diagram is already in phase polynomial form (up to map state duality and colour changing the spiders with Hadamard edges in their vertex operators) without us having to go through rGS-LC form. To rewrite this diagram into canonical form, all that remains is to pivot along the edge connecting the bottom left qubit to the bottom right qubit, giving the following diagram:



We have therefore rewritten D and D' into the same canonical form diagram. Every rule used to re-write D and D' to canonical form is invertible and the inverses preserve Pauli flow, giving us a sequence of flow preserving rewrite rules taking D to D'.

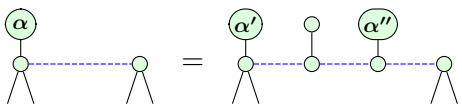
Chapter 6

Neighbour unfusion

The content in this chapter can be found in the paper [47].

In [60], a rewrite rule called *neighbour unfusion* was used to reduce the number of two-qubit gates in circuits via the ZX-calculus. Using neighbour unfusion allowed for the two-qubit gate count to be greatly reduced, but introduced a new problem: neighbour unfusion, which introduces two new qubits in each application, was found to not always preserve gflow. Yet a flow is needed to be able to translate back to a circuit after the application of the two-qubit gate count reduction algorithm. We now show that neighbour unfusion preserves the existence of Pauli flow, so circuit re-extraction is always possible.

Corollary 6.0.1. *By applying vertex splitting with $|W| = 1$, we obtain the following 'neighbour unfusion' rule, where $\alpha = \alpha' + \alpha''$ (the measurement for the right-most vertex is not drawn as it can be measured in any plane, or even be an output).*



As vertex splitting preserves the existence of Pauli flow, we then have that neighbour unfusion also preserves the existence of Pauli flow as it is a specific case of vertex splitting.

Switching to using Pauli flow in the algorithm of [60] ended up being slower than using

gflow due to the circuit extraction procedure undoing a lot of the extra uses of neighbour unfusion, as well as the circuit extraction being slower itself. It is then natural to question under what conditions neighbour unfusion preserves the existence of gflow. Staudacher et al. [60] state that, in the case of only XY-measurements, neighbour unfusion empirically fails to preserve gflow if the two vertices to which neighbour unfusion is applied are extracted to different qubits in the circuit extraction. This was stated without proof and it is not immediately obvious which qubits are extracted onto the same qubit, thus finding the right condition remained an open problem.

We originally struggled to find a condition that is both sufficient and necessary for neighbour unfusion to preserve gflow and instead developed separate sufficient and necessary conditions. Since then, we have been able to find the "correct" condition that is both sufficient and necessary. We shall begin by introducing the separate necessary and sufficient conditions for neighbour unfusion to preserve the existence of gflow, give examples of where these fail, then provide the proof of the condition that is both sufficient and necessary.

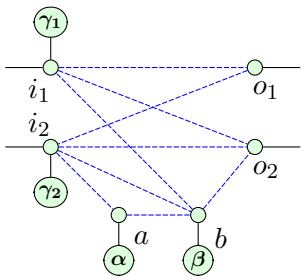
6.1 A sufficient condition for preservation of gflow

Proposition 6.1.1. *Let $\Gamma = (G, I, O, \lambda)$ be a labelled open graph and suppose a, b are two adjacent vertices in G with $\lambda(a) = \lambda(b) = \mathrm{XY}$. Suppose Γ has focused gflow $(g, \prec)$ where $b \in g(a)$ and for all $w \in V \setminus \{a, b\}$, $w \prec b \implies w \prec a$ and $a \prec w \implies b \prec w$. Let Γ' be the labelled open graph after applying neighbour unfusion to a and b. Then Γ' has gflow. The same holds with the roles of a and b reversed.*

This condition is sufficient but not necessary - we will illustrate the neighbour unfusion process with an example that shows some choices of unfusion which do preserve gflow and others which do not, then give an example of when neighbour unfusion preserves gflow without this condition being satisfied.

Example 6.1.2. Consider the following MBQC-form diagram, which appeared in a dif-

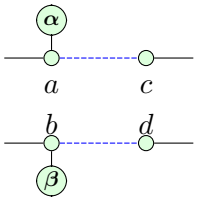
ferent context in [50].



This has a focused gflow $(g, \prec)$ with $g(i_1) = \{a, o_2\}$, $g(i_2) = \{a, b, o_2\}$, $g(a) = \{b, o_2\}$ and $g(b) = \{o_1, o_2\}$ and partial order $i_1, i_2 \prec a \prec b \prec o_1, o_2$. Then neighbour unfusion along one of the edges (i_2, a), (a, b) or (b, o_2) preserves gflow by Proposition 6.1.1.

On the other hand, the pair (i_2, o_2) satisfies the condition $o_2 \in g(i_2)$ but satisfies neither $w \prec o_2 \implies w \prec i_2$ nor $i_2 \prec w \implies o_2 \prec w$ since a and b sit in between the pair in the partial order. Applying neighbour unfusion to i_2 and o_2 does not preserve the existence of gflow since the odd neighbourhood of $\{o_1, o_2\}$ would become $\{i_2, i_2', b\}$ and thus no vertex can be corrected solely using the outputs in the resulting diagram. An analogous argument holds for the pair (i_2, o_1) for which $o_1 \notin g(i_2)$.

Example 6.1.3. The condition in 6.1.1 requires that there are no vertices measured at the same time as a or b. This does not have to be the case for neighbour unfusion to preserve gflow - consider the following diagram.



This has a (focused) gflow with $g(a) = \{c\}$, $g(b) = \{d\}$ and $a, b \prec c, d$. Applying neighbour unfusion along the edge between a and c (or between b and d) preserves the existence of Pauli flow despite the condition '$\forall w \in V \setminus \{a, c\}$, $w \prec c \to w \prec a$' from 6.1.1not being satisfied.

6.2 A necessary condition for preservation of gflow

We now show that, for MBQC-form diagrams with equal numbers of inputs and outputs, the condition $b \in g(a)$ (or instead $a \in g(b)$) is necessary for neighbour unfusion to preserve gflow.

Proposition 6.2.1. *Let $\Gamma = (G, I, O, \lambda)$ be a labelled open graph with $|I| = |O|$ with focused gflow $(g, \prec)$. Suppose a, b are two adjacent vertices in G with $\lambda(a) = \lambda(b) = \mathrm{XY}$. Let Γ' be the labelled open graph after applying neighbour unfusion to a and b in Γ, and suppose Γ' has gflow. Then we must have $b \in g(a)$ or $a \in g(b)$ for the focused gflow on Γ.*

Proof. Suppose we have a pattern Γ' with the subdiagram (6.1) and assume that Γ' has a focused gflow $(g', \prec')$. Let $\Gamma'' = (G'', I, O, \lambda'')$ be the induced sub-pattern containing only those vertices of G' that are either outputs or measured in the XY-plane; this must include all inputs since those cannot be measured in planes XZ or YZ. This new measurement pattern still contains the subdiagram (6.1) and it has gflow [9, Lemma 3.15]. In fact, since $(g', \prec')$ is focused, it implicitly follows from [9, Proposition 3.14 and Lemma 3.15] that the gflow of the new pattern is just the restriction of the old gflow function to a smaller domain, and this is still focused; denote it by $(g'', \prec'')$.

Now every focused gflow in a pattern with only XY-plane measurements and equal numbers of inputs and outputs can be reversed in a very strict sense: Let $\Gamma''' = (G'', O, I, \lambda''')$ be the reversed pattern with the roles of inputs and outputs swapped and λ''' mapping all non-outputs to XY. Then there exists a focused gflow $(g''_{rev}, \prec''_{rev})$ for Γ''' where $\prec''_{rev}$ is the reverse of $\prec''$ and $u \in g''_{rev}(v)$ if and only if $v \in g''(u)$ [48], cf. also [9, Corollary 2.47].

As x is XY-measured and has two neighbours, to satisfy $x \in \mathsf{Odd}_{G''}(g''(x))$ and $x \in \mathsf{Odd}_{G''}(g''_{rev}(x))$ we require the following to hold, where $\oplus$ is the exclusive-or operator:

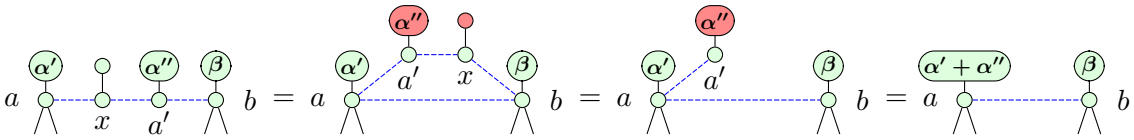
$$(a \in g''(x) \wedge x \in g''(a')) \oplus (a' \in g''(x) \wedge x \in g''(a)).$$

As a' is also XY-measured and has two neighbours, by the same reasoning we obtain:

$$(b \in g''(a') \wedge a' \in g''(x)) \oplus (x \in g''(a') \wedge a' \in g''(b)).$$

Then, as we cannot have both $x \in g''(a')$ and $a' \in g''(x)$, we have either that $a \in g''(x)$, $x \in g''(a')$ and $a' \in g''(b)$ or that $b \in g''(a')$, $a' \in g''(x)$ and $x \in g''(a)$ for $(g'', \prec'')$ to be a gflow for Γ''. But $(g'', \prec'')$ is the restriction of $(g', \prec')$ to the XY-measured vertices in Γ'. Thus either $a \in g'(x)$, $x \in g'(a')$ and $a' \in g'(b)$ or that $b \in g'(a')$, $a' \in g'(x)$ and $x \in g'(a)$.

Now, consider the following sequence of rewrites corresponding to the inverse of neighbour unfusion:



where we first pivot along the edge (x, a'), then apply Z-deletion to a' and finally apply the phase gadget identity rule of [43] to add the phase of a' to that of a. Each of these rules preserves the existence of gflow, thus the inverse of neighbour unfusion preserves the existence of gflow. Moreover, if $x \in g'(a)$, $a' \in g'(x)$ and $b' \in g'(a')$, then after applying the inverse of neighbour unfusion we get a focused gflow $(g, \prec)$ for Γ with $b \in g(a)$ (and similarly if $a \in g'(x)$, $x \in g'(a')$ and $a' \in g'(b)$ we get a focused gflow with $a \in g(b)$). Therefore, if the measurement pattern after neighbour unfusion has gflow, then the original pattern has a focused gflow where b is in the correction set of a, or a focused gflow where a is in the correction set of b. □

An analogous argument to the above works if b is an output, in which case the only option is for b to be in the correction set of a. Therefore the above proposition covers all the cases relevant to Staudacher et al.'s work on patterns where all measurements are in the XY-plane.

Example 6.2.2. The following two measurement patterns are related by neighbour unfusion along the edge between vertices a and b:

In the first pattern, a and b are both inputs and thus cannot appear in correction sets. Hence the pattern does not have a gflow where a is in the correction set of b or where b is in the correction set of a. Yet it does have a gflow $(g, \prec)$ with $g(a) = \{c\}$, $g(b) = \{d\}$ and $a, b \prec c, d$.

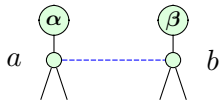
For the second pattern to have a flow $(p, \prec')$, we require $a' \in p(x)$ and $x \in p(a')$ since both vertices need to be in the odd neighbourhood of their correction set and inputs cannot appear in correction sets. This diagram can therefore not have a gflow, as the gflow conditions would require that $x \prec' a'$ and $a' \prec' x$ simultaneously, so $\prec'$ would not be strict. This diagram does have a Pauli flow however, as the X-measured vertex x does not need to come after a' in the partial order in the case of Pauli flow. The Pauli flow satisfies $p(a) = \{c\}$, $p(b) = \{d\}$, $p(x) = \{d, a'\}$ and $p(a') = \{c, x\}$ with $x \prec a, b, a' \prec c, d$.

6.3 The 'correct' condition to ensure neighbour unfusion preserves gflow

We shall now introduce the condition that is both sufficient and necessary for neighbour unfusion to preserve the existence of gflow.

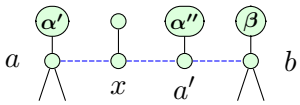
Theorem 6.3.1. *Given some labelled open graph $\Gamma = (V, E, I, O, \lambda)$ with $|I| = |O|$ and neighbouring vertices $a, b \in V$ such that $\lambda(a) = \lambda(b) = XY$, the graph $\Gamma' = (V', E', I, O, \lambda')$ obtained by applying neighbour unfusion to a and b in Γ has gflow if and only if there exists a focused gflow $(g, \prec)$ on Γ such that $a \prec b$ and there is no vertex v satisfying $a \prec v \prec b$ (or equivalently with the roles of a and b reversed).*

Proof. Consider a labelled open graph Γ which has a focused gflow $(g, \prec)$ and contains the subdiagram



Assume without loss of generality that Γ has a focused gflow $(g, \prec)$ satisfying $a \prec b \wedge \ / \exists v : a \prec v \prec b$. Then as $\lambda(a) = \lambda(b) = XY$, we have that $b \in g(a)$.

Neighbour unfusion yields a labelled open graph $\Gamma' = (G', I, O, \lambda')$ with the following subdiagram, where $\alpha' + \alpha'' = \alpha$:



(6.1)

We can construct a focused gflow for the new pattern by defining the correction sets as follows:

$$g'(v) = \begin{cases} g(v) \cup \{x, a'\} & \text{if } a \in g(v) \wedge b \in g(v) \\ g(v) \cup \{a'\} & \text{if } a \in g(v) \wedge b \notin g(v) \\ g(v) \cup \{x\} & \text{if } a \notin g(v) \wedge b \in g(v) \\ g(b) \cup \{a'\} & \text{if } v = x \\ g(a) & \text{if } v = a' \\ g(v) & \text{otherwise.} \end{cases}$$

This choice leaves invariant the odd neighbourhoods of the correction sets of any original (non-output) vertices. Furthermore we have

$$\mathsf{Odd}_{G'}\left(g'(x)\right) = \mathsf{Odd}_{G'}\left(g(b) \cup \{a'\}\right) = \mathsf{Odd}_{G'}\left(g(b)\ \Delta\ \{a'\}\right) = \mathsf{Odd}_{G}\left(g(b)\right)\ \Delta\ \{x, b\}$$

since $a, a' \notin g(b)$, and

$$\mathsf{Odd}_{G'}\left(g'(a')\right) = \mathsf{Odd}_{G'}\left(g(a)\right) = \mathsf{Odd}_{G}\left(g(a)\right)\ \Delta\ \{a, a'\}$$

since $b \in g(a)$. Therefore $x \in \mathsf{Odd}_{G'}\left(g'(x)\right)$ and $a' \in \mathsf{Odd}_{G'}\left(g'(a')\right)$ as desired, and furthermore the correction sets for the new vertices are focused. Take $\prec'$ to be the transitive closure of

$$\prec \cup \{(a, x), (x, a'), (a', b)\} \cup \{(w, x) | w \prec b \wedge w \not\prec a\} \cup \{(a', w) | a \prec w \wedge b \not\prec w\},$$

then $(g', \prec')$ is a focused gflow for Γ': Firstly, the relation $\prec'$ is a strict partial order.

As $\prec'$ is a transitive closure, it is transitive by construction. Is then remains to show that it is asymmetric or irreflexive. We shall assume that there exist $v, w \in V'$ satisfying $v \prec' w$ and $w \prec' v$ for a contradiction. We then have the following cases to consider.

1. $v = x$ and $w = a'$ (or equivalently with $v = a'$ and $w = x$)
2. $v \in \{x, a'\}$ and $w \in V' \setminus \{x, a'\}$ (or equivalently $v \in V' \setminus \{x, a'\}$ and $w \in \{x, a'\}$).
3. $v \in V' \setminus \{x, a'\}$ and $w \in V' \setminus \{x, a'\}$

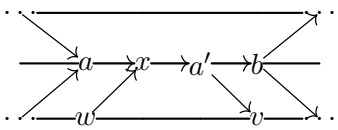
In the first case, if $v = x$ and $w = a'$ then $w \not\prec' v$ by the definition of $\prec'$.

For the second case where we have $v = x$ and $w \in V$, $w \prec' v$ implies that $w = a$, $w \prec a$ or $w \prec b$ and $w \not\prec a$. $v \prec' w$ implies that $w = b$, $b \prec w$ or $a \prec w$ and $b \not\prec w$, none of which are compatible with the conditions implied by $w \prec' v$ as $\prec$ is a strict partial order. Therefore the second case cannot hold.

For the final case, if we have $v, w \notin \{x, a'\}$ then the condition $w \prec' v$ implies that either $w \prec v$ or $w \prec' x \prec' a' \prec' v$. If $w \prec v$ then as $\prec$ is a strict partial order, we could not have $v \prec' w$ as this would give both $w \prec v$ and $v \prec w$.

If $w \prec' x \prec' a' \prec' v$ then we either have that $w = a$, $w \prec a$ or $w \prec b$ and $w \not\prec a$. If $w = a$ or $w \prec a$ then $w \prec' v$ implies $w \prec v$ and $v \prec' w$ implies $v \prec w$, contradicting the fact that $\prec$ is a strict partial order.

We then must have that $w \prec b$ and $w \not\prec a$. From $w \prec' x \prec' a' \prec' v$ we obtain $v = b$, $b \prec v$ or $a \prec v$ and $b \not\prec v$. In the first two cases, $w \prec v$ and thus these cases are trivial as before. We therefore have that $w \prec' v$ implies that $a \prec v$ and $b \not\prec v$ as well as $w \prec b$ and $w \not\prec a$. Visually, we have the following (with arrows representing the order).



Then, $v \prec' w$ then gives us that $a \prec v \prec w \prec b$, contradicting the assumption that $\nexists u$

such that $a \prec u \prec b$. Therefore we have that $w \prec' v$ implies $v \not\prec' w$ - as $\prec'$ is transitive, it is therefore a strict partial order.

To show that the gflow conditions are satisfied by $(g', \prec')$, it suffices to consider that the modified correction sets are compatible with the new partial order:

- For any $v \in V$, we have $a' \in g'(v)$ only if $a \in g(v)$. The latter implies $v \prec a$ and thus $v \prec' a \prec' a'$.
- For any $v \in V$, we have $x \in g'(v)$ only if $b \in g(v)$. The latter implies $v \prec b$ and thus by assumption $v \prec a$. Then, as in the previous case, $v \prec' a \prec' x$.
- If $w \in g'(x)$, then either $w = a'$ or $w \in g(b)$. The former is straightforward as $x \prec' a'$ by definition. For the latter, we have $b \prec w$ since $(g, \prec)$ is a gflow, and thus $x \prec' b \prec' w$.
- If $w \in g'(a')$, then $w \in g(a)$. This implies $a \prec w$ since $(g, \prec)$ is a gflow, and furthermore $b \prec w$ by assumption. Thus, $a' \prec' b \prec' w$.

All of the other gflow conditions are satisfied as $(g, \prec)$ is a gflow for Γ.

For the other implication, suppose Γ' has gflow. Then it has a focused gflow $(g', \prec')$. Every focused gflow with $|I| = |O|$ is reversible, thus as x and a' are XY-measured and have two neighbours, both must be in the correction set of one of their neighbours and have the other neighbour in their own correction set so that they are in their own odd neighbourhoods in both g' and its reversed version.

In particular, we have the following two statements.

$$(a \in g'(x) \wedge x \in g'(a')) \vee (a' \in g'(x) \wedge x \in g'(a))$$

$$(x \in g'(a') \wedge a' \in g'(b)) \vee (a' \in g'(x) \wedge b \in g'(a'))$$

As we cannot have both $x \in g'(a')$ and $a' \in g'(x)$ as this would mean $x \prec' a'$ and

$a' \prec' x$, we get that either $x \in g'(a)$, $a' \in g'(x)$ and $b \in g'(a')$ or $a \in g'(x)$, $x \in g'(a')$ and $a' \in g'(b)$. We shall assume that we have the former case where $a \prec' x \prec' a' \prec' b$ without loss of generality. Note that we must have $g'(a') = \{b\} \bigtriangleup \bigtriangleup_{w \in N_{G'}(b)} g'(w)$, $g'(x) = \{a'\} \bigtriangleup g'(b)$ and $g'(a) = \{x\} \bigtriangleup g'(a')$ for $(g', \prec')$ to be focused. We then also have that $Odd_{G'}(g'(a)) \setminus \{a\} = Odd_{G'}(g'(a')) \setminus \{a'\}$ and $Odd_{G'}(g'(x)) \setminus \{x\} = Odd_{G'}(g'(b)) \setminus \{b\}$.

Suppose there exists some vertex $v \notin \{x, a'\}$ satisfying $a \prec' v \prec' b$. We have the three following cases.

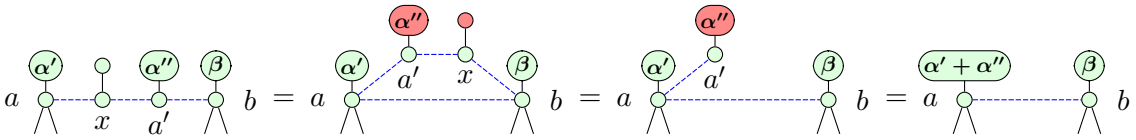
1. $a \prec' v \prec' x \prec' a' \prec' b$
2. $a \prec' x \prec' v \prec' a' \prec' b$
3. $a \prec' x \prec' a' \prec' v \prec' b$

As $(g', \prec')$ is focused, $\prec'$ only depends on g' - thus in the first case, if v is XY-measured we have that $v \in g'(a)$ (similarly $v \in Odd_{G'}(g'(a))$ if v is XZ or YZ measured). As $g'(a) = \{x\} \bigtriangleup g'(a')$ and $Odd_{G'}(g'(a)) \setminus \{a\} = Odd_{G'}(g'(a')) \setminus \{a'\}$, we then have that $v \in g'(a)$ and $v \notin \{a, x, x', b\}$ implies $v \in g'(a')$ (and similarly for odd neighbourhoods) and thus $a' \prec' v$, contradicting the statement that $v \prec' a'$.

In the second case, we have $v \in g'(x)$ if v is XY measured (similarly $v \in Odd'_G g'(x)$ if v is XZ or YZ measured). As $g'(x) = \{a'\} \bigtriangleup g'(b)$ and $Odd_{G'}(g'(x)) \setminus \{x\} = Odd_{G'}(g'(b)) \setminus \{b\}$, we can deduce that $v \in g'(x)$ and $v \notin \{a, x, x', b\}$ implies $v \in g'(b)$ (and similarly for odd neighbourhoods) and thus $b \prec' v$, contradicting the statement that $v \prec' b$

For the final case, $v \prec' b$ implies $b \in g'(v)$ if b is XY-measured. Then as g' is focused, we must have $x \in g'(v)$ so that $a' \notin Odd'_G(g'(v))$, implying that $v \prec' x$ which contradicts the statement that $x \prec' v$.

We shall now use the fact that v cannot be between a and b in the partial order of the post neighbour unfusion diagram to show that it also cannot be between a and b in the pre-neighbour unfusion diagram. Consider the following sequence of rewrites corresponding to the inverse of neighbour unfusion:



where we first pivot along the edge (x, a'), then apply Z-deletion to a' and finally apply the phase gadget identity rule of [43] to add the phase of a' to that of a. Each of these rules preserves the existence of gflow, thus the inverse of neighbour unfusion preserves the existence of gflow. Moreover, if $a \prec' b$ and $\nexists v \in V' \setminus \{x, a'\}$ such that $a \prec' v \prec' b$ then we have that $a \prec b$ and $\nexists v \in V$ such that $a \prec v \prec b$.

Therefore, if the measurement pattern after neighbour unfusion has gflow, then the original pattern has a focused gflow where b is in the correction set of a, or a focused gflow where a is in the correction set of b, and no vertices are measured between a and b.

□

We now have a full classification for when neighbour unfusion preserves the existence of gflow. This should allow for a speed-up in the runtime of the two-qubit gate count reduction algorithm of [60] as we now can easily find exactly all cases where we can apply neighbour unfusion while maintaining the existence of gflow.

Chapter 7

Conclusions and future work

In this thesis, I have shown that there are multiple Pauli-flow-preserving rewrite rules that have not been previously considered, with applications in many areas of quantum computing. Previously, only rules which reduce the number of spiders had been considered, which is natural as reducing the number of spiders is one of the goals in optimization – however, I have shown that rules which increase the number of spiders can be powerful and are worth investigating further.

I have shown that using the Z-deletion and local complementation rules along with our newly introduced Z-insertion rule, we are able to rewrite any two stabilizer MBQC-form diagrams into one another. In the process of doing this we have introduced the first unique normal form for stabilizer ZX-diagrams, a result has since been generalized to the stabilizer qupit ZX-calculus for all prime dimensions [55].

Finally, I have also proved that neighbour unfusion, a specific case of our "splitting a vertex" rule, preserves the existence of Pauli flow in all situations but only preserves gflow in specific cases, of which I have found a sufficient and necessary condition for. On top of this, I have introduced a weaker necessary condition which is easy to check for and can quickly tell us when neighbour unfusion will not preserve the existence of gflow.

There are several avenues of future research which arise from this work.

7.1 Further work: More flow-preserving rewrite rules and completeness

Flow-preserving rewrite rules are clearly very powerful and have been used long before the work in this thesis [27]. It is therefore interesting to consider what other rewrite rules we can find that preserve the existence of Pauli flow and in which cases these rules also preserve the existence of gflow.

While I have shown that the MBQC-form ZX-calculus is complete for the stabilizer fragment, this fragment is classically simulable. If someone was able to find a flow-preserving rewrite rule set for a more general (ideally universal) fragment of quantum computing such as the Clifford+T or the universal ZX-calculus, this would have a big impact on the field of circuit optimization as we would then be able to do all of our optimization using flow-preserving rewrite rules and then know that we would still easily be able to get a circuit back out at the end without undoing a lot of the optimization, which is a problem with current ZX-based circuit optimization techniques.

Additionally, the unique stabilizer normal form I have introduced has been generalized to the stabilizer qupit ZX-calculus [55] as well as the qufinite ZXW-calculus [63]. If this normal form could be generalized to a universal fragment of the ZX-calculus in some way then this would be incredibly powerful both as a theoretical result and as part of a circuit optimization process.

7.2 Further work: Applications of our flow-preserving rewrite rules

The work I have done on flow-preserving rewrite rules has been theoretical, yet has already found applications in blind quantum computing [17] and in circuit optimization [60]. It is therefore interesting to consider which other areas of quantum computing research these rules may find applications in.

Circuit optimization strategies - both those using the ZX-calculus and using other, more traditional methods - regularly focus on the stabilizer fragment, for example by removing as many stabilizer spiders as possible as in [27]. Since we have now found a unique normal form for stabilizer ZX-diagrams, there may be applications of our complete rewrite ruleset to circuit optimization which are worth further consideration.

A normal form similar to the unique stabilizer normal form we introduced has recently found applications in quantum compilation [42] and so it is also interesting to consider what else we can learn from the ability to find a unique normal form for stabilizer MBQC-form diagrams.

Bibliography

[1] Samson Abramsky & Bob Coecke (2007): *A categorical semantics of quantum protocols*. Available at `https://arxiv.org/abs/quant-ph/0402130`.

[2] Matthew Amy (2019): *Towards Large-scale Functional Verification of Universal Quantum Circuits*. *Electronic Proceedings in Theoretical Computer Science* 287, p. 1–21, doi:10.4204/eptcs.287.1. Available at `http://dx.doi.org/10.4204/EPTCS.287.1`.

[3] Matthew Amy, Parsiad Azimzadeh & Michele Mosca (2018): *On the controlled-NOT complexity of controlled-NOT–phase circuits*. *Quantum Science and Technology* 4(1), p. 015002, doi:10.1088/2058-9565/aad8ca. Available at `https://dx.doi.org/10.1088/2058-9565/aad8ca`.

[4] Matthew Amy, Dmitri Maslov & Michele Mosca (2014): *Polynomial-Time T-Depth Optimization of Clifford+T Circuits Via Matroid Partitioning*. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33(10), pp. 1476–1489, doi:10.1109/TCAD.2014.2341953.

[5] Matthew Amy & Michele Mosca (2019): *T-Count Optimization and Reed–Muller Codes*. *IEEE Transactions on Information Theory* 65(8), pp. 4771–4784, doi:10.1109/TIT.2019.2906374.

[6] Miriam Backens (2014): *The ZX-calculus is complete for stabilizer quantum mechanics*. *New Journal of Physics* 16(9), p. 093021, doi:10.1088/1367-2630/16/9/093021.

[7] Miriam Backens (2015): *Making the stabilizer ZX-calculus complete for scalars. Electronic Proceedings in Theoretical Computer Science* 195, p. 17–32, doi:10.4204/eptcs.195.2.

[8] Miriam Backens & Aleks Kissinger (2019): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity. Electronic Proceedings in Theoretical Computer Science* 287, p. 23–42, doi:10.4204/eptcs.287.2. Available at `http://dx.doi.org/10.4204/EPTCS.287.2`.

[9] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2021): *There and back again: A circuit extraction tale. Quantum* 5, p. 421, doi:10.22331/q-2021-03-25-421.

[10] Miriam Backens, Simon Perdrix & Quanlong Wang (2017): *A Simplified Stabilizer ZX-calculus. Electronic Proceedings in Theoretical Computer Science* 236, p. 1–20, doi:10.4204/eptcs.236.1. Available at `http://dx.doi.org/10.4204/EPTCS.236.1`.

[11] Niel de Beaudrap, Aleks Kissinger & John van de Wetering (2022): *Circuit Extraction for ZX-Diagrams Can Be #P-Hard.* In Mikołaj Bojańczyk, Emanuela Merelli & David P. Woodruff, editors: *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, *Leibniz International Proceedings in Informatics (LIPIcs)* 229, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 119:1–119:19, doi:10.4230/LIPIcs.ICALP.2022.119.

[12] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek & Alán Aspuru-Guzik (2022): *Noisy intermediate-scale quantum algorithms. Reviews of Modern Physics* 94(1), doi:10.1103/revmodphys.94.015004. Available at `http://dx.doi.org/10.1103/RevModPhys.94.015004`.

[13] André Bouchet (1987): *Graphic Presentations of Isotropic Systems. Journal of Combinatorial Theory, Series B* 45(1), p. 58–76, doi:10.1016/0095-8956(88)90055-X.

[14] P.O. Boykin, T. Mor, M. Pulver, V. Roychowdhury & F. Vatan (1999): *On universal and fault-tolerant quantum computing: a novel basis and a new constructive proof of universality for Shor's basis.* In: *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, pp. 486–494, doi:10.1109/SFFCS.1999.814621.

[15] Daniel E Browne, Elham Kashefi, Mehdi Mhalla & Simon Perdrix (2007): *Generalized flow and determinism in measurement-based quantum computation.* *New Journal of Physics* 9(8), p. 250–250, doi:10.1088/1367-2630/9/8/250.

[16] Jin-Yi Cai, Pinyan Lu & Mingji Xia (2018): *Dichotomy for Real Holant^c Problems.* In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics, pp. 1802–1821, doi:10.1137/1.9781611975031.118.

[17] Shuxiang Cao (2023): *Multi-agent blind quantum computation without universal cluster states.* *New Journal of Physics* 25(10), p. 103028, doi:10.1088/1367-2630/acfab6. Available at `http://dx.doi.org/10.1088/1367-2630/acfab6`.

[18] Titouan Carette (2021): *When Only Topology Matters.* Available at `https://arxiv.org/abs/2102.03178`.

[19] Francois Charton, Alexandre Krajenbrink, Konstantinos Meichanetzidis & Richie Yeung (2023): *Teaching small transformers to rewrite ZX diagrams.* In: *The 3rd Workshop on Mathematical Reasoning and AI at NeurIPS'23.* Available at `https://openreview.net/forum?id=btQ7Bt1NLF`.

[20] Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix & Benoît Valiron (2023): *A Complete Equational Theory for Quantum Circuits.* In: *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, IEEE, doi:10.1109/lics56636.2023.10175801. Available at `http://dx.doi.org/10.1109/LICS56636.2023.10175801`.

[21] Bob Coecke & Ross Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics.* *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-

2630/13/4/043016. Available at http://dx.doi.org/10.1088/1367-2630/13/4/043016.

[22] Bob Coecke & Aleks Kissinger (2017): *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, doi:10.1017/9781316219317.

[23] Alexander Cowtan & Simon Burton (2024): *CSS code surgery as a universal construction*. *Quantum* 8, p. 1344, doi:10.22331/q-2024-05-14-1344. Available at http://dx.doi.org/10.22331/q-2024-05-14-1344.

[24] Vincent Danos & Elham Kashefi (2006): *Determinism in the one-way model*. *Phys. Rev. A* 74, p. 052310, doi:10.1103/PhysRevA.74.052310.

[25] Vincent Danos, Elham Kashefi & Prakash Panangaden (2005): *Parsimonious and robust realizations of unitary maps in the one-way model*. *Physical Review A* 72(6), p. 064301, doi:10.1103/PhysRevA.72.064301.

[26] Jeroen Dehaene & Bart De Moor (2003): *Clifford group, stabilizer states, and linear and quadratic operations over GF(2)*. *Phys. Rev. A* 68, p. 042318, doi:10.1103/PhysRevA.68.042318.

[27] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. *Quantum* 4, p. 279, doi:10.22331/q-2020-06-04-279.

[28] Ross Duncan & Simon Perdrix (2009): *Graph States and the Necessity of Euler Decomposition*. In Klaus Ambos-Spies, Benedikt Löwe & Wolfgang Merkle, editors: *Mathematical Theory and Computational Practice*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 167–177, doi:10.1007/978-3-642-03073-4_18.

[29] Ross Duncan & Simon Perdrix (2010): *Rewriting Measurement-Based Quantum Computations with Generalised Flow*. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide & Paul G. Spirakis, editors: *Automata,*

Languages and Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 285–296.

[30] Matthew B. Elliott, Bryan Eastin & Carlton M. Caves (2008): *Graphical description of the action of Clifford operators on stabilizer states. Phys. Rev. A* 77, p. 042307, doi:10.1103/PhysRevA.77.042307.

[31] Joseph F. Fitzsimons (2017): *Private quantum computation: an introduction to blind quantum computing and related protocols. npj Quantum Inf.* 3(1), p. 23, doi:10.1038/s41534-017-0025-3.

[32] Thomas Fösel, Murphy Yuezhen Niu, Florian Marquardt & Li Li (2021): *Quantum circuit optimization with deep reinforcement learning.* Available at `https://arxiv.org/abs/2103.07585`.

[33] Lov K. Grover (1996): *A fast quantum mechanical algorithm for database search.* Available at `https://arxiv.org/abs/quant-ph/9605043`.

[34] Amar Hadzihasanovic (2015): *A Diagrammatic Axiomatisation for Qubit Entanglement.* Available at `https://arxiv.org/abs/1501.07082`.

[35] Amar Hadzihasanovic, Kang eng Ng & Quanlong Wang (2018): *Two complete axiomatisations of pure-state qubit quantum computing.* In: *LICS '18: 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, doi:10.1145/3209108.3209128.

[36] Christiaan Heunen & Jamie Vicary (2019): *Categories for quantum theory: an introduction.* Oxford University Press, United Kingdom.

[37] Luke E Heyfron & Earl T Campbell (2018): *An efficient quantum compiler that reduces T count. Quantum Science and Technology* 4(1), p. 015004, doi:10.1088/2058-9565/aad604. Available at `https://dx.doi.org/10.1088/2058-9565/aad604`.

[38] Alexander Tianlin Hu & Andrey Boris Khesin (2021): *Improved Graph Formalism for Quantum Circuit Simulation*, doi:10.48550/ARXIV.2109.10210.

[39] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *A Complete Axiomatisation of the ZX-Calculus for Clifford+T Quantum Mechanics.* In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, Association for Computing Machinery, New York, NY, USA, p. 559–568, doi:10.1145/3209108.3209131.

[40] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *Diagrammatic Reasoning beyond Clifford+T Quantum Mechanics.* In: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, Association for Computing Machinery, New York, NY, USA, p. 569–578, doi:10.1145/3209108.3209139.

[41] Andrey Boris Khesin & Alexander Li (2024): *Equivalence Classes of Quantum Error-Correcting Codes.* Available at `https://arxiv.org/abs/2406.12083`.

[42] Andrey Boris Khesin, Jonathan Z. Lu & Peter W. Shor (2024): *Graphical quantum Clifford-encoder compilers from the ZX calculus.* Available at `https://arxiv.org/abs/2301.02356`.

[43] Aleks Kissinger & John van de Wetering (2020): *Reducing the number of non-Clifford gates in quantum circuits.* *Physical Review A* 102(2), p. 022406, doi:10.1103/PhysRevA.102.022406.

[44] Aleks Kissinger & John van de Wetering (2022): *Simulating quantum circuits with ZX-calculus reduced stabiliser decompositions.* *Quantum Science and Technology* 7(4), p. 044001, doi:10.1088/2058-9565/ac5d20. Available at `http://dx.doi.org/10.1088/2058-9565/ac5d20`.

[45] Atul Mantri, Tommaso F. Demarie, Nicolas C. Menicucci & Joseph F. Fitzsimons (2017): *Flow Ambiguity: A Path Towards Classically Driven Blind Quantum Computation.* *Physical Review X* 7(3), doi:10.1103/physrevx.7.031004. Available at `http://dx.doi.org/10.1103/PhysRevX.7.031004`.

[46] Tommy McElvanney & Miriam Backens (2023): *Complete Flow-Preserving Rewrite Rules for MBQC Patterns with Pauli Measurements. Electronic Proceedings in Theoretical Computer Science* 394, p. 66–82, doi:10.4204/eptcs.394.5. Available at `http://dx.doi.org/10.4204/EPTCS.394.5`.

[47] Tommy McElvanney & Miriam Backens (2023): *Flow-preserving ZX-calculus Rewrite Rules for Optimisation and Obfuscation. Electronic Proceedings in Theoretical Computer Science* 384, p. 203–219, doi:10.4204/eptcs.384.12. Available at `http://dx.doi.org/10.4204/EPTCS.384.12`.

[48] Mehdi Mhalla, Mio Murao, Simon Perdrix, Masato Someya & Peter S Turner (2011): *Which graph states are useful for quantum information processing?* In: *Conference on Quantum Computation, Communication, and Cryptography*, Springer, pp. 174–187, doi:10.1007/978-3-642-54429-3_12.

[49] Mehdi Mhalla, Simon Perdrix & Luc Sanselme (2022): *Characterising Determinism in MBQCs involving Pauli Measurements.*

[50] Jisho Miyazaki, Michal Hajdušek & Mio Murao (2015): *Analysis of the trade-off between spatial and temporal resources for measurement-based quantum computation. Physical Review A* 91(5), p. 052302, doi:10.1103/PhysRevA.91.052302.

[51] Maarten Van den Nest, Jeroen Dehaene & Bart De Moor (2004): *Graphical description of the action of local Clifford transformations on graph states. Phys. Rev. A* 69, p. 022316, doi:10.1103/PhysRevA.69.022316.

[52] Kang Feng Ng & Quanlong Wang (2017): *A universal completion of the ZX-calculus.*

[53] Michael A. Nielsen & Isaac L. Chuang (2010): *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press.

[54] Elijah Pelofske, Andreas Bärtschi & Stephan Eidenbenz (2022): *Quantum Volume in Practice: What Users Can Expect From NISQ Devices. IEEE Transactions on*

Quantum Engineering 3, p. 1–19, doi:10.1109/tqe.2022.3184764. Available at http://dx.doi.org/10.1109/TQE.2022.3184764.

[55] Boldizsár Poór (2022): *A unique normal form for prime-dimensional qudit Clifford ZX-calculus*. Master's thesis, University of Oxford.

[56] Department for Science Innovation, Technology & The Rt Hon Peter Kyle MP (2024): *Over £100 million boost to quantum hubs to develop life-saving blood tests and resilient security systems*. https://www.gov.uk/government/news/over-100-million-boost-to-quantum-hubs-to-develop-life-saving-blood-tests-and- Accessed: 2024-09-06.

[57] Peter W. Shor (1997): *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. *SIAM Journal on Computing* 26(5), p. 1484–1509, doi:10.1137/s0097539795293172. Available at http://dx.doi.org/10.1137/S0097539795293172.

[58] Will Simmons (2021): *Relating Measurement Patterns to Circuits via Pauli Flow*. In Chris Heunen & Miriam Backens, editors: Proceedings 18th International Conference on Quantum Physics and Logic, Gdansk, Poland, and online, 7-11 June 2021, *Electronic Proceedings in Theoretical Computer Science* 343, Open Publishing Association, pp. 50–101, doi:10.4204/EPTCS.343.4.

[59] Will Simmons (2023): *Personal communication*.

[60] Korbinian Staudacher, Tobias Guggemos, Sophia Grundner-Culemann & Wolfgang Gehrke (2022): *Reducing 2-qubit gate count for ZX-calculus based quantum circuit optimization*. Available at https://elib.dlr.de/188470/. To appear in Proceedings QPL '22.

[61] Matthew Sutcliffe & Aleks Kissinger (2024): *Fast classical simulation of quantum circuits via parametric rewriting in the ZX-calculus*. Available at https://arxiv.org/abs/2403.06777.

[62] Maarten Van Den Nest (2010): *Classical Simulation of Quantum Computation, the Gottesman-Knill Theorem, and Slightly Beyond.* Quantum Info. Comput. 10(3), p. 258–271, doi:10.5555/2011350.2011356.

[63] Quanlong Wang & Boldizsár Poór (2023): *Completeness of qufinite ZXW calculus, a graphical language for mixed-dimensional quantum computing.* arXiv preprint arXiv:*2309.13014*.

[64] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*, doi:10.48550/ARXIV.2012.13966.

[65] John van de Wetering (2022): *Personal communication.*