Theory of measurement-based quantum computing

by

Jonathan Robert Niel de Beaudrap

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Combinatorics & Optimization

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In the study of quantum computation, data is represented in terms of linear operators which form a generalized model of probability, and computations are most commonly described as products of unitary transformations, which are the transformations which preserve the quality of the data in a precise sense. This naturally leads to *unitary circuit models*, which are models of computation in which unitary operators are expressed as a product of "elementary" unitary transformations. However, unitary transformations can also be effected as a composition of operations which are not all unitary themselves: the *one-way measurement model* is one such model of quantum computation.

In this thesis, we examine the relationship between representations of unitary operators and decompositions of those operators in the one-way measurement model. In particular, we consider different circumstances under which a procedure in the one-way measurement model can be described as simulating a unitary circuit, by considering the combinatorial structures which are common to unitary circuits and two simple constructions of one-way based procedures. These structures lead to a characterization of the one-way measurement patterns which arise from these constructions, which can then be related to efficiently testable properties of graphs. We also consider how these characterizations provide automatic techniques for obtaining complete measurement-based decompositions, from unitary transformations which are specified by operator expressions bearing a formal resemblance to path integrals. These techniques are presented as a possible means to devise new algorithms in the one-way measurement model, independently of algorithms in the unitary circuit model.

Acknowledgements

Many people have helped to contribute to the writing of this thesis. The following deserve special recognition:

Elham Kashefi, for introducing me to the one-way measurement model and to the questions about it which she was interested in; for the many stimulating discussions in Waterloo, Oxford, Grenoble, and Edinburgh; and for being a patient hostess.

Robert Raussendorf, for the handful of discussions in Waterloo that we had about the model which he helped to define. His feedback helped me to better understand the differences in perspectives about how to present the problems which Elham and I were interested in, at a time when I was unsure of how to communicate them to the wider community.

Michele Mosca and Ashwin Nayak, for being very liberal supervisors, and highly supportive of international voyages to conferences and collaborators alike.

Anne Broadbent and Joe Fitsimons, for insights which helped to enrich my perspective of flows and measurement-based quantum computation.

Martin Pei for being, quite simply, an excellent collaborator.

John Watrous and Daniel Gottesman, for being the instructors wose provided me with (more than anyone else) the *conceptual* framework necessary to study and describe the subject of this thesis.

Marg Feeney, for being very patient in reminding me of deadlines and requirements, and helpful in answering my questions about administrivia; she is the most excellent administrative staff member that I've ever encountered.

Finally, I'd like to thank Donny Cheung, Dave Evans, Alma Juarez, Cory Kasper, Ben Korvermaker, Mike LaCroix, Eva Μιχαλ, Judy Nowak, Mike Patterson, Carlos Pérez-Delgado, Joel Reardon, Bill Rosgen, Lana Sheridan, Craig Sloss, Devin Smith, Jason Skomorowski, Stephanus du Toit, Rob Warren, Greg Zaverucha, and Joanna Zimbiecka for helping to make my time as a student in Waterloo as pleasant as it was.

Contents

List of Figures

List of Algorithms

Preamble

and overview of the thesis

1 Introduction

Quantum computation developed as an attempt to respond to two questions:

1. How can we build a computing device to efficiently simulate physical systems?
2. What class of problems can be efficiently solved by physical systems, when we regard them as performing computations?

These questions have helped to spur research, continuing in the tradition of Landauer [88, 89], to study information and computations in physical terms.

Richard Feynman [62] put forward the thesis that a model of computation founded on quantum mechanics was necessary and fruitful, in order to more efficiently solve problems such as the simulation of quantum mechanical systems. Because of the interesting interplay between wave mechanics and discrete event detection at work in quantum mechanics, it seems to defy efficient simulation by "classical" models of computation, such as Turing machines [128]. Feynman proposed a model for a computer which exploited quantum mechanical effects as an antidote to this situation. In 1985, David Deutsch proposed a programmable model of universal computation [50], and posited that the difficulty of efficiently simulating quantum mechanics with classical models of computation was a sign of a fundamental computational advantage of models founded on quantum mechanical principles. This prompted the study of quantum/classical computational separations [18, 121], which eventually led to Peter Shor's discovery of efficient algorithms for the discrete logarithm and integer factoring problems in such a model [120]. As these problems are widely regarded as being "difficult" to solve by classical models of computation, the discovery of these algorithms sparked significant interest in the question of how a quantum computer might be physically realized.

Due to the link with Schrödinger evolution in quantum mechanics, quantum computation is most often (and usually conveniently) described in terms of *unitary transformations*, which are transformations of those configurations of the system which preserve the property of yielding point-mass probability distributions for a suitably chosen measurement operation. The "standard" model of quantum computation is therefore that of *unitary circuits* (described in Section 1.3), which expresses unitary transformations by products of other unitary transformations, reducing the evolution of quantum states

to some set of elementary operations which are considered to be likely to be physically realizable.

In 2001, Robert Raussendorf and Hans Briegel advanced a model [112] for a quantum computer by performing controlled measurement operations performed in a fixed state of a regular spin network. This gave rise to an alternative model of quantum computation, called the *one-way measurement model* (described in Chapter 2). This model contrasted sharply with the picture of quantum computation via unitary circuits in that measurement operations were used to effectively simulate unitary transformation on a fixed initial state, whereas measurements are often portrayed as being prototypical examples of non-unitary operations. While it is not exceedingly difficult (having the results of [112] in hand) to describe simple principles by which universal quantum computation may be driven by measurements on a spin network, this contrast of perspectives on quantum computation between transformation by unitaries, and transformation by measurements, makes it difficult to interpret measurement based procedures as "first class" descriptions of unitary transformations of quantum states.

In this thesis, we explore the relationship between the one-way measurement model and unitary transformations, with the aim of exploring, on the one hand, techniques to recognize one-way measurement procedures which represent unitary transformations and translate them into the circuit model, and on the other hand to identify when expressions for unitary transformations may be easily translated to a one-way measurement procedure. A secondary, underlying theme is the subject of combinatorial structures which unify different representations of unitary transformations: throughout the thesis, it is by identifying and recognizing such combinatorial structures that the task of translating between different representations of unitary transformations (such as unitary circuits and one-way measurement-based procedures) becomes tractable.

2 Structure of the thesis

This thesis contains five chapters, discounting the present one.

The first, introductory chapter, introduces the mathematical tools for describing quantum computation, including three individual models of importance:

1. *Unitary circuit models* — more properly speaking, this is a family of computational models, in which elementary unitary operations are composed to produce unitary operations of greater complexity. In practise, however, almost all models of unitary circuits considered are essentially equivalent to a "standard" choice of elementary gates, which we introduce here. We also define a non-standard, but closely related, unitary circuit model which will come to dominate the analysis of unitary circuits throughout the thesis; and a non-standard representation of unitary circuits which also plays an important role in the analysis of the later chapters.

2. *Classically controlled unitary circuits* — an elaboration of unitary circuit models by explicitly representing a classical control, we present these models essentially in order to concretely describe the way in which they are usually translated back into "fully quantum" unitary circuits.

3. *Clifford circuits* — a model of computation with restricted power, the limited operations of this model nevertheless provides useful analytical tools for describing transformations of a special class of states by measurements.

The second chapter introduces measurement-based computation, with emphasis on the *one-way measurement-based model* in particular. This model is the main subject of the thesis. In particular, we describe how universal quantum computation is effected in both the original *cluster-state* model, and the less restrictive model based on *open graphical encodings*, presenting two standard constructions for measurement-based computation to do so.

The third chapter is concerned with algorithms to recognize a certain class of one-way measurement-based procedures that perform unitary transformations, and to translate these procedures into unitary circuits. This is done by the graph-theoretical characterization and examination of a combinatorial structure arising out of the non-standard circuit representation introduced in Chapter 1. Using this combinatorial structure, we may reduce the problem of recognizing and translating such patterns to previously solved problems in graph theory. We then consider how the structure can be generalized to extend the scope of these recognition and translation algorithms. We also consider how combinatorial structures underlying one-way measurement-based computations may allow the certification that they perform unitary transformations, even in the absence of efficient algorithms to translate them to unitary circuits.

The fourth chapter takes advantage of the certifying structures described in the preceding chapter, to suggest techniques to automatically synthesize decompositions of unitary transformations (as unitary circuits, or as measurement-based procedures) from concise matrix expressions for such transformations. These techniques are presented as a stepping stone towards devising "native" idioms for solving problems in the one-way measurement model, which would contribute to the usefulness of the one-way measurement model as a useful model of computation, in addition to being a potentially useful proposal for implementation. At the same time, we consider the matrix expressions themselves, in particular noting their recurring role in quantum computation, as well as the possibility that they may naturally arise from representations of propagators of for physical systems as path integrals.

In the final chapter, we consider natural research directions, which essentially consist of the ways that the analysis of Chapter 4 may be most naturally extended.

3 Contributions presented in this thesis

The following is a list of the contributions made by myself, as presented in this thesis, to the understanding of one-way measurement-based computation, the context in which these results arose, and the subsequent work in the literature which followed. The description of where the results have been previously published, in each case, may be found in the pre-amble of the corresponding chapters.

Graph-theoretic descriptions of flows, and flow-finding algorithms

Sections 3.3 and 3.4 present work by myself (in the case of 3.3.5, with Martin Pei) on "flows", which may be interpreted as a certificate for the unitarity of a one-way measurement-based procedure [38], arising from a particular construction of measurement-based procedures (which is described in Section 2.2.5 as the *simplified DKP construction*).

The characterizations of Section 3.3 led to the first efficient algorithm for determining whether a geometry has a flow, albeit conditioned on the input and output subsystems of the corresponding measurement-based procedure being of the same size (corresponding to the case of a unitary bijection). These algorithms were made possible by an identification of a uniqueness result in 3.3.6. The result of 3.3.5 then helps to further bound the running time.

Subsequently, a faster and more general algorithm for finding flows unconditionally has been presented in [96]. This algorithm is likely to be optimal; bounds on its running time are also improved by the result of Section **??**. An examination of the algorithm of [96] imply a second graph-theoretic characterization of flows, which I briefly describe in Section 3.6.3 in relation to other work in Chapter 2.

Translation of one-way measurement procedures to unitary circuits

One application of flows as a certificate of unitarity is to obtain a unitary circuit which is equivalent [38], both in complexity (in a sense described in Section 3.1) and in the transformation which it performs. However, an explicit description of such an algorithm has not yet appeared in the literature, and involves a non-trivial amount of combinatorial analysis.

Section 3.5 presents an explicit, efficient algorithm to translate one-way measurement procedures with flow certificates to unitary circuits in this way. In particular, Section 3.5.1 characterize the dependency relations of measurement patterns with flows which have been fully standardized: this completes the work begin in [24] on the dependency relations in measurement patterns, which is pertinent to the topic of depth complexity. Section 3.5.2 then presents a new decomposition result for geometries with flows, which is pertinent to the faster flow-finding algorithm of [96]. These results allow for the verification of the correctness of the measurement dependencies in the measurement procedure, and then construction an appropriate circuit.

Extension of flows

In Sections 3.6.2, I present an extension of the "flow" certificate, which I call "extended flows". These extended flows may be taken as a certificate of unitarity for a different construction for one-way procedures (which is described in Section 2.2.6 as the *simplified RBB construction*). Exploiting the description given in Section 3.6.3 of the algorithm of [96], I present an efficient algorithm to detect extended flows, with essentially no change to the running time of [96]. I also present a sketch of how the efficient mapping to unitary circuits, which was presented for flows, may be similarly extended.

Measurement Pattern Interpolation

In Chapter 4, as part of joint work with Elham Kashefi, Vincent Danos, and Martin Roetteler, I present the definition of a quadratic form expansion. Similar objects have occurred previously in the literature; ours was the first to explicitly connect them with measurement-based computation, although a link through graph-states [116] was previously known. The contribution of Chapter 4 as a whole is to outline how algorithms may be naturally compiled for the one-way measurement model, and to propose that techniques along these lines may be applicable to the problem of approximating path integrals for physical propagators. There does not appear to be prior work on compiling for the one-way measurement model, except for the standard constructions via the circuit model.

The algorithms presented in Section 4.2.1 are essentially applications of a principle forwarded by Drs. Kashefi and Danos: that for any certificate that a measurement pattern performs a unitary transformation which is based on the stabilizer formalism, there exists an efficient algorithm to produce such a measurement pattern from a description of how it acts conditioned on one particular sequence of measurement results. The contribution of Section 4.2.1 consists of the analysis of applying this principle to each "instance", including making the necessary connections with the implementation of diagonal operations presented in [25], and with the concept of "fractional weight" flows.

Section 4.2.3 presents a refinement of the formulae for Clifford group operations presented in [48], which allows them to be presented as quadratic form expansions. Having found this connection, I present an algorithm to obtain a measurement pattern, essentially via an application of the stabilizer formalism [68]. This construction of one-way measurement procedures for the Clifford group elements is shown to be minimal with respect to the reduction techniques of [74], and is proven to run in time comparable to the algorithm of [2] for producing unitary circuits for Clifford group (and strictly faster than an algorithm using [2] as a subroutine to produce minimal one-way measurement procedures for the Clifford group).

4 Conventions

Notation and Pseudocode

Throughout the thesis, Dirac ("bra-ket") notation is used, along with conventional notations for Hilbert spaces: see *e.g.* [103] for an introduction to how it is used conventionally. However, owing to the regular discussion of operations in terms of the measurement-based model, transformations of quantum states are often represented as completely positive trace preserving linear superoperators (or *CPTP maps*, described on page 6) rather than as unitary matrices acting on column vectors. Other notations pertaining to linear operations are a minor variation on that of [133].

Special notations for operators, states, and other variables are usually introduced where they are used. However, it may be useful to note the following conventions, which may help to avoid possible ambiguities of interpretation:

- The constant i in italics always represents a square root of -1 in the complex numbers $\mathbb{C}$. In particular, it is never used as a variable to iterate over the terms of a sum, or the vertices of a path. (For the latter two purposes, variables such as $h, j, \ell, \ldots$ are most often used.)

- The constant e in an upright typeface always represents Euler's constant, the base of natural logarithms. (This should be contrasted with an italic e, with may represent a generic element of a finite ground set V.)

- The identity operator (or superoperator) for a physical system is always represented as $\mathbb{1}$. (This should be contrasted with the variable I, which always represents a subset of some larger ground-set V.)

- Superoperators are always presented as capital sans-serif letters such as E, or fraktur letters such as $\mathfrak{P}$. Operations in fraktur font in particular always represent procedures in a measurement-based model of computation. (This should be contrasted with capital italic letters, which usually represent unitary operations.)

- Vectors over $\mathbb{Z}_2$ (the integers modulo 2) are represented in a serif bold-face font, usually as a variable $\mathbf{x} \in \mathbb{Z}_2^V$ for some set V. (This should be contrasted with the sans-serif letters x, y, z, which are generic labels common to the notation for certain state vectors, measurements, and unitary transformations.)

- In the presentation of an algorithm, $a = b$ always denotes an equality comparison. (This should be contrasted with $a \leftarrow b$, which denotes an assignment.)

- In the presentation of an algorithm, the notation a++ represents an instruction to increment the integer variable a.

- In the presentation of an algorithm, a *procedure* takes as input *only* those variables which are explicitly specified in its invocation: *i.e.* it has no "side effects". (This should be contrasted with a *subroutine*, which may affect the values of data which are presented as being "global variables" available to the subroutine.)

Colours and Hyperlinking

In the figures, I have tried wherever appropriate to emphasize distinct elements using grayscale shading rather than colour. Where I found it important to provide additional emphasis with colour, I have tried also to supplement this with non-coloured visual cues (*e.g.* different styles of dashed lines).

This thesis was written using PDFLaTeX, and the electronic version is hyperlinked. Links are coloured, I hope unobtrusively, according to the nature of the reference:

red represents a link to another part of the main text, *e.g.* Theorem 1-3;

green represents a link to the bibliography, *e.g.* [74];

blue represents a link to a URL on the internet, *e.g.* [arXiv:0806.1972].

The entries of the Table of Contents, List of Figures, and List of Algorithms are also hyperlinked, although they are not coloured.

Chapter 1

Fundamentals

of quantum computation

Computation is the act of transforming a piece of data by physical processes, into a form which may be interpreted as representing *e.g.* the solution to a mathematical problem. Data may be stored as any reliably measurable (and manipulable) property of a physical system, in which case computations are most appropriately described in terms of operations on that system. Observations along these lines made by Landauer [88] spurred research into computation as a physical process, culminating in a series of papers by Benioff [12, 13, 14] proposing an implementation of Turing machines [128] along quantum mechanical principles.

In [62], Feynman considered the complementary question of whether or not quantum mechanical phenomena could be efficiently simulated by the existing models of computation, and concluded that a new type of computer was required in order to intrinsically model quantum mechanical phenomena. A model of a computation operation using quantum mechanical states as data was presented by Deutsch [50], and the theory of *quantum computation* was subsequently developed by Bernstein and Vazirani [18, 19] and Yao [135]; however, the mathematical tools for this description of information were largely developed by von Neumann [130].

Quantum computation is most often described in terms of *unitary circuits*, in which quantum data is represented as unit ℓ_2-norm *state vectors* over $\mathbb{C}^d$ for some $d \geqslant 2$, which generalize the notion of a point-mass probability distribution; transformations of the state are given by *unitary matrices*, which are the linear transformations which preserve the set of these vectors. In this model, measurements are postponed to the end of the computation, or (in the case of a computation which is to be used as a subroutine of a larger quantum computation) not performed at all, and so the description of measurements and post-measurement states is often simplified. However, it will prove essential for the discussion of the *one-way measurement-based model* to establish conventions and definitions for the more general description of quantum computation in terms of *density operators* which generalize both state vectors and arbitrary probability distributions, and the transformations which preserve this set of objects, which are *completely positive and trace-preserving superoperators* (or CPTP maps).

In this chapter, I will introduce the principal mathematical objects of quantum computation and standard results concerning universal quantum computation, abstracted away

from particular applications or implementations. I also introduce the conventions used in this thesis to describe *unitary circuits*, the principal language for describing quantum computation.

New results. Unsurprisingly, the vast majority of the work in this chapter is prior art; however, the section "Non-standard representations of unitary circuits" (pages 28 – 38) presents new results of an elementary nature, which fit most naturally in this chapter.

1.1 Quantum data, and quantum transformations

1.1.1 Qualitative remarks on the nature of quantum data

We begin with some remarks on quantum states, and the differences between them and classical probability, as a pre-amble to the technical definitions that follow.

The utility of probabilistic theories of evolution and computation is that they allow the transformation of data when one does not have perfect information about a system. A probability distribution (usually) represents some amount of information about a physical system, but (usually) not complete information; and transformations of such a distribution represents the information that can be obtained about that system under certain conditions. In particular, a probability distribution is a *model* of the state of a physical system, or of the distribution of inputs to a problem, and stochastic transformations represent a model of the evolution of a system, without committing to any answer as to whether a more precise model for the same physical system is possible. However, there is an implicit assumption that there is no penalty for coarseness of resolution of events compared to the possible states of the system in a traditional probabilistic model, except in the coarseness of the corresponding predictions.

Let us say that one set of events R is a *refinement* of another set S if any system which is determined with respect to R (*i.e.*, such that there is an event $r \in R$ which occurs with certainty), then the system is also determined with respect to S. Conceptually, there if a function from each $r \in R$ to some $s \in S$, which determines the events in S which occur conditioned on the events in R. The more refined a set of events is, the more precisely it allows description of a physical system; and a *maximally* refined set of events R for a system is one where any property which can be tested simultaneously with the events of R can in principle be *defined* in terms of events $r \in R$. Then, let us say that a system is in a *pure state* if, for some maximally refined set of events R for the system, the system can be described by the event $r \in R$. Classical probability theory is then a means of dealing with probabilities of events and stochastic processes on the premise that there may be in principle a unique set of distinguishable pure states, and that analysis of other distributions may be performed with respect to this unique set of states of complete knowledge.

By contrast, quantum mechanics presents us with physical systems in which there seems to be a mismatch between observable events (such as the outcomes of measurements in experiments) and the possible physical states of a system. It is possible to prepare physical systems such as the polarization of light signals, or the orientation of the spin of ions, in different ways which correspond to point-mass distributions of certain events,

but where these events cannot be simultaneously tested. This results in states which correspond to maximal information regarding certain physical properties, but seem only to be distinguishable from one another statistically, through multiple preparations.

What is more, the way in which such physical systems evolve demonstrate that the inability to resolve between different maximally refined sets of events in such systems is crucial: they may exhibit different evolutions depending on whether they interact with other systems, even when those interactions may only be used to test for some event. An example is with light filtered with a polarizing lens: vertically polarized light cannot pass through a horizontally polarized lens. However, there is some probability of the light passing through the horizontal filter if another filter (aligned diagonally to the horizontal axis) is placed between the source of polarized light and the horizontal filter. The intermediate filter causes an increase in the probability of light being detected after the original blocking filter, suggesting that the light has been transformed *en route*. However, unless the physical system is sensitive to the context in which the intermediate filter is being used, there is no reason why this filter should have a different effect than it would when used to either let diagonally polarized light pass through, or filter out light with the opposite diagonal polarization, in an attempt to measure in a different experiment which of the two diagonal polarizations it may have.

The example of diagonally versus horizontally or vertically polarized light illustrates that physical systems can be prepared in different pure states which can only be distinguished statistically, and that these states are sensitive to testing for properties of the system, and are generally changed by measurement of a property to a state consistent with having a definite value of that property. This suggested to Einstein, Podolsky, and Rosen [57] that quantum mechanics is incomplete, as they posited that there must be a single underlying maximally refined set of events, which contained complete information corresponding to what they referred to as *elements of reality* (*i.e.* point-mass functions over any testable set of events). This corresponds to the some of the current interpretation of quantum mechanics (*e.g.* the Bohmian interpretation [20]) involving *hidden variables* which we are prevented from testing precisely. However, the more common view has been that there do not exist properties which we are prevented for fundamental reasons from testing, and that there does not exist a *unique* maximally refined set of events, although there do exist multiple *incompatible* such maximally refined sets, in the sense that (by necessity) they cannot be tested for simultaneously, and that testing for one will disturb point-mass functions over the others.

The apparent absence of a maximally refined set of events for certain physical systems prompted the development of early quantum theory, in which tests for events (and the set of events which is being tested for) must be specified explicitly. This led to the specialized concept of *measurement* in quantum mechanics (and consequently in quantum computation as well) which will play a central role in this thesis.

1.1.2 Density operators

Quantum states may be described by *density operators*, and property testing on these states may be represented as *projections* on these operators [131]. We will illustrate this formalism of quantum states and measurements as an extension of classical probability theory, as follows.

Description of classical probability distributions

We describe probability in terms of an event space $\mathsf{E}(S)$ consisting of the set of functions $f : S \longrightarrow \mathbb{R}$, for some finite set of events S. The set of probability distributions $\mathsf{Prob}(S)$ in this space are the non-negative functions $p \in \mathsf{E}(S)$ of unit ℓ_1-norm,

$$\|p\|_1 \;=\; \sum_{s\in S} p(s) \;=\; 1\,. \tag{1.1}$$

Point-mass functions are given by characteristic functions $\chi_{\{s\}}$ for $s \in S$, where for $E \subseteq S$, χ_E is the characteristic function of E:

$$\chi_E(s) \;=\; \left\{ \begin{array}{ll} 1\,, & \text{if } s \in E \\ 0\,, & \text{otherwise} \end{array} \right\}\,; \tag{1.2}$$

we may then decompose $p \in \mathsf{Prob}(S)$ as a *convex combination*[1] of operators $\chi_{\{s\}}$ for $s \in S$,

$$p \;=\; \sum_{s\in S} p(s)\chi_{\{s\}} \;\;. \tag{1.3}$$

Events correspond to subsets $E \subseteq S$, and the probability of E corresponds to the sum of $p(s)$ for $s \in E$; equivalently, we can express it in terms of the adjoint functional $\chi_E^\dagger$, which maps characteristic functions χ_A to their "overlap" $|E \cap A|$:

$$\Pr_p(E) \;=\; \chi_E^\dagger\, p \;=\; \langle \chi_E, p\rangle \;\;, \tag{1.4}$$

where $\langle p, q\rangle$ is the usual inner product over Euclidean space, and $A^\dagger$ is the Hermitian adjoint for an operator A. Joint probability mass functions $g \in \mathsf{Prob}(S\times T)$ are maps from pairs of events $(s,t) \in S \times T$ to their probabilities $g(s,t)$, and can be described as convex combinations of the point-mass functions $\chi_{(s,t)}$; for independently distributed variables $p \in \mathsf{Prob}(S)$ and $q \in \mathsf{Prob}(T)$, we may describe such a joint distribution as a tensor product of the distributions p and q, by identifying distributions in $\mathsf{Prob}(S \times T)$ with the unit ℓ_1-norm functions in the event space $\mathsf{E}(S) \otimes \mathsf{E}(T)$. The probability distributions $\mathsf{Prob}(S\times T)$ then correspond to the non-negative, unit trace distributions of $\mathsf{E}(S)\otimes\mathsf{E}(T) \;\cong\; \mathsf{E}(S \times T)$.

Transformations U of probability distributions are given by the linear transformations which preserve ℓ_1-norm on $\mathsf{E}(S)$, *i.e.* by *stochastic transformations*. By virtue of their linearity, they may be characterized by conditional probabilities, *i.e.* how U transforms point-mass functions $\chi_{\{s\}}$. Independent transformations U on an event space $\mathsf{E}(R)$ and V on an event space $\mathsf{E}(T)$ then given by the tensor product $U \otimes V$; in particular, the marginal distribution over S of a probability distribution over $S \times T$ can be recovered by setting the map V to be the unique "stochastic" transformation $V : \mathsf{E}(T) \longrightarrow \mathbb{R}$, using the equivalence $\mathsf{E}(S) \otimes \mathbb{R} \;\cong\; \mathsf{E}(S \times \{1\}) \;\cong\; \mathsf{E}(S)$.

An isomorphic choice of mathematical structure would be to substitute real-valued diagonal matrices, and the Euclidean inner product with the Hilbert–Schmidt inner product,

$$\langle A, B\rangle_{\mathrm{Tr}} \;=\; \mathrm{Tr}(A^\dagger B)\,. \tag{1.5}$$

[1] A *convex combination* of a set X is a linear combination of the form $\sum_{x\in X} \lambda(x)x$, for some non-negative function $\lambda : X \longrightarrow \mathbb{R}$ with unit ℓ_1-norm.

For a maximally refined set of measurable events R on a system, let $\{|s\rangle\}_{s\in S}$ represent the standard basis for the Hilbert space $\mathcal{H}_S$. Then, we may use the following constructions for $\mathsf{E}(S)$ and $\mathsf{Prob}(S)$:

$$\Pi_{\{s\}} = |s\rangle\langle s| \text{ is a rank-1 projector } \big(\textit{c.f.}\text{ point-mass functions } \chi_{\{s\}}\big); \tag{1.6a}$$

$$\mathsf{E}(S) = \operatorname{span}_{\mathbb{R}}\big\{|s\rangle\langle s| \;\big|\; s\in S\big\} \;\; \big(\textit{c.f.}\text{ span of the functions } \chi_{\{s\}}\big); \tag{1.6b}$$

$$\begin{aligned}\mathsf{Prob}(S) = {} & \big\{p\in\mathsf{E}(S)\,\big|\,p \text{ non-negative, with unit trace}\big\}\\ & (\textit{c.f.}\text{ non-negative functions with unit } \ell_1\text{-norm});\end{aligned} \tag{1.6c}$$

$$\begin{aligned}\Pr_p(E) = {} & \operatorname{Tr}(\Pi_E\, p) \text{ for } p\in\mathsf{Prob}(S) \text{ and } E\subseteq S, \text{ where } \Pi_E = \sum_{s\in E}|s\rangle\langle s|\\ & (\textit{c.f.}\text{ Equation (1.4) above}).\end{aligned} \tag{1.6d}$$

In particular, distinct point-mass distributions are orthogonal with respect to the Hilbert–Schmidt inner product, and probabilities of events $E\subseteq R$ are given by the *operator functionals* Π_E^* characterized by mapping other projectors Π_S to the overlap $\Pi_E^*\Pi_A = \langle\Pi_E,\Pi_A\rangle_{\mathrm{Tr}} = |E\cap A|$.

Generalization to arbitrary quantum distributions

As we noted above, a single maximally refined set of events does not seem to suffice to describe the pure states accessible to quantum systems. For a quantum system which admits an event space $\mathsf{E}(S)$ as constructed above, we will generalize the set of pure states to all rank-1 projectors $|\psi\rangle\langle\psi|$: a maximally refined set of events is then a maximal collection

$$B = \Big\{|\psi_j\rangle\langle\psi_j|\Big\}_{j=1}^{|S|} \tag{1.7}$$

of rank-1 projections on $\mathcal{H}_S$, subject to these projectors being orthogonal with respect to the Hilbert–Schmidt inner product. We may then consider events of which this set is a refinement, which correspond to sets of projectors which all commute with the projections in S; and we may consider states of the system which are convex combinations of elements of S.

Considering all such sets of orthogonal rank-1 projectors prompts us to generalize the event space of the system from $\mathsf{E}(S)$ to the set of all matrices which can be decomposed as a real-valued linear combination of orthogonal rank-1 projectors: that is, the Hermitian operators (normal operators with real eigenvalues) $\mathsf{Herm}(S)$ over $\mathcal{H}_S$. Distributions describing states of the system are then as follows:

Definition 1-1. The set of *density operators* over S is the set of positive semidefinite Hermitian operators over $\mathcal{H}_S$ with unit trace,

$$\mathsf{D}(S) = \Big\{\rho\in\mathsf{Herm}(S)\;\Big|\;\rho\geqslant 0 \text{ and } \operatorname{Tr}(\rho)=1\Big\}\,. \tag{1.8}$$

As before, joint distributions are formed by tensor products of the event space: we have

$$\mathsf{Herm}(S\times T) = \mathsf{Herm}(S)\otimes\mathsf{Herm}(T)\,, \tag{1.9}$$

which can be easily proven by considering a basis for $\mathsf{Herm}(S\times T)$ over $\mathbb{R}$ consisting of the tensor product of two such bases for $\mathsf{Herm}(S)$ and $\mathsf{Herm}(T)$. For joint distributions $\tau\in\mathsf{D}(S\times T)$, we then take the non-negative, unit-trace elements of $\mathsf{Herm}(S\times T)$.

Transformations of density operators

The natural extension of a transformation between probability distributions is a *linear superoperator* — that is, a linear operation that itself acts on linear operations $\rho \in \mathsf{Herm}(R)$ — which in particular maps density operators to density operators. It follows that valid transformations of quantum data consist of *trace-preserving, positive* superoperators $\Phi : \mathsf{D}(\mathcal{H}_S) \longrightarrow \mathsf{D}(\mathcal{H}_{S'})$, *i.e.* that Φ maps operators of $\mathsf{L}(\mathcal{H}_S)$ to operators with the same trace, and positive semidefinite operators to other positive semidefinite operators. In order for such a superoperator Φ to be valid when applied to part of a joint distribution of two systems, we also require that $\Phi \otimes \mathrm{id}_{\mathsf{D}(T)}$ be a valid transformation (and therefore a positive operator) on $\mathsf{D}(S \otimes T)$ for any T: we say that Φ is *completely positive*.

Definition 1-2. A *transformation* of a set of density operators $\mathsf{D}(S)$ is a completely positive and trace preserving mapping (or *CPTP map*) $\Phi : \mathsf{D}(S) \longrightarrow \mathsf{D}(S')$ for some state-space $\mathsf{D}(S')$.

It is possible to show (see *e.g.* [133]) that for $\Phi : \mathsf{L}(\mathcal{H}_S) \longrightarrow \mathsf{L}(\mathcal{H}_{S'})$ completely positive and trace preserving, there exist linear operators $\{A_j\}_{j=1}^n$, for some $n \in \mathbb{N}$ and $A_j : \mathcal{H}_S \longrightarrow \mathcal{H}_{S'}$ non-zero, such that

$$\Phi(\rho) \;=\; \sum_{j=1}^{n} A_j \,\rho\, A_j^\dagger\,, \qquad \text{where} \qquad \sum_{j=1}^{n} A_j^\dagger A_j \;=\; \mathbb{1}_{\mathcal{H}_S}\,. \tag{1.10}$$

Such operators A_j are called *Kraus* operators, and are non-unique. For another set T, if $\Psi : \mathsf{L}(\mathcal{H}_T) \longrightarrow \mathsf{L}(\mathcal{H}_{T'})$ is another CPTP map given by

$$\Psi(\sigma) \;=\; \sum_{k=1}^{m} B_k \,\sigma\, B_k^\dagger\,, \tag{1.11}$$

we may form the transformation performed by independent transformations $\Phi : \mathsf{D}(S) \longrightarrow \mathsf{D}(S')$ and $\Psi : \mathsf{D}(T) \longrightarrow \mathsf{D}(T')$ on a *joint* distribution over $\mathsf{D}(S \times T)$ by taking tensor products of the Kraus operators: we write

$$(\Phi \otimes \Psi)(\tau) \;=\; \sum_{j=1}^{n}\sum_{k=1}^{m} (A_j \otimes B_k)\,\tau\,(A_j \otimes B_k)^\dagger \tag{1.12}$$

for $\tau \in \mathsf{L}(\mathcal{H}_{S\times T}) \longrightarrow \mathsf{L}(\mathcal{H}_{S\times T})$.

We may use this to determine the marginal distributions of joint density operators $\tau \in \mathsf{D}(S \times T)$. Consider the map $\mathrm{tr}_T : \mathsf{Herm}(T) \longrightarrow \mathbb{R}$ which maps every Hermitian operator over $\mathcal{H}_T$ to its trace, given *e.g.* by

$$\mathrm{tr}_T(\sigma) \;=\; \sum_{t\in T} \langle t|\,\sigma\,|t\rangle \;: \tag{1.13}$$

note that tr_T is the unique CPTP map from $\mathsf{Herm}(T)$ to $\mathbb{R}$. Then, we may obtain the marginal distribution τ_S over $\mathsf{D}(S)$ of a density operator $\tau \in \mathsf{D}(S \times T)$ by the *partial trace* over T, defined by

$$\Big(\mathrm{id}_{\mathsf{D}(S)} \otimes \mathrm{tr}_T\Big)(\tau) \;=\; \sum_{t\in T} \Big(\mathbb{1}_{\mathcal{H}_S} \otimes \langle t|\Big)\,\tau\,\Big(\mathbb{1}_{\mathcal{H}_S} \otimes |t\rangle\Big)\,. \tag{1.14}$$

We refer to the process of applying such a map as *tracing out* the system represented by the state space $\mathsf{D}(T)$; and we commonly write such a map by tr_T, leaving the identity on any other subsystems implicit in the description of the map.

Quantum and classical registers

Based on the preceding, we will make the following definitions:

Definition 1-3. A *quantum register* Q over a finite set Q is an abstract physical system with state-space $\mathsf{D}(Q) \leqslant \mathsf{Herm}(Q)$, consisting of unit trace operators $\rho \geqslant 0$ supported on a Hilbert space $\mathcal{H}_Q$. Similarly, a *classical register* C over a finite set C is an abstract physical system with state-space $\mathsf{Prob}(C) \leqslant \mathsf{D}(C)$; and where furthermore, we require that for any joint state $\rho \in \mathsf{D}(C \times R_1 \times \cdots \times R_n)$ of C together with any collection of registers $\{\mathsf{R}_1, \ldots, \mathsf{R}_n\}$, ρ is a convex combination of operators $\mathbf{p}_j \otimes \sigma_j$, for $\mathbf{p}_j \in \mathsf{Prob}(C)$ and $\sigma_j \in \mathsf{D}(R_1 \times \cdots \times R_n)$. For either a classical register with state-space $\mathsf{P}(R)$ or quantum register with state-space $\mathsf{D}(R)$, the *dimension* of the register is $|R|$.

The motivation for defining classical registers in addition to quantum registers is to abstractly represent physical systems which we cannot (either for fundamental reasons or practical limitations) put into general quantum states: examples include read-out devices in laboratory settings, and hand-written notes on paper, whose states do not exhibit obvious quantum behavior but which still may not be determinable in advance.

Notation. Let X and Y be convex sets in complex Hilbert spaces. In analogy to the linear extension $V \times W \hookrightarrow V \otimes W$ for vector spaces, we will let $X \otimes Y$ represent the *convex* extension of the set $X \times Y$, which consists of convex combinations of tensor products $\rho \otimes \sigma$ for $\rho \in X$ and $\sigma \in Y$.

In particular, for ρ a joint state of a classical register C together with any collection of registers $\{\mathsf{R}_1, \ldots, \mathsf{R}_n\}$ as in Definition 1-3 above, we have $\rho \in \mathsf{Prob}(C) \otimes \mathsf{D}(R_1 \times \cdots \times R_n)$.

Given the definition of a classical register above, the following Lemma allows us to interpret aggregates of classical registers as a single, compound classical register:

Lemma 1-1. *Let ρ be a joint state of a collection of quantum registers $\{\mathsf{Q}_1, \ldots, \mathsf{Q}_n\}$ and classical registers $\{\mathsf{C}_1, \ldots, \mathsf{C}_m\}$. Then we have*

$$\rho \in \mathsf{D}(Q_1 \times \cdots \times Q_n) \otimes \mathsf{Prob}(C_1 \times \cdots \times C_m), \tag{1.15}$$

where Q_j (respectively C_j) are the finite sets over which the registers Q_j (respectively C_j) are defined.

Proof —

We prove this by induction on m. If $m = 0$, there is nothing to prove. Otherwise, suppose the statement holds for all $m < M$ for some integer $M \geqslant 1$. As C_m is a

classical register, ρ is a convex combination of the form

$$\rho \;=\; \sum_{c_m \in C_m} \lambda_m(c_m)\; \sigma_{c_m} \otimes |c_m\rangle\langle c_m| \;, \tag{1.16}$$

where $\lambda_m \in \mathsf{Prob}(C_m)$ is the marginal distribution of ρ on C_m. For $\lambda_m(t) > 0$, the density operators σ_t are the states of the other registers conditioned on C_m being in the state $|t\rangle\langle t|$: we may verify this for events $E \in \mathsf{Herm}(Q_1 \times \cdots \times Q_n \times C_1 \times \cdots \times C_{m-1})$ by

$$\begin{aligned}
\Pr_\rho(E|\mathsf{C}_m = t) \;&=\; \frac{\Pr_\rho(E,t)}{\Pr_\rho(t)} \;=\; \frac{\mathrm{Tr}\left(\Big[E \otimes |t\rangle\langle t|\Big]\rho\right)}{\mathrm{Tr}\left(\,|t\rangle\langle t|\;\mathrm{tr}_R(\rho)\right)} \\
&=\; \frac{\mathrm{Tr}\left(\sum_{c_m} \lambda_m(c_m)\; E\,\sigma_{c_m} \otimes |t\rangle\langle t\,|c_m\rangle\langle c_m|\;\right)}{\mathrm{Tr}\left(\sum_{\mathbf{r}}\sum_{c_m} \lambda_m(c_m)\;\langle \mathbf{r}|\,\sigma_{c_m}\,|\mathbf{r}\rangle \otimes\; |t\rangle\langle t\,|c_m\rangle\langle c_m|\right)} \\
&=\; \frac{\lambda_m(t)\,\mathrm{Tr}(E\,\sigma_t)}{\lambda_m(t)} \;=\; \mathrm{Tr}(E\,\sigma_t)\;;
\end{aligned} \tag{1.17}$$

we may let σ_t be arbitrary for $\lambda_m(t) = 0$. Let $\tilde{C} = C_1 \times \cdots \times C_{m-1}$: by hypothesis, we can decompose each σ_{c_m} as a combination

$$\sigma_{c_m} \;=\; \sum_{\tilde{\mathbf{c}} \in \tilde{C}} \tilde{\lambda}_{c_m}(\tilde{\mathbf{c}})\; \tau_{(\tilde{\mathbf{c}},c_m)} \otimes |\tilde{\mathbf{c}}\rangle\langle\tilde{\mathbf{c}}| \tag{1.18}$$

again for some marginal distribution $\tilde{\lambda}_{c_m} \in \mathsf{Prob}(\tilde{C})$ depending on $c_m \in C_m$, and for conditional states $\tau_{(\tilde{\mathbf{c}},c_m)} \in \mathsf{D}(Q_1 \times \cdots Q_n)$. As before, we may identify tuples and standard basis vectors as follows:

$$\mathbf{c} \;=\; (\tilde{\mathbf{c}}, c_m) \;\in\; C_1 \times \cdots \times C_{m-1} \times C_m \;=\; C\,, \tag{1.19a}$$

$$|\mathbf{c}\rangle \;=\; \big|\,(\tilde{\mathbf{c}}, c_m)\,\big\rangle = |\tilde{\mathbf{c}}\rangle \otimes |c_m\rangle \;\in\; \mathcal{H}_C\;; \tag{1.19b}$$

then, if we let $\lambda(\mathbf{c}) = \lambda_m(c_m)\tilde{\lambda}_{c_m}(c_1, \ldots, c_{m-1})$, we have

$$\begin{aligned}
\rho \;&=\; \sum_{\substack{\tilde{\mathbf{c}} \in \tilde{C} \\ c_m \in C_m}} \lambda_m(c_m)\tilde{\lambda}_{c_m}(\tilde{\mathbf{c}})\;\; \tau_{(\tilde{\mathbf{c}},c_m)} \otimes |\tilde{\mathbf{c}}\rangle\langle\tilde{\mathbf{c}}| \otimes |c_m\rangle\langle c_m| \\
&=\; \sum_{\mathbf{c} \in C} \lambda(\mathbf{c})\; \tau_{\mathbf{c}} \otimes |\mathbf{c}\rangle\langle\mathbf{c}|\;,
\end{aligned} \tag{1.20}$$

where for each $\mathbf{c} \in C_1 \times \cdots \times C_m$, we have $\tau_{\mathbf{c}} \in \mathsf{D}(Q_1 \times \cdots \times Q_n)$. Then ρ is a convex combination of the form described by the Lemma. ■

This allows us to "factor" a composite system, consisting of classical and quantum registers, into classical and quantum *subsystems* which we may treat as a single (composite) quantum register, together with a single (composite) classical register; and the states of these subsystems as consisting of "classically probabilistic" and "irreducibly quantum" data.

1.1.3 Measurement of quantum data

As we noted before, quantum systems are sensitive to testing for events, which requires that we describe how they evolve under event testing. We will define three senses of measurement, starting with the following:

Definition 1-4. For a finite set Q, a set of operators $\{\Pi_r\}_{r\in R} \subseteq \mathsf{Herm}(Q)$ is a *complete set of orthogonal projectors* if

$$\sum_{r\in R} \Pi_j \;=\; \mathbb{1}_Q \qquad \text{and} \qquad \Pi_j\Pi_k \;=\; \delta_{j,k}\,\Pi_j\,, \tag{1.21}$$

where

$$\delta_{j,k} \;=\; \left\{ \begin{array}{ll} 1\,, & \text{if } j=k \\ 0\,, & \text{otherwise} \end{array} \right\} \tag{1.22}$$

is the Kronecker delta. A *projective measurement* with respect to $\{\Pi_r\}_{r\in R}$ on a quantum register Q over Q is a mapping $\mathsf{P} : \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q)$ given by

$$\mathsf{P}(\rho) \;=\; \sum_{r\in R} \Pi_r\,\rho\,\Pi_r\,. \tag{1.23}$$

We call P a *complete* projective measurement if each Π_r has rank 1.

Note that for any state ρ with which the projectors Π_j all commute, we have $\mathsf{P}(\rho) = \rho$: that is, if there is a set of orthogonal point-mass distributions (pure states) on the system which can be used to decompose both ρ and the projectors Π_s, the measurement does not affect the state of the system. However, for any other $\rho \in \mathsf{D}(R)$, a projective measurement will disturb the state.

We usually suppose that measurement corresponds to a process of actually measuring a physical parameter, the result of which is recorded in the state of another system used for read-out (represented by a *classical* register). This corresponds to a sense of measurement defined by von Neumann [131]:

Definition 1-5. A *von Neumann measurement* with respect to a complete set of orthonormal projectors $\{\Pi_r\}_{r\in R} \subseteq \mathsf{Herm}(Q)$, on a quantum register Q over a finite set Q, is a mapping $\mathsf{N} : \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q) \otimes \mathsf{Prob}(R)$ given by

$$\mathsf{N}(\rho) \;=\; \sum_{r\in R} \Big(\Pi_r \otimes |r\rangle\Big)\rho\Big(\Pi_r \otimes \langle r|\Big) \;=\; \sum_{r\in R} \Pi_r\,\rho\,\Pi_r \;\otimes\; |r\rangle\langle r|\,. \tag{1.24}$$

We call N a *complete* von Neumann measurement if each Π_r has rank 1.

The output space consists of joint distributions across the original quantum system Q and a classical *result register* R, whose marginal distribution may be interpreted as simply a probability distribution.

A special case of a von Neumann measurement which it will be often useful to discuss is a measurement *with respect to an observable*. In quantum mechanics, Hermitian operators A may correspond to physical properties of a system (such as energy or angular

momentum), where the property described by A has the expected value λ for any system in a state ρ such that $\mathrm{Tr}(A\rho) = \lambda$. We may describing measurement of an observable A with a von Neumann measurement, as follows:

Definition 1-6. Let $A \in \mathsf{Herm}(Q)$ be an operator with spectrum given by

$$\Lambda = \left\{ \lambda \in \mathbb{R} \,\middle|\, \det(A - \lambda \mathbb{1}_Q) = 0 \right\} . \tag{1.25}$$

A von Neumann measurement *of the observable* A on a quantum register Q over Q is a mapping $\mathsf{N} : \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q) \otimes \mathsf{Prob}(\Lambda)$ given by

$$\mathsf{N}(\rho) \;=\; \sum_{\lambda \in \Lambda} \Pi_\lambda \rho \, \Pi_\lambda \otimes |\lambda\rangle\langle\lambda| \;, \tag{1.26}$$

where each operator Π_λ is the projector onto the λ-eigenspace of A.

As a mathematical concept, this can be trivially extended to measurement of any normal operator on a quantum register, but this will not be necessary for this thesis.

We will regard the state of a classical subsystem (such as a result register) to be known in practise — that is, without an infinite regress of measurements — even if it is not determined in advance; then, we may consider the state of the quantum registers conditioned on a particular state of the classical subsystem. For a post-measurement state $\bar{\rho}$ resulting from a von Neumann measurement, we may consider post-measurement distributions $\bar{\rho}_R \in \mathsf{D}(R)$ conditioned on a particular value of the result register after measurement t, which for an arbitrary projector E over $\mathcal{H}_R$ describing (by a conventional abuse of notation) an event E, is given by

$$\begin{aligned}
\mathrm{Tr}(E\,\bar{\rho}_R) \;=\; \Pr_{\bar{\rho}}(E|t) \;&=\; \frac{\Pr_{\bar{\rho}}(E,t)}{\Pr_{\bar{\rho}}(t)} \;=\; \frac{\mathrm{Tr}\left(\left[E \otimes |t\rangle\langle t| \,\right] \bar{\rho} \right)}{\mathrm{Tr}\left(|t\rangle\langle t| \;\mathrm{tr}_R(\bar{\rho}) \right)} \\
&= \frac{\mathrm{Tr}\left(E\,\Pi_t \rho\, \Pi_t \otimes |t\rangle\langle t| \right)}{\mathrm{Tr}\left(\sum_{r \in R} \langle r|\, \Pi_t \rho \Pi_t \,|r\rangle \cdot |t\rangle\langle t| \right)} \\
&= \mathrm{Tr}\left(E \left[\frac{\Pi_t \rho\, \Pi_t}{\mathrm{Tr}\left(\Pi_t \rho\, \Pi_t \right)} \right] \right) ,
\end{aligned} \tag{1.27}$$

which entails by linearity that

$$\bar{\rho}_R \;=\; \frac{\Pi_t \rho\, \Pi_t}{\mathrm{Tr}\left(\Pi_t \rho\, \Pi_t \right)} \tag{1.28}$$

conditioned on the result register being in the state t.

A final variety of measurement which we consider is one in which the system being measured is in a sense destroyed by the measurement, as in the case of photons: this occurs *e.g.* in the detection of photons. In this case, a measurement is performed and a result obtained, but no conditional state of the original system meaningfully exists. We may describe such a measurement by:

Definition 1-7. Let $\{|\psi_j\rangle\}_{j=1}^n \subseteq \mathcal{H}_Q$ be a collection of (not necessarily normalized) vectors such that

$$\sum_{j=1}^{n} |\psi_j\rangle\langle\psi_j| \;=\; \mathbb{1}_Q\,, \tag{1.29}$$

and consider a mapping $r : \{1,\ldots,n\} \longrightarrow R$. Then a *destructive measurement*[2] on a state space $\mathsf{D}(Q)$ with respect to the states $\{|\psi_j\rangle\}_{j=1}^n$ and mapping r is a transformation $\mathsf{M} \,:\, \mathsf{D}(Q) \longrightarrow \mathsf{E}(R)$ of the form

$$\mathsf{M}(\rho) \;=\; \sum_{j=1}^{n} \big|r(j)\big\rangle\big\langle\psi_j\big|\;\rho\;\big|\psi_j\big\rangle\big\langle r(j)\big|\;. \tag{1.30}$$

We say that M is an *orthonormal* (destructive) measurement if the states $|\psi_j\rangle$ are orthonormal, and that it is also *complete* if r is injective.

We will generally be interested in the special case of complete orthonormal (destructive) measurements, in which case the vectors $|\psi_j\rangle$ form an orthonormal basis. In particular, we may define a sense of the destructive measurement of an observable:

Definition 1-8. Let $A \in \mathsf{Herm}(Q)$ be an operator with spectrum given by

$$\Lambda = \big\{\lambda \in \mathbb{R} \,\big|\, \det(A - \lambda\mathbb{1}_Q) = 0\big\}\;, \tag{1.31}$$

and for each $\lambda \in \Lambda$, let $\{|\psi_{\lambda,j}\rangle\}_{j=1}^{m_\lambda}$ be an orthogonal basis of the λ-eigenspace of A. Then a destructive measurement *of the observable* A on a quantum register Q over Q is a mapping $\mathsf{M} : \mathsf{D}(Q) \longrightarrow \mathsf{Prob}(\Lambda)$ given by

$$\mathsf{M}(\rho) \;=\; \sum_{\lambda\in\Lambda}\sum_{j=1}^{m_\lambda} |\lambda\rangle\langle\psi_{\lambda,j}|\,\rho\,|\psi_{\lambda,j}\rangle\langle\lambda| \;=\; \big(\mathrm{tr}_Q \circ \mathsf{N}\big)(\rho)\,, \tag{1.32}$$

where $\mathsf{N} \,:\, \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q)\otimes\mathsf{Prob}(\Lambda)$ is a von Neumann measurement of the observable A, and tr_Q is a trace-out operation on Q.

Orthonormal destructive measurements and projective measurements can both can be recovered from von Neumann measurements: projective measurements represent the marginal distribution of the system which is measured, and projective destructive measurements represent the marginal distribution of the result register.

[2] Destructive measurements as defined here correspond to *positive operator-valued measurements*, or POVMs (see *e.g.* [103]): we may define

$$E_t \;=\; \sum_{j\,:\,r(j)=t} |\psi_j\rangle\langle\psi_j|\;,$$

from which we obtain

$$\mathrm{Tr}(E_t\,\rho) \;=\; \mathrm{Tr}\left(\sum_{j=1}^{n} |t\rangle\langle t\,|r(j)\rangle\langle\psi_j|\,\rho\,|\psi_j\rangle\langle r(j)|\right) \;=\; \mathrm{Tr}\Big(|t\rangle\langle t|\,\mathsf{M}(\rho)\Big)\;;$$

the primary difference between POVMs and destructive measurements as presented here is in the explicit description of the classical register retaining the measurement result.

1.1.4 Quantum state vectors and qubits

Although it will often be important to describe quantum computation in terms of the more general mathematical apparatus described so far, we now provide some specialized definitions and notation which will help to streamline discussion in many cases.

State vectors for pure states

Recall the definition of *pure state* on page 2 as a point-mass distribution over a maximally refined set of events. As we described on page 5, a point-mass function corresponds to a choice of one projector $|\psi_j\rangle\langle\psi_j|$, which is characterized by vector $|\psi_j\rangle$ out of an orthonormal basis. In those cases when we restrict our attention to pure states, we may simplify our analysis by considering the following in place of density operators:

Definition 1-9. For a quantum register R system in a pure state $\rho \in \mathsf{D}(R)$, the *(pure) state vector* of R is a unit ℓ_2-norm vector $|\psi\rangle \in \mathcal{H}_R$ such that $\rho = |\psi\rangle\langle\psi|$.

Note that state vectors are defined only up to scalar factors, as we have $\big[\omega\,|\psi\rangle\big]\big[\omega^*\langle\psi|\big] = |\psi\rangle\langle\psi|$ for any vector $|\psi\rangle$ and scalar factor $\omega \in \mathbb{C}$ with $|\omega| = 1$. As well, we may more generally represent some pure state ρ by any non-zero vector $|\psi\rangle$ by first "normalizing" $|\psi\rangle$ to obtain a unit (pure state) vector for ρ, as follows:

$$\rho \;=\; \left[\frac{|\psi\rangle}{\||\psi\rangle\|_2}\right]\left[\frac{\langle\psi|}{\||\psi\rangle\|_2}\right] \;=\; \frac{|\psi\rangle\langle\psi|}{\langle\psi\,|\psi\rangle}\,. \tag{1.33}$$

Thus, we will primarily be interested in proportionality rather than equality when treating pure-state vectors.

Qubits and qubit state vectors

We will be primarily interested in a specific type of quantum register, which defines the units of quantum data that we will manipulate:

Definition 1-10. A *qubit* is a quantum register over the set $\{0,1\}$: we may write $\mathsf{D}(2)$ for the state-space of a qubit, and $\mathcal{H}_2 = \mathrm{span}_{\mathbb{C}}\big\{\,|0\rangle, |1\rangle\,\big\}$ for the Hilbert space on which the operators of $\mathsf{D}(2)$ are supported. (We will occasionally write $\mathsf{D}(2^n) = \mathsf{D}(2 \times \cdots \times 2)$ for the state space of n generic qubits.) By analogy, we consider a *classical bit* (considered as a physical system) to be a classical register over $\{0,1\}$.

The emphasis that will be placed on bits and qubits in this thesis motivate some simplifying notations and conventions:

Notation. We will tend to represent qubits by lower-case roman letters. We will denote the state space of a qubit v by $\mathsf{D}(v)$, the Hilbert space on which it is supported by $\mathcal{H}_v$, and the identity operator on $\mathcal{H}_v$ by $\mathbb{1}_v$, to distinguish these from spaces/operators corresponding to different qubits.

Notation. In a mild abuse of notation, the joint state spaces of multiple qubits $u, v, \ldots$ will be denoted $\mathsf{D}(u, v, \ldots)$, the Hilbert space which supports it by $\mathcal{H}_2^{\otimes\{u,v,\ldots\}}$, and the identity operator of that Hilbert space variously as $\mathbb{1}_2^{\otimes\{u,v,\ldots\}}$ and $\mathbb{1}_{u,v,\ldots}$. (We will also tend to represent the identity superoperator $\mathrm{id}_{\mathsf{D}(u,v,\ldots)}$ acting on $\mathsf{D}(u, v, \ldots)$ by the latter two symbols.)

We will also extend the above conventions to state-spaces and operations on bits, although these will be less common in practise.

Notation. We will define the following alternative and short-hand notations for various qubit state vectors:

$$\begin{array}{lll} |{+}\mathsf{z}\rangle \;=\; |0\rangle\,, & |{+}\mathsf{x}\rangle \;=\; \frac{1}{\sqrt{2}}\Big[\,|0\rangle + |1\rangle\,\Big], & |{+}\mathsf{y}\rangle \;=\; \frac{1}{\sqrt{2}}\Big[\,|0\rangle + i\,|1\rangle\,\Big], \\ |{-}\mathsf{z}\rangle \;=\; |1\rangle\,, & |{-}\mathsf{x}\rangle \;=\; \frac{1}{\sqrt{2}}\Big[\,|0\rangle - |1\rangle\,\Big], & |{-}\mathsf{y}\rangle \;=\; \frac{1}{\sqrt{2}}\Big[\,|0\rangle - i\,|1\rangle\,\Big]. \end{array} \tag{1.34}$$

We often further abbreviate $|{+}\rangle = |{+}\mathsf{x}\rangle$ and $|{-}\rangle = |{-}\mathsf{x}\rangle$. We may also write $|{\pm}\rangle$ or $|{\pm}\mathsf{x}\rangle$ to denote one or both of the state vectors $|{+}\mathsf{x}\rangle$ or $|{-}\mathsf{x}\rangle$, and similarly for $|{\pm}\mathsf{y}\rangle$ and $|{\pm}\mathsf{z}\rangle$.

1.2 Unitary evolution and elementary gate sets

We have already partially described transformations that can be performed on quantum data: by CPTP maps in general, and by measurement operations (projective, von Neumann, and destructive) as a special case. We now describe the theory by which general CPTP maps may be decomposed into (or approximated by) the product of discrete elementary operations, which are supposed to be physically implementable in principle. At the same time, we present the theoretical importance of *unitary transformation* of quantum data, which will motivate the questions explored in this thesis.

1.2.1 Unitary evolution of quantum states

Evolution of pure states (described by state-vectors $|\psi_t\rangle \in \mathcal{H}_Q$) in time are governed by the Schrödinger equation [118, 131],

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_t\rangle \;=\; H_t\,|\psi_t\rangle\,, \tag{1.35}$$

where H_t is the *Hamiltonian* of the system, a linear operator on $\mathcal{H}_Q$ which may depend on time, whose eigenvalues are possible energy levels of pure states of the system. Energy is represented here by real numbers: abstracting away the physical content of this equation, H_t is therefore a Hermitian operator. Then, we have

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\,\langle\psi_t|\psi_t\rangle \;&=\; \Big[\frac{\mathrm{d}}{\mathrm{d}t}\,\langle\psi_t|\Big]\,|\psi_t\rangle \;+\; \langle\psi_t|\,\Big[\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi_t\rangle\Big] \\ &=\; \Big[\frac{i}{\hbar}\,\langle\psi_t|\,H_t^\dagger\Big]\,|\psi_t\rangle \;+\; \langle\psi_t|\,\Big[\frac{-i}{\hbar}H_t\,|\psi_t\rangle\Big] \;=\; 0\,; \end{aligned} \tag{1.36}$$

that is, Schrödinger evolution preserves the ℓ_2-norm of vectors (*i.e.* it is well-defined as an evolution of pure state vectors). The evolution of an input state $|\psi_0\rangle$ from $t = 0$ to some $t = \tau$ can then be described by some operator U_τ on the pure states of the system which preserves the ℓ_2-norm. For a Hilbert space $\mathcal{H}_Q$ on a finite set Q, we may consider the action of U_τ on the standard basis of $\mathcal{H}_Q$: we have $\langle q| U_\tau^\dagger U_\tau |q\rangle = 1$ by the preservation of norm of the states $|q\rangle$ for $q \in Q$; and for any combination $|\phi\rangle = \frac{1}{\sqrt{2}}|p\rangle + \frac{1}{\sqrt{2}}\mathrm{e}^{i\theta}|q\rangle$ for distinct $p, q \in Q$ and arbitrary angles $\theta \in \mathbb{R}$, we have

$$\begin{aligned} 1 &= \langle\phi| U_\tau^\dagger U_\tau |\phi\rangle \\ &= \tfrac{1}{2}\Big[\langle p| U_\tau^\dagger U_\tau |p\rangle + \mathrm{e}^{i\theta}\langle p| U_\tau^\dagger U_\tau |q\rangle + \mathrm{e}^{-i\theta}\langle q| U_\tau^\dagger U_\tau |p\rangle + \langle q| U_\tau^\dagger U_\tau |q\rangle\Big] \\ &= 1 + \mathrm{Re}\left(\mathrm{e}^{i\theta}\langle p| U_\tau^\dagger U_\tau |q\rangle\right). \end{aligned} \tag{1.37}$$

Then $\mathrm{e}^{i\theta}\langle p| U_\tau^\dagger U_\tau |q\rangle$ has no real component for any angle θ, which implies $\langle p| U_\tau^\dagger U_\tau |q\rangle = 0$ for distinct $p, q \in Q$. We therefore have $U_\tau^\dagger U_\tau = \mathbb{1}_Q$: that is, U_τ is a *unitary operation*. Applied to the evolution of density operators, the time-evolution Φ_τ of density operators corresponding to pure states from time $t = 0$ to $t = \tau$ is then

$$\Phi_\tau\big(|\psi_0\rangle\langle\psi_0|\big) = |\psi_\tau\rangle\langle\psi_\tau| = U_\tau |\psi_0\rangle\langle\psi_0| U_\tau^\dagger, \tag{1.38a}$$

which by linearity implies that

$$\Phi_\tau(\rho) = U_\tau \rho U_\tau^\dagger \tag{1.38b}$$

for all $\rho \in \mathsf{D}(Q)$, which is a valid CPTP map. We say that a CPTP map is unitary if it can be expressed as in (1.38b); and we denote the set of unitary operators on $\mathcal{H}_S$ by $\mathsf{U}(S)$.

Precisely how *measurement processes* (as described by CPTP maps of the kind described in Definitions 1-4 through 1-8, on pages 9 – 11) might arise from unitary evolution, represent an explicit departure from unitary evolution, or represent something entirely subjective, is a subject of ongoing research and debate (see *e.g.* [9, 67, 109, 132, 136]). Such considerations are beyond the scope of this thesis: in practise, we adopt the approach of von Neumann [131], and treat unitary evolution and measurements as complementary ways in which states may be transformed; and we supplement this also with maps of the form $\rho \longmapsto \rho \otimes \sigma$ for density operators ρ and σ, corresponding to augmenting the system by introducing an auxiliary system in the state σ.

A remark on continuous time evolution. As we have noted above, Schrödinger evolution of physical systems can be described as being continuous in time. Despite this, we will generally describe computation as being performed by *discrete* transformations, *e.g.* by applying unitary transition operators, rather than Schrödinger evolution. The reasons for this are practical in nature: because we can only perform operations with finite precision in any particular setting, and because arbitrarily high precision can only be achieved with a commensurate decrease in the speed of computation, we limit ourselves to models of quantum computation where (like measurement and the introduction of auxiliary systems) the evolution of a quantum system is performed in discrete steps. As well, the same unitary operation can be realized in different ways by Schrödinger

evolution: for instance, considering time-independent Hamiltonians alone, a unitary U_τ can be achieved in time τ by Schrödinger evolution by any Hamiltonian H satisfying $U_\tau = \exp(-i\hbar H\tau)$, where

$$\exp(A) \;=\; \sum_{n\in\mathbb{N}} \frac{A^n}{n!} \tag{1.39}$$

for any Hermitian matrix A; the set of such Hamiltonians is an equivalence class of operators with common eigenvectors, and whose eigenvalues differ by integer multiples of 2π. By considering discrete-time unitary evolution, we abstract away details such as the particular Hamiltonian, which in an implementation may be determined later.

1.2.2 Elementary sets of gates

To perform computations, we suppose that we are provided a set of elementary operations (or *elementary gates*) as follows:

Definition 1-11. A set GATES of elementary gates on a set of classical registers B and a set of quantum registers V is a set of operators $\mathsf{Op}_{\sigma:\alpha/\delta}$, each of which consists of some CPTP map

$$\mathsf{Op} \;:\; \mathsf{D}\big(\{0,1\}^{n+m}\big) \;\longrightarrow\; \mathsf{D}\big(\{0,1\}^{n+\ell}\big)\,, \tag{1.40}$$

which operates on finite sequences of registers given by σ, α, and δ (where the input and output distributions of Op are also consistent with the additional constraints imposed by the state space for bits as classical registers). In particular, $\alpha = (a_j)_{j=1}^{\ell}$ is a sequence of *allocated* registers which are produced as output but which were not present as input; $\delta = (d_j)_{j=1}^{m}$ is a sequence of *discarded* registers which are taken as input but not produced as output; and $\sigma = (s_j)_{j=1}^{n}$ is a sequence *stable* registers taken as input and produced as output, yielding an operation

$$\mathsf{Op}_{\sigma:\alpha/\delta} \;:\; \mathsf{D}\big(\sigma;\delta\big) \;\longrightarrow\; \mathsf{D}\big(\sigma;\alpha\big)\,. \tag{1.41}$$

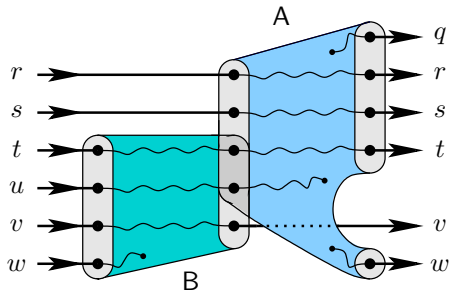
We say that the *type* of the operation is $[\sigma : \alpha/\delta]$.

The sequence of registers for an elementary operation, once fixed, is significant. This may be illustrated with a map $\mathsf{M}_{a,b:r/\varnothing}$ performing a von Neumann measurement on one of two qubits a and b, producing a single result bit r, defined by

$$\mathsf{M}_{a,b:r/\varnothing}(\rho) \;=\; \sum_{\beta\in\{0,1\}} \Big(\mathbb{1}_a \otimes |\beta\rangle\langle\beta|_b\Big)\,\rho\,\Big(\mathbb{1}_a \otimes |\beta\rangle\langle\beta|_b\Big) \otimes |\beta\rangle\langle\beta|_r\;; \tag{1.42}$$

then, for *e.g.* two qubits u and v in independent pure states, $\mathsf{M}_{u,v:r/\varnothing}$ performs a measurement on v (producing the result in r) while leaving the state of u undisturbed, while $\mathsf{M}_{v,u:r/\varnothing}$ does the opposite.

In order for operations on bits and qubits — or more generally, any register — to be unambiguous, there must be exactly one "copy" of a given register which is to be operated upon; and any register to be operated on by some operation cannot have been explicitly discarded by the preceding operations without being re-allocated. We formalize these constraints through the concept of the types defined for each gate, as follows:



$$\mathsf{B} \sim [t,u,v : \varnothing/w] \qquad \mathsf{A} \sim [r,s,t : q,w/u]$$
$$\sigma_B = \{t,u,v\} \qquad \sigma_A = \{r,s,t\}$$
$$\alpha_B = \varnothing \qquad \alpha_A = \{q,w\}$$
$$\delta_B = \{w\} \qquad \delta_A = \{u\}$$

$$\mathsf{C} = \mathsf{A} \circ \mathsf{B} \sim [r,s,t,v,w : q/u]$$

FIGURE 1-1. [*colour online*] Illustration of the type of the composition of two operations A and B acting on named registers (represented as "wires" progressing in time from left to right). Registers not explicitly required by the input or output of B are assumed to be available to be either allocated, discarded, or transformed by A as necessary, and vice-versa. Otherwise, the types of A and B must agree on what bits or qubits are available to be operated on after the application of B and before the application of A.

Lemma 1-2. *For two operations* $\mathsf{A}_{\sigma_A:\alpha_A/\delta_A}$ *and* $\mathsf{B}_{\sigma_B:\alpha_B/\delta_B}$ *with types* $[\sigma_A : \alpha_A/\delta_A]$ *and* $[\sigma_B : \alpha_B/\delta_B]$ *respectively, the composition* $\mathsf{A}_{\sigma_A:\alpha_A/\delta_A} \circ \mathsf{B}_{\sigma_B:\alpha_B/\delta_B}$ *is well-formed if and only if*

(*i*) *there is no intersection between the registers in* α_A*, and the registers in* α_B *or* σ_B *which are explicitly specified in the output of* $\mathsf{B}_{\sigma_B:\alpha_B/\delta_B}$*;*

(*ii*) *there is no intersection between the registers of* δ_B*, and the registers in* δ_A *or* σ_A *which are explicitly specified in the input of* $\mathsf{A}_{\sigma_A:\alpha_A/\delta_A}$*.*

If the above conditions hold, the composition $\mathsf{C}_{\sigma_C:\alpha_C/\delta_C} = \mathsf{A}_{\sigma_A:\alpha_A/\delta_A} \circ \mathsf{B}_{\sigma_B:\alpha_B/\delta_B}$ *has type* $[\sigma_C : \alpha_C/\delta_C]$*, where the sequences* σ_C*,* α_C*, and* δ_C *are arbitrary orderings of the registers in the following "set" constructions:*

$$\sigma_C = (\sigma_A \cap \sigma_B) \cup (\alpha_A \cap \delta_B)\,, \tag{1.43a}$$
$$\alpha_C = (\alpha_A \setminus \delta_B) \cup (\alpha_B \setminus \delta_A)\,, \tag{1.43b}$$
$$\delta_C = (\delta_B \setminus \alpha_A) \cup (\delta_A \setminus \alpha_B)\,. \tag{1.43c}$$

The type composition relations of (1.43) are illustrated in Figure 1-1.[3]

[3]The emphasis given here on the composability of CPTP maps is meant largely to prefigure similar discussions for particular, "non-standard" representations of quantum computation.

A particular set GATES of elementary operations defines a specific *circuit model*, consisting of the well-formed compositions of those gates:

Definition 1-12. Given a set GATES of elementary gates on a set B of bits and V of qubits, a *circuit in the model defined by* GATES (or *in the* GATES *model*) is the composition C of a finite sequence of operations of GATES. The *size* of such a circuit is the number of operations it uses, and the *depth* of the circuit is the minimum integer D such that the circuit may be decomposed in the form,

$$C = \prod_{t=1}^{D}{}^{>} \left(\mathsf{Op}_{t,1} \otimes \mathsf{Op}_{t,2} \otimes \cdots \otimes \mathsf{Op}_{t,m_t}\right) \tag{1.44}$$

i.e. as an ordered product of D parallel (tensor) products of gates on independent sets of qubits.

From this point onwards, we will consider only models of quantum computation acting on bits and qubits (*i.e.* where the set of elementary gates only acts on aggregates of registers over $\{0,1\}$). We also assume that the number of available bits and qubits is infinite, and that the only limitation on the number of qubits which a circuit may operate upon is imposed by the set of elementary gates.

Notation. Operators U written in italics represent linear operators (*e.g.* unitary transformations), and superscripts denote exponents of these operators. Operators Υ written in upright sans-serif represent CPTP maps. Superscripts of CPTP maps denote either real-valued parameters or parameters provided by an interactive classical control, rather than exponents. For two CPTP maps A and B, we will often write AB for $\mathsf{A} \circ \mathsf{B}$ when this composition is well-defined (although we will continue to write in terms of compositions for much of this section).

1.2.3 Universality for quantum computation/unitary evolution

In order to have a robust model of computation, it is helpful not only to have a set of discrete operations with which to define a sense of computational complexity, but also to have a well-defined notion of the range of operations which that set generates. This motivates a discussion of senses of *universality* of quantum operations.

Universality for generating/approximating distributions

For quantum computation on aggregates of qubits, the natural candidate is the the ability to generate all CPTP maps $\Phi : \mathsf{D}(2^n) \longrightarrow \mathsf{D}(2^n)$ for any $n \geqslant 1$. Alternatively, if we wish to consider models of computation with only a finite set of elementary operations, we cannot generate the uncountably[4] many CPTP maps from finite compositions from a finite set of operations; however, we may define a sense of *approximate* universality by defining a sense in which a CPTP map may be approximated with some precision

[4] "Uncountable" is used here in the set-theoretic sense, meaning that the cardinality of the set of such maps is greater than that of the natural numbers.

$\varepsilon > 0$. We will present a formal definition of approximating CPTP maps below. However, defining universality in terms of generating or approximating arbitrary CPTP maps misses important models of computation (such as some models of measurement-based quantum computation, which we present in Section **??**) which do not take quantum states as input, but which achieve what is essentially the main practical motivation of quantum computation: a means of efficiently preparing a larger class of probability distributions via measurement than is apparently possible with, say, randomized Turing machines.

The complementary alternative — considering only probability distributions that may be prepared by a set of gates — presents no distinction between quantum and classical computation. While the set of functions computable by classical and by quantum computers from explicitly specified primitives are the same, it is commonly thought that the functions which can be *efficiently* computed by each are not the same. We might wish to investigate descriptions of universality for quantum computation which strictly subsumes what classical computers can achieve, if we are interested in possible indications of the differences in what can be efficiently done by classical computation and by quantum computation. One approach could be to explicitly consider computational resources in any discussion of universality: to describe *e.g.* pertinent bounds (or lack of bounds) on time, space, multiparty communication, *etc.* when describing senses of universality. A discussion of different senses of universality is presented in [49], which raises the issue of addressing universality and computational complexity at the same time. However, for the sake of simplicity, it is more convenient to completely separate concerns of efficiency and universality.[5]

A reasonable compromise position is to define a sense of universality which strictly extends the sense of generating probability distributions, but which does not presume that arbitrary quantum inputs are possible inputs to the operations of the model — that is, simply the generation or approximation of arbitrary density operators. The natural distance on probability distributions is that induced by the ℓ_1-norm, which — using the correspondence of (1.6) — is the sum of the absolute values of the entries of a diagonal matrix. By [83, Lemma 10.2], the *operator ℓ_1-norm* (also called the trace norm) on linear operators M between finite-dimensional Hilbert spaces,

$$\|M\|_1 = \sum_j \sigma_j \,, \qquad \text{where } \sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant 0 \text{ are the singular values of } M, \tag{1.45a}$$

$$= \operatorname{Tr}\left(\sqrt{M^\dagger M}\right) \,, \quad \text{for } \sqrt{M^\dagger M} = S \text{ such that } S \geqslant 0 \text{ and } S^2 = M^\dagger M; \tag{1.45b}$$

and the corresponding metric on density operators,

$$d(\rho, \sigma) = \|\rho - \sigma\|_1 \tag{1.45c}$$

are the natural generalization to arbitrary density operators. Then, we define a sense of universality for quantum computation as follows:

[5] The rarity of such senses of universality in the literature is likely due, no doubt, to the success of the Strong Church-Turing thesis, which posits that any "physically realizable" model of computation is polynomial-time equivalent to the Turing machine model of computation. Even in light of the apparent failure of the Strong Church-Turing thesis with the advent of quantum computation, there seems to be remarkably few natural senses of universal computation arising out of "physically realistic" models of computation which are not provably poly-time equivalent, except for those which are equivalent to constraining another "physically realistic" model in some way.

Definition 1-13. For two density operators $\rho, \tilde{\rho}$, we say that $\tilde{\rho}$ *approximates* ρ *with precision* ε if $0 \leqslant \|\tilde{\rho} - \rho\|_1 < \varepsilon$. A collection of elementary gates GATES is *universal for quantum computation* if any density operator $\rho \in \mathsf{D}(2^n)$ on n qubits can be generated as a circuit in the GATES model, for every $n \geqslant 1$; alternatively, GATES is *approximately universal for quantum computation* if any such operator ρ can be approximated to arbitrary precision $\varepsilon > 0$ by some circuit in the GATES model.

This definition of universality for quantum computation corresponds to the definition of "**CQ**-universality" presented in [49].

Universality for generating/approximating CPTP maps

Although generation or approximation of density operators is what we have chosen to define universality, it is certainly sufficient to be able to generate or approximate all CPTP maps on n qubits (corresponding to "**QQ**-universality" as described in [49]). The ultimate significance of a CPTP map from the point of view of quantum computation is in the probability distributions arising from measurement. A natural definition of an approximation $\tilde{\Phi}$ of a CPTP map Φ is then when the result of a measurement (*e.g.* a destructive measurement as in Definition 1-7) on either $\tilde{\Phi}(\rho)$ or $\Phi(\rho)$ yields similar probability distributions for any input state ρ; more generally, we may consider measurements performed on $\big[\tilde{\Phi} \otimes \mathbb{1}\big](\rho)$ and $\big[\Phi \otimes \mathbb{1}\big](\rho)$, where ρ is a state of some larger system. Then, considering upper bounds on the difference of the effects of two maps $\tilde{\Phi} : \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q')$ and $\Phi : \mathsf{D}(Q) \longrightarrow \mathsf{D}(Q')$ on arbitrary density operators, we are led to the following metric on superoperators:

$$\begin{aligned} \Delta\big(\tilde{\Phi}, \Phi\big) &= \sup_{|A| \geqslant 1} \sup_{\rho \in \mathsf{D}(Q \times A)} d\left(\Big(\tilde{\Phi} \otimes \mathbb{1}_A\Big)(\rho)\,,\ \Big(\Phi \otimes \mathbb{1}_A\Big)(\rho)\right) \\ &= \sup_{|A| \geqslant 1} \sup_{\substack{M \in \mathsf{Herm}(Q \times A) \\ M \geqslant 0,\, M \neq 0}} \frac{\left\|\Big(\tilde{\Phi} \otimes \mathbb{1}_A\Big)(M) - \Big(\Phi \otimes \mathbb{1}_A\Big)(M)\right\|_1}{\|M\|_1}\,. \end{aligned} \tag{1.46}$$

We may simplify this definition slightly by removing the emphasis on positive operators, and defining the *superoperator norm* (introduced as the *diamond norm* in [4]), in the fashion of [83]:[6]

$$\|\Psi\|_\diamond = \sup_{|A| \geqslant 1} \sup_{\substack{M \in \mathsf{L}(\mathcal{H}_Q \otimes \mathcal{H}_A) \\ M \neq 0}} \frac{\left\|\Big(\Psi \otimes \mathbb{1}_A\Big)(M)\right\|_1}{\|M\|_1}\,. \tag{1.47}$$

It is easy to see that the value of the inner supremum is non-decreasing with the size of $|A|$, as we may isometrically embed $\mathcal{H}_Q \otimes \mathcal{H}_A$ into $\mathcal{H}_Q \otimes \mathcal{H}_{A'}$ for any $|A'| \geqslant |A|$. For the outer supremum, it suffices to take $|A| = |Q|$, as shown in [83, Theorem 11.1]; then, we

[6] This definition actually corresponds to the description of $\|\Psi\|_\diamond$ on page 109 and to the statement of Theorem 11.1 on page 110, in [83]. The definition presented here also differs in that the descriptions in [83] do not involve a supremum over $|A|$.

may equivalently define

$$\|\Psi\|_\diamond = \sup_{\substack{M \in \mathsf{L}(\mathcal{H}_Q \otimes \mathcal{H}_Q) \\ M \neq 0}} \frac{\left\| \left(\Psi \otimes \mathbb{1}_Q \right)(M) \right\|_1}{\|M\|_1} . \tag{1.48}$$

Therefore, we present the following extension of Definition 1-13 for the purposes of describing the generation or approximation of CPTP maps:

Definition 1-14. For two CPTP maps $\Phi, \tilde{\Phi}$ with the same domain and range, we say that $\tilde{\Phi}$ *performs* Φ if $\tilde{\Phi} = \Phi$, and that $\tilde{\Phi}$ *approximates* Φ *with precision* ε if $0 \leqslant \left\| \tilde{\Phi} - \Phi \right\|_\diamond < \varepsilon$. A collection of elementary gates GATES is *universal for CPTP transformation* if any CPTP map $\Phi : \mathsf{D}(2^n) \longrightarrow \mathsf{D}(2^m)$ from n qubits to m qubits can be performed by a circuit in the GATES model, for every $n, m \geqslant 1$; alternatively, GATES is *approximately universal for CPTP transformation* if any such map Φ can be approximated to arbitrary precision $\varepsilon > 0$ by some circuit in the GATES model.

Because a density operator $\rho \in \mathsf{D}(Q)$ is equivalent to a CPTP map $\mathsf{D}(1) \longrightarrow \mathsf{D}(Q)$, (approximate) universality for CPTP transformation entails (approximate) universality for quantum computation.

Remark on approximation of composite CPTP maps. We can approximate any composition of CPTP maps $\Psi_N \circ \cdots \circ \Psi_2 \circ \Psi_1$ by using approximations to each of the maps Ψ, with an imprecision which is at worst the sum of the imprecisions of each of the components: that is, imprecision in approximations compose *additively*, as shown in [4]. We may show this by first noting that for superoperators $\Psi : \mathsf{L}(\mathcal{H}_Q) \longrightarrow \mathsf{L}(\mathcal{H}_{Q'})$ and $\Phi : \mathsf{L}(\mathcal{H}_{Q'}) \longrightarrow \mathsf{L}(\mathcal{H}_{Q''})$, we have

$$\begin{aligned}
\|\Phi \circ \Psi\|_\diamond &= \sup_{\substack{M \in \mathsf{L}(\mathcal{H}_Q \otimes \mathcal{H}_Q) \\ M \neq 0}} \frac{\left\| \left((\Phi \circ \Psi) \otimes \mathbb{1}_Q \right)(M) \right\|_1}{\|M\|_1} \\
&= \sup_{\substack{M \in \mathsf{L}(\mathcal{H}_Q \otimes \mathcal{H}_Q) \\ M \neq 0}} \left[\frac{\left\| \left(\Phi \otimes \mathbb{1}_Q \right) \left[\left(\Psi \otimes \mathbb{1}_Q \right)(M) \right] \right\|_1}{\left\| \left(\Psi \otimes \mathbb{1}_Q \right)(M) \right\|_1} \right] \left[\frac{\left\| \left(\Psi \otimes \mathbb{1}_Q \right)(M) \right\|_1}{\|M\|_1} \right] \\
&\leqslant \left[\sup_{\substack{M' \in \mathsf{L}(\mathcal{H}_{Q'} \otimes \mathcal{H}_{Q'}) \\ M \neq 0}} \frac{\left\| \left(\Phi \otimes \mathbb{1}_{Q'} \right)(M') \right\|_1}{\|M'\|_1} \right] \left[\sup_{\substack{M \in \mathsf{L}(\mathcal{H}_Q \otimes \mathcal{H}_Q) \\ M \neq 0}} \frac{\left\| \left(\Psi \otimes \mathbb{1}_Q \right)(M) \right\|_1}{\|M\|_1} \right] \\
&= \|\Phi\|_\diamond \|\Psi\|_\diamond ,
\end{aligned} \tag{1.49}$$

where we implicitly take the supremum over all auxiliary spaces for the second-last inequality. It is possible to show that any CPTP map has unit superoperator norm:[7]

[7] Using the definition on page of $\|\Psi\|_\diamond$ in [83, page 110], and Theorem 11.1 which follows shortly after, this follows from Theorem 1-3 below. The fact that CPTP maps have *at most* unit superoperator norm may alternatively be obtained from Theorem 1-3 and the sub-multiplicative property shown above, together with the fact that $\|\Phi \otimes \Psi\|_\diamond = \|\Phi\|_\diamond \|\Psi\|_\diamond$, and from the fact that the trace operation and isometries each have unit superoperator norm.

then, if $\tilde{\Psi} : \mathsf{L}(Q) \longrightarrow \mathsf{L}(Q')$ is a CPTP map approximating Ψ with precision ε and $\tilde{\Phi} : L(Q') \longrightarrow \mathsf{L}(Q'')$ is a CPTP map approximating Φ with precision ε', we have

$$\begin{aligned} \left\| \tilde{\Phi} \circ \tilde{\Psi} - \Phi \circ \Psi \right\|_\diamond &\leqslant \left\| \tilde{\Phi} \circ \left(\tilde{\Psi} - \Psi \right) \right\|_\diamond + \left\| \left(\tilde{\Phi} - \Phi \right) \circ \Psi \right\|_\diamond \\ &\leqslant \left\| \tilde{\Phi} \right\|_\diamond \left\| \tilde{\Psi} - \Psi \right\|_\diamond + \left\| \tilde{\Phi} - \Phi \right\|_\diamond \left\| \Psi \right\|_\diamond < \varepsilon + \varepsilon' \,. \end{aligned} \tag{1.50}$$

Thus, to approximate a composite CPTP map within a small margin of error, it suffices to approximate some set of gates which is universal for CPTP transformation, with sufficiently good precision to bound the cumulative error.

Universality for generating/approximating unitary transformations

One of the primary tools of quantum computation is to reduce this sense of universality for quantum computation to the ability to perform/approximate unitary transformations in particular. This is possible due to the following corollary (see *e.g.* [133, Lecture 5] and [87] for explicit treatments) to a theorem by Stinespring [124]:

Theorem 1-3. *For Hilbert spaces $\mathcal{H}$ and $\mathcal{K}$ of finite dimension, a superoperator $\Phi : L(\mathcal{H}) \longrightarrow \mathsf{L}(\mathcal{K})$ is a CPTP map if and only if, for some auxiliary Hilbert space $\mathcal{E}$ and some isometry $T : \mathcal{H} \longrightarrow \mathcal{K} \otimes \mathcal{E}$, we have*

$$\Phi(\rho) = \mathrm{tr}_{\mathcal{E}}(T \rho T^\dagger) \,. \tag{1.51}$$

If we define $\mathsf{T}(\rho) = T \rho T^\dagger$, this is a decomposition of Φ into an isometry and a trace-out operation. When $\mathcal{H}$, $\mathcal{E}$, and $\mathcal{K}$ as above are restricted to tensor products of $\mathcal{H}_2$, the isometry $T : \mathcal{H} \longrightarrow \mathcal{K} \otimes \mathcal{E}$ can be decomposed into the preparation of an arbitrary state $|\phi\rangle \in \mathcal{A} \cong {\mathcal{H}_2}^{\otimes m}$ (for some number m of auxiliary qubits such that $\mathcal{H} \otimes \mathcal{A} \cong \mathcal{K} \otimes \mathcal{E}$), and a unitary transformation $U : \mathcal{H} \otimes \mathcal{A} \longrightarrow \mathcal{K} \otimes \mathcal{E}$:

$$T = U\left[\mathbb{1}_{\mathcal{H}} \otimes |\phi\rangle_{\mathcal{A}} \right] . \tag{1.52}$$

Then, the decomposition of a CPTP map Φ on qubits as in (1.51) can be further elaborated as being of the form

$$\Phi(\rho) = \mathrm{tr}_{\mathcal{E}} \left(U\left[\rho \otimes |\phi\rangle\langle\phi| \right] U^\dagger \right) = \left(\mathrm{tr}_{\mathcal{E}} \circ \Upsilon \circ \mathsf{A} \right)(\rho) \,, \tag{1.53a}$$

$$\text{where} \quad \Upsilon(\rho) = U \rho U^\dagger, \qquad \mathsf{A}(\rho) = \rho \otimes |\phi\rangle\langle\phi| \,. \tag{1.53b}$$

The characterization of CPTP maps by such decompositions, is an additional motivation (apart from the physical motivation of Schrödinger evolution) to analyze quantum computation in terms of unitary transformations, even if we are interested in CPTP maps in general.

In order to obtain a similar sense of universality as for CPTP maps, it will be useful to discuss a norm on operators analogous to the superoperator norm described above. We define the *operator sup-norm* of an operator $M : \mathcal{H}_Q \longrightarrow \mathcal{H}_{Q'}$ by

$$\|M\|_\infty = \sup_{\substack{\mathbf{v} \in \mathcal{H}_Q \\ \mathbf{v} \neq \mathbf{0}}} \frac{\|M\mathbf{v}\|_2}{\|\mathbf{v}\|_2} \,; \tag{1.54}$$

it is easy to verify that $\|M\|_\infty$ is the largest singular value of M. By a similar derivation as in (1.49), it is easy to prove that

$$\|LM\|_\infty \leqslant \|L\|_\infty \|M\|_\infty \; ; \tag{1.55a}$$

it is also easy to verify that the following properties hold:

$$\|M\|_\infty = 1 \quad \text{for } U \text{ a unitary embedding,} \tag{1.55b}$$
$$\|M\|_\infty = \|M\|_1 \quad \text{for } M \text{ with rank 1,} \tag{1.55c}$$
$$\left\|M^\dagger\right\|_\infty = \left\|M\right\|_\infty \; , \tag{1.55d}$$
$$\|L \otimes M\|_\infty = \|L\|_\infty \|M\|_\infty \; . \tag{1.55e}$$

Then for operators $U, \tilde{U} \in \mathsf{U}(N)$ and $|\psi\rangle \in \mathcal{H}_N \otimes \mathcal{H}_M$ for positive integers N and M, we have

$$\begin{aligned}
&\left\|(\tilde{U} \otimes \mathbb{1}_M) |\psi\rangle\langle\psi| (\tilde{U}^\dagger \otimes \mathbb{1}_M) - (U \otimes \mathbb{1}_M) |\psi\rangle\langle\psi| (U^\dagger \otimes \mathbb{1}_M)\right\|_1 \\
&\quad \leqslant \left\|(\tilde{U} \otimes \mathbb{1}_M) |\psi\rangle\langle\psi| \left([\tilde{U}^\dagger - U^\dagger] \otimes \mathbb{1}_M\right)\right\|_1 + \left\|\left([\tilde{U} - U] \otimes \mathbb{1}_M\right) |\psi\rangle\langle\psi| (U^\dagger \otimes \mathbb{1}_M)\right\|_1 \\
&\quad = \left\|(\tilde{U} \otimes \mathbb{1}_M) |\psi\rangle\langle\psi| \left([\tilde{U}^\dagger - U^\dagger] \otimes \mathbb{1}_M\right)\right\|_\infty + \left\|\left([\tilde{U} - U] \otimes \mathbb{1}_M\right) |\psi\rangle\langle\psi| (U^\dagger \otimes \mathbb{1}_M)\right\|_\infty \\
&\quad \leqslant \left\|\tilde{U}\right\|_\infty \left\||\psi\rangle\langle\psi|\right\|_\infty \left\|\tilde{U}^\dagger - U^\dagger\right\|_\infty + \left\|\tilde{U} - U\right\|_\infty \left\||\psi\rangle\langle\psi|\right\|_\infty \left\|U^\dagger\right\|_\infty \\
&\quad = 2\left\|\tilde{U} - U\right\|_\infty \; .
\end{aligned} \tag{1.56}$$

By convexity, the same inequality holds if we replace $|\psi\rangle\langle\psi|$ with any operator $\rho \in \mathsf{D}(N \times M)$. As a result, for $\Phi(\rho) = U\rho U^\dagger$ and $\tilde{\Phi}(\rho) = \tilde{U}\rho\tilde{U}^\dagger$ we then have

$$\left\|\tilde{\Phi} - \Phi\right\|_\diamond \leqslant 2\left\|\tilde{U} - U\right\|_\infty \; . \tag{1.57}$$

Finally, note that replacing a unitary U with some operator $U' = \mathrm{e}^{i\theta}U$ (for any real θ) in the definition of the map Φ as above yields the same CPTP map Φ. Because our interest in unitary operators arises from the CPTP maps which they induce as above, Theorem 1-3 together with (1.57) motivates the following definition:

Definition 1-15. For two unitary operations $U, \tilde{U} \in \mathsf{U}(N)$ for some $N \geqslant 1$, we say that $\tilde{U}$ *performs* U if $\tilde{U}$ is proportional to U, and that $\tilde{U}$ *approximates* U *with precision* ε if there is an angle $\theta \in \mathbb{R}$ such that $\left\|\tilde{U} - \mathrm{e}^{i\theta}U\right\|_\infty \leqslant \frac{\varepsilon}{2}$. A set $\mathcal{S}$ of unitary operations is *universal for unitary transformation* if any unitary U on n qubits can be performed by a composition of operators from $\mathcal{S}$, for every $n \geqslant 1$; alternatively, $\mathcal{S}$ is *approximately universal for unitary transformation* if any such map U can be approximated to arbitrary precision $\varepsilon > 0$ by some product of operators from $\mathcal{S}$.

This definition differs from the standard treatments of approximate universality (see *e.g.* [83, 103]) by the multiplication of ε by $\frac{1}{2}$, which has been inserted for consistency with Definition 1-14, and by the insertion of the scalar factor $\mathrm{e}^{i\theta}$. The former variation is inconsequential because of the Solovay-Kitaev Theorem, which we discuss below; the latter variation is made to facilitate description of unitary circuits, starting in Section 1.3.

It is useful to note that a similar derivation as in (1.50) allows us to show that errors in approximating unitaries are also additive: if $\left\|\tilde{U}-U\right\|_\infty < \varepsilon$ and $\left\|\tilde{V}-V\right\|_\infty < \varepsilon'$ for $U, V, \tilde{U}, \tilde{V} \in \mathsf{U}(N)$ for some $N \geqslant 1$, we then have

$$\left\|\tilde{U}\tilde{V}-UV\right\|_\infty \leqslant \left\|\tilde{U}\left(\tilde{V}-V\right)\right\|_\infty + \left\|(\tilde{U}-U)V\right\|_\infty < \varepsilon+\varepsilon' . \tag{1.58}$$

Because we will be interested in describing unitary CPTP maps with non-unitary CPTP maps, we may more generally define the following:

Definition 1-16. A set GATES of elementary operations is *universal for unitary transformation* if any unitary CPTP map Φ from n qubits to n qubits, for any $n \geqslant 1$, can be performed by a composition of operations from GATES, and is *approximately universal for unitary transformation* if any such map Φ can be approximated to arbitrary precision $\varepsilon > 0$ by some composition of operations from GATES.

Note that for approximate universality for unitary transformation of a set GATES, the approximating compositions may not themselves be unitary; they may only approach arbitrarily close to being unitary. By Theorem 1-3, a set of elementary gates which is (approximately) universal for unitary transformation, allows for the preparation of states on arbitrarily many qubits, and which allows the tracing-out of qubits, is also (approximately) universal for CPTP transformation.

These results and concepts allow us to reduce the notion of universal quantum computation to (approximate) universality for unitary transformation: but there remains a question of how sensitive quantum computational complexity is to the particular choice of elementary operations. This question was addressed by Kitaev [81, Lemma 4.7], who notes that the result was independently discovered by Robert Solovay: detailed treatment of the result can be found in [42, 83]. The following is a paraphrasing of the result as described by [42]:

Solovay-Kitaev Theorem. *Let $U \in \mathsf{U}(N)$ be a unitary operator for some $N \geqslant 1$, and $\mathcal{S}$ a set of unitary operations which is closed under inversion and generates a dense subgroup of $\mathsf{U}(N)$. Then there is a $\mathrm{polylog}(1/\varepsilon)$-time deterministic algorithm which produces a product $\tilde{U} = \tilde{U}_L \cdots \tilde{U}_2 \tilde{U}_1$, where $L \in \mathrm{polylog}(1/\varepsilon)$ and $\tilde{U}_j \in \mathcal{S}$ for each $1 \leqslant j \leqslant L$, such that $\tilde{U}$ approximates U with precision ε.*

There are several variations on the algorithm, with different complexities for the run-time and the size of the approximating product; the complexity for both described in [83] is $O(\log(1/\varepsilon)^{3+\delta})$ for arbitrary $\delta > 0$. This result holds for any fixed N; however, the running-time of the algorithm for producing an approximating unitary $\tilde{U} \in \mathsf{U}(N)$ (*e.g.* as described by [83, Theorem 8.5]) is exponential in N^2. Nevertheless, from a theoretical point of view, the particular set of unitaries by which we describe unitary circuits is essentially a matter of convenience, provided that we compare only sets of elementary operations which act on a small, and fixed, number of dimensions.

1.3 Unitary circuit models

While quantum computation is likely in practise to involve some aspect of classical control, it is often presented in terms of decompositions of CPTP maps

$$\Phi \quad = \quad \mathsf{M} \circ \Upsilon \circ \mathsf{A}\,, \tag{1.59}$$

where A prepares some number of qubits in a fixed joint state, Υ is a unitary operation on the set of qubits (and in particular does not involve any explicit interaction with any classical data registers), and M is a composition of complete orthonormal (destructive) measurements on some of the qubits and trace-outs. By the discussion following Theorem 1-3 on page 21, every CPTP map can be decomposed in such a form.

In this section, we consider models of computation of this type:

Definition 1-17. A model of quantum computation is a *unitary circuit model* if its elementary gates consist only of preparation of some number of qubits in a fixed initial state, unitary transformations of some number of qubits, complete (orthogonal & destructive) measurements on some number of qubits, and trace-out operations.

Because unitary quantum circuits do not have any operations which can act on classical bits, we may assume without loss of generality that all measurements are performed at the end; similarly, we may assume that any auxiliary qubits are introduced at the beginning, leading to a decomposition as in (1.59).

1.3.1 Particular unitary operations of interest

Throughout the rest of this thesis, we will be interested in referring to the particular unitary operations defined as follows (in matrix form):

Definition 1-18. The *Pauli operators* are the operators $\{\mathbb{1}_2, X, Y, Z\}$, where the latter three operations are defined by

$$X \quad = \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad Y \quad = \quad \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \qquad Z \quad = \quad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}; \tag{1.60}$$

and we define the X-rotation, Y-rotation, and Z-rotation operators for angles $\alpha \in \mathbb{R}$ by

$$R_{\mathsf{x}}(\alpha) \quad = \quad \exp(-iX\alpha/2) \quad = \quad \begin{bmatrix} \cos(\frac{\alpha}{2}) & -i\sin(\frac{\alpha}{2}) \\ -i\sin(\frac{\alpha}{2}) & \cos(\frac{\alpha}{2}) \end{bmatrix}, \tag{1.61a}$$

$$R_{\mathsf{y}}(\alpha) \quad = \quad \exp(-iY\alpha/2) \quad = \quad \begin{bmatrix} \cos(\frac{\alpha}{2}) & -\sin(\frac{\alpha}{2}) \\ \sin(\frac{\alpha}{2}) & \cos(\frac{\alpha}{2}) \end{bmatrix}, \tag{1.61b}$$

$$R_{\mathsf{z}}(\alpha) \quad = \quad \exp(-iZ\alpha/2) \quad = \quad \begin{bmatrix} \mathrm{e}^{-i\alpha/2} & 0 \\ 0 & \mathrm{e}^{i\alpha/2} \end{bmatrix}. \tag{1.61c}$$

Definition 1-19. We define the *controlled-Z transform* $\wedge Z$, the *controlled-X transform* $\wedge X$, the *Hadamard transform* H, and the $J(\alpha)$ *transform* as follows:

$$\wedge Z \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \qquad \wedge X \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{1.62a}$$

$$H \;=\; \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad J(\alpha) \;=\; \frac{1}{\sqrt{2}} \begin{bmatrix} \mathrm{e}^{-i\alpha/2} & \mathrm{e}^{i\alpha/2} \\ \mathrm{e}^{-i\alpha/2} & -\mathrm{e}^{i\alpha/2} \end{bmatrix}. \tag{1.62b}$$

1.3.2 Two related unitary circuit models

We may use the unitaries defined in the previous section to describe two useful families of elementary gates, which are universal or approximately universal for unitary transformation for appropriate choices of angle parameters. We describe how this may be done by summarizing the development presented in [103, Chapter 4]. We start with single-qubit unitaries.

First, note that an arbitrary single-qubit unitary $U \in \mathsf{SU}(2)$ can be decomposed as a product of Y and Z transformations,

$$U \;=\; R_{\mathsf{z}}(\alpha) R_{\mathsf{y}}(\beta) R_{\mathsf{z}}(\gamma), \tag{1.63}$$

for some angles $\alpha, \beta, \gamma \in (-\pi, \pi]$. Note that we may decompose

$$R_{\mathsf{y}}(\beta) \;=\; R_{\mathsf{z}}(\pi/2) H R_{\mathsf{z}}(\beta) H R_{\mathsf{z}}(-\pi/2)\,, \tag{1.64}$$

which implies that

$$U \;=\; R_{\mathsf{z}}(\alpha + \tfrac{\pi}{2})\, H\, R_{\mathsf{z}}(\beta)\, H\, R_{\mathsf{z}}(\gamma - \tfrac{\pi}{2}), \tag{1.65}$$

so that $\mathsf{SU}(2)$ can be generated with Z-rotations and Hadamard transforms alone. For approximation of single-qubit unitaries, following the development in [23], the operations

$$J(\pi/4) \;=\; H R_{\mathsf{z}}(\pi/4), \qquad J(-\pi/4) \;=\; H R_{\mathsf{z}}(-\pi/4) \;=\; H J(\pi/4)^{\dagger} H, \tag{1.66}$$

have eigenvalues $\mathrm{e}^{\pm i\theta}$ for $\theta = \arccos(\frac{2+\sqrt{2}}{4})$, which is an irrational multiple of π. Such an angle then generates a dense subgroup of $\mathbb{R}/2\pi\mathbb{Z}$, the group of real angles modulo complete cycles of 2π.[8] Let $|\pm\hat{\mathsf{n}}\rangle$ represent the unit $\mathrm{e}^{\pm i\theta}$-eigenvectors of $J(\pi/4)$; then, if we define operators

$$R_{\hat{n}}(\alpha) \;=\; \mathrm{e}^{i\alpha}\, |{+\hat{\mathsf{n}}}\rangle\langle{+\hat{\mathsf{n}}}| \;+\; \mathrm{e}^{-i\alpha}\, |{-\hat{\mathsf{n}}}\rangle\langle{-\hat{\mathsf{n}}}|\,, \tag{1.67}$$

[8] We may effectively prove this by the Pigeon-Hole principle. For any $\varepsilon > 0$, let N be the smallest positive integer such that $\frac{\pi}{N} < \varepsilon$. We may divide the interval $[0, 2\pi)$ into N partitions $I_m = [\frac{2m\pi}{N}, \frac{2(m+1)\pi}{N})$, which we use to represent a connected subset $I_m + 2\pi\mathbb{Z}$ of the compact group $\mathbb{R}/2\pi\mathbb{Z}$. Because θ is an irrational multiple of π, $(\theta + 2\pi\mathbb{Z})$ is an element of $\mathbb{R}/2\pi\mathbb{Z}$ with infinite order. As a result, there is at least one interval $I_{\tilde{m}} + 2\pi\mathbb{Z}$ which contains at least two distinct elements t, t' of the group $\langle \theta + 2\pi\mathbb{Z}\rangle \leqslant \mathbb{R}/2\pi\mathbb{Z}$. Then, there are integers K and L such that $(K\theta - L\theta) = (t - t') \in I_0 + 2\pi\mathbb{Z}$ — that is, for some $M \in \mathbb{Z}$, there exists an angle $\theta' = K\theta - L\theta + 2M\pi \in [0, \frac{2\pi}{N})$. Note that any element of $\mathbb{R}$ differs from a multiple of θ' by at most $\frac{\pi}{N} < \varepsilon$. Thus, for any ε and for any $x \in \mathbb{R}/2\pi\mathbb{Z}$, there exists a multiple of $\theta + 2\pi\mathbb{Z}$ which is within distance ε of x.

we may take powers of $J(\pi/4) = R_{\hat{\mathsf{n}}}(\theta)$ to obtain arbitrarily good approximations to $R_{\hat{n}}(\alpha)$ for any angle $\alpha \in (-\pi, \pi]$. Similarly, if $|\pm\hat{\mathsf{m}}\rangle$ represent the unit $\mathrm{e}^{\pm i\theta}$-eigenvectors of $J(-\pi/4)$, and we define the operators

$$R_{\hat{\mathsf{m}}}(\alpha) \;=\; \mathrm{e}^{i\alpha}\,|{+}\hat{\mathsf{m}}\rangle\langle{+}\hat{\mathsf{m}}| \;+\; \mathrm{e}^{-i\alpha}\,|{-}\hat{\mathsf{m}}\rangle\langle{-}\hat{\mathsf{m}}|\,, \tag{1.68}$$

we may take powers of $J(-\pi/4) = R_{\hat{\mathsf{m}}}(\theta)$ to obtain arbitrarily good approximations to $R_{\hat{\mathsf{m}}}(\alpha)$ for any angle $\alpha \in (-\pi, \pi]$. By [103, Section 4.2] (but see [102] for an erratum), because $J(\pi/4)$ and $J(-\pi/4)$ do not commute, we may then approximate any unitary $U \in \mathsf{SU}(2)$ to arbitrary precision using the group generated by $\{J(\pi/4),\, J(-\pi/4)\}$.

The set of operations $\{\wedge X, U\}_{U\in\mathsf{U}(2)}$ is universal for unitary transformation [103, Section 4.5.2]. By the preceding remarks about single-qubit unitaries, we may perform arbitrary n-qubit unitaries using controlled-not transformations, Hadamard transformations, and arbitrary Z-rotations; and using the remarks on approximation of unitary operations in Section 1.2.3, we approximate arbitrary unitaries (up to scalar factors) using controlled-not transformations and $J(\pm\pi/4)$ transformations. Finally, using the following equivalencies,

$$\wedge X \;=\; (\mathbb{1}_2 \otimes H)\,\wedge Z\,(\mathbb{1}_2 \otimes H)\,, \qquad J(0) \;=\; H, \qquad J(\alpha) \;=\; H R_{\mathsf{z}}(\alpha)\,, \tag{1.69}$$

we obtain the following:

Lemma 1-4. *For a set of angles $\mathbb{A} \subseteq \mathbb{R}$, the two parameterized sets of unitary transformations $\{H, R_{\mathsf{z}}(\alpha), \wedge Z\}_{\alpha\in\mathbb{A}}$ and $\{J(\alpha), \wedge Z\}_{\alpha\in\mathbb{A}}$ are both universal for unitary transformation when $\mathbb{A} = \mathbb{R}$, and approximately universal for unitary transformation when $\mathbb{A} = \frac{\pi}{4}\mathbb{Z}$.*

Because $R_{\mathsf{z}}(-\pi/4) = R_{\mathsf{z}}(\frac{7\pi}{4})$ and because we may approximate H to arbitrary precision using $J(\pm\pi/4)$, we have actually shown that the sets $\{H, R_{\mathsf{z}}(\pi/4), \wedge Z\}$, $\{J(0), J(\pi/4), \wedge Z\}$, and $\{J(-\pi/4), J(\pi/4), \wedge Z\}$ are approximately universal for unitary transformation; however, the somewhat weaker statement presented in Lemma 1-4 will be more useful in further discussion, as each set of unitaries described can be described by the same family of angles, and can be used to exactly generate the other.

We define the following unitary CPTP maps corresponding to the unitaries described in Lemma 1-4, acting on density operators $\rho \in \mathsf{D}(2)$ and $\varrho \in \mathsf{D}(2\times 2)$:

$$\mathsf{H}(\rho) \;=\; H\rho H\,, \qquad \mathsf{R}^{\alpha}(\rho) \;=\; R_{\mathsf{z}}(\alpha)\,\rho\, R_{\mathsf{z}}(\alpha)^{\dagger}\,, \tag{1.70a}$$

$$\mathsf{J}^{\alpha}(\rho) \;=\; J(\alpha)\,\rho\, J(\alpha)^{\dagger}\,, \qquad \mathsf{E}\,(\varrho) \;=\; \wedge Z\,\varrho\,\wedge Z\,. \tag{1.70b}$$

The map E we will tend to call an *entangling* or *entangler* operation, referring to the fact that it maps states of two independent qubits to *entangled* states (*i.e.* joint states which are not independent), provided that neither of the qubits are initially in a state which is a convex combination of the standard basis.[9] We may define three more non-unitary

[9] The map $\rho \longmapsto \mathsf{E}\,(\rho)$ is not the only map with this property, of course, but it is the map which will arise most often in this thesis. In particular, E is the only entangling operation which is allowed in the one-way measurement-based model of quantum computing described in Section **??**.

CPTP maps: preparation maps N^{x} and N^{z} for the pure states $|+\rangle = |+\mathsf{x}\rangle$ and $|0\rangle = |+\mathsf{z}\rangle$ respectively,

$$\begin{aligned} \mathsf{N}^{\mathsf{x}}_v : \{1\} &\longrightarrow \mathsf{D}(v)\,, & \mathsf{N}^{\mathsf{z}}_v : \{1\} &\longrightarrow \mathsf{D}(v)\,, \\ \mathsf{N}^{\mathsf{x}}_v(1) &= |+\mathsf{x}\rangle\langle+\mathsf{x}|_v\,, & \mathsf{N}^{\mathsf{z}}_v(1) &= |+\mathsf{z}\rangle\langle+\mathsf{z}|_v\,; \end{aligned} \tag{1.71}$$

and complete measurements M^{z} in the standard basis (discarding the qubit v on which it operates and allocating a bit $\mathsf{s}[v]$ to store the result),

$$\begin{aligned} \mathsf{M}^{\mathsf{z}}_v &: \mathsf{D}(v) \longrightarrow \mathsf{Prob}(\mathsf{s}[v]) \\ \mathsf{M}^{\mathsf{z}}_v(\rho) &= \sum_{r\in\{0,1\}} \langle r|_v\, \rho\, |r\rangle_v \otimes |r\rangle\langle r|_{\mathsf{s}[v]}\,. \end{aligned} \tag{1.72}$$

Then, we define the following sets of gates:

Definition 1-20. For a set $\mathbb{A} \subseteq \mathbb{R}$, an infinite set V of qubits, and an infinite set B of bits, we define the following parameterized sets of elementary gates,

$$\textsc{Phases}(\mathbb{A}) = \left\{ \mathsf{H}_v\,,\ \mathsf{R}^{\alpha}_v\,,\ \mathsf{E}_{u,v}\,,\ \mathsf{N}^{\mathsf{z}}_v\,,\ \mathsf{N}^{\mathsf{x}}_v\,,\ \mathsf{M}^{\mathsf{z}}_v\,,\ \mathrm{tr}_v \;\middle|\; \alpha \in \mathbb{A} \text{ and } v, w \in V \right\}, \tag{1.73a}$$

$$\textsc{Jacz}(\mathbb{A}) = \left\{ \mathsf{J}^{\alpha}_v\,,\ \mathsf{E}_{u,v}\,,\ \mathsf{N}^{\mathsf{z}}_v\,,\ \mathsf{N}^{\mathsf{x}}_v\,,\ \mathsf{M}^{\mathsf{z}}_v\,,\ \mathrm{tr}_v \;\middle|\; \alpha \in \mathbb{A} \text{ and } v, w \in V \right\}. \tag{1.73b}$$

where the map $v \longmapsto \mathsf{s}[v]$ is an injective map from V to B.

The circuit models arising from these two sets of elementary operations are both unitary circuit models; by Lemma 1-4, both are (approximately) universal for quantum computation for an appropriate choice of angles $\mathbb{A} \subseteq \mathbb{R}$. As well, circuits over each gate set may be easily transformed to the other using the equivalences $\mathsf{J}^{\alpha} = \mathsf{H} \circ \mathsf{R}^{\alpha}$ and $\mathsf{R}^{\alpha} = \mathsf{H} \circ \mathsf{J}^{\alpha}$.

1.3.3 Representations of unitary circuits

One of the topics of this thesis is a description of transformations between unitary circuits, and the *one-way measurement-based model* of quantum computation, which we describe in Chapter 2. As well, we will often be interested in presenting unitary circuits diagrammatically. We now present different ways in which unitary circuits may be represented.

Standard representations of unitary circuits

The two conventional ways of representing unitary circuits are as products of unitary operators (as we have been doing so far), and by *circuit diagrams*.

When writing unitary circuits as products of operators, we conventionally fix representation of unitary operators with respect to the standard basis. For instance, for an n-qubit unitary, we would write something of the form

$$U = \left[U_{\mathbf{x}}{}^{\mathbf{y}} \right]_{\mathbf{x},\mathbf{y}\in\{0,1\}^n} \tag{1.74}$$

where superscripts denote row indices and subscripts denote column indices. We identify the bit-positions in the strings $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ with particular qubits:[10] for a sequence $(v_j)_{j=1}^n$ of such qubits, we may then write a particular unitary operation acting on these qubits as $U_{v_1,\ldots,v_n}$. This describes a transformation of pure state vectors, which is how quantum computation is conventionally described for unitary circuit models. Products of such operators are then defined as performing the appropriate operation on the qubits listed as operands, and performing the identity otherwise: for example, we may write

$$W_{b,c}V_aU_{a,b,c}T_b\left|\psi\right\rangle_{b,c} \quad = \quad (\mathbb{1}_2 \otimes W)(V \otimes \mathbb{1}_2 \otimes \mathbb{1}_2)U(\mathbb{1}_2 \otimes \left|\psi\right\rangle) \tag{1.75}$$

for generic unitary operators $W \in \mathsf{U}(8)$, $V, T \in \mathsf{U}(2)$, $U \in \mathsf{U}(8)$, and $\left|\psi\right\rangle \in {\mathcal{H}_2}^{\otimes 2}$, labelling the tensor factors as a, b, and c in order. The operator specified by this notation clearly depends on the order of the composition, but only up to permutations of commuting operators.

Circuit diagrams are an alternative method of representing quantum circuits, as in Figures 1-2 and 1-3. Distinct qubits are represented by distinct *wires* (often drawn as progressing from left to right, as already illustrated in Figure 1-1), schematically representing the trajectory of a qubit. Operations on qubits are then drawn with boxes or other symbols representing multi-qubit interactions. Figure 1-2 illustrates simple examples of quantum circuit diagrams for some generic unitary operators and state vectors in terms of products of operators on specified qubits. This notation allows operators which can be taken in tensor product to be represented more clearly as being simultaneously applicable, but is still strongly dependent on the sequence of operations specified — and has difficulty expressing operations on qubits which are not adjacent in the linear arrangement of qubits in the diagram.

Circuit diagrams readily admit an improvement over the operator notation, by making commutation relations more apparent through the following notational devices. Given an operator U which operates on some collection of n qubits, if the eigenvectors of U can be expressed by vectors $\left|\phi_1\right\rangle \otimes \cdots \otimes \left|\phi_m\right\rangle$ where each $\left|\phi_j\right\rangle$ ranges over (possibly different) orthonormal bases for ${\mathcal{H}_2}^{\otimes n_j}$, where $n_1 + \cdots + n_m = n$ is a partition of n, we may represent U using special symbols on each of the partition classes of qubits corresponding to the tensor decomposition of the eigenvectors. Then, for different operators represented in this way, the operators commute whenever they act on common sets qubits and are represented by common symbols. Standard notational devices are illustrated in Figure 1-3, along with an example which exploits these devices to show the equivalence of two circuits. Such devices can also partially overcome the problem of specifying operations which act on non-adjacent qubits.

Non-standard representations of unitary circuits

One of the main topics of this thesis is translations between different representations of quantum computations. Therefore, it will be useful to have a notation for unitary circuits in which irrelevant details such as the order of commuting gates are essentially

[10] This identification is an injection from the set $\{1,\ldots,n\}$ to some subset L of n distinct qubits among the set of all qubits in the model of computation being considered; this induces a particular isometry between $\mathsf{U}\left({\mathcal{H}_2}^{\otimes n}\right)$ and the unitary group $\mathsf{U}\left({\mathcal{H}_2}^{\otimes V}\right)$, where V is the set of *all* qubits admitted by the model of computation.

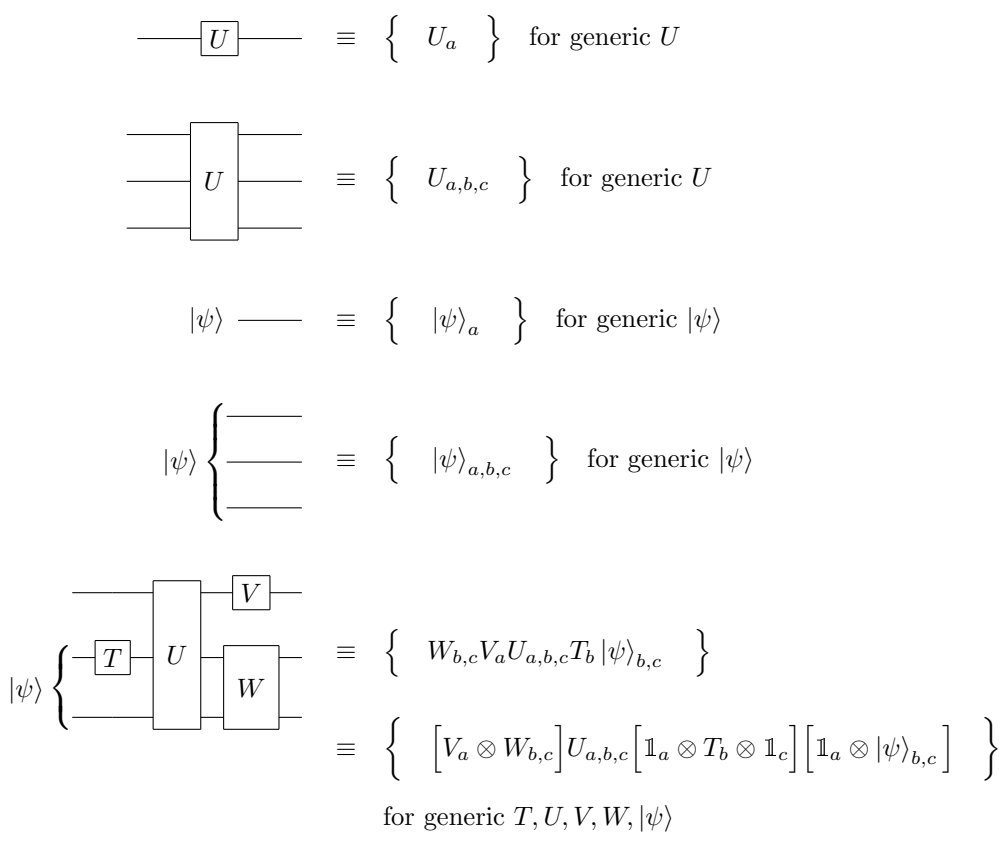


FIGURE 1-2. Examples of simple quantum circuit diagrams. In the operator expressions for these examples, we arbitrarily label the qubits of each circuit from top to bottom in alphabetical order $(a, b, c, \ldots)$. Diagrams compose from left to right, in contrast to the right-to-left composition in written notation; this represents an arrow of time in the diagrams from left to right.

absent, while being at the same time easy to produce by an algorithm. This motivates the definition of an alternative representation for unitary circuits.

One alternative notation (albeit one which is rare in the quantum computing literature) is Einstein summation notation [56, Section B5], where we provide different indices for the domain and range of each operator (*e.g.* as in (1.74) above, for the rows and columns), and identify the output indices of operators with the input indices of any subsequent operator acting on the same qubit. We implicitly sum over any indices which are repeated (representing matrix multiplication of the matrix representation of the operators). Thus, for example, for a composition of operators

$$C \quad = \quad W_{b,c}V_aU_{a,b,c}T_b\,|\psi\rangle_{b,c} \tag{1.76a}$$

for generic unitary operators $W \in \mathsf{U}(8)$, $V, T \in \mathsf{U}(2)$, $U \in \mathsf{U}(8)$, and $|\psi\rangle \in {\mathcal{H}_2}^{\otimes 2}$, we may equivalently write

$$C_{a_0}^{a_2,b_3,c_2} \quad = \quad \left[W_{b_2,c_1}^{\ \ b_3,c_2}\right]\left[V_{a_1}^{a_2}\right]\left[U_{a_0,b_1,c_0}^{\ \ a_1,b_2,c_1}\right]\left[T_{b_0}^{b_1}\right]\left[\,|\psi\rangle^{b_0,c_0}\right]. \tag{1.76b}$$

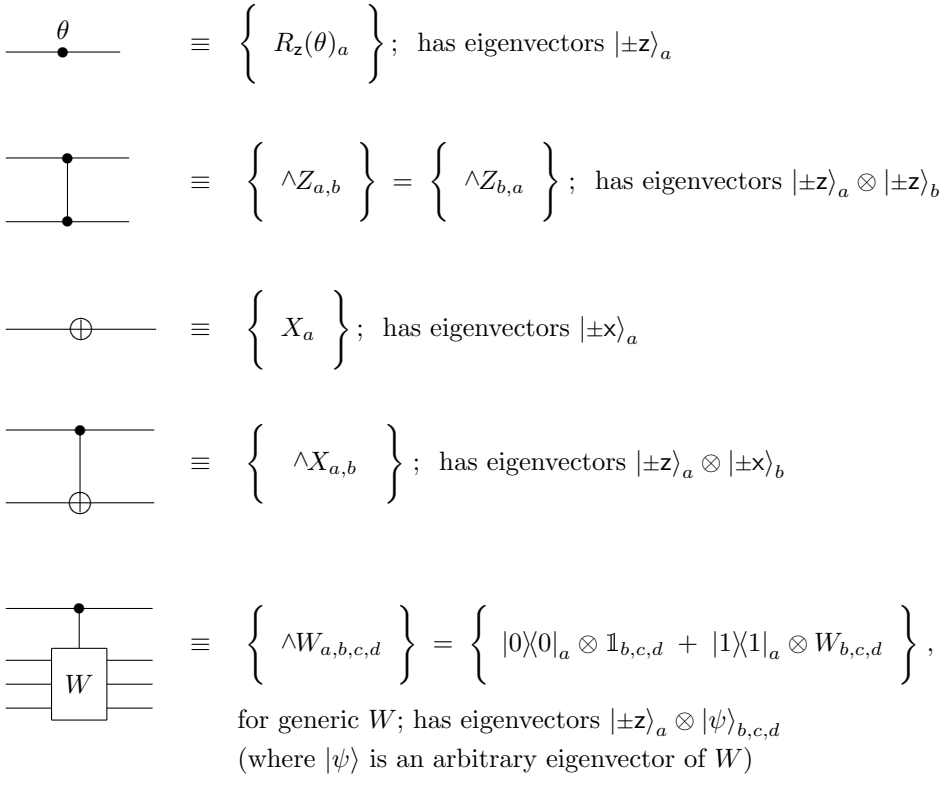


FIGURE 1-3. Examples of special notations for quantum circuit diagrams. In the operator expressions for these examples, we again arbitrarily label the qubits of each circuit from top to bottom in alphabetical order $(a, b, c, \ldots)$. The bottom-most circuits illustrate two equivalent circuits under commutation relations, where these relations are made visually apparent by common symbols on collections of wires.

Indices which are repeated are *bound* variables over which we sum, and indices which are not repeated correspond to *free* variables corresponding to rows (for subscripted indices) or columns (for superscripted indices) of a matrix representation of an operator. (The brackets here are provided for additional emphasis.) Because the sense of composition of tensors here is provided by the indices, we may freely rearrange the factors of such an expression without changing the meaning of the expression, which implies that the following composition of operators

$$\begin{aligned}&\Big(\langle a_2| \otimes \langle b_3| \otimes \langle c_2| \Big) C \, |a_0\rangle \\ &\quad = \sum_{a_1} \left[\sum_{b_0,b_1,b_2} \left(\sum_{c_0,c_1} W_{b_2,c_1}^{\ \ b_3,c_2} \ V_{a_1}^{a_2} \ U_{a_0,b_1,c_0}^{\ \ a_1,b_2,c_1} \ T_{b_0}^{b_1} \ |\psi\rangle^{b_0,c_0} \right) \right] \end{aligned} \tag{1.76c}$$

(where each index a_h, b_j, c_k ranges over $\{0,1\}$) can be expressed as a sum over a commuting product of scalars. Thus, this notation has added flexibility over the conventional operator notation, in that the order of the factors is not significant. However, because the meaning is preserved by indicating the order of the multiplication in the tensor indices, circuits which are equivalent up to rearrangements of commuting operations still give rise to inequivalent expressions: for instance, although it is easy to verify that the operations $\wedge Z_{a,b}$, $\wedge Z_{a,c}$, and $\wedge Z_{b,c}$ commute, the two products

$$\wedge Z_{b,c} \wedge Z_{a,c} \wedge Z_{a,b} \ \equiv \ \wedge Z_{b_1,c_1}^{\ \ b_2,c_2} \ \wedge Z_{a_1,c_0}^{\ \ a_2,c_1} \ \wedge Z_{a_0,b_0}^{\ \ a_1,b_1} \, , \tag{1.77a}$$

$$\wedge Z_{a,c} \wedge Z_{b,c} \wedge Z_{a,b} \ \equiv \ \wedge Z_{a_1,c_1}^{\ \ a_2,c_2} \ \wedge Z_{b_1,c_0}^{\ \ b_2,c_1} \ \wedge Z_{a_0,b_0}^{\ \ a_1,b_1} \, , \tag{1.77b}$$

still give rise to expressions which are not equivalent in Einstein notation by a relabeling of the indices and re-ordering of the terms.

A compromise between Einstein summation notation, and the approach of using special symbols in circuit diagrams, would be an Einstein-like tensor notation in which an index can be repeated multiple times when it corresponds to elements of an orthonormal basis which are preserved by several tensors — that is, where the tensor indices may be interpreted as indicating elements of an orthonormal basis $\{|\phi_j\rangle\}$ of some subset of the qubits, such that for several successive unitary operators U in succession in the circuit, there exist a collection of unitaries V_j for which $U\big(|\phi_j\rangle \otimes |\psi\rangle \big) = |\phi_j\rangle \otimes V_j \, |\psi\rangle$ holds for all states $|\psi\rangle$ of the remaining qubits.

To this end, we may consider a variant of Einstein summation notation, as follows:

Stable index tensor notation. For a unitary operator U acting on qubits $v_1, \ldots, v_n$, let σ and δ be sequences of n_1 and n_2 qubits respectively (where $n = n_1 + n_2$) such that the eigenvectors of U consist entirely of vectors $|\mathbf{s}\rangle_\sigma \otimes |\psi\rangle_\delta$, for standard basis vectors $|\mathbf{s}\rangle$ on the qubits σ and vectors $|\psi\rangle$ which are *not known* to be decomposable as tensor products over the standard basis for qubits in δ. We then write the matrix coefficients of U in the standard basis by

$$U_{\mathbf{x}}^{\ \mathbf{y}} \ = \ U\big[\begin{smallmatrix} \mathbf{s} & \mathbf{a} \\ \mathbf{d} & \end{smallmatrix} \big] \ , \tag{1.78a}$$

where $\mathbf{x} = (\mathbf{s}; \mathbf{d})$ and $\mathbf{y} = (\mathbf{s}; \mathbf{a})$ are decompositions of $\mathbf{x}$ and $\mathbf{y}$ into the appropriate

substings. That is, we define $U\left[\mathbf{s}\,\frac{\mathbf{a}}{\mathbf{d}}\right]$ by the operator equality

$$U \;=\; \sum_{\substack{\mathbf{s}\in\{0,1\}^{n_1}\\ \mathbf{a},\mathbf{d}\in\{0,1\}^{n_2}}} U\left[\mathbf{s}\,\frac{\mathbf{a}}{\mathbf{d}}\right] \;\left(|\mathbf{s}\rangle\langle\mathbf{s}|_\sigma \otimes |\mathbf{a}\rangle\langle\mathbf{d}|_\delta\right). \tag{1.78b}$$

We call the indices of $\mathbf{s}$ *stable*, the indices of $\mathbf{d}$ *deprecated*, and the indices of $\mathbf{a}$ *advanced*. The sequence of the advanced indices correspond to the same qubits being operated on as the sequence of deprecated qubits, in order. A product of such expressions is well-formed if each index is advanced at most once and deprecated at most once; and we sum implicitly over indices which are both advanced and deprecated in a product expression.

A further shorthand notation. For a generic unitary operator $U \in \mathsf{U}(2^n)$ which is not known to have eigenvectors of the form $|s\rangle_{v_1} \otimes |\psi\rangle_{v_2,\ldots,v_n}$ for arbitrary standard basis states $|s\rangle = |\pm\mathsf{z}\rangle$, for any qubit v_1 that U acts upon, we may represent U by the symbol $U\left[\frac{\mathbf{a}}{\mathbf{d}}\right]$ (*i.e.* without stable indices). Conversely, if the eigenvectors of such an operator U are all standard basis vectors, we may represent U by the symbol $U\left[\mathbf{s}\right]$ (*i.e.* without advanced or deprecated indices). Finally, for a unitary embedding, there may be advanced indices which do not correspond to any deprecated indices; we may pad out the sequence of deprecated indices by a space, dot, or similar placeholder in this case.

Example 1. For a product of generic operators $C = W_{b,c}V_aU_{a,b,c}T_b\,|\psi\rangle_{b,c}$ as described in (1.76a), we may write

$$C\left[\tfrac{a_2,b_3,c_2}{a_0,\cdot,\cdot}\right] \;=\; W\left[\tfrac{b_3,c_2}{b_2,c_1}\right] V\left[\tfrac{a_2}{a_1}\right] U\left[\tfrac{a_1,b_2,c_1}{a_0,b_1,c_0}\right] T\left[\tfrac{b_1}{b_0}\right] |\psi\rangle\left[\tfrac{b_0,c_0}{}\right]. \tag{1.79}$$

Because the operators are not specified as preserving standard basis vectors over any of their operands, there are no stable indices; then, the meaning of this expression essentially reduces to that of Einstein notation, where we just sum over repeated indices (*i.e.* indices which are advanced in some factor and deprecated in some factor).

Example 2. For some generic unitary $W \in \mathsf{U}(4)$, we may define the operator

$$\wedge W \;=\; |0\rangle\langle 0| \otimes \mathbb{1}_2^{\otimes 2} \;+\; |1\rangle\langle 1| \otimes W \tag{1.80}$$

analogously as in the example of Figure 1-3; then, the stable index representations of the circuits $\wedge X_{a,d}\wedge W_{a,b,c}$ and $\wedge W_{a,b,c}\wedge X_{a,d}$ on four qubits may be given by

$$\wedge X\left[a\,\tfrac{d_1}{d_0}\right] \wedge W\left[a\,\tfrac{b_1,c_1}{b_0,c_0}\right] \qquad \text{and} \qquad \wedge W\left[a\,\tfrac{b_1,c_1}{b_0,c_0}\right] \wedge X\left[a\,\tfrac{d_1}{d_0}\right], \tag{1.81}$$

respectively. Note that these are equivalent up to permutations of the factors.

The informal semantics of stable index tensor notation is to provide a description of circuits in terms of computational paths. In particular, stable index notation for circuits was partially inspired by the *path-labelling* of circuits in [41], which is explicitly concerned

with descriptions of circuits in terms of computational paths: the tensor indices here correspond to labels of "wire segments" in that article, which are separated by *e.g.* Hadamard gates, or other operations which do not preserve the standard basis. For a given operator, the stable indices then represent qubits for which each vector of the standard basis is preserved, with coefficients being assigned (conditioned on a particular value of the stable indices) to transitions from certain values of the deprecated indices to various values of the advanced indices. For products of operators which preserve standard basis vectors on a common set of qubits, it is not necessary (as is done in Einstein notation) to introduce distinct indices for the domain and the range of the operator, given that the coefficients for all of the cross-terms will be zero. Stable index tensor notation then consists simply of elaborating Einstein notation by so-called "stable" indices.

By construction, there is a close analogy between *stable, deprecated, and advanced indices* in the stable index notation, and *stable, discarded, and allocated qubits* as described in Definition 1-11. This correspondence could be made exact if we imposed the constraint for any CPTP map Φ, we discard any qubit for which Φ does not preserve the marginal distributions $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$, substituting it if desired by a differently distributed qubit, and never re-allocating any qubit which have already been discarded. This correspondence will play a useful role in our discussion of the one-way measurement-based model in Chapter 2.

Constructing stable index representations. We may automatically generate a stable index representation S for a unitary circuit from a representation as a composition of operators C on specified qubits, as follows. Let L be the set of qubit labels in C. We may then transliterate the terms in C, defining a set of tensor indices incrementally as we do so, as follows:

1. For the first operator in C to act on a given qubit v, we define a tensor index v_0 , and designate v_0 as *the current tensor index* for v.

2. For any operator $U_{a,b,\ldots}$ in C, we translate $U_{a,b,\ldots}$ into a stable index term $U\left[\mathbf{s}_U \frac{\mathbf{a}_U}{\mathbf{d}_U}\right]$ with tensor indices as follows. For any qubit v upon which U acts, let v_j be the current tensor index for v (for some $j \in \mathbb{N}$).

   - If $U_{a,b,\ldots}$ is a unitary embedding which allocates v, then v_0 is an advanced index in $\mathbf{a}_U$, without a corresponding deprecated index.
   - If $U_{a,b,\ldots}$ is a unitary embedding which takes a state on v as an input, and $U_{a,b,\ldots}$ does not preserve standard basis states on v, then we define a *new* current tensor index v_{j+1} for v; then v_{j+1} is an advanced index in $\mathbf{a}_U$, corresponding to the deprecated index v_j in $\mathbf{d}_U$.
   - If $U_{a,b,\ldots}$ is a unitary embedding which takes a state on v as an input, and $U_{a,b,\ldots}$ preserves standard basis states on v, then v_j remains the current tensor index for v, and is a stable index in $\mathbf{s}_U$.

   In particular, every new index which is advanced by a gate $U\left[\mathbf{s}\,\frac{\mathbf{a}}{\mathbf{d}}\right]$ must be distinct from any other index being advanced, as well as from all of the indices which have occurred to that point. Performing this transliteration of operators, sequentially for all terms in C, produces the corresponding stable index tensor expression S.

For a given circuit decomposition C of a unitary operation, we will refer to the above construction (or one which differs from it only by a relabelling of indices) as the stable index tensor expression S corresponding to C. This construction leads to a natural (partial) mapping f on the set $V(S)$ of indices of a stable index tensor expression S; for each index v_j, we define $f(v_j) = v_{j+1}$ if the latter is well-defined. More generally, for each index v which is deprecated in some term of S, we define $f(v)$ to be the corresponding index which is advanced. If I is the set of "input indices" (ones which are not advanced by any term in S) and O the set of "output indices" (which are not deprecated by any term in S), f is then an injective map from $V(S) \setminus O$ to $V(S) \setminus I$ by construction.

Example 2 on page 32 illustrates the intuitive purpose of stable index notation: by allowing stable indices to be repeated multiple times while remaining "free" variables, we also discard some redundant combinatorial information about the order of the product of operators. If we allow the terms of the product to be permuted arbitrarily, information about order of commuting operators is then lost completely. Circuits which are congruent up to the re-ordering of commuting gates may then give rise to "equivalent" stable index tensor expressions, in the following sense:

Definition 1-21. A *homomorphism* of stable index tensor expressions is a bijection λ of the *index labels* of one stable index expression S_1 to those of another such expressions S_2, such that for any vector $(\mathbf{s}; \mathbf{a}; \mathbf{d})$ of index labels for which there is an operator $U\big[\mathbf{s}\ \frac{\mathbf{a}}{\mathbf{d}}\big]$ in S_1, there is an operator $W\big[\lambda(\mathbf{s})\ \frac{\lambda(\mathbf{a})}{\lambda(\mathbf{d})}\big]$ in S_2, where we write $\lambda(\mathbf{s}) = \lambda(s_j)_{j=1}^n = \big(\lambda(s_1), \ldots, \lambda(s_n)\big)$, and similarly for $\lambda(\mathbf{a})$ and $\lambda(\mathbf{d})$. An *isomorphism* of stable index tensor expressions is such a homomorphism where there is also a bijective mapping τ of *terms* of S_1 to those of S_2, where we require

$$\tau\left(U\big[\mathbf{s}\ \tfrac{\mathbf{a}}{\mathbf{d}}\big]\right) \;=\; U\Big[\lambda(\mathbf{s})\ \tfrac{\lambda(\mathbf{a})}{\lambda(\mathbf{d})}\Big] \tag{1.82}$$

for each term $U\big[\mathbf{s}\ \frac{\mathbf{a}}{\mathbf{d}}\big]$ in S_1.

Remark. Homomorphisms of stable index representations are *combinatorial* homomorphisms, not *algebraic* ones. In particular, for two isomorphic stable index representations $S = U_n\big[*\big] \cdots U_2\big[*\big]\, U_1\big[*\big]$ and $S' = U'_n\big[*\big] \cdots U'_2\big[*\big]\, U'_1\big[*\big]$, whose isomorphism is witnessed by an appropriate bijection τ of the terms, it need not be the case that $\tau\big(U_j\big[*\big]\ \big) = U'_j\big[*\big]$.

Isomorphic stable index expressions represent not only equivalent unitary operations (which follows from the implicit summation convention), but equivalent circuit decompositions as well:

Lemma 1-5. *Let C_1 and C_2 be two sequences of unitary operators on a common set of qubits, composed from a common set of unitaries. If the stable index tensor representations of C_1 and C_2 are isomorphic, then C_1 and C_2 are congruent up to re-ordering of commuting operators.*

Proof —

Consider the coarsest equivalence relation $\cong$ on circuits which consist of permutations of the gates of C_1, such that $C \cong C'$ if C differs from C' by a transposition

of commuting gates. By definition, the equivalence classes of $\cong$ are sets of circuits which are congruent up to re-ordering of commuting operators. Consider stable index tensor expressions S and S' arising from two unitary circuits C and C', where S acts on the same index set as S', and S differs from S' only by a transposition of two terms $U_1\left[\mathbf{s}_1 \frac{\mathbf{a}_1}{\mathbf{d}_1}\right]$ and $U_2\left[\mathbf{s}_2 \frac{\mathbf{a}_2}{\mathbf{d}_2}\right]$, where without loss of generality the former precedes the latter in C, and vice-versa in C'. Then, the corresponding gates U_1 and U_2 in C occur with U_1 preceding U_2, and occur in C' in the opposite order. Consider the index sets for these two operations in the stable index tensor expression:

- If the sequences of indices $(\mathbf{s}_1; \mathbf{a}_1; \mathbf{d}_1)$ and $(\mathbf{s}_2; \mathbf{a}_2; \mathbf{d}_2)$ do not overlap, the operations U_1 and U_2 act on disjoint sets of qubits, and so they commute.
- Suppose the sequences of indices $(\mathbf{s}_1; \mathbf{a}_1; \mathbf{d}_1)$ and $(\mathbf{s}_2; \mathbf{a}_2; \mathbf{d}_2)$ overlap, and let v be an index occurring in both sequences. In the construction of S, because v occurs in $U_2\left[\mathbf{s}_2 \frac{\mathbf{a}_2}{\mathbf{d}_2}\right]$, it cannot be a deprecated index in $U_1\left[\mathbf{s}_1 \frac{\mathbf{a}_1}{\mathbf{d}_1}\right]$; and because it occurs in $U_1\left[\mathbf{s}_1 \frac{\mathbf{a}_1}{\mathbf{d}_1}\right]$, it cannot be an advanced index in $U_2\left[\mathbf{s}_2 \frac{\mathbf{a}_2}{\mathbf{d}_2}\right]$. By the construction of S', we similarly have that v cannot be a deprecated index in $U_2\left[\mathbf{s}_2 \frac{\mathbf{a}_2}{\mathbf{d}_2}\right]$ or an advanced index in $U_1\left[\mathbf{s}_1 \frac{\mathbf{a}_1}{\mathbf{d}_1}\right]$. Then, v is a stable index of both terms.

Therefore, the only qubits v on which U_1 and U_2 both act are those whose standard basis states are preserved by both U_1 and U_2. Then, U_1 and U_2 commute, in which case $C \cong C'$.

Let S_1 and S_2 be the stable index tensor representations of C_1 and C_2, where the isomorphism is given by an index relabelling map λ and mapping of the terms τ. Let S_2' be the representation obtained by applying λ to all of the indices in S_2: then S_1 and S_2' consist of a product of the same terms in different orders. Without loss of generality, we will then suppose that S_2 and S_1 differ only by a reordering of terms. Because the terms of S_1 differ from those of S_2 by a permutation, by induction on the decomposition of this permutation into transpositions, we therefore have $C_1 \cong C_2$. ■

Rearranging the terms of a stable index tensor expression for a circuit not only preserves the meaning (by summation over indices which are both advanced and deprecated) as a unitary operator, but also preserves structural information about the circuit itself, while discarding information about the ordering of commuting operators. As a consequence, stable index notation facilitates identification of congruent circuits; and we may freely re-arrange the terms of such an expression while preserving a congruence class of unitary circuits.

However, not all congruent circuits have isomorphic tensor index expansions: congruent circuits may still yield non-isomorphic stable index expressions when the common operands of two commuting gates do not have their standard basis states preserved.

Example 3. The two circuits $\wedge X_{c,b} \wedge X_{a,b} H_a$ and $\wedge X_{a,b} \wedge X_{c,b} H_a$ on three qubits a, b, c have the following respective stable index expressions:

$$\wedge X\left[c \frac{b_3}{b_2}\right] \wedge X\left[a_1 \frac{b_2}{b_1}\right] H\left[\frac{a_1}{a_0}\right] \qquad \text{and} \qquad \wedge X\left[a_1 \frac{b_3}{b_2}\right] \wedge X\left[c \frac{b_2}{b_1}\right] H\left[\frac{a_1}{a_0}\right] . \tag{1.83}$$

These two expressions are non-isomorphic: in the first expression, the leftmost factor contains the indices c (which is neither advanced nor deprecated in the entire expression) and b_3 (which is not deprecated in the entire expression); the second expression does not contain any factor with these properties. The inequivalence of the two expressions above is due to the fact that the two operations $\wedge X_{a,b}$ and $\wedge X_{b,c}$ preserve the basis $|\pm\mathsf{x}\rangle$ on b, rather than the basis $|\pm\mathsf{z}\rangle$.

One solution to this would be to diagonalize every element of U of a circuit, decomposing them into diagonal operations flanked by "virtual" unitaries performing a some change of basis from the standard basis to the eigenbasis of U, essentially representing changes in reference frame rather than "real" unitary transformations performed by the circuit.[11] Interpreting the "virtual" unitaries strictly as a change of reference frame rather than as part of the evolution of the system, the diagonal representations of each gate U would then represent preservation of the eigenbasis of U in place of the standard basis. For sequences of operators which commute, a judicious choice of change-of-basis operators would then allow the diagonalized gates to share stable indices in common. However, we can avoid such decompositions into "real" and "virtual" unitary operations if we consider elementary gate sets for which the commutation relations[12] are sufficiently restricted:

Definition 1-22. A set S of unitary operations is *parsimonious* if the eigenvectors of every multi-qubit gate in S are standard basis states, and if distinct gates in S which act on the same qubits commute if and only if their eigenvectors are standard basis states.[13]

For example, for any set $\mathbb{A} \subseteq \mathbb{R}$, the two universal sets of unitaries $\{H, R_{\mathsf{z}}(\alpha), \wedge Z\}_{\alpha\in\mathbb{A}}$ and $\{J(\alpha), \wedge Z\}_{\alpha\in\mathbb{A}}$ described in Section 1.3.1 are parsimonious. This is evident for the former set of unitaries; for the latter, it is sufficient to note that

$$\begin{aligned} J(\alpha)^\dagger J(\beta)^\dagger J(\alpha) J(\beta) &= R_{\mathsf{z}}(-\alpha)\, H R_{\mathsf{z}}(-\beta)\, H\, H\, R_{\mathsf{z}}(\alpha)\, H R_{\mathsf{z}}(\beta) \\ &= R_{\mathsf{z}}(-\alpha) R_{\mathsf{x}}(\alpha-\beta) R_{\mathsf{z}}(\beta)\,, \end{aligned} \tag{1.84}$$

which is equal to $\mathbb{1}_2$ if and only if $\alpha = \beta$.

The restrictions imposed on parsimonious sets of unitary operations make it simple to determine whether two operations on some collection of qubits commute. This allows us to prove the following:

Lemma 1-6. *Let C_1 and C_2 be two unitary circuits composed from a common parsimonious set of unitaries. If C_1 is congruent to C_2 up to re-ordering of commuting operators, then the stable index representations of C_1 and C_2 will be isomorphic.*

[11] It is plausible that the "virtual" gates may represent in some sense *actual* work that would be required to change the Hamiltonian of a physical system, in order to perform time-dependent unitary evolution in some physical implementations. However, such issues lie outside the scope of this thesis.

[12] In this context, by "commutation relations" we mean the relation of *whether* or not two unitaries commute, for all pairs of unitaries in the set, without any further comment on *e.g.* the values of commutators of these operators.

[13] The title of [39] predates this definition of "parsimonious" by four years, and does not refer to the concept we have defined here; however, as we show immediately below, the set of unitary operations described there is parsimonious in this sense.

Proof —

Let $K_0 \cong K_1 \cong \cdots \cong K_\ell$ be a sequence of circuits which differ only by a transposition of two commuting gates, where $C_1 = K_0$ and $C_2 = K_\ell$. We may show by induction on ℓ that the stable index tensor expressions for C_1 and C_2 are isomorphic by induction on ℓ by proving the case $\ell = 1$.

Let S_0 and S_1 be the stable index representations of K_0 and K_1. Up to a relabeling, we may assume that the indices of S_0 and S_1 are all of the form $v_0, v_1, \ldots$ for different qubits v acted on by K_0 and K_1: we set v_0 to be the first index corresponding to a given qubit in either S_0 or S_1, and v_{j+1} to be an advanced index corresponding to each deprecated index v_j for $j \in \mathbb{N}$. Let U_1 and U_2 be the pair of gates whose order differs between the two circuits; and consider the truncation S_0' of S_0 just prior to the terms corresponding to U_1 and U_2, and similarly for S_1'. Because K_0 and K_1 differ only by a transposition of commuting gates U_1 and U_2, $S_0' = S_1'$. Let S_0'' and S_1'' be the truncations of S_0 and S_1 just after the terms corresponding to U_1 and U_2 :

- If U_1 and U_2 act on disjoint sets of qubits, the terms corresponding to U_1 and U_2 in S_0'' may differ from those in S_1'' only by the indices which are advanced in each; and as the indexing scheme we have fixed is the same for each, and the indices which will be advanced for U_1 and U_2 are independent of each other in both expressions, S_0'' differs from S_1'' only by a transposition of those two terms.

- If U_1 and U_2 both have the standard basis as their eigenbases, their corresponding terms in S_0 and S_1 only have stable indices, and therefore do not advance any new indices. Then the term in S_0 and in S_1 corresponding to U_1 are the same, and similarly for U_2 ; the difference between S_0'' and S_1'' is a transposition of those two terms, and so they are isomorphic.

- If U_1 and U_2 perform the same single-qubit operation, there will be some common tensor index u_j which is current for that qubit prior to the application of U_1 and U_2 in both S_0' and S_1'. Then, the terms corresponding to U_1 and U_2 in S_0 will be $U_2\left[\frac{u_{j+2}}{u_{j+1}}\right] U_1\left[\frac{u_{j+1}}{u_j}\right]$, and the corresponding terms in S_1 will be $U_1\left[\frac{u_{j+2}}{u_{j+1}}\right] U_2\left[\frac{u_{j+1}}{u_j}\right]$ for some indices v'' and w''. Given that U_1 and U_2 perform the same operation, we then have $S_0'' = S_1''$.

Because the operations of K_0 and K_1 are identical after U_1 and U_2, the differences between S_0 and S_1 after the terms corresponding to U_1 and U_2 can only arise from the indices advanced by each term; because we have fixed the indexing scheme, the only difference between S_0 and S_1 is then the (possibly trivial) transposition of those two terms. Thus, S_0 and S_1 are isomorphic. ■

Thus, for unitary circuits constructed from a parsimonious set of gates, isomorphism of stable index tensor expressions is equivalent to congruence up to rearrangements of commuting gates. We will take advantage of this feature of this tensor notation in later chapters of the thesis.

We conclude our discussion of stable tensor index expressions with two remarks.

1. **Recovering the order of unitary operations.** As we have noted, isomorphism classes of stable tensor index expressions (for a parsimonious set of operations) correspond to congruency classes of unitary circuits under permutations of commuting gates. In any such circuit, the order of *non-commuting* gates can be determined from the indexing: for a tensor index v_j which is deprecated in a given stable index expression and for $v_{j+1} = f(v_j)$, any operation acting on v_{j+1} must occur after any (other) operation in the circuit acting on v_j , with the only operator acting on both being the operator which deprecates v_j and correspondingly advances v_{j+1} . The complement of the relation "U must occur after V" is then a pre-order[14] which characterizes the *possible* orderings of the operations in a stable-index expression, up to commuting operations.

2. **Induced combinatorial structures on the indices.** The operations of a stable-index tensor expression also induce structures on the tensor indices. From any stable-index tensor expression S, we may construct a graph G whose vertices are the indices of S, and where $vw \in E(G)$ for two indices v and w if and only if there exists an operator in S involving both v and w: we may call such a graph an *interaction graph* for a stable index expression. If I is the set of non-advanced indices of S, and O is the set of non-deprecated indices, the function f which maps each deprecated index to an advanced index then describes vertex-disjoint $I - O$ directed paths in G.

   The interaction graph G encodes much of the information of the original expression S: in particular, from G and the function f, we may recover the order in which tensor indices are deprecated (if they are deprecated at all) in any ordering of the terms of of S consistent with a conventional circuit notation. Expressed as a partial order $\preccurlyeq$, we clearly have $v \preccurlyeq f(v)$, as $f(v)$ is advanced in the same term which deprecates v; and for any w which is adjacent to $f(v)$, we have $v \preccurlyeq w$, as $f(v)$ and w are acted on by an operator U in common and therefore must be current at the same time at the stage where U is performed. These same graph and ordering structures arise in the one-way model, and form the core of the topic of Chapter 3.

1.4 Classically controlled unitary circuits

We may extend unitary circuits to obtain a description of quantum computation which involves *classical control*: that is, dependency of operations on classical bits, which in particular may be the results of measurements. In this section, we briefly consider such models of quantum computation, and their relationship to unitary circuits without classical control. This will also allow us to define operations which will be necessary to describe the *one-way measurement model* in Chapter 2.

[14] A pre-order on a class S is a binary relation which is reflexive ($x \preccurlyeq x$ for all $x \in S$) and transitive ($x \preccurlyeq y$ and $y \preccurlyeq z$ implies $x \preccurlyeq z$ for all $x, y, z \in S$). An instructive example from quantum information processing is the binary relation "$\rightarrow$" on bipartite pure states $|\Psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$, where $|\Phi\rangle \rightarrow |\Psi\rangle$ if and only if $|\Psi\rangle$ can be obtained by LOCC on $\mathcal{H}_A$ and $\mathcal{H}_B$ from $|\Phi\rangle$. Every state can be obtained via LOCC from itself, and if $|\Phi\rangle \rightarrow |\Psi\rangle$ and $|\Psi\rangle \rightarrow |\Upsilon\rangle$, then $|\Phi\rangle \rightarrow |\Upsilon\rangle$ by composing protocols. There are also pairs of states with $|\Phi\rangle \rightarrow |\Psi\rangle$ and $|\Psi\rangle \not\rightarrow |\Phi\rangle$, *e.g.* for $|\Phi\rangle$ an entangled state and $|\Psi\rangle$ a product state; and there are distinct states with $|\Phi\rangle \rightarrow |\Psi\rangle \rightarrow |\Phi\rangle$, *i.e.* if $|\Phi\rangle$ and $|\Psi\rangle$ only differ by local unitaries. (A result of Nielsen [99] shows that that there also exist pairs of states such that $|\Phi\rangle \not\rightarrow |\Psi\rangle$ and $|\Psi\rangle \not\rightarrow |\Phi\rangle$, so this pre-order exhibits the complete spectrum of possible relations between pairs of elements.)

We may define a "classically controlled" CPTP operation using the following construction:

Definition 1-23. For a sequence of bit-registers $\mathbf{c} = (c_j)_{j=1}^n$ and a map $f : \{0,1\}^n \to S$ for some set S, define the completely positive (but *not* trace-preserving) operators $\textsc{Cond}^{f,s} : \mathsf{Prob}(\mathbf{c}) \longrightarrow \mathsf{Prob}(\mathbf{c})$ for each $s \in S$, given by

$$\textsc{Cond}^{f:s}(\rho) \;=\; \Pi_s \,\rho\, \Pi_s \;, \qquad \text{where} \quad \Pi_s \;=\; \sum_{\mathbf{x} \in f^{-1}(s)} |\mathbf{x}\rangle\langle\mathbf{x}| \,. \tag{1.85}$$

For sequence of generic registers σ, α, and δ, a *classically controlled* CPTP map is a sum of superoperators, of the form

$$\Phi^{f(\mathbf{c})}_{\sigma:\alpha/\delta} : \mathsf{Prob}(\mathbf{c}) \otimes \mathsf{D}(\sigma;\delta) \longrightarrow \mathsf{Prob}(\mathbf{c}) \otimes \mathsf{D}(\sigma;\alpha) \tag{1.86a}$$

$$\Phi^{f(\mathbf{c})}_{\sigma:\alpha/\delta}(\rho) \;=\; \sum_{\mathbf{x} \in \{0,1\}^n} \textsc{Cond}^{f:\mathbf{x}}_{\mathbf{c}} \otimes \Psi^{(\mathbf{x})}_{\sigma:\alpha/\delta} \;, \tag{1.86b}$$

for some family of *conditionally applied CPTP maps* $\left\{\Psi^{(s)}\right\}_{s\in S}$ which all have the same domain and range.

A common example of a classically controlled operation is a *classically-controlled not* operation, where we perform an X operation on a qubit depending on the value of a classical bit:

$$\begin{aligned} \Phi_{c,q}(\rho) \;&=\; \Big[\, |0\rangle\langle 0|_c \otimes \mathbb{1}_q \Big] \rho \Big[\, |0\rangle\langle 0|_c \otimes \mathbb{1}_q \Big] \;+\; \Big[\, |1\rangle\langle 1|_c \otimes X_q \Big] \rho \Big[\, |1\rangle\langle 1|_c \otimes X_q \Big] \\ &=\; \Big[\Big(\textsc{Cond}^{\mathrm{id}:0}_c \otimes \mathbb{1}_q \Big) + \Big(\textsc{Cond}^{\mathrm{id}:1}_c \otimes \mathsf{X}_q \Big) \Big](\rho), \end{aligned} \tag{1.87}$$

where $\mathsf{X} : \mathsf{D}(2) \longrightarrow \mathsf{D}(2)$ is given by $\mathsf{X}(\rho) = X\rho X$. It is easy to see that classically controlled CPTP maps Φ are indeed CPTP maps, by forming the Kraus operators of Φ from the tensor products of the $\textsc{Cond}^{f:s}$ operations and the conditionally applied maps $\Psi^{(s)}$ which define Φ.

We may distinguish two classes of conditionally controlled CPTP maps, which will play an important role in this thesis:

Definition 1-24. A *classically controlled unitary* operation is a conditionally controlled CPTP map Φ whose conditionally applied maps $\left\{\Psi^{(s)}\right\}_{s\in S}$ are all unitary operations. Similarly, a *classically controlled measurement* operation is a conditionally controlled CPTP map Φ whose conditionally applied maps $\left\{\Psi^{(s)}\right\}_{s\in S}$ are all measurement operations.

Our interest in unitary circuit models (as opposed to models with classical control) is largely due to the convenience of omitting explicit reference to classical control in the analysis of quantum algorithms. However, as instances of computational problems are usually described as being represented with classical information, a robust experimental set-up which is capable of solving different instances of a problem will very probably control which particular unitary evolution occurs by classical control of the unitary evolution.

Definition 1-25. A model of quantum computation is a *classically controlled unitary circuit model* if its elementary gates consist only of operations which

(a) prepare some number of qubits in a fixed initial state;

(b) perform a (possibly classically-controlled) unitary operation on some number of qubits; or

(c) perform a (possibly classically-controlled) complete (orthogonal & destructive) measurement on some number of qubits.

1.4.1 Classically controlled extensions, and deferred measurement

For a given unitary circuit model defined by some gate-set GATES, it is common to extend to a classically controlled unitary circuit model by including classically controlled versions Υ^c of the unitary gates $\Upsilon \in \text{GATES}$, which perform either Υ or the identity superoperator depending on the value of a single classical bit c. We may also consider classically controlled versions K^c of measurement gates, where $\mathsf{K}^0, \mathsf{K}^1 \in \text{GATES}$ are different measurement gates which are performed on a given set of qubits depending on the value of a classical bit c. Such an extended gate set, including controlled versions of unitaries and measurements in GATES, we will call CC(GATES).[15] In such a circuit model, classically controlled CPTP maps may either perform the identity or perform a non-trivial operation on a set of qubits, depending on the results of measurements performed in the middle of the circuit. This is the primary qualitative difference between unitary circuit models and classically controlled versions of those models.

It is also common to transform classically-controlled circuits into unitary circuits without classical control, following a folklore result reported in [19] and attributed to [17]. For a gate-set GATES, we define a gate-set QC(GATES) as follows:

- If GATES doesn't include the measurement operator M^{z}, we include it in QC(GATES) ;
- For any unitary $\Upsilon \in \text{GATES}$ performing $\rho \longmapsto U\rho U$ for $\rho \in \mathsf{D}(2^n)$, we include a (*coherently*) *controlled* version $\wedge\Upsilon \in \text{QC}(\text{GATES})$, which performs the mapping

$$\wedge\Upsilon(\rho) \;=\; \Big(|0\rangle\langle 0| \otimes \mathbb{1}_2^{\otimes n} \;+\; |1\rangle\langle 1| \otimes U\Big)\rho\Big(|0\rangle\langle 0| \otimes \mathbb{1}_2^{\otimes n} \;+\; |1\rangle\langle 1| \otimes U^\dagger\Big); \quad (1.88)$$

- For any complete measurement K given by

$$\mathsf{K}(\rho) \;=\; \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle\langle\psi_{\mathbf{x}}|\,\rho\,|\psi_{\mathbf{x}}\rangle\langle\mathbf{x}| \quad (1.89)$$

for some orthonormal basis $\big\{\,|\psi_{\mathbf{x}}\rangle\,\big\}_{\mathbf{x}\in\{0,1\}^n}$ of ${\mathcal{H}_2}^{\otimes n}$, we include the unitary change of basis operation $\mathsf{B}^{\mathsf{K}} \in \text{QC}(\text{GATES})$, defined by

$$\mathsf{B}^{\mathsf{K}}(\rho) \;=\; B\rho B^\dagger, \qquad \text{where} \quad B \;=\; \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle\langle\psi_{\mathbf{x}}| \;. \quad (1.90)$$

We also include a coherently controlled version $\wedge\mathsf{B}^{\mathsf{K}} \in \text{QC}(\text{GATES})$, defined in terms of B^{K} by (1.88).

[15] We may easily extend this definition to CC(GATES) when GATES does not define a unitary circuit model, by considering any pair of conditionally applied CPTP maps $\Psi^0, \Psi^1 \in \text{GATES}$; however, when Ψ^0 and Ψ^1 are unitaries, it is conventional to require that one of them be the identity superoperator.

- Every state-preparation, unitary operation, or trace-out operation in GATES is included in QC(GATES), but not any measurement operation except for M^{z}.

Having defined the gate-set QC(GATES), we may translate any circuit composed from the gate set CC(GATES) into a "coherent" version in the gate-set QC(GATES) by replacing classical input registers with quantum input registers (thereby extending the domain of the input), and commuting all of the measurement operations to the end of the circuit, as follows:

1. Decompose each measurement operation K on n qubits (other than M^{z}) into a change of basis, from the measurement basis of K to the standard basis, followed by a measurement in the standard basis:

$$\mathsf{K}_{\alpha/\delta} \quad = \quad \left[\bigotimes_{v\in\delta} \mathsf{M}^{\mathsf{z}}_{v}\right] \circ \mathsf{B}^{\mathsf{K}}_{\delta} \,, \tag{1.91}$$

   where we attribute classical bits $\mathsf{s}[v] \in \alpha$ to each $v \in \delta$, in sequence. We similarly replace controlled measurement operators with *classically controlled* changes of basis followed by measurement in the standard basis:

$$\mathsf{K}^{c}_{\alpha/\delta} \quad = \quad \left[\bigotimes_{v\in\delta} \mathsf{M}^{\mathsf{z}}_{v}\right] \circ \left(\mathsf{B}^{\mathsf{K}^1}\right)^{c}_{\delta} \circ \left(\mathsf{B}^{\mathsf{K}^0}\right)^{1-c}_{\delta} . \tag{1.92}$$

2. We substitute any classically-controlled operation Υ^c depending on a classical input bit c, including instances of such operations arising from measurements in the previous step, by a (coherently) controlled unitary $\wedge\Upsilon$ acting on the input qubit which has replaced the input bit;

3. For each qubit v and any unitary $\Upsilon \in$ GATES, we perform the following substitution repeatedly until there are no further classically controlled unitaries remaining:

$$\Upsilon^{\mathsf{s}[v]}_{\sigma}\ \mathsf{M}^{\mathsf{z}}_{v} \quad \longmapsto \quad \mathsf{M}^{\mathsf{z}}_{v}\ \wedge\Upsilon_{v,\sigma} \tag{1.93}$$

Because each classically controlled operation in a CC(GATES) circuit acts either on an input bit or a bit allocated by a measurement, the above procedure will eventually remove every classically controlled unitary (either from the original circuit or from the decomposition of classically controlled measurements), and replace it with a coherently controlled unitary, thereby producing a unitary circuit. Replacing the classical registers at the input with equivalent quantum registers also leaves the effect of the circuit as a superoperator unchanged on the original input space: we refer to this as a *coherent extension* of the original circuit. In summary:

Principle of Deferred Measurement [17, 19]. *For any gate set* GATES *giving rise to a unitary circuit model, every classically controlled circuit C in a gate set* CC(GATES) *can be transformed to a unitary circuit Q in the gate set* QC(GATES)*, where Q is a coherent extension of C, and where the difference in the complexity of Q and C is bounded above by the number of measurements in C.*

The principle of deferred measurement is what allows us in practise to consider quantum computation in terms of unitary circuits. Further, if GATES is universal for quantum computation, we may further decompose the operations of QC(GATES) into operations of GATES; the Solovay-Kitaev theorem then places an upper bound on the factor of increase of the resulting GATES circuit over the original CC(GATES) circuit. Therefore, we do not incur very large computational costs by restricting ourselves to unitary circuit models, rather than always explicitly considering classically controlled unitary circuits.

1.5 Clifford circuits and the stabilizer formalism

A final model of quantum computation which we will consider in this Chapter is one known *not* to be universal for quantum computation; and furthermore, one which is known to be efficiently simulatable on classical computers. This model is a controlled-unitary circuit model whose unitary operations are restricted to what is called the *Clifford group*, and whose preparation/measurement operations are similarly restricted. In this section, we present the results of [68] about this model of computation, as it provides an illustrative pre-amble to measurement-based computation.

1.5.1 The Pauli and Clifford groups

Definition 1-26. The *Pauli group* on n qubits is the group of operators $\mathcal{P}^{\otimes n}$, consisting of n-fold tensor products of the group $\mathcal{P} = \langle \mathbb{1}_2, X, Y, Z \rangle = \langle i\mathbb{1}_2, X, Z \rangle$ generated by the Pauli operators (see Definition 1-18). The *Clifford group* on n qubits is the normalizer of the Pauli group in $\mathsf{U}(2^n)$.

The following Lemma is proven for the set $\{H, R_{\mathsf{z}}(\pi/2), \wedge X\}$ by [69], and therefore also holds as a corollary for the sets $\{H, R_{\mathsf{z}}(\alpha), \wedge Z\}_{\alpha \in \frac{\pi}{2}\mathbb{Z}}$, and $\{J(\alpha), \wedge Z\}_{\alpha \in \frac{\pi}{2}\mathbb{Z}}$ by the remarks preceding Lemma 1-4:

Lemma 1-7. *For any $n \geqslant 1$, the Clifford group on n qubits can be generated by the following sets of unitaries together with tensor products with scalar multiples of the identity: $\{H, R_{\mathsf{z}}(\pi/2), \wedge X\}$, $\{H, R_{\mathsf{z}}(\alpha), \wedge Z\}_{\alpha \in \frac{\pi}{2}\mathbb{Z}}$, and $\{J(\alpha), \wedge Z\}_{\alpha \in \frac{\pi}{2}\mathbb{Z}}$.*

A Clifford group operation U can be characterized (up to an unimportant scalar factor) by the action induced on the Pauli group by conjugation by U. Because $UY_vU^\dagger = U(iX_vZ_v)U^\dagger = i(UX_vU^\dagger)(UZ_vU^\dagger)$ for a Y operation on any single qubit v, and $U\mathbb{1}U^\dagger = \mathbb{1}$, we may further reduce this to how U transforms tensor products of the operators X and Z on single qubits by conjugation. This then leads to a natural representation of Clifford group operations as CPTP maps, albeit acting as a transformation of the zero-trace, non-positive tensor products of X and Z operators with tensor powers of $\mathbb{1}_2$. For

instance, it is easy to verify by calculation using the CPTP maps defined by (1.70) that

$$\mathsf{H}(X) = Z\,, \qquad \mathsf{H}(Y) = -Y\,, \qquad \mathsf{H}(Z) = X\,; \tag{1.94a}$$

$$\mathsf{R}^{\pi/2}(X) = Y\,, \qquad \mathsf{R}^{\pi/2}(Y) = -X\,, \qquad \mathsf{R}^{\pi/2}(Z) = Z\,; \tag{1.94b}$$

$$\begin{aligned} \mathsf{E}\,(X \otimes \mathbb{1}_2) &= X \otimes Z\,, & \mathsf{E}\,(Y \otimes \mathbb{1}_2) &= Y \otimes Z\,, & \mathsf{E}\,(Z \otimes \mathbb{1}_2) &= Z \otimes \mathbb{1}_2\,, \\ \mathsf{E}\,(\mathbb{1}_2 \otimes X) &= Z \otimes X\,, & \mathsf{E}\,(\mathbb{1}_2 \otimes Y) &= Z \otimes Y\,, & \mathsf{E}\,(\mathbb{1}_2 \otimes Z) &= \mathbb{1}_2 \otimes Z\,. \end{aligned} \tag{1.94c}$$

(Note also that the middle column of these equations can be easily computed from the left and right columns by $Y = iXZ = -iZX$.) The way in which compositions of these CPTP maps transform $\mathcal{P}^{\otimes n}$ can then be easily computed.

The Clifford group acts *transitively* on elements of the Pauli group with eigenvalues ± 1: that is, for any two Hermitian Pauli group elements $P_1, P_2 \in \mathcal{P}^{\otimes n} \setminus \left\{\pm \mathbb{1}_2^{\otimes n}\right\}$, there is a Clifford group operation U such that $P_2 = UP_1U^\dagger$. This can easily be seen from the fact that for any Pauli operator P, we may apply $\mathsf{R}^{\pi/2}$ operations to obtain an operator P' for which at least one qubit a is acted on by X, and the others with Z; and then perform E operations on pairs of qubits (a, v) for which v is acted on by P' by a Z operation, which will yield the operator X_a acting only on a. By defining the mapping

$$\mathsf{HSWAP}_{a,b} = \mathsf{E}_{a,b}\mathsf{H}_a\mathsf{H}_b\mathsf{E}_{a,b}\mathsf{H}_a\mathsf{H}_b\mathsf{E}_{a,b}\,, \tag{1.95a}$$

which for product operators $\rho \otimes \sigma \in \mathsf{L}(\mathcal{H}_2 \otimes \mathcal{H}_2)$ performs the mapping

$$\mathsf{HSWAP}(\rho \otimes \sigma) = \mathsf{H}(\sigma) \otimes \mathsf{H}(\rho)\,, \tag{1.95b}$$

we can see that performing an $\mathsf{HSWAP}_{a,n}$ on a and the n^{th} qubit, we obtain operation Z_n acting only on the n^{th} qubit, up to a scalar factor of ± 1 (not directly attributable to an operation on any given qubit). Finally, we may transform $-Z_n$ to Z_n by conjugation by $X_n = H_nR_{\mathsf{z}}(\pi/2)_n^2H_n$; and so, any Pauli operator $P \in \mathcal{P}^{\otimes n}$ may be mapped by a Clifford group operation to $Z_n = \mathbb{1}_2^{\otimes n-1} \otimes Z$. We can do this for any Pauli operator; furthermore, by performing such a Clifford transformation in reverse, we may also obtain an arbitrary Pauli operator from a Z on the n^{th} qubit. This proves our claim of transitivity.

1.5.2 Stabilizer groups and stabilizer codes

It is easy to verify that any two Pauli operators $P \in \mathcal{P} = \langle \mathbb{1}_2, X, Y, Z\rangle$ either commute or anticommute with each other; and that therefore the same holds for two elements of $\mathcal{P}^{\otimes n}$. We may then consider subgroups of $\mathcal{P}^{\otimes n}$ generated by commuting sets of operators. Any such group may be characterized by a set of independent generators $\{S_1, \ldots, S_{n-k}\}$ for some $k \leqslant n$. As $\{X, Y, Z\}$ are all self-inverse, such a group will be of size 2^{n-k}, and in particular isomorphic to a subgroup of $\mathbb{Z}_2^n$.

We may also show that for any $0 \leqslant k \leqslant n$, the Clifford group acts transitively on abelian subgroups of the Pauli group of size 2^{n-k} whose operators all have $+1$ eigenspaces (which we call *Pauli stabilizer groups*, often simply *stabilizer groups*). We may show this by induction on $n - k$, as follows. The case of $n = k$ is trivial, as there is only one such group, $\left\{\mathbb{1}_2^{\otimes n}\right\}$. For $n > k$, let $\{S_1, \ldots, S_{n-k}\}$ be a set of generators for the

group: we may reduce the problem by considering a Clifford group operation U such that $US_{n-k}U^\dagger = Z_n$, using the transitivity of the action of the Clifford group on the non-trivial Hermitian elements of $\mathcal{P}^{\otimes n}$ Because two Pauli operators P_1 and P_2 commute if and only if $UP_1U^\dagger$ and $UP_2U^\dagger$ commute, it is easy to show that the operators $US_jU^\dagger$ must act on the n^{th} qubit with either the identity or a Z operation. For those generators S_j such that the latter holds, define $S'_j = S_jS_{n-k}$; then, $US'_jU^\dagger$ acts on the n^{th} qubit with the identity. If we define $S'_j = S_j$ for the other stabilizers, $\{S'_j\}_{j=1}^{n-k}$ generates the same abelian group, and by construction is transformed via conjugation by U to a set of operators $\{T_1 \otimes \mathbb{1}_2\,,\ \ldots\ ,\ T_{n-k-1} \otimes \mathbb{1}_2\,,\ Z_n\}$. The first $n-k-1$ of these generate an abelian subgroup on the first $n-1$ qubits, which by induction we may transform by a Clifford group operation to the group generated by $\{Z_{k+1}, \ldots, Z_{n-1}\}$. Again, this transformation may be reversed to yield an arbitrary stabilizer code of size 2^{n-k}; then any such abelian group of size 2^{n-k} to be mapped to any other.

A set of Pauli operators which commutes, by that very fact, shares a common set of eigenvectors. The group $\langle Z_{k+1}, Z_{k+2}, \ldots, Z_n\rangle$ in particular admits a 2^k-dimensional subspace of ${\mathcal{H}_2}^{\otimes n}$ which are $+1$-eigenvectors of the entire group (specifically, the set of states of the form $|\psi\rangle \otimes |0\rangle^{\otimes n-k}$ for arbitrary $|\psi\rangle \in {\mathcal{H}_2}^{\otimes k}$); by transitivity of the Clifford group on stabilizer groups of similar size, every stabilizer group of size 2^{n-k} which does not include $-\mathbb{1}_2^{\otimes n}$ has a 2^k-dimensional space of joint $+1$-eigenvectors. We say that a vector $|\psi\rangle$ is *stabilized* by an operator M if $|\psi\rangle = M\,|\psi\rangle$, *i.e.* if it is a $+1$-eigenvector of M, which motivates the following terminology:

Definition 1-27. A *stabilizer code* $\mathcal{E}$ on n qubits is the space of joint $+1$-eigenvectors of a stabilizer group on n qubits. In particular, a *stabilizer state* on n qubits is a state $\rho = |\psi\rangle\langle\psi|$, where $|\psi\rangle$ is a joint $+1$-eigenvector of a commuting group of 2^n Pauli operators on n qubits. The *stabilizer group of the code* is the group for which the code is the joint $+1$-eigenspace.

Example 4. Let $S \in \{X, Y, Z\}$, and let $|{+}\mathsf{s}\rangle$ represent the corresponding state vector in $\{|{+}\mathsf{x}\rangle, |{+}\mathsf{y}\rangle, |{+}\mathsf{z}\rangle\}$. It is easy to verify that $|{+}\mathsf{s}\rangle$ is stabilized by S; then, for any sequence of non-trivial Pauli operators $\left(S_j\right)_{j=1}^n$, where each operator S_j also acts on qubit j and stabilizes the state vector $|{+}\mathsf{s}_j\rangle$, the product state

$$|\psi\rangle \;=\; \bigotimes_{j=1}^{n} |{+}\mathsf{s}_j\rangle \tag{1.96}$$

is stabilized by the abelian group $\langle S_1,\ \ldots,\ S_n\rangle$. As a special case, the state $|0\rangle^{\otimes n} = |0\rangle \otimes \cdots \otimes |0\rangle$ is stabilized by the group $\langle Z_1, \ldots, Z_n\rangle$.

1.5.3 The stabilizer formalism

As stabilizer codes can be characterized by a short list of generators for its stabilizer group, the state space of a system which is initially prepared in a pure state in some stabilizer code and then transformed by Clifford group operations can be efficiently computed. We may extend this result further: consider the circuits which can be composed of the unitary

CPTP maps H, $\mathsf{R}^{\pi/2}$, and E; the state preparation maps

$$\mathsf{N}^{\mathsf{x}}_v(\rho) \;=\; \rho \otimes |{+}\mathsf{x}\rangle\langle{+}\mathsf{x}|_v \;, \tag{1.97a}$$

$$\mathsf{N}^{\mathsf{y}}_v(\rho) \;=\; \rho \otimes |{+}\mathsf{y}\rangle\langle{+}\mathsf{y}|_v \;, \tag{1.97b}$$

$$\mathsf{N}^{\mathsf{z}}_v(\rho) \;=\; \rho \otimes |{+}\mathsf{z}\rangle\langle{+}\mathsf{z}|_v \;; \tag{1.97c}$$

the von Neumann measurement operations

$$\mathsf{P}^{\mathsf{x}}_{v:r}(\rho) \;=\; |{+}\mathsf{x}\rangle\langle{+}\mathsf{x}|_v\,\rho\,|{+}\mathsf{x}\rangle\langle{+}\mathsf{x}|_v \otimes |0\rangle\langle 0|_r \;+\; |{-}\mathsf{x}\rangle\langle{-}\mathsf{x}|_v\,\rho\,|{-}\mathsf{x}\rangle\langle{-}\mathsf{x}|_v \otimes |1\rangle\langle 1|_r \;, \tag{1.98a}$$

$$\mathsf{P}^{\mathsf{y}}_{v:r}(\rho) \;=\; |{+}\mathsf{y}\rangle\langle{+}\mathsf{y}|_v\,\rho\,|{+}\mathsf{y}\rangle\langle{+}\mathsf{y}|_v \otimes |0\rangle\langle 0|_r \;+\; |{-}\mathsf{y}\rangle\langle{-}\mathsf{y}|_v\,\rho\,|{-}\mathsf{y}\rangle\langle{-}\mathsf{y}|_v \otimes |1\rangle\langle 1|_r \;, \tag{1.98b}$$

$$\mathsf{P}^{\mathsf{z}}_{v:r}(\rho) \;=\; |{+}\mathsf{z}\rangle\langle{+}\mathsf{z}|_v\,\rho\,|{+}\mathsf{z}\rangle\langle{+}\mathsf{z}|_v \otimes |0\rangle\langle 0|_r \;+\; |{-}\mathsf{z}\rangle\langle{-}\mathsf{z}|_v\,\rho\,|{-}\mathsf{z}\rangle\langle{-}\mathsf{z}|_v \otimes |1\rangle\langle 1|_r \;; \tag{1.98c}$$

and the corresponding destructive measurement operations,

$$\mathsf{M}^{\mathsf{x}}_v(\rho) \;=\; \langle{+}\mathsf{x}|_v\,\rho\,|{+}\mathsf{x}\rangle_v \otimes |0\rangle\langle 0|_{\mathsf{s}[v]} \;+\; \langle{-}\mathsf{x}|_v\,\rho\,|{-}\mathsf{x}\rangle_v \otimes |1\rangle\langle 1|_{\mathsf{s}[v]} \;, \tag{1.99a}$$

$$\mathsf{M}^{\mathsf{y}}_v(\rho) \;=\; \langle{+}\mathsf{y}|_v\,\rho\,|{+}\mathsf{y}\rangle_v \otimes |0\rangle\langle 0|_{\mathsf{s}[v]} \;+\; \langle{-}\mathsf{y}|_v\,\rho\,|{-}\mathsf{y}\rangle_v \otimes |1\rangle\langle 1|_{\mathsf{s}[v]} \;, \tag{1.99b}$$

$$\mathsf{M}^{\mathsf{z}}_v(\rho) \;=\; \langle{+}\mathsf{z}|_v\,\rho\,|{+}\mathsf{z}\rangle_v \otimes |0\rangle\langle 0|_{\mathsf{s}[v]} \;+\; \langle{-}\mathsf{z}|_v\,\rho\,|{-}\mathsf{z}\rangle_v \otimes |1\rangle\langle 1|_{\mathsf{s}[v]} \;. \tag{1.99c}$$

Definition 1-28. For an infinite set V of qubits and B of bits, the *Clifford circuit model* is the circuit model generated by $\textsc{Clifford} = \textsc{cc}(\mathcal{G})$,[16] where

$$\mathcal{G} \;=\; \left\{ \mathsf{H}_v \,,\; \mathsf{R}^{\pi/2}_v \,,\; \mathsf{E}_{u,v} \,,\; \mathsf{N}^{\sigma}_v \,,\; \mathsf{P}^{\sigma}_{v:r} \,,\; \mathsf{M}^{\sigma}_v \;\middle|\; \sigma \in \{\mathsf{x},\mathsf{y},\mathsf{z}\} \,,\; r \in B, \text{ and } v, w \in V \right\} . \tag{1.100}$$

Note that circuits in the $\textsc{Phases}(\frac{\pi}{2}\mathbb{Z})$ model which do not use trace-out operations are also circuits in the Clifford model.

Assuming an arbitrary input state, the state-space of the output to a Clifford circuit conditioned on particular values of each bit in the circuit (and in particular, conditioned on particulars outcomes of each von Neumann measurement performed) may be efficiently computed by transforming generators of the stabilizer group of the system. The techniques for doing this, described below, were presented by [68] and are referred to collectively as *the stabilizer formalism.*

The state preparation operators N^{σ}_v simply adjoin another qubit v on which the existing generators act with the identity, and also add another generator S_v stabilizing the state $|{+}\sigma\rangle_v$. As noted before, the effect of the unitary operators H, $\mathsf{R}^{\pi/2}$, and E may be efficiently computed on each generator. This leaves the von Neumann measurements $\mathsf{P}^{\sigma}_{v:r}$ and destructive measurement M^{σ}_v, which we describe following the treatment of [68]. Consider a Pauli operator P_v on a single qubit v, for $P \in \{X, Y, Z\}$. For the corresponding label $p \in \{\mathsf{x},\mathsf{y},\mathsf{z}\}$, a $\mathsf{P}^{p}_{v:r}$ measurement is equivalent to a measurement of the observable P_v (recall Definitions 1-6 and 1-8), and relabelling the measurement results $(-1)^r \mapsto r$ to obtain

[16] To be precise, we are using here the slightly extended sense of the function $\textsc{Gates} \longmapsto \textsc{cc}(\textsc{Gates})$, described in the footnote (15) on page 40, for when Gates does not define a unitary circuit model.

measurement results $r \in \{0,1\}$. Abusing our earlier terminology, we will momentarily speak of the measurements in the Clifford circuit model as being measurements of the corresponding Pauli operator as an observable in this fashion.

Consider a state $\rho = |\psi\rangle\langle\psi|$ for $|\psi\rangle \in \mathcal{E}$, where we let $\mathcal{E}$ be the space stabilized by a stabilizer group $\mathcal{S} = \langle S_1, \ldots, S_{n-k}\rangle$; and define the projectors

$$\Pi_0 \;=\; \tfrac{1}{2}\left(\mathbb{1}_2^{\otimes n} + P_v\right) \qquad \text{and} \qquad \Pi_1 \;=\; \tfrac{1}{2}\left(\mathbb{1}_2^{\otimes n} - P_v\right) . \tag{1.101}$$

- If $P_v \in \mathcal{S}$, then $|\psi\rangle$ is a $+1$-eigenvector of P_v, in which case a von Neumann P_v measurement will yield the result $\mathsf{s}[v] = 0$ and leave the state of the quantum part of the system unchanged.

- If $P_v \notin \mathcal{S}$ but P_v commutes with all of the generators S_j, then we may decompose the code $\mathcal{E}$ stabilized by $\mathcal{S}$ into two orthogonal subspaces $\mathcal{E}_0$ and $\mathcal{E}_1$, stabilized by $\mathcal{S}_0 = \mathcal{S} \oplus \langle P_v\rangle$ and $\mathcal{S}_1 = \mathcal{S} \oplus \langle -P_v\rangle$ respectively. (That these subspaces are orthogonal follows from the orthogonality of the ± 1-eigenspaces of P_v.) Let $|\psi_0\rangle = \Pi_0 |\psi\rangle$ and $|\psi_1\rangle = \Pi_1 |\psi\rangle$: then a von Neumann P_v measurement yields the state $|\psi_0\rangle\langle\psi_0| \otimes |0\rangle\langle 0|_r + |\psi_1\rangle\langle\psi_1| \otimes |1\rangle\langle 1|_r$, where r is the bit containing the measurement result. For each $r \in \{0,1\}$, we have

  $$S_j |\psi_r\rangle \;=\; S_j \Pi_r |\psi\rangle \;=\; \Pi_r S_j |\psi\rangle = |\psi_r\rangle \tag{1.102}$$

  for every $1 \leqslant j \leqslant n-k$; then, the state after measurement conditioned on the result bit having the value r is is in the space $\mathcal{E}_r$ stabilized by $\mathcal{S}_r = \mathcal{S} \oplus \left\langle (-1)^r P_v \right\rangle$. Because the initial state $|\psi\rangle$ may be an *arbitrary* state in $\mathcal{E}_0$ or $\mathcal{E}_1$ (which would then be unaffected by measurement), this fully characterizes the post-measurement state space. Because the number of generators of the stabilizer group increases in this case, $\mathcal{E}_0$ and $\mathcal{E}_1$ each have half the dimension of $\mathcal{E}$, and so this operation is non-unitary.

- If P_v anticommutes with a generator S_j, without loss of generality we may suppose that P_v anticommutes with S_1 in particular. Then, define a new set of generators $\{S'_j\}_{j=1}^{n-k}$ for $\mathcal{S}$, such that

  $$S'_j \;=\; \left\{ \begin{array}{ll} S_j\,, & \text{if } j = 1 \text{ or } S_j \text{ commutes with } P_v \\ S_1 S_j\,, & \text{otherwise} \end{array} \right\} ; \tag{1.103}$$

  then S'_1 anticommutes with P_v, and S'_j commutes with P_v for $j \geqslant 2$. Let $\mathcal{S}_0 = \left\{P_v, S'_2, \ldots, S'_{n-k}\right\}$, and $\mathcal{S}_1 = \left\{-P_v, S'_2, \ldots, S'_{n-k}\right\}$, and let $\mathcal{E}_0$ and $\mathcal{E}_1$ be the codes stabilized by these groups. Once more, let $|\psi_0\rangle = \Pi_0 |\psi\rangle$ and $|\psi_1\rangle = \Pi_1 |\psi\rangle$: then a von Neumann P_v measurement yields the state $|\psi_0\rangle\langle\psi_0| \otimes |0\rangle\langle 0|_r + |\psi_1\rangle\langle\psi_1| \otimes |1\rangle\langle 1|_r$, where r is the bit containing the measurement result. For each $r \in \{0,1\}$, we again have

  $$S'_j |\psi_r\rangle \;=\; S'_j \Pi_r |\psi\rangle \;=\; \Pi_r S'_j |\psi\rangle = |\psi_r\rangle\,; \tag{1.104}$$

  for every $2 \leqslant j \leqslant n-k$; however, we have

  $$S'_1 |\psi_0\rangle \;=\; \tfrac{1}{2} S'_1 \left(\mathbb{1}_2^{\otimes n} + P_v\right) |\psi\rangle \;=\; \tfrac{1}{2}\left(\mathbb{1}_2^{\otimes n} - P_v\right) S'_1 |\psi\rangle \;=\; |\psi_1\rangle\,, \tag{1.105}$$

and similarly $S'_1 |\psi_1\rangle = |\psi_0\rangle$. Then, the state after measurement conditioned on the result bit having the value r is in the space $\mathcal{E}_r$ stabilized by the group $\mathcal{S}_r$, and the two possible post-measurement results may be mapped to one another by the operation S'_1. (In particular, if we wish to select for the post-measurement state that would arise for the 0 state, it suffices to apply the operation S_1 if instead the measurement result 1 arises.)

Note that the probabilities of the two outcomes in this case are both $\frac{1}{2}$: we have

$$\begin{aligned} \langle\psi_0|\psi_0\rangle &= \tfrac{1}{2}\langle\psi|\,(\mathbb{1}_2^{\otimes n} + P_v)\,|\psi\rangle &&= \tfrac{1}{2}\langle\psi|\,S'_1(\mathbb{1}_2^{\otimes n} - P_v)S'_1\,|\psi\rangle \\ &&&= \langle\psi_1|\psi_1\rangle\ , \end{aligned} \tag{1.106}$$

which implies that applying a P_v measurement on $|\psi\rangle$ yields $+1$-eigenstates and -1-eigenstates of P_v each with equal probability. Because S'_1 maps the spaces $\mathcal{E}_0$ and $\mathcal{E}_1$ into one another, we may consider arbitrary vectors $|\psi_0\rangle \in \mathcal{E}_0$ and $|\psi_1\rangle \in \mathcal{E}_1$ such that $|\psi_0\rangle = S'_1\,|\psi_1\rangle$. If $\langle\psi_0|\psi_0\rangle = \langle\psi_1|\psi_1\rangle = \frac{1}{2}$, then $|\psi\rangle = |\psi_0\rangle + |\psi_1\rangle$ is a state vector stabilized by S'_1, and also by S'_j for $j \geqslant 2$. Then, for any state vector $|\psi'\rangle$ in either $\mathcal{E}_0$ or $\mathcal{E}_1$, we may construct a state $|\psi\rangle$ stabilized by $\mathcal{S}$ for which $|\psi'\rangle$ is a possible post-measurement state. This implies that the codes $\mathcal{E}_0$ and $\mathcal{E}_1$ characterize the possible post-measurement state-spaces.

Finally, for any two states $|\psi\rangle, |\phi\rangle$ stabilized by $\mathcal{S}$, we have

$$\langle\phi|\,P_v\,|\psi\rangle \;=\; \langle\phi|\,S'_1 P_v\,|\psi\rangle \;=\; -\langle\phi|\,P_v S'_1\,|\psi\rangle \;=\; -\langle\phi|\,P_v\,|\psi\rangle\ , \tag{1.107}$$

so that $\langle\phi|\,P_v\,|\psi\rangle = 0$. Define the state-vectors

$$\left|\bar{\psi}_r\right\rangle \;=\; \frac{\Pi_r\,|\psi\rangle}{\sqrt{\langle\psi|\,\Pi_r|\psi\rangle}} \;=\; \sqrt{2}\,\Pi_r\,|\psi\rangle\ , \tag{1.108}$$

$$\left|\bar{\phi}_r\right\rangle \;=\; \frac{\Pi_r\,|\phi\rangle}{\sqrt{\langle\phi|\,\Pi_r|\phi\rangle}} \;=\; \sqrt{2}\,\Pi_r\,|\phi\rangle \tag{1.109}$$

for the states arising after a P_v measurement on $|\psi\rangle$ and $|\phi\rangle$ respectively, conditioned on obtaining the result $r \in \{0,1\}$. Then, the inner product of $\left|\bar{\phi}_r\right\rangle$ with $\left|\bar{\psi}_r\right\rangle$ can be given by

$$\begin{aligned} \left\langle\bar{\phi}_r\middle|\bar{\psi}_r\right\rangle &= 2\langle\phi|\,\Pi_r\,|\psi\rangle &&= \langle\phi|\,(\mathbb{1}_2^{\otimes n} + (-1)^r P_v)\,|\psi\rangle \\ &&&= \langle\phi|\psi\rangle\ ; \end{aligned} \tag{1.110}$$

thus the transformation performed by the P_v measurement (conditioned on either result) preserves inner products, and is therefore a unitary transformation.

- Finally, for any destructive P_v measurement, note that the state of v after a similar von Neumann measurement is characterized by being a ± 1-eigenvector of P_v determined by the measurement result; and in particular is independently distributed from the other qubits. Then no information about the rest of the qubits is lost by discarding the qubit. If the measurement result is $r \in \{0,1\}$, let $R_v = (-1)^r P_v$ be the operator stabilizing the state of v in the post-measurement stabilizer group $\mathcal{S}_r = \{S'_j\}_{j=1}^{n-k'}$ — where we may easily verify that $k' = k-1$ in the case where P_v commuted with (but was not generated by) the pre-measurement stabilizer group,

and $k' = k$ otherwise. Because all the generators S'_j commute, any other operator $S'_j \neq R_v$ must acts on v with either the identity or a P_v operation. We may then form a new set of generators

$$S''_j \quad = \quad \left\{ \begin{array}{ll} S'_j \, , & \text{if } S'_j = R_v \text{ or if } S'_j \text{ acts on } v \text{ with } \mathbb{1}_2 \\ S'_j R_v \, , & \text{otherwise} \end{array} \right\} \tag{1.111}$$

for $\mathcal{S}_r$, where $R_v \in \{S''_j\}_{j=1}^{n-k'}$ is the only operator which acts non-trivially on v. We may then trace v out by removing the generator R_v, and tracing out v from the domain of the other generators S''_j .

Thus, the state-space of a system produced by a Clifford circuit, assuming a arbitrary pure input state, can be efficiently computed for any possible value of the measurement results, and even whether the evolution will be unconditionally unitary.

Applied to the special case when there are *no* quantum inputs (*i.e.* where all qubits are allocated by the circuit itself, and prepared in one of the states $|{+}\mathsf{x}\rangle$, $|{+}\mathsf{y}\rangle$, or $|{+}\mathsf{z}\rangle$), this yields the following result, reported in [69] and attributed to Emmanuel Knill:[17]

Gottesman-Knill Theorem. *A circuit in the Clifford circuit model which has no quantum inputs can be simulated efficiently on a classical computer.*

Note that the CLIFFORD model is not even approximately universal for quantum computation: because there are only finitely many elements of $\mathcal{P}^{\otimes n}$, there are also only finitely many stabilizer groups of size 2^n on n qubits, and thus finitely many stabilizer states. Thus, there exist CPTP maps $\Phi : \{1\} \longrightarrow \mathsf{D}(2^n)$ which cannot be approximated to arbitrary precision with a Clifford circuit.

As we remarked above, circuits in the PHASES($\frac{\pi}{2}\mathbb{Z}$) model without trace-out operations are also Clifford circuits; and therefore circuits in the JACZ($\frac{\pi}{2}\mathbb{Z}$) models without trace-out operations can be easily simulated by Clifford circuits; then these models are also not approximately universal for quantum computation.[18]

1.6 Conclusion

In this Chapter, we have described quantum states as being represented by density operators, and transformations of them as being by CPTP maps, which may be performed or approximated by composing discrete operations. We have also noted the important role that unitary evolution has in quantum mechanics, and described the role of unitary transformations in defining the most commonly used models of quantum computation.

[17] The result stated in [69] is actually an equivalent result, in which the only preparation and measurement operations are N^{z} and $\mathsf{P}^{\mathsf{z}}_{v:r}$ (and the unitary gate $\wedge X$ is used in place of $\wedge Z$).

[18] A further result of Aaronson and Gottesman [2] show that simulating Clifford circuits is a complete problem for $\oplus\mathsf{L}$ [28], the class of problems which can be solved by a nondeterministic Turing machine (NTM) which uses only logarithmic workspace, and where in particular the "*yes*" instances of the problem have an odd number of accepting paths for that NTM. Solving linear equations over $\mathbb{Z}_2$ is another problem which is complete for $\oplus\mathsf{L}$ [35], and which is conjectured to be insufficient even to perform universal polynomial-time *classical* computation.

Other significantly different models of quantum computation exist which are not obviously special cases of either unitary circuit or classically-controlled unitary circuit models, but which can be shown to be polynomial-time equivalent:

Adiabatic quantum computation [59] rests on the adiabatic theorem first described by Born and Fock [22] (see [6] for a modern treatment), and which is a model of computation relying on the evolution of states under a slowly and continuously varying Hamiltonian. This model lends itself naturally to optimization problems: an analysis of the strengths and limitations of this model, and its relation to unitary circuits, can be found in [34].

Topological quantum computation [82] is a model which exploits properties of particle statistics to approximate unitaries to arbitrary precision by braiding the trajectories of "virtual particles" which can be simulated via the quantum Hall effect [7]. This is a natural model for describing algorithms for such problems as approximating knot invariants [5, 77]; and by exploiting the topological properties of the particle trajectories, it is possible to naturally achieve fault-tolerant computation. A review of toplogical quantum computation can be found in [97].

Continuous-time quantum walks [60] represent a generalization of diffusion processes in graphs by random walks by replacing the diffusion equation with Schrödinger evolution. (Discrete-time quantum walks can also be formulated naturally by generalizing random walks [134], but these may be easily subsumed by classically controlled unitary circuits.) This model led to the discovery of a quantum speed-up for evaluating NAND-trees [58]; the universality of this model is shown in [29].

In this thesis, we are primarily interested in one further model of quantum computation, which may be subsumed into classically controlled unitary circuit models, but in which measurements depending on many classical controls play a significant role, resulting in a model with a significantly different character than unitary circuit models. This model is the *one-way measurement-based model*, which we introduce in the next chapter.

Chapter 2

One-way measurement-based computation

In the absence of efficient algorithms for simulating quantum mechanics or for solving difficult number-theoretic problems such as factoring or discrete logarithms, quantum computers represent a potential breakthrough in scientific computing. However, the technical challenges involved in controlling the evolution of a quantum mechanical system for the purposes of computation have prompted the exploration of different "computational primitives" for quantum computation — meaning essentially different sets of operations which (a) form a model which is universal for quantum computation, and (b) may be easier to implement physically.

In order to achieve universal quantum computation via unitary circuits, multi-qubit entangling gates are required: otherwise, only states which are tensor-products of single-qubit states can be produced. As scalable quantum control of multi-qubit systems is an open research problem, reducing the number of rounds involving multi-qubit interactions is one plausible route to achieving practical scalable quantum computation. However, to perform or approximate any unitary evolution which cannot be efficiently simulated on a classical computer, many rounds of multi-qubit operations such as $\wedge X$ or $\wedge Z$ (as defined in (1.62)) are required in a unitary circuit model.[1]

If we are interested in describing unitary transformations of states (or *approximately* unitary transformations, in the sense of approximating a unitary CPTP map with precision ε), while allowing classically controlled measurements to play a significant role in the computation, a single round of multi-qubit operations — or alternatively, access to appropriate multi-qubit resource states — will suffice to achieve universal quantum computation. The models which make this possible are the *measurement based models* of quantum computation.

In [70], Gottesman and Chuang showed that the preparation of a special set of multi-qubit states, together with teleportation [16], was universal for quantum computation.

[1] To be precise, although it is widely believed that the measurement results arising out of quantum mechanics (and arbitrary unitary circuits in particular) cannot be efficiently approximated by a classical model of computation such as Turing machines, any unitary circuit on n qubits in which each qubit is operated on fewer than $O(\log(N))$ times by a two-qubit gate can be simulated in $n \cdot \text{poly}(N)$-time by a classical computer [76].

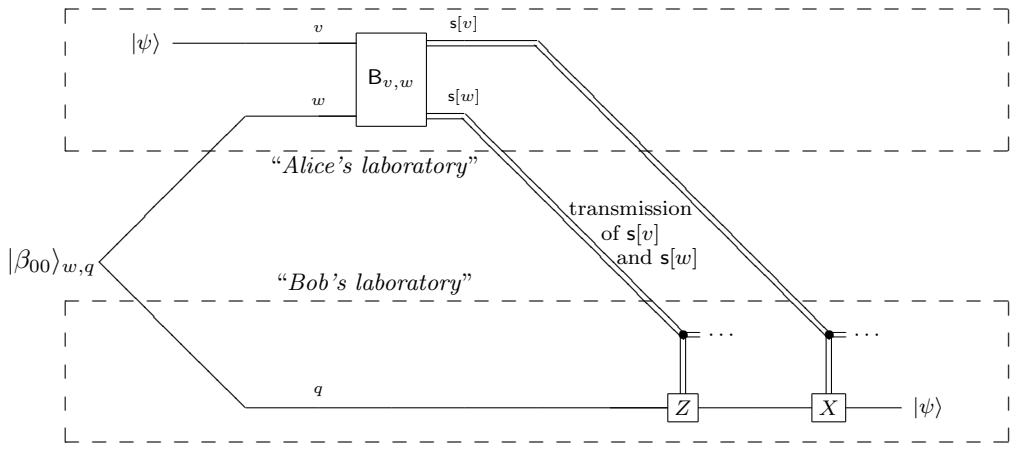


FIGURE 2-1. An illustration of the "teleportation" protocol defined in (2.3). We use similar conventions as in Figures 1-2 and 1-3 to illustrate circuits with measurements and classical control; double-lines represent classical bits produced by measurement. The system is divided into two separate "laboratories" controlled by separate parties ("Alice" and "Bob"), between which we suppose that coherently controlled operations cannot occur. Diagonal lines represent the sharing of information resources: the preparation of the joint initial state $|\beta_{00}\rangle$ of w and q, and the transmission of the bits $\mathsf{s}[v]$ and $\mathsf{s}[w]$.

Teleportation is essentially a means of implementing the identity operation by means of measurement and classical control: if we define the *Bell basis* by the states

$$\begin{aligned} |\beta_{00}\rangle &= \tfrac{1}{\sqrt{2}}|0\rangle\otimes|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle\otimes|1\rangle\,, & |\beta_{10}\rangle &= \tfrac{1}{\sqrt{2}}|0\rangle\otimes|1\rangle + \tfrac{1}{\sqrt{2}}|1\rangle\otimes|0\rangle\,, \\ |\beta_{01}\rangle &= \tfrac{1}{\sqrt{2}}|0\rangle\otimes|0\rangle - \tfrac{1}{\sqrt{2}}|1\rangle\otimes|1\rangle\,, & |\beta_{11}\rangle &= \tfrac{1}{\sqrt{2}}|0\rangle\otimes|1\rangle - \tfrac{1}{\sqrt{2}}|1\rangle\otimes|0\rangle\,, \end{aligned} \tag{2.1}$$

and a *Bell basis measurement* operation by the map

$$\mathsf{B}_{v,w}(\rho_{v,w}) = \sum_{x,z\in\{0,1\}} \langle\beta_{xz}|\,\rho\,|\beta_{xz}\rangle_{v,w} \left[\,|x\rangle\langle x|_{\mathsf{s}[v]}\otimes|z\rangle\langle z|_{\mathsf{s}[w]}\right], \tag{2.2}$$

then teleportation (as described in [70]) is the map $\mathsf{T}_{q/v}:\mathsf{D}(v)\longrightarrow\mathsf{D}(q)$ given by

$$\mathsf{T}_{q/v}(\rho_v) = \left(\mathsf{X}_q^{\mathsf{s}[v]}\circ\mathsf{Z}_q^{\mathsf{s}[w]}\circ\mathsf{B}_{v,w}\circ\left[\mathbb{1}_v\otimes|\beta_{00}\rangle\langle\beta_{00}|_{w,q}\right]\right)(\rho_v), \tag{2.3}$$

where Z_q^s and X_q^s are classically-controlled Z and X operations depending on the bit s. The result of [16] is that $\mathsf{T}_{q/v}$ as defined in (2.3) performs the map $\mathbb{1}_{q/v}$: that is, the state of v is "teleported" unchanged into q.[2] This remains true even though the qubit q may never have interacted with the qubit v, and may be separated from v by a large distance: if the qubits w and q are controlled by different parties (conventionally named "Alice" and "Bob") after their joint preparation in the state $|\beta_{00}\rangle$, it suffices for Alice to communicate the results $\mathsf{s}[v]$ and $\mathsf{s}[w]$ to Bob, and for Bob to perform the appropriate operations depending on those bits. This is illustrated in Figure 2-1.

[2]The description of the output neglects the classical measurement results $\mathsf{s}[v]$ and $\mathsf{s}[w]$, which are also produced as output and which will be uncorrelated and uniformly randomized bits; it is conventional to implicitly discard classical measurement results when they are no longer required.

Teleportation is thus an important primitive for transmitting quantum information, using the correlations of the Bell state $|\beta_{00}\rangle\langle\beta_{00}|$ and classical communication; [70] showed that by generalizing the initial states used as a resource, teleportation (and in particular Bell measurements) could also be used as a *computational* primitive. In the three years following Gottesman and Chuang's result, three different proposals for quantum computation based on measurement operations arose. Nielsen [100] described a model of quantum computation based on four-qubit measurement operations; this together with the result of [70] which was subsequently refined by Leung [90] into what is called the *teleportation-based model of quantum computation* in [30]. The third proposal, the *one-way measurement-based model* presented by Robert Raussendorf and Hans Briegel in [112], forms the starting point for the main topic of this thesis.

In this chapter, we will describe different variants of the one-way measurement model in terms of the *stabilizer formalism* presented in Section 1.5.3, and provide an overview of applications and potential schemes for implementing one-way measurement based quantum computing.

New results. Strictly speaking, none of the results of this Chapter are new. However, we present explicit definitions for three distinct (but closely related) constructions of measurement-based computation, to which we will regularly refer in later chapters. These constructions are the *simplified* and *complete DKP constructions* (after Danos, Kashefi, and Panangaden [40]), defined in Section 2.2.5, and the *simplified RBB construction* (after Raussendorf, Browne, and Briegel [113]), defined in Section 2.2.6.

2.1 Overview and proposals for implementation

As the one-way measurement model was originally proposed as a scheme for implementations of quantum computation, we will provide a quick overview of the model in order to hilight the aspects which make it seem reasonable as a candidate for scalable quantum computation. Before describing the model in full detail, we will give an overview of different proposals for fulfilling the requirements of the one-way measurement model.

2.1.1 A brief description of the one-way model

The one-way measurement model of quantum computing proposed in [112] is a model of computation which involves no explicit multi-qubit operations: instead, it allows the preparation of arbitrary states from a uniform family of states,[3] called *cluster states*.

Definition 2-1. The $n \times d$ *grid* is a graph $G_{n,d}$ whose vertex set consists of *grid sites* $v \in \{1,\ldots,n\} \times \{1,\ldots,d\}$, and whose edges are given by adjacent grid sites, *i.e.* pairs vw of vertices $v, w \in V(G)$ for which $\|v - w\|_1 = 1$, where $\|x\|_1$ is the ℓ_1-norm. The $n \times d$ *cluster state* is then a stabilizer state whose stabilizer group

[3] A uniform family of states is one which can be efficiently specified by a classical, deterministic Turing machine in time polynomial in the size of the desired state.

is given by $\mathcal{C}_{n,d} = \langle K_v \rangle_{v \in V(G)}$, where

$$K_v \;=\; X_v \prod_{w \in N(v)} Z_w \;, \tag{2.4}$$

and where $N(v)$ denotes the vertices adjacent to v in the graph $G_{n,d}$.

The $n \times d$ cluster state can be produced by evolution by an Ising interaction operating for an appropriate length of time, where each qubit in the grid is initially prepared in the $|+\rangle$ state [112]: several schemes for how this may be done are described in the original article.

Having a cluster state, computation is then performed using single qubit measurements. Qubits are destructively measured either using Z observables, or observables of the form $M(\theta) = \cos(\theta)X + \sin(\theta)Y$ which anticommute with Z; the qubits measured with a Z observable are effectively removed from the computation, while those measured with an observable $M(\theta)$ are used to control what computation is performed. It is possible to show that each qubit in the cluster is maximally entangled with its' nearest neighbors; then, each measurement result is maximally random. The different measurement outcomes corresponds to different transformations of the data in the computation: in order to simulate *e.g.* a unitary circuit, it becomes necessary to retain the results of the measurements long enough to counteract the randomness, *e.g.* by changing the measurement observables of later measurements. Thus, the one-way measurement model is a *classically controlled* model of computation. Furthermore, the number of measurement results that an operation may depend upon may be arbitrarily large, in general scaling with the length of the computation.

We will describe the one-way measurement model more fully in Section 2.2: for the time being, we will remark that the necessary elements of the one-way model as a scheme for for implementations of quantum computation are the possibility of creating a family of large entangled states; the ability to perform classically controlled measurements; and the ability to compute and transmit measurement results quickly enough to act as the classical control of further measurements. We can broaden the criteria for implementation somewhat by observing the following:

- The large entangled states required do not necessarily have to be cluster states. We will show how the model of [112] may be generalized to allow quantum computation with general *graph states*, which we may define in analogy to cluster states by replacing the $n \times d$ grid with an arbitrary graph.
- Rather than adapting the measurements, we may allow classically controlled unitary operations just prior to each measurement, which performs a change of reference frame which maps some "logical" measurement observable to a particular "default" measurement observable. Generalizing still further, we may provide for the preparation of arbitrary density operators if we leave some qubits unmeasured to support a final state; in this case, classically controlled unitaries are necessary to enable the preparation of pure states.

These are the primary features which make the one-way measurement model an attractive prospect for implementation: it provides a strict divide between a stage involving

the creation of entanglement, and a stage involving only single-qubit operations. The requirement that these operations depend on the results of previous measurements may present a challenge, but this challenge is of a significantly different character than the control of a many-body Hamiltonian for the purpose of performing multi-qubit transformations. As well, once a cluster state has been prepared, it can in principle be safely stored until ready for use, provided a sufficiently reliable quantum memory.

2.1.2 Proposals for implementation

One of the promises of the one-way measurement model is that the state involved is quite simple (in particular, it is a stabilizer state) and can in principle be easily prepared. The original proposal [112] itself suggested that the state may be possible to produce by controlled Ising interactions on qubits prepared in the $|+\rangle$ state. We now consider the proposals for schemes for producing cluster states and graph states, and for performing one-way measurement-based computation generally.

Optical proposals

The first proposal for efficient quantum computing by linear optics (albeit not implementing the one-way measurement model) was presented in [85], which introduced an entangling gate using number counting photo-detectors (thus introducing an effective non-linearity) which failed with finite probability using techniques similar to [70]. However, in order to perform scalable quantum computation, a significant amount of overhead in error correction is required even in the case of ideal apparatus as there is a significant probability of photon loss due to destructive measurement.

The approach of [85] was improved upon in [104], which creates a cluster state as an entangled resource by successive "fusion" operators, used to append qubits onto an existing graph state to make a larger graph state. The overhead due to the possibility of failure of the fusion operators is then realized as repeated attempts to create a cluster state: in the case of ideal apparatus, no error correction is required, resulting in a reduction of overhead. A further improvement is made by [27], who employ fusion operators realized via a "simplified" application of the Hong-Ou-Mandel effect [75][4] to reduce the sensitivity of the interferometer on path perturbations.

Another approach [93] known as "repeat until success" uses a simple error correcting code in conjunction with linear optics and measurement, to implement entangling gates in such a way that the quantum information of the system is not destroyed when the logical gate fails: this allows cluster states to be built with a much better expected time, as in the ideal case no losses arise from a failed operation.

In all of the above models, as with all optical set-ups, the primary challenges to be overcome are inefficiencies associated with photon creation and detection: see *e.g.* [92] for a discussion of these issues; more information on linear optical proposals can be found in [86].

[4] The simplification entailed in the results of [27] is that the Hong-Ou-Mandel effect does not occur as a part of a Mach-Zehnder interferometer, which is sensitive to perturbations in path length on the order of one wavelength.

Matter qubits and "hybrid" matter-optics approaches

In [10], a scheme for constructing cluster states by interacting "stationary" qubits (specifically, defects in diamonds) is proposed which probabilistically entangles pairs of excited qubits by performing interferometry on the photons emitted in decay processes. Combining this with the "repeat until success" techniques from linear optics led to the proposal of [92], which promises to provide the best features of both proposals with regards to resilience against noise: *i.e.* stable quantum memory provided by the spatially separated "stationary" qubits, whose interactions are mediated by deterministic photon gates provided by the optical elements.

An alternative approach to the previous, explicitly hybrid model between matter and optics is to employ a *broker/client model* as described in [15], in which fragments of graph states are generated using fast-reacting but unstable subsystems (such as electron spin), and then transferred to more stable sub-systems (such as nuclear spins) to reduce the probability of loss..

Optical lattices form another setting for proposals of one-way measurement-based computation, wherein cold atoms are localized in a periodic electromagnetic field. One such proposal [32] proposes to represent qubits by the local modes of a linear array of non-interacting fermions, which become entangled simply as a consequence of Fermi statistics as the state of the fermions undergo mirror inversion. Another proposal [79] proposes to mediate interactions by using global control to transport a "virtual" mediating qubit, represented by a pair of reserved excited states for each lattice site arising from the tuning of the lattice.

Almost all of the models mentioned in this and the preceding section address only the creation of the cluster state (which is admittedly likely to be the more difficult task in performing one-way measurement based computation, due to the need to control multi-qubit operations). A departure from this is a proposal for producing cluster states with charge and flux qubits with quantum dots is described in [126], which is robust to the nonuniformity of the dot formation process. Measurement adaptations are explicitly performed by controlled rotations, which are performed by moderating the same continuous time Hamiltonian which gives rise to the cluster state itself.

A result of [105] is that neither the cluster state itself, (nor any other universal resource for the one-way model, can be the unique ground state of any "physically realistic" Hamiltonian; despite this, [11] describes a nearest-neighbor two-local Hamiltonian with a unique ground state on the grid, which may be interpreted as an *encoded* cluster state. This indicates that a cluster state (or equivalent) is likely to be physically attainable by cooling an appropriate system to its ground state, and performing measurements to obtain the cluster state itself (equivalently, performing single-qubit measurements in such a way as to *simulate* the a one-way measurement-based computation using a cluster).

Finally, a different approach to the creation of a cluster state is to encode the qubits in different energy modes of a single physical system. One proposal of this type [95] is to implement cluster states in the modes of an optical cavity, using highly peaked (or *squeezed*) distributions over continuous variables to represent computational basis states. The high degree of coherent control over optical cavities is touted as an advantage for this scheme; however, the degree of "squeezing" required to perform any given computation is not yet known.

2.1.3 Approaches to fault-tolerance in the one-way model

A pre-requisite of scalable quantum computation is the ability to protect the state of a system against noise. An alternative to achieving the ability to directly control the interactions of the system versus the environment is to apply generic techniques for protecting quantum information against uncontrolled operations: this approach is what is generally referred to as *fault tolerance* of quantum computation [84, 91].

Most of the attention to fault-tolerant approaches to one-way measurement-based computation have focused on the linear-optical proposals for implementation, where the noise operations (aside from those induced by imperfect realizations of logical gates) can be described in terms of loss errors and decoherence errors. A scheme for loss-tolerance is proposed in [129], which (provided perfect logical gates) can cope with photon losses of up to 50%, by using stabilizer techniques to allow would-be measurement results of lost photons to be inferred. However, [115] describe that this yields an exponential blow-up of error rate for other reasons. A modification of the scheme of [129] is also presented in [115], for which they suggest a trade-off exists between the error rates of different types of error. An optical protocol protecting against both polarization and loss errors is presented in [43], requiring Bell pairs and employing the fusion operator techniques of [27].

An analysis applicable to linear optical set-ups, but also to optical lattices, is presented in [80]. Using renormalization theory, they present a means of treating losses incurred due to non-deterministic entangling gates which eliminates the need to "route" states (*i.e.* conditionally operate in the case of other, failed, operations). This analysis yields scaling of physical requirements similar to hypothetical schemes where the entangling gates are deterministic, which may substantially improve the prospects of those physical implementations which rely on lossy, probabilistic operations.

Analyses of one-way based computation in decoherence free subspaces (*i.e.* subspaces of the state-space of a system which are naturally protected from noise by symmetries of the system) are presented in [125] for phase-damping errors in an optical lattice setting; the authors propose that the construction should be extensible to more general errors. An experimental demonstration for a corresponding proposal in linear optics is presented in [110].

Finally, a generic scheme for performing topological quantum computation natively in a three-dimensional cluster state (*i.e.* in which two-dimensional grid graphs are replaced by three-dimensional grids) is presented in [114]. The computation is performed in this proposal by simulating the evolutions of "defects" in a two-dimensional lattice over discrete time-steps; the trajectories of the defects trace out paths in the three-dimensional lattice, which may be braided to effect quantum computation (as briefly described for topological quantum computation in general, in Section 1.6).

2.2 Computing in the one-way measurement model

The one-way measurement model can be used to achieve universal quantum computation by performing *adaptive single-qubit measurements* — classically controlled measurement operations, which may depend on the results of prior measurements — and classically

controlled Pauli operations on the output qubits, which compensate for the randomness of the prior measurement results. (In the original presentation [112], the correction or *byproduct* operation is never explicitly performed, and is instead used to adapt the final measurement used to produce the read-out of the computation: in order to describe this model of computation as being universal in the sense of Definition 1-13, we include the conditional corrections in the description.[5])

Every operation after the preparation of the cluster state acts only on individual qubits — albeit possibly with classical control — and (except for the classically controlled unitaries by which we extend the model) discards the qubits it acts on. The cluster state is therefore consumed as a computational resource via measurement; hence the name of the model, which is irreversible, in contrast with the reversibility of closed-system Schrödinger dynamics.

We will spend most of this thesis considering aspects of measurement-based models of quantum computation which have been abstracted from the model of [112], with emphasis on one such model in particular (Definition 2-4) which retains what can be regarded as the essence of the original proposal in [112]. In doing so, we will implicitly describe schemes of translating unitary circuits into various models of quantum computation based on adaptive single-qubit measurements.

A remark on terminology. Despite the existence of other models of measurement-based computation, such as teleportation-based computation [90] or further relaxations of the one-way model as in [72], we will generally refer only to the models presented in this Chapter when we use the term *measurement-based quantum computation*. Because of the emphasis on *unitary* circuits associated with most research in quantum computation, we will use the terms *pattern* or *procedure* in place of the word "circuit" when describing well-defined compositions of operations in measurement-based models of computation, following conventions present in the literature (*e.g.* [38, 40, 26]), in order to reinforce the prominence of non-unitary operations in the model.

2.2.1 The cluster state model

We consider a slightly extended version of the model of [112] (which we call the *cluster state model*), as follows. For any $n, d \geqslant 1$, we include the map $\mathsf{N}_{G_{n,d}}$ which prepares the $n \times d$ cluster state on fresh qubits. The measurement operations used in [112] are M^{z} measurements as in (1.99) and, XY*-plane measurements* M^{α}, defined for angles $\alpha \in (-\pi, \pi]$ by

$$\mathsf{M}^{\alpha}_{v}(\rho) \;=\; \langle +_{\alpha}|_{v}\, \rho\, |+_{\alpha}\rangle_{v} \otimes |0\rangle\langle 0|_{\mathsf{s}[v]} \;+\; \langle -_{\alpha}|_{v}\, \rho\, |-_{\alpha}\rangle_{v} \otimes |1\rangle\langle 1|_{\mathsf{s}[v]}\;, \tag{2.5}$$

where the states $|\pm_{\alpha}\rangle$ are the ± 1-eigenstates of the operator $\cos(\alpha)X + \sin(\alpha)Y$,

$$|\pm_{\alpha}\rangle \;=\; \frac{1}{\sqrt{2}}|0\rangle \;\pm\; \frac{\mathrm{e}^{i\alpha}}{\sqrt{2}}|1\rangle\;. \tag{2.6}$$

[5] The one-way model, as presented in [112], is a model of "**CC**-universal" quantum computation in the terminology of [49]: however, the way in which we extend it by including classically controlled unitaries to produce a "**CQ**-universal" model of computation is already implicit in the analysis of [112].

We refer to α as the *measurement angle* of the measurement operator.[6] The XY-plane measurements are usually classically controlled, in that the sign of the measurement angle may depend on the parity of some sequence of bits — in particular, some sequence of prior measurement results.[7] Therefore, we introduce the abbreviation

$$\mathsf{M}_v^{\alpha;\beta} \quad = \quad \mathsf{M}_v^{(-1)^\beta \cdot \alpha} \tag{2.7}$$

for a classically controlled measurement performing either M^α or $\mathsf{M}^{-\alpha}$, depending on a boolean expression $\beta = \mathsf{s}[a] + \mathsf{s}[b] + \cdots$ representing the sum (modulo 2) of some set of measurement results. We refer to α as the *default angle* of the measurement (or the *default measurement angle*). It will be convenient to extend this further to accommodate changes in angle by a possible addition by π, which is not a part of the model of [112], but which will facilitate analysis later: therefore, we will write

$$\mathsf{M}_v^{\alpha;\beta,\gamma} \quad = \quad \mathsf{M}_v^{(-1)^\beta \cdot \alpha + \gamma\pi} \tag{2.8}$$

for boolean expressions β and γ; we call β a *sign dependency* γ a π*-dependency*. Similarly, as previously mentioned, we also extend the model of [112] by allowing operations X_v^β and Z_v^β on arbitrary qubits v, which perform the identity if $\beta = 0$ and an X or Z operation (respectively) on v if $\beta = 1$. Thus, the extended version of the model of [112] can be described as follows:

Definition 2-2. For any set of angles $\mathbb{A} \subseteq \mathbb{R}$ the *cluster state model* is the model of computation with the elementary gate set

$$\textsc{Cluster}(\mathbb{A}) \quad = \quad \left\{ \mathsf{N}_{G_{n,d}} \,,\ \mathsf{M}_v^{\mathsf{z}} \,,\ \mathsf{M}_v^{\alpha;\beta,\gamma} \,,\ \mathsf{X}_v^\beta \,,\ \mathsf{Z}_v^\beta \,,\ \mathrm{tr}_v \ \middle|\ n, d \geqslant 1,\ \alpha \in \mathbb{A} \,, \right.$$
$$\left. \beta, \gamma \ = \sum_{\mathsf{s}[v] \in S} \mathsf{s}[v] \ \text{ for any } S \subseteq B, \text{ and } v \in \mathbb{N} \times \mathbb{N} \right\}, \tag{2.9}$$

where B is an infinite set of bits and $\mathsf{s} : \mathbb{N} \times \mathbb{N} \longrightarrow B$ is injective. We will say that a $\mathfrak{P}$ procedure in the $\textsc{Cluster}(\mathbb{A})$ model is in *standard form* if it can be decomposed as $\mathfrak{P} = \mathsf{C} \circ \mathsf{M} \circ \mathsf{N}_{G_{n,d}}$, where M is a composition of *adaptive measurements* $\mathsf{M}_v^{\alpha;\beta}$ without π-dependencies, and where C is a composition of *correction operations* X_v^β and Z_v^β. We then call $\mathsf{N}_{G_{n,d}}$ the *preparation stage* of the procedure, M the *measurement stage* of the procedure, and C the *correction stage* of the procedure.

Because cluster states are stabilizer states, procedures in the $\textsc{Cluster}(\frac{\pi}{2}\mathbb{Z})$ model can be efficiently simulated by classical circuits by the Gottesman-Knill Theorem (page 48). However, procedures in $\textsc{Cluster}(\frac{\pi}{4}\mathbb{Z})$ fall outside of the domain of the stabilizer formalism; no corresponding result is known in that case.

[6] Representing the states $|\pm_\alpha\rangle$ in the Bloch sphere, α is the angle between of the state $|+_\alpha\rangle$ and the $|+\rangle = |+\mathsf{x}\rangle$ state, *i.e.* the X-axis.

[7] Note that this is a slight departure from the description of classically controlled measurements on page 40, where the measurement performed depends explicitly on the value of a *bit*, rather than abstractly on the parity of a set of many bits. While it is likely that the measurement basis would depend explicitly on some bit which stores the value of such a boolean expression in practical implementations, we will not consider explicitly how this parity is evaluated. However, in a serious consideration of *e.g.* the depth-complexity of quantum operations in the one-way model, how the parities are evaluated is an important detail.

In order to perform universal quantum computation, the one-way model must be able in particular to prepare (or approximate) arbitrary pure states for the qubits on which it act. The measurement process, however, produces random outcomes. In order to describe how the cluster state model can produce pure states by adaptive measurements and single-qubit unitary corrections, we give an account of this model in terms of unitary circuits. This can be facilitated by considering further extensions beyond the cluster state model.

2.2.2 The graph state model

Consider the varieties of states which may be obtained after we perform all Z measurements in a cluster state measurement pattern, and before we apply any other operation. Because Z measurements are not subject to adaptations based on prior measurements, we may commute the Z measurements to the beginning of the measurement phase of the procedure (whether or not it is in standard form) by applying the relations

$$\mathsf{M}^{\mathsf{z}}_v \, \mathsf{Z}^{\beta}_v \;=\; \mathsf{M}^{\mathsf{z}}_v \,, \qquad\qquad \mathsf{M}^{\mathsf{z}}_v \, \mathsf{X}^{\beta}_v \;=\; \mathsf{S}^{\beta}_{\mathsf{s}[v]} \, \mathsf{M}^{\mathsf{z}}_v \,; \tag{2.10}$$

for the latter relation, we define the *shift* operator[8] $\mathsf{S}^{\beta}_{\mathsf{s}[v]}$,

$$\mathsf{S}^{\beta}_{\mathsf{s}[v]} : \mathsf{Prob}(\mathsf{s}[v]) \longrightarrow \mathsf{Prob}(\mathsf{s}[v]) \tag{2.11a}$$

$$\mathsf{S}^{\beta}(\rho) \;=\; \sum_{x \in \{0,1\}} |x \oplus \beta\rangle\langle x| \; \rho \; |x\rangle\langle x \oplus \beta| \tag{2.11b}$$

acting on the bit $\mathsf{s}[v]$, where $a \oplus b$ is the sum of bits a and b modulo 2.

Because the cluster state is a stabilizer state, we may determine the state of the system after the Z measurements. For a measurement on a qubit v, K_v is the only generator of the stabilizer group $\mathcal{C}_{n,d}$ (as described in Definition 2-1) which does not commute with the observable Z_v. By the stabilizer formalism (Section 1.5.3), the state of the other qubits after a Z_v measurement where we obtain $\mathsf{s}[v] = r$ is then given by

$$\tilde{\mathcal{C}}^{(r)} \;=\; \left\langle \tilde{K}^{(r)}_u \right\rangle_{u \in V(G) \setminus v} \,, \tag{2.12}$$

where we have

$$\tilde{K}^{(r)}_u \;=\; \left\{ \begin{array}{rll} K_u \;= X_u \prod\limits_{w \sim u} Z_w \,, & & \text{if } u \not\sim v \\ (-1)^r Z_v K_u \;= (-1)^r \, X_u \prod\limits_{\substack{w \sim u \\ w \neq v}} Z_w \,, & & \text{if } u \sim v \end{array} \right\} , \tag{2.13}$$

and where $\sim$ is the adjacency relation in the grid. In the case where $\mathsf{s}[v] = 0$, the state of the system is stabilized by a group generated by $\tilde{K}^{(0)}_u$, which like the generators K_u are

[8] The shift operator is usually not defined in accounts of measurement-based computation (with [40] being an exception), probably because it is a very simple "classical" operation. However, in any implementation of quantum computers based on single-qubit measurements, shift operators are likely to play an important role in describing how the parity expressions β are computed in the midst of a computation. Thus, while much of the analysis later in the thesis follows the convention of eliminating shift operations from measurement-based procedures, I explicitly allow shift operators as elementary gates, and extend the sense of "standard forms" found in the literature to allow shift operators to occur between measurements.

tensor products of an X operator and some number of Z operators. Otherwise, we may convert the state of the system to that stabilized by $\tilde{\mathcal{C}}^{(0)}$ by performing the operations Z_u on every qubit u adjacent to v in $G_{n,d}$: because Z_u anticommutes with X_u and commutes with Z_w for any qubit w, we have

$$\mathsf{Z}_u\left(\tilde{K}_u^{(1)}\right) \;=\; Z_u\left[-X_u\prod_w Z_w\right]Z_u \;=\; X_u\prod_w Z_w \;=\; \tilde{K}_u^{(0)}\,. \tag{2.14}$$

Then, we can select for the post-measurement state corresponding to $\mathsf{s}[v]=0$ by performing the operation

$$\mathsf{B}_v \;=\; \prod_{u\sim v}\mathsf{Z}_u^{\mathsf{s}[v]}\,. \tag{2.15}$$

We call such an operation a *byproduct* operation. To perform pure-state computation, there is no loss in generality in selecting for the post-measurement state corresponding to $\mathsf{s}[v]=0$, rather than $\mathsf{s}[v]=1$, as they are equivalent up to conditional unitaries which may be performed in the cluster-state model in any case.

As we noted, the post-measurement state produced in this way is stabilized by operators $\tilde{K}_u^{(0)}$ for $u\in G_{n,d}\setminus v$, which are each tensor products of X and Z operators, but which are distinct from the stabilizers of $\mathcal{C}_{n,d}$. Furthermore, by construction, each $\tilde{K}_u^{(0)}$ acts on u with an X operation, and acts on a qubit $w\in V(G_{n,d})$ with a Z operation if and only if $w\neq v$ and $w\sim u$: that is, if w is adjacent to u in the graph $G_{n,d}\setminus v$. Thus, a Z measurement on v (followed by the byproduct B_v if required) corresponds to removing a vertex represented by v from the $n\times d$ grid graph, producing a state similar to the cluster state, but defined on a different graph. This motivates the following definition:

Definition 2-3. For an arbitrary graph G, the *graph state* corresponding to G is a stabilizer state $\rho_G=|G\rangle\langle G|$, where $|G\rangle$ is a unit-vector stabilized by the group $\mathcal{S}_G=\langle K_{G,u}\rangle_{u\in V(G)}$ for generators given by

$$K_{G,v} \;=\; X_v\prod_{w\in N_G(v)} Z_w\,, \tag{2.16}$$

and where $N_G(v)$ denotes the set of vertices adjacent to v in the graph G.

Applying the same analysis as for performing a Z measurement on a qubit v of the $n\times d$ cluster state, we may show that for any graph G and vertex $v\in V(G)$, we have

$$\left[\prod_{w\in N_G(v)}\mathsf{Z}_w^{\mathsf{s}[v]}\right]\mathsf{M}_v^{\mathsf{z}}(\rho_G) \;=\; \rho_{[G\setminus v]}\,. \tag{2.17}$$

Any Z corrections on qubits w which are also to be measured with a Z measurement can be absorbed into the $\mathsf{M}_w^{\mathsf{z}}$ operation as in (2.10). Let S be the set of vertices measured with a M^{z} operation: Z corrections then accumulate on the neighbors of vertices in S, yielding the transformation

$$\left[\prod_{w\in N(S)}\mathsf{Z}_w^{\left(\sum_{v\in S_w}\mathsf{s}[v]\right)}\right]\left[\prod_{v\in S}\mathsf{M}_v^{\mathsf{z}}\right], \tag{2.18}$$

where $N(S)$ is the set of qubits in $G_{n,d}$ adjacent to (but not contained in) S in the $n \times d$ grid, and $S_w = N_{G_{n,d}}(w) \cap S$. This operation serves to transform $\rho_{G_{n,d}}$ to a graph state corresponding to some induced subgraph, $\rho_{[G_{n,d} \setminus S]}$.

We may abstract this process of producing graph states from cluster states via vertex removal, to describe a more general measurement-based model. For an *arbitrary* graph G, let N_G be the map preparing the graph state $|G\rangle\langle G|$. Then we may consider a *graph state model* of computation, generalizing the cluster state model, by allowing preparation maps N_G for arbitrary G (or, for instance, for G from some broader family of graphs) rather than just for grids $G_{n,d}$.

2.2.3 Open graph states, geometries, and the general "one-way" model

As with other models of quantum computing, in order to describe how to achieve universal quantum computation in measurement-based models of quantum computing, it is convenient to reduce to universality for unitary transformations. This motivates another extension beyond graph-state models — which can only prepare independent graph states and perform classically controlled single-qubit operations on them — to a model which permits multi-qubit operations on previously allocated qubits, just as we generalized universality for quantum computation to universality for CPTP transformation.

A natural approach (and one also suggested in [112]) is to extend beyond graph states to maps which embed quantum states $\rho \in \mathsf{D}(I)$ into a stabilizer code similar to a graph state. Here, I is a set representing an *input system* of qubits, which we include in the vertex-set $I \subseteq V(G)$ of a graph G: we then consider the stabilizer code described by a group $\mathcal{S}_{G,I}$ generated by the operators $K_{G,v}$ as in (2.16), but ranging only over qubits $v \in V(G) \setminus I$. In order to describe an explicit map for this embedding, consider how the operators $K_{G,v}$ may be generated by Clifford group operations from simpler operations. From (1.94), we may observe that

$$K_{G,u} \quad = \quad \left[\prod_{v \in N_G(u)} \mathsf{E}_{u,v} \right] (X_u) \,, \tag{2.19}$$

where this product may be taken in any order; then, the code which is stabilized by $\mathcal{S}_{G,I} = \langle K_{G,u} \rangle_{u \in V(G) \setminus I}$ can be obtained from the code stabilized by $\mathcal{S}_{T,I} = \langle X_u \rangle_{u \in V(G) \setminus I}$ by performing the operation

$$\mathsf{E}_G \quad = \quad \prod_{uv \in E(G)} \mathsf{E}_{u,v} \,. \tag{2.20}$$

The operators X_u also generate the stabilizer group of a graphical code, for the "trivial" graph T on the vertices $V(G)$ but with no edges. This code consists of states $|+\rangle_u$ on each $u \in V(G) \setminus I$, with no constraints imposed on the qubits of I. Thus, the map $\mathcal{E}_{G,I} : \mathsf{D}(I) \longrightarrow \mathsf{D}(V(G))$ encoding the qubits of I into the code stabilized by $\mathcal{S}_{G,I}$ can be given by

$$\mathcal{E}_{G,I} \quad = \quad \left[\prod_{uv \in E(G)} \mathsf{E}_{u,v} \right] \left[\prod_{u \in V(G) \setminus I} \mathsf{N}^{\mathsf{x}}_u \right] \quad = \quad \mathsf{E}_G \, \mathsf{N}_{I^{\mathrm{c}}} \,, \tag{2.21}$$

where we abbreviate $I^{\mathsf{c}} = V(G) \setminus I$. We may refer to $\mathcal{E}_{G,I}$ as an *open graph state* encoding.[9] However, in contrast to the CLUSTER($\mathbb{A}$) model, we do not make the embedding $\mathcal{E}_{G,I}$ a primitive operation of the corresponding model of computation, and instead define the following model:

Definition 2-4. For any set of angles $\mathbb{A} \subseteq \mathbb{R}$, infinite set V of qubits, and infinite set B of bits, the *one-way model* is the model of computation with the elementary gate set

$$\text{ONEWAY}(\mathbb{A}) \;=\; \left\{ \mathsf{N}_v^{\mathsf{x}},\ \mathsf{E}_{u,v},\ \mathsf{M}_v^{\mathsf{z}},\ \mathsf{M}_v^{\alpha;\beta,\gamma},\ \mathsf{X}_v^{\beta},\ \mathsf{Z}_v^{\beta},\ \mathsf{S}_{\mathsf{s}[v]}^{\beta},\ \mathrm{tr}_v \;\middle|\; u,v \in V,\ \alpha \in \mathbb{A}, \right.$$
$$\left. \text{and}\ \ \beta,\gamma \;=\; \sum_{\mathsf{s}[v]\in S} \mathsf{s}[v] \ \ \text{for any } S \subseteq B \right\}. \tag{2.22}$$

We will say that a procedure $\mathfrak{P}$ in the ONEWAY($\mathbb{A}$) model is in *standard form* if it can be decomposed as $\mathfrak{P} = \mathsf{C} \circ \mathsf{M} \circ \mathsf{E}_G \circ \mathsf{N}_S$ for C consisting of corrections, M consisting of shift operations and measurement operations with no π-dependencies, E_G consisting of a product of *entangler* operations $\mathsf{E}_{u,v}$ for distinct pairs $\{u,v\} \subseteq V(G)$ for some graph G, and N_S consisting of preparation operations $\mathsf{N}_v^{\mathsf{x}}$ for $v \in S \subseteq V(G)$. We refer to these as the *correction*, *measurement*, *entangling*, and *preparation* stages respectively.

The model of measurement-based quantum computing which later chapters will be primarily concerned with is the model ONEWAY($\mathbb{A}$) for sets of angles $\mathbb{A} \subseteq \mathbb{R}$. We will refer to models of this form generically as *the one-way model*, and refer to "one-way procedures", "one-way patterns", *etc.* when describing algorithms in the one-way model.

The purpose of extending to open graph encodings is to make possible the *composition* of one-way patterns. Therefore, in addition to the extension to admit an input subsystem, we also explicitly identify *output* subsystems of measurement-based procedures, which simply consists of those qubits which are not measured (and the classical results of the measurements). We may then achieve universality for unitary transformations by performing an appropriate set of elementary unitary transformations — effectively simulating a unitary circuit in doing so.

A specification of an open graph state encoding, together with the set of qubits to be left unmeasured, effectively captures the *structural* information (that aspect which is independent of measurement observables and measurement results) of a one-way measurement procedure, which we may describe as follows:

Definition 2-5. For a measurement based procedure $\mathfrak{P}$ in the one-way model, the *geometry underlying* $\mathfrak{P}$ is a triple (G, I, O) where

(i) G is the *entanglement graph*, whose vertices are labelled by the qubits on which $\mathfrak{P}$ acts, and whose edges are given by the operations of the entangling phase of $\mathfrak{P}$;

[9] This terminology is similar to, but different from, that used in *e.g.* [26, 96], who use the term "open graph state" to refer to what we call a *geometry* in Definition 2-5 (in agreement with [37]).

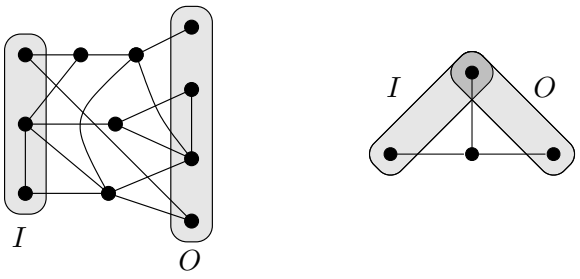


FIGURE 2-2. Illustration of two geometries for one-way procedures. Labels for the vertices have been omitted. Note in particular that the sets I and O may overlap.

(*ii*) $I \subseteq V(G)$ is the *input subsystem* of $\mathfrak{P}$, consisting of the qubits which are not allocated by the preparation phase of $\mathfrak{P}$;

(*iii*) $O \subseteq V(G)$ is the *output subsystem* of $\mathfrak{P}$, consisting of the qubits which are not discarded by the measurement phase of $\mathfrak{P}$.

We will often illustrate geometries in the style shown in Figure 2-2.

When composing one-way computations, we will often be interested in converting the composition into standard form, in which case no qubit can be allocated after measurement operations have occurred. Therefore, when composing one-way procedures $\mathfrak{P} = \mathfrak{P}_2 \circ \mathfrak{P}_1$, we will be interested in restricting the qubits which are allocated in $\mathfrak{P}_2$ to be distinct from any qubits involved in $\mathfrak{P}_1$. We can express this in terms of a sense of composing the *geometries* of these patterns, using the constraints described in Lemma 1-2 (page 16):

Definition 2-6. For two geometries $\mathcal{G}_1 = (G_1, I_1, O_1)$ and $\mathcal{G}_2 = (G_2, I_2, O_2)$, we say that $\mathcal{G}_1$ and $\mathcal{G}_2$ are *composable* if $\big[V(G_1) \cap V(G_2)\big] \subseteq \big[I_2 \cap O_1\big]$. The *composition* $\bar{\mathcal{G}} = \mathcal{G}_2 \circ \mathcal{G}_1$ is then defined by $\bar{\mathcal{G}} = (\bar{G}, \bar{I}, \bar{O})$, where

$$V(\bar{G}) = V(G_1) \cup V(G_2)\,, \tag{2.23a}$$
$$E(\bar{G}) = E(G_1) \triangle E(G_2)\,, \tag{2.23b}$$
$$\bar{I} = I_1 \cup (I_2 \setminus O_1)\,, \tag{2.23c}$$
$$\bar{O} = O_2 \cup (O_1 \setminus I_2)\,, \tag{2.23d}$$

where $S \triangle T = [S \setminus T] \cup [T \setminus S]$ is the symmetric difference of S and T. Two one-way procedures $\mathfrak{P}_1$ and $\mathfrak{P}_2$ are said to be *properly composable* if their underlying geometries $\mathcal{G}_1$ and $\mathcal{G}_2$ are composable; the underlying geometry of the procedure $\mathfrak{P}_2 \circ \mathfrak{P}_1$ is then $\mathcal{G}_2 \circ \mathcal{G}_1$.

Figure 2-3 illustrates the composition of two geometries. From this point onwards, we will suppose that two composable patterns $\mathfrak{P}_1$ and $\mathfrak{P}_2$ are properly composable: we lose no generality in doing this, as we may "relabel" the qubits acted on by the patterns $\mathfrak{P}_1$ and $\mathfrak{P}_2$ to achieve this if necessary.

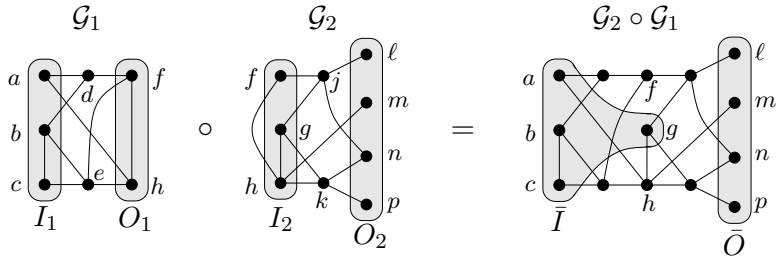


FIGURE 2-3. Illustration of the composition of two generic geometries for one-way procedures. Notice that composition in the diagrams is left-to-right, while the composition in written notation is right-to-left.

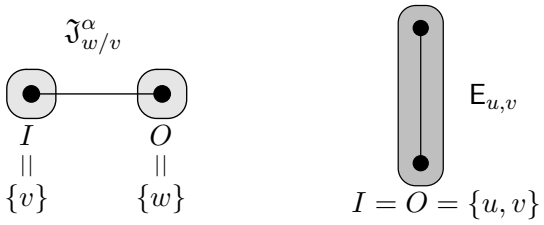


FIGURE 2-4. Illustration of the geometries of two primitive one-way patterns. (Note that the input and output subsystems of the pattern on the right coincide.)

2.2.4 Universality of the one-way and graph-state models

We may now illustrate how to perform universal quantum computation in the one-way model, using the construction illustrated in [40], which constructs measurement patterns which "simulate" unitary circuits. Consider the one-way patterns

$$\mathfrak{J}^{\alpha}_{w/v} \;=\; \mathsf{X}^{\mathsf{s}[v]}_{w}\,\mathsf{M}^{-\alpha}_{v}\,\mathsf{E}_{v,w}\,\mathsf{N}^{\mathsf{x}}_{w}\;, \qquad\qquad \mathsf{E}_{u,v}\;, \tag{2.24}$$

whose underlying geometries are illustrated in Figure 2-4. The single entangler operation $\mathsf{E}_{u,v}$ in particular is a simple one-way procedure in its own right, and can be interpreted as representing the same operation between two wires in a unitary circuit model such as PHASES($\mathbb{A}$). To describe the $\mathfrak{J}^{\alpha}_{w/v}$ operation, note that the first two operations put the system into a stabilizer code which is stabilized by the two operator group $\mathcal{S} = \{\mathbb{1}_v \otimes \mathbb{1}_w\,,\; Z_v \otimes X_w\}$. We can decompose $\mathsf{M}^{-\alpha}_{v}$ into a Z-rotation and an X measurement:

$$\begin{aligned}
\mathsf{M}^{-\alpha}_{v}(\rho) \;&=\; \langle +_{(-\alpha)}|_v\;\rho\;|+_{(-\alpha)}\rangle_v \;\otimes\; |0\rangle\langle 0|_{\mathsf{s}[v]} \;+\; \langle -_{(-\alpha)}|_v\;\rho\;|-_{(-\alpha)}\rangle_v \;\otimes\; |1\rangle\langle 1|_{\mathsf{s}[v]} \\
&=\; \langle +|_v \Big[R_{\mathsf{z}}(\alpha)_v\;\rho\;R_{\mathsf{z}}(-\alpha)_v\Big]\,|+\rangle_v \otimes |0\rangle\langle 0|_{\mathsf{s}[v]} \;+ \\
&\qquad\qquad \langle -|_v \Big[R_{\mathsf{z}}(\alpha)_v\;\rho\;R_{\mathsf{z}}(-\alpha)_v\Big]\,|-\rangle_v \otimes |1\rangle\langle 1|_{\mathsf{s}[v]} \\
&=\; \mathsf{M}^{\mathsf{x}}_{v}\,\mathsf{R}^{\alpha}_{v}(\rho)\;.
\end{aligned} \tag{2.25}$$

Because the operator $R_{\mathsf{z}}(\alpha)$ is diagonal, R_v^α commutes with the operation $\mathsf{E}_{v,w}$ as well as $\mathsf{N}_w^{\mathsf{x}}$. Then, we have

$$\begin{aligned}\mathfrak{J}_{w/v}^\alpha &= \mathsf{X}_w^{\mathsf{s}[v]}\,\mathsf{M}_v^{-\alpha}\,\mathsf{E}_{v,w}\,\mathsf{N}_w^{\mathsf{x}} = \mathsf{X}_w^{\mathsf{s}[v]}\,\mathsf{M}_v^{\mathsf{x}}\,\mathsf{R}_v^\alpha\,\mathsf{E}_{v,w}\,\mathsf{N}_w^{\mathsf{x}} \\ &= \mathsf{X}_w^{\mathsf{s}[v]}\,\mathsf{M}_v^0\,\mathsf{E}_{v,w}\,\mathsf{N}_w^{\mathsf{x}}\,\mathsf{R}_v^\alpha = \mathfrak{J}_{w/v}^0\,\mathsf{R}_v^\alpha\,. \end{aligned} \tag{2.26}$$

Then, it suffices to consider the effect of $\mathfrak{J}_{w/v}^0$, which by linearity we may characterize by the effect on any four linearly independent 2×2 operators. We will consider the effect of $\mathfrak{J}_{w/v}^0$ on the Pauli operators via the stabilizer formalism, as follows. As we noted above, the maps $\mathsf{E}_{v,w}\,\mathsf{N}_w^{\mathsf{x}}$ encodes a single-qubit state into the code stabilized by $\{\mathbb{1}_v\otimes\mathbb{1}_w\,,\ Z_v\otimes X_w\}$. An X_v observable measurement then performs an isometry, and performing a $\mathsf{X}_w^{\mathsf{s}[v]}$ correction selects the post-measurement state result corresponding to $\mathsf{s}[v]=0$. Thus, we may reduce how $\mathfrak{J}_{w/v}^0$ transforms the Pauli operators on v to how it transforms X_v and Z_v in particular. We may then compute

$$\langle\, X_v\,\rangle \xmapsto{\mathsf{N}_w^{\mathsf{x}}} \left\langle \begin{matrix} X_v\otimes\mathbb{1}_w\ , \\ \mathbb{1}_v\otimes X_w \end{matrix} \right\rangle \xmapsto{\mathsf{E}_{v,w}} \left\langle \begin{matrix} X_v\otimes Z_w\ , \\ Z_v\otimes X_w \end{matrix} \right\rangle \xmapsto{\mathsf{X}_w^{\mathsf{s}[v]}\,\mathsf{M}_v^{\mathsf{x}}} \langle\, Z_w\,\rangle\ , \tag{2.27a}$$

$$\begin{aligned}\langle\, Z_v\,\rangle \xmapsto{\mathsf{N}_w^{\mathsf{x}}} \left\langle \begin{matrix} Z_v\otimes\mathbb{1}_w\ , \\ \mathbb{1}_v\otimes X_w \end{matrix} \right\rangle \xmapsto{\mathsf{E}_{v,w}} \left\langle \begin{matrix} Z_v\otimes\mathbb{1}_w\ , \\ Z_v\otimes X_w \end{matrix} \right\rangle \\ = \left\langle \begin{matrix} Z_v\otimes\mathbb{1}_w\ , \\ \mathbb{1}_v\otimes X_w \end{matrix} \right\rangle \xmapsto{\mathsf{X}_w^{\mathsf{s}[v]}\,\mathsf{M}_v^{\mathsf{x}}} \langle\, X_w\,\rangle\ . \end{aligned} \tag{2.27b}$$

In both cases, we have $\mathfrak{J}_{w/v}^0(\sigma_v) = \mathsf{H}_w(\sigma_w)$, which then holds for all $\sigma\in\mathsf{L}(\mathcal{H}_v)$. Therefore, we have

$$\mathfrak{J}_{w/v}^\alpha(\rho_v) = [\mathsf{H}\circ\mathsf{R}^\alpha(\rho)]_w = [\mathsf{J}^\alpha(\rho)]_w\ . \tag{2.28}$$

Thus, the operations performed by the patterns $\mathfrak{J}_{w/v}^\alpha$ and $\mathsf{E}_{v,w}$ correspond to the unitary operations of the Jacz($\mathbb{A}$) unitary circuit model defined in Definition 1-20. Using stable index notation, we may then define an operator homomorphism Φ mapping unitary circuits in the Jacz($\mathbb{A}$) model to (non-standard form) measurement procedures in the OneWay($\mathbb{A}$) model as follows. For a stable index tensor expression C, we may require without loss of generality that the terms of C are in an order corresponding to the order in which the operations are performed in the circuit (*e.g.* as in an ordinary operator expression for C). Identifying advanced/deprecated tensor indices in C with allocated/discarded qubits in the one way pattern (using the correspondence described on page 33), we define

$$\Phi\Big(J(\alpha)^{[\frac{w}{v}]}\Big) = \mathfrak{J}_{w/v}^\alpha\,; \qquad \Phi\Big(\wedge Z^{[v,w]}\Big) = \mathsf{E}_{v,w}\ : \tag{2.29}$$

the unitary transformation described by a unitary circuit C in stable index notation is then also performed by the one-way procedure $\mathfrak{P}=\Phi(C)$.[10]

[10] We have deliberately neglected the allocation of the result register $\mathsf{s}[v]$ in this account of how unitaries are performed, in part because we are primarily interested in how the quantum registers are transformed rather than the classical information produced as a result. We conventionally ignore classical bits in discussions of the "transformations performed" by measurement patterns: this can be formalized by adopting a convention of implicitly tracing out all result registers at the end of a quantum computation.

2.2.5 Standardization of one-way patterns, and the DKP constructions

The homomorphism Φ from circuits to one-way patterns is the main instrument in the simplest constructions of standardized measurement patterns, presented by Danos, Kashefi, and Panangaden in [40]. In this section, we describe these constructions, which we will refer to as the DKP constructions for one-way patterns.

For any one-way procedure — and for the one-way procedures arising from the homomorphism Φ of (2.29) in particular — we may obtain a standardized one-way procedure by accumulating correction operations from the beginning of the pattern and moving them to the end. To do so, we must commute them past preparation operations (which we may easily do) and entangler operations (which induces further corrections), and absorbing them into the measurement operations to describe how measurements are adapted. We describe how this is done in this section, essentially via the stabilizer formalism, representing the correction operations as a product of Pauli operators, together with boolean expressions representing their classical dependencies.

Starting from the beginning of the pattern, we scan towards the end of the pattern, maintaining a set of correction operations that we encounter. We attempt to commute each correction operator past the other operators in the pattern, as follows:

1. As the preparation maps $\mathsf{N}_v^{\mathsf{x}}$ are always the first operations on any qubit $v \in I^{\mathsf{c}}$ (in particular because, as we have restricted ourselves to properly composable patterns, no qubit is discarded and subsequently re-allocated), any operations which precede a preparation map also commutes with it. Thus, we may trivially commute any correction operations past $\mathsf{N}_v^{\mathsf{x}}$ operations.

2. Similarly, we may commute correction operations past any entangling operator $\mathsf{E}_{u,v}$. However, operations of the form $\mathsf{E}_{u,w}$ do not commute with correction operations X_v^{β} : in particular, we have

$$\mathsf{E}_{u,v}\mathsf{X}_v^{\beta} \;=\; \mathsf{Z}_u^{\beta}\mathsf{X}_v^{\beta}\mathsf{E}_{u,v} \;. \tag{2.30}$$

   Thus, in commuting an X_v^{β} operation past an entangler acting on w and another qubit u, we induce an additional correction Z_u^{β}, increasing the complexity of the correction operation.

3. Finally, for a measurement on any qubit v that we encounter, we absorb the corrections accumulated on v into the bases of measurement, according to the equations

$$\mathsf{M}_w^{\alpha;\beta,\gamma}\mathsf{X}_w^{\beta'} \;=\; \mathsf{M}_w^{\left[(-1)^{\beta+\beta'}\cdot\,\alpha\,+\,\gamma\pi\right]} \;=\; \mathsf{M}_w^{\alpha;(\beta+\beta'),\gamma} \;, \tag{2.31a}$$

$$\mathsf{M}_w^{\alpha;\beta,\gamma}\mathsf{Z}_w^{\gamma'} \;=\; \mathsf{M}_w^{\left[(-1)^{\beta}\cdot\,\alpha\,+\,(\gamma+\gamma')\pi\right]} \;=\; \mathsf{M}_w^{\alpha;\beta,(\gamma+\gamma')} \;. \tag{2.31b}$$

   This decreases the complexity of the corrections which we keep track of, transferring the information of the corrections on v to the measurement on v.

The result is a measurement procedure with a phase of classically controlled correction operations on the outputs, preceded by a mixed sequence of preparation maps, entanglers,

and corrections with both sign- and π-dependencies. Again, as the preparations are always the first operation performed on a qubit, and the measurements always the last, the operations in this mixed sequence all commute; we may then separate them into preparation, entangling, and measurement phases.

The complexity of the operations described above may be performed in time polynomial in the number of qubits n operated on in the pattern. For an arbitrary one-way pattern, at each point in time, we must track at most one $\mathsf{X}_v^{\beta_v}$ operation and one $\mathsf{Z}_v^{\gamma_v}$ operation per qubit, with both β and γ being boolean expressions of complexity at most $O(n)$. Each entangler operation acting on a qubit v then induces a change in the expression $\mathsf{Z}_v^{\gamma_v}$ of complexity $O(n)$, and there are $O(\deg_G(v))$ entanglers which may act on v, where G is the graph of the geometry underlying the pattern. Then, the complexity of updating the corrections on each qubit throughout the pattern is at most $O(n \deg_G(v))$; accumulated across all qubits, the work required is then $O\big(n \sum_v \deg_G(v)\big) = O(nm)$, where $m = |E(G)|$.

For one-way patterns arising out of the homomorphism Φ, we may achieve a much better bound, arising from the fact that the corrections are not fully general: in particular, the correction on each qubit w is initially just an operation $\mathsf{X}_w^{\mathsf{s}[v]}$ for some qubit v. Following the same analysis as above, the work required to standardize such a pattern is then at most $O\big(\sum_v \deg_G(v)\big) = O(m)$.

In the case of the one-way patterns arising from Φ, the resulting (non-standard form) procedure is one of the constructions for one-way patterns described in [40], which we will refer to in later chapters as follows:

Definition 2-7. The *simplified DKP construction for one-way patterns* is the procedure for producing OneWay($\mathbb{A}$) patterns from unitary circuits C in the Jacz($\mathbb{A}$) model by

1. constructing the measurement pattern $\Phi(C)$, where Φ is the mapping defined in (2.29);[11]
2. commuting the preparations and entangler maps of the resulting concatenated procedure to the beginning of the procedure;
3. absorbing the resulting classically controlled corrections X_v^{β} and Z_v^{γ} to the end of the measurement procedure, absorbing them into the measurements on the qubits in $V(G) \setminus O$ when required by the relations of (2.31).

A standardized procedure can be easily obtained from the simplified DKP construction by eliminating π-dependencies from measurements by observing that $|\pm_{(\alpha+\pi)}\rangle = |\mp_\alpha\rangle$. Then, π-dependencies may be abstracted away from measurements via shift operators, using the equation

$$\mathsf{M}_v^{\alpha;\beta,\gamma} \;=\; \mathsf{M}_v^{(-1)^\beta\cdot\alpha+\gamma\pi} \;=\; \mathsf{S}_{\mathsf{s}[v]}^{\gamma}\,\mathsf{M}_v^{(-1)^\beta\cdot\alpha} \;=\; \mathsf{S}_{\mathsf{s}[v]}^{\gamma}\,\mathsf{M}_v^{\alpha;\beta}\,. \tag{2.32}$$

[11] The construction of [40] does not make use of stable index tensor notation for circuits, but rather through an equivalent consideration of formal semantics of symbols in a measurement calculus corresponding to the CPTP maps in the gate set OneWay($\mathbb{A}$) which we defined above.

That is, rather than potentially "rotating the apparatus by 180 degrees" to exchange the states $|\pm_\alpha\rangle$ in a given measurement, we conditionally change the measurement result $\mathsf{s}[v]$ by adding (modulo 2) the expression γ.

There are two further techniques for simplifying a measurement procedure arising from the simplified DKP construction, which are also described in [40]:

Pauli simplifications. It is easy to verify that $|\pm_0\rangle = |\pm\mathsf{x}\rangle$, and $|\pm_{\pi/2}\rangle = |\pm\mathsf{y}\rangle$; we may use this to further reduce measurement dependencies for measurements $\mathsf{M}^{\alpha;\beta}$ for α a multiple of $\pi/2$, as follows. For α any multiple of π, the measurement operators M_v^α and $\mathsf{M}_v^{-\alpha}$ on any qubit v are equivalent, as they differ by an integer multiple of 2π: changes of the sign of the angle of measurement have no impact, and as a result such a measurement does not have any classical dependencies. For α an odd multiple of $\pi/2$, we have $-\alpha \equiv \alpha + \pi \pmod{2\pi}$, in which case we may also extract the classical dependency into a shift operator

$$\mathsf{M}_v^{(-1)^\beta \cdot \pi/2} \quad = \quad \mathsf{M}_v^{\pi/2 + \beta\pi} \quad = \quad \mathsf{S}_{\mathsf{s}[v]}^\beta \, \mathsf{M}_v^{\pi/2} \; . \tag{2.33}$$

These measurements (together with any Z measurements $\mathsf{M}_v^{\mathsf{z}}$) are described as *Pauli measurements*, and the procedure of eliminating dependencies of these measurements on prior measurement results as *Pauli simplifications*. By performing Pauli simplifications, the Pauli measurements in a one-way pattern may be performed independently of any other measurement result, and in particular they may be commuted to the beginning of the measurement phase of any one-way pattern.

Signal shifting. After bringing a one-way pattern from the simplified DKP construction into standard form and performing Pauli simplifications, we may eliminate any shift operators by commuting them to the end of a measurement procedure and discarding them. For any one-way model operation $\mathsf{Op}_w^{\langle\!\langle\, \mathsf{s}[v] \,\rangle\!\rangle}$ on a qubit w which depends on a measurement result $\mathsf{s}[v]$, we have the relation

$$\mathsf{Op}_u^{\langle\!\langle\, \mathsf{s}[w] \,\rangle\!\rangle} \, \mathsf{S}_{\mathsf{s}[w]}^\beta \quad = \quad \mathsf{S}_{\mathsf{s}[w]}^B \, \mathsf{Op}_u^{\langle\!\langle\, \mathsf{s}[w] + \beta \,\rangle\!\rangle} \tag{2.34}$$

where $\langle\!\langle\, \mathsf{s}[v] + \beta \,\rangle\!\rangle$ is the result of substituting $\mathsf{s}[v]$ in the boolean expression with the value of $\mathsf{s}[v] + \beta$. We may use such relations to commute all shift operators to the end of a one-way procedure. Because shift operators are purely classical operations, shift operations at the end of a measurement procedure have no effect on the quantum output, and so they may be eliminated. We call this procedure of eliminating shift operations *signal shifting*. (If this is performed simultaneously with the standardization procedure described above, we may accumulate shift operators along with the correction operations, and ultimately commute the shift operators past the corrections: an identical analysis for the run-time of this procedure holds as for standardization without signal shifting.)

The measurement pattern that arises from performing these transformations is then a standard form one-way pattern, where the measurement phase begins with a collection of commuting Pauli measurements; and where the operational dependencies of every measurement are represented explicitly in the classical controls of each operation, as a result of eliminating the shift operations. We will refer to this construction as follows:

Definition 2-8. The (*complete*) *DKP construction for one-way patterns* is the procedure for producing OneWay($\mathbb{A}$) patterns from unitary circuits in the Jacz($\mathbb{A}$) model by

1. obtaining a measurement pattern from C, via the simplified DKP construction of Definition 2-7 ;
2. eliminating π-dependencies, via the relation (2.32) ;
3. performing Pauli simplifications and signal shifting; and
4. commuting Pauli measurements to the beginning of the measurement phase.

The analysis of these constructions prove the following result, by the discussion following Definition 1-20 on page 27:

Theorem 2-1. *The* OneWay($\mathbb{A}$) *model is* (a) *universal for quantum computation and for unitary transformations when* $\mathbb{A} = \mathbb{R}$*; and* (b) *approximately universal for quantum computation and for unitary transformations when* $\mathbb{A} = \frac{\pi}{4}\mathbb{Z}$*. Furthermore, we may without loss of generality require that the measurement pattern is standardized, contains no shift operations, and that Pauli measurements are performed before any non-Pauli measurement operation.*

2.2.6 The simplified RBB construction

We may apply the techniques of the preceding sections to show the universality of the cluster-state model, by adapting the construction to the constrained case where the graph is an $n \times d$ grid. We will do this by showing how the techniques of the preceding sections may be used to describe (a modest extension of) the elementary constructions of Raussendorf, Browne, and Briegel [113] for quantum computation in the cluster state model.[12] We will refer to the resulting construction as "the simplified RBB construction".

The main difference between cluster-state based computation and what we have described in Definition 2-4 as general one-way patterns are the topological constraints on the interactions between qubits imposed by the grid graph. These parallel a commonly imposed constraint in unitary circuit models: a *linear nearest neighbor* unitary circuit model is one where we a impose a linear ordering on the qubits by assigning them to points on a line (possibly corresponding *e.g.* to physical locations of the qubits as physical systems), and operations are restricted to operate only on adjacent qubits (or more generally, on blocks consecutive qubits, for operations acting on two or more qubits). Linear nearest neighbor models can simulate more general models by performing swap operations on adjacent qubits, effecting permutations of the qubits in the linear ordering,

[12] The constructions of [113, page 5] assume arbitrary precision of measurement angles, allowing the exact implementation of arbitrary single-qubit using a fixed "horizontal width" when embedded in the grid. For an account of approximate universality, a trivial but useful extension is to describe computations in terms of "chains" of $J(\alpha)$ operations on distinct qubits of arbitrary length, and then describe how to compensate for differences in horizontal positions in the grid.

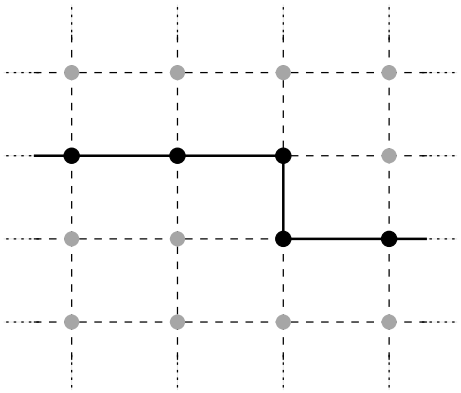


FIGURE 2-5. Illustration of an embedding of a chain in the grid. Grey dots and broken lines represent qubits and entanglement relations which are effectively removed from the cluster state by Z measurements, to produce a graph state whose underlying graph contains a sequence of vertices with degree 2.

in order to allow arbitrary collections of qubits to interact. It is easy to show that the operation

$$\mathsf{SWAP}_{u,v} \quad = \quad \mathsf{H}_u\mathsf{H}_v\mathsf{E}_{u,v}\mathsf{H}_u\mathsf{H}_v\mathsf{E}_{u,v}\mathsf{H}_u\mathsf{H}_v\mathsf{E}_{u,v} \tag{2.35}$$

interchanges independent pure states of two neighboring qubits u and v: thus, for universality, it suffices to map linear nearest-neighbor unitary circuits into the cluster-state model.

To represent single-qubit gates, we may compose patterns of the form $\mathfrak{J}^{\alpha}_{w/v}$ as before. However, in order to represent a chain of patterns

$$\mathfrak{J}^{\alpha_{N-1}}_{v_N/v_{N-1}} \cdots \mathfrak{J}^{\alpha_1}_{v_2/v_1}\mathfrak{J}^{\alpha_0}_{v_1/v_0} \,, \tag{2.36}$$

which we may refer to as a *chain pattern*, we require that the qubits v_j for $0 < j < N$ all have degree 2 in the corresponding entanglement graph (as any neighbors aside from v_{j-1} and v_{j+1} represents an operation not in the pattern described above). To embed the corresponding graph state in the grid without additional neighborhood relationships, we must remove every neighbor of these qubits v_j except for those specified by the pattern, which we do using Z measurements. An illustration of such an embedding is illustrated in Figure 2-5. Typically, such chain patterns would be embedded as a horizontal path through the grid, but as Figure 2-5 also shows, we may also employ more general paths in the grid.

Independent single-qubit unitaries acting on independent qubits may then be embedded in the grid as a collection of vertex-disjoint paths, where in particular the distance between two vertices of distinct paths is at least 2 (in order to maintain a buffer of at least one grid site between two chain patterns). The paths so traced out in the grid then correspond to *e.g.* the wires of a unitary circuit diagram. In the construction, we may then represent the logical qubits of the quantum circuit by parallel horizontal paths in the grid whose vertical separation is precisely 2. Note that the paths which in general may be of different lengths, as each horizontal step corresponds to a single-qubit $J(\alpha)$ transformation for some given α. However, we may extend any path in the grid by a

path of length 2 or more in a way which represents the identity operation, using chain patterns of the form

$$\mathfrak{Id}^2_{v_2/v_0} \;=\; \mathfrak{J}^0_{v_2/v_1}\,\mathfrak{J}^0_{v_1/v_0}\,, \qquad\qquad \mathfrak{Id}^3_{v_3/v_0} \;=\; \mathfrak{J}^{\pi/2}_{v_3/v_2}\,\mathfrak{J}^{\pi/2}_{v_2/v_1}\,\mathfrak{J}^{\pi/2}_{v_1/v_0}\,; \tag{2.37}$$

by (2.29), described in stable index notation, these operations are equivalent to unitary circuits $H\big[\frac{v_2}{v_1}\big]\,H\big[\frac{v_1}{v_0}\big]$ and $J(\pi/2)\big[\frac{v_3}{v_2}\big]\,J(\pi/2)\big[\frac{v_2}{v_1}\big]\,J(\pi/2)\big[\frac{v_1}{v_0}\big]$ respectively. The former obviously implements the single-qubit identity operation $\mathbb{1}\big[\frac{v_2}{v_0}\big]$, as H is self-inverse; for the latter, recalling that $J(\pi/2) = HR_{\mathsf{z}}(\pi/2)$, we may characterize its effect by conjugation on linear operators via the stabilizer formalism:

$$X \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; -Y \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; Z \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; X\,, \tag{2.38a}$$

$$Z \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; X \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; -Y \;\xmapsto{HR_{\mathsf{z}}(\pi/2)}\; Z\,; \tag{2.38b}$$

then $J(\pi/2)J(\pi/2)J(\pi/2)$ performs $\mathbb{1}_2$ as well. Thus, both $\mathfrak{Id}^2_{v_2/v_0}$ and $\mathfrak{Id}^3_{v_3/v_0}$ map states ρ_{v_0} to ρ_{v_j} for some $j \geqslant 2$; the first using a path of length 2 in the grid, and the second using a path of length 3. Using combinations of these, we can then implement a path of any length $\ell \geqslant 2$ in the grid, representing a sequence of operations which performs the identity $\mathbb{1}_2$ on a given qubit. Thus, whenever required, we may suppose that the paths in the grid corresponding to any two qubits are of the same length, regardless of the number of (non-trivial) unitary transformations performed on them.

In order to achieve universality for quantum computation, it then suffices to implement a logical $\wedge Z$ operation (or $\wedge X$ operation) in the cluster-state, between two qubits which are at the same horizontal position, and are vertically separated in the grid by a distance of 2 (equivalently, by one grid site). One approach to doing this may be described using a technique described in [25, Section 3.8]. Consider the pattern

$$\mathfrak{Z}\mathfrak{z}_{u,v} \;=\; \mathsf{Z}_v^{\mathsf{s}[a]}\,\mathsf{Z}_u^{\mathsf{s}[a]}\,\mathsf{M}_a^{\pi/2}\,\mathsf{E}_{a,v}\,\mathsf{E}_{a,u}\,\mathsf{N}_a^{\mathsf{x}} \tag{2.39}$$

with input and output subsystems $I = O = \{u, v\}$. The preparation and entangling phases encode the input state into the code stabilized by $\{\mathbb{1}_u \otimes \mathbb{1}_a \otimes \mathbb{1}_v\,, Z_u \otimes X_a \otimes Z_v\}$: the operation $\mathsf{M}_a^{\pi/2}$ then represents a Y measurement which performs an isometry on this code transformation, with the subsequent corrections effectively selecting for the result $\mathsf{s}[a] = 0$. Using the stabilizer formalism, we may then determine

$$\begin{aligned}\langle X_u \otimes \mathbb{1}_v\rangle \;&\xmapsto{\mathsf{E}_{a,v}\,\mathsf{E}_{a,u}\,\mathsf{N}_a^{\mathsf{x}}}\; \left\langle \begin{array}{l} X_u \otimes Z_a \otimes \mathbb{1}_w\,, \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \\ &= \left\langle \begin{array}{l} Y_u \otimes Y_a \otimes Z_w\,, \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \xmapsto{\mathsf{Z}_v^{\mathsf{s}[a]}\,\mathsf{Z}_u^{\mathsf{s}[a]}\,\mathsf{M}_a^{\pi/2}} \langle Y_u \otimes Z_w\rangle\,, \end{aligned} \tag{2.40a}$$

$$\begin{aligned}\langle \mathbb{1}_u \otimes X_v\rangle \;&\xmapsto{\mathsf{E}_{a,v}\,\mathsf{E}_{a,u}\,\mathsf{N}_a^{\mathsf{x}}}\; \left\langle \begin{array}{l} \mathbb{1}_u \otimes Z_a \otimes X_w\,, \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \\ &= \left\langle \begin{array}{l} Z_u \otimes Y_a \otimes Y_w\,, \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \xmapsto{\mathsf{Z}_v^{\mathsf{s}[a]}\,\mathsf{Z}_u^{\mathsf{s}[a]}\,\mathsf{M}_a^{\pi/2}} \langle Z_u \otimes Y_w\rangle\,, \end{aligned} \tag{2.40b}$$

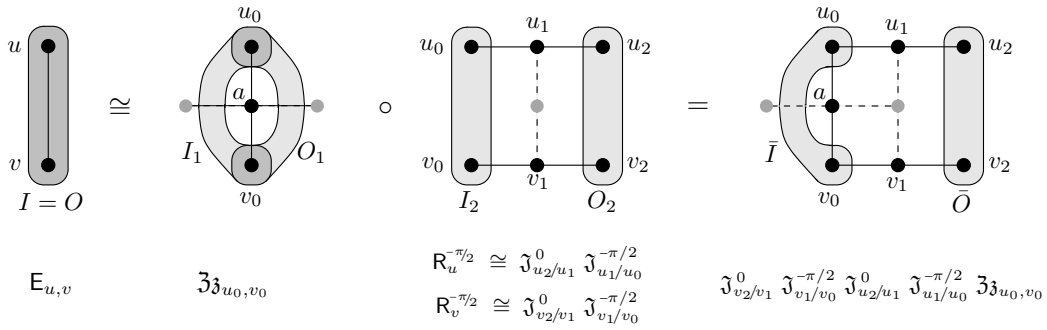


FIGURE 2-6. Illustration of an embedding of $\wedge Z_{u,v}$ in the grid in terms of geometries, using the decomposition of (2.42). Congruency relations represent equivalence up to changes of the input/output system. Grey dots and broken lines indicate qubits and entanglement relations of the grid state which must be removed via Z measurements in order not to interfere with the corresponding measurement pattern (displayed below each geometry).

$$\langle Z_u \otimes \mathbb{1}_v \rangle \xmapsto{\mathsf{E}_{a,v}\mathsf{E}_{a,u}\mathsf{N}^{\mathsf{x}}_a} \left\langle \begin{array}{l} Z_u \otimes \mathbb{1}_a \otimes \mathbb{1}_w \ , \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \xmapsto{\mathsf{Z}^{\mathsf{s}[a]}_v \mathsf{Z}^{\mathsf{s}[a]}_u \mathsf{M}^{\pi/2}_a} \langle Z_u \otimes \mathbb{1}_w \rangle \ , \qquad (2.40c)$$

$$\langle \mathbb{1}_u \otimes Z_v \rangle \xmapsto{\mathsf{E}_{a,v}\mathsf{E}_{a,u}\mathsf{N}^{\mathsf{x}}_a} \left\langle \begin{array}{l} \mathbb{1}_u \otimes \mathbb{1}_a \otimes Z_w \ , \\ Z_v \otimes X_w \otimes Z_w \end{array} \right\rangle \xmapsto{\mathsf{Z}^{\mathsf{s}[a]}_v \mathsf{Z}^{\mathsf{s}[a]}_u \mathsf{M}^{\pi/2}_a} \langle \mathbb{1}_u \otimes Z_w \rangle \ ; \qquad (2.40d)$$

in particular, it is a symmetric operation on u and v which leaves Z_u and Z_v invariant, so it applies a two qubit diagonal operation. It is then easy to verify that the CPTP map which $\mathfrak{Z}\mathfrak{z}$ performs is

$$\mathfrak{Z}\mathfrak{z}_{u,v}(\rho) \ = \ \mathsf{R}^{\pi/2}_u \mathsf{R}^{\pi/2}_v \mathsf{E}_{u,v}(\rho) \ = \ \mathrm{e}^{-i\pi Z_u \otimes Z_v/4} \, \rho \, \mathrm{e}^{i\pi Z_u \otimes Z_v/4} \ ; \qquad (2.41)$$

equivalently, we may substitute any entangler operation $\mathsf{E}_{u,v}$ in the DKP construction with the composite CPTP map

$$\mathsf{E}_{u,v} \ = \ \mathsf{R}^{-\pi/2}_v \, \mathsf{R}^{-\pi/2}_u \, \mathfrak{Z}\mathfrak{z}_{u,v} \ . \qquad (2.42)$$

To obtain an equivalent one-way pattern, we may decompose $\mathsf{R}^{-\pi/2} = \mathsf{J}(0)\mathsf{J}(-\pi/2)$, and then substitute the rotations in (2.42) with chain patterns $\mathfrak{J}^0_{u''/u'} \mathfrak{J}^{-\pi/2}_{u'/u}$ (and similarly for v); this construction is illustrated in Figure 2-6. Alternatively, if the logical entangler operation $\mathsf{E}_{u,v}$ precedes $\mathfrak{J}^{\alpha_u}_{u'/u}$ and $\mathfrak{J}^{\alpha_v}_{v'/v}$ operations in a measurement pattern, we may employ the relations

$$\begin{aligned} \mathfrak{J}^{\alpha_v}_{v'/v} \, \mathfrak{J}^{\alpha_u}_{u'/u} \, \mathsf{E}_{u,v} \ &= \ \mathfrak{J}^0_{v'/v} \, \mathsf{R}^{\alpha_v}_v \, \mathsf{R}^{-\pi/2}_v \, \mathfrak{J}^0_{u'/u} \, \mathsf{R}^{\alpha_u}_u \mathsf{R}^{-\pi/2}_u \, \mathfrak{Z}\mathfrak{z}_{u,v} \\ &= \ \mathfrak{J}^{\alpha_v - \pi/2}_{v'/v} \, \mathfrak{J}^{\alpha_u - \pi/2}_{u'/u} \, \mathfrak{Z}\mathfrak{z}_{u,v} \ , \end{aligned} \qquad (2.43)$$

which follow from (2.26).

A remark on terminology. We have taken some license in referring to the above construction as "the simplified RBB construction", in that some of the constructions above (*e.g.* the construction of the pattern $\mathfrak{Z}\mathfrak{z}$) do not appear explicitly in [113]. However, all of these constructions are clearly implicit in that work. The construction above also completely omits the cleverer constructions of [113], including compact patterns for reversing an entire consecutive block of qubits, non-nearest neighbor $\wedge X$ operations, the quantum Fourier transform over $\mathbb{Z}_{2^k}$ for arbitrary k, and addition circuits for $\mathbb{Z}_{2^k}$.

2.2.7 Universality of the cluster state model

We may use the constructions above, and the pattern transformations described for the DKP constructions in Section 2.2.5, to prove the (approximate) universality of the cluster-state model pattern in standard form, as follows.

By composing the elementary patterns of the simplified RBB construction above, we may construct a one-way pattern in grid graphs $G_{n,d}$ for sufficiently large $n, d \in \mathbb{N}$ which implement arbitrary unitary circuits in a nearest-neighbor version of the Jacz($\mathbb{A}$) model. Restricting to one-way patterns which initialize each input qubit in the $|+\rangle$ state and performing the appropriate trace-out operations, we may then prepare arbitrary density operators by Theorem 1-3. This yields (approximate) universality for the model Cluster($\mathbb{A}$), for any $\mathbb{A} \subseteq \mathbb{R}$ for which Jacz($\mathbb{A}$) is (approximately) universal.

Using the standardization procedures of Section 2.2.5, we may obtain a one-way pattern in which the first stage is to apply an open graph state encoding on the input subsystem: the preparation and entanglement phases of the resulting pattern can then be replaced with the preparation of a sufficiently large cluster state, and removing the qubits which do not play a role in the pattern via Z measurements. Thus, we may prepare arbitrary density operators in the cluster-state model; and by the approximate universality of Jacz($\frac{\pi}{4}\mathbb{Z}$), the cluster-state model is approximately universal for quantum computing for measurement angles which are multiples of $\pi/4$. Furthermore, by performing signal shifting and Pauli simplifications then yield a measurement pattern involving no π-dependencies or shift operations, thus yielding a pattern in the cluster-state model in standard form.

The above then reproduces the following result of [112, 113]:

Theorem 2-2. *The* Cluster($\mathbb{A}$) *model is* (a) *universal for quantum computation when* $\mathbb{A} = \mathbb{R}$*; and* (b) *approximately universal for quantum computation when* $\mathbb{A} = \frac{\pi}{4}\mathbb{Z}$*. Furthermore, we may without loss of generality require that the measurement pattern is standardized, and that Pauli measurements are performed before any non-Pauli measurement operation.*

2.3 Other constructions in the one-way model

Before closing this chapter, we remark upon two other constructions in the one-way measurement model which we will refer to in later chapters, which lie beyond the DKP and the simplified RBB construction schemes described above.

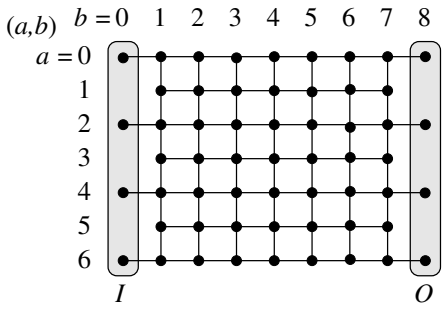


FIGURE 2-7. Geometry for the measurement pattern of [113, Fig. 10] for reversing a sequence of qubits in the cluster-state model. Every qubit (except for the elements of O) is measured with an X measurement, selecting for the state $|+\rangle$: the corresponding correction operations may be determined by the stabilizer formalism. Numbering the rows and columns of grid-sites from 0 at the upper-left, the resulting transformation maps states $\rho_{(0,0),(0,2),(0,4),(0,6)}$ to $\rho_{(8,6),(8,4),(8,2),(8,0)}$, *i.e.* exchanging the logical qubits of rows 0 and 6 as well as rows 2 and 4.

2.3.1 Qubit reversal pattern in the grid.

Figure 2-7 illustrates a geometry for a one-way pattern for reversing a sequence of consecutive logical qubits in a linear nearest neighbor model, using only X measurements, illustrated in [113, Fig. 10]. (This pattern is a special case of a more general transformation, which performs a unitary $e^{i\varphi Z\otimes\cdots\otimes Z}$ transformation in addition to the qubit reversal by measuring a particular qubit in a different basis.) Numbering grid sites (a, b) according to the rows and columns of the grid (starting from 0 at the upper-left), it is possible to show that the resulting transformation is unitary, and in particular, it reverses order of the the "logical" qubits of rows $\{0, 2, 4, 6\}$.

This may be shown using *e.g.* the stabilizer formalism: to do so, however, it is useful to note the presence of correlations in the measurement results of the qubits in the interior block of columns $1-7$. Consider the twelve-qubit set S given by

$$S \;=\; \left\{(a,b) \text{ adjacent to } (d,d) \,\middle|\, d \in \{2,3,4,5,6\}\right\} \;: \tag{2.44}$$

it is easy to verify that the product of $K_{(a,b)} = X_{(a,b)}Z_{(a-1,b)}Z_{(a+1,b)}Z_{(a,b-1)}Z_{(a,b+1)}$ for $(a, b) \in S$ is a tensor product of $X_{(a,b)}$ operators, also for (a, b) ranging over S. If we measure the qubits in this set column-by-column with X observable measurements, this implies that the unique qubit in S in the seventh column will yield the ±1 result with certainty if the product of the preceding eleven measurements is ±1. We may similarly show that every qubit in the seventh column (except for $(0, 7)$ and $(6, 7)$) will yield measurement results correlated in a similar way with a subset of qubits in the preceding six columns; the measurements on those qubits then perform the identity, while the other measurements will anticommute with some generators of the stabilizer group $\langle K_v\rangle_{v\notin O}$.

Note that these correlations arise from cancelling out Z operators in the interior block; by a similar technique of cancelling such Z operators, it is possible to show that X and Z observables on each of the input qubits get mapped by measurements (and selecting for the $|+\rangle$ measurement result) to the same observable on the "reversed" output qubit. By scaling this measurement pattern in the obvious way, we may obtain a qubit-reversal pattern for an arbitrary number of qubits.

This measurement pattern is noteworthy in that it falls outside of the domain of automated approaches to inferring the semantics of a measurement-based pattern (one of which we describe in Chapter 3), precisely because of the correlations which produce measurement results with certainty. In the special case of swapping two consecutive qubits, it is also more compact than the construction of $\mathsf{SWAP}_{u,v}$ described in (2.35) using the construction for $\mathsf{E}_{u,v}$ in the simplified RBB construction above.

2.3.2 Concise patterns for $Z \otimes \cdots \otimes Z$ Hamiltonians.

The construction of the $\mathfrak{Zz}$ pattern in (2.39) is a special case of a more general construction in [25] in the more general one-way model for applying unitaries arising from $Z^{\otimes k}$ Hamiltonians for k arbitrarily large. We may define the states $|\pm \mathsf{yz}_\theta\rangle$ as follows:

$$|\pm \mathsf{yz}_\theta\rangle \;=\; \frac{1}{\sqrt{2}}\,|+\mathsf{x}\rangle \;\pm\; \frac{\mathrm{e}^{-i\theta}}{\sqrt{2}}\,|-\mathsf{x}\rangle \;; \tag{2.45}$$

it is possible to verify that these are ± 1 eigenvectors of the observable $\cos(\theta)Z + \sin(\theta)Y$. Define the measurement operator

$$\mathsf{M}_a^{(\mathsf{YZ},\theta)}(\rho) \;=\; \langle +\mathsf{yz}_\theta|_a\,\rho\,|+\mathsf{yz}_\theta\rangle_a \otimes |0\rangle\langle 0|_{\mathsf{s}[a]} \;+\; \langle -\mathsf{yz}_\theta|_a\,\rho\,|-\mathsf{yz}_\theta\rangle_a \otimes |1\rangle\langle 1|_{\mathsf{s}[a]} \tag{2.46}$$

which performs a YZ-plane measurement similar to the XY-plane measurements defined by (2.5). Then, the measurement pattern

$$\underbrace{\mathfrak{Zz}\cdots\mathfrak{z}}^{\theta}_{v_1,\cdots,v_k} \;=\; \left[\prod_{j=1}^{k} \mathsf{Z}_{u_j}^{\mathsf{s}[a]}\right] \mathsf{M}_a^{(\mathsf{YZ},\theta)} \left[\prod_{j=1}^{k} \mathsf{E}_{a,u_j}\right] \mathsf{N}_a^{\mathsf{x}} \tag{2.47a}$$

k times

performs a unitary rotation

$$\underbrace{\mathfrak{Zz}\cdots\mathfrak{z}}^{\theta}_{v_1,\cdots,v_k}(\rho) \;=\; \mathrm{e}^{-i\theta Z_{v_1}\otimes\cdots\otimes Z_{v_k}/2}\,\rho\,\mathrm{e}^{i\theta Z_{v_1}\otimes\cdots\otimes Z_{v_k}/2}\,. \tag{2.47b}$$

k times

The operation $\mathfrak{Zz}_{u,v}$ is a special case for $k = 2$ and $\theta = \pi/2$; the geometry for the more general measurement pattern is illustrated in Figure 2-8.

It is easy to verify the behavior of $\mathfrak{Zz}\cdots\mathfrak{z}$ on standard basis state vectors, as follows. Using the fact that $\wedge Z\big(|+\rangle \otimes |0\rangle\big) = |+\rangle \otimes |0\rangle$ and $\wedge Z\big(|+\rangle \otimes |1\rangle\big) = |-\rangle \otimes |1\rangle$, the state vector produced by the preparation and entanglement procedure of $\mathfrak{Zz}\cdots\mathfrak{z}^{\theta}_{v_1,\ldots,v_k}$ for an input state $|\mathbf{x}\rangle_{v_1,\ldots,v_k}$ (for $\mathbf{x} \in \{0,1\}^k$ arbitrary) is

$$\begin{aligned}
|\psi_{\mathbf{x}}\rangle \;&=\; \left[\prod_{j=1}^{k} \wedge Z_{a,u_j}\right] |+\rangle_a \otimes |\mathbf{x}\rangle_{v_1,\ldots,v_k} \\
&=\; \left\{ \begin{array}{ll} |+\rangle_a \otimes |\mathbf{x}\rangle_{v_1,\ldots,v_k}\,, & \text{if } x_1 \oplus \cdots \oplus x_k \equiv 0 \pmod 2 \\ |-\rangle_a \otimes |\mathbf{x}\rangle_{v_1,\ldots,v_k}\,, & \text{if } x_1 \oplus \cdots \oplus x_k \equiv 1 \pmod 2 \end{array} \right\} \\
&=\; |\pm\rangle_a \otimes |\mathbf{x}\rangle_{v_1,\ldots,v_k} \quad \text{for } |\mathbf{x}\rangle \text{ a } \pm 1\text{-eigenvector of } Z^{\otimes k}.
\end{aligned} \tag{2.48}$$

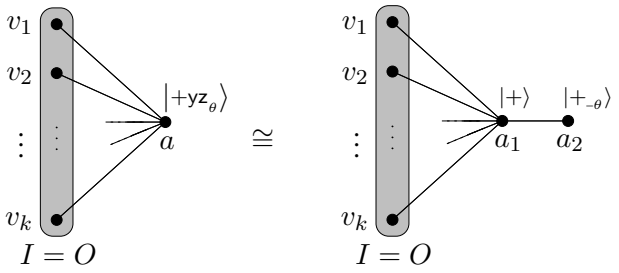


FIGURE 2-8. Illustration of the geometry for the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ applied to k qubits, $v_1, \ldots, v_k$. In the left-most figure, the auxiliary qubit is to be measured in the basis $|\pm\mathsf{yz}_\theta\rangle$, with corrections to be performed to effectively select the positive result. The right-most figure illustrates an equivalent pattern using only XY-plane measurements, in which the two auxiliary qubits are to be measured, performing corrections if necessary to select for the measurement results given.

Note that $\langle+\mathsf{yz}_{2\theta}|+\rangle = \frac{1}{\sqrt{2}}$ and $\langle+\mathsf{yz}_\theta|-\rangle = \frac{\mathrm{e}^{i\theta}}{\sqrt{2}}$; then, if we apply the projection $|+\mathsf{yz}_\theta\rangle\langle+\mathsf{yz}_\theta|$ to $|\psi\rangle$ as described above and renormalize, we obtain

$$\begin{aligned}
|+\mathsf{yz}_{2\theta}\rangle\langle+\mathsf{yz}_\theta|_a\,|\psi\rangle_{a,v_1,\ldots,v_k} &= \left\{ \begin{array}{ll} \frac{1}{\sqrt{2}}\,|\mathbf{x}\rangle_{v_1,\ldots,v_k}\,, & \text{if } |\mathbf{x}\rangle \text{ a } +1\text{-eigenvector of } Z^{\otimes k} \\ \frac{\mathrm{e}^{i\theta}}{\sqrt{2}}\,|\mathbf{x}\rangle_{v_1,\ldots,v_k}\,, & \text{if } |\mathbf{x}\rangle \text{ a } -1\text{-eigenvector of } Z^{\otimes k} \end{array} \right\} \\
&\propto \left\{ \mathrm{e}^{\mp i\theta/2}\,|\mathbf{x}\rangle_{v_1,\ldots,v_k}\,, \quad \text{for } |\mathbf{x}\rangle \text{ a } \pm 1\text{-eigenvector of } Z^{\otimes k} \right\} \\
&= \left[\mathrm{e}^{-i\theta Z^{\otimes k}/2}\,|\mathbf{x}\rangle\right]_{v_1,\ldots,v_k}\,; \qquad (2.49)
\end{aligned}$$

a similar analysis for the projection $|+\mathsf{yz}_\theta\rangle\langle+\mathsf{yz}_\theta|$ yields the result

$$|-\mathsf{yz}_\theta\rangle\langle-\mathsf{yz}_\theta|_a\,|\psi\rangle_{a,v_1,\ldots,v_k} \;\propto\; \left[\mathrm{e}^{-i\theta Z^{\otimes k}/2}\,Z^{\otimes k}\,|\mathbf{x}\rangle\right]_{v_1,\ldots,v_k}\,, \qquad (2.50)$$

so that the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}_{v_1,\cdots,v_k}$ performs the rotation described, by linearity.

In order to perform such a measurement-based procedure in the one-way model as described above, we must represent the measurement $\mathsf{M}_a^{(\mathsf{YZ},\theta)}$ in terms of XY-plane measurements. We may simulate a YZ-plane measurement using XY-plane measurements by noting that

$$\mathsf{H}\big(\cos(\theta)Z + \sin(\theta)Y\big) \;=\; \cos(-\theta)X + \sin(-\theta)Y\,; \qquad (2.51)$$

that is, a Hadamard transformation effects a change of reference frame, transforming YZ-plane measurement observables to some corresponding XY-plane measurement observables. Then, we have the following congruency,

$$\mathsf{M}_a^{(\mathsf{YZ},\theta)} \;\cong\; \mathsf{M}^{(\mathsf{YZ},\theta)}_{\mathbf{s}[a_2]/a_1} \;=\; \mathsf{M}^{-\theta}_{a_2}\,\mathsf{X}^{\mathbf{s}[a_1]}_{a_2}\,\mathsf{M}^{0}_{a_1}\,\mathsf{E}_{a_1,a_2}\,\mathsf{N}^{\mathsf{x}}_{a_2}\,, \qquad (2.52)$$

where congruency is up to changes in the input and output subsystems; the measurement on a_1 performs a Hadamard transform to effect the necessary change of reference frame

to perform the YZ-plane measurement via XY-plane measurements. Absorbing the correction on a_2 into the measurement, we may then transform $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ into a pattern in the one-way model as described in this section, by the congruence

$$\underbrace{\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}}_{k \text{ times}}{}^{\theta}_{v_1,\cdots,v_k} \quad \cong \quad \left[\prod_{j=1}^{k} \mathsf{Z}^{\mathsf{s}[a_2]}_{u_j}\right] \mathsf{M}^{-\theta;\mathsf{s}[a_1]}_{a_2} \mathsf{M}^{0}_{a_1} \left[\prod_{j=1}^{k} \mathsf{E}_{a_1,u_j}\right] \mathsf{E}_{a_1,a_2} \mathsf{N}^{\mathsf{x}}_{a_2} \mathsf{N}^{\mathsf{x}}_{a_1} \ . \tag{2.53}$$

The geometry for this measurement pattern is also illustrated in Figure 2-8. Nevertheless, the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ described in (2.47a) illustrates the potential usefulness of extending beyond XY-plane measurements in the one-way model; and we will also refer to this construction in later Chapters.

2.4 Conclusion

In this section, we have described the one-way measurement model in detail, by way of defining constructions for measurement procedures which simulate unitary circuit models; and we have also given an overview of approaches to robustly implementing quantum computers via the one-way model.

Simulation of unitary circuits is the standard approach to obtaining one-way measurement procedures to perform quantum computation: however, it is possible to construct measurement patterns which do not clearly correspond to the simulation of a unitary circuit. An example of one technique for doing so are the simplification techniques of [**?**], which allows the transformation of any one-way measurement procedure involving measurements of Pauli observables into an equivalent procedure without Pauli observable measurements, on fewer qubits. The way in which this is done takes advantage of a correspondence between *local Clifford operations* (*i.e.* products of single-qubit unitaries from the Clifford group) on graph state, and *local complementation* of the graphs which describe the graph states: we describe this correspondence in Lemma 4-6. This gives rise to transformations of graph states, and subsequently of measurement-based procedures, where the resulting procedures no longer correspond to a simulation of a unitary circuit in any clear way (see for example [74, Figure 16]).

This raises the question of the conditions under which we may identify when a measurement pattern may be meaningfully considered to simulate some unitary circuit: this problem is the one which we consider in Chapter 3.

Chapter 3

Semantics

for unitary one-way patterns

One-way measurement based quantum computation describes unitary transformations of quantum states as a composition of CPTP maps, many of which are not themselves unitary. The fact that the resulting transformation is unitary is due to an appropriate combination of measurement observables and entanglement operations which, as in the analysis of the pattern $\mathfrak{J}^\alpha$ in (2.27), may be treated with the stabilizer formalism (Section 1.5.3). The post-measurement residual state in either case will be an isometric image of the pre-measurement state; and the possible post-measurement states corresponding to the different measurement results may be mapped to one another by single-qubit unitary transformations. As a result, we may perform a correction after each measurement to simulate postselection of the $+1$ measurement result, yielding a mixed one-way measurement pattern; equivalently, we may absorb the byproduct operations into future measurements, interpreting them as changes of reference frame, in order to obtain a one-way measurement based computation in standard form.

As we are interested in unitary transformations, we are presented with natural decision problems when considering quantum computation in the one-way measurement model. Measurements do not transform quantum states unitarily — is it possible to tell whether a one-way measurement based computation performs a unitary transformation between its input and output subsystems? Is it possible to describe precisely *how* such computations transform quantum states, by translation to *e.g.* a quantum circuit of comparable complexity, using a reasonable set of elementary gates? If not, is it at least possible to efficiently verify whether it performs some given transformation?

The problem of determining when a measurement problem performs an isometry seems similar to to the more general question of when a classically controlled circuit in general performs an isometry. By [101], this may be rephrased as asking whether each measurement performed has the same bias towards one measurement result or another, across all possible input states. This problem is NP-hard if we require exponential precision, as we can encode the evaluation of arbitrary boolean functions $f : \{0,1\}^n \longrightarrow \{0,1\}$ to produce the state

$$|\mathrm{SAT}_f\rangle \;=\; \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(x)} |x\rangle \tag{3.1}$$

and perform measurements in the $|\pm\rangle$ basis. If any such measurement has non-zero probability of yielding the $|-\rangle$ state, this would indicate that f is satisfiable but not a tautology. However, it is easy to verify that the natural approach to producing a state such as $|\text{SAT}_f\rangle$ requires the use of non-Clifford group operations, whereas the open graphical encoding procedures described in (2.21) consist of Clifford group operations, and prepare states in a known stabilizer code. As well, we require that the post-measurement residual states be related to each other by local unitary operations in order to be able to adaptively perform the (single-qubit) measurements of a one-way procedure. It may be hoped that the additional structure imposed by these constraints yields a tractable problem.

These questions arise because simple unitary circuit models such as PHASES($\mathbb{A}$) defined in Section 1.3 are the *de facto* standard models of quantum computation: they seem to provide the most natural set of idioms for uniformly expressing algorithms independently of architecture. (We will commonly refer "the" unitary circuit model, by which we will mean the PHASES($\mathbb{A}$) or JACZ($\mathbb{A}$) models for some $\mathbb{A} \subseteq \mathbb{R}$.) While alternative models are promoted as abstractions of proposals for physically implementing a quantum computer (*e.g.* measurement-based models [70, 112] and the adiabatic model [59]), and "purely physical" idioms seemingly divorced from uniform circuit families have led to new quantum algorithms (as with the NAND tree algorithm of [58], inspired by transmission and reflection rates associated with energy barriers), the unitary circuit model provides what seems to be the simplest mathematical model for describing quantum computation. Because of the present uncertainty as to *which* implementation proposal will prove successful, the unitary circuit model represents a natural candidate for an *intermediate language*[1] for representing algorithms independently of the target architecture, even if alternative models suggest new algorithms.

Therefore, I argue that how effectively one can translate decompositions of unitaries to and from the unitary circuit model is currently an important topic for any proposed alternative model of quantum computation. Unless some one proposal for implementation comes to dominate over the others, or an unlikely breakthrough (*e.g.* the discovery of a proof that $\mathsf{BQP} = \mathsf{BPP}$) is made, translation to a "simple" model of quantum computation such as the unitary circuit model is the best that can be reasonably hoped for as a *uniform* way of obtaining low-level semantics of a quantum computation. In other words: the most reasonable means of understanding the behavior of a quantum algorithm is currently by reduction to the unitary circuit model. This motivates the problem of effective translations from those one-way pattern which perform unitary embeddings, into the unitary circuit model.

The topic of this chapter is one such reduction, from those one-way measurement patterns which arise out of the DKP constructions to unitary circuits. We consider this topic by examining a *flow* property which emerges from that construction, which gives rise to efficiently detectable combinatorial structures and combinatorial decompositions of the corresponding measurement pattern.

Previous appearances of this work. Earlier versions of many of the results of this chapter appear in [44, 45, 47]. The work presented in Section 3.3 (except for 3.3.5) and

[1] *Intermediate language* refers here to a language used to facilitate the design of a suite of compilers, which translate programs from one or many source language (or from many source languages) to one of several different target architectures.

Section 3.4 first appeared in a preliminary form in [44]; the results of the former was published as [45] in essentially the form presented here. The results of Section 3.3.5 are joint work with Martin Pei, and were published in [47]. The rest of the work in this Chapter, except where indicated, are developments original to this thesis.

Graph theoretic notations. We will assume basic familiarity with graph theory: an introduction to the subject and basic definitions can be found in [54]. Whenever a graph G is clear from context, $\sim$ will denote the adjacency relation of G. In graphs, we will denote an edge between v and w by vw, and in directed graphs, we will let $v \to w$ represent an arc between v and w; and in graphs or directed graphs, we will represent (directed) paths by concatenated sequences of edges/arcs, $v_1 v_2 \cdots v_n$ or $v_1 \to v_2 \to \cdots \to v_n$. We adopt the convention that graphs do not have self-loops on vertices.

3.1 Defining semantics by reduction to unitary circuits

Before proceeding, we will put forward a precise definition of how the semantics of a measurement-based computation may be reduced to another model, such as the unitary circuit model.

We are generally not interested in *arbitrary* translations to and from the unitary circuit models: in this instance, the particular circuit model becomes significant. For instance, in the case of the one-way measurement model, we may easily obtain a circuit in the Phases($\mathbb{A}$) or Jacz($\mathbb{A}$) models by describing the measurement pattern as a circuit in a classically controlled unitary circuit model, and applying the principle of deferred measurement (page 41) to obtain a unitary circuit in some unitary circuit model CoherCtrl which includes (coherently) controlled unitary transformations performing conditional change of basis operations to simulate adaptive measurements. Such a circuit can then be decomposed in *e.g.* the Jacz($\mathbb{A}$) model by decomposing the individual transformations of the model CoherCtrl. However, such a direct translation scheme may require many more qubits of workspace, and many more multi-qubit operations, than is actually necessary. For instance, if we translate a circuit of the form $J(\alpha_n) \cdots J(\alpha_2) J(\alpha_1)$ on a single qubit into the one-way model using the DKP construction, and then obtain a unitary circuit from that pattern via the principle of deferred measurement as above, we will obtain a circuit involving $n+1$ qubits and $O(n)$ two-qubit operations, whereas the original circuit required only operations on a single qubit.

In contrast with the above, we may hope to obtain a translation scheme to unitary circuit models which in some sense preserves the complexity of the computation. In order to make this sense of complexity-conservation well defined despite the differences in how resources are used in different models of computation, we may make reference (as we have done above for the DKP construction) to another function which translates from unitary circuits to the computational model of interest. In particular, suppose we have a map $\mathcal{R}$ which maps unitary circuits in some gate model Gates to some target model M of quantum computation. We would like to consider a mapping $\mathcal{S}$ such that

(*i*) $\mathcal{S}$ is a map from a superset of $\mathrm{img}(\mathcal{R})$ to unitary circuits using the gate-set Gates (where in particular, $\mathrm{dom}(\mathcal{S})$ includes any procedures in M which may be in some sense "congruent" to the image of a circuit in $\mathcal{R}$);

(*ii*) $\mathcal{S} \circ \mathcal{R}$ maps each circuit in $\mathrm{dom}(\mathcal{R})$ to an equivalent circuit of lesser or equal gate complexity;

(*iii*) $\mathcal{R} \circ \mathcal{S}$ maps procedure in $\mathrm{img}(\mathcal{R})$ to a "congruent" procedure in $\mathcal{R}$.

We will not formally define here when two computational procedures are "congruent" in an arbitrary model of computation: however, for both the unitary circuit model and for the one-way measurement based model, we will adopt the convention of Section 1.3 that computations are congruent in either model if they only differ by transpositions of commuting operations (and by a possible relabelling of the qubits).

It may seem unproductive to consider a map $\mathcal{S}$ from a model M as above to unitary circuits, which is explicitly defined with reference to a *existing* translation procedure $\mathcal{R}$ from unitary circuits to M, as the most obvious source of a computation in $\mathrm{img}(\mathcal{R})$ is to take an existing unitary circuit C and to produce $\mathcal{R}(C)$. However, given the current prominence of the unitary circuit model, I would argue that a mature understanding of a model M of quantum computation currently requires the ability to perform meaningful translations *in both directions* between M and the unitary circuit model. Should new insights arise for performing algorithms in the model M, these insights can then also be translated back to the unitary circuit model.

In practise we will be largely interested in mappings between congruency classes of unitary circuits and of other models of computation. In the case where $\mathcal{R}$ is an injective map from a unitary circuit model to some model M, we may simply wish for $\mathcal{S}$ to be an inverse for the action of $\mathcal{R}$ on congruency classes of unitary circuits. However, for more general transformations $\mathcal{R}$ which may map incongruent circuits (but which perform equivalent CPTP maps) to congruent procedures in some other model of computation, the conditions described above may be of interest in the absence of a left inverse for $\mathcal{R}$.

In this thesis, we are primarily interested in the one-way measurement model, so we consider the case of $\mathrm{M} = \textsc{OneWay}(\mathbb{A})$ for some $\mathbb{A} \subseteq \mathbb{R}$. A function $\mathcal{R}$ as above provides an image of unitary circuit "idioms" — structural formulas for circuits, which are used to perform certain tasks — in the one-way model: a one-way measurement based computation which is in the image of such a function $\mathcal{R}$ may be said to be a representation of some unitary circuit in the one-way model. We may therefore call $\mathcal{R}$ a *representation* map, and the one-way measurement based routines in $\mathrm{img}(\mathcal{R})$ *representations* of circuits.

The map $\mathcal{S}$ is then a means of recognizing one-way measurement patterns which simulate circuits in this sense, revealing semantics for these one-way measurement patterns in terms of unitary circuits, in a way which (a) preserves the structure of some representative of each class of circuits which are translated by $\mathcal{R}$ into a given one-way measurement based procedure, and (b) such that the one-way measurement based algorithm can be recovered from the circuit through the "representation" map. We may call a map $\mathcal{S}$ of this sort a *semantic* map, and say that a circuit in $\mathrm{img}(\mathcal{S})$ provides the *semantics* of a one-way pattern in terms of some unitary circuit model.

It is not difficult to imagine more properties which we would like the map $\mathcal{S}$ to have beyond the ones described above, for instance:

- It is obviously desirable for $\mathcal{S}$, and the membership predicate for $\mathrm{dom}(\mathcal{S})$, to be efficiently computable.

- It would be convenient for a semantic map $\mathcal{S}$ to also be able to recognize *parts* of one-way measurement based computations which resemble "idioms" from the circuit model, and be able to translate them into components of the corresponding circuit. This would allow us to describe the semantics of a one-way measurement pattern by reduction to smaller pieces.

- We might also like $\mathcal{S}$ to be well defined on the set of *all* one-way measurement based computations performing unitary transformations, which may be a proper superset of the range of $\mathcal{R}$: this may suggest which extensions to the unitary circuit model can be easily translated into a one-way measurement pattern, and which may therefore be useful tools to adopt for designing quantum algorithms.

We are of course primarily interested in efficient algorithms, and will restrict our attention without further comment to semantic maps which can be performed in polynomial time in the size of the specification for a one-way measurement based computation. The second of the extensions above, while beyond the scope of this thesis to formalize, is easy to achieve in an intuitive sense in some instances which will be described in this chapter. The third of these extensions is more difficult, in part because it is not yet known how to determine whether a one-way measurement based algorithm realizes a unitary transformation or not: for any such "extended semantics" map $\mathcal{S}$, a polynomial-time algorithm for the membership predicate of $\mathrm{dom}(\mathcal{S})$ is not yet known.

We have defined semantic maps $\mathcal{S}$ above relative to a map $\mathcal{R}$ for representing unitary circuits in the one-way measurement based model, and it is conceivable that for some interesting maps $\mathcal{R}$, it may be difficult or impossible to find a map $\mathcal{S}$ which satisfies all of the properties (*i*) – (*iii*) on pages 80 – 81. In this case, we may be satisfied by an $\mathcal{S}$ which *approximately* achieves these properties for a given $\mathcal{R}$. For instance, we may require that for a unitary circuit c, the number of single-qubit and two-qubit gates in the circuit $(\mathcal{S} \circ \mathcal{R})(C)$ are increased beyond those of C only by small scalar factors (*e.g.* less than $1 + \varepsilon$ for a suitably small ε), or that even if $\mathcal{R}$ is not a left-inverse of $\mathcal{S}$, that there is an efficiently computable reduction map f on $\mathrm{dom}(\mathcal{S})$ such that $f \circ \mathcal{R} \circ \mathcal{S}$ is the identity on $\mathrm{dom}(\mathcal{R})$. However, the less closely we adhere to the conditions that $\mathcal{R} \circ \mathcal{S}$ is equivalent to the identity and that $\mathcal{S} \circ \mathcal{R}$ does not inflate the complexity of unitary circuits, the less useful the mapping $\mathcal{S}$ is as a means of providing semantics for a one-way measurement based computation in terms of the unitary circuit model.

The main result of this chapter is to describe a mapping $\mathcal{S}$ which acts as a semantic map with respect to the DKP construction described in Section 2.2, where we further restrict to those patterns produced from circuits without terminal measurements or trace-out operations. This allows us to identify a class of one-way patterns which perform unitary transformations, and provide semantics for them in terms of unitary circuits. The semantic map $\mathcal{S}$ in this case is defined in terms of a *flow* property of the geometry underlying the one-way pattern, which will be the main subject of this chapter. Towards the end of the Chapter, we sketch similar results for a semantic map for the simplified RBB construction by using a slight extension of flows, building on the analysis for flows and on improved flow-finding algorithms to do so.

A remark on terminal measurement and trace-out operations

As we have described it in Section 2.2, the model OneWay($\mathbb{A}$) includes trace-out operations. When considering how to transform a pattern $\mathfrak{P}$ in the OneWay($\mathbb{A}$) model to a unitary circuit model, given that the pattern is congruent to a pattern in the image of the DKP construction, we may augment the output subsystem O. This results in a measurement pattern $\mathfrak{P}'$ which also performs a unitary embedding, and is also congruent to a pattern from the DKP construction; the unitary circuit C corresponding to the original pattern $\mathfrak{P}$ may be obtained by adding trace-out operations to the circuit C' corresponding to $\mathfrak{P}'$. However, patterns produced by the DKP construction from circuits with terminal measurements pose a more difficult problem, as it seems difficult to distinguish between qubits which are measured in order to drive the transformation of data and qubits which are measured in order to yield a classical probabilistic output of interest (speaking of the measurements in their roles of representing operations of the unitary circuit).

Given that we are primarily interested in how unitary transformations may be described by measurement-based computation, we set aside the issue of how to provide semantics for arbitrary patterns which may be produced by the DKP construction, and focus on those which arise from circuits which do not perform measurements or trace-out operations. Fortunately, by the comments made above, semantics for patterns in $\mathrm{img}(\mathcal{R})$ which perform trace-outs may still be efficiently recovered; but for patterns arising from circuits with measurements, it seems that more general tools will be required. We will pass over this subject for the remainder of the thesis.

3.2 Flows: structure underlying the DKP construction

As we defined it in Definition 2-7, the simplified DKP construction produces a measurement pattern from a circuit in the Jacz($\mathbb{A}$) model (for some $\mathbb{A} \subseteq \mathbb{R}$) by transforming each gate $J(\alpha)$ and $\wedge Z$ into a simple measurement pattern via the mapping Φ defined in (2.29), composing those patterns in the appropriate manner. The resulting measurement dependencies exhibits a simple structure which may be described in terms of local properties of the entanglement graph, as follows.

For a unitary circuit C in the Jacz($\mathbb{A}$) model, consider the last gate $J(\alpha)$ performed. (If there is more than one such gate which may be performed in parallel, we may choose an arbitrary one.) We may decompose C into circuits $C_v C'$, where C_v consists of the final $J(\alpha)$ gate on v, and any controlled-Z operations on v which follow it, and C' consists of the rest of the operations of C. We can recursively apply this procedure to decompose C into layers of $\wedge Z_{u,v}$ gates separated by $J(\alpha)_v$ gates for various qubits u and v, where the $\wedge Z$ gates of each layer following a $J(\alpha)_v$ gate all act on the qubit v. We will refer to this as a *star decomposition* of a Jacz($\mathbb{A}$) circuit. Consider the patterns produced by the DKP construction, for each sub-circuit C_v consistsing of a $J(\alpha)_v$ gate and the subsequent

layer of $\wedge Z$ gates: writing C_v in stable-index tensor notation (pages 28 – 38), we have

$$\begin{aligned}
\Phi(C_v) &= \Phi\left(\left(\prod_u \wedge Z_{[u,v']}\right) J(\alpha)\left[\tfrac{v'}{v}\right]\right) \\
&= \left(\prod_u \mathsf{E}_{u,v'}\right) \mathfrak{J}^{\alpha}_{v'/v} \\
&= \left(\prod_u \mathsf{E}_{u,v'}\right) \mathsf{X}^{\mathsf{s}[v]}_{v'} \mathsf{M}^{-\alpha}_{v} \mathsf{E}_{v,v'} \mathsf{N}^{\mathsf{x}}_{v'} \ , \\
&\cong \left(\prod_u \mathsf{Z}^{\mathsf{s}[v]}_{u}\right) \mathsf{X}^{\mathsf{s}[v]}_{v'} \left(\prod_u \mathsf{E}_{u,v'}\right) \mathsf{M}^{-\alpha}_{v} \mathsf{E}_{v,v'} \mathsf{N}^{\mathsf{x}}_{v'} \ . \qquad (3.2)
\end{aligned}$$

where in the last step we commute the correction operation $\mathsf{X}^{\mathsf{s}[v]}_{v'}$ to the right. The $J(\alpha)$ gate in the original circuit C gives rise to a qubit v' which is a "successor" of the qubit v, in the sense that the pattern $\mathfrak{J}^{\alpha}_{v'/v}$ transfers the state of v to v' (up to the given unitary transformation); and every other neighbor of v' in the entanglement graph will be subject to a correction operation which depends on the result $\mathsf{s}[v]$ of the measurement on v.

If we compose measurement patterns of this form together, this structure of correction is (almost) preserved: because the $\mathsf{Z}^{\mathsf{s}[v]}_{u}$ operations commute with any entangling operations acting on the qubits u, the only additional correction which can be induced is a correction $\mathsf{Z}^{\mathsf{s}[v]}_{v''}$ on a possible successor of v' in cases where the same qubit is later acted on by another $J(\alpha')$ gate in C. Absorbing these correction into measurements where appropriate, these give rise to sign- and π-dependencies on v of the measurements on qubits adjacent to v' in the entanglement graph (other than v itself).

In the case of qubits measured with X measurements (*i.e.* whose measurement angle is zero), the "effect" of X corrections are trivial; in every other case, however, there is either a non-trivial effect on the basis of measurement, or (speaking somewhat counterfactually) on the result obtained upon measurement in the case of a π-dependency. In particular, while π-dependencies could be removed by performing signal-shifting, *i.e.* by introducing an operator which conditionally toggles the measurement result, we could also speak informally of the "measurement result that would occur" being affected by the change in the measurement by the addition of π to the measurement angle.[2] Thus, we interpret the angle adaptations given by the simplified DKP construction as potential "influences" on the measurement outcomes, neglecting the imperviousness of X measurements to sign dependencies for the sake of uniformity.

Signal shifting and Pauli measurement simplifications can be applied to reduce the logical depth of a measurement-based computation, and so produce a pattern of the *complete* DKP construction. However, the dependencies described by the *simplified* DKP construction are local in nature, and can be described simply by the geometry (G, I, O) of the measurement pattern (Definition 2-5), together with a function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ mapping each qubit v in the pattern to its "successor" (where $O^{\mathsf{c}} = V(G) \setminus O$, and

[2] This intuition of "counterfactual" change of measurement results could perhaps be given an ontological foundation in terms of hidden variables, thereby removing its counterfactual nature; a formal discussion of this is beyond the scope of this thesis.

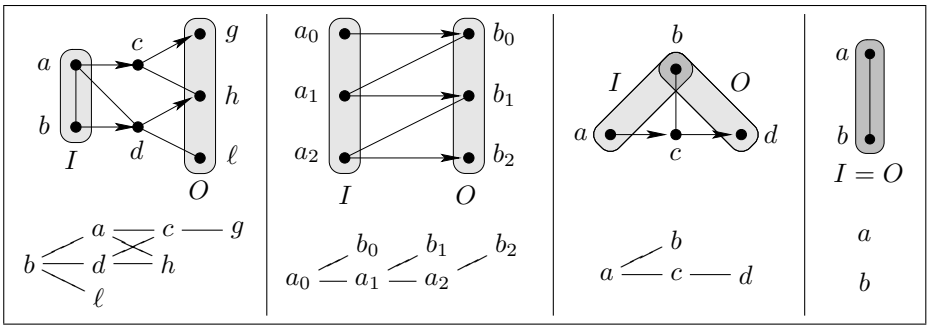


FIGURE 3-1. Examples of geometries with flows. Arrows indicate the action of a function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ along otherwise undirected edges. Corresponding partial orders $\preccurlyeq$ for each example are given by Hasse diagrams below the graphs (with minimal elements on the left and maximal elements on the right). In the right-most example, the two vertices a and b are incomparable, *i.e.* there is no order relation between them.

$I^{\mathsf{c}} = V(G) \setminus I$), arising from the $\mathfrak{J}^{\alpha}$ patterns. In particular, the measurement of each qubit $v \in O^{\mathsf{c}}$ influences the measurement of its successor $f(v)$ by a sign dependency, and the measurements of the other neighbors of $f(v)$ by a π-dependency. Furthermore, these dependencies entail that both $f(v)$ and every qubit adjacent to $f(v)$ — except for v itself — would be measured strictly after v in the measurement order arising from the simplified DKP construction.

"Flows" were defined by Danos and Kashefi [38] in an approximate[3] early attempt to characterize those one-way measurement patterns which perform unitary transformations, based on this structure of dependencies arising from this construction. Without reference to a pattern known to be obtained from the DKP construction, a flow is defined as follows:

Definition 3-1. For a geometry (G, I, O), a *flow* is an ordered pair $(f, \preccurlyeq)$ consisting of a function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, and a partial order[4] $\preccurlyeq$ on $V(G)$, such that the conditions

$$v \sim f(v)\,, \tag{3.3a}$$

$$v \preccurlyeq f(v)\,, \quad \text{and} \tag{3.3b}$$

$$w \sim f(v) \implies v \preccurlyeq w \tag{3.3c}$$

hold for all $v \in O^{\mathsf{c}}$ and $w \in V(G)$.

Examples of geometries with and without flows are illustrated in Figures 3-1 and 3-2.

The function f and the partial order $\preccurlyeq$ capture the essential structure of the dependencies in the simplified DKP construction: f represents the mapping of qubits to their

[3] The presentations of the one-way model in [112, 113] already contained several constructions which do not have "flows", which were formulated later. However, at least in the case of the the simplified RBB construction presented in the previous chapter, there is a simple correspondence to the DKP construction which allows these to be related to geometries with flows. It was thought that a modest extension of the definition of flows would also capture those one-way measurement based algorithms [36].

[4] A partial order $\preccurlyeq$ is a pre-order (see footnote (14) on page 38) which is anti-symmetric: that is, for which $x \preccurlyeq y$ and $y \preccurlyeq x$ imply $x = y$. Examples include the divisibility relation $x \,|\, y \iff (y \div x \in \mathbb{N})$ for complex numbers x and y, and the subset relation $X \subseteq Y \iff \forall z : (z \in X \Rightarrow z \in Y)$ on sets X and Y.

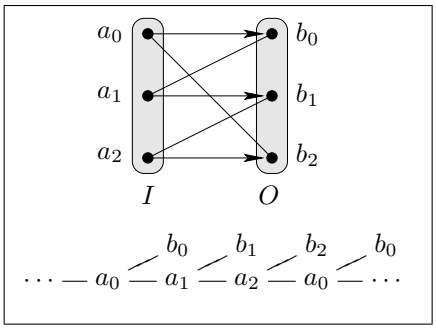


FIGURE 3-2. Example of a geometry with no flow (*c.f.* the second geometry from the left in Figure 3-1). Arrows indicate the action of an injective function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ along otherwise undirected edges. Also given is a reflexive and transitive binary relation $\preccurlyeq$ which satisfies conditions (3.3b) and (3.3c) for this function f, but which is not antisymmetric.

successors, implementing logical "wires" via single-qubit state transfer, and the partial order $\preccurlyeq$ represents *one possible* order (not necessarily one of minimum depth) in which the qubits may be measured to perform a unitary computation.

By [38], any geometry which has a flow underlies some one-way measurement pattern which can be obtained by the DKP construction, and which therefore performs a unitary transformation: moreover, the proof in [38] is by reduction to unitary circuits. We may then use flows to obtain a semantic map $\mathcal{S}$ (as described on page 81), corresponding to the choice of the *complete* DKP construction as a choice of representation map $\mathcal{R}$ from unitary circuits to the one-way measurement model, if (a) we can find a flow for a geometry (G, I, O); and (b) we can verify whether the *dependencies and corrections* of a measurement pattern are consistent with one arising from the DKP construction. The main subject of this Chapter is essentially how to perform these tasks efficiently.

3.2.1 Direct interpretation of flows in terms of unitary circuit structure

On pages 28 – 38, we presented the stable-index representation of unitary circuits, and in particular described the combinatorial structure of the "interaction graph", whose vertices consist of tensor indices (corresponding to "wire segments") of the unitary circuit. The edges of this graph consist of pairs of indices which are involved in a single unitary gate; and in particular, there is a function f mapping each deprecated index to a corresponding advanced index, imposing an arrow of time on the indices.

The map Φ of (2.29) from the JACZ($\mathbb{A}$) model to the ONEWAY($\mathbb{A}$) model, which forms the first part of the simplified DKP construction of Section **??**, identifies the interaction graph of a circuit in the JACZ($\mathbb{A}$) model with the geometry (G, I, O) of the corresponding one-way pattern. In particular, advanced indices correspond to prepared (or non-input) qubits, and deprecated indices to measured (or non-output) qubits; and the function f on the indices of the stable index expression may then be identified with a function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ on the geometry (G, I, O).

A flow consists of a function f and a partial order $\preccurlyeq$, where f may be considered to correspond to the mapping of deprecated indices to advanced indices in a stable index representation of a unitary circuit, and the partial order $\preccurlyeq$ to correspond to the order

described on page 38 in the discussion on "Recovering the order of unitary operations". Flows may then also be interpreted as describing the structure of a unitary circuit, expressed as a stable index tensor product, from the geometry (G, I, O) of a measurement pattern.

3.3 Graph constructions and characterizations for flows

The main result of this chapter is to characterize flows in graph-theoretic terms, which allows us to reduce the problem of deciding whether a geometry has a flow to problems in graph theory with efficient solutions, and to examine (and solve) an extremal problem which further bounds the running time of those algorithms. This will ultimately enable us to obtain an efficiently computable semantic map $\mathcal{S}$ (as described on page 81) for the DKP construction described in Section **??**, and provide a strong upper bound on its running time.

In this section, we consider useful constructions of directed graphs, culminating in the aforementioned graph-theoretic characterizations. These constructions presented in this section will also allow us to more easily describe the generality of the geometries which have flows. We also make use of these constructions to obtain uniqueness results, and bounds on the number of edges of graphs with flows, which ultimately lead to the first efficient algorithm for determining when a geometry (G, I, O) has a flow.

3.3.1 Generalizing to path covers/successor functions

We will begin by exploring how to relax the notion of a flow to admit a larger class of objects, starting by noting the most important properties of a flow.

Lemma 3-1. *If $(f, \preccurlyeq)$ is a flow for a geometry (G, I, O), then f is injective.*

Proof —
Suppose $x, y \in O^{\mathsf{c}}$ are such that $f(x) = f(y)$. Then $y \sim f(y) = f(x)$, so that $x \preccurlyeq y$, and $x \sim f(x) = f(y)$, so that $y \preccurlyeq x$. It follows that $x = y$. ■

The limitation to injective functions f motivates a description of flows in terms of vertex-disjoint collections of paths (or more precisely, an adaptation of the concept of a "path cover" [54]):

Definition 3-2. Let (G, I, O) be a geometry. A collection $\mathcal{C}$ of (possibly trivial[5]) directed paths in G is a *path cover* of (G, I, O) if the following conditions hold:

(i) each $v \in V(G)$ is contained in exactly one path (i.e. the paths cover G and are vertex-disjoint);

[5] A (directed) path or walk is *trivial* if it is of length zero, *i.e.* it starts and ends at a single vertex without traversing any edges (respectively, arcs).

(*ii*) each path in $\mathcal{C}$ is either disjoint from I, or intersects I only at its initial point;

(*iii*) each path in $\mathcal{C}$ intersects O only at its final point.

In the case $|I| = |O|$, a path cover of (G, I, O) is a collection of vertex-disjoint paths from I to O which covers all the vertices of G. For a flow $(f, \preccurlyeq)$, there is a natural connection between the function f and path covers for the geometry (G, I, O):

Lemma 3-2. *Let $(f, \preccurlyeq)$ be a flow on a geometry (G, I, O). Then there is a path cover $\mathcal{P}_f$ of (G, I, O) such that, for all vertices $v, w \in V(G)$, $v \to w$ is an arc in some path of $\mathcal{P}_f$ if and only if $w = f(v)$.*

Proof —

Let $(f, \preccurlyeq)$ be a flow on (G, I, O). Define a digraph P on the vertices of G, and with arcs $v \to f(v)$ for $v \in O^{\mathsf{c}}$. Because f is both a function and injective, every vertex in P has maximal out-degree and maximal in-degree 1. Thus, P is a collection of vertex-disjoint dipaths and closed walks.[6] Furthermore, for every arc $(v \to w) \in A(P)$, we have $v \preccurlyeq w$; by induction, $v \preccurlyeq z$ whenever there is a dipath from v to z in P. Then if v and z are such that there are dipaths from v to z and from z to v, then $v \preccurlyeq z$ and $z \preccurlyeq v$, in which case $x = z$ and the dipaths are trivial. Thus, P is acyclic, so P consists entirely of vertex-disjoint dipaths.

Let $\mathcal{P}_f$ be the collection of maximal dipaths in P. We show that $\mathcal{P}_f$ satisfies each of the criteria of Definition 3-2:

(*i*) Any vertex v which is neither in $\mathrm{dom}(f)$ nor $\mathrm{img}(f)$ will be isolated in P: then, the trivial path on v is an element of $\mathcal{P}_f$. All other vertices are in either $\mathrm{dom}(f)$ or $\mathrm{img}(f)$, and so are contained in a non-trivial path of $\mathcal{P}_f$. As these paths are vertex-disjoint, each vertex is contained in exactly one path.

(*ii*) Each vertex in I has in-degree 0, and so may only occur at the beginning of any path in $\mathcal{P}_f$.

(*iii*) The vertices in P which have out-degree 0 are precisely the output vertices O: therefore one occurs at the end of every path, and they may only occur at the end of paths in $\mathcal{P}_f$.

Then $\mathcal{P}_f$ is a path cover, whose paths contain only arcs $v \to f(v)$, as required. ■

It will be occasionally be convenient to alternate between descriptions of problems in terms of path covers for a given geometry, and functions f such that $\mathcal{P}_f$ is a path cover. In particular, we may define:

Definition 3-3. A *successor function* for a geometry (G, I, O) is an injective function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ such that the maximal orbits of f form a path cover for (G, I, O). A

[6] A closed walk is one which begins and ends at the same vertex.

successor function f for (G, I, O) is a *flow function* if there exists a partial order $\preccurlyeq$ on $V(G)$ such that $(f, \preccurlyeq)$ is a flow.

The terminology of "successor function" is meant to be suggestive of the mapping $v_j \mapsto v_{j+1}$ between successive indices in a stable index tensor expression for a single qubit, which in turn correspond (by the comments made on page 32) to successive segments of a wire (separated by single-qubit gates which do not preserve standard basis states) in a quantum circuit diagram. The generalization from flow functions to successor functions will prove very helpful in the analysis below.

3.3.2 Generalizing from partial orders to pre-orders

Figure 3-2 (on page 86) illustrates a geometry with successor functions, but which has no flow functions. There are only two possible successor functions f and f' for that geometry, given by

$$f(a_j) = b_j \ , \qquad f'(a_j) = b_{(j-1) \bmod 3} \tag{3.4}$$

(*i.e.* taking the appropriate representative modulo 3 for the index in the case of f'). The successor function f is the function actually illustrated in Figure 3-2, and $f' : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ corresponds to instead directing the diagonal edges of the graph illustrated there. To show that neither of these are flow functions, we may show that any reflexive and transitive binary relation satisfying the conditions (3.3b) and (3.3c) will fail to be anti-symmetric; in which case there can be no partial order $\leqslant$ such that either $(f, \leqslant)$ or $(f', \leqslant)$ is a flow for that geometry. Specifically, to satisfy the flow conditions for f, such a binary relation would satisfy the constraints $a_0 \leqslant a_1 \leqslant a_2 \leqslant a_0$, while for f' it would have to satisfy the constraints $a_0 \geqslant a_1 \geqslant a_2 \geqslant a_0$. In either case, the vertices $a_j \in I$ are distinct but mutually related vertices: so any such relation fails to be a partial order. Because no partial order can satisfy the conditions (3.3b) and (3.3c) for either f or f', that geometry has no flow.

Analysis of the form above will be generally useful, and motivates a generalization from partial orders to pre-orders similar to our generalization from flow functions to successor functions.

Definition 3-4. Let (G, I, O) be a geometry and $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ be a successor function for (G, I, O). The *influence relation* $\lhd$ for f is the binary relation such that $v \lhd w$ for $v \in O^{\mathsf{c}}$ and $w \in V(G)$ if and only if $v \neq w$ and either $w = f(v)$ or $w \sim f(v)$. An *influencing pre-order* $\preccurlyeq$ for f is a pre-order on $V(G)$ which extends $\lhd$, *i.e.* a reflexive and transitive relation such that

$$v \preccurlyeq f(v) \tag{3.5a}$$

$$w \sim f(v) \implies v \preccurlyeq w \tag{3.5b}$$

holds for all $v \in O^{\mathsf{c}}$ and $w \in V(G)$. A *causal order* for f is a influencing pre-order for f which is also a partial order, *i.e.* a binary relation such that $(f, \preccurlyeq)$ is a flow.

Note that the influence relation $v \lhd w$ describes exactly the conditions (3.3b) and (3.3c) on a causal order: the definition of an influencing pre-order then relaxes nothing more of

the flow conditions than the condition that $\preccurlyeq$ be anti-symmetric. The proof given at the beginning of this section that the geometry of Figure 3-2 has no flows is essentially that every influencing pre-order of f or f' must somehow fail to be a full-fledged partial order.

3.3.3 Deciding if a successor function is a flow function

Every successor function has an influencing pre-order, and for each f, we identify one in particular:

Definition 3-5. Let (G, I, O) be a geometry and $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ be a successor function for (G, I, O). The *natural pre-order* $\preccurlyeq$ for f is the transitive and reflexive closure[7] of the relation $\lhd$.

The natural pre-order $\preccurlyeq$ for a successor function f is by definition an influencing pre-order, and in particular the coarsest influencing pre-order for f. As a result, it can be used to determine whether or not f is a flow function:

Lemma 3-3. *A successor function f is a flow function if and only if its natural pre-order $\preccurlyeq$ is a partial order.*

Proof —

If $\preccurlyeq$ is a partial order, then $(f, \preccurlyeq)$ is a flow, and so f is a flow function. For the converse, suppose that there is *some* partial order $\leqslant$ such that $(f, \leqslant)$ is a flow: then $\leqslant$ is an influencing pre-order for f. Being a partial order, for each distinct pair of vertices $v, w \in V(G)$, at least one of $v \not\leqslant w$ or $w \not\leqslant v$ holds; then, for such pairs of vertices, at least one of $v \not\preccurlyeq w$ or $w \not\preccurlyeq v$ holds as well, as $\preccurlyeq$ is coarser than $\leqslant$. Thus $\preccurlyeq$ is a partial order. ■

We can use this to efficiently decide when a successor function f is a flow function by reduction to the *transitive closure* problem on directed graphs: specifically, by constructing $\preccurlyeq$ from the influence relation $\lhd$ for f. While the algorithms will be simpler to describe in terms of the binary relations $\lhd$ and $\preccurlyeq$ themselves, we will describe the graph-theoretic presentation of this problem: this is the presentation used in the relevant literature, and the graph constructions we describe here will also prove convenient for analysis in the following sections.

Definition 3-6. For a successor function f on a geometry (G, I, O), the *influence digraph* is a directed graph with vertex-set $V(G)$, and with an arc $v \to w$ between $v, w \in V(G)$ if and only if $v \lhd w$.

An example of this construction is illustrated in Figure 3-3. We may then characterize the natural pre-order $\preccurlyeq$ of f in terms of the *transitive closure* of $\mathfrak{I}_f$:

[7] That is, $\preccurlyeq$ is the coarsest reflexive and transitive relation which extends $\lhd$, and so may be described as the logical conjunction of all influencing pre-orders. As there exists at least one influencing pre-order, namely the relation R such that vRw for all $v, w \in V(G)$, this closure is guaranteed to exist.

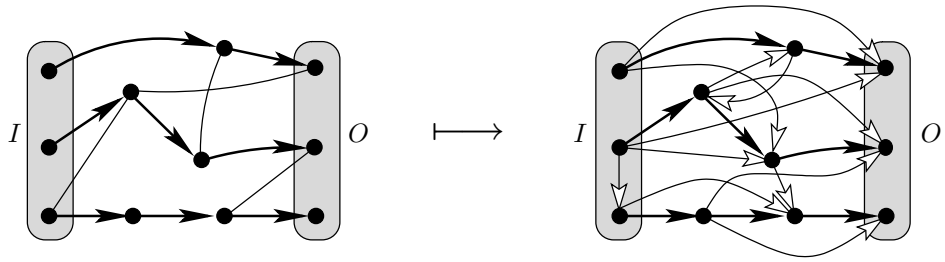


FIGURE 3-3. Illustration of the influence digraph construction. On the left: a geometry (G, I, O) with a successor function f, represented by the corresponding path cover $\mathcal{P}_f$. On the right: the corresponding influence digraph $\mathcal{I}_f$. Solid arrows represent arcs of the form $v \to f(v)$, and hollow arrows represent arcs $v \to w$ for $w \sim f(v)$ in the graph G.

Definition 3-7. The *transitive closure of a digraph* D is a directed graph $T(D)$ with $V(T(D)) = V(D)$, and such that $(v \to w) \in A(T(D))$ if and only if there is a non-trivial directed path from v to w in the digraph D.

If we let $\mathcal{T}_f$ be the transitive closure of $\mathcal{I}_f$, the natural pre-order of a successor function f is then the relation $\preccurlyeq$ such that $v \preccurlyeq w$ if and only if either $v = w$ or $(v \to w) \in A(\mathcal{T}_f)$ for two vertices $v, w \in V(\mathcal{I}_f)$. This allows us to reduce the construction of $\preccurlyeq$ to the following problem applied to $\mathcal{I}_f$:

Transitive Closure Problem: For a digraph D, construct its transitive closure $T(D)$.

The transitive closure problem is solvable in polynomial time by the Floyd–Warshall algorithm [65, 33]. We will present a faster algorithm using techniques presented in [106, Chapters 3 – 4], based on Tarjan's algorithm [127, 33] for determining the *strongly connected components* of a digraph (equivalence classes of vertices in a digraph which are mutually reachable by directed paths). We will describe this algorithm in detail in Section 3.4, where we will prove the following:

Theorem 3-4. *Let f be a successor function of a geometry (G, I, O), and let $n = |V(G)|$ and $k = |O|$. Then there is an algorithm constructing the natural pre-order $\preccurlyeq$ for f, or which determines that $\preccurlyeq$ is not a partial order, in time $O(k^2 n)$.*

3.3.4 Influencing walks and causal path covers

While a detailed analysis of the algorithms described above will wait until Section 3.4, it will be useful to consider the graph-theoretic structures which are involved in those algorithms. These structures also present a means of proving uniqueness results which led to the first algorithms for determining whether a geometry has a flow, and extremal results which allow refinements in their running times.

In the previous section, we observed that not only does the geometry of Figure 3-2 have a cyclic underlying graph, but also a complete cycle of relationships between some of its vertices in any influencing pre-order $\leqslant$ for a given successor function for that

geometry. These cycles of relationships will manifest themselves as directed cycles in the corresponding influence digraphs $\mathfrak{I}_f$, but can be traced also to circuit[8] in the original graph G, of the following sort:

Definition 3-8. Let G be a graph, and $\mathcal{P}$ a family of vertex-disjoint directed paths in G. A walk $W = u_0 u_1 \cdots u_\ell$ is an *influencing walk* for $\mathcal{P}$ if it is a concatenation of zero or more paths (*segments* of the influencing walk) of the following two types:

(i) $v \to w$, where this is an arc in some path of $\mathcal{P}$;

(ii) $v \to z \to w$, where $v \to z$ is an arc in some path of $\mathcal{P}$ and $wz \in E(G)$ is an edge not covered by $\mathcal{P}$.

A *vicious circuit* for $\mathcal{P}$ is a closed influencing walk for $\mathcal{P}$ with at least one segment.

It is easy to show (by induction) that we may also characterize an influencing walk W, for a family of vertex-disjoint paths $\mathcal{P}$, as one whose arcs (a) never traverse an edge of G in a direction opposite that of an arc of $\mathcal{P}$, and (b) do not traverse two edges in a row which are not covered by $\mathcal{P}$. Influencing walks were first identified as objects of interest by Broadbent and Kashefi [24], who examine their role in the depth complexity of unitaries in the one-way measurement model.

We will occasionally refer to the two types of segments as *type* (i) and *type* (ii) respectively. The motivation for the two types of segment is in the two ways in which distinct vertices $v, w \in V(G)$ may be related by the influence relation $\lhd$ for a successor function:

Lemma 3-5. *Let $\mathcal{P}_f$ be a path cover for (G, I, O) with successor function f, and let $\lhd$ and $\preccurlyeq$ be the influence relation and the natural pre-order of f respectively. Then $v \preccurlyeq w$ if and only if there is an influencing walk $W = v \to \cdots \to w$ for $\mathcal{P}_f$ from v to w, and in particular, $v \lhd w$ if and only if there is an influencing walk W for $\mathcal{P}_f$ from v to w which consists of one segment.*

Proof —

Suppose W is an influencing walk for $\mathcal{P}_f$ from v to w. A segment of type (i) is a dipath $v \to w$ between vertices $w = f(v)$, and a segment of type (ii) is a dipath $v \to z \to w$ for vertices v, z, and w such that $w \sim z$ and $z = f(v)$, and where $w \neq v$ (because the edge vz is covered by the path cover $\mathcal{P}_f$). In both cases, we have $v \lhd w$. Similarly, if $v \lhd w$, then there is either an influencing walk $W = v \to w$ or an influencing walk $W' = v \to z \to w$ for $z = f(v)$. The Lemma then holds for $\lhd$ and influencing walks of one segment.

An influencing walk in general (of *zero* or more segments) then corresponds to the reflexive and transitive closure of the influence relation, *i.e.* to the natural pre-order $\preccurlyeq$ of f. In particular, an influencing walk $v \to \cdots \to w$ of *zero* segments is the trivial walk on $v = w$, in which case $v \preccurlyeq w$. Influencing walks W of

[8] In this context, we are using the graph-theoretic term *circuit*, which usually refers to closed walks of length 3 or more.

$n \geqslant 1$ segments from v to w we may decompose into an influencing walk W' of $n-1$ segments from v to some vertex u, and a single segment from u to w. By induction, we have $v \preccurlyeq u \lhd w$, so $v \preccurlyeq w$. For the converse, suppose $v \preccurlyeq w$ for some vertices $v, w \in V(G)$. By definition, there is then a sequence of vertices $(u_j)_{j=0}^{\ell}$ for some $\ell \in \mathbb{N}$, such that $v = u_0 \lhd u_1 \lhd \cdots \lhd u_\ell = w$. Then, for each $0 \leqslant j < \ell$, we have $u_j \neq u_{j+1}$, and either $u_{j+1} = f(u_j)$, or $u_{j+1} \sim f(u_j)$, and we may define an dipath $W_j = u_j \to u_{j+1}$ or $W_j = u_j \to f(u_j) \to u_{j+1}$, respectively. In the latter case, if $u_{j+1} = f(f(u_j))$, the path W_j is a concatenation of two influencing walk segments of type (i); otherwise, W_j is itself a single influencing walk segment of either type (i) or type (ii). In any case, the concatenation $W = W_0 \cdots W_{\ell-1}$ of these paths is an influencing walk for $\mathcal{P}_f$ from v to w. ■

Because of the correspondence between segments of an influencing walk and the influence relation $\lhd$, a single segment influencing walk essentially corresponds to a single arc of the influence digraph $\mathfrak{I}_f$, and an influencing walk in general to a directed walk in $\mathfrak{I}_f$.

The close correspondence between influencing walks for $\mathcal{P}_f$ and the natural pre-order for f motivates the following definition:

Definition 3-9. A *causal path cover* $\mathcal{C}$ for a geometry (G, I, O) is a path cover which has no vicious circuits.

Using this further definition, we may then easily characterize flows in terms of path covers:

Theorem 3-6. *Let f be a successor function of a geometry (G, I, O) and let $\mathcal{P}_f$ be the path cover induced by f. Then f is a flow function if and only if $\mathcal{P}_f$ is a causal path cover.*

Proof —

Let $\preccurlyeq$ be the natural pre-order of f. By Lemma 3-5, we have $v \preccurlyeq w$ if and only if there is an influencing walk for $\mathcal{P}_f$ from v to w: then we have $v \preccurlyeq w$ and $w \preccurlyeq v$ for distinct $v, w \in V(G)$ if and only if there are influencing walks $W' = v \to \cdots \to w$ and $W'' = w \to \cdots \to v$ for $\mathcal{P}_f$. If there are such walks, then $C = W'W''$ is a vicious circuit for $\mathcal{P}_f$. Conversely, a vicious circuit $C = \sigma_1 \cdots \sigma_\ell$ for $\mathcal{P}_f$ (concatenated from some non-zero number of segments σ_j) can be decomposed into an influencing walk $W' = \sigma_1 \cdots \sigma_{\ell-1}$ between two distinct vertices v and w, and another influencing walk $W'' = \sigma_\ell$ from w to v. Then if there is such a circuit C, we have $v \preccurlyeq w$ and $w \preccurlyeq v$ for some two distinct vertices v and w, so that $\preccurlyeq$ is not a partial order. Thus, $\preccurlyeq$ is a partial order if and only if $\mathcal{P}_f$ lacks vicious circuits. By Lemma 3-3, f is a flow function if and only if its natural pre-order $\preccurlyeq$ is a partial order; the theorem then holds. ■

This completes our graph-theoretic characterization of flows. The main utility of these constructions and characterizations is to reduce the properties of the natural pre-order $\preccurlyeq$ of a function f to influencing walks for $\mathcal{P}_f$, or to walks in the influence digraph $\mathfrak{I}_f$. This change in emphasis will serve to clarify the graph-theoretic results which follow.

3.3.5 Bounding the number of edges of a geometry with a flow

For a geometry (G, I, O) with a flow-function f, let us call an edge $vw \in E(G)$ a *flow edge* if either $w = f(v)$ or $v = f(w)$, *i.e.* if vw is an edge covered by the path-cover $\mathcal{P}_f$; every other edge of G is a *non-flow* edge. We have seen how influencing walks for path covers $\mathcal{P}_f$ correspond to the natural pre-order $\preccurlyeq$ for f: and in this terminology, from the remarks made just after Definition 3-8, an influencing walk W is essentially a walk which follows the paths of $\mathcal{P}_f$, but which occasionally jumps across paths by traversing a non-flow edge (under the constraint that it does not traverse two such edges in a row). Thus, the more non-flow edges there are, the more freedom there is in how an influencing walk may be constructed.

It is reasonable to suppose that the more non-flow edges there are, the more likely there are to be influencing walks $W' = v \to \cdots \to w$ and and $W'' = w \to \cdots \to v$ for some distinct vertices $v, w \in V(G)$, producing a vicious circuit for $\mathcal{P}_f$. We might then ask if there is an upper bound on the number of non-flow edges that a geometry with a flow may have. Note that, whether or not a geometry (G, I, O) has a flow, a path cover $\mathcal{C}$ for (G, I, O) must have exactly $k = |O|$ paths, as all the paths of $\mathcal{C}$ must terminate at an output vertex, and all output vertices terminate such a path. These paths will collectively cover exactly $n - k$ edges of G, where $n = |V(G)|$, as there is a one-to-one correspondence between elements of O^{c} and the arcs of $\mathcal{C}$ which leave those vertices; all other edges of G are non-flow edges relative to this path cover $\mathcal{C}$. Then, we may describe this question without reference to any particular path cover in terms of the following:

Definition 3-10. For $n \geqslant k \geqslant 1$, define $\Gamma(n, k)$ to be the maximum number of edges in a geometry (G, I, O) which has a flow, subject to $|V(G)| = n$ and $|O| = k$.

For the question of whether *there exists* a geometry on n vertices and with m edges which admits a flow, we can consider any graph G with n vertices, under the constraint that it admits some family of vertex-disjoint (directed) paths $\mathcal{P} = \{P_1, \ldots, P_k\}$ covering the entire graph. Without loss of generality, we may then let O be the final points of those paths, and I be (an arbitrary subset of) the initial points of those paths. Because we may consider the outputs and inputs to be given by a family of paths in this way, we will often refer to the successor function of a family of paths $\mathcal{P}$ which covers the vertices of a graph, whether or not it is explicitly a path cover for a geometry.

As we noted on page 91, we may reduce this question to the question of whether the strongly-connected components of the influence digraph $\mathfrak{I}_f$ (Definition 3-6, page 90) all consist of exactly one vertex. This holds if and only if $\mathfrak{I}_f$ lacks directed circuits, *i.e.* if $\mathfrak{I}_f$ is *acyclic*. We may then consider whether the successor function f for $\mathcal{P}$ has a causal order, and consider the problem in terms of adding non-flow edges to a graph G, which initially contains only the paths of $\mathcal{P}$, under the constraint of keeping $\mathfrak{I}_f$ acyclic. This leads to a simple upper bound on $\Gamma(n, k)$:

Theorem 3-7. *$\Gamma(n, k) \leqslant kn - \binom{k+1}{2}$ for all integers $n \geqslant k \geqslant 1$.*

Proof —

We may base this proof on two simple observations about a path cover $\mathcal{P}_f =$

$\{P_1, \ldots, P_k\}$ such that $\mathfrak{I}_f$ is acyclic. For each $1 \leqslant j \leqslant k$, let n_j denote the number of vertices in the path P_j.

Observation 1. Consider a path $P_j = v_1 \to v_2 \to \cdots \to v_{n_j}$. If $v_a v_b \in E(G)$ for $b > a+1$, then $\mathfrak{I}_f$ will contain the directed cycle $v_a \to v_{a+1} \to \cdots \to v_{b-1} \to v_a$. Then, if $\mathfrak{I}_f$ is acyclic, we have $v_a v_b \in E(G)$ for $a < b$ if and only if $b = a+1$, that is if $v_a v_b$ is a flow edge.

Observation 2. Consider any two distinct paths $P_h = v_1 \to v_2 \to \cdots \to v_{n_h}$ and $P_j = w_1 \to w_2 \to \cdots \to w_{n_j}$. If there are edges $v_a w_b$, $v_c w_d \in E(G)$ for some $a < c$ and $b > d$, then $\mathfrak{I}_f$ will contain the directed cycle $v_a \to \cdots \to v_{c-1} \to w_d \to \cdots \to w_{b-1} \to v_a$. Then, if $\mathfrak{I}_f$ is acyclic, we have non-flow edges $v_a w_b$, $v_c w_d \in E(G)$ only if the relation

$$a < c \implies b \leqslant d \tag{3.6}$$

holds. (The condition $b > d \implies a \geqslant c$ is equivalent, up to relabelling).

The first observation implies that the only non-flow edges that (G, I, O) may contain if it has a flow are edges between distinct paths P_h and P_j; and the second imposes a useful constraint on the endpoints of pairs of non-flow edges. We may then assume that these constraints hold for all non-flow edges without loss of generality.

We define a function λ from the non-flow edges to $\mathbb{N}$ as follows. For a non-flow edge $v_a w_b$ between the a^{th} vertex of a path $P_h = v_1 \to \cdots \to v_{n_h}$ and the b^{th} vertex of a path $P_j = w_1 \to \cdots \to w_{n_j}$, we let

$$\lambda(v_a w_b) = a + b\,. \tag{3.7}$$

Suppose $\mathfrak{I}_f$ is acyclic. For two distinct edges $v_a w_b$ and $v_c w_d$ are non-flow edges between the two paths P_h and P_j. Observation 2 then implies that $\lambda(v_a w_b)$ and $\lambda(v_c w_d)$ differ:

- If $a < c$, we have $b \leqslant d$, in which case $\lambda(v_a w_b) = a + b < c + d = \lambda(v_c w_d)$;
- If $c < a$, we have $d \leqslant b$, in which case $\lambda(v_c w_d) = c + d < a + b = \lambda(v_a w_b)$.

Then, restricted to the non-flow edges between P_h and P_j, λ is injective. Because $2 \leqslant \lambda(e) \leqslant n_h + n_j$ for all non-flow edges e between these paths, there are at most $n_h + n_j - 1$ non-flow edges between those two paths of $\mathcal{P}$.

Applying this to all pairs of paths P_h and P_j, the number of non-flow edges in G is then bounded above by

$$\sum_{1 \leqslant h < j \leqslant k} (n_h + n_j - 1) \;=\; \frac{1}{2}\left[\sum_{h=1}^{k}\sum_{j=1}^{k}(n_h + n_j - 1) \;-\; \sum_{h=1}^{k}(2n_h - 1)\right]$$

$$= \frac{1}{2}\left[\sum_{h=1}^{k}(kn_h + n - k) \;-\; \sum_{h=1}^{k}(2n_h - 1)\right]$$
$$= \frac{1}{2}\left[2kn - k^2 - 2n + k\right]$$
$$= kn - n - \tfrac{1}{2}(k^2 - k)\,. \tag{3.8}$$

As the number of edges in the paths P_j themselves is collectively $n - k$, the total number of edges G may have if $\mathfrak{I}_f$ is acyclic is at most $kn - k - \frac{1}{2}(k^2 - k) = kn - \binom{k+1}{2}$. ■

This result in itself will prove sufficient to strongly bound the running time of the algorithms presented in this chapter. For completeness, however, we will consider a construction which saturates this upper bound. Consider the following construction for any $n \geqslant k \geqslant 1$:

Definition 3-11. Let $n_1, n_2, \ldots, n_k$ be an integer partition of n such that $n_1 \leqslant n_2 \leqslant \cdots \leqslant n_k$. For each $1 \leqslant j \leqslant k$, let $P_j = v_{j,1}\, v_{j,2} \cdots v_{j,n_j}$. Define $\mathcal{G}(n_1, \ldots, n_k)$ to be the graph containing these paths, as well as the following non-flow edges for each $1 \leqslant h < j \leqslant k$:

(*i*) If $n_h > 1$, then for each $1 \leqslant a < n_h$, we include the edge $v_{h,a}v_{j,a}$;

(*ii*) If $n_j > 1$, then for each $1 \leqslant a < n_h$, we include the edge $v_{h,a+1}v_{j,a}$;

(*iii*) For each $n_h \leqslant a \leqslant n_j$, we include the edge $v_{h,n_h}v_{j,a}$.

We will describe (*i*) – (*iii*) as the *types* of non-flow edges in $\mathcal{G}(n_1, \ldots, n_k)$. We also define $\mathcal{D}(n_1, \ldots, n_k)$ to be the influence digraph $\mathfrak{I}_f$ for the successor function f of the family of paths $\{P_1, \ldots, P_k\}$.

Figure 3-4 illustrates an example of this graph construction for $\mathcal{G}(6, 8, 9)$. (A diagram of the corresponding digraph $\mathcal{D}(6, 8, 9)$ has a rather large number of arcs, and is not shown.)

For the sake of brevity, let $G = \mathcal{G}(n_1, \ldots, n_k)$. Define $\mathcal{P} = \{P_1, \ldots, P_k\}$, and let f be the successor function for this family of paths. Then each non-flow edge in G induces up to two arcs in the associated digraph $\mathfrak{I}_f$. For each $1 \leqslant h < j \leqslant k$, the arcs of $\mathfrak{I}_f$ induced by non-flow edges between the paths P_h and P_j are of six different types, labelled (a) – (f), which we group together by the type of the non-flow edge which induces them:

$$(i) \longmapsto \begin{cases} \text{(a)}\;\; v_{h,a-1} \to v_{j,a} & \text{for } 1 < a \leqslant n_h \text{ (if } n_h > 1\text{), and} \\ \text{(b)}\;\; v_{j,a-1} \to v_{h,a} & \text{for } 1 < a \leqslant n_h \text{ (if } n_j > 1\text{);} \end{cases} \tag{3.9a}$$

$$(ii) \longmapsto \begin{cases} \text{(c)}\;\; v_{h,a} \to v_{j,a} & \text{for } 1 \leqslant a < n_h - 1 \text{ (if } n_h > 1\text{), and} \\ \text{(d)}\;\; v_{j,a-1} \to v_{h,a+1} & \text{for } 1 < a \leqslant n_h - 1 \text{ (if } n_h > 1\text{);} \end{cases} \tag{3.9b}$$

$$(iii) \longmapsto \begin{cases} \text{(e)}\;\; v_{h,n_h-1} \to v_{j,a} & \text{for } n_h \leqslant a \leqslant n_j\text{, and} \\ \text{(f)}\;\; v_{j,a-1} \to v_{h,n_h} & \text{for } \max\{n_h, 2\} \leqslant a \leqslant n_j \text{ (if } n_j > 1\text{).} \end{cases} \tag{3.9c}$$

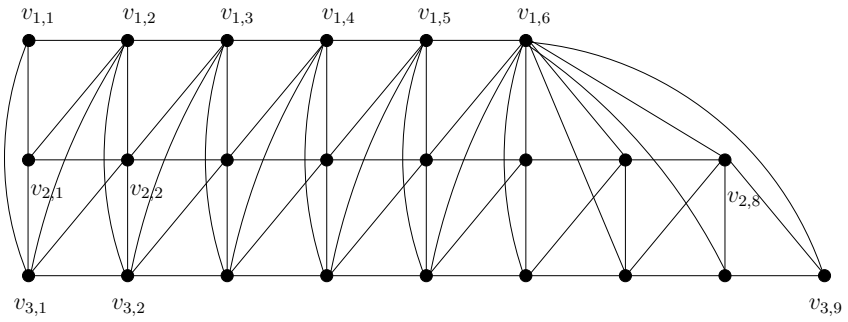


FIGURE 3-4. The graph $\mathcal{G}(n_1, n_2, n_3)$ for $n_1 = 6$, $n_2 = 8$, $n_3 = 9$.

We may refer to (a) – (f) as *rules* for inclusion of arcs in $\mathcal{D}(n_1, \ldots, n_k)$. In addition to these arcs, $\mathcal{D}(n_1, \ldots, n_k)$ also contains arcs $v_{j,a} \to v_{j,a+1}$ arising from orienting the paths P_j themselves, and arcs $v_{j,a} \to v_{j,a+2}$ from traversing two path edges in a row (corresponding to the adjacency relation $v_{j,a+2} \sim f(v_{j,a})$, which yields $v_{j,a} \lhd v_{j,a+2}$).

Lemma 3-8. *For any $n \geqslant k \geqslant 1$ and any integer partition $n_1 \leqslant \cdots \leqslant n_k$ of n, the digraph $\mathcal{D}(n_1, \ldots, n_k)$ is acyclic.*

Proof —

Let $D = \mathcal{D}(n_1, \ldots, n_k)$ for the sake of brevity. All of the arcs in D produced by the rules (a) – (e) are either of the form $v_{h,a} \to v_{j,b}$ with $a < b$ and no constraints on h and j, or $v_{h,a} \to v_{j,a}$ with $h < j$. In either case, if an arc $v_{h,a} \to v_{j,b}$ is of one of the types (a) – (e), we have $(a, h) < (b, j)$ in the lexicographic ordering on ordered pairs of integers. The same also holds for the arcs $v_{h,a} \to v_{h,a+1}$ and $v_{h,a} \to v_{h,a+2}$ yielded by the paths P_h. Then, if there are arcs in D of the form $v_{j,b} \to v_{h,a}$ where $(b, j) > (a, h)$, they must arise from the rule (f), in which case $a = n_h$.

Note that none of the rules (a) – (f) produce arcs which leave the final vertex v_{h,n_h} of any path P_h, so there are no non-trivial walks in D which leave such a vertex. Then, it is easy then to show by induction that if there is a directed walk in D between distinct vertices $v_{h,a}$ and $v_{j,b}$, either $(a, h) < (b, j)$ in the lexicographic order, or $b = n_j$.

Let $v_{h,a}$ and $v_{j,b}$ be two vertices, with a directed walk W from $v_{h,a}$ to $v_{j,b}$. Because of the existence of W, we know that $a \neq n_h$; then, there is a directed walk from $v_{j,b}$ to $v_{h,a}$ only if $(b, j) < (a, h)$. We would then have $b = n_j$, in which case there are no directed walks from $v_{j,b}$ to *any* other vertices in D. So, for any two distinct vertices $v_{h,a}$ and $v_{j,b}$, there cannot both be a directed walk $v_{h,a} \to \cdots \to v_{j,b}$ and also a directed walk $v_{j,b} \to \cdots \to v_{h,a}$. Thus, D is acyclic. ■

As well as giving rise to an acyclic digraph $D(G, P_1, \ldots, P_k)$, we also have:

Lemma 3-9. $\left|E\big(\mathcal{G}(n_1, \ldots, n_k)\big)\right| = kn - \binom{k+1}{2}$, *for any $n \geqslant k \geqslant 1$ and integer partition $n_1 \leqslant \cdots \leqslant n_k$ of n.*

Proof —

Between any pair of paths P_h and P_j in $\mathcal{G}(n_1, \dots, n_k)$, there are $n_h - 1$ edges of type (i), $n_h - 1$ edges of type (ii), and $n_j - n_h + 1$ edges of type (iii). There are then $n_h + n_j - 1$ non-flow edges between P_h and P_j. This saturates the upper bound for non-flow edges between pairs of paths in Theorem 3-7: summed over all pairs of paths, and including the edges in the paths P_h, the total number of edges in $\mathcal{G}(n_1, \dots, n_k)$ is then $kn - \binom{k+1}{2}$. ■

Lemmas 3-8 and 3-9, together with Theorem 3-7, prove:

Theorem 3-10. *$\Gamma(n,k) = kn - \binom{k+1}{2}$ for all integers $n \geqslant k \geqslant 1$. That is, for every $n \geqslant k \geqslant 1$ and any geometry with $|V(G)| = n$ and $|O| = k$, the geometry (G, I, O) has a flow only if $m \leqslant nk - \binom{k+1}{2}$, where $m = |E(G)|$; and there exist geometries (G, I, O), e.g. with $G = \mathcal{G}(n_1, \dots, n_k)$, for any integer partition $n_1 \leqslant \cdots \leqslant n_k$ of n, which saturate this bound.*

In later sections, we will only need the result of Theorem 3-7: the matching lower bound only serves to prove that no improvement in the upper bound is possible.

3.3.6 Uniqueness in the case $|I| = |O|$

Influencing walks for path-covers are closely related to *walks which alternate with respect to* $\mathcal{P}$, as defined by Diestel [54] in his presentation of a clever proof of Menger's Theorem (a forerunner of the Min-Flow/Max-Cut theorem) by [21].

Definition 3-12 [54, page 64]**.** Let $\mathcal{P} = \{P_1, \dots, P_k\}$ be a collection of vertex-disjoint $I - O$ paths $P_j = v_{j,1} \cdots v_{j,n_j}$. A walk $W = u_0 u_1 \cdots u_\ell$ *alternates with respect to* $\mathcal{P}$ if it starts at a vertex in I not covered by $\mathcal{P}$, and if the following three conditions hold for all $0 \leqslant j \leqslant \ell$:

- (i) if $u_j u_{j+1}$ is covered by $\mathcal{P}$, then $u_j = v_{h,a+1}$ and $u_{j+1} = v_{h,a}$ for some $1 \leqslant h \leqslant k$ and $1 \leqslant a < n_h$;
- (ii) if $u_j = u_h$ for some $h \neq j$, then u_j is covered by $\mathcal{P}$;
- (iii) if u_j is covered by $\mathcal{P}$ for some $0 < j < \ell$, then at least one of $u_{j-1}u_j$ or $u_j u_{j+1}$ is covered by $\mathcal{P}$.

Such walks share with influencing walks the property that, for any vertex covered by $\mathcal{P}$, at least one of the two edges incident to the vertex are covered by $\mathcal{P}$ as well. By relaxing the requirement that W must start at a vertex not covered by $\mathcal{P}$, we may also consider "alternating walks" with respect to path covers: it is easy to see that closed "alternating walks" of this sort correspond to vicious circuits whose arcs have been reversed.

The role that alternating walks play in the proof of Menger's Theorem in [21], which we present as Lemma 3-13 on page 102, provides some intuition of the importance of the link with alternating walks. For completeness, the statement of the Theorem is as follows (see *e.g.* [54]):

Menger's Theorem. *Let G be a graph and $I, O \subseteq V(G)$. Then the minimum size of a set S such that $V \setminus S$ is disconnected, with I and O in different components, is equal to the maxmum number of vertex-disjoint $I - O$ paths in G.*

Alternating walks arise as a sort of "difference" of distinct collections of vertex-disjoint $I - O$ paths.[9] Specifically, an $I - O$ walk W which alternates with respect to a collection of vertex-disjoint paths $\mathcal{P}$ can be used to obtain a new collection $\mathcal{P}'$ with strictly more paths by using W as an *augmenting walk*, as in a network flow:[10] interpreting W as a "difference" of the two families $\mathcal{P}'$ and $\mathcal{P}$, we can obtain $\mathcal{P}'$ by "adding" W to $\mathcal{P}$. These addition and subtraction analogies can be made formal by considering the skew-symmetric directed adjacency matrices A and A' of the families of paths $\mathcal{P}$ and $\mathcal{P}'$ as directed subgraphs of G, given by

$$A_{uv} = \left\{ \begin{array}{cl} 1, & \text{if } u \to v \text{ is an arc of } \mathcal{P} \\ -1, & \text{if } v \to u \text{ is an arc of } \mathcal{P} \\ 0, & \text{otherwise} \end{array} \right\} \tag{3.10}$$

(and similarly for A' and $\mathcal{P}'$), and considering the digraphs yielded by taking sums and differences of these adjacency matrices when those sums/differences are $\{-1, 0, 1\}$-matrices.

Maximal collections of vertex-disjoint paths do not have such "augmenting" walks, and path covers for geometries (G, I, O) are by necessity a maximal collection of such paths. A difference of two path covers would then not contain any augmenting walks. However, it is not difficult to see that the symmetric difference will contain walks which are very much like alternating walks, except that they must begin and end at the same point, which is necessarily covered by the path cover. This intuition suggests that path covers which lack vicious circuits should be unique. In fact, we may prove something somewhat stronger:

Theorem 3-11. *Let (G, I, O) be a geometry such that $|I| = |O|$. If (G, I, O) has a causal path cover $\mathcal{C}$, then $\mathcal{C}$ is also the only maximum-size collection of vertex-disjoint dipaths from I to O.*

Proof —

Suppose that $\mathcal{C}$ is a path cover for (G, I, O) with successor function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, and suppose there is a maximum-size collection $\mathcal{F}$ of vertex-disjoint $I - O$ dipaths which differs from $\mathcal{C}$. Let $S \subseteq V(G)$ be the set of vertices not covered by $\mathcal{F}$: because $|\mathcal{F}| = |\mathcal{C}| = |I| = |O|$, we have $S \cap I = \varnothing$ and $S \cap O = \varnothing$, in which case $\mathcal{F}$ is a path cover for the geometry $(G \setminus S, I, O)$. Let $\tilde{f}$ be the successor function of $\mathcal{F}$ on that geometry: then $\tilde{f}$ is injective. Because $|I| = |O|$, $\tilde{f}$ will also be a bijection from $V(G) \setminus (O \cup S)$ to $V(G) \setminus (I \cup S)$. Then, let $\tilde{g}$ be the inverse function of $\tilde{f}$.

Because $\mathcal{C}$ and $\mathcal{F}$ differ, there must exist a vertex $v \in O^{\mathsf{c}}$ such that $v \to f(v)$ is not an arc of any path of $\mathcal{F}$. Note also that for $z \in \mathrm{dom}(f)$, $f(z) \notin \mathrm{dom}(f)$ holds

[9] The role of "alternating walks" and "augmenting walks" for vertex-disjoint $I - O$ paths, phrased in terms of differences of sets of paths, is similar to the analogous notions for *e.g.* matchings in graphs, and have a basis (so to speak) in matroid theory. The analogy here is used here only for illustrative purposes: a thorough discussion of augmenting walks in the context of matroid theory is beyond the scope of this thesis.

[10] For a discussion of augmenting walks in the context of network flows, see *e.g.* [33].

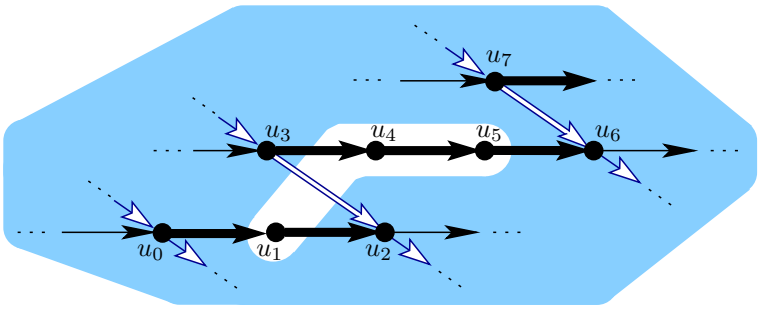


FIGURE 3-5. [*colour online*] Illustration of an unbounded influencing walk, for a path cover $\mathcal{C}$ (solid arrows), induced by the presence of another maximum-size collection $\mathcal{F}$ of paths from I to O (hollow arrows). The shaded area represents the set of vertices covered by $\mathcal{F}$, which in this illustration does not include all of the vertices. The arcs of $\mathcal{C}$ and of $\mathcal{F}$ which are thicker are those involved in the influencing walk.

only if $f(z) \in O \setminus I \subseteq \mathrm{dom}(\tilde{g})$; that is, $f(z) \in \mathrm{dom}(f) \cup \mathrm{dom}(\tilde{g})$. Then, define a vertex sequence $(u_j)_{j\in\mathbb{N}}$ in G by setting $u_0 = v$, $u_1 = f(v)$, and

$$u_{j+1} = \left\{ \begin{array}{l} f(u_j)\ , \quad u_j \in S \text{ or } u_j \neq f(u_{j-1}) \\ \tilde{g}(u_j)\ , \quad u_j \notin S \text{ and } u_j = f(u_{j-1}) \end{array} \right\} \tag{3.11}$$

for all $j \geqslant 1$. We then have $u_j \sim u_{j+1}$ for all $j \in \mathbb{N}$, so this defines a directed walk $u_0 \to u_1 \to \cdots$ in G. Figure 3-5 illustrates this construction.

Observation 1. Any three consecutive vertices u_j, u_{j+1}, u_{j+2} are distinct. That u_{j+1} differs from u_j and u_{j+2} follows from $u_j \sim u_{j+1}$ and $u_{j+1} \sim u_{j+2}$, as we do not permit G to have self-loops. We have either $u_2 = f(u_1)$ or $u_2 = \tilde{g}(u_1)$, with the latter being equivalent to $\tilde{f}(u_2) = u_1$. In the former case, u_0 and u_2 lie along on a common path, so $u_0 \neq u_2$; and because $\tilde{f}(u_0) \neq f(u_0) = u_1$ by the choice of u_0, we have $u_2 \neq u_1$ in the latter case. We may induct similarly, if the proposition holds up to some particular $j \geqslant 0$:

- Suppose $u_{j+3} = f(u_{j+2})$. If we also have $u_{j+2} = f(u_{j+1})$, then u_{j+3} and u_{j+1} lie along a common path of $\mathcal{C}$. Otherwise, $u_{j+1}u_{j+2}$ is an edge not covered by $\mathcal{C}$ while $u_{j+2}u_{j+3}$ is. In either case, $u_{j+3} \neq u_{j+1}$.
- Suppose $u_{j+3} = \tilde{g}(u_{j+2})$. Then $u_{j+2} \notin S$ and $u_{j+2} = f(u_{j+1})$; the latter of which implies that either $u_{j+1} \in S$ or $u_{j+1} = \tilde{g}(u_j)$. In the former case, $u_{j+1} \notin \mathrm{img}(\tilde{g})$, so that $u_{j+1} \neq u_{j+1}$; and if $u_j = \tilde{f}(u_{j+1})$, we have

$$u_{j+3} = \tilde{g}(u_{j+2}) \neq \tilde{g}(u_j) = u_{j+1}\ . \tag{3.12}$$

In any case, u_{j+1}, u_{j+2}, and u_{j+3} all differ, so the claim holds by induction.

Observation 2. For any $\ell \geqslant 0$, the walk $W_\ell = u_0 \to \cdots \to u_\ell$ is an influencing walk. By definition, $u_0 \to u_1$ is an arc in some path of $\mathcal{C}$, so this is an influencing walk, as is the trivial walk on u_0. Otherwise, suppose that

for some particular $\ell \geqslant 0$ we have both $W_\ell = u_0 \to \cdots \to u_\ell$ and $W_{\ell+1} = u_0 \to \cdots \to u_{\ell+1}$ are influencing walks.

- If $u_\ell \to u_{\ell+1}$ is not an arc of $\mathcal{C}$, then we have $u_{\ell+1} \neq f(u_\ell)$, in which case we have $u_{\ell+2} = f(u_{\ell+1})$. Similarly, if $u_{\ell+1} \in S$, we also have $u_{\ell 21} = f(u_\ell)$. In either case, $u_{\ell+1} \to u_{\ell+2}$ is a segment of type (i).
- If $u_\ell \to u_{\ell+1}$ is an arc of $\mathcal{C}$ and $u_{\ell+1} \notin S$, then we have $u_{\ell+2} = \tilde{g}(u_{\ell+1})$. Because $u_{\ell+2} \neq u_\ell$ by Observation 1, the arc $u_{\ell+1}u_{\ell+2}$ is not an arc covered by $\mathcal{C}$, in which case $u_\ell \to u_{\ell+1} \to u_{\ell+2}$ is a segment of type (ii).

Then, $W_{\ell+2} = u_0 \to \cdots \to u_{\ell+2}$ is either a concatenation of $W_{\ell+1}$ with a segment of type (i), or a concatenation of W_ℓ with a segment of type (ii), and is in either case also an influencing walk. The claim then holds by induction.

Because G is a finite graph, the Pigeon Hole Principle implies that there must be integers $\ell, \ell' \in \mathbb{N}$ with $\ell < \ell'$, $u_\ell = u_{\ell'}$, and $u_{\ell+1} = u_{\ell'+1}$. Because $W_{\ell'+1}$ is an influencing walk, either u_ℓ is at the beginning of a segment, or $u_{\ell+1}$ is (or both). if u_ℓ is at the beginning of a segment, the closed walk $u_\ell \to \cdots \to u_{\ell'}$ is a concatenation of segments from W, and so is a closed influencing walk for $\mathcal{C}$. Otherwise, $u_{\ell+1} \to \cdots \to u_{\ell'+1}$ is a concatenation of segments from W, and is in this case a closed influencing walk for $\mathcal{C}$. In either case, there exists a vicious circuit for $\mathcal{C}$, in which case $\mathcal{C}$ is not a causal path cover.

Thus, if $\mathcal{C}$ is a causal path cover, there can be no such vertex sequence $(u_j)_{j\in\mathbb{N}}$ as defined above, and so there can be no maximum family of vertex-disjoint $I - O$ paths $\mathcal{F}$ which differs from $\mathcal{C}$. $\mathcal{C}$ is then the unique such family of paths. ■

In the case where $|I| = |O|$, *i.e.* for geometries of one-way patterns corresponding to unitary circuits without measurements or trace-out, we may then reduce the problem of finding a flow to that of finding an arbitrary collection $\mathcal{C}$ of vertex-disjoint $I - O$ paths of maximum size. If this collection of paths is not a path cover, we know that any path cover for (G, I, O) cannot be causal, and so the geometry has no flow; otherwise, we may test the successor function f of $\mathcal{C}$ to see if it is a flow function, which by Lemma 3-3 and Theorem 3-4 (page 90 and following) can be determined in polynomial time. This will allow us to prove the following in Section 3.4:

Theorem 3-12. *Let (G, I, O) be a geometry with $|I| = |O| = k$ and $|V(G)| = n$. Then there is an algorithm which determines that (G, I, O) does not have a flow, or constructs a flow $(f, \preccurlyeq)$ for (G, I, O) such that $\preccurlyeq$ is the coarsest causal order for f, in time $O(k^2 n)$.*

3.4 Flow-finding algorithms

Throughout this section, we will suppose that we are given as input some particular geometry (G, I, O), and define $n = |V(G)|$, $m = |E(G)|$, and $k = |O|$. In some parts of our analysis, we will further suppose that $|I| = |O|$.

Partial function data types. A mutable variable for a partial function f between two sets A and B (which we will denote $f : A \rightharpoonup B$) will be represented as arrays $f : A \longrightarrow B \cup \{\texttt{nil}\}$, where `nil` is a reserved value not contained in any set of vertices or indices. The domain of f is then the set of $x \in A$ for which $f(x) \neq \texttt{nil}$. We will also adopt the convention that a partial function f is injective when $f(x) = f(y) \neq \texttt{nil} \iff x = y$ holds for all $x, y \in A$.

Set data types. For a mutable set variable S, we will consider to be implemented using a mixed array/linked list structure, where an array entry $S(v)$ either contains a value `nil` when $v \notin S$, or a node with links to at most two other nodes, representing arbitrarily designated "previous" and "next" elements of the set (using `nil` if one or both of these are not defined) when $v \in S$. A specially designated master node is kept to point to the "first" element of the set, or `nil` if the set is empty: newly added elements of the set may be inserted to the beginning of this list of nodes. This will allow for constant-time elementhood testing, element addition and removal, and comparison with the empty set, and iteration across the set can be done in $O(|S|)$ steps; it also provides a total ordering of S if required.

3.4.1 Eliminating geometries with too many edges

As we proved in Theorem 3-7, if G has more than $kn - \binom{k+1}{2}$ edges, then (G, I, O) cannot have a flow. Then, as a preliminary step, we may reject any geometry which has more than this number of edges, so we may assume that $m \in O(kn)$for the purposes of run-time analysis. (As we proved in Lemma 3-9, the bound of $m \leqslant kn - \binom{k+1}{2}$ is tight for geometries with flows.)

3.4.2 Finding a potential flow-function f when $|I| = |O|$

In the case where $|I| = |O|$, the uniqueness result of Theorem 3-11 allows us to reduce finding a causal path cover to finding an arbitrary maximum-size vertex-disjoint family of $I - O$ paths. This problem may be reduced to an instance of network flow, by finding a maximum integer flow in a digraph N with unit edge capacities, constructed from G so that edge-disjoint paths in N correspond to vertex-disjoint paths in G [31]. Here, we describe a different algorithm, based on the proof of the lemma in [21] (reproduced here with minor changes in order to facilitate description of the algorithm) used to prove Menger's Theorem :

Lemma 3-13 [21, Theorem 3]**.** *Let G be a digraph, $I, O \subseteq V(G)$ such that I cannot be separated from O by a set of κ or fewer vertices, for some $\kappa \in \mathbb{N}$. If $\mathcal{W} \subseteq G$ is a subgraph consisting of κ vertex-disjoint $I - O$ paths in G, there is a subgraph $\mathcal{U} \subseteq G$ consisting of $\kappa + 1$ disjoint $I - O$ paths, such that the terminal points of the paths of $\mathcal{U}$ contain those of $\mathcal{W}$.*

Proof —

We will induct on supersets of O contained in the vertex-set $V(G)$. If $O = V(G)$,

then $I \subseteq O$, in which case $\mathcal{U} = G[I]$ contains at least $\kappa+1$ (trivial) paths from I to O. Otherwise, we suppose that the Lemma holds for all proper supersets $O' \supset O$.

Let $R = O \cap V(\mathcal{W})$: as $|R| = \kappa$, removing R from G does not suffice to separate I from O, in which case the graph $G \setminus R$ contains a path P from I to O. If P is disjoint from $\mathcal{W}$, we may let $\mathcal{U} = \mathcal{W} \cup P$. Otherwise, let x be the final vertex on P which intersects $\mathcal{W}$, and let W be the path of $\mathcal{W}$ containing x.

Let P_1 be the subpath of P from x to O, W' be the subpath of W from I to x, and P_2 be the subpath of W from x to O. We may augment the set O by defining $O' = O \cup V(P_1) \cup V(P_2)$, and correspondingly reduce the path W by defining $\mathcal{W}' = (\mathcal{W} \setminus W) \cup W'$; then $\mathcal{W}'$ consists of κ paths from I to O'. As $O \subset O' \subseteq V(G)$, by hypothesis there exists a subgraph $\mathcal{U}' \subseteq G$ consisting of $\kappa+1$ vertex-disjoint paths, such that the terminal points of $\mathcal{U}'$ contain those of $\mathcal{W}'$.

Let $\omega \in O'$ be the terminal point in $\mathcal{U}'$ which is not covered by $\mathcal{W}'$, and let z be the endpoint of W in O. If ω is a vertex of $O \setminus \{z\}$, then P_2 extends a path of $\mathcal{U}'$ from x to z; then $\mathcal{U} = \mathcal{U}' \cup P_2$ covers ω, as well as all of the vertices of O covered by $\mathcal{W}$. Otherwise, we have $\omega \in V(P_1 \cup P_2)$ and $\omega \neq x$, and there is an $\omega - O$ subpath $Q \subseteq P_j$ (for one of $j = 1$ or $j = 2$). Either P_1 or P_2 will extend a path of $\mathcal{U}'$ from x to O, and Q will extend a path of $\mathcal{U}'$ from ω to O. Then, $\mathcal{U} = \mathcal{U}' \cup P_{3-j} \cup Q$ covers all of the vertices of O covered by $\mathcal{W}$, as well as the endpoint of P in O. ■

The following few sections will concern a translation of this proof to a depth-first search along alternating walks.

Translation to depth-first search

The connection between the induction proof above of Theorem 3-13 and walks which alternate with respect to $\mathcal{W}$ can be made as follows:

- If P is disjoint from $\mathcal{W}$, it (trivially) alternates with respect to $\mathcal{W}$.
- If P intersects $\mathcal{W}$ at a vertex not in O, but we have $\omega \in O \setminus \{z\}$, this corresponds to a case where an $I - O$ path is found which is totally independent (*i.e.* disjoint) from P, which may be found *e.g.* after a search from the terminal point of P has been exhausted. Similarly, if $\omega \in V(P_1) \setminus \{x\}$, this corresponds to a case where an $I - O$ path is found which intersects P_1 at some internal vertex, which may be found *e.g.* after a search through x has been exhausted.
- If P_1 intersects $\mathcal{W}$ and $\omega \in V(P_2) \setminus \{x\}$, this corresponds to a case where ω is an intermediary point of an alternating walk which backtracks (by at least one edge) along W from ω to x, and then progresses along P_1 to meet O. In this case, the induction hypothesis essentially reduces the problem to finding a vertex ω along P_2 such that there is an alternating walk with respect to $\mathcal{W}'$ from I to ω.

In any case, the alternating walk may be used to construct the new family of paths $\mathcal{U}$ incrementally, by including and excluding path segments as in the induction steps of the proof of Theorem 3-13 above. This process is illustrated in Figure 3-6.

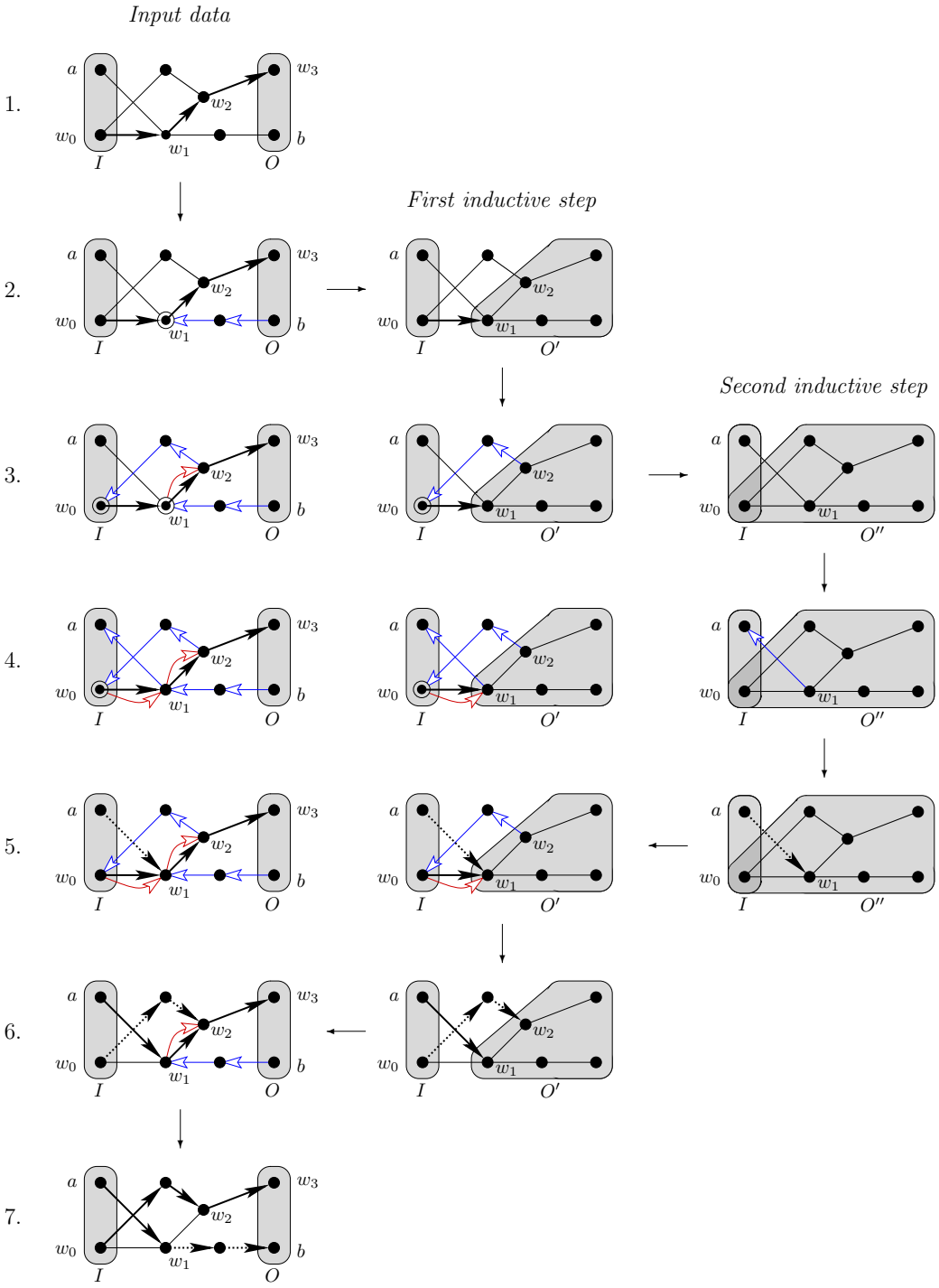


FIGURE 3-6. [*colour online*] Illustration of a depth-first search along alternating walks, based on the proof of Theorem 3-13. Each column corresponds to a successive reduction of the problem by augmenting the set O, as in the inductive step: arrows between the diagrams represent the problem reduction suggested by the proof. Steps $1-3$ consist of a search for a path P, which stops when some $a \in I$ or a pre-existing path W is encountered. This search in the reduced problems corresponds to a search along alternating walks (indicated here with hollow arrows), illustrated in the left-most column. A solution for a reduced problem is extended in the parent problem, corresponding to backtracking along the walk: this amounts to "subtracting" the alternating walk from the existing family of paths, as described in the paragraph leading up to the equation (3.10).

The fact that this walk should be an alternating walk is not the main concern, but rather that $\mathcal{U}$ can be obtained at all by searching along a walk with simple structure. We will proceed by describing the reductions that will make it easy to prove the correctness of a depth-first search.

We may refine the proof of Theorem 3-13 by extending the set $O \longmapsto O'$ regardless of whether P intersects $\mathcal{W}$. In particular: the question of whether there is an $I-O$ path $P = p_0 \cdots p_{\ell-1}p_\ell$, which is either disjoint from $\mathcal{W}$ or such that $p_j \notin V(\mathcal{W})$ for some range of consecutive vertices $r < j \leqslant \ell$, can be reduced to whether there is a similar $I-O'$ path $P' = p_0 \cdots p_{\ell-1}$, for $O' = O \cup \{p_{\ell-1}\}$. Similarly, the question of whether there is an $I-O$ path $P = p_0 \cdots xp_\ell$ such that $x \in V(\mathcal{W})$ can be reduced (as it is in the induction step of Theorem 3-13) to the question of whether there exists a path P' from I to some vertex of the set $O' = O \cup \{x, x', x'', \ldots\}$, where $xx'x'' \cdots$ is the tail of the path W in $\mathcal{W}$ which covers x; we also require that P' does not end at x itself, as this is covered by the truncated path $W' \subseteq W$ ending at x. The depth-first search may then be described in terms of the valid ways in which we may perform an augmentation $O^{(r)} \longmapsto O^{(r+1)}$, in order to reduce the search for a path $P^{(r)}$ from I to $O^{(r)}$ to a search for a path $P^{(r+1)}$ from I to $O^{(r+1)}$. To re-iterate, the valid extensions would be:

1. For any $v \in O^{(r)}$, extend to $O^{(r+1)} = O^{(r)} \cup \{w\}$ for any $w \sim v$ such that $w \notin V(\mathcal{W}^{(r)})$ and $w \notin O^{(r)}$. We then attempt to find $\mathcal{U}^{(r+1)}$ which covers w, and the terminal points of $\mathcal{W}^{(r+1)} = \mathcal{W}^{(r)}$.

2. For any $v \in O^{(r)}$, extend to $O^{(r+1)} = O^{(r)} \cup \{x, f(x), f^2(x), \ldots\}$, for any $x \in V(\mathcal{W}^{(r)}) \setminus O^{(r)}$, where f is the "successor" function mapping each vertex covered by $\mathcal{W}$ to the one which follows it (orienting the paths towards O). If W is the path of $\mathcal{W}^{(r)}$ on which x lies, let W be the segment of that path from I to x; we then attempt to find $\mathcal{U}^{(r+1)}$ which covers the terminal points of $\mathcal{W}^{(r+1)} = \mathcal{W}^{(r)} \triangle \{W, W'\}$, and one of the vertices $f^\ell(x)$ for $\ell \geqslant 1$.

The path families $\mathcal{W}^{(r)}$ we can keep track of by simply "ignoring" any arcs of $\mathcal{W}$ contained within $O^{(r)}$, as such edges are guaranteed not to be in $\mathcal{W}^{(r)}$ by construction. The extended set $O^{(r)}$ will be the original set of output vertices O together with the vertices traversed by the depth-first search to that point.

We will not keep track of of $O^{(r)}$ explicitly, but instead by "marking" most of the vertices traversed in the walk (which we may also use to avoid non-terminating recursive loops). The exceptions, those vertices which are traversed but not marked, are what we will call *entry points*. Let us call a vertex x an *entry point* of the search into a path of $\mathcal{W}$ if the edge vx is not covered by $\mathcal{W}$, where v is the previous vertex in the search. (Examples where a depth-first search encounters entry points are illustrated in Figure 3-6, where entry points are circled at each stage of the algorithm where they have been traversed only once: see *e.g.* the vertex w_1 in the left column, in steps $2-3$.) When such a vertex is discovered in the r^{th} problem reduction for some $r \geqslant 0$, the set $O^{(r)}$ is augmented to a set $O^{(r+1)} = O^{(r)} \cup \{x, f(x), \ldots\}$; however, x is then an element of $O^{(r+1)}$ covered by $W^{(r+1)}$ (specifically, by the path W truncated after x), and is not a valid vertex from which to search for an augmented family of paths $\mathcal{U}^{(r+1)}$. Nevertheless, a later problem reduction may truncate the path W' on which x lies, in which case it becomes uncovered in that problem reduction, and a valid vertex from which to later perform a depth-first search

— in particular, it may be necessary to traverse x a second time. (Figure 3-6 illustrates an example where visiting an entry point twice, w_1 in that case, is necessary to find the augmented family of paths.) Because of this special behavior of entry points, we do not mark them when they are first traversed, in order to allow a second traversal if necessary. However, we may still identify vertices of $O^{(r)}$ as vertices which are in O, vertices which are marked, or vertices covered by $\mathcal{W}$ which immediately precede marked vertices (the latter case representing entry points).

To implement a depth-first search, we will be interested in describing the problem reductions described above in terms of local transitions in the graph G. We may define

$$f(v) = \left\{ \begin{array}{ll} \texttt{nil}\,, & v \in O \text{ or } v \text{ not covered by } \mathcal{W} \\ w\,, & v \to w \text{ an arc of } \mathcal{F} \end{array} \right\}, \tag{3.13a}$$

$$g(w) = \left\{ \begin{array}{ll} \texttt{nil}\,, & w \in I \text{ or } v \text{ not covered by } \mathcal{W} \\ v\,, & v \to w \text{ an arc of } \mathcal{F} \end{array} \right\}. \tag{3.13b}$$

In a slight abuse of the original terminology, we call f the *successor function* of $\mathcal{W}$ (which will be a family of vertex-disjoint paths, albeit perhaps not a path cover); we similarly call g the *predecessor function* of $\mathcal{W}$. A transition which will be permitted for the depth-first search, in the r^{th} problem reduction, will be one from a vertex $v \in O^{(r)}$ not covered by $\mathcal{W}^{(r)}$ to a vertex w satisfying one of the following conditions:

(*i*) $v \sim w$, w is not covered by $\mathcal{W}$, and $w \notin O^{(r)}$: this corresponds to the first reduction in the discussion above. The latter two conditions are equivalent to w not covered by $\mathcal{W}^{(r)}$, and w not a marked vertex.

(*ii*) $w = f(x) \neq \texttt{nil}$ for some $x \sim v$ on some path of $\mathcal{W}$ with $x \notin O^{(r)}$, and where vx is not covered by a path of $\mathcal{W}^{(r)}$: this corresponds to the first two vertices in the second reduction in the discussion above. The conditions that x be covered by $\mathcal{W}^{(r)}$ and $x \notin O^{(r)}$ is equivalent to $w = f(x)$ being well defined (as we already require), and neither x nor w being marked.

(*iii*) $w = f(v) \neq \texttt{nil}$, and w on some path of $\mathcal{W}^{(r)}$ — in particular, where either $w \notin O^{(r)}$, or $w \in O^{(r)}$ is the vertex at the end of some path $W \in \mathcal{W}^{(r)}$: this corresponds to all but the first two vertices in the second reduction of the discussion above. In the latter case, w was an entry point of some ancestral problem reduction, in which case w is unmarked, but $f(w)$ is well defined and marked. Then, this case is equivalent to $w = f(v) \neq \texttt{nil}$ and w not a marked vertex.

In order to perform the second reduction properly, *i.e.* all at once as described, this transition must be performed preferentially when it is valid; the reduction $O^{(r)} \mapsto O^{(r+1)}$ is only complete when the endpoint of the path $W \in \mathcal{W}^{(r)}$ has been reached, and other transitions explored from that vertex.

We mark a vertex when it plays the role of w in such a local transition (in particular, leaving any entry point x initially unmarked). In each case, we need only consider the neighbors $z \sim v$ which have not yet been marked, such that either $z \notin O$ or $z = f(v)$; in the case where z is an entry point, we further require that $w = f(z)$ has not been marked. Then, in the depth-first search, it will suffice at each step to explore the unmarked vertices z such that either $z = f(v) \neq \texttt{nil}$, or such that $z \sim v$ and $z \notin O$.

Input : G, a graph with subsets I, O.
Output: successor/predecessor functions $f, g : V(G) \longrightarrow V(G)$ for a maximum-size family $\mathcal{P}$ of vertex-disjoint $I-O$ paths

```
1  procedure ObtainMaxFamilyPaths(G, I, O):
2  begin
3      let f, g : V(G) ⟶ V(G);
4      let marked : V(G) ⟶ ℕ;
5      let iter ← 0;
6      for all v ∈ V(G) do
7          marked(v) ← 0;
8          f(v) ← nil;  g(v) ← nil;
9      endfor
10     repeat
11         iter++;
12         let foundpath ← false;
13         for all v ∈ O such that v ∉ I and g(v) = nil do
14             AlternPathSearch(v);
15             if foundpath then escape this loop;
16         endfor
17     until foundpath = false ;
18     return (f, g)
19 end
```

ALGORITHM 3-1. An iterative algorithm to obtain a maximum-size family of vertex-disjoint $I-O$ paths. The subroutine `AlternPathSearch` is described in Algorithm 3-2 on page 109.

Algorithms 3-1 and 3-2 together describe an iterative procedure to build a maximum-size family of vertex-disjoint paths, based on this reduction to a depth-first search; where sets of disjoint paths $\mathcal{W}$ are represented by successor/predecessor functions and set consisting of those vertices not covered by $\mathcal{W}$. In the following pages, we use the reduction just described to prove their correctness.

Overview of Algorithm 3-1

The procedure `ObtainMaxFamilyPaths` described in Algorithm 3-1 above uses a subroutine `AlternPathSearch` (described in Algorithm 3-2, on page 109) to obtain larger and larger families of vertex disjoint paths, until a maximum size family is found. Before turning to the description of the more detailed subroutine `AlternPathSearch`, we will describe Algorithm 3-1 as a whole.

As noted before, we will represent a family of vertex-disjoint paths $\mathcal{W}$ using partial functions f and g, which will store the successor and predecessor functions of $\mathcal{W}$. In the loop on lines $6-9$, the partial functions are initialized to represent a family of paths consisting only of trivial paths $v \in I \cap O$ (without loss of generality, we may assume that $\mathcal{W}$ contains the trivial paths on whatever elements there may be in $I \cap O$): we

will consider a vertex v to be covered by $\mathcal{W}$ if and only if at least one of $v \in I \cap O$, $f(v) \neq \texttt{nil}$, or $g(v) \neq \texttt{nil}$ holds. Also defined and initialized are an array `marked` and a counter `iter`: as we construct larger families of paths, `iter` will represent the iteration of the construction, and `marked` will keep track of the most recent iteration that each vertex was marked in the depth-first search along alternating walks. These variables will all be used in the subroutine `AlternPathSearch` as global variables, all of which (except for `iter`) may be modified.

The iterative component of `ObtainMaxFamilyPaths` increments `iter`, and defines a flag `foundpath` which will indicate whether the pre-existing family of paths $\mathcal{W}$ represented by (f, g) has been successfully augmented; this is also a global parameter which may be modified by `AlternPathSearch`. The loop on lines 13 – 16 iterates through all of the elements of O which are not covered by $\mathcal{W}$. We will maintain the constraint that $f(v) = \texttt{nil}$ for all $v \in O$: then, $v \in O$ is covered by $\mathcal{W}$ if and only if $g(v) \neq \texttt{nil}$ or $g(v) \in I$.

We will later show that so long as `foundpath` = `false`, the elements of O which have been marked are those for which `AlternPathSearch` has been invoked, but has failed to find a path P similar to that in the proof of Theorem 3-13. We will also show that, once an augmenting walk is found, `AlternPathSearch` updates (f, g) to represent a collection of vertex-disjoint paths $\mathcal{U}$ which covers the terminal points of $\mathcal{W}$ and one additional vertex in O, and that `foundpath` is set to `true`; we may then progress to the next iteration of the construction. If `foundpath` retains the value `false`, the variables (f, g) will remain unchanged since the last iteration. Finally, if all uncovered elements of O are visited in an iteration without the value of `foundpath` changing, the pre-existing path family $\mathcal{W}$ is of maximum size, and the variables (f, g) representing it are returned.

Having described the data provided to, and the requirements made of, the subroutine `AlternPathSearch`, we will turn to the description of it in Algorithm 3-2.

Analysis of Algorithm 3-2

The subroutine `AlternPathSearch` is described in Algorithm 3-2 (page 109). We will analyze it in terms of the problem reductions which we derived on page 106 from the proof of Theorem 3-13. In particular, we will consider an invocation of `AlternPathSearch` on a vertex v, representing the r^{th} problem reduction (or part of it) for some $r \geqslant 0$. (Note that even if $v \in O$ is not covered by $\mathcal{W}$, we may have $r > 0$, if v is not the first such vertex for which `AlternPathSearch` has been invoked in a given iteration.)

Recall that a vertex z is marked iff $\texttt{marked}(z) = \texttt{iter}$: we will assume as in the analysis on page 106 that $O^{(r)}$ consists of vertices z which are marked, vertices x for which $f(x)$ is non-`nil` and marked, and the elements of O. A truncated family of paths $\mathcal{W}^{(r)}$ then corresponds to those vertices covered by $\mathcal{W}$ which are not marked (and the edges covered by $\mathcal{W}$ incident to those vertices). Equivalently, we will say that the variables (f, g) represent a family of vertex disjoint paths $\mathcal{W}^{(r)}$ from I to $O^{(r)}$ if the following hold:

(*i*) the vertices covered by $\mathcal{W}^{(r)}$ are those which satisfy at least one of $v \in I \cap O^{(r)}$, $\left[g(v) \neq \texttt{nil} \text{ and } g(v) \notin O^{(r)}\right]$, or $\left[f(v) \neq \texttt{nil} \text{ and } v \notin O^{(r)}\right]$;

(*ii*) $\mathcal{W}$ covers every edge vw for which $v \notin O^{(r)}$, $w = f(v)$, and $v = g(w)$.

```
    Data : G, a graph with subsets I, O;
           f and g, successor/predecessor f^ns for a family W of disjoint I – O paths;
           iter, the current iteration number of path-family augmentations;
           foundpath, a flag for indicating if an augmenting path was found.
 1  subroutine AlternPathSearch(v):
 2  begin
 3      marked(v) ← iter;
 4      if v ∈ I then
 5          foundpath ← true;
 6          return
 7      endif
 8      let w ← f(v);
 9      if w ≠ nil and marked(w) < iter then
10          AlternPathSearch(w);
11          if foundpath then
12              f(v) ← nil;  g(v) ← nil;
13              return
14          endif
15      endif
16      for each w ∼ v such that w ∉ O and marked(w) < iter do
17          if f(w) = nil then
18              AlternPathSearch(w);
19              if foundpath then
20                  f(w) ← v;  g(v) ← w;
21                  return
22              endif
23          else
24              let x ← w;  w ← f(x);
25              if marked(w) < iter then
26                  AlternPathSearch(w);
27                  if foundpath then
28                      f(x) ← v;  g(v) ← x;
29                      return
30                  endif
31              endif
32          endif
33      endfor
34      return
35  end
```

ALGORITHM 3-2. A depth-first search subroutine to find an alternating walk (with respect to a family of disjoint paths $\mathcal{W}$) from I to O, traversed in reverse.

(The same variables (f, g) may represent multiple families of paths from I to $O^{(r)}$, corresponding to different levels r of problem reduction.)

The first action performed by `AlternPathSearch` is to set `marked`$(v) \leftarrow$ `iter`, corresponding to the inclusion of v into $O^{(r)}$ (and if $g(v) \neq$ `nil`, the inclusion of that vertex into $O^{(r)}$ as well). If it happens that $v \in I$, then the trivial path on v is a path from I to $O^{(r)}$ which is disjoint from the paths of $\mathcal{W}^{(r)}$; we may then define $\mathcal{U}^{(r)} = \mathcal{W}^{(r)} \cup \{v\}$, which covers v and the terminal points of $\mathcal{W}^{(r)}$, and which will already be represented by (f, g). We then set `foundpath` $\leftarrow$ `true`, and return to the previous recursive call.

If $v \notin I$, we then test on lines 8 – 9 whether if v is on a path and has an unmarked successor $w = f(v)$. If so, we are in the midst of a problem reduction of the form $O^{(r-1)} \mapsto O^{(r-1)} \cup \{x, f(x), f^2(x), \ldots\}$ for some preceding vertex x on the same path as v. To complete the reduction, we must accumulate any unmarked vertices $w, f(w), \ldots$ that there may be, and perform the depth-first search through them as we mark them for inclusion. To do this, we recur the depth-first search at $w = f(v)$.

We will consider the effects of the conditional block starting on line 11 further on: in any case, we regard all the operations from line 11 onwards as being part of an r^{th}-level problem reduction for some $r \geqslant 0$, and the state of `marked` at this point to define $O^{(r)}$, any further changes made to `marked` by further recursive calls notwithstanding.

We require that at line 11, the variables (f, g) represent a family of vertex disjoint paths $\mathcal{W}^{(r)}$, and that $v \in O^{(r)}$ is not covered by $\mathcal{W}^{(r)}$. (In particular, this will be true when $v \in O$ is uncovered by $\mathcal{W}$ and is the only vertex in $V(G)$ which is marked, *i.e.* for $r = 0$). We will then prove by induction that, after an invocation of `AlternPathSearch`(v) for v not covered by $\mathcal{W}^{(r)}$, the following hold if `foundpath` evaluates to `true`:

(i) The variables (f, g) represent a family of vertex-disjoint paths $\mathcal{U}^{(r)}$ from I to $O^{(r)}$, which covers the terminal points of $\mathcal{W}^{(r)}$ and one other vertex;

(ii) If v is not covered by $\mathcal{W}$, then the additional vertex covered by $\mathcal{U}^{(r)}$ is v, and $f(v)$ will remain unchanged;

(iii) The variables f and g may only change for vertices $z \in V(G) \setminus O^{(r-1)}$ (or for $z \in V(G)$ when $r = 0$); and they will remain unchanged for any vertex for which a recursive call `AlternPathSearch`(z) returns with `foundpath` $=$ `false`.

Note that if (f, g) represents a family of vertex disjoint paths $\mathcal{W}^{(r)}$ which does not cover $v \in I$ at line 4, the above conditions are satisfied; this forms the base case of the induction.

As we noted on page 106, for variables (f, g) representing a family of paths $\mathcal{W}^{(r)}$ from I to $O^{(r)}$, and for a vertex $v \in O^{(r)}$ which is not covered by $\mathcal{W}^{(r)}$, the possible problem reductions can be obtained by considering the vertices which are iterated through on line 16 (of which there are at most $\deg(v)$ in number, and which can easily be iterated over provided an adjacency list representation of G). We will now consider the reductions performed in the loop on lines 16 – 33, together with the conditional block at line 11:

- For a vertex w for which $f(w) =$ `nil` (on line 17): we may attempt a problem reduction by defining $O^{(r+1)} = O^{(r)} \cup \{w\}$: to do this, we recur the depth-first search at w. If `foundpath` $=$ `true` after this reduction attempt, (f, g) represents

a new family of paths $\mathcal{U}^{(r+1)}$ from I to $O^{(r+1)}$ covering w and the terminal points of $\mathcal{W}^{(r+1)}$. Let $W' \in \mathcal{U}^{(r+1)}$ be the path covering w: we may obtain a path W covering v by extending W' by the edge wv. As v is not covered by $\mathcal{W}^{(r+1)}$, it is not covered by $\mathcal{U}^{(r+1)}$, and so $\mathcal{U}^{(r)} = \mathcal{U}^{(r+1)} \triangle \{W', W\}$ is a vertex-disjoint family of paths covering v and the terminal points of $\mathcal{W}^{(r)}$. To represent $\mathcal{U}^{(r)}$, we set $f(w) \leftarrow v$ and $g(v) \leftarrow w$. The only vertices for which (f, g) are changed are then v and those changed by the recursive call, which consist of w and some vertices $z \in V(G) \setminus O^{(r+1)} \subseteq V(G) \setminus O^{(r-1)}$.

- For a vertex $w = f(x)$ (on line 11): as noted before, if w is well-defined, then we must perform a recursive call on w to complete a reduction of the form $O^{(r+1)} = O^{(r)} \cup \{x, f(x), \ldots\}$ for some x preceding v and w on a common path $P \in \mathcal{W}^{(r)}$. In particular, such an x is by definition an entry point, and as we will see in the next case, `AlternPathSearch` is not invoked on vertices which occur as entry points; then v is distinct from x, and is not the end point of any path of $\mathcal{W}^{(r+1)}$.

  If `foundpath` = `true` after this reduction attempt, (f, g) represents a new family of paths $\mathcal{U}^{(r+1)}$ from I to $O^{(r+1)}$ covering the terminal points of $\mathcal{W}^{(r+1)}$, and one more vertex ω on the segment of P starting at w (in particular, we have $\omega \neq v$). Let $W' \in \mathcal{U}^{(r+1)}$ be the path ending at ω, and let P', P'', and P''' be the segment of W ending at x, the segment of W strictly bounded between x and ω (and not including those endpoints[11]), and the segment of W starting at ω respectively. By definition, ω is the first vertex from the end of W for which the recursive call to `AlternPathSearch` returned with `foundpath` = `true`; in particular, for every vertex $z \in V(P''' \setminus \omega)$, the recursive call returned with `foundpath` = `false`. Then, the status of such vertices z remain unchanged with respect to (f, g, S): for each such vertex, we have $f(g(z)) = z$, and $g(f(z)) = z$ if $f(z) \neq$ `nil`. As well, by the analysis for the preceding case, the value of $g(\omega)$ will have changed to the vertex $\tilde{w}$ which precedes ω on W', and $f(\omega)$ will remain unchanged.

  The reduction described in the proof of Theorem 3-13 involves constructing a family of paths $\mathcal{U}^{(r)}$ by extending the path W' from I to $O^{(r+1)}$ to a path $W = W'P'''$ from I to $O^{(r)}$, and extending the path P' from I to $O^{(r+1)}$ by a single edge (as x must in this case be adjacent to a vertex $\tilde{v} \in O^{(r)}$ which is not covered by $\mathcal{W}^{(r)}$, from which x was reached in the depth-first search). In particular, the vertices of P'' are not covered by $\mathcal{U}^{(r+1)}$, and are not elements of $O^{(r)}$. In order to represent $\mathcal{U}^{(r)}$, it is then necessary to remove the vertices $z \in V(P'')$ from the domains of f and g. Therefore, we set $f(v) \leftarrow$ `nil` and $g(v) \leftarrow$ `nil`; by induction on the length of P'', this will then be performed for all vertices in $V(P'')$. The only vertices for which (f, g) have changed are then ω, the vertices of P'' from v onwards, and some vertices $z \in V(G) \setminus O^{(r)} \subseteq V(G) \setminus O^{(r-1)}$.

- For a vertex w for which $f(x) \neq$ `nil` (on line 23): because of the recursive call in the conditional block on line 11, the vertex $f(v)$ has already been marked, if it is defined; because w is unmarked, we know that $w \neq f(v)$. Then either w is an entry point, or $w = g(v)$. We relabel the possible entry point w as x, overwrite $w \leftarrow f(x)$, and test whether $w = f(x)$ has been marked. If not, then we have $x \neq g(v)$, as $f(g(v)) = v$ is a marked vertex; in this case, x is indeed an entry point. We may

[11] Note that the path P'' may be an "empty path", consisting of zero vertices.

then attempt a problem reduction by defining $O^{(r+1)} = O^{(r)} \cup \{x, f(x), f^2(x), \ldots\}$: to do this, we recur the depth-first search at w. (Marking w will implicitly include both $x, w \in O^{(r+1)}$; the test at line 11 ensures that all further vertices along the same path, up to its endpoint in $O^{(r)}$, will be immediately included as well.)

If `foundpath` = `true` after this reduction attempt, (f, g) represents a new family of paths $\mathcal{U}^{(r+1)}$ from I to $O^{(r+1)}$ covering the endpoints of $\mathcal{W}^{(r+1)}$, which include x, and one more vertex $f^{\ell}(x)$ for some $\ell \geqslant 1$. By the analysis of the preceding case, the variables (f, g) also represent a family of paths $\mathcal{U}'$ from I to $O^{(r)} \cup \{x\}$, covering the terminal points of $\mathcal{W}^{(r)}$ as well as the additional vertex x. Let $W' \in \mathcal{U}'$ be the path ending at x: we may obtain a path W covering v by extending W' by the edge xv. As v is not covered by either $\mathcal{U}^{(r+1)}$ or $\mathcal{W}^{(r)}$ (and in particular not by the path $P \in \mathcal{W}^{(r)}$ which covers x), $\mathcal{U}^{(r)} = \mathcal{U}' \triangle \{W, W'\}$ is a vertex-disjoint collection of paths covering v and the terminal points of $\mathcal{W}^{(r)}$. To represent $\mathcal{U}^{(r)}$, we set $f(x) \leftarrow v$ and $g(v) \leftarrow x$. The only vertices for which (f, g) are changed are then v, the vertices $O^{(r+1)} \setminus O^{(r)} = \{x, f(x), f^2(x), \ldots\}$ for which the recursive call to `AlternPathSearch` returned with `foundpath` = `true`, and some vertices $z \in V(G) \setminus O^{(r+1)} \subseteq V(G) \setminus O^{(r-1)}$.

It is easy to verify that if the preconditions are satisfied that (f, g) represent a family of vertex disjoint paths $\mathcal{W}^{(r)}$ from I to $O^{(r)}$, and that $v \in O^{(r)}$ is not the endpoint of such a path, that they are also satisfied for each recursive call. Then, by induction, if initially the variables (f, g) represents a family $\mathcal{W}^{(r)}$ of vertex disjoint $I - O^{(r)}$ paths, $v \in O$ is not covered by $\mathcal{W}^{(r)}$, and `AlternPathSearch`(v) returns with `foundpath` = `true`, then afterwards (f, g) represents a family $\mathcal{U}$ of vertex-disjoint $I - O^{(r)}$ paths covering v and the terminal points of $\mathcal{U}$. If such a collection of paths $\mathcal{U}$ exists, then it is the only family of vertex disjoint $I - O$ paths larger than $\mathcal{W}$ in the a graph $G \setminus S$ where we define

$$S = \left\{ z \in O^{(r)} \;\middle|\; \Big(g(z) = \texttt{nil} \text{ or } g(z) \in O^{(r)} \Big) \text{ and } z \neq x \right\}, \tag{3.14}$$

i.e. where we remove every element of $O^{(r)}$ not covered by $\mathcal{W}$ *except* for x. Because Algorithm 3-2 reduces this problem to finding an appropriate family of paths ending in at least one of its neighbors, Theorem 3-13 guarantees that `AlternPathSearch`(v) will terminate with `foundpath` = `true` in the case that such a family of paths $\mathcal{U}$ does exist.

Analysis of Algorithm 3-1

Because a call to `AlternPathSearch`(v) returns with `foundpath` = `true` if and only if either $v \in I$ or a recursive call returns with `foundpath` = `true`, and because (f, g) are only changed when such a recursive call returns with `foundpath` = `true`, we know that they will be unchanged if `AlternPathSearch`(v) returns with `foundpath` = `false`. If this occurs for an invocation `AlternPathSearch`(v) where initially no vertices were marked, the loop on lines 13 – 16 of Algorithm 3-1 will select another element $v' \in O$ such that $g(v') =$ `nil`, and invoke `AlternPathSearch`(v'). In this case, the data (f, g) represent $\mathcal{W}$ as a family of paths from I to O', for some superset $O' \supseteq O$ such that, for all $w \in O' \setminus O$, w is not itself the endpoint of any family of $I - O'$ vertex-disjoint paths $\mathcal{U}'$ which also covers the terminal points of $\mathcal{W}$. Then, any attempt to reduce the problem to finding such a family of paths for $w \in O' \setminus O$ will fail, and is therefore unnecessary. Therefore, by

induction, subsequent invocations of `AlternPathSearch`(v') in the loop on lines 13 – 16 of Algorithm 3-1 will return with `foundpath` = `true` if and only if there exists a family $\mathcal{U}'$ of vertex disjoint $I-O$ paths which covers v' as well as the terminal points of $\mathcal{W}$; and (f,g) represents this family of paths.

In each repetition of this loop, `AlternPathSearch` is invoked at most once on each vertex $v \in V(G)$ in the depth-first search. Because the assignments and conditional tests for each vertex can be performed in time $O(1)$, the time consumed by each invocation (not including the time consumed by recursive invocations) is dominated by the time required to iterate through the neighbors of v on lines 16 – 33 of Algorithm 3-2, of which there are $\deg(v)$. Summing over all vertices, the time required to perform one iteration of the "while" loop on lines 10 – 17 of Algorithm 3-1 is then

$$O\left(\sum_{v\in V(G)} \deg(v)\right) \;=\; O\Big(2\,|E(G)|\Big) \;=\; O(m)\,. \tag{3.15}$$

Because there are at most k vertices in O, there can be at most k vertex disjoint paths from I to O; then, at most k iterations of the "while" loop on lines 10 – 17 of Algorithm 3-1 can terminate with `foundpath` = `true`; then this loop will terminate in time $O(km)$. By induction, regardless of the value of `foundpath`, the variables (f,g) represent a family of vertex disjoint paths $\mathcal{W}$ from I to O after each such iteration; and by the analysis above, we have `foundpath` = `false` only if there does not exist a family of vertex disjoint $I-O$ paths in G which is larger than $\mathcal{W}$. The run-time of Algorithm 3-1 is dominated by the "while" loop, as the initialization loop can be performed in time $O(n)$. Therefore:

Lemma 3-14. *Let (G,I,O) be a geometry with $|E(G)| = m$ and $|O| = k$. Then the procedure* `ObtainMaxFamilyPaths`(G,I,O) *terminates in time $O(km)$, and returns data (f,g) such that there exists a maximum-size family $\mathcal{W}$ of vertex disjoint $I-O$ paths with the following properties:*

(i) $\mathrm{dom}(f) = V(\mathcal{W}) \setminus O$, *and* $\mathrm{dom}(g) = V(\mathcal{W}) \setminus I$;

(ii) $vw \in E(G)$ *is covered by* $\mathcal{W}$ *if and only if* $v \in \mathrm{dom}(f)$, $w \in \mathrm{dom}(g)$, $w = f(v)$, *and* $v = g(w)$.

Remark on this algorithm and its run-time. It should be noted that the algorithms presented above are essentially an adaptation of the Ford-Fulkerson algorithm for Max-Flow,[12] for a flow network with multiple indistinguishable sources and sinks, and unit edge capacities. For such a flow-network, a solution to Max Flow is a maximum-size collection of *edge*-disjoint $I-O$ paths: the adaptation made here is the special treatment of entry points treated as an added special case, to ensure *vertex*-disjointness of the families of paths. Thus, the run-time of $O(km)$ should be unsurprising, as this is also an upper found for the Ford-Fulkerson algorithm for a flow-network whose maximum integer flow has value k.

[12] For more information about the Max-Flow problem and the Ford-Fulkerson algorithm to solve it, the interested reader may consult any standard text on polynomial time graph algorithms, *e.g.* [33]. The term "flow" in this context is not directly related to the definition used throughout this chapter.

Determining if we have found a successor function

The final step towards finding a potential flow-function (*i.e.* a successor function in the sense of Definition 3-3, to which we will again restrict ourselves to for the remainder of the chapter) is to check if the family of paths $\mathcal{W}$ described by the variables (f, g) constructed by `ObtainMaxFamilyPaths`(G, I, O) is in fact a path cover. We may do this by simply verifying that

$$\mathrm{dom}(f) \cup \mathrm{dom}(g) \cup (I \cap O) = V(G)\,, \tag{3.16}$$

that is, that there are no vertices $v \in I^{\mathsf{c}} \cup O^{\mathsf{c}}$ for which $f(v) = g(v) = \texttt{nil}$, as the set of vertices which would lie outside of both the domain and the image of a successor function is precisely $I \cap O$. We can test whether there are any such vertices v in time $O(n)$.

By Theorem 3-11, if (G, I, O) is a geometry where $|I| = |O|$ and which has a causal path cover $\mathcal{C}$, then that path cover is the unique maximum family of vertex-disjoint $I - O$ paths. Then, if the family of paths $\mathcal{W}$ represented by the variables (f, g) is not a path cover, the geometry (G, I, O) does not have a flow. We may then reject any geometries for which this occurs.

3.4.3 Constructing the natural pre-order for a successor function

As we remarked in Section 3.3.3, the problem of determining if a successor function is a flow function can be reduced to the problem of constructing the natural pre-order $\preccurlyeq$ for f. The natural pre-order is the transitive and reflexive closure of the influence relation $\lhd$ for f defined by Definition 3-4: thus, we may reduce finding $\preccurlyeq$ to the Transitive Closure problem. In this section, we will describe algorithms for doing this which take advantage of the addition structure provided by the presence of the successor function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ (and its inverse function $g : I^{\mathsf{c}} \longrightarrow O^{\mathsf{c}}$) based on techniques presented in Nuutila [106].

Adapting Tarjan's Algorithm to find the natural pre-order

Tarjan's algorithm [127, 33] is a depth-first search algorithm for determining the strongly connected components of a digraph D, which are the equivalence classes of vertices which are mutually reachable from each other by directed walks. A straightforward adaptation of Tarjan's algorithm due to [123] can be used to solve the Transitive Closure problem by storing the set of vertices which can be reached from each vertex (which we refer to as the *descendants* of a vertex), thus determining the transitive closure of the digraph D. A clear presentation of such an algorithm can be found in [106, page 49].

In order to efficiently store the sets of descendants of a given vertex, we may make use of a *chain decomposition* of the digraph D, a technique proposed in [122]. A chain decomposition of a digraph D is a decomposition of $V(D)$ into a family of vertex-disjoint directed paths $\mathcal{P}$ in the digraph D. Then, for each vertex v and each path $P \in \mathcal{P}$, we may store the minimal vertex $w \in V(P)$ such that w is a descendant of v. If it is easy to determine when one vertex within a path P is a descendant of another, we may then use this to obtain an efficient method to make reachability comparisons between arbitrary vertices.

We may adopt these techniques to efficiently obtain a representation of the natural pre-order $\preccurlyeq$ of the successor function f. Described in terms of directed graphs, the relevant problem is to find the (reflexive and) transitive closure of the influence digraph $\mathfrak{I}_f$ (Definition 3-6, page 90). Note that for each $v \in O^{\mathsf{c}}$ and $w = f(v)$, the arc $v \to w$ is an arc of $\mathfrak{I}_f$: then, the path cover $\mathcal{P}_f$ is a chain decomposition of $\mathfrak{I}_f$. Thus, if functions f and g have been computed, we have a ready-made chain decomposition of $\mathfrak{I}_f$ which we may exploit. As well, if $\mathfrak{I}_f$ contains directed circuits (*i.e.* if it has strongly connected components consisting of more than one vertex), then the pre-order $\preccurlyeq$ is not antisymmetric: rather than storing strongly connected components, as in Tarjan's algorithm, we may simply determine whether we have found a directed circuit in $\mathfrak{I}_f$ and terminate once one is found.

The way in which the natural pre-order $\preccurlyeq$ will be constructed is as follows. We will construct and array $\texttt{Path} : V(G) \longrightarrow \mathbb{N}$ which assigns a "path label" to each vertex which is constant along paths, and distinct between distinct paths, of $\mathcal{P}_f$. For each vertex $v \in V(G)$, we will also store a distance $\texttt{Dist}(v)$ which measures the distance of each vertex from the initial point of the path on which it lies: more precisely, to the largest integer $\ell \in \mathbb{N}$ such that $g^{\ell}(v) \neq \texttt{nil}$. Thus, for vertices v, w on a common path, we have $v \preccurlyeq w$ if and only if $\texttt{Dist}(v) \leqslant \texttt{Dist}(w)$. We will call a pair of such arrays $(\texttt{Dist}, \texttt{Path})$ a *path-labelling* of the orbits of f.

To exploit the chain decomposition to efficiently store the partial order $\preccurlyeq$, we will construct an array of lower bounds for each vertex, which we will use to store the "ancestors" rather than the descendants of a vertex (*i.e.* for a vertex w, the vertices v from which w can be reached by directed walks). The *infimum* $\inf_P(w)$ of a vertex w in a path $P \in \mathcal{P}_f$ will simply be the maximal vertex v in $\preccurlyeq$ such that $v \in V(P)$ and $v \preccurlyeq w$. Rather than explicitly storing the vertices $\inf_P(w)$, we may instead store the distance of $\inf_P(w)$ from the initial point on its path. Then, if we label the paths of $\mathcal{P}_f$ arbitrarily by integers $p \in \{1, \ldots, k\}$,[13] we may record the infima of vertices $w \in V(G)$ in an array $\texttt{Inf} : V(G) \times \{1, \ldots, k\} \longrightarrow \mathbb{Z}$ such that $\texttt{Inf}(w, p)$ is the maximum distance of a vertex v such that $\texttt{Path}(v) = p$ from the initial point of its path, such that $v \preccurlyeq w$. (If there is no such vertex v, we may set $\texttt{Inf}(w, p)$ to some negative integer.) This suggests the following definition:

Definition 3-13. For a geometry (G, I, O) with a successor function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, a *system of infima* for a pre-order $\preccurlyeq$ on $V(G)$ is a triple $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ of arrays $\texttt{Path}, \texttt{Dist} : V(G) \longrightarrow \mathbb{N}$ and $\texttt{Inf} : V(G) \times \mathbb{N} \longrightarrow \mathbb{Z}$ such that

$$v \preccurlyeq w \iff \texttt{Dist}(v) \leqslant \texttt{Inf}(w, \texttt{Path}(v)) \tag{3.17}$$

holds for all $v, w \in V(G)$.

Algorithms 3-3 and 3-4 together describe an iterative procedure for constructing a system of infima for $\preccurlyeq$ as described above, and in doing so, test whether the natural pre-order $\preccurlyeq$ is antisymmetric. In the following pages, we will prove their correctness.

[13] Recall that each path of $\mathcal{P}_f$ ends in O and each vertex of O ends such a path: thus $|\mathcal{P}_f| = |O| = k$.

Data : (G, I, O), a geometry with $|O| = k$
$f : V(G) \longrightarrow V(G)$, a successor function for (G, I, O)
$g : V(G) \longrightarrow V(G)$, the inverse function of f
an array $\texttt{status} : V(G) \longrightarrow \{\texttt{untouched}, \texttt{incomplete}, \texttt{complete}\}$
$\texttt{Inf} : V(G) \times \{1, \ldots, k\} \longrightarrow \mathbb{Z}$, an array to store path-infima for vertices

```
1  subroutine FindInfima(w):
2  begin
3      if status(w) = incomplete then foundcircuit = true;
4      if status(w) ≠ untouched then return ;
5      status(w) ← incomplete;
6      let v ∈ V(G);
7      for each z ∼ w such that z ≠ f(w) and (z = g(w) or g(z) ≠ nil) do
8          if z = g(w)
9              then v ← z ;
10             else  v ← g(z) ;
11         FindInfima(v);
12         if foundcircuit then return ;
13         for all p ∈ {1, ..., k} do
14             if Inf(w, p) < Inf(v, p) then Inf(w, p) ← Inf(v, p);
15     endfor
16     status(w) ← complete;
17     return
18 end
```

ALGORITHM 3-3. An algorithm to obtain the "ancestors" of a vertex w (and by extension of all vertices $v \prec w$), and to determine in the process whether $\prec$ is antisymmetric.

Analysis of Algorithm 3-3

Algorithm 3-3 describes a recursive subroutine `FindInfima`, with implicit inputs defined by an embedding procedure shown in Algorithm 3-4 (page 122). Before describing the more general algorithm there, we will first describe behavior of `FindInfima`.

Suppose that $(\texttt{Dist}, \texttt{Path})$ is a path-labelling of the orbits of f. We will say that w is *untouched* with respect to a function $\texttt{Inf} : V(G) \times \{1, \ldots, k\} \longrightarrow \mathbb{N}$ if we have $\texttt{status}(w) = \texttt{untouched}$ and

$$\texttt{Inf}(w, p) = \left\{ \begin{array}{cc} \texttt{Dist}(w)\,, & \text{if } p = \texttt{Path}(w) \\ -1\,, & \text{otherwise} \end{array} \right\} \tag{3.18}$$

for all $p \in \{1, \ldots, k\}$; and that w is *complete* with respect to `Inf` if for all $v \preccurlyeq w$, the following holds:

(*i*) $\texttt{status}(v) = \texttt{complete}$,

(*ii*) $\texttt{Inf}(v, p) \geqslant -1$ for all $p \in \{1, \ldots, k\}$, and

(*iii*) $u \preccurlyeq v \iff \texttt{Dist}(u) \leqslant \texttt{Inf}(v, \texttt{Path}(u))$ for all $u \in V(G)$.

For any vertex w which is complete with respect to `Inf`, we also have v complete with respect to `Inf` whenever $v \preccurlyeq w$: this represents a subset of $(v, w) \in V(G) \times V(G)$ where the relation $v \preccurlyeq w$ is faithfully represented by a comparison as in 3.17. We will be interested in situations where `FindInfima` is called for an untouched vertex, conditioned on subsets of $V(G)$ being either untouched or complete with respect to `Inf`.

Note that the instructions of the loop on lines 7 – 15 are executed for precisely those vertices $z = f(v) \neq w$ and $z = v = g(w)$, where $v \lhd w$; and on lines 9 and 10, we assign to the *variable* v that corresponding vertex $v \lhd w$. Then, the iteration over z on line 7 essentially stands in for an iteration over all vertices $v \lhd w$. We will use this fact without further comment throughout our analysis of `FindInfima`.

Next, note that in any invocation of $\texttt{FindInfima}(w)$ terminates with $\texttt{foundcircuit} = \texttt{false}$ only if either $\texttt{foundcircuit} = \texttt{false}$ and $\texttt{status}(w) = \texttt{complete}$ before the invocation, or if the program control proceeds beyond for loop on lines 7 – 15. The value of $\texttt{Inf}(w, p)$ can only change for some value of $p \in \{1, \ldots, k\}$ in the latter case, in which case the value of $\texttt{status}(w)$ changes from `untouched` to `complete` in the course of the invocation. Furthermore, this will only happen if every recursive call to $\texttt{FindInfima}(v)$ made also returns with $\texttt{foundcircuit} = \texttt{false}$, in which case the same is true of all $v \lhd w$. Thus, an invocation of $\texttt{FindInfima}(w)$ terminates with $\texttt{foundcircuit} = \texttt{false}$ only if:

(*i*) $\texttt{foundcircuit} = \texttt{false}$ before the invocation;

(*ii*) for any $v \in V(G)$ the value of $\texttt{Inf}(v, p)$ is only changed for some $p \in \{1, \ldots, k\}$ if $\texttt{status}(v)$ is also changed; and

(*iii*) `status` is changed by the invocation only in that $\texttt{status}(v)$ is set to `complete` for some set of vertices $v \preccurlyeq w$, including w in particular.

Using this, we will consider certain conditions under which an invocation $\texttt{FindInfima}(w)$ will return with $\texttt{foundcircuit} = \texttt{true}$.

Definition 3-14. We call a set of vertices S *noxious* if, when $\texttt{status}(v) = \texttt{untouched}$ or $\texttt{status}(v) = \texttt{complete}$ for every vertex $v \in V(G) \setminus S$, an invocation of $\texttt{FindInfima}(s)$ will terminate with $\texttt{foundcircuit} = \texttt{true}$ for any $s \in S$.

Observation. Suppose that $\texttt{status}(v) = \texttt{incomplete} \iff v \in S$ for some set $S \subseteq V(G)$. Then S is a noxious set, as an invocation of $\texttt{FindInfima}(v)$ for any $v \in S$ will terminate with $\texttt{foundcircuit} = \texttt{true}$.

The concept of a vertex in a noxious set is meant as a generalization of the case above, of a vertex $v \in V(G)$ for which $\texttt{status}(v) = \texttt{incomplete}$. We will apply this concept ultimately to demonstrate simple conditions on `status` under which a circuit in $\mathfrak{I}_f$ will cause an invocation of `FindInfima` to return with $\texttt{foundcircuit} = \texttt{true}$. The main tool in this approach is the following Lemma:

Lemma 3-15. *Let $S \subseteq V(G)$ be a noxious set defined solely in terms of the properties of vertices with respect to adjacency in G, and the arrays* `status`*, f, and g. Suppose that $w \in V(G)$ is such that for all $v \preccurlyeq w$, either* `status`$(v) =$ `untouched`*,* `status`$(v) =$ `complete`*, or $v \in S$; and that there is some $s \in S$ such that $s \preccurlyeq w$. Then an invocation of* `FindInfima`(w) *will terminate with* `foundcircuit` $=$ `true`*.*

Proof —

For any such w, let S' be the subset of vertices $s \in S$ such that there is a chain

$$s \;=\; u_0 \lhd u_1 \lhd \cdots \lhd u_d = w \tag{3.19}$$

for some $d \geqslant 0$, such that $u_j \notin S$ for all $j > 0$; and let $\ell(w)$ be the length of the longest such chain. (We may extend this to vertices w for which there are no vertices $s \preccurlyeq w$, by adopting the convention $\ell(w) = \infty \geqslant N$ for any $N \in \mathbb{N}$.) If $\ell(w) = 0$, we have $w \in S$, in which case an invocation of `FindInfima`(w) will terminate with `foundcircuit` $=$ `true` by definition. Otherwise, suppose the claim holds for all such vertices $v \in V(G)$ for which $\ell(v) \leqslant L$, and suppose that $\ell(w) = L + 1$. Let N be the set of vertices $u \lhd w$ such that $s \preccurlyeq u$ for some $s \in S'$, and consider the operations performed in an invocation of `FindInfima`(w) under the stated initial conditions, and with `foundcircuit` $=$ `false`.

If the invocation `FindInfima`(w) terminates without setting v to an element of N in the loop from lines 7 – 15, this is because the condition on line 12 was met, in which case the invocation terminates with `foundcircuit` $=$ `true`. Otherwise, all of the preceding recursive invocations `FindInfima`(v) for $v \lhd w$ terminated with `status`$(v) =$ `complete` and `foundcircuit` $=$ `false`, in which case the value of `status`(u) is unchanged for any vertex u such that $u \not\preccurlyeq v$ for all of the preceding values of v. Also, by the induction hypothesis, because these preceding recursive invocations `FindInfima`(v) did not terminate with `foundcircuit` $=$ `true`, we have $\ell(v) > L$.

Let $u_0 \lhd w$ be the first vertex of S' to occur as the value of v in the loop on lines 7 – 15 of the invocation `FindInfima`(w). Then, any chain $s = u_d \lhd u_{d-1} \lhd \cdots \lhd u_1 \lhd u_0$, such that $s \in S$ and $u_j \notin S$ for $j < d$, has length $d \leqslant L$ by the definition of $\ell(w)$. Because $\ell(v) > L$ for any value of $v \lhd w$ from an earlier iteration, there cannot be a chain $s \lhd v_{d-1} \lhd \cdots \lhd v_1 \lhd v$, where $d \leqslant \ell$, for any such v; and because there is no such chain for $d > \ell$ by the definition of $\ell(w)$, there cannot be a chain $u_j \lhd v_{N-1} \lhd \cdots \lhd v_1 \lhd v$ for any length N, and for any $0 \leqslant j \leqslant d$.

Thus, for all $0 \leqslant j \leqslant d$ and for all $v \lhd w$ from earlier iterations of the for loop on lines 7 – 15, we have $u_h \not\preccurlyeq v$. In particular, values of `status`(u_j) are unchanged from their initial values before the invocation of `FindInfima`(w), and so for each $0 \leqslant j \leqslant d$, $u_j \in S$ before the invocation of `FindInfima`(u_0) if and only if $u_j \in S$ before the invocation of `FindInfima`(w). Therefore, just prior to the invocation of `FindInfima`(u_0), there is a chain $s \lhd u_{d-1} \lhd \cdots \lhd u_1 \lhd u_0$ for $d \leqslant L$, such that $s \in S$ and $u_j \notin S$ for $j < d$. By the induction hypothesis, the invocation of `FindInfima`(u_0) terminates with `foundcircuit` $=$ `true`: then the invocation of `FindInfima`(w) also terminates with `foundcircuit` $=$ `true`. By induction on $\ell(w)$, the claim then holds. ■

Definition 3-15. A vertex $w \in V(G)$ is *circuitous* if there is a vertex $v \prec w$ for which $w \prec v$, *i.e.* if there is a circuit in $\mathfrak{I}_f$ containing w.

The order $\preccurlyeq$ is a partial order if and only if there are no such vertices. We would like to describe simple conditions under which such a vertex causes an invocation of `FindInfima` to return with `foundcircuit` = `true`. We will show:

Lemma 3-16. *Suppose that every vertex $v \in V(G)$ is either untouched or complete with respect to* `Inf`*, and in particular that every circuitous vertex is untouched with respect to* `Inf`*. Then the set of circuitous vertices is noxious.*

Proof —

Let w be a circuitous vertex, and let N be the set of vertices $v \lhd w$ such that $w \prec v$. Because $\texttt{status}(w) = \texttt{untouched}$ initially, it follows that any vertex $u \in V(G)$ for which $w \preccurlyeq u \preccurlyeq w$ holds is not complete with respect to `Inf`; therefore each such u is untouched with respect to `Inf`. In particular, for each vertex u_j in any chain $w \lhd u_{d-1} \lhd \cdots \lhd u_0 \lhd w$, we have $\texttt{status}(u_j) = \texttt{untouched}$.

Consider an invocation of $\texttt{FindInfima}(w)$. On line 5, we set $\texttt{status}(w) \leftarrow \texttt{incomplete}$. Then, for all $u \in N$, there is a chain $w \lhd u_{d-1} \lhd \cdots \lhd u_0 = u$ such that $\texttt{status}(w) = \texttt{incomplete}$ and $\texttt{status}(u_j) = \texttt{untouched}$ for $0 \leqslant j < d$. If the invocation $\texttt{FindInfima}(w)$ terminates without setting v to an element of S in the loop from lines 7 – 15, this is because the condition on line 12 was met, in which case the invocation terminates with `foundcircuit` = `true`. Otherwise, let u_0 be the first vertex in S which is assigned to v in the loop on lines 7 – 15. By the very fact that $v \notin N$ in the preceding iterations, we do not have $w \preccurlyeq v$ for those iterations, and in particular there are not any vertices u' such that $w \preccurlyeq u' \preccurlyeq v$; then the values of $\texttt{status}(u')$ are unchanged throughout the recursive invocation $\texttt{FindInfima}(v)$ for these preceding iterations. In particular, for all vertices $u' \preccurlyeq u_0$, we have $\texttt{status}(u') = \texttt{untouched}$, $\texttt{status}(u') = \texttt{complete}$, or $\texttt{status}(u') = \texttt{incomplete}$. As remarked above, any set of vertices S consisting of vertices u' for which $\texttt{status}(u') = \texttt{incomplete}$ is noxious. Then, by Lemma 3-15, an invocation of $\texttt{FindInfima}(u_0)$ will return with `foundcircuit` = `true`: so the invocation of $\texttt{FindInfima}(w)$ will terminate with `foundcircuit` = `true` as well. Thus, under these conditions, the set of circuitous vertices is noxious. ■

Lemmas 3-15 and 3-16 provide a sufficient condition for an invocation $\texttt{FindInfima}(w)$ to return with `foundcircuit` = `true`. The following will provide a useful partial converse:

Lemma 3-17. *Suppose that, for some vertex $w \in V(G)$ every vertex $v \preccurlyeq w$ is either untouched or complete with respect to* `Inf`*. If there are no circuitous vertices $s \preccurlyeq w$ and* `foundcircuit` = `false`*, an invocation* $\texttt{FindInfima}(w)$ *will return with* `foundcircuit` = `false`*, and only change the values of* `status` *and* `Inf` *to make w complete with respect to* `Inf`*.*

Proof —

If w is complete with respect to `Inf`, then `FindInfima` terminates immediately

with no changes to any variables; otherwise, we may assume that w is untouched with respect to `Inf`. We will prove the claim by induction on the *depth* of w, which we define as the length of the longest chain $v_0 \lhd v_1 \lhd \cdots \lhd v_d = w$ in $V(G)$: if there are no circuitous vertices $s \preccurlyeq w$, d is guaranteed to be bounded above by $n = |V(G)|$.

Consider the base case $d = 0$, The conditions on lines 3 and 4 are not fulfilled as $\texttt{status}(w) = \texttt{untouched}$, so the first operation performed is to set $\texttt{status}(w) \leftarrow \texttt{incomplete}$. No vertex z satisfies the conditions of line 7, by assumption of the minimality of w, and so the instructions within that for loop are never executed; we then set $\texttt{status}(w) \leftarrow \texttt{complete}$, and return. As well, because `foundcircuit` is left unaltered, we still have $\texttt{foundcircuit} = \texttt{false}$. Then, the only change performed by $\texttt{FindInfima}(w)$ is to set $\texttt{status}(w) \leftarrow \texttt{complete}$; but because $v \preccurlyeq w \iff v = w$ in this case, and because we had

$$\texttt{Inf}(w, p) \;=\; \left\{ \begin{array}{ll} 0\,, & \text{for } p = \texttt{Path}(w) \\ -1\,, & \text{otherwise} \end{array} \right\} \tag{3.20}$$

prior to the invocation, this is sufficient to cause w to be complete with respect to `Inf`.

Suppose then that the claim holds for all vertices for with some depth $d \in \mathbb{N}$, and that w has depth $d + 1$. The conditions on lines 3 and 4 are not fulfilled, as $\texttt{status}(w) = \texttt{untouched}$; we then set $\texttt{status}(w) \leftarrow \texttt{incomplete}$. In the loop on lines 7 – 15, we iterate over all $v \lhd w$: by assumption, every such v is either untouched or complete with respect to `Inf`, as are all vertices $u \preccurlyeq v$ for such v. It is clear that $\texttt{FindInfima}(v)$ simply returns without changing `Inf` or `status` when initially we have $\texttt{status}(v) = \texttt{complete}$; then by induction, after each recursive call to $\texttt{FindInfima}(v)$ on line 11, each vertex v is complete with respect to `Inf` and $\texttt{foundcircuit} = \texttt{false}$. The condition on line 12 is then unfulfilled, so we execute the for loop on line 13 and proceed to the next vertex z.

Initially, for every $p \in \{1, \ldots, k\} \setminus \{\texttt{Path}(w)\}$, we have $\texttt{Inf}(w, p) = -1$: then, for such p, the initial value of $\texttt{Inf}(w, p)$ is bounded above by the values of $\texttt{Inf}(v, p)$ for each $v \lhd w$ after the recursive calls $\texttt{FindInfima}(v)$. Then by induction, it is easy to show that after the loop on lines 7 – 15,

$$\texttt{Inf}(w, p) \;=\; \max_{v \lhd w} \texttt{Inf}(v, p)\,, \quad \text{for } p \neq \texttt{Path}(w). \tag{3.21}$$

For any $u \in V(G)$ with $\texttt{Path}(u) = p \neq \texttt{Path}(w)$, we have in particular $u \neq w$, and so $u \preccurlyeq w$ if and only if $u \preccurlyeq v$ for some $v \lhd w$; and this holds if and only if $\texttt{Dist}(u) \leqslant \texttt{Inf}(v, \texttt{Path}(u)) \leqslant \texttt{Inf}(w, \texttt{Path}(u))$ for that same v after the loop on lines 7 – 15, by the induction hypothesis. Finally, because there are no circuitous vertices $u \preccurlyeq w$, we have in particular $w \not\preccurlyeq v$ for all $v \lhd w$; then $\texttt{Inf}(v, \texttt{Path}(w)) < \texttt{Dist}(w)$ for all $v \lhd w$, in which case we have $\texttt{Inf}(w, \texttt{Path}(w)) = \texttt{Dist}(w)$ after line 15. Therefore, we also have $u \preccurlyeq w \iff \texttt{Dist}(u) \leqslant \texttt{Inf}(w, \texttt{Path}(u))$ when $\texttt{Path}(u) = \texttt{Path}(w)$: thus, the claim holds for w as well. By induction, the Lemma holds. ■

Lemmas 3-15, 3-16, and 3-17 will form the basis of our analysis of Algorithm 3-4 in the next section.

Analysis of Algorithm 3-4

In the previous section, we described the behavior of the subroutine `FindInfima` under conditions of the vertices being either untouched or complete with respect to `Inf`. The role of the procedure `TestNaturalPreorder` in Algorithm 3-4 (page 122) is to establish a path-labelling (`Dist`, `Path`) of the orbits of f, initialize arrays `status` and `Inf` so that every vertex is untouched with respect to `Inf`, and then repeatedly invoke `FindInfima` in an attempt to change the status of every vertex to that of being complete with respect to `Inf`. In doing so, we will either determine that $\mathfrak{I}_f$ has circuits (and so $\preccurlyeq$ is not a partial order), or assign values of $\texttt{Inf}(w, p)$ for all $w \in V(G)$ and $p \in \{1, \ldots, k\}$ so that (`Dist`, `Path`, `Inf`) forms a system of infima for $\preccurlyeq$.

To establish a path-labelling (`Dist`, `Path`) for the orbits of f, we use a subroutine `InitializePath`. It is straightforward to verify that $\texttt{InitializePath}(w, P)$ sets $\texttt{Path}(v) = P$ for $v = w$ and for all vertices v which precede w on the same path, and correctly assigns their distance along the path from the first vertex of the path. As well, for all such vertices v, it initializes $\texttt{status}(v)$ and $\texttt{Inf}(v, p)$ for each $p \in \{1, \ldots, k\}$ so that v is untouched with respect to `Inf`. In doing so, it is easy to see that it does not affect the status of any other vertex, or the labelling of any other paths. `TestNaturalPreorder` calls $\texttt{InitializePath}(v, p)$ for each vertex $w \in O$,[14] setting the path-label $p = \texttt{nextpath}$ of each vertex in O to a distinct integer $1 \leqslant p \leqslant k$. This initializes every vertex to be untouched with respect to `Inf`, and establishes a path-labelling (`Dist`, `Path`).

Next, we set $\texttt{foundcircuit} \leftarrow \texttt{false}$, and we invoke $\texttt{FindInfima}(w)$ for each vertex $w \in O$, aborting in the case that we discover $\texttt{foundcircuit} = \texttt{true}$. We return the triple (`Dist`, `Path`, `Inf`) if the loop on lines 13 – 16 terminates without ending the procedure call.

Consider a loop precondition on the values of `foundcircuit`, `status`, and `Inf` where, prior to some iteration of the loop on lines 13 – 16, we have

(*i*) $\texttt{foundcircuit} = \texttt{false}$,

(*ii*) every vertex is either untouched or complete with respect to `Inf`, and

(*iii*) any circuitous vertices in particular are untouched with respect to `Inf`.

By Lemma 3-16, the set of circuitous vertices is noxious under these conditions; then, by Lemma 3-15, $\texttt{FindInfima}(w)$ will return with $\texttt{foundcircuit} = \texttt{true}$ in the case that there exists some circuitous vertex $v \preccurlyeq w$. If such a circuitous vertex v exists, `TestNaturalPreorder` terminates with a return value of `nil`. Otherwise, by Lemma 3-17, `TestNaturalPreorder` terminates with $\texttt{foundcircuit} = \texttt{false}$ and w complete with respect to `Inf`. In particular, `status` and `Inf` in such a way as to make w complete with respect to `Inf`, in which case it is still the case that every vertex $v \in V(G)$ is untouched or complete with respect to `Inf`.

[14] For an arbitrary path cover $\mathcal{C}$ for a geometry (G, I, O), the initial point of some paths of $\mathcal{C}$ may not be a vertex in I. While we are primarily interested in the case where $|I| = |O|$, in which case there is a one-to-one correspondence between paths and elements of I, we may make the algorithm more general by labelling paths from their endpoint. In doing so, this algorithm may also be used to construct the natural pre-order for f in cases where $|I| < |O|$.

Input : (G, I, O), a geometry with $|O| = k$
$f : V(G) \longrightarrow V(G)$, a successor function for (G, I, O)
$g : V(G) \longrightarrow V(G)$, the inverse function of f
Output: Either a triple $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ representing the natural pre-order for f (in the case that f is a flow function), or `nil` (in the case that f is not a flow function).

```
1  procedure TestNaturalPreorder(G, I, O, f, g):
2  begin
3      let status : V(G) ⟶ {untouched, incomplete, complete};
4      let Dist : V(G) ⟶ ℕ;
5      let Path : V(G) ⟶ ℕ;
6      let Inf : V(G) × {1, ..., k} ⟶ ℕ;

7      let nextpath ← 1;
8      for all w ∈ O do
9          InitializePath(w, nextpath);
10         nextpath++;
11     endfor

12     let foundcircuit ← false;
13     for all w ∈ O do
14         FindInfima(w);
15         if foundcircuit then return nil;
16     endfor
17     return (Dist, Path, Inf);
18 end


19 subroutine InitializePath(w, P):
20 begin
21     if g(w) = nil then Dist(w) ← 0;
22     else
23         InitializePath(g(w));
24         Dist(w) ← Dist(g(w)) + 1;
25     endif
26     Path(w) ← P;

27     for all p ∈ {1, ..., k} do
28         if p = P
29             then  Inf(w, p) ← Dist(w);
30             else  Inf(w, p) ← −1 ;
31     endfor
32     status(w) ← untouched;

33     return
34 end
```

ALGORITHM 3-4. An algorithm to test whether or not the natural pre-order of a successor function f for a geometry (G, I, O) is a partial order. The subroutine `FindInfima` is described in Algorithm 3-3 on page 116.

By induction on the iteration over $w \in O$ on line 13, we may then prove that an invocation of $\texttt{TestNaturalPreorder}(G, I, O, f, g)$ returns $\texttt{nil}$ if the natural pre-order $\preccurlyeq$ is not antisymmetric, and returns with a triple $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ forming a system of infima for $\preccurlyeq$ in the case that $\preccurlyeq$ is antisymmetric. If there are any circuitous vertices v, eventually there will be an invocation $\texttt{FindInfima}(w)$ for which $v \preccurlyeq w$; because the loop precondition above is maintained until such a vertex w is found, it follows that $\texttt{TestNaturalPreorder}(G, I, O, f, g)$ will terminate with return value $\texttt{nil}$. Otherwise, it will eventually terminate with return value $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ such that $(\texttt{Dist}, \texttt{Path})$ is a path-labelling of the orbits of f, and where every vertex $w \in O$ is complete with respect to $\texttt{Inf}$. Because every $v \in V(G)$ is bounded above by some $w \in O$, it follows that $\texttt{status}(v) = \texttt{complete}$, $\texttt{Inf}(v, p) \geqslant -1$ for all $p \in \{1, \ldots, k\}$, and

$$u \preccurlyeq v \iff \texttt{Dist}(u) \leqslant \texttt{Inf}(v, \texttt{Path}(u)) \qquad \text{for all } u, v \in V(G). \tag{3.22}$$

Thus, the triple $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ forms a system of infima for $\preccurlyeq$.

In $\texttt{TestNaturalPreorder}(G, I, O, f, g)$, the subroutine $\texttt{InitializePath}(w, P)$ is run once for each vertex along each path, and so is invoked $n = |V(G)|$ times; and in each invocation, the run-time is dominated by the loop on lines 27 – 31, which performs k iterations of a constant-time assignment. Thus, the loop on lines 8 – 11 runs in time $O(kn)$. However, we will show that this time requirement is swamped by cumulative run-time of the invocations of $\texttt{FindInfima}$.

For vertices $v \in V(G)$ in general, the subroutine $\texttt{FindInfima}(v)$ is invoked on line 14 for $w \in O$, and line 11 of Algorithm 3-3 for $v \in V(G)$ such that $v \lhd w$ for some $w \in V(G)$. It is easy to show that every $w \in O$ is maximal in $\preccurlyeq$, in which case it is invoked only once for such vertices; for any other vertex v, $\texttt{FindInfima}(v)$ is invoked once for each $w \in V(G)$ such that $v \lhd w$; because this holds precisely for the set of vertices $w = f(v)$ or $w \sim z = f(v)$, this is bounded above by the degree of $f(v)$. The first time that $\texttt{FindInfima}(v)$ is invoked for a particular vertex v, v is untouched with respect to $\texttt{Inf}$; for any subsequent invocation made, v is complete with respect to $\texttt{Inf}$, and the invocation terminates in time $O(1)$. Then, the time requirements of all invocations of $\texttt{FindInfima}(v)$ for a particular vertex v (but not including the time consumed by recursive invocations for vertices $u \lhd v$) is bounded above by the time spent in the first invocation of $\texttt{FindInfima}(v)$, plus $\deg(f(v))$ for all subsequent invocations combined. The time required by the first invocation is dominated by by the loop at line 13, inside the loop from lines 7 – 15 (of Algorithm 3-3). The inner loop performs k iterations of condition tests and assignments requiring $O(1)$ time, which is repeated for each $u \lhd v$ by the outer loop: as the number of such vertices u is bounded above by $\deg(v)$, the outer loop then requires time $O(k \deg(v))$.

By a temporary abuse of notation, let $\deg(f(v)) = 0$ for vertices $v \in O$. The total run-time of all invocations of $\texttt{FindInfima}(v)$ is then $O(k \deg(v) + \deg(f(v)))$ for any $v \in V(G)$: summing over all vertices v, the cumulative run-time of the invocations of $\texttt{FindInfima}$ is then

$$\begin{aligned} O\left(\sum_{v \in V(G)} \Big[k \deg(v) + \deg(f(v))\Big]\right) &= O\left(\sum_{v \in V(G)} k \deg(v) \;+\; \sum_{v \in V(G)} \deg(f(v))\right) \\ &= O\Big(2k\,|E(G)| \,+\, 2\,|E(G)|\Big) \;=\; O(km)\,. \end{aligned} \tag{3.23}$$

We have therefore shown:

Lemma 3-18. *Let (G, I, O) be a geometry with $|E(G)| = m$ and $|O| = k$, $f : V(G) \longrightarrow V(G)$ be a successor function for (G, I, O), and let $g : V(G) \longrightarrow V(G)$ be the inverse of f. Then the procedure* `TestNaturalPreorder`(G, I, O, f, g) *terminates in time $O(km)$, and returns either*

- `nil`*, if the natural pre-order $\preccurlyeq$ of f is not antisymmetric; or*
- *a system of infima* $(\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ *for $\preccurlyeq$, where in particular* $(\texttt{Dist}, \texttt{Path})$ *is a path-labelling of the orbits of f.*

Restricting to those geometries satisfying the extremal bound of Theorem 3-7, this provides a proof of Theorem 3-4 on page 91.

3.4.4 The complete flow-finding algorithm

As a summary of the results of the previous sections, we now describe an algorithm for discovering whether a geometry (G, I, O) has a flow $(f, \preccurlyeq)$ in the case that $|I| = |O|$, and to construct one if so.

On line 6, we eliminate any geometry which violates the extremal bound given by Theorem 3-7, thus restricting to geometries for which $m < kn$. The invocation of `ObtainMaxFamilyPaths`(G, I, O) constructs mutually inverse partial functions on $V(G)$ whose orbits describe a maximum-size family of vertex disjoint paths, by Lemma 3-14; the loop starting at line 8 then determines whether or not this family is a path cover, by seeing if at least one of f and g are defined for all vertices not in $I \cap O$. If there are vertices $v \notin \mathrm{dom}(f) \cup \mathrm{dom}(g) \cup (I \cap O)$, then the family of paths obtained is in particular not a causal path cover: by Theorems 3-11 and 3-6, (G, I, O) then has no flow. Otherwise, f is a successor function for (G, I, O), and we may test whether or not its natural pre-order $\preccurlyeq$ is a partial order by invoking `TestNaturalPreorder`(G, I, O, f, g) on line 11. By Lemma 3-18, if `partialOrd` $=$ `nil` on line 13, then $\preccurlyeq$ is not a partial order, and subsequently $\mathcal{P}_f$ is not a causal path cover; again by Theorems 3-11 and 3-6, (G, I, O) then has no flow. Otherwise, `partialOrd` contains a system of infima for $\preccurlyeq$, and we return $(f, g,$ `partialOrd`$)$.

The run time for `FindFlow` is dominated by those of `ObtainMaxFamilyPaths` and `TestNaturalPreorder`, which are both $O(km)$. Due to the restriction imposed at line 6, we then have the following:

Theorem 3-19. *For a geometry (G, I, O) with $|V(G)| = n$ and $|I| = |O| = k$, the procedure* `FindFlow`(G, I, O) *terminates in time $O(k^2 n)$, returns* `nil` *if (G, I, O) has no flow, and a triple $(f, g,$* `partialOrd`*) otherwise; where $f : V(G) \longrightarrow V(G)$ is a successor function for (G, I, O), $g : V(G) \longrightarrow V(G)$ is the inverse of f, and* `partialOrd` *is a system of infima for the natural pre-order $\preccurlyeq$ of f. In particular, if (G, I, O) has a flow, then $(f, \preccurlyeq)$ is a flow for (G, I, O).*

Input : G, a graph;
$I, O \subseteq V(G)$, sets of vertices of equal size.
Output: Either `nil`, if (G, I, O) does not have a flow; or a successor function $f : V(G) \rightharpoonup V(G)$ for (G, I, O), its inverse function $g : V(G) \rightharpoonup V(G)$, and a system of infima $\texttt{partialOrd} = (\texttt{Dist}, \texttt{Path}, \texttt{Inf})$ for the natural pre-order $\preccurlyeq$ for f, in the case where (G, I, O) has a flow.

```
1  procedure FindFlow(G, I, O):
2  begin
3      let n ← |V(G)|;
4      let m ← |E(G)|;
5      let k ← |O|;
6      if m > kn − C(k+1, 2) then return nil;
7      let (f, g) ← ObtainMaxFamilyPaths(G, I, O);
8      for all v ∈ V(G) do
9          if v ∉ I ∩ O and f(v) = g(v) = nil then return nil;
10     endfor
11     let partialOrd ← TestNaturalPreorder(G, I, O, f, g);
12     if partialOrd = nil
13         then return nil;
14         else  return (f, g, partialOrd) ;
15 end
```

ALGORITHM 3-5. A complete algorithm for determining if a geometry (G, I, O) has a flow in the case that $|I| = |O|$, using the procedures defined in Algorithms 3-1 and 3-4 (pages 107 and 122 respectively).

3.5 A semantic map for the DKP representation

The purpose of this Section is to show how to obtain a right-inverse to the DKP representation map $\mathcal{R}$, from unitary circuits to one-way measurement based algorithms. We show how this may be done for a one-way measurement patterns with an underlying geometry (G, I, O), and whose operations satisfy certain constraints related to a flow.

3.5.1 Influencing walks and measurement adaptation

For a geometry (G, I, O) with a flow $(f, \preccurlyeq)$, the partial order $\preccurlyeq$ is meant to describe an order of measurements which allow a unitary embedding to be performed. However, it represents a *sufficient* condition, not a necessary one: and in particular, as shown in [37], it is a sufficient condition under the constraints that a measurement of a qubit $v \in O^{\mathsf{c}}$ affects the following operations:

(i) it induces a sign dependency on the measurement of its successor $f(v)$, and a π-dependency on the measurements of its other neighbors $w \sim f(v)$, and affects no other measurements; and

(*ii*) it induces an X correction on $f(v)$ if this is an element of O, and a Z correction on any neighbor $w \sim f(v)$ which are elements of O, and induces no other corrections.

These constraints essentially arise from the simplified DKP construction, as described in Section 3.2.

As we noted in Section 2.2.5, we may eliminate π-dependencies in a one-way measurement pattern based computation by performing signal shifting: this may result in more qubits depending on a given measurement result, but relaxes the measurement dependencies to allow transformations to be done with less depth complexity (neglecting the cost of computing boolean expressions for adapting the measurements). As well, measurements along Pauli axes are not affected by any previous measurements (although their results may be affected, in the counterfactual sense described in Section 3.2). We will see that, as a result, the appropriate tool for considering the measurement order for standardized one-way measurement patterns is not the partial order $\preccurlyeq$, but rather a special class of influencing walks.[15]

For a unitary circuit C, we may perform the DKP construction by first producing a pattern $\tilde{P}$ via the *simplified* DKP construction. As we remarked in Section 3.2, we have $v \preccurlyeq w$ for two distinct qubits involved in $\tilde{P}$ if the measurement of v induces X or Z corrections onto w, which are absorbed into the measurement basis for w if $w \in O^{\mathsf{c}}$. These corrections are induced if and only if either $w = f(v)$, in which case the induced correction is $\mathsf{X}^{\mathsf{s}[v]}$, or $w \sim f(v)$, in which case the induced correction is $\mathsf{Z}^{\mathsf{s}[v]}$. We will refer to these dependencies as *type* (*i*) and *type* (*ii*) dependencies. Note that the relationships of qubits with dependencies of type (*i*) or type (*ii*) are slight generalizations[16] of the relationships occurring in segments of type (*i*) and type (*ii*) of influencing walks; what is significant about these types is in particular whether the dependency arises from an X or a Z correction induced by the measurement of a qubit.

Consider the further operations required to standardize $\tilde{P}$, into a standardized pattern:

1. As we noted in Section 2.2.5, measurements along a Pauli axis can be performed independently of any previous measurement; then, for qubits $v \in O^{\mathsf{c}}$ and $w = f(v) \in O^{\mathsf{c}}$, we may eliminate any dependency of type (*i*) of w on v if the former is to be measured with an angle $\alpha \in \frac{\pi}{2}\mathbb{Z}$.

2. For a measurement along the Y axis in particular (*i.e.* for measurement angles $\alpha \in \pi\mathbb{Z} + \pi/2$), the way in which type (*i*) dependencies are eliminated are to replace sign-dependencies with π-dependencies, which is the effect of a dependency of type (*ii*). Thus, when $w = f(v)$ for some $v \in O^{\mathsf{c}}$ and $w \in O^{\mathsf{c}}$ measured with an odd multiple of $\pi/2$, we may replace the type (*i*) dependency of w on v with a dependency of type (*ii*).

[15] The importance of influencing walks for depth-complexity in one-way measurement based computation was first realized by Broadbent and Kashefi, who first coined the term *influencing path* while working on the results of [24]. The recognition of influencing paths as objects of interest helped to inspire simplifications in the work which I've presented in this Chapter form their original forms, and were incorporated in the published version of these results in [45]. We will consider here a different approach to the analysis of depth of one-way measurement based procedures in terms of influencing walks than the work in [24].

[16] A segment of type (*ii*) with respect to the path cover $\mathcal{P}_f$ is a path of the form $v \to z \to w$, where the edge zw is not covered by a path of $\mathcal{P}_f$; measurement dependencies of type (*ii*), by contrast, include the case $w = f(f(v))$.

3. Whenever a qubit $w \in O^{\mathsf{c}}$ has a type (ii) dependency on a qubit v, the effect of the measurement result of v is not to change the basis of measurement for w, but rather the significance of the measurement result of w, which is formalized in the process of signal shifting. Specifically, for any qubit w which has a type (ii) dependency on another qubit v, we may eliminate the dependency of w on v if we make every qubit x with a type (i) dependency on w also depend on v, and similarly for qubits y with a type (ii) dependency on w.

Recursively applying the last rule, we may describe walks in G similar to influencing walks, of the kind described below:

Definition 3-16. For a geometry (G, I, O), a successor function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, and a vector $\mathbf{a} = (\alpha_v)_{v \in O^{\mathsf{c}}}$ of default measurement angles for each qubit, a *segment of an adapting walk* for f is any directed path of one of the following three forms:

(i) $v \to w$, where $w = f(v)$, and w is not to be measured with an angle $\alpha \in \frac{\pi}{2}\mathbb{Z}$;

(ii) $v \to z \to w$, where $v \in O^{\mathsf{c}}$, $z = f(v)$, and $w \sim z$;

(iii) $v \to w$, where $w = f(v)$, and $\alpha_w \in \pi\mathbb{Z} + \pi/2$.

An *adapting walk* for f is any directed walk W in G which may be formed concatenation of such segments, subject to the constraint that a type (i) segment may only occur at the end of the walk, and that either W ends with a segment of type (i) or at a vertex in O.

Note that an adapting segment for f of type (iii) is also a segment of type (i) of an influencing walk of $\mathcal{P}_f$. Thus, all adapting walks for f are also influencing walks of $\mathcal{P}_f$.

We will show that adapting walks describe the propagation of measurement or correction dependency on previous measurement results for one-way measurement based patterns arising from the DKP construction, with the type of adapting walk determining the nature of the dependency, and that these dependencies may cancel when an even number of them converge on a given qubit:

Theorem 3-20. *Let $P = \mathcal{R}(C)$ be the image of a unitary circuit C in the DKP construction $\mathcal{R}$, where C is given as a stable index tensor expression. Let (G, I, O) be the geometry consisting of the graph G of entangling operations on qubits of P (which are indexed by wire-segments of C), the set I of non-prepared qubits, and the set O of non-measured qubits, and let $\mathbf{a} = (\alpha_v)_{v \in O^{\mathsf{c}}}$ be the vector of default measurement angles for each qubit. If $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ is the function mapping each deprecated index in C to the corresponding advanced index, then f is a flow function for (G, I, O); and for two distinct qubits $v, w \in V(G)$, the following holds:*

(i) *there is a measurement on $w \in O^{\mathsf{c}}$ where the sign of the measurement angle depends on $\mathsf{s}[v]$, or a potential X correction on $w \in O$ depending on $\mathsf{s}[v]$, if and only if there are an odd number of adapting walks for f from v to w which end in a segment of type (i);*

(*ii*) *there is a potential Z correction on $w \in O$ depending on $\mathsf{s}[v]$ if and only if there are an odd number of adapting walks for f from v to w which end in a segment of type* (*ii*).

Proof —

Consider the one-way measurement pattern $\tilde{P}$ obtained from C by the *simplified* DKP construction. From the discussion in Section 3.2, the function f is a flow-function by construction. We may obtain the one-way measurement pattern P by standardizing the procedure $\tilde{P}$.

Consider how the process of standardization propagates the dependencies of qubits w on a particular qubit v. Let us say that a qubit w has an *X-dependency* on a qubit v in a measurement procedure if either the measurement of $w \in O^{\mathsf{c}}$ has a sign dependency on $\mathsf{s}[v]$, or if $w \in O$ may be subject to an X correction depending on the value of $\mathsf{s}[v]$; and similarly, a qubit w has a *Z-dependency* on a qubit v in a measurement procedure if either the measurement of w has a π-dependency on $\mathsf{s}[v]$, or if w may be subject to a Z correction depending on the value of $\mathsf{s}[v]$. If a qubit w has a Z-dependency on another qubit v, performing signal shifting on the measurement of w will cause any qubits with an X-dependency on w to (a) accrue an X-dependency on v, if it had none previously, or (b) lose its X-dependency on v, if it already had such a dependency, due to cancellation modulo 2 ; and similarly for Z-dependencies.

We may then describe how standardization transformations cause dependencies to propagate, in terms of "locally describable" dependencies. Starting from the measurement procedure $\tilde{P}$, the only X-dependencies which exist are of vertices $f(v)$ on their respective qubits $v \in O^{\mathsf{c}}$; then we may reduce the problem to tracking Z dependencies. Because the Z-dependencies of a qubit w may always be removed unless $w \in O$, we may model this propagation in the form of a "binary walk" of a measurement result $\mathsf{s}[v]$, starting from v and ending at vertices which have unremovable Z-dependencies. We do this as follows:

1. For a qubit $w \in O^{\mathsf{c}}$ and $z \sim f(w)$ distinct from w, signal shifting on the measurement at w will propagate the Z-dependencies of w to z (possibly cancelling modulo 2 as described above).

2. For a qubit $w \in O^{\mathsf{c}}$ and a qubit $z = f(w)$ which is to be measured with an angle $\alpha \in \pi\mathbb{Z} + \pi/2$, the X-dependency of z on w is equivalent to a Z-dependency; signal shifting on the measurement at w will then propagate the Z-dependencies of w to z (possibly cancelling modulo 2 as described above).

Then, we may describe the propagation of the effects of a measurement result $\mathsf{s}[v]$ by an $V(G) \times V(G)$ matrix T over $\mathbb{Z}$, given by

$$T_{wv} \quad = \quad \left\{ \begin{array}{ll} 1\,, & \text{if } v \in O^{\mathsf{c}},\ w \sim f(v), \text{ and } w \neq v \\ 1\,, & \text{if } v \in O^{\mathsf{c}},\ w = f(v), \text{ and } w \text{ is to be measured with angle } \alpha \in \pi\mathbb{Z} + \pi/2 \\ 0\,, & \text{otherwise} \end{array} \right\} . \tag{3.24}$$

Note that $T_{wv} = 1$ only if v and w are connected by a segment of an influencing walk for $\mathcal{P}_f$, and in particular, an adapting segment for f of type (ii) or (iii). Then, $\hat{\mathbf{e}}_v \cdot (T^\ell \, \hat{\mathbf{e}}_w) \neq 0$ only if v and w are connected by an influencing walk for $\mathcal{P}_f$ consisting of ℓ segments. Because influencing walks of $\mathcal{P}_f$ are of bounded length (in particular because there are no vicious circuits), there exists an ℓ for which $T^\ell = 0$. In particular, there are no eigenvectors of T with eigenvalue 1: then, the matrix $\mathbb{1}_n - T$ has trivial kernel, and is therefore invertible (and in particular, invertible modulo 2).

The matrix T represents the propagation of a Z-measurement dependency due to signal shifting or simplification of Pauli measurements, but neglects X-dependencies of qubits not to be measured along a Pauli axis, as well as the persistent Z-dependencies of elements of O, which cannot be signal-shifted — these are dependencies which cannot be eliminated by pattern transformation. If after some number of standardization operations, a qubit z has an unremovable dependency on a qubit w with a (removable) Z-dependency on some qubits, the dependencies of w are accumulated into the dependencies of z by a signal shifting on the measurement at w. We describe this accumulation separately for X- and Z-dependencies.

(i) Consider another $V(G) \times V(G)$ matrix F over $\mathbb{Z}$, given by

$$F_{wv} \;=\; \left\{ \begin{array}{ll} 1\,, & \text{if } v \in O^{\mathsf{c}},\ w = f(v), \text{ and } w \text{ is not to} \\ & \text{be measured with an angle } \alpha \in \pi\mathbb{Z} + \pi/2 \\ 0\,, & \text{otherwise} \end{array} \right\}\,. \tag{3.25}$$

Then F represents the effect of a signal shift on the X-dependencies of vertices $f(v)$ on their predecessors $v \in O^{\mathsf{c}}$. For qubits of the form $w = f(z)$ for $z \in O^{\mathsf{c}}$ arbitrary, the contribution of a possible dependency of w on a qubit v arising from the ℓ^{th} signal shift for $\ell \geqslant 0$ is then given by

$$\hat{\mathbf{e}}_w \cdot \left(F T^\ell \, \hat{\mathbf{e}}_v \right); \tag{3.26}$$

and the cumulative dependency over all signal shifting operations is then given by

$$\begin{aligned} \hat{\mathbf{e}}_w \cdot \left(\sum_{\ell \in \mathbb{N}} F T^\ell \, \hat{\mathbf{e}}_v \right) \;&=\; \hat{\mathbf{e}}_w \cdot F \left(\sum_{\ell \in \mathbb{N}} T^\ell \right) \hat{\mathbf{e}}_v \\ &=\; \hat{\mathbf{e}}_w \cdot F \left(\mathbb{1}_n - T \right)^{-1} \hat{\mathbf{e}}_v\,, \end{aligned} \tag{3.27}$$

as $\mathbb{1} - T$ is non-singular. If this evaluates to zero modulo 2, there is no X-dependency of w on v in the pattern P.

(ii) For $w \in O$, the Z-dependency of w on an arbitrary qubit v can be computed as the cumulated contributions from each possible number ℓ of signal shifts, which for each $\ell \geqslant 1$ is given by

$$\hat{\mathbf{e}}_w \cdot (T^\ell \, \hat{\mathbf{e}}_v); \tag{3.28}$$

then, similarly to the previous case, the cumulative dependency over all signal shifting operations is then given by

$$\hat{\mathbf{e}}_w \cdot \left(\sum_{\ell \in \mathbb{N}} T^\ell \, \hat{\mathbf{e}}_v \right) \;=\; \hat{\mathbf{e}}_w \cdot (\mathbb{1}_n - T)^{-1} \, \hat{\mathbf{e}}_v \,, \tag{3.29}$$

as $\mathbb{1} - T$ is non-singular. Again, if this evaluates to zero modulo 2, there is no Z-dependency of w on v in the pattern P.

After an "infinite" number of such dependency propagations performed on the initial procedure $\tilde{P}$ (equivalent to some finite number of shifts, with subsequent propagations having no effect), we obtain the measurement procedure P.

Note that the coefficients of F corresponds to adapting segments for f of type (i), and those of T correspond to adapting segments of type (ii) and (iii). In the procedure P, possible dependencies of the signs of measurement angles for a qubit w depending on $\mathsf{s}[v]$, of potential X corrections on a qubit w depending on $\mathsf{s}[v]$, exist then if there are an odd number of walks from v to w consisting of some number of type (ii) and (iii) adapting segments followed by a type (i) segment; and possible Z corrections on w depending on $\mathsf{s}[v]$ exist if there are an odd number of walks from v to $w \in O$ consisting entirely of type (ii) and (iii) adapting segments. This proves the Theorem. ■

Algorithm 3-6 describes an algorithm which, provided a one-way measurement pattern P in standard form (and without shift operators) and a flow f for the geometry underlying P, tests if the dependencies of P are consistent with the flow-function those described in Theorem 3-20. Let N be the length of the measurement pattern P, and let $n = |V(G)|$, $m = |E(G)|$, and $k = |O|$. There will then be exactly m entangling operations, $n - k$ measurement operations, and k measurement operations in P, where the latter two classes of operations may in principle depend on arbitrary subsets of $V(G)$; then, the length of the pattern is $N \in O(m + n^2) = O(n^2)$. The time required to construct the matrices T and F is $O(n^2)$, and the time to compute D and E (using *e.g.* the algorithm of [108] to perform matrix inversion modulo 2) is $O(n^3/\log n)$. The set S constructed on line 10 may be constructed in time $O(n)$; then, the run-time of Algorithm 3-6 is dominated by the nested for loops and repeated construction of S, both of which requires $O(Nn) \subseteq O(n^3)$ time in total to execute.

The characterization of dependencies between qubits in a pattern $P \in \mathrm{img}(\mathcal{R})$ is the penultimate step towards obtaining a semantic map for $\mathcal{R}$. Together with an algorithm to find flows, Algorithm 3-6 provides an efficient algorithm for testing membership in $\mathrm{img}(\mathcal{R})$: however, the proof of this claim will await the construction of a complete semantic map for $\mathcal{R}$.

3.5.2 Star decomposition of geometries with flows

For a geometry (G, I, O) with a flow-function f, the influence relation $\lhd$ for f suggests a decomposition of (G, I, O) into smaller geometries similar to the star decomposition of circuits described in Section 3.2. For each vertex $v \in V(G)$, define the vertex-set

$$V_{v\star} \;=\; \left\{ w \in V(G) \,\middle|\, w = v \text{ or } \left[v \lhd w \text{ and } w \neq f(f(v)) \right] \right\} \,, \tag{3.30}$$

Input : P, a standardized one-way measurement pattern;
f, a flow-function for the geometry (G, I, O) underlying P
Output: A boolean result indicating if P is consistent with the constraints of Theorem 3-20.

```
1  procedure TestDependencies(P, f):
2  begin
3      let T ← [ matrix over ℤ₂ depending on f described in Eqn (3.24) ];
4      let F ← [ matrix over ℤ₂ depending on f described in Eqn (3.25) ];
5      let D ← (𝟙_n − T)^{-1};
6      let E ← FD;
7      for each measurement or correction operation Op_w^B in P do
8          let xType ← ((Op = M) or (Op = X));
9          let zType ← (Op = Z);
10         let S ← {v ∈ V(G) | s[v] occurs in B};
11         for each v ∈ V(G) do
12             if v ∈ S then
13                 if (xType and E_wv = 0) or (zType and D_wv = 0)
14                     then return false;
15             else
16                 if (xType and E_wv = 1) or (zType and D_wv = 1)
17                     then return false;
18             endif
19         endfor
20     endfor
21     return true;
22 end
```

ALGORITHM 3-6. An algorithm to test whether the dependencies of a standardized one-way measurement pattern based measurement procedure is consistent with those produced by the DKP construction (see Theorem 3-20).

and from these, define the geometry

$$G_{v\star} = G[V_{v\star}] - \{xw \,|\, x, w \neq f(v)\} , \tag{3.31a}$$

$$I_{v\star} = V_{v\star} \setminus \{f(v)\} , \tag{3.31b}$$

$$O_{v\star} = V_{v\star} \setminus \{v\} . \tag{3.31c}$$

The graph $G_{v\star}$ consists of a star graph consisting of a single edge vw for $w = f(v)$, and some additional edges wx for $w = f(v)$ and $v \lhd x$. We will call the geometry $(G_{v\star}, I_{v\star}, O_{v\star})$ a *star geometry*[17] with *root* v, or with *center* $f(v)$.

[17] Star geometries as we have defined them are similar to, but distinct from, the geometries underlying *star patterns* in [38]. The decomposition result of Lemma 3-21 was motivated by the discussion of star geometries in that article.

Lemma 3-21. *Let (G, I, O) be a geometry with flow-function f. Then there is a decomposition of (G, I, O) into star geometries on each of the vertices $v \in O^{\mathsf{c}}$, together with the geometry $\big(G\big[\,\tilde{I}\,\big], I, \tilde{I}\,\big)$, where $\tilde{I} = V(G) \smallsetminus \mathrm{img}(f)$.*

Proof —

We proceed by induction on the size of $\mathrm{img}(f)$. If $\mathrm{img}(f) = \varnothing$, then $\mathrm{dom}(f) = \varnothing$ as well, in which case $I \subseteq V(G) = O$; we then have $(G, I, O) = \big(G\big[\,\tilde{I}\,\big], I, \tilde{I}\,\big)$. Otherwise, suppose that the claim holds for geometries (G, I, O) with flow-functions f in which $|\mathrm{img}(f)| \leqslant N$ for some $N \in \mathbb{N}$, and that $|\mathrm{img}(f)| = N + 1$. Because f is a flow function, $\mathfrak{I}_f$ is acyclic; then, there is a vertex $v \in V(G)$ such that $v \lhd w$ only for w maximal in the natural pre-order $\preccurlyeq$. Let $\mathcal{G}_{v\star} = (G_{v\star}, I_{v\star}, O_{v\star})$, and $\mathcal{G}' = (G', I, O')$, where

$$G' \;=\; G \smallsetminus f(v)\,, \quad \text{and} \qquad O' \;=\; O \mathbin{\triangle} \{v, f(v)\}\;. \tag{3.32}$$

Note that $O_{v\star} \subseteq O$ by the choice of v, and $I_{v\star} = O_{v\star} \mathbin{\triangle} \{v, f(v)\} \subseteq O \mathbin{\triangle} \{v, f(v)\}$. Because $V(G_{v\star}) = I_{v\star} \cup \{f(v)\}$, we then have $V(G') \cap V(G_{v\star}) \subseteq I_{v\star} \cap O'$: then the composition $(\bar{G}, \bar{I}, \bar{O}) = \mathcal{G}_{v\star} \circ \mathcal{G}'$ is well-defined, and we have

$$\begin{aligned} \bar{G} \;&=\; G_{v\star} \cup G' \\ &=\; \big\{xw \in G \,\big|\, w = f(v)\big\} \cup \Big[G \smallsetminus f(v)\Big] \;=\; G\,; \end{aligned} \tag{3.33a}$$

$$\bar{I} \;=\; \big(I_{v\star} \smallsetminus O'\big) \cup I \;=\; \varnothing \cup I \;=\; I\,; \tag{3.33b}$$

$$\bar{O} \;=\; O_{v\star} \cup \big(O' \smallsetminus I_{v\star}\big) \;=\; O_{v\star} \cup \big(O \smallsetminus O_{v\star}\big) \;=\; O\,. \tag{3.33c}$$

Then, we have $(G, I, O) = \mathcal{G}_{v\star} \circ \mathcal{G}'$. The restriction f' of the flow-function f to $\mathcal{G}'$ does not include $f(v)$, and therefore has cardinality at most N; then, $\mathcal{G}'$ has a decomposition into star geometries and the geometry $(G' \smallsetminus \mathrm{img}(f'), I, V(G') \smallsetminus \mathrm{img}(f'))$. Finally, note that $V(G) \smallsetminus \mathrm{img}(f') = V(G) \smallsetminus \mathrm{img}(f) = \tilde{I}$, by the definition of f': the lemma then holds by induction. ■

Figure 3-7 illustrates a decomposition of a geometry with a flow into star geometries in the manner described above.

For a geometry (G, I, O) with a given flow-function f, we will say that a star geometry $(G_{v\star}, I_{v\star}, O_{v\star})$ is *maximal* in (G, I, O) if, as in the proof of Lemma 3-21 above, every $w \in V(G)$ such that $v \lhd w$ is maximal in $\preccurlyeq$; then, we may decompose (G, I, O) by extracting maximal star geometries. The maximal elements of $V(G)$ in the natural pre-order are exactly the elements of O (as $z \preccurlyeq f(z)$ for any $z \in O^{\mathsf{c}}$). Thus, the root v of a maximal star geometry in (G, I, O) is one such that all neighbors $z \sim f(v)$ are elements of O or equal to v itself. There always exists such a maximal star for a geometry with a flow: an independent observation of a similar nature has since given rise to an improved algorithm for finding flows, which we describe in Section 3.6.3.

Algorithm 3-7 describes an procedure to obtain a vertex $w = f(v) \in V(G)$ which is the center of a maximal star geometry in some geometry $(G[V'], V' \smallsetminus W', O')$, using the fact that w is the center of such a star geometry if and only if all neighbors of w in the graph $G[V']$ are either equal to v or are elements of O'. Having found a vertex at the center of a

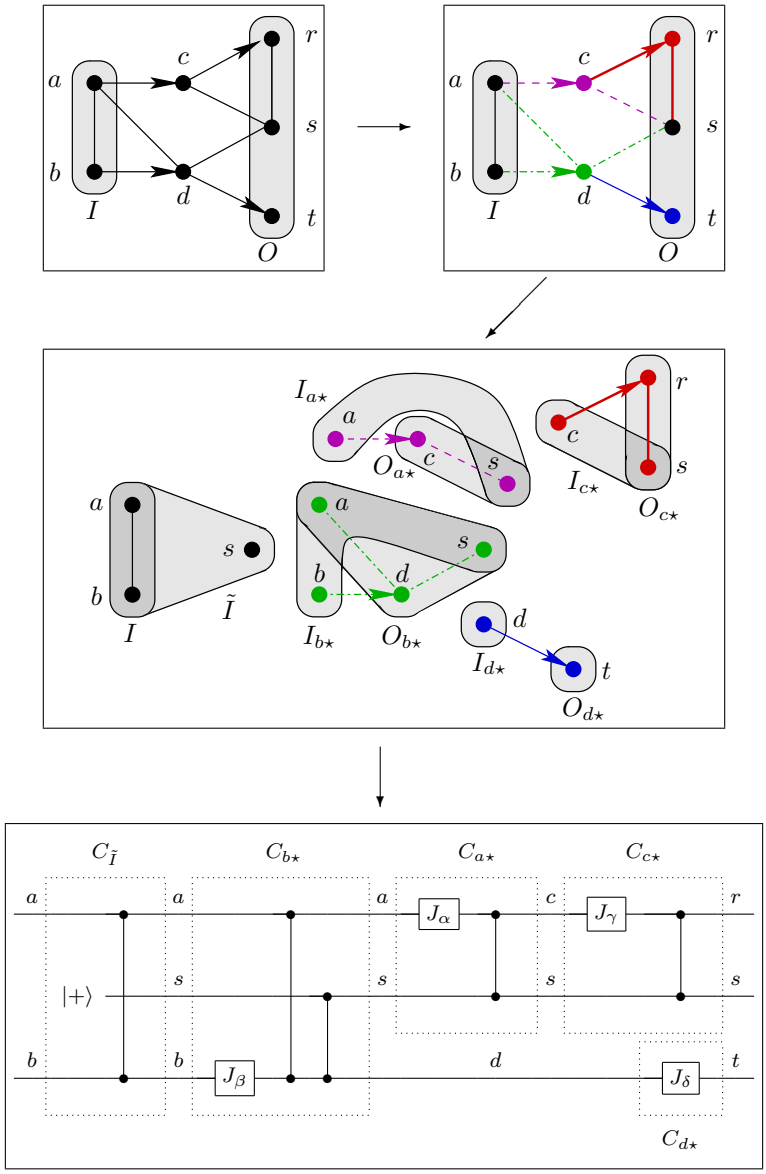


FIGURE 3-7. [*colour online*] Illustration of a decomposition of a geometry (G, I, O) with a flow-function f (denoted by arrows in the upper three panels) into star geometries for each $v \in O^{\mathsf{c}}$, and a reduced geometry $\big(G[\,\tilde{I}\,], I, \tilde{I}\big)$ for $\tilde{I} = V(G) \setminus \mathrm{img}(f)$. The middle panel illustrates each star geometry separately, located next to the geometries with which it may be composed to form the original geometry (G, I, O); vertices are repeated for each geometry in which they occur. The bottom panel illustrates the unitary circuits corresponding to these star geometries (including a labelling of wire-segments, *i.e.* of stable tensor indices), and the complete circuit obtained from composing them.

Input : G, a graph underlying a geometry (G, I, O);
V', W', O', sets describing a sub-geometry $(G[V'], V' \setminus W', O')$;
$f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, a flow-function function for (G, I, O)
Output: A vertex $w \in W'$ such that every neighbor v of w such that $v \in V'$ either satisfies $v = g(w)$ or $v \in O'$

```
1  procedure MaximalStarVertex(G, V', W', O', f):
2  begin
3      for each w ∈ W' do
4          let maximalStar ← true;
5          for all v ∼ w do
6              if v ∈ V' \ O' and w ≠ f(v) then
7                  maximalStar ← false;
8                  escape this loop;
9              endif
10         endfor
11         if maximalStar then return w;
12     endfor
13 end
```

ALGORITHM 3-7. An algorithm to obtain a vertex v which is the center of a maximal star geometry of a specified geometry.

maximal geometry of (G, I, O), we may obtain the decomposition $(G, I, O) = \mathcal{G}_{v\star} \circ \mathcal{G}'$ by constructing these geometries according to (3.31) and (3.32), respectively. The run-time of Algorithm 3-7 is taken up by constant-time comparisons for up to all of the neighbors of every element of O', and so requires time

$$\begin{aligned} O\left(\sum_{w\in O'} \deg(w)\right) \;\subseteq\; O\left(\sum_{w\in V(G)} \deg(w)\right) \;&=\; O(2\,|E(G)|) \\ &=\; O(m) \;\subseteq\; O(kn)\,, \end{aligned} \tag{3.34}$$

by the extremal bound of Theorem 3-7.

3.5.3 A semantic map from star decompositions

Star geometries $(G_{v\star}, I_{v\star}, O_{v\star})$ correspond to simple unitary circuits in the following manner. We may construct a circuit $C_{v\star}$ with an input wire for each qubit $a \in I_{v\star}$, and which performs the transformation

$$C_{v\star} \;=\; \left(\prod_{\substack{w\in O_{v\star}\\ w\sim f(v)}} \wedge Z^{[w,f(v)]}\right) J(\alpha)^{\left[\frac{f(v)}{v}\right]} \tag{3.35}$$

expressed in stable index notation (pages 38 – 38); we map the labels of the vertices in the geometries to the tensor index names in the expansion, and the angle α is a parameter

which we may relate to the measurement angle for the qubit v. Similarly, we may define a circuit $C_{\tilde{I}}$ which corresponds to the geometry $(G[\tilde{I}], I, \tilde{I})$:

$$C_{\tilde{I}} \;=\; \left(\prod_{vw \in E(G[\tilde{I}])} \wedge Z^{[v,w]}\right)\left(\prod_{v \in \tilde{I} \smallsetminus I} |+\rangle^{[v]}\right) . \tag{3.36}$$

We will call such circuits the *star circuits* corresponding to these geometries; these are the same circuits described in Section 3.2 in the star decomposition of circuits. Examples of such circuits for a particular geometry (G, I, O) are illustrated in the bottom panel of Figure 3-7, in which they are composed together to form a larger circuit. In that example, we may recover the geometry in the upper left-hand panel by composing

$$\begin{aligned}(G, I, O) \;=\; (G_{c\star}, I_{c\star}, O_{c\star}) \circ (G_{d\star}, I_{d\star}, O_{d\star}) \circ (G_{a\star}, I_{a\star}, O_{a\star}) \circ \\ (G_{b\star}, I_{b\star}, O_{b\star}) \circ \left(G[\tilde{I}], I, \tilde{I}\right);\end{aligned} \tag{3.37}$$

and similarly, the larger circuit C in Figure 3-7 is obtained by composing

$$C \;=\; C_{c\star} \circ C_{d\star} \circ C_{a\star} \circ C_{b\star} \circ C_{\tilde{I}} . \tag{3.38}$$

Algorithm 3-8 describes a procedure $\texttt{SemanticDKP}(G, I, O, P)$ which provides a semantic map $\mathcal{S}$ for the DKP representation map $\mathcal{R}$. We will show:

Lemma 3-22. *Let P be a standardized one-way measurement pattern with underlying geometry (G, I, O), where $|I| = |O|$. Then* $\texttt{SemanticDKP}(G, I, O, P)$ *returns a unitary circuit C such that $\mathcal{R}(C) \cong P$ if P is congruent to some element of* $\mathrm{img}(\mathcal{R})$*, and returns* `nil` *otherwise.*

Proof —

As noted before, because we can perform the DKP construction by first applying the *simplified* DKP construction, we know that (G, I, O) has a flow if $P \in \mathrm{img}(\mathcal{R})$. Similarly, if $P \in \mathrm{img}(\mathcal{R})$, then the dependencies of P will be consistent with the constraints imposed by Theorem 3-20 for some flow function. In the case where $|I| = |O|$, there is at most one flow-function; so if $P \in \mathrm{img}(\mathcal{R})$, its dependencies will in particular be consistent with the constraints of Theorem 3-20 for any flow-function discovered for (G, I, O) by `FindFlow`. Neither of these are affected by commuting independent operations of P past one another. Therefore, $\texttt{SemanticDKP}(G, I, O, P)$ returns `nil` at lines 5 or 9 of Algorithm 3-8, P is not equivalent to any $P' \in \mathrm{img}(\mathcal{R})$.

On lines 10 – 12, we initialize vertex sets V', W', and O' which will represent various sub-geometries of (G, I, O) of the form

$$(G[V'], V' \smallsetminus W', O') . \tag{3.39}$$

(More precisely, W' represents $\mathrm{img}(f')$, for successive restrictions f' of the flow-function f to V'; in the case $|I| = |O|$, this is equivalent to $V' \smallsetminus I$.) We also initialize an empty circuit C to which we pre-compose other circuits in the loops on lines 14 – 23 and lines 25 – 28, represented in stable index notation; we will sometimes refer to the tensor indices as "wire segments" (see the discussion on

Input : P, a standardized one-way measurement pattern, with an equal number of input and output qubits;
(G, I, O), the geometry underlying P (in particular, $|I| = |O|$).
Output: Either `nil`, if (G, I, O) has no flow or the dependencies of the operations of P are not consistent with the constraints of Theorem 3-20; or a unitary circuit C performing the same transformation as P, otherwise.

procedure `SemanticDKP`(G, I, O, P):
begin
  let `flow` $\leftarrow$ `FindFlow`(G, I, O);
  if `flow` $=$ `nil`
    then return `nil` ;
    else let $(f, g,$ `partialOrd`$) \leftarrow$ `flow` ;
  let $\theta : V(G) \longrightarrow \mathbb{R}$;
  for each *measurement operation* $\mathsf{M}_v^{\alpha;B}$ *in* P **do** $\theta(v) \leftarrow -\alpha$;
  if `TestDependencies`$(P, f) =$ `false` **then return** `nil`;
  let $O' \leftarrow O$;
  let $V' \leftarrow V(G)$;
  let $W' \leftarrow V(G) \setminus I$;
  let $C \leftarrow$ [the empty circuit];
  while $W' \neq \varnothing$ **do**
    let $w \leftarrow$ `MaximalStarVertex`(G, V', W', O', f);
    let $v \leftarrow g(w)$;
    let $C' \leftarrow J(\theta(v))\left[\frac{w}{v}\right]$;
    for all $z \sim w$ **such that** $z \neq v$ **do** $C' \leftarrow \wedge Z\left[w,z\right] \circ C'$;
    $C \leftarrow C \circ C'$;
    $V' \leftarrow V' \setminus \{w\}$;
    $W' \leftarrow W' \setminus \{w\}$;
    $O' \leftarrow \big(O' \setminus \{w\}\big) \cup \{v\}$;
  endwhile
  let $G' \leftarrow G[V']$;
  for all $v \in V'$ **do**
    for all $w \in V'$ **such that** $vw \in E(G')$ **do** $C \leftarrow C \circ \wedge Z\left[v,w\right]$;
    $V' \leftarrow V' \setminus \{v\}$;
  endfor
  return C;
end

ALGORITHM 3-8. An algorithm for the semantic map $\mathcal{S}$ corresponding to the DKP representation map $\mathcal{R}$, in the case of one-way measurement patterns with an equal number of input and output qubits, using the procedures defined in Algorithms 3-5, 3-6, and 3-7 (pages 125, 131, and 134 respectively).

page 32) for the sake of clarity. We will consider the circuits which we compose to C in reverse order, considering sub-circuits proceeding from the inputs to the outputs of the final circuit returned on line 29.

- In the loop on lines 25 – 28, we compose to C one controlled-Z operation between wire segments labelled by the vertices contained in V' at that stage of the algorithm, for each edge in the graph $G' = G[V']$. This loop is only entered once $W' = \varnothing$; as we only transform W' by removing one vertex w at a time on line 21 (which we also remove from V' on line 20), the value of V' just before line 25 is $V' = V(G) \setminus \mathrm{img}(f)$. Thus, the cumulative effect of lines 25 – 28 is to compose $C_{\tilde{I}}$ to the beginning of C, for $\tilde{I} = V(G) \setminus \mathrm{img}(f)$.

- Using the procedure `MaximalStarVertex` defined in Algorithm 3-7, each iteration of the loop on lines 14 – 23 produces a vertex w which is the center of a maximal star geometry of $(G[V'], V' \setminus W', O')$. For each such w, we obtain $v = g(w)$, and construct the star circuit C' corresponding to the geometry $(G_{v\star}, I_{v\star}, O_{v\star})$, which we append to the input of C. We then reduce the geometry $(G[V'], V' \setminus W', O')$ by setting $V' \leftarrow V' \setminus \{w\}$, and $O' \leftarrow O' \triangle \{v, w\}$, corresponding to the reduction described in (3.32); we also reduce $W' \leftarrow W' \setminus \{w\}$, corresponding to a further restriction of f to the new value of V'.

  By induction on $\mathrm{img}(f)$, as in the proof of Lemma 3-21, this sequence of star geometries leads to a decomposition of the original geometry (G, I, O) into star geometries, leaving behind the geometry $(G[\tilde{I}], I, \tilde{I})$. For each such star geometry, we obtain the star circuit and compose it to the input end of C; then, prior to line 25, C consists of a composition of the star circuits corresponding to the geometries in a star decomposition of (G, I, O).

Thus, the value of C on line 29 is a composition of the star circuits corresponding to the decomposition of (G, I, O) into star geometries, where in particular for each gate $J(\theta(v))^{\left[\frac{w}{v}\right]}$, we have $\theta(v) = -\alpha_v$, for α_v the angle of the measurement on v in the case where all of the preceding measurements M_z produced the result $\mathsf{s}[z] = 0$.

By construction, the output is a circuit C constructed with the elementary gate set $\textsc{Jacz}(\mathbb{A})$ for some $\mathbb{A} \subseteq \mathbb{R}$. Such a circuit is in the domain of $\mathcal{R}$; consider then the value of $\bar{P} = \mathcal{R}(C)$ for such a circuit C, and let $(\bar{G}, \bar{I}, \bar{O})$ be the geometry underlying $\bar{P}$.

- The DKP construction preserves the existing wire segment labels of C as the labels of vertices in $\bar{G}$, $\bar{I}$, and $\bar{O}$, for the geometry $(\bar{G}, \bar{I}, \bar{O})$ underlying the measurement pattern $\bar{P}$. As the labels of the wire-segments in C are precisely the vertex-labels of G, with the non-advanced indices arising from I and the non-deprecated indices arising from O, we then have $V(\bar{G}) = V(G)$, $\bar{I} = I$, and $\bar{O} = O$.

- Because the flow function f for (G, I, O) is used to define the correspondence between deprecated and advanced indices in C, and (as we noted in Section 3.2) this correspondence is a flow-function for $(\bar{G}, \bar{I}, \bar{O})$ arising from C by the DKP construction, the same function f is then a flow-function for

$(\bar{G}, \bar{I}, \bar{O})$ as well. In particular, for each pair of deprecated/advanced indices v, w in C, there is an edge $vw \in E(\bar{G})$.

- For every operation $\wedge Z\big[{}^{v,w}\big]$ in C, there will be an edge $vw \in E(\bar{G})$ which is not a flow-edge for the function f. Because each such controlled-Z operation can be attributed to a star circuit for a unique star geometry in a decomposition of (G, I, O) — as well as for the flow edges arising from the $J(\theta)$ operations — we then have $E(\bar{G}) = E(G)$, so that $\bar{G} = G$.

- Because the angle θ of the $J(\theta)$ gate between wire segments v and $f(v)$ is the negative of the default measurement angle of v in the pattern P (by the construction of θ) and the negative of the default measurement angle of v in the pattern $\bar{P}$ (by the construction of $\bar{P}$), each qubit $v \in O^{\mathsf{c}}$ has the same default measurement angle in $\bar{P}$ as in P.

- By Theorem 3-20, the measurement dependencies in $\bar{P}$ are described by the matrices $F(\mathbb{1}_n - T)^{-1}$ and $(\mathbb{1}_n - T)^{-1}$, for matrices T and F given by (3.24) and (3.25) respectively, depending on the flow-function f. However, P satisfies the same constraints by the test on lines 9. It follows that the measurement operations on v in both P and $\bar{P}$ are the same for $v \in O^{\mathsf{c}}$, and the correction operations are similarly identical for $v \in O$, including dependencies on other measurement results.

Thus, P and $\bar{P}$ contain the same operations, albeit possibly in a different order. Finally, because P is well-formed by assumption, and $\bar{P}$ is so by construction, the only possible difference in the ordering of corresponding operations between P and $\bar{P}$ is in the commuting of independent terms by the type composition constraints of Lemma 1-2. Thus, P is congruent to $\bar{P}$; and in particular, $\texttt{SemanticDKP}(G, I, O, P)$ returns `nil` only if P is not congruent to an element of $\mathrm{img}(\mathcal{R})$. ■

The above Lemma then allows us to show:

Theorem 3-23. *Let $\tilde{\mathcal{R}}$ be the restriction of the DKP construction map $\mathcal{R}$ to to unitary circuits without terminal measurements or trace-out, and with the same number of output wires as input wires; and let $\mathcal{S}$ be the map taking one-way measurement pattern P to the circuit $\texttt{SemanticDKP}(G, I, O, P)$, where (G, I, O) is the geometry underlying P. Then $\mathcal{S}$ is the semantic map for $\tilde{\mathcal{R}}$.*

Proof —

Taken as a partial function on its input data — and neglecting the trivial detail that its domain is on tuples (G, I, O, P), rather than just on patterns P — the domain of `SemanticDKP` is the set of well-formed measurement procedures which are congruent to elements of $\mathrm{img}(\tilde{\mathcal{R}})$, which in particular contains $\mathrm{img}(\tilde{\mathcal{R}})$; and each element of $\mathrm{img}(\tilde{\mathcal{R}})$ will itself be mapped by $\tilde{\mathcal{R}} \circ \mathcal{S}$ to a congruent one-way measurement pattern. Thus, `SemanticDKP` satisfies property *(iii)* on page 81.

For a circuit C for a unitary bijection composed from the gate set $\textsc{Jacz}(\mathbb{A})$ for some $\mathbb{A} \subseteq \mathbb{R}$, let $\tilde{P} = \tilde{\mathcal{R}}(C)$ and $\tilde{C} = \mathcal{S}(\tilde{P})$, and let (G, I, O) be the geometry underlying $\tilde{P}$; then, consider the correspondence between the elements of C, $\tilde{P}$,

and $\tilde{C}$. There is by construction a bijective correspondence between the wire segments of C to the vertices of $V(G)$, and from $V(G)$ to the wire segments of $\tilde{C}$. By construction, this bijective correspondence is an isomorphism of stable index expansions:

(*i*) Successive wire-segments in C correspond to paths $v, f(v), f^2(v)$ in a path-cover $\mathcal{P}_f$ in G, which in turn correspond to successive path segments in $\tilde{C}$.

(*ii*) Wire-segments in C which interact via a $\wedge Z$ gate in C correspond to neighbors $v \sim w$ in G such that $v \neq f(w)$ and $w \neq f(v)$, which in turn give rise (via the contribution of some star geometry in the decomposition of (G, I, O), as remarked in the analysis of the preceding section) to wire segments in $\tilde{C}$ which interact via a $\wedge Z$ gate. As there are no other operations in $\tilde{C}$ than this, and as every operation in C gives rise to an edge in G, the interaction graphs of C and $\tilde{C}$ are the same. Because C and $\tilde{C}$ are both defined on the same gate-set, which in particular is parsimonious (in which any multi-qubit gate preserves standard-basis states), there is a homomorphism of stable-index expressions from C to $\tilde{C}$ (where in particular the index labels are identical).

(*iii*) The wire segments in C as in $\tilde{C}$ are separated by gates of the form $J(\theta)$; and by construction, in both C and $\tilde{C}$, the angles θ of the $J(\theta)$ gates are the negative of the default measurement angle of the qubit v in $\tilde{P}$, where v is the label of the deprecated index for the $J(\theta)$ gate for C and $\tilde{C}$ alike. As there is only one variety of two-qubit gate, the homomorphism between C and $\tilde{C}$ is then an isomorphism.

Thus, C is isomorphic to $\tilde{C}$ as a quantum circuit, and differ only in the ordering of commuting gates. Thus, the map $\mathcal{S}$ is a semantic map for $\tilde{\mathcal{R}}$ in the sense defined on page 81. ■

Remarks and run-time analysis

The run-time for `SemanticDKP`(G, I, O, P) can be accounted for as follows, for $|V(G)| = n$, $|I| = |O| = k$, and $|P| = N$:

- As we showed on pages 124 and 130, `FindFlow`(G, I, O) requires time $O(k^2 n)$ and `TestDependencies`(P, f) requires time $O(Nn) \subseteq O(n^3)$.
- For each iteration of the loop on lines 14 – 23, we require time $O(kn)$ to run `MaximalStarVertex`(G, V', W', O', f), by the discussion on page 132, and time $O(\deg(v))$ (for a different $v \in O^{\mathsf{c}}$ in each iteration) for the loop on line 18; all other operations require constant time. The cumulative run time of the loop is

then

$$O\left(\sum_{v\in V(G)\setminus \mathrm{img}(f)}\Big[kn+\deg(v)\Big]\right) \subseteq O\left(\sum_{v\in V(G)} kn \;+\; \sum_{v\in V(G)}\deg(v)\right)$$
$$= O(kn^2 + 2\,|E(G)|)$$
$$= O(kn^2+2m) \;=\; O(kn^2)\,, \qquad (3.40)$$

by the extremal result of Theorem 3-7.

- For each iteration of the loop on lines 25 – 28, we perform at most $\deg(v)$ iterations of the loop on line 26, plus one constant-time set removal. As noted above, just prior to line 25, we have $V' = I$. The cumulative run time of this loop is then

$$O\left(\sum_{v\in I}\deg(v)\right) \subseteq O\left(\sum_{v\in V(G)}\deg(v)\right) = O(2\,|E(G)|)$$
$$= O(m) \subseteq O(kn)\,, \qquad (3.41)$$

again by the extremal bound of Theorem 3-7.

The time required by `SemanticDKP`(G, I, O, P) is then dominated by the time required by `TestDependencies`(P, f), which is $O(n^3)$.

3.6 Further developments

The Algorithm 3-5 of Section 3.4 is (a revision of) the first efficient algorithm to determine whether a broad class of geometries have a flow. Since the presentation of the original version of that algorithm in [44], improved algorithms — and extensions to the concept of a flow itself — have been introduced. As well, flows provide us additional tools to compare the DKP and RBB constructions of one-way measurement patterns. In this penultimate section of Chapter 3, we present an overview of these results to put the earlier results of this chapter into context.

3.6.1 Flows in relation to more general measurement-based constructions

In Sections 2.2.6 and 2.3, we presented the simplified RBB construction representing the building blocks of the patterns in [113], as well as two other patterns from the more elaborate constructions which are not obviously reproducible by the DKP construction. Using the graph-theoretic characterization of flows in this chapter, we can show that they cannot be produced by the DKP construction.

Qubit reversal pattern

The pattern described in Section 2.3.1 for reversing the order of a sequence of "logical qubits" in the cluster-state model can be shown not to have a flow, using the characterization of Section 3.3, because the underlying geometry does not admit a causal path

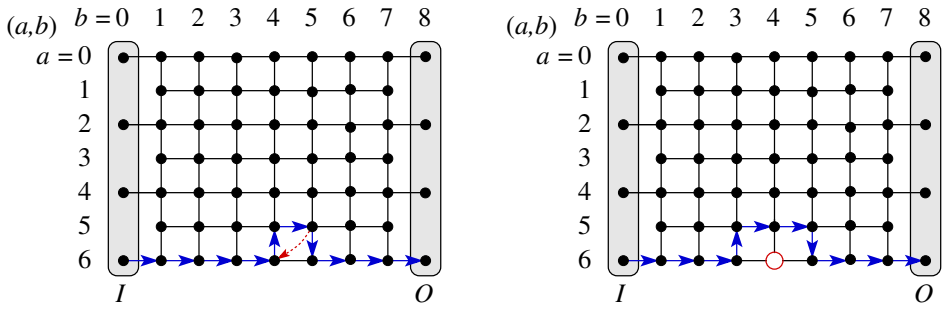


FIGURE 3-8. [*colour online*] Examples of an $I-O$ path in the geometry for the qubit-reversal pattern illustrated in Figure 2-7. Arrows on the grid lines in these diagrams represent orientations of edges of the geometry corresponding to a single $I-O$ path P_k. In the example on the left, a path leaves the row $2(k-1)$ and returns to cover all of the vertices in that row; this induces a vicious circuit (emphasized by the dotted arrow, representing an arc from the influencing digraph induced by an arc of P_k and an edge not covered by P_k). In the example on the right, a path leaves the row $2(k-1)$ and returns, but fails to cover all of the vertices in that row, leaving a vertex uncovered (shown as a large hollow circle).

cover. We may show this by sketching various constructions of families of vertex-disjoint $I-O$ paths, in the case of $|I| = |O| = k$, as follows.

In the following, we label vertices (a, b) by rows and columns; the j^{th} input vertex is in the row $2(j-1)$, and the zeroth column.

- Suppose the path P_k starting on the row $2(k-1)$ leaves that row, so that it covers the vertex $(2k-2, \ell)$ but not the edge between $(2k-2, \ell)$ and $(2k-2, \ell+1)$. Either it returns to the row $2(k-1)$ to cover the vertex $(2k-2, \ell+1)$, or it does not cover $(2k-2, \ell+1)$. In the former case, consider the function $\tilde{f}$ mapping each vertex covered by P_k to its successor in P_k; then we have $(2k-2, \ell+1) = f^h(2k-2, \ell)$ for some $h \in \mathbb{N}$. Then, the path $(2k-2, \ell) \to f(2k-2, \ell) \to \cdots \to f^h(2k-2, \ell) \to (2k-2, \ell)$ is a vicious circuit, in which case any collection of paths including P_k is not a causal path cover. In the latter case where P_k does not cover $(2k-2, \ell+1)$, it then also separates that vertex from any other $I-O$ path, as a path starting in any other row must intersect P_k to reach $(2k-2, \ell+1)$. These two cases are illustrated in Figure 3-8.

  Thus, if a path starting in the row $2(k-1)^{\text{st}}$ does not remain constrained to that row, any family of paths containing that path is not a causal path cover; and a similar analysis also holds for the path beginning in the first row. (This suffices then to show that the geometry does not have a flow *e.g.* for $k = 2$.)

- Suppose the path starting in the row $2(k-1) = 2k-2$ remains in that row. Then, the only path in a collection of vertex-disjoint $I-O$ paths that can cover vertices in the row $2k-3$ is the path P_{k-1} starting in the row $2k-4$ (as the odd numbered rows do not have input vertices, and the path P_{k-1} will separate any other path from the row $2k-3$). In order to cover all of the vertices in the row $2k-3$, the path P_{k-1} must cover the first vertex $(2k-3, 1)$ in that row; and once in that row, by a similar analysis as applied for the row $2(k-1)$ above, it must remain in that

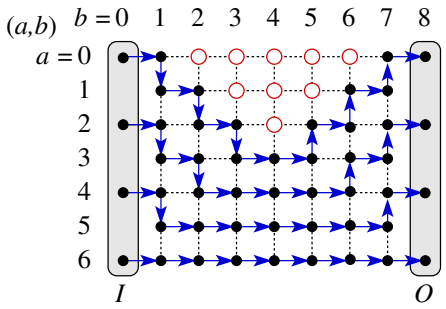


FIGURE 3-9. [*colour online*] Illustration of a family of $I - O$ paths in the the geometry for the qubit-reversal pattern illustrated in Figure 2-7. Arrows on the grid lines in these diagrams represent orientations of edges of the geometry corresponding to a family of vertex-disjoint $I - O$ paths; the other edges in the grid have been made dashed to emphasize the flow edges. This family of paths is constructed to cover all of the vertices in the higher-numbered rows; as a result, vertices in the lower-numbered rows are necessarily left uncovered (represented here by large hollow circles).

row until it reaches the final vertex $(2k-3, 2k-1)$ in that row. In order for the output vertex in row $2k-4$ to be covered, the path P_{k-1} must then terminate there, determining that path.

By induction, we can show that the paths in every lower-numbered row must follow a staircase pattern to higher-numbered rows if those higher-numbered rows are to be completely covered. (Such a family of paths is illustrated in Figure 3-9). In particular, the path P_1 will then leave the first row, so that by the previous case the resulting family of paths will not be a path cover.

Thus, the geometry of the qubit reversal pattern of [113] does not have a causal path cover, by Theorem 3-6, does not have a flow. As a result, it cannot be produced by the DKP construction.

Despite this, we may gain some information about the qubit-reversal pattern using the smenatic map for the DKP construction by an appropriate extension of the input and output systems, as exhibited in [30]. It is easy to see that if we include the left-most element of each row into I, and the right-most into O, the resulting altered geometry has a flow with a successor function which maps each vertex to the one immediately to the right. Chains in the natural pre-order $\preccurlyeq$ have monotonically increasing column numbers: and therefore $\preccurlyeq$ is antisymmetric. Given that the default measurement angle for each qubit is zero, and assuming that the dependencies of the measurement pattern are consistent with the those of the DKP construction given the flow function described, we may then obtain a measurement pattern which is in the range of the DKP construction. The geometry for this measurement pattern, and the corresponding unitary circuits in the JACZ($\mathbb{A}$) model and a related unitary circuit involving $\wedge X$ gates, are illustrated in Figure 3-10. The final circuit illustrated there can easily be described as a reversible classical computation: for seven input bits $(v_0, \ldots, v_6)$, the computation performed by that circuit is

$$\begin{aligned}&(v_0,\ v_1,\ v_2,\ v_3,\ v_4,\ v_5,\ v_6)\\ &\qquad\longmapsto\quad (v_6,\ v_4 \oplus v_5 \oplus v_6,\ v_4,\ v_2 \oplus v_3 \oplus v_4,\ v_2,\ v_0 \oplus v_1 \oplus v_2,\ v_0)\ .\end{aligned} \tag{3.42}$$

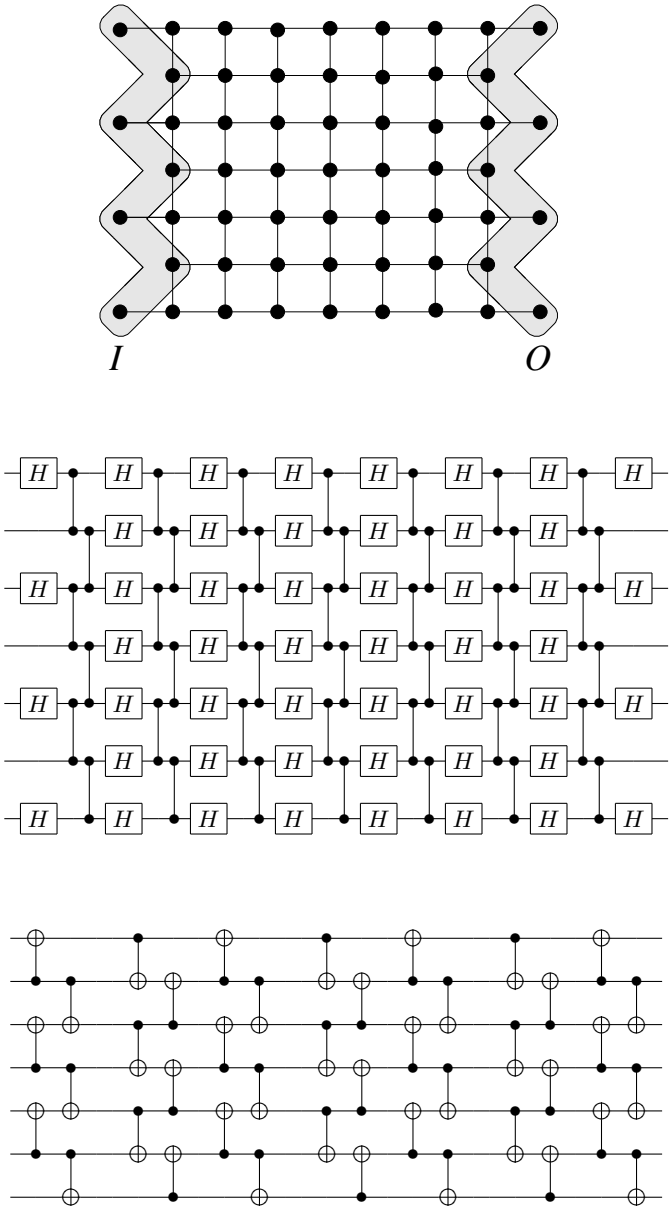


FIGURE 3-10. Illustration of the geometry for the qubit-reversal pattern illustrated in Figure 2-7 with augmented input and output subsystems, and corresponding unitary circuits. The circuit diagram in the middle represents the circuit produced by the procedure `SemanticDKP`, assuming that each default measurement is zero, and that the dependencies are those consistent with the DKP construction, given a flow function mapping each vertex in the geometry to the one immediately to the right. The circuit diagram on the bottom represents an equivalent circuit resulting from gathering the Hadamard operations of the circuit above.

If we apply this circuit instead to qubits, with the odd-indexed qubits initialized to the $|+\rangle$ state (*i.e.* a uniform superposition of the bit-values described above), we the result is that the odd-indexed qubits remain in the $|+\rangle$ state, and the even-indexed qubits are reversed, as required.

We may then understand the qubit-reversal pattern in [113] as corresponding to a circuit which prepares auxiliary qubits in the $|+\rangle$ state, and subsequently removes them by an X measurement. The fact that we are able to obtain the circuit in this case is possible because the highly structured geometry makes it easier to guess which qubit measurements correspond to terminal measurements in a unitary circuit. Without exploiting this structure, however, there are no known approaches to *algorithmically* provide semantics for such a measurement pattern in terms of unitary circuits.

Concise pattern for $Z \otimes \cdots \otimes Z$ Hamiltonians

As we noted in Section 2.3.2, the pattern $\mathfrak{Z}\mathfrak{z}_{u,v}$ on two qubits u and v of the simplified RBB construction (Section 2.2.6) is a special case of the family of patterns $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}_{v_1,\ldots,v_k}$ on many qubits. The geometry for that pattern consists of the k qubits $I = O = \{v_1, \ldots, v_k\}$, and a single auxiliary qubit $a \notin I, O$. For this geometry, there is exactly one injective function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, which is the identity function on $\{a\}$, which does not satisfy the constraint that $a \sim f(a)$. (For the sake of uniformity, we may wish to restrict ourselves to patterns performing XY plane measurements only, in which case we would consider the pattern for this CPTP map given by (2.53); however, the identity function on $\{a_1, a_2\}$ suffers from the same problem as that on $\{a\}$ for the pattern (2.47a), and the natural pre-order for the involution on $\{a_1, a_2\}$ is not antisymmetric.) Thus, this pattern cannot be produced via the DKP construction.

The YZ-plane measurement performed by the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$, with respect to the $|\pm\mathsf{yz}_\theta\rangle$ basis defined in (2.45), is equivalent to a measurement of the observable

$$\mathcal{O}^{\mathsf{YZ},\theta} \;=\; |+\mathsf{yz}_\theta\rangle\langle+\mathsf{yz}_{2\theta}| \;-\; |-\mathsf{yz}_\theta\rangle\langle-\mathsf{yz}_{2\theta}| \;=\; \cos(\theta)Z + \sin(\theta)Y\,; \tag{3.43}$$

this is a Hermitian operator which anticommutes with the Pauli X operator for any θ. We can use this to obtain an extended sense of a flow which arises from the simplified RBB construction in the same way that flows arise from the DKP construction.

3.6.2 Flows as a certificate of unitarity, and extended flows

In this section, we will consider the significance of flows not only as a structure which may underly one-way patterns, but also as a certificate which makes it possible to verify that a one-way pattern performs a unitary transformation. In doing so, we will arrive naturally at an extension of the definition of flows which describes structures underlying the simplified RBB construction, and more generally constructions of one-way patterns from $\mathfrak{J}^{\alpha}$ and $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ patterns.

Recall that for the *simplified* DKP construction, for each qubit $v \in O^{\mathsf{c}}$, the dependencies which arise are an X-dependency on $f(v)$, and a Z-dependency on every qubit

adjacent to $f(v)$ except for v itself; this may be considered to arise from a correction operation

$$B_v \quad = \quad X_{f(v)} \left(\prod_{\substack{w \sim f(v) \\ w \neq v}} Z_w \right) \tag{3.44}$$

which is absorbed into the measurements on those respective qubits, to be applied in the case that $\mathsf{s}[v] = 1$. This unitary operation closely corresponds to the generator $K_{f(v)}$ of the open graphical encoding performed by the preparation map and entanglement graph at the beginning of the measurement pattern,

$$K_{f(v)} \quad = \quad X_{f(v)} \left(\prod_{w \sim f(v)} Z_w \right) \ : \tag{3.45}$$

the state of the system just prior to the measurement phase in the simplified DKP construction is stabilized by the group $\langle K_w \rangle_{w \in I^{\mathsf{c}}}$ of such operators. Flows may then be understood as a way of certifying that a measurement pattern performs a unitary transformation by the stabilizer formalism. We may observe that the analysis of measurements by a single-qubit observable P_v in the stabilizer formalism (as presented in Section 1.5.3) does not at any point require that P_v be a Pauli operator, so long as it either commutes or anticommutes with every generator of the subgroup of the Pauli group which stabilizes the state of the system. Note that, just as we have described above for YZ-plane measurements, XY-plane measurements can be described by measurements with respect to observables of the form

$$\mathcal{O}^{\mathsf{XY},\theta} \quad = \quad |+_\theta\rangle\langle+_\theta| \quad - \quad |-_\theta\rangle\langle-_\theta| \quad = \quad \cos(\theta) X + \sin(\theta) Y \ , \tag{3.46}$$

and that this is an observable which anticommutes with the Pauli Z operator. Then, for a flow function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ for a geometry (G, I, O), an observable $\mathcal{O}_v^{\mathsf{XY},\theta}$ anticommutes with $K_{f(v)}$; and $\preccurlyeq$ describes an order in which stabilizer generators may be "removed" so that the entire computation may be treated via the stabilizer formalism.

We may then show that if $(f, \preccurlyeq)$ is a flow for a geometry (G, I, O) that $\preccurlyeq$ is a partial specification of a total ordering of the qubits $v \in O^{\mathsf{c}}$, and f a map associating each qubit $v \in O^{\mathsf{c}}$ to a generator K_w for $w \in I^{\mathsf{c}}$, such that the measurement of each such qubit v can be certified to perform an isometry by the stabilizer formalism via successive removals of the generators $K_{f(v)}$ — provided that the corresponding corrections are performed in response to each measurement, and that the measurements are performed in an ordering extending $\preccurlyeq$. In order to do this, we require that $\mathcal{O}_v^{\mathsf{XY},\theta}$ commutes for any $v \in O^{\mathsf{c}}$ with any remaining generator K_w ; for this to hold for arbitrary θ, this entails that such a generator K_w does not act on v — which can be guaranteed by requiring that any such K_w in the original stabilizer group be eliminated prior to the measurement of v. This can be done by requiring the measurement a qubit z such that $w = f(z)$ prior to measuring v. Thus, it is sufficient for all z to that $z \preccurlyeq f(z)$ and that $v \sim f(z) \implies z \preccurlyeq v$, which the flow conditions guarantee. We can then interpret flows as a way of connecting the DKP construction to the stabilizer formalism, by identifying the generators $\{K_w\}_{w \in I^{\mathsf{c}}}$ in some order as the generators by which each measurement may be treated.

The converse statement — that any ordering the individual generators $\{K_w\}_{w\in I^{\mathsf{c}}}$ as a means of treating measurement-based computation by the stabilizer formalism corresponds to a flow — is not true, however; and the $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ pattern is a counterexample. We may define an *extended flow*[18] (or *eflow*) as follows:

Definition 3-17. For a tuple (G, I, O, T) consisting of a geometry (G, I, O) and a set $T \subseteq O^{\mathsf{c}} \cap I^{\mathsf{c}}$ of qubits to be measured in the YZ-plane (all other qubits assumed to be measured in the XY-plane), an *extended flow* is an ordered pair $(f, \preccurlyeq)$ consisting of a function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$, and a partial order $\preccurlyeq$ on $V(G)$, such that the conditions

$$f(v) \sim v\,, \quad \text{for } v \in T^{\mathsf{c}}\,; \tag{3.47a}$$
$$f(v) = v\,, \quad \text{for } v \in T\,; \tag{3.47b}$$
$$v \preccurlyeq f(v)\,; \quad \text{and} \tag{3.47c}$$
$$w \sim f(v) \implies v \preccurlyeq w \tag{3.47d}$$

hold for all $v \in O^{\mathsf{c}}$ and $w \in V(G)$.

The special case of qubits indicated as being measured in the YZ-plane applies in the case of $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ as defined in (2.47a): the identity function on $\{a\}$ is an "eflow function", with equality being an "eflow order". This corresponds to an identification of the operator K_a as a generator of the open graphical encoding with which $\mathcal{O}^{\mathsf{YZ},\theta}$ anticommutes for any θ. This indicates that the unitarity of the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ may also be certified by the stabilizer formalism.

In the same way as for flows, an eflow for a tuple (G, I, O, T) certifies that any pattern on with the geometry (G, I, O) and performing the appropriate types of measurement will perform a unitary transformation, provided also that it performs the appropriate corrections (or measurement adaptations) in response to the measurements. Again as for flows, eflows do this by attributing generators $\{K_w\}_{w\in I^{\mathsf{c}}}$ to qubit $v \in O^{\mathsf{c}}$ to be measured. To show this, it suffices to show that the observables for each measurement commutes with the generators which remain at the point that the measurement is performed, except for a single one with which it anticommutes. The argument is similar to the one for flows: it is necessary and sufficient for $K_{f(v)}$ to be the only generator which acts on v at the time of the measurement of v; which holds if and only if v is measured no earlier than any qubit z for which $v \sim f(z)$ or $v = f(z)$, which are guaranteed for eflows as for flows.

Any attribution of generators $K_{f(v)}$ for $f(v) \in I^{\mathsf{c}}$ (as opposed to nontrivial products of such operators) to qubits $v \in O^{\mathsf{c}}$ must satisfy $f(v) = v$ or $f(v) \sim v$ in order for $K_{f(v)}$ to act on v; and in order to anticommute with the appropriate measurement observable describing the measurement on the qubit v for *arbitrary* default measurement angles, we require precisely the eflow conditions to hold. Thus, eflows characterize those geometries, together with specifications of measurement planes, which may underlie measurement patterns which can be certified to perform unitary transformations by the stabilizer formalism.

[18] This extended sense of flow was described in [37] for the special case of Y measurements, but does not seem to have been seriously examined, having been superceded by the concept of generalized flows (or *glows*) [26] which we will discuss in Section 3.6.5.

The simplified RBB construction give rise to tuples (G, I, O, T) with eflows just as the DKP construction gives rise to geometries with flows: the Z corrections in the pattern $\mathfrak{Z}\mathfrak{z}$ of (2.41) used for two-qubit interactions commute with all of the entangler maps of subsequent measurement patterns which are composed with it, and induce π-dependencies on the qubits they act upon; the same informal meaning of influencing the measurement result applies as that described in Section 3.2. After Pauli simplifications and signal shifting, we may arrive at a similar characterization of the dependencies arising in the simplified RBB construction as that described in Theorem 3-20 for the DKP construction: the same matrices T and F describe the propagation of X- and Z-dependencies in that case. As well, the analysis of star decompositions may similarly be generalized: we may augment the definition of a star geometry to include those, like that of the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$, for which $I = O$ and for which I^{c} is a singleton set consisting of a qubit to be measured in the YZ-plane; the decomposition of geometries (G, I, O) into maximal geometries then generalizes straightforwardly. Then, provided an algorithm to efficiently find eflows, we may similarly arrive at a semantic map for the simplified RBB construction — or for any augmentation of the DKP construction by including patterns of the form $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ — just as we have above for the DKP construction.

3.6.3 A faster and more general algorithm for finding flows

The algorithm `FindFlow` for finding flows presented in Section 3.4 runs in time $O(k^2 n)$, where $k = |I| = |O|$ and $n = |V(G)|$; in particular, it is constrained to the case where the input and output subsystems are of the same size. In [96], Mhalla and Perdrix present an algorithm which is both more general than `FindFlow`, in that it can also determine the presence of a flow for geometries such that $|I| < |O|$,[19] and which runs in time $O(kn)$.

The speed-up associated with the algorithm of [96] essentially arises out of the observation that the flow function f and partial order $\preccurlyeq$ can be found simultaneously, by considering the length of the longest chains of each qubit from the output set, and constructing "layer sets" consisting of vertices of similar depth. The main analytical tool for the algorithm is that of a "maximally delayed" flow:

Definition 3-18 [96, Definitions 4 & 5]**.** For a geometry (G, I, O) and a flow $(f, \leqslant)$, let $V_0^{<}(G)$ be the set of vertices of $V(G)$ which are maximal in $\leqslant$, and let $V_{j+1}^{<}(G)$ be the maximal set of vertices of the set

$$W_j^{<} \;=\; V(G) \setminus \left(\bigcup_{\ell=0}^{j} V_j^{<} \right) \tag{3.48}$$

in the order $\preccurlyeq$, for any $j > 0$. A flow $(f, \preccurlyeq)$ is *more delayed* than another causal order $(f', \leqslant)$ if we have $\left|W_j^{\preccurlyeq}\right| \leqslant \left|W_j^{<}\right|$ for all $j \in \mathbb{N}$, where this inequality is strict for at least one such j; and $(f, \preccurlyeq)$ is *maximally delayed* if no flow for (G, I, O) is more delayed.

[19] Note that a geometry with $|I| > |O|$ cannot have a flow, as the set of input vertices must be a subset of the initial vertices of a path cover, and the set of output points must coincide with the set of final vertices of a path cover.

Mhalla an Perdrix show that a maximally delayed flow is also a flow of minimum depth, *i.e.* that the causal order of such a flow has the same depth as the natural pre-order for the given flow-function f. For a maximally delayed flow $(f, \preccurlyeq)$, let $\ell : V(G) \longrightarrow \mathbb{N}$ be the function mapping each vertex $v \in V(G)$ to the integer j such that $v \in V_j^{\prec}$. Then, the partial order $\leqslant$ characterized by $v \leqslant w \iff [v = w \text{ or } \ell(v) > \ell(w)]$ also yields a minimum-depth flow by construction. To find a flow of minimum depth, it then suffices to find the sets $V_j^{\prec}$ of a maximally delayed flow $(f, \preccurlyeq)$.

Mhalla and Perdrix show for a maximally delayed flow $(f, \preccurlyeq)$ for (G, I, O) that $V_0^{\prec} = O$, and $V_1^{\prec}$ is the set of elements of $W_0^{\prec}$ such that, for some $w \in O$, v is the only neighbor of w which is contained in $W_0^{\prec}$. By induction (by augmenting the set O to include successive "layers" $V_j^{\prec}$), one may show that we then have

$$V_{j+1}^{\prec} \;=\; \left\{ v \in W_j^{\prec} \;\middle|\; \exists w \in V(G) \setminus W_j^{\prec} \,:\, N_G(w) \cap W_j^{\prec} = \{v\} \right\} , \tag{3.49}$$

where again $N_G(w)$ is the set of vertices adjacent to w in G. By a depth-first search from O, we may find in a finite number of iterations of j those qubits $w \in V(G) \setminus W_j^{\prec}$ which have such a neighbor v, and accumulate them into new layers $V_{j+1}^{\prec}$.

This decomposition of the geometry (G, I, O) into layers $V_j^{\prec}$ corresponds strongly to the star decomposition of the geometry (G, I, O) into maximal star geometries; if (as in [96]) we accumulate sets $V_j^{\prec}$ to produce successively larger output subsystems $O^{(j)} = V(G) \setminus W_j^{\prec}$, the set $V_{j+1}^{\prec}$ as described above are precisely those vertices which are the root of a maximal geometry.[20] Thus, in addition to finding flows more efficiently, the algorithm of [96] may also be used to unify the stages of finding a flow for the geometry underlying a pattern P, and obtaining the corresponding unitary circuit for P, in a semantic map for the DKP construction when $|I| = |O|$.[21]

3.6.4 An efficient algorithm for finding extended flows

We may easily extend the analysis and the algorithm of [96] for flows to also efficiently find eflows. For an eflow $(f, \preccurlyeq)$ for a tuple (G, I, O, T), define the sets $V_j^{\prec}$ and $W_j^{\prec}$ as in Definition 3-18. Then for each $j \in \mathbb{N}$, $V_{j+1}^{\prec}$ consists of those vertices $v \in W_j^{\prec}$ such that, for any $w \in V(G)$ such that $w = f(v)$ or $w \sim f(v)$, we have $w = v$ or $w \in V(G) \setminus W_j^{\prec}$. For $v \notin T$, this implies that (as in the analysis of [96]) the only neighbor of $f(v)$ which lies in $W_j^{\prec}$ is v itself; and for $v \in T$, this implies that v has no neighbors in $W_j^{\prec}$. Algorithm 1 of [96] can then be easily extended to test for the latter condition, yielding a procedure to construct an eflow. We will now describe such an algorithm.

Algorithm 1 of [96] operates on the basis of maintaining a set of "correcting vertices" which, for each new layer, may potentially be the successor of some vertex in the new layer. We take a similar approach, but must somehow accomodate the vertices of T, whose neighbors we must check *before* including them in a layer. In order to ensure that this modification does not substantially increase the runtime of the flow-finding algorithm

[20] While the algorithm of [96] predates the analysis of star geometries presented in this thesis, the two results were developed independently.

[21] In the case that $|I| < |O|$, there are additional difficulties which arise due to the fact that there need not be a unique flow-function — which implies that there is not a unique set of X- and Z-dependencies which must arise for measurement patterns with such geometries. We discuss these issues in Section 3.7.

Data : ℓ, an array $V(G) \longrightarrow \mathbb{N}$ for storing layers of vertices of a graph G
W, the set of vertices for which ℓ is not yet well-defined
`layer`, the value of the current layer
`emptyLayer`, a flag indicating if the present layer is empty
D, a directed graph such that vertices with non-zero in-degree are in W
T, a set of vertices representing qubits measured in the YZ-plane
f, an array $V(G) \longrightarrow V(G)$ for storing "successors" of vertices
S', a set of vertices to examine as possible "successors" in the next layer
$v \in V(G)$, a vertex to add to the current layer
$w \in V(G)$, a vertex to be designated the "successor" of v

Effect: Sets $f(v) \leftarrow w$, $\ell(v) \leftarrow$ `layer`, removes all of the arcs in D into v, and checks the neighbors of v in D for terminal vertices in T to include into S' (for inclusion into the next layer)

```
1  subroutine AddToLayer(v, w):
2  begin
3      emptyLayer ← false;
4      f(v) ← w ; ℓ(v) ← layer;
5      W ← W ∖ {v};
6      for all z ∈ neighbors(D, v) do
7          neighbors(D, z) ← neighbors(D, z) ∖ {v};
8          if z ∈ T and neighbors(D, z) = ∅ then S' ← S' ∪ {z};
9      endfor
10 end
```

ALGORITHM 3-9. A subroutine to attribute a vertex to the current layer, for use in an iterative procedure for finding the layer sets of a maximally-delayed eflow. This subroutine is used in particular to maintain the arcs of a digraph D (a "sub-diagraph" of a graph G, interpreted as a symmetric digraph) from which we remove any arc ending in a vertex with a well-defined layer.

of [96], it will prove helpful to maintain a digraph whose arcs point in either direction along each edge of G, and from which we delete any arc ending at a vertex with a defined layer. As a result, we may easily determine whether a vertex with a defined layer has a single neighbor which does not yet have a defined layer, and also easily determine whether a vertex without a defined layer has any neighbors without a defined layer. Algorithm 3-9 defines a subroutine `AddToLayer` to perform the necessary edge deletions whenever we define the layer of a vertex, as well as several other operations useful to an iterative procedure for building an eflow by layer-sets, by operating on several global variables.

Using the subroutine `AddToLayer`, we may easily define a procedure similar to Algorithm 1 of [96] to build an eflow by layer sets. We begin by creating the digraph D, copying the neighborhood relations of G, but removing any arcs ending in O (which is necessarily the zeroeth layer). Any output vertex which is in the range of a "successor" function $f : O^{\mathsf{c}} \longrightarrow I^{\mathsf{c}}$ is a potential successor: we define a set of candidate successors from the elements of $O \setminus I$. Vertices without a defined layer (initially the elements of O^{c}) are included in the set W. For each layer, we iterate through the set of potential successors, which initially includes no elements of T; we maintain the invariant that $v \in T$

Input : G, a graph with subsets I, O, T, where $T \subseteq V(G) \setminus O$.
Output: Either `nil`, if (G, I, O, T) does not have an eflow; or an ordered pair (f, ℓ) consisting of a partial function $f : V(G) \rightharpoonup V(G)$ and a function $\ell : V(G) \to \mathbb{N}$ such that $(f, \preccurlyeq)$ is an eflow, for a partial order $\preccurlyeq$ given by $v \preccurlyeq w \iff [\, v = w$ or $\ell(v) > \ell(w)\,]$.

```
1  procedure FindEflow(G, I, O, T):
2  begin
3      let f : V(G) ⟶ V(G);
4      let ℓ : V(G) ⟶ ℕ;
5      let W, S ← ∅;
6      let D ← [ completely disconnected digraph on V(G) ];
7      for all v ∈ V(G) do
8          neighbors(D, v) ← neighbors(G, v) ∖ O;
9          if v ∈ O then
10             ℓ(v) ← 0 ; f(v) ← nil;
11             if v ∉ I then S ← S ∪ {v};
12         else
13             W ← W ∪ {v};
14         endif
15     endfor
16     let layer ← 0;
17     let emptyLayer ← false;
18     repeat
19         emptyLayer ← true ; layer++;
20         let S′ ← ∅;
21         for all w ∈ S do
22             if neighbors(D, w) = {v} such that v ∉ T then
23                 S′ ← S′ ∪ {v};
24                 AddToLayer(v, w);
25             else if w ∈ T then
26                 AddToLayer(w, w);
27             else
28                 S′ ← {w};
29             endif
30         endfor
31         S ← S′;
32     until emptyLayer ;
33     if W = ∅
34         then return (f, ℓ) ;
35         else return nil;
36 end
```

ALGORITHM 3-10. An iterative algorithm to obtain an eflow, using the maximally-delayed layers technique of [96] and the subroutine `AddToLayer`.

becomes a candidate successor only if it has out-degree zero (*i.e.* all of its neighbors are in some defined layer). For any viable successor w — one satisfying the eflow constraints that $w \in T$ and $w \prec u$ implies that $u \in V_j^{\prec}$ for some $j <$ `layer`, or that $w = f(v)$ for v the unique vertex not in $V^{|}prec_j$ for some $j <$ `layer` — we assign w as the successor of the appropriate vertex v, and define the layer of that vertex v. If this vertex v is different from w, it then becomes a potential successor for the next layer; and any neighbors of v which are in T, and which (subsequent to the removal of any arcs ending in v) have out-degree zero, also become viable successors for the next layer. Otherwise, if w was not a viable successor for this layer, we carry it forward for the next layer. We assign the list of new candidate successors to S before the end of the iteration; and we repeat until we cannot assign any new vertices to a layer.

If we cannot assign any vertices to new layers, this is either because we have exhausted the vertices (in which case $W = \varnothing$), or because there are no vertices which are reachable from the existing layers which have viable successors. In the latter case, there is no eflow; and in the former, we return the eflow "sucessor" function and the layer function.

Analysis of Algorithm 3-10

In the subroutine `AddToLayer`(v, w), the run-time is dominated by the loop over the neighbors z of v, which requires time at most $O(\deg_G(v))$. This subroutine is called at most once for every vertex $w \in V(G)$: then, the cumulated run-time of `AddToLayer`(v, w) over the execution of `FindEflow` is $O\big(\sum_v \deg_G(v)\big) = O(m)$, where $m = |E(G)|$.

Let $n = |V(G)|$ and $k = |O|$. Initially, there are $k = |O|$ elements of S, none of which are elements of T; and as every element $w \in S \setminus T$ corresponds to a unique element of $S' \setminus T$ (either itself, or the vertex v for which w is the successor), there always remain at most k elements of $S \setminus T$ at each iteration of the loop on lines 18 – 32. The comparisons on lines 22 and 25 can both be performed in constant time using the set data structures described at the beginning of Section 3.4, as can the set-inclusions on lines 23 and 28. The assignment on line 31 can be performed simply by re-attributing the element list of S' to S, as we will immediately re-assign S' to the empty set in any subsequent iteration. There are at most n iterations of the loop on lines 18 – 32, as it only repeats as long as at least one vertex has been assigned to a new layer. Then, setting aside the work performed for vertices $w \in S \cap T$ in the loop on lines 18 – 32 and the work performed by `AddToLayer`, the work performed by that loop is $O(kn)$.

As elements of T are only included into S' when they are guaranteed to be attributed to the next layer, the total work performed by that loop (again aside from work performed by `AddToLayer`) for elements of T is then $O(n)$. The run-time of `FindFlow`(G, I, O, T) is then $O(kn + m)$, dominated by the work performed by `AddToLayer`(v, w) for each $w \in V(G)$, and by the iteration of the loop on lines 18 – 32 for at most n layers.

This is precisely the run-time described for Algorithm 1, except that Mhalla and Perdrix also use the extremal result of Theorem 3-7 presented in [47] to simplify this to $O(kn)$. It is an open question whether there is a similar extremal result for eflows.

As we remarked following Definition 3-17, an efficient algorithm for eflows allows us to obtain a semantic map for the simplified RBB construction using similar techniques as we have presented for the DKP construction throughout the Chapter, as well as for

any extension of the DKP or simplified RBB constructions which makes use of the more general $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ pattern.

3.6.5 Generalized flows

In Definition 3-17, we have described one sense in which flows may be generalized, by observing that the role of flows is essentially to permit the stabilizer formalism to certify that the measurement pattern performs a unitary operation by attributing some generator K_w to each qubit $v \in O^{\mathsf{c}}$ to be measured. A more general approach than this was defined in [26] in *generalized flows* (or *gflows*), which extends this both to arbitrary generators of the initial stabilizer group $\langle K_w \rangle_{w \in I^{\mathsf{c}}}$ (*i.e.* to non-trivial products of the operators K_w) and to multiple planes of measurement.

Analogously to XY-plane measurements and YZ-plane measurements as we have described above, we may define XZ-*plane measurements* to be measurements with respect to a basis of the form

$$|\pm\mathsf{xz}_\theta\rangle \;=\; \frac{1}{\sqrt{2}}\,|{+}\mathsf{y}\rangle \;\pm\; \frac{\mathrm{e}^{i\theta}}{\sqrt{2}}\,|{-}\mathsf{y}\rangle\;. \tag{3.50}$$

(For the duration of this section, we will write $|\pm\mathsf{xy}_\theta\rangle = |\pm_\theta\rangle$ for the sake of uniformity.) Generalized flows encode a set of conditions imposed on measurements in the XY, YZ, and XZ planes so that, as we described in Section 3.6.1, each observable to be measured anticommutes with exactly one of the remaining generators of the initial stabilizer group, commuting with the rest. Let $\lambda : O^{\mathsf{c}} \to \{\mathsf{XY}, \mathsf{YZ}, \mathsf{XZ}\}$ be a function which designates, for each qubit to be measured, the plane in which it is to be measured. The generalized flow conditions are as follows:

Definition 3-19 (paraphrased from [26])**.** Let (G, I, O) be a geometry and λ be a function defined on O^{c} indicating measurement planes as above; and let $\mathcal{P}$ be the power-set function, and $\mathrm{odd}(S)$ be the set of vertices in G which are adjacent to an odd number of elements of S. Then a *gflow* for (G, I, O, λ) is a tuple $(g, \preccurlyeq)$ consisting of a function $g : O^{\mathsf{c}} \longrightarrow \mathcal{P}(I^{\mathsf{c}})$ and a partial order $\preccurlyeq$ which satisfy the following conditions:

$$w \in g(v) \implies v \preccurlyeq w, \tag{3.51a}$$
$$w \in \mathrm{odd}(g(v)) \implies v \preccurlyeq w, \tag{3.51b}$$
$$\lambda(v) = \mathsf{XY} \implies v \in \mathrm{odd}(g(v)) \smallsetminus g(v), \tag{3.51c}$$
$$\lambda(v) = \mathsf{YZ} \implies v \in g(v) \smallsetminus \mathrm{odd}(g(v)), \tag{3.51d}$$
$$\lambda(v) = \mathsf{XZ} \implies v \in g(v) \cap \mathrm{odd}(g(v)). \tag{3.51e}$$

The conditions above are sufficient conditions for there to exist a measurement pattern, with underlying geometry (G, I, O) and performing measurements in the planes specified by λ, which performs a unitary for all specific choices of measurement angles. We may show this as follows.

We represent measurements in the XY, YZ, and XZ planes by observables $\mathcal{O}^{\mathsf{XY},\theta}$, $\mathcal{O}^{\mathsf{YZ},\theta}$, and $\mathcal{O}^{\mathsf{XZ},\theta}$ respectively, which (like Pauli operators) have eigenvalues ± 1, with

corresponding eigenvectors $|\pm\mathsf{xy}_\theta\rangle$, $|\pm\mathsf{yz}_\theta\rangle$, and $|\pm\mathsf{xz}_\theta\rangle$ respectively:

$$\mathcal{O}^{\mathsf{XY},\theta} = |+\mathsf{xy}_\theta\rangle\langle+\mathsf{xy}_\theta| - |-\mathsf{xy}_\theta\rangle\langle-\mathsf{xy}_\theta| = \cos(\theta)X + \sin(\theta)Y \; ; \tag{3.52a}$$

$$\mathcal{O}^{\mathsf{YZ},\theta} = |+\mathsf{yz}_\theta\rangle\langle+\mathsf{yz}_\theta| - |-\mathsf{yz}_\theta\rangle\langle-\mathsf{yz}_\theta| = \cos(\theta)Z + \sin(\theta)Y \; , \tag{3.52b}$$

$$\mathcal{O}^{\mathsf{XZ},\theta} = |+\mathsf{xz}_\theta\rangle\langle+\mathsf{xz}_\theta| - |-\mathsf{xz}_\theta\rangle\langle-\mathsf{xz}_\theta| = \cos(\theta)X + \sin(\theta)Z \; . \tag{3.52c}$$

After applying the preparation and entangling operations $\mathsf{E}_G\mathsf{N}_{I^c}$ in a one-way measurement pattern, the state-space is stabilized by the group $\mathcal{S}_{G,I} = \langle K_v \rangle_{v\in I^c}$. As we noted in Section 3.6.1, to describe how measurement of these observables transform the space stabilized by $\mathcal{S}_{G,I}$, the analysis of measurements by a single-qubit observable P_v in the stabilizer formalism requires only that each measurement observable either commutes or anticommutes with every generator of the subgroup of the Pauli group which stabilizes the state of the system. If in particular, for all $1 \leqslant t \leqslant |O^c|$, the state of the system after the j^{th} measurement is stabilized by a group $\mathcal{S}^{(t)}$, such that the $t+1^{\text{st}}$ measurement operator anticommutes with at least one element of $\mathcal{S}^{(t)}$, the evolution of the system at each stage will be unitary, and the state post-measurement (selecting for the $+1$ measurement result) will be stabilized by a subgroup $\mathcal{S}^{(t+1)}$ of the group $\mathcal{S}^{(t)}$.

Suppose we require that the system at all times be in a joint $+1$-eigenstate of some subgroup of $\mathcal{S}$ after each measurement. It then suffices to find a generating set $\{S_1, \ldots, S_{|I|^c}\}$ for $\mathcal{S}_{G,I}$ such that the state of the system after the t^{th} measurement (and a classically controlled correction, if necessary) is stabilized by $\langle S_j \rangle_{j>t}$. In particular, we require S_t to be the generator which anticommutes with the observable of the t^{th} measurement (*i.e.* with $\mathcal{O}_v^{\mathsf{XY},\theta}$, $\mathcal{O}_v^{\mathsf{YZ},\theta}$, or $\mathcal{O}_v^{\mathsf{XZ},\theta}$ as appropriate), and for every S_j to commute with that observable for $j > t$, *regardless* of the value of θ. For a measurement on a qubit v, this imposes the following constraints:

- for an XY-plane measurement, we require that S_k acts on v with a Z operation, and S_j acts on v with the identity operation for $j > k$;
- for a YZ-plane measurement, we require that S_k acts on v with an X operation, and S_j acts on v with the identity operation for $j > k$;
- for an XZ-plane measurement, we require that S_k acts on v with an Y operation, and S_j acts on v with the identity operation for $j > k$.

In particular, we require that the stabilizer groups $\langle S_j \rangle_{j>k}$ for each $1 \leqslant k \leqslant |O^c|$ do not act on the first k qubits which were measured.

From these constraints, we can re-derive the gflow conditions, as follows. Consider a fixed choice of generating set $\{S_j\}_{j=1}^{|I^c|}$ for which the conditions above may be satisfied, for *some* ordering of the generators: each such operator consists of a product of operators K_v for $v \in I^c$. Define the function $\gamma : \{1, \ldots, |O^c|\} \longrightarrow \mathcal{P}(I^c)$ such that

$$\begin{aligned} S_j = \prod_{v\in\gamma(j)} K_v &= \prod_{v\in\gamma(j)} \left(X_v \prod_{w\sim v} Z_w \right) \\ &\propto \left[\prod_{v\in\gamma(j)} X_v \right] \left[\prod_{w\in \text{odd}(\gamma(j))} Z_w \right] . \end{aligned} \tag{3.53}$$

Let $\sigma : O^c \longrightarrow \mathbb{N}$ describe an arbitrary sequence in which the qubits of O^c are measured; let $\preccurlyeq$ describe the coarsest partial order (relative to the choice of generators $\{S_j\}_{j=1}^{|I^c|}$) consistent with every such ordering of O^c; and let $g = \gamma \circ \sigma$. In order for the commutation/anticommutation properties above to hold, we then require the following for the qubit v which is the k^{th} qubit measured:

(*i*) As S_k does not act on any qubits yet to be measured, any qubit $w \in \gamma(k)$ must be measured later than v, and similarly for any qubit $w \in \text{odd}(\gamma(k))$. Therefore, if either $w \in g(v)$ or $w \in \text{odd}(g(v))$, we require $v \preccurlyeq w$.

(*ii*) If v is to be measured in the XY plane, S_k must act on v with a Z operation, in order for the observable $\mathcal{O}_\theta^{\mathsf{XY}}$ to anticommute with S_k; this holds if and only if $v \in \text{odd}(g(v))$ and $v \notin g(v)$.

(*iii*) If v is to be measured in the YZ plane, S_k must act on v with an X operation, in order for the observable $\mathcal{O}_\theta^{\mathsf{YZ}}$ to anticommute with S_k; this holds if and only if $v \in g(v)$ and $v \notin \text{odd}(g(v))$.

(*iv*) If v is to be measured in the XZ plane, S_k must act on v with an Y operation, in order for the observable $\mathcal{O}_\theta^{\mathsf{XZ}}$ to anticommute with S_k; this holds if and only if $v \in g(v)$ and $v \in \text{odd}(g(v))$.

Then, the conditions for applying the extended stabilizer formalism independently of the measurement angles are precisely those of of Definition 3-19.

As noted above, if each measurement results in a collapse to their respective $+1$-eigenvalues, the resulting transformation of the state-space is an isometry; and for any t such that the t^{th} measurement results in the -1-eigenvalue of the measurement observable, we may apply S_t as a correction operation to steer it into the post-measurement state corresponding to the $+1$ measurement result.

Therefore, the existence of a gflow for (G, I, O, λ) acts as a certificate that a measurement pattern which performs the appropriate corrections (or measurement adaptations) performs a unitary transformation, via the stabilizer formalism. This fully generalizes the extent to which the stabilizer formalism (relaxed to admit non-Pauli observables) can be used to define certificates for unitarity without requiring precise information about individual measurement observables. In particular, eflows are a special case for gflows, where we require that $g(v)$ is a singleton set for all $v \in O^c$; and flows are a further restriction where we require $v \neq f(v)$ for all $v \in O^c$.

It is not known whether gflows correspond to any construction of interest for one-way measurement patterns (in the sense that flows correspond to the DKP construction and eflows to the simplified RBB construction, as described in Sections 3.2 and 3.6.1 respectively); however, as an extension of flows, generalized flows seem to extend the semantic map `SemanticDKP` to accommodate measurement patterns which may be translated into circuits using closed time-like curves [78] (a notion introduced into quantum information theory by Deutsch [53]).[22] By virtue of having an eflow, the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}$ of (2.47a) has

[22] In general, the availability of closed time-like curves are a powerful computational resource for both classical and quantum computation [3]; however, the extension of `SemanticDKP` to patterns with gflows suggested by Kashefi does not seem to produce arbitrary circuits which use closed time-like loops (*c.f.* the discussion in the introduction to Chapter 4 on postselection).

a gflow. However, for any measurement pattern whose corresponding tuple (G, I, O, λ) has a gflow, every measurement will produce independent and maximally random measurement results; therefore, the qubit-reversal pattern of Section 2.3.1 (which, as we noted on page 74, yields some measurement results with certainty conditioned on prior results) does not have a gflow.

In [96], Mhalla and Perdrix describe an $O(n^4)$ algorithm (where $n = |V(G)|$) for determining whether a geometry (G, I, O) has a gflow, conditioned on every qubit being measured in the XY-plane; it is not known whether there exists an efficient algorithm for finding a gflow under more general conditions, although this seems likely.

3.6.6 Ignoring measurement dependencies when inferring unitary circuits

In the preceding discussion of flows, eflows, and gflows as certificates of unitarity in terms of the stabilizer formalism, we have quickly brushed over the issues of the appropriate measurement adaptations and corrections depending on measurement results. The stabilizer formalism itself furnishes a choice of immediate corrections which suffice to guarantee unitarity, which can then be absorbed into measurements and further refined by Pauli simplifications and signal shifting; however, in situations where there exist more than one "flow-like" certificate (which may occur for flows when $|I| < |O|$, or for gflows), it may be difficult to verify whether a particular measurement pattern meets conditions which are sufficient to show that it performs a unitary transformation. However, if the geometry underlying a particular measurement pattern (together with additional information such as the planes of measurement for each qubit) admits a flow-like certificate, we may if we wish consider the scheme of dependencies which arise from that certificate in order to "force" the pattern to represent of a unitary transformation, changing the measurement/correction dependencies as required. Because the dependencies serve only to simulate the selection of a particular post-measurement result, the particular dependency scheme does not matter (except possibly for complexity reasons): then one one arising from an arbitrary certificate of this kind will suffice. We will consider a potential application of this approach in Chapter 4.

If we are willing to ignore any existing dependencies in a measurement pattern and simply impose the dependencies arising from some flow-like certificate, we may streamline any algorithm for semantic maps corresponding to constructions of one-way measurement patterns by omitting the step for checking the dependencies, which *e.g.* dominates the run-time of Algorithm 3-8.

3.7 Review and open problems

In this chapter, we have reviewed the concept of flows presented in [38], identifying it as arising out of combinatorial constraints on expressions for unitary circuits. We have presented a characterization of flows in terms of path covers, and presented uniqueness and extremal results, which contributed to the discovery of the first efficient algorithm for finding flows for geometries of one-way measurement patterns, and also help to bound the run-time of the best existing algorithms for finding flows. We used these results to describe a function $\mathcal{S}$ which provides semantics for mapping measurement patterns

congruent to those produced by the DKP construction, in terms of unitary circuit models $\textsc{Jacz}(\mathbb{A})$ for $\mathbb{A} \subseteq \mathbb{R}$ defined in Section 1.3, when those patterns have input and output subsystems of the same size.

The final constraint above in the semantic map $\mathcal{S}$, that $|I| = |O|$, does not merely arise out of the corresponding constraint for the procedure `FindFlow` of Algorithm 3-5. In order to verify that the measurement and correction dependencies for a pattern suffice for the pattern to perform a unitary, the condition $|I| = |O|$ provides the guarantee (by Theorem 3-11) that there is at most one flow-function, and therefore at most one set of dependencies which are possible after Pauli simplifications and signal shifting. For a pattern where $|I| < |O|$ whose geometry admits more than one flow-function, it becomes necessary to verify that there exists a flow function which gives rise to the dependencies observed in the pattern. It is possible that the dependency information in the pattern may prove useful for discovering whether such a flow-function exists, but no such algorithm is known.

As we have observed at many points throughout this Chapter, with regards to uniqueness, extremal results, and decompositions in terms of star geometries, the presence of a flow gives rise to many structural constraints, some of which limit the possible complexity of the measurement pattern. It seems plausible that these may lead to interesting bounds on the size and depth complexity of measurement patterns arising from the DKP construction, or on the time required to compute the dependency matrices described in Theorem 3-20.

Chapter 4

Measurement Pattern Interpolation

from Quadratic Form Expansions

Can measurement-based quantum computation provide new idioms for designing quantum algorithms? That is: in addition to being universal for quantum computation by *e.g.* representing unitary circuits via the DKP or RBB constructions, can it also provide analytical tools which suggest how currently *unsolved* problems might be solved on a quantum computer? There are currently no obvious high-level concepts of the sort provided by the circuit model (such as eigenvalue estimation or amplitude amplification) or quantum walks (such as deflection by engineered energy barriers) which may lead to interesting quantum algorithms; however, the ability to describe novel solution techniques is part of what distinguishes a potentially useful model of computation from a model which is "merely" a potentially useful scheme for physical implementation.

The largest obstacle to high-level descriptions of algorithms in the measurement based model — without depending on semantics being provided by another model, *e.g.* a unitary circuit model — seems to be in the details of the classical feed-forward which specifies how measurements are to be adapted based on previous measurement results. To perform a unitary transformation with a measurement-based algorithm requires in practise a case analysis at nearly every measurement, which is a distraction from any potential "uniform" description of the computation. Therefore, a likely prerequisite for developing intuitions about how to solve problems in measurement-based models is to abstract away this detail of measurement-based computation.

By definition, in a measurement-based procedure which performs a unitary transformation, the final quantum state is independent of the measurement results, even if the operations performed to obtain that state were not. This suggests an approach where we omit all details of the measurement results or the probabilities of them occurring, except for some particular "distinguished" sequence of measurement results which can occur with non-zero probability. For instance, we may consider the measurements performed by the "default" measurement angles (with respect to which all of the measurement adaptations are performed): if the measurement results $\mathsf{s}[v] = 0$ occur for each $v \in O^{\mathsf{c}}$ with non-zero probability, the adaptation in a procedure which performs a unitary serves only to select for the same transformation as would occur for that branch. In this case, a measurement-

based procedure for a unitary transformation may be regarded as *simulating postselection* of this "distinguished" sequence of results, which is feasible due to the special choice of initial state and measurements involved.[1]

If we could actually postselect the distinguished measurement results, then we would not need to perform any adaptation of the measurement procedure, and dispense with classical feed-forward. This would leave us with a set of data from which one could plausibly form an analytical intuition:

1. a set of qubits V, with an input subsystem $I \subseteq V$ supporting an arbitrary input state, with the others prepared initially in the $|+\rangle$ state;
2. a graph G on V, describing a set of entangling operations performed between pairs of qubits;
3. a *set* (without any ordering) of measurements to perform on the qubits of V; and
4. the subset of qubits $O \subseteq V$ on which no measurements are to be performed, in order to leave a residual quantum state as output.

Such data represents a procedure to solve a problem using a *postselection-based* (rather than measurement-based) model of quantum computing.

Supposing that one can form intuitions about quantum computation in these terms, and even derive tuples (G, I, O, M) consisting of a geometry (G, I, O) and a set of default measurements M encoding a solution to a computational problem, it would then remain to produce from this a description of a realizable algorithm. In this case, it is sufficient to specify an ordering of the measurements in the postselection-based "algorithm", as well as a specification of how those measurements are to be adapted and any final corrections to be applied in case the distinguished measurement results do not occur. If this is possible, the postselection-based procedure is meaningfully an algorithm, which may be extended in principle to a realizable procedure to perform the operation with bounded error.

In the case where every measurement is performed in the XY plane, the basis of each measurement is of the form $|\pm_\alpha\rangle \propto |0\rangle \pm \mathrm{e}^{i\alpha}|1\rangle$ for some angle α, as defined in (2.6). Suppose that we are given a geometry (G, I, O), and a vector $\mathbf{a} = (\alpha_v)_{v \in O^\mathsf{c}}$ of the default measurement angles for each $v \in O^\mathsf{c}$, which together define a postselection-based transformation $\Phi : \mathsf{D}(I) \longrightarrow \mathsf{D}(O)$ in the case where all of the measurements on the qubits v yield $\mathsf{s}[v] = 0$. Such a postselection-based procedure will also be a special case of some measurement-based procedure, with different computational branches for each sequence of measurement results: we refer to the branch which performs the desired postselection procedure, given by the measurement sequence $\big(\mathsf{s}[v]\big)_{v \in O^\mathsf{c}} \equiv 0$, as the *positive branch*.

Under these constraints, we may formalize the problem of mapping postselection-based procedures to measurement-based algorithms as follows:

[1] Aaronson [1] shows that allowing postselection even of single-qubit measurement results, together with a set of (approximately) universal unitary operations and preparation of the $|0\rangle$ state, leads to circuit models which can efficiently solve problems in PP: this is taken as a sign of the unreasonability of models allowing postselection. In contrast, the way in which measurements are simulated in the stabilizer formalism (and in the context of the flow/eflow/gflow techniques outlined in Section 3.6) allows the evolution of the state-space to be efficiently modelled due to the existence of a highly structured system of correlations of the states being measured, and the relationships between the measurement observables and those correlations.

Measurement Pattern Interpolation (MPI) Problem: Let $(G, I, O, \mathbf{a})$ be a partial specification of a measurement pattern $\mathfrak{P}$ in the OneWay($\mathbb{A}$) model for some $\mathbb{A} \subseteq \mathbb{R}$, where $\mathfrak{P}$ has underlying geometry (G, I, O) and default measurement angles $\mathbf{a} = (\alpha_v)_{v \in O^c} \in \mathbb{A}^{O^c}$. Given that $\mathfrak{P}$ performs a unitary transformation $U : \mathcal{H}_I \longrightarrow \mathcal{H}_O$ conditioned on its' positive branch, determine a measurement pattern $\overline{\mathfrak{P}}$ with underlying geometry (G, I, O) which performs U unconditionally, if such a pattern exists.

(This problem may readily be extended beyond measurements performed in the XY-plane, to which OneWay($\mathbb{A}$) is restricted; however, we will be primarily concerned with the formulation presented above.)

In this chapter, I will describe solutions for special cases of the Measurement Pattern Interpolation problem, which revolve around generalized flow conditions on the geometry (G, I, O), or on the special case of "Pauli measurements", *i.e.* when we have $|\pm_{\alpha_v}\rangle = |\pm\mathsf{x}\rangle$ or $|\pm_{\alpha_v}\rangle = |\pm\mathsf{y}\rangle$ for each $v \in O^c$.

In the absence of "algorithmic insights" provided by measurement-based computation — and setting aside theoretical motivations, such as obtaining a more robust understanding of the scope of those measurement-based procedures which perform unitary transformations — solving **MPI** is only of practical importance if there exists a source of "problem instances" $(G, I, O, \mathbf{a})$ for which we wish to obtain measurement patterns. In this chapter, I also present *quadratic form expansions*, which is a format in which unitaries may be expressed concisely. These are plausibly a "natural" source of instances of the Measurement Pattern Interpolation problem, in that these expressions (and similar ones) have arisen multiple times in the course of research in quantum information independently of the formulation of **MPI**.

Previous appearances of this work. Some of the work in this chapter are developments original to this thesis, including the substitution described in (4.26), and the solved case of the **MPI** for geometries derived from eflows described in Section 4.2.2. The rest of the work of this chapter is the result of a collaboration with Elham Kashefi, Vincent Danos, and Martin Roetteler, and first appeared in [46].

4.1 Quadratic Form Expansions

4.1.1 Notation and definitions

We first present a notational device which will prove most useful:

Notation. For an index set V, and for a vector $\mathbf{x} \in \{0,1\}^V$, and for a subset $S \subseteq V$, the vector $\mathbf{x}_S$ denotes the restriction of $\mathbf{x}$, considered as a function $\mathbf{x} : V \longrightarrow \{0,1\}$, to the set S. In particular, for an operator $|\mathbf{x}\rangle \in {\mathcal{H}_2}^{\otimes V}$ given by

$$|\mathbf{x}\rangle \;=\; \bigotimes_{v \in V} |x_v\rangle \,, \tag{4.1}$$

we define the corresponding standard-basis vector $|\mathbf{x}_S\rangle \in {\mathcal{H}_2}^{\otimes S}$ by

$$|\mathbf{x}_S\rangle \;=\; \bigotimes_{v\in S} |x_v\rangle \;. \tag{4.2}$$

Definition 4-1. Let V be a set of n indices, and $I, O \subseteq V$ be (possibly intersecting) subsets. Then a *quadratic form expansion* for an operator $M : {\mathcal{H}_2}^{\otimes I} \longrightarrow {\mathcal{H}_2}^{\otimes O}$ is an expression of the form

$$M \;=\; C \sum_{\mathbf{x}\in\{0,1\}^V} \mathrm{e}^{iQ(\mathbf{x})}\; |\mathbf{x}_O\rangle\langle\mathbf{x}_I| \;, \tag{4.3}$$

where Q is a quadratic form[2] in $\mathbb{R}^V$, and C is a complex scalar.

For an operator expressed in matrix form, a quadratic form expansion provides an expression for each matrix coefficient as a scalar multiple of a sum of complex numbers with modulus 1 (or *complex units*), where the scalar factor and the arguments of these complex units are given by a uniform formula. As such, it is essentially an interpretation of an operator as an abstract "sum over paths", where each path has the same amplitude, but a different phase: each such "path" is given by a particular string $\mathbf{x}$, and the coefficient for a transition $\mathbf{a} \longmapsto \mathbf{b}$ is given by summing over all paths satisfying the constraints $\mathbf{x}_I = \mathbf{a}$ and $\mathbf{x}_O = \mathbf{b}$.

4.1.2 Related work in the literature

Similar expressions to those in (4.3) have occurred several times previously in quantum information or quantum mechanics literature (albeit without the terminology or notation presented here). We will provide a brief overview of some examples.

Stabilizer codes and Clifford group operations

Perhaps the earliest presentation of quadratic form expansions is by Schlingemann and Werner [117], where they are used to describe a special class of stabilizer codes (but using significantly different notation, and where the expansions are extended to $\mathbf{x} \in \mathbb{Z}_d^V$ for $d \geqslant 2$ rather than only for $\mathbf{x} \in \{0,1\}^V$). Their analysis corresponds in the boolean case to quadratic form expansions where Q is obtained from the adjacency matrix of a graph, and where I, O form a partition of the index set V. This treatment was extended to describe generalized stabilizer codes and Clifford group operations over $\mathsf{U}(d)$ in Schlingemann [116]. The link between the work of Schlingemann and Werner (as well as their notations), and quadratic form expansions as they are defined and treated in this thesis, is perhaps best hilighted by the result of [71], which proved that all stabilizer codes are locally equivalent to some graphical code.

Dehaene and de Moor [48] also present an analysis of Clifford group operations over $\mathcal{H}_2$, in which they present an operator formula for a generic Clifford group operation which may be obtained from a description of how it transforms the Pauli group by conjugation. We will describe this matrix formula, and specify how it may be computed, in Section 4.2.3.

[2] A *quadratic form* is a polynomial where every term has degree 2.

The quantum Fourier transform

One concrete example of a unitary matrix which is naturally expressed as a quadratic form expansion is the quantum Fourier transform over the cyclic group $\mathbb{Z}_{2^k}$, which we may write

$$\begin{aligned} \mathcal{F}_{2^k} &= \frac{1}{\sqrt{2^k}} \sum_{x,y \in \mathbb{Z}_{2^k}} \mathrm{e}^{2\pi i x y / 2^k} \, |y\rangle\langle x| \\ &\cong \frac{1}{\sqrt{2^k}} \sum_{\substack{\mathbf{x} \in \{0,1\}^k \\ \mathbf{y} \in \{0,1\}^k}} \exp\left(\frac{i\pi}{2^{k-1}} \left[\sum_{j=0}^{k-1} 2^h x_j \right] \left[\sum_{h=0}^{k-1} 2^j y_h \right] \right) |\mathbf{y}\rangle\langle\mathbf{x}| \,, \end{aligned} \tag{4.4}$$

identifying the integers $x, y \in \mathbb{Z}_{2^k}$ with their usual representations as binary strings; this is a quadratic form expansion with V partitioned into disjoint index sets associated with the bits of $\mathbf{x}$ and $\mathbf{y}$, and a quadratic form $Q : \mathbb{R}^{2k} \longrightarrow \mathbb{R}$ given by

$$Q(\mathbf{x}, \mathbf{y}) = \sum_{h=0}^{k-1} \sum_{j=0}^{k-h-1} \frac{2^{(h+j)}}{2^{k-1}} \, \pi \, x_j \, y_h \,. \tag{4.5}$$

On page 179, we will show how this structure can be exploited to reproduce the linear-nearest neighbor implementation of $\mathcal{F}_{2^k}$ described by Fowler, Devitt, and Hollenberg in [66] by essentially automatic techniques.

Remark. The operation $\mathcal{F}_{2^k}$ is an instance of a generalized Clifford group operation over $\mathsf{SU}(2^k)$; however, the description of $\mathcal{F}_{2^k}$ above differs from the treatment of *e.g.* [116] in that we expand it here in terms of operations on qubits, rather than leaving it as an atomic operation on $\mathcal{H}_{2^k}$.

Sums over paths

A similar representation of operators as a sum over paths as in (4.3) was used by Dawson *et al.* in [41] to provide a succinct proof of $\mathsf{BQP} \subseteq \mathsf{PP}$, using cubic polynomials over $\{0, 1\}$ rather than quadratic polynomials; however, comments made in Section VI of their work anticipate quadratic form expansions with discretized coefficients, and imply a method (which we will describe in Section 4.1.3) of obtaining these expansions from unitary circuits using the gate-set $\textsc{Phases}(\frac{\pi}{4}\mathbb{Z})$ defined on page 27.

The expressions considered in [41] are explicitly described as a sum over computational paths which arise in the evolution of a quantum state through a unitary circuit: the evolution of the system in each path is described by a time-ordered sequence of standard basis states given by boolean strings, where these states differ in at most one bit position for consecutive time-steps. The "paths" in this case are walks in the boolean cube $\{0, 1\}^k$ (for a circuit acting on $k \leqslant n$ qubits) between boolean strings which differ in zero or one bit position (*i.e.* including self-loops). The bit whose value may change at each step is the same across all paths; these walks can then be described by the initial position, followed by one bit for each edge traversed, which may be described by a boolean function $\{0, 1\}^V$

describing the history of the computation in that path. Summing over all computational paths gives rise to the unitary transition matrix.

The analogy of such matrix formulae to a sum-over-paths, suggested both in [41] and in this thesis, is prompted in part by the formal similarity between *e.g.* quadratic form expansions and *path integrals*. Introduced by Feynman [61], the path integral formalism is an alternative approach to describing the evolution of quantum states by associating amplitudes to trajectories of the system (*e.g.* through space-time or phase space), and describing the probabilities of events upon measurement using the cumulated amplitude over all paths starting from the initial state and ending at a given configuration. This "summation over paths" can be arrived at by taking integrals over all possible configurations at intermediate times (taking the limit as the intervals between these intermediate times tend to zero), which amounts to integrating over a collection of "virtual events" for these intermediate times. This leads to a formulation where the transition amplitude between the initial and final configurations depends on some quantity with the units of energy (usually the *action* of the system) depending on physical parameters associated with these virtual events, which can then be attributed to trajectories in space. Further details can be found in [61, 64, 119]. In a quadratic form expansion, the indices of the vector $\mathbf{x}$ which are neither in I or O may be considered to correspond to these virtual events, and the quadratic form associated with the "computational paths" of the computation seems to correspond to the action (which in most applications of path integration determines the phase of each path). This suggests that a quadratic form expansion, or similar matrix expression, may be a means of discretizing a path integral in a way which may be amenable to study for quantum algorithms.

There are some technical discrepancies on a formal level, most notably that the phase of a system with multi-body interactions generally is not a quadratic polynomial in the dynamical variables of the system; or we may wish to evaluate path integrals where different paths have amplitudes of different weights (*e.g.* with lower amplitudes away from the stationary points of the action, due to cancellation of contributions of similar paths) in an attempt to obtain an approximation which is easier to estimate. We will show in in Section 4.3.1 how matrix expressions similar to quadratic form expansions, with *unequally weighted* paths or where the action is of degree higher than 2, may be transformed into quadratic form expansions. The simplicity of these transformations suggest that quadratic form expansions are not unduly restrictive from the point of view of representing the discretization of a path integral. Thus, an examination of when we can synthesize quantum algorithms from quadratic form expansions would plausibly lead to automatic techniques for efficiently estimating path integrals on a quantum computer.

Remarks on quadratic form expansions as a representation of operators

Given the possibility of quadratic form expansions arising generically from path-integral descriptions of physical systems, as well as the fact that these and similar matrix formulae have repeatedly arisen in the literature, I conjecture that quadratic form expansions can be expected to arise naturally as mathematical expressions of interest for operators in quantum information theory. Based on this, I will suppose for the purposes of this thesis that a quadratic form expansion is a reasonable format in which a unitary operator U may be specified, in the context of the problem of obtaining a succinct decomposition of a

CPTP map performing U in some standard model of quantum computing (*e.g.* a unitary circuit model or measurement based model).

Setting aside the special case of the quantum Fourier transform $\mathcal{F}_{2^k}$, we will see in Section 4.2.3 how such an expression of a unitary U may be obtained (without first having a unitary circuit or measurement-based algorithm for U) in the case of Clifford group operations. While this represents a limited class of operations, it nevertheless represents a generic family of operators U for which quadratic form expansions may be readily obtained.

4.1.3 Constructing quadratic form expansions

As noted above, we will be primarily interested in the problem of transforming a quadratic form expansion into a quantum circuit or measurement-based procedure. However, to illustrate the generality of quadratic form expansions, and how they relate to unitary circuit models and the measurement based model of quantum computing, we will show how to perform the reverse transformation, from unitary circuits and measurement-based procedures to quadratic form expansions.

Construction as a sum over computational paths in circuits

As we remarked in Section 4.1.2, Dawson *et al.* describe a method for obtaining quadratic form expansions from unitary circuits in [41, Section VI]; the particular set of unitary transformations they use is the set $\{H, R_{\mathsf{z}}(\pi/4), \wedge X\}$. The construction is closely related to the stable index circuit notation presented in Chapter 1 (on page 31: we will sketch this construction using the somewhat more general parameterized gate set $\left\{H, Z^t, \wedge Z^t\right\}$, where

$$Z^t = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{i\pi t} \end{bmatrix} \propto R_{\mathsf{z}}(\pi t), \qquad \wedge Z^t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{i\pi t} \end{bmatrix}, \tag{4.6}$$

and where t is allowed to range over some subset of $(-1, 1]$.

Consider a quantum circuit C implementing U exactly, using the operations H, $\wedge Z^t$, and Z^t. As in a stable index representation of C, we assign indices (or *path labels*, to use the terminology of [41]) to the segments of each wire in the circuit, with each segment bounded by an input terminal, and output terminal or a Hadamard operation. In particular:

1. We enumerate the "input" segments of the circuit from 1 to k, and for each wire $1 \leqslant j \leqslant k$ introduce a path label x_j corresponding to an input bit $x_j \in \{0, 1\}$; and we set $I = \{1, \ldots, k\}$.

2. Reading the circuit from the inputs to the outputs, we introduce new path variables x_j for $j > k$ for each wire segment "introduced" by the presence of a Hadamard operation, corresponding to the "random" bit which is introduced by applying the Hadamard to the "previous value" of the bit.

We may then describe computational paths described by the circuit by a walk in the boolean cube $\{0,1\}^k$, with one bit possibly being flipped each time a Hadamard gate is performed.

If the total number of path labels introduced is n, a single computational path can therefore be described by setting all of the the path variables $x_1 \cdots x_n$ collectively to some particular binary string in $\{0,1\}^n$. For a path which is given by a string $\mathbf{x} \in \{0,1\}^n$, the phase of that path (which governs how it interferes with other paths in the unitary evolution described by C) is given by a function $\varphi(\mathbf{x})$ which may be iteratively constructed from the gates of C as follows:

(*i*) For every Hadamard gate between two wire segments labelled with path variables x_h and x_j, we include a term $x_h x_j$.

(*ii*) For every $\wedge Z^t$ operation between two wires, with a path variable x_h labelling the segment of one wire and x_j labelling the segment of the other wire in which the $\wedge Z^t$ operation is performed, we add a term $t\, x_h x_j$.

(*iii*) For every Z^t operation on a wire segment labelled with a path variable x_j , we add a term $t\, x_j^2$. (As the path variable x_j ranges over $\{0,1\}$, the extra power of 2 has no effect.)

We may collect $\wedge Z^t$ and Z^t terms so that there is at most one $\wedge Z^t$ gate between segments labelled with x_h and x_j (for $h \neq j$) for some value of t, and at most one Z^t gate on a segment labelled with x_h, for any $h, j \in \{1, \ldots, n\}$. As there is at most one Hadamard operation between two wire segments by construction of the circuit, this implies that the construction of φ will include at most one $x_h x_j$ term for each $\{h, j\} \subseteq \{1, \ldots, n\}$. The phase of a given path, described by a bit-string $\mathbf{x} \in \{0,1\}^n$, is then given by $(-1)^{\varphi(\mathbf{x})} = \mathrm{e}^{i\pi\varphi(\mathbf{x})}$. Each path also has an associated amplitude of $2^{-r/2}$, where $r = n - k$ is the number of Hadamard gates in the circuit.[3]

Let O be the set of indices j such that some wire is labelled by the path-variable x_j at its' output end. Then, the initial points of computational paths are described by bit-vectors $\mathbf{a} \in \{0,1\}^I$, and the terminal points of paths are described by $\mathbf{b} \in \{0,1\}^O$. The coefficients ${U_{\mathbf{a}}}^{\mathbf{b}}$ can then be given as the sum of the contributions of all paths beginning at $\mathbf{x}_I = \mathbf{a}$ and ending at $\mathbf{x}_O = \mathbf{b}$:

$$ {U_{\mathbf{a}}}^{\mathbf{b}} \;=\; \langle \mathbf{b} |\, U \,| \mathbf{a} \rangle \;=\; \frac{1}{\sqrt{2^r}} \sum_{\substack{\mathbf{x} \in \{0,1\}^n \\ \mathbf{x}_I = \mathbf{a} \\ \mathbf{x}_O = \mathbf{b}}} \mathrm{e}^{i\pi\varphi(\mathbf{x})} \,. \tag{4.7} $$

As $\varphi(\mathbf{x})$ is a quadratic form over $\mathbf{x}$ by construction, this is an expression of the coefficients of U as a quadratic form expansion.

Any stable index tensor expression for U (using any set of elementary gates) can be interpreted as a sum-over-paths formulation of U, identifying paths with a setting of all

[3] Although it is quite reasonable to consider φ to be simply a polynomial over $\mathbb{R}$, in terms of the descriptions used in Section VI of [41], one may consider φ to be a polynomial over the ring $\mathbb{R}/2\mathbb{Z}$. If we restrict to $t \in \frac{\pi}{4}\mathbb{Z}$, we may simplify this to the finite ring $\mathbb{Z}_8$ by multiplying all of the coefficients by 4, and using it to describe powers of $\sqrt{i}$ rather than of -1 (but still labelling each path by a vector $\mathbf{x} \in \{0,1\}^n \subseteq \mathbb{Z}_8^n$).

of the indices in the expression. However, this will not in general yield a *quadratic form expansion*, as the modulus of the amplitude of each path need not be uniform, as in a quadratic form expansion: we then cannot express U a uniformly normalized sum. As well, gates with generic three-body or higher-degree interactions will give rise to cubic terms in the phase function φ. Thus, as a construction for quadratic form expansions, the construction outlined above cannot be extended to arbitrary gate sets (nor even arbitrary "parsimonious" gate sets, in the sense of Definition 1-22), but apparently only to gate sets closely related to PHASES($\mathbb{A}$).

Inductive construction from circuit decompositions

An equivalent, and perhaps more illustrative, construction to the above can be made by supposing that one has a unitary circuit C decomposed into unitaries, each of which has a known quadratic form expansion. In such a case, we can compose the quadratic form expansions by matrix multiplication.

Suppose we are given a stable index tensor representation of a unitary U in terms of any set of unitary transformations, provided that every unitary transformation W in the decomposition has a known quadratic form expansion,

$$W \;=\; C_W \sum_{\mathbf{x}\in\{0,1\}^{V_W}} \mathrm{e}^{iQ_W(\mathbf{x})}\, |\mathbf{x}_{O_W}\rangle\langle\mathbf{x}_{I_W}|\,, \tag{4.8}$$

for some constant $C_W \in \mathbb{C}$, quadratic form Q_W, and sets V_W, I_W, O_W depending on W. Relabelling the elements of the particular sets V_W, I_W, and O_W in this expansion does not substantially change the operator, except in changing the labels of the space on which it operates. Consider an instance $W^{(j)}$ of the unitary W in the stable index tensor expression for U, with input indices $I^{(j)}$ and output indices $O^{(j)}$: we may then make a copy of the above expansion by bijectively mapping $I_W \longrightarrow I^{(j)}$ and $O_W \longmapsto O^{(j)}$, and defining an extended index set $V_W \longmapsto V^{(j)}$ of indices such that, for any other set $V^{(h)}$ corresponding to any other unitary in the circuit, we have $v \in V^{(j)} \cap V^{(h)}$ only if $v \in I^{(j)}$ or $v \in O^{(j)}$ (in which case we also have $v \in O^{(h)}$ or $v \in I^{(h)}$ respectively).

Having obtained a quadratic form expansion for *each* unitary in the decomposition of U, we may then apply the following Lemma, whose proof is trivial:

Lemma 4-1. *Let W_1, W_2 be operators given by quadratic form expansions of the form*

$$W_j \;=\; C_j \sum_{\mathbf{x}\in\{0,1\}^{V_j}} \mathrm{e}^{iQ_j(\mathbf{x})}\, \big|\mathbf{x}_{O_j}\big\rangle\big\langle\mathbf{x}_{I_j}\big|\;. \tag{4.9}$$

In the following, $C = C_1C_2$, and sums are over $\{0,1\}^{V_1 \cup V_2}$.

(i) If $V_1 \cap V_2 = I_2 = O_1$, then $U_2U_1 \;=\; C \sum_{\mathbf{x}} \mathrm{e}^{\,iQ_1(\mathbf{x}) + iQ_2(\mathbf{x})}\, |\mathbf{x}_{O_2}\rangle\langle\mathbf{x}_{I_1}|\,.$

(ii) If V_1 and V_2 are disjoint, then $U_1 \otimes U_2 \;=\; C \sum_{\mathbf{x}} \mathrm{e}^{\,iQ_1(\mathbf{x}) + iQ_2(\mathbf{x})}\, |\mathbf{x}_O\rangle\langle\mathbf{x}_I|\,,$ where $I = I_1 \cup I_2$ and $O = O_1 \cup O_2$.

By construction, the quadratic form expansions derived for each of the unitaries in the decomposition of U satisfy the above properties, and so we may compose them to obtain a quadratic form expansion for the whole.

For the gates H, Z^t, and $\wedge Z^t$ in particular, described in the sum-over-paths construction, we have the following expansions:

$$H = \frac{1}{\sqrt{2}} \sum_{\mathbf{x}\in\{0,1\}^2} \mathrm{e}^{i\pi x_1 x_2} |x_2\rangle\langle x_1| , \qquad V=\{1,2\}, \quad I=\{1\}, \qquad O=\{2\}; \tag{4.10a}$$

$$Z^t = \sum_{\mathbf{x}\in\{0,1\}^1} \mathrm{e}^{i\pi t x_1^2} |x_1\rangle\langle x_1| , \qquad V=\{1\}, \qquad I=\{1\}, \qquad O=\{1\}; \tag{4.10b}$$

$$\wedge Z^t = \sum_{\mathbf{x}\in\{0,1\}^2} \mathrm{e}^{i\pi t x_1 x_2} |x_1 x_2\rangle\langle x_1 x_2| , \qquad V=\{1,2\}, \quad I=\{1,2\}, \quad O=\{1,2\}. \tag{4.10c}$$

None of these unitaries involve "internal" indices in the sense described above, and so a stable index tensor expression for a unitary U in terms of these unitaries may be easily transformed into a quadratic form expansion for U. (The quadratic form expansions constructed with this set of unitaries will be equivalent to those obtained from the sum-over-paths construction of the previous section.)

Remark. We may also decompose unitaries U as a product of *non-unitary* operators with known phase-map decompositions as well. For instance, we may define the matrix

$$L = \mathrm{e}^{-i\pi/6} \begin{bmatrix} 1 & \mathrm{e}^{2\pi i/3} \\ 1 & -1 \end{bmatrix} = \mathrm{e}^{-i\pi/6} \sum_{x_1,x_2\in\{0,1\}} \mathrm{e}^{i\pi(2x_1+x_1x_2)/3} |x_2\rangle\langle x_1| ; \tag{4.11}$$

it is easy to verify that L is self-inverse, in which case the above expansion gives rise to another quadratic form expansion decomposition for $\mathbb{1}_2$:

$$\mathbb{1}_2 = \mathrm{e}^{-i\pi/3} \sum_{\mathbf{x}\in\{0,1\}^3} \mathrm{e}^{i\pi\left[(2x_1^2+x_1x_2)+(2x_2^2+x_2x_3)\right]/3} |x_3\rangle\langle x_1| . \tag{4.12}$$

Construction from one-way measurement patterns

The qubit-reversal pattern and the many-qubit measurement pattern $\mathfrak{Z}\mathfrak{Z}\cdots\mathfrak{z}^\theta$, which we have considered in Sections 2.3.1 and 2.3.2, do not have any decomposition into smaller elements due to (currently known) generic constructions from unitary circuits into the one-way model. This motivates considering quadratic form expansions in terms of measurement-based quantum computing, independently of unitary circuit models.

Suppose we have a measurement-based procedure with underlying geometry (G, I, O) which performs a unitary transformation $U : \mathcal{H}_I \longrightarrow \mathcal{H}_O$, and that there is a sequence of measurement results which occurs with non-zero probability for which there are no final corrections to be performed on the qubits in O. Let $(\alpha_v)_{v\in O^c}$ be the choice of measurement angles for this branch of the computation: we then designate this branch of

the computation as the positive branch. Then we may express U (modulo a scalar factor due to the probability of postselecting the positive branch) in the form

$$U \;\propto\; \left[\bigotimes_{v\in O^{\mathsf{c}}} \langle +_{\alpha_v}|_v\right] \left[\prod_{uv\in E(G)} \wedge Z_{u,v}\right] \left[\bigotimes_{v\in I^{\mathsf{c}}} |+\rangle_v\right], \tag{4.13}$$

which represents the operations of preparing the qubits of I^{c}, performing the entangling operations, and postselecting the states $|+_{\alpha_v}\rangle$ for $v \in O^{\mathsf{c}}$ which occur in the positive branch. Note that we can factor the postselection operators $\langle +_\alpha|$ into a Z-rotation and postselection of the state $|+\rangle$,

$$\langle +_{\alpha_v}| \;=\; \langle +|\, R_z(-\alpha_v); \tag{4.14}$$

factoring these rotations into the middle, we may collect them together with the $\wedge Z$ operations corresponding to the edges of the entanglement graph G into a single diagonal unitary operation arising from the exponential of a diagonal Hermitian matrix,

$$\left[\bigotimes_{v\in O^{\mathsf{c}}} R_{\mathsf{z}}(-\alpha_v)\right] \left[\prod_{uv\in E(G)} \wedge Z_{u,v}\right] \;=\; \mathrm{e}^{iJ}, \qquad \text{where} \tag{4.15a}$$

$$J \;=\; \sum_{\substack{\{u,v\}\subseteq V(G)\\ u\neq v}} \pi\Big[\,|1\rangle\langle 1|_u \otimes |1\rangle\langle 1|_v\,\Big] \;-\; \sum_{v\in O^{\mathsf{c}}} \alpha_v\, |1\rangle\langle 1|_v \;. \tag{4.15b}$$

Define a quadratic form $Q : \mathbb{R}^{V(G)} \longrightarrow \mathbb{R}$ by

$$Q(\mathbf{x}) \;=\; \sum_{\substack{\{u,v\}\subseteq V(G)\\ u\neq v}} \pi x_u x_v \;-\; \sum_{v\in O^{\mathsf{c}}} \alpha_v x_v^2\,, \tag{4.16}$$

and note that, for any $S \subseteq V(G)$, we can expand

$$\begin{aligned}\bigotimes_{v\in S^{\mathsf{c}}} |+\rangle_v \;\propto\; \sum_{\mathbf{x}'\in\{0,1\}^{S^{\mathsf{c}}}} \mathbb{1}_2^{\otimes S} \otimes |\mathbf{x}'\rangle \;&=\; \sum_{\mathbf{x}\in\{0,1\}^{V(G)}} \Big[\,|\mathbf{x}_S\rangle\langle \mathbf{x}_S|\,\Big] \otimes |\mathbf{x}_{S^{\mathsf{c}}}\rangle \\ &=\; \sum_{\mathbf{x}\in\{0,1\}^{V(G)}} |\mathbf{x}\rangle\langle \mathbf{x}_S|\;. \end{aligned} \tag{4.17}$$

Then, we may obtain

$$\begin{aligned} U \;&\propto\; \left[\bigotimes_{v\in O^{\mathsf{c}}} \langle +_{\alpha_v}|_v\right] \mathrm{e}^{iJ} \left[\bigotimes_{v\in I^{\mathsf{c}}} |+\rangle_v\right] \\ &\propto\; \left[\sum_{\mathbf{y}\in\{0,1\}^{V(G)}} |\mathbf{y}_O\rangle\langle \mathbf{y}|\right] \mathrm{e}^{iJ} \left[\sum_{\mathbf{x}\in\{0,1\}^{V(G)}} |\mathbf{x}\rangle\langle \mathbf{x}_I|\right] \\ &=\; \sum_{\mathbf{x},\mathbf{y}\in\{0,1\}^{V(G)}} |\mathbf{y}_O\rangle \Big[\mathrm{e}^{i\langle \mathbf{y}|J|\mathbf{x}\rangle}\Big] \langle \mathbf{x}_I| \;=\; \sum_{\mathbf{x}\in\{0,1\}^{V(G)}} \mathrm{e}^{iQ(\mathbf{x})}\; |\mathbf{x}_O\rangle\langle \mathbf{x}_I|\,, \end{aligned} \tag{4.18}$$

which is a quadratic form expansion for U.

4.1.4 Interpretation as a description of a postselection procedure

In the context of **MPI** (page 159), the construction above of quadratic form expansions from measurement-based computations hints at an approach for doing the reverse: to turn a quadratic form expansion into a measurement-based procedure, and more precisely into an instance of the **MPI**. Consider a unitary U given by a quadratic form expansion as in (4.3), where the quadratic form Q is given by

$$Q(\mathbf{x}) \;=\; \sum_{\{u,v\}\subseteq V} \theta_{uv} x_u x_v \tag{4.19}$$

for any $\mathbf{x} \in \{0,1\}^V$, for some real parameters $\{\theta_{uv}\}_{u,v\in V}$; and where the sum includes singleton sets corresponding to the case $u = v$. We may express $Q(\mathbf{x})$ as an inner product $\langle \mathbf{x}|J|\mathbf{x}\rangle$, where J is a 2-local diagonal operator similar in construction to that of (4.15b):

$$J \;=\; J_O + J_1 + J_2 \,, \qquad \text{where} \tag{4.20a}$$

$$J_O \;=\; \sum_{w\in O} \theta_{ww} \,|1\rangle\langle 1|_w \;, \qquad J_1 \;=\; \sum_{v\in O^{\mathsf{c}}} \theta_{vv} \,|1\rangle\langle 1|_v \;, \tag{4.20b}$$

$$J_2 \;=\; \sum_{\substack{\{u,v\}\subseteq V\\ u\neq v}} \theta_{uv} \Big[\, |1\rangle\langle 1|_u \otimes |1\rangle\langle 1|_v \,\Big]. \tag{4.20c}$$

Then we may decompose U as

$$U \;\propto\; \sum_{\mathbf{x}\in\{0,1\}^V} |\mathbf{x}_O\rangle\langle \mathbf{x}|\, \mathrm{e}^{iJ}\, |\mathbf{x}\rangle\langle \mathbf{x}_I| \;\propto\; \Big[\bigotimes_{w\in O^{\mathsf{c}}} \langle +|\Big]\, \mathrm{e}^{iJ_O}\mathrm{e}^{iJ_1}\mathrm{e}^{iJ_2} \Big[\bigotimes_{v\in I^{\mathsf{c}}} |+\rangle\Big] \;. \tag{4.21}$$

Note that e^{iJ_1} consists simply of single-qubit Z rotations applied to the elements of O^{c},

$$\mathrm{e}^{iJ_1} \;=\; \prod_{v\in O^{\mathsf{c}}} \mathrm{e}^{i\theta_{vv}|1\rangle\langle 1|_v} \;\propto\; \bigotimes_{v\in O^{\mathsf{c}}} R_{\mathsf{z}}(\theta_{vv}) \;, \tag{4.22}$$

and similarly for e^{iJ_O}; and we may express e^{iJ_2} by a product of (potentially fractional) powers of controlled-Z on distinct pairs of qubits:

$$\mathrm{e}^{iJ_2} \;=\; \prod_{\substack{\{u,v\}\subseteq V\\ u\neq v}} \mathrm{e}^{i\theta_{uv}|1\rangle\langle 1|_u\otimes|1\rangle\langle 1|_v} \;=\; \prod_{\substack{\{u,v\}\subseteq V\\ u\neq v}} \wedge Z_{u,v}^{\,\theta_{uv}/\pi} \;. \tag{4.23}$$

Let $\alpha_v = -\theta_{vv}$ for each $v \in O^{\mathsf{c}}$: then, commuting e^{iJ_O} leftwards and (similarly) absorbing e^{iJ_1} into the projection operations, we obtain

$$U \;\propto\; \mathrm{e}^{iJ_O} \Big[\bigotimes_{w\in O^{\mathsf{c}}} \langle +_{\alpha_v}|\Big] \Big[\prod_{\substack{\{u,v\}\subseteq V\\ u\neq v}} \wedge Z_{u,v}^{\,\theta_{uv}/\pi}\Big] \Big[\bigotimes_{v\in I^{\mathsf{c}}} |+\rangle\Big] \;, \tag{4.24}$$

where we have applied the relation $|+_\alpha\rangle \propto R_{\mathsf{z}}(\alpha)\,|+\rangle$. This is an expression of U as the preparation of some number of $|+\rangle$ states, followed by a diagonal unitary operator consisting of two-qubit (fractional) controlled-Z operations, followed by post-selection of the states $|+_{\alpha_v}\rangle$ for $v \in O^{\mathsf{c}}$ (which succeeds with non-zero probability, provided the initial scalar factor of the quadratic form expansion was nonzero), and finally applying some single-qubit Z rotations on the remaining qubits.

If $\theta_{uv} \in \{0, \pi\}$ for all distinct $u, v \in V$, and $\theta_{vv} = 0$ for $v \in O$, the above describes precisely the action of the positive branch of a measurement-based computation in which the default measurement angles on the qubits $v \in O^{\mathsf{c}}$ are given by $-\theta_{vv}$, and where we condition on all measurements yielding the result $\mathsf{s}[v] = 0$. Thus, such a quadratic form expansion encodes an instance of **MPI**.

Furthermore, we may transform any quadratic form expansion into such a form. For quadratic form expansions with $\theta_{vv} \neq 0$ for $v \in O$, we may obtain a modified geometry by composing to such output qubits a quadratic form expansion for the identity operation,

$$\mathbb{1}_2 \;=\; \sum_{x_v, x_{v''}} \delta_{x_v, x_{v''}} \,|x_{v''}\rangle\langle x_v| \;=\; \frac{1}{2} \sum_{x_v, x_{v'}, x_{v''}} (-1)^{x_v x_{v'} + x_{v'} x_{v''}} \,|x_{v''}\rangle\langle x_v| \tag{4.25}$$

which corresponds to the pattern $\mathfrak{Id}^2$ of (2.37); this yields a quadratic form expansion with output indices O' such that $v \notin O'$, but where $v'' \in O'$ and $\theta_{v''v''} = 0$. To eliminate cross-term coefficients which are not integer multiples of π, we may use the equality

$$\mathrm{e}^{i\phi(x_u \oplus x_v)} \;=\; \frac{1}{2} \sum_{\mathbf{a} \in \{0,1\}^2} \mathrm{e}^{i\left[\pi(x_u a_1 + x_v a_1 + a_1 a_2) + \phi a_2^2\right]} \tag{4.26}$$

where $x_u \oplus x_v = x_u + x_v - 2x_u x_v$ is the sum of x_u and x_v modulo 2; we may eliminate factors of $\mathrm{e}^{i\theta_{uv} x_u x_v}$ such that $\theta_{uv} \notin \pi\mathbb{Z}$ for distinct u, v by substituting the equality above. Using this substitution amounts to decomposing the operator $\wedge Z^t$ using the construction for $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ illustrated in Figure 2-8, as Figure 4-1 illustrates.

In this way, we may translate any quadratic form expansion into a postselection procedure which uses only the entangling and measurement operations available to the OneWay($\mathbb{A}$) model, where $\mathbb{A}$ is the set of angles which occur in the quadratic form. In particular, for a quadratic form expansion as above where $\theta_{uv} \in \{0, \pi\}$ for distinct u, v and $\theta_{vv} = 0$ for $v \in O$, we may obtain a tuple $(G, I, O, \mathbf{a})$ describing an instance of **MPI** by defining G to be the graph whose vertices are the indices of V and whose edges are those pairs uv such that $\theta_{uv} \neq 0$ and by defining $\mathbf{a} = (-\theta_{vv})_{v \in O^{\mathsf{c}}}$. Thus, any such tuple for which the **MPI** has an efficient solution corresponds to a quadratic form expansion which we may translate to a measurement-based procedure performing a unitary transformation.

4.2 Solved cases of Measurement Pattern Interpolation

At present, there is no general solution to **MPI**, which seems to be a difficult problem. Current approaches towards a solution involve analyzing those measurements which can be treated essentially by the stabilizer formalism, where (as in Sections 3.6.1 and 3.6.5) we may relax the requirement that the measurement observables be Pauli operators. These lead to the following solved cases:

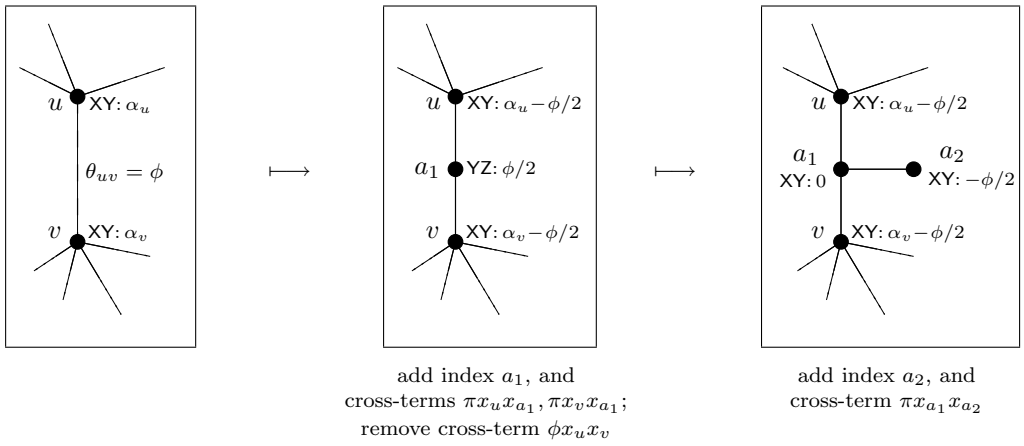


FIGURE 4-1. A transformation of quadratic form expansions to eliminate cross-term angles $\theta_{uv} \notin \{0, \pi\}$ which would correspond to fractional entangling operations, illustrated here in terms of geometries and measurement operations of a postselection procedure. This transformation essentially consists of applying the decomposition of $\wedge Z$ into $\mathfrak{Zz}$ and single-qubit rotations described in (2.42), extended to make use of the description of $\mathfrak{Zz} \cdots \mathfrak{z}^{\phi/2}$ in terms of XY-plane measurements for more general ϕ, which is illustrated in Figure 2-8. The coefficients of the quadratic form expansion in the left and right panels are given by the corresponding measurement angles in the XY plane; the center panel does not directly correspond to any quadratic form expansion, as it involves a measurement outside of the XY plane.

1. Tuples $(G, I, O, \mathbf{a})$ such that, for the constant function $\lambda : O^{\mathsf{c}} \longrightarrow \{\mathsf{XY}\}$, the tuple (G, I, O, λ) has a gflow (Definition 3-19);
2. Tuples $(G, I, O, \mathbf{a})$ such that, by applying certain simple graph transformations on G to obtain a smaller graph $\tilde{G}$ and a vertex set $T \subseteq V(\tilde{G})$, we obtain a tuple $(\tilde{G}, I, O, T)$ having an eflow (Definition 3-17);
3. Tuples $(G, I, O, \mathbf{a})$ such that $\alpha_v \in \frac{\pi}{2}\mathbb{Z}$ for every angle $\mathbf{a} = (\alpha_v)_{v \in O^{\mathsf{c}}}$ (*i.e.* where each measurement is performed with respect to either the $|{\pm}\mathsf{x}\rangle$ basis or $|{\pm}\mathsf{y}\rangle$ basis).

In this section, we describe these cases and illustrate how **MPI** may be solved for each.

As we described in the preceding section, any quadratic form expansion can be transformed into an instance of **MPI** by first transforming it into another quadratic form expansion whose cross-terms θ_{uv} are integer multiples of π. However, it will in some cases be convenient to synthesize implementations for unitaries directly from a quadratic form expansion having cross-terms which are fractional multiple of π, corresponding to postselecting in an extended measurement-based model which may use fractional powers of $\wedge Z$. This motivates the corresponding extension of the notion of a "geometry", which we define here for quadratic form expansions:

Definition 4-2. For a quadratic form expansion

$$C \sum_{\mathbf{x} \in \{0,1\}^V} \mathrm{e}^{iQ(\mathbf{x})} \, |\mathbf{x}_O\rangle\langle\mathbf{x}_I| \qquad \text{where} \quad Q(\mathbf{x}) = \sum_{\{u,v\} \subseteq V} \theta_{uv} x_u x_v \ , \tag{4.27}$$

the *geometry induced by the quadratic form* is a triple (G, I, O), where G is a *weighted* graph with vertex-set V, edge-set $\{uv \mid u \neq v \text{ and } \theta_{uv} \neq 0\}$, and edge-weights $W_G(uv) = \frac{\theta_{uv}}{\pi}$. Without loss of generality, we may require $W_G(uv) \in (-1, 1]$ for all uv; we may then refer to edges uv being of *unit weight* if $W_G(uv) = 1$, and *fractional weight* if $\theta_{uv} \notin \{0, 1\}$.

Geometries with only edges of unit weight correspond to geometries as described in Definition 2-5 in Section 2.2.3; we will refer to such geometries as *unit weight geometries*. Geometries having one or more fractional edge-weights we refer to as *fractional weight geometries*. We will note in a given context whether or not we are including geometries with fractional weight edges in the context of any given instance of **MPI**; however, unless otherwise noted, a reference to a "geometry" should be understood as referring to the unit weight geometries which arise in the one-way measurement model.

4.2.1 Geometries with gflows

The approach taken for **MPI** based on gflows is to ignore all angle information, and consider those tuples $(G, I, O, \mathbf{a})$ which represent postselection-schemes which may be interpolated regardless of the value of $\mathbf{a}$. By doing so, the objective is to consider post-selection schemes which *generically* perform unitary transformations, in the following sense:

Generic Measurement Pattern Interpolation (GMPI) Problem: Given an input geometry (G, I, O), determine if there exists a family of measurement patterns $\mathfrak{P}(\mathbf{a})$ in the model OneWay($\mathbb{R}$) having the underlying geometry (G, I, O) and default measurement angles $\mathbf{a} \in \mathbb{R}^{O^c}$, such that every pattern $\mathfrak{P}(\mathbf{a})$ performs a unitary embedding.

For a tuple $(G, I, O, \mathbf{a})$, if the geometry (G, I, O) is a "*yes*" instance of **GMPI**, we can then solve **MPI** by constructing the particular pattern $\mathfrak{P}(\mathbf{a})$.

The genericity of the measurement angles corresponds to the constraint that each measurement may be performed with respect to an arbitrary observable which anticommutes with Z. The approach taken by analysis with gflows, as we described in Section 3.6.5, is to determine whether this suffices to ensure unitarity of the post-selection procedure via the stabilizer formalism. In the case that the stabilizer formalism does guarantee the unitarity of the transformation, we also obtain a correction strategy which provides an interpolation of the measurement procedure to all possible measurement sequences. Recall the definition of a gflow:

Definition 4-3. Let (G, I, O) be a geometry and λ be a function defined on O^c indicating measurement planes as above; and let $\mathcal{P}$ be the power-set function, and odd(S) be the set of vertices in G which are adjacent to an odd number of elements of S. Then a *gflow* for (G, I, O, λ) is a tuple $(g, \preccurlyeq)$ consisting of a function

$g : O^{\mathsf{c}} \longrightarrow \mathcal{P}(I^{\mathsf{c}})$ and a partial order $\preccurlyeq$ which satisfy the following conditions:

$$w \in g(v) \implies v \preccurlyeq w, \tag{4.28a}$$
$$w \in \mathrm{odd}(g(v)) \implies v \preccurlyeq w, \tag{4.28b}$$
$$\lambda(v) = \mathsf{XY} \implies v \in \mathrm{odd}(g(v)) \smallsetminus g(v), \tag{4.28c}$$
$$\lambda(v) = \mathsf{YZ} \implies v \in g(v) \smallsetminus \mathrm{odd}(g(v)), \tag{4.28d}$$
$$\lambda(v) = \mathsf{XZ} \implies v \in g(v) \cap \mathrm{odd}(g(v)). \tag{4.28e}$$

As we saw in Section 3.6.5, the function $g : O^{\mathsf{c}} \longrightarrow \mathcal{P}(I^{\mathsf{c}})$ essentially encodes a list of generators $\{S_j\}_{j=1}^{n-k}$ for the state-space prepared by the map $\mathsf{E}_G \circ \mathsf{N}_{I^{\mathsf{c}}}$. Provided that the qubits $v \in O^{\mathsf{c}}$ are measured in an order extending $\preccurlyeq$, the set $g(v)$ describes the set of qubits acted on with the X operations of some stabilizer generator; qubits $w \in \mathrm{odd}(g(v))$ are acted on with Z operations, and qubits in $g(v) \cap \mathrm{odd}(g(v))$ are acted on by both (or equivalently, by a $Y \propto XZ$ operation). The generator S_j described then anticommutes with the measurement observable $\mathcal{O}_v^{\lambda(v),\alpha_v}$ defined in (3.52), for α_v arbitrary. The post-measurement state corresponding to the result $\mathsf{s}[v] = 0$ can then be selected by performing S_j as a post-measurement correction, which we may re-express as an adaptation the measurement observables for the qubits on which it acts.

Geometries of unit weight with XY-plane gflows

In the case where λ is a constant function $\lambda : O^{\mathsf{c}} \longrightarrow \{\mathsf{XY}\}$ and where $|V(G)| = n$, Mhalla and Perdrix [96] present an efficient algorithm to find a gflow $(g, \preccurlyeq)$, where $\preccurlyeq$ is defined in terms of a layer function $\ell : V(G) \longrightarrow \mathbb{N}$ such that $v \preccurlyeq w \iff [\, v = w \text{ or } \ell(v) > \ell(w) \,]$. This leads immediately to a solution to **MPI** in the case of unit weight geometries (G, I, O) when the geometry has a gflow, as follows. We may obtain a linear order $\leqslant$ extending $\preccurlyeq$ by constructing level sets $\mathcal{V}_j^{\prec}$ for each $j \in \mathbb{N}$, inserting each vertex $v \in V(G)$ into the set $\mathcal{V}_{\ell(v)}^{\prec}$. Using the representation of sets described in Section 3.4, the sets $\mathcal{V}_j^{\prec}$ are themselves linearly ordered by construction, leading to a linear ordering of $V(G)$ by merging these linear orders in such a way that any $v \in \mathcal{V}_{j+1}^{\prec}$ precedes any $w \in \mathcal{V}_j^{\prec}$ for each j. We may then simulate the postselection of the residual state corresponding to the result $\mathsf{s}[v] = 0$ for each qubit $v \in O^{\mathsf{c}}$, by composing patterns of the form

$$\left[\prod_{u \in g(v)} \mathsf{X}_u^{\mathsf{s}[v]}\right] \left[\prod_{\substack{w \in \mathrm{odd}(g(v)) \\ w \neq v}} \mathsf{Z}_w^{\mathsf{s}[v]}\right] \mathsf{M}_v^{\alpha_v} \tag{4.29}$$

in the ordering given by $\leqslant$. (The operations used here are those of the OneWay($\mathbb{R}$) model defined by Definition 2-4. The restriction of $w \neq v$ in the right-most product is due to the fact that $v \in \mathrm{odd}(g(v))$ has been discarded by the measurement.) Then, the following CPTP map performs a unitary embedding for any choice of angles $\mathbf{a} = (\alpha_v)_{v \in O^{\mathsf{c}}}$:

$$\left(\prod_{v \in O^{\mathsf{c}}}^{\geqslant} \left[\prod_{u \in g(v)} \mathsf{X}_u^{\mathsf{s}[v]}\right] \left[\prod_{\substack{w \in \mathrm{odd}(g(v)) \\ w \neq v}} \mathsf{Z}_w^{\mathsf{s}[v]}\right] \mathsf{M}_v^{\alpha_v}\right) \left(\prod_{uv \in E(G)} \mathsf{E}_{u,v}\right) \left(\prod_{v \in I^{\mathsf{c}}} \mathsf{N}_v^{\mathsf{x}}\right), \tag{4.30}$$

where the left-most product is ordered left-to-right according to the order $\geqslant$. By the analysis of Section 4.1.4, if $(G, I, O, \mathbf{a})$ is an instance of the **MPI** derived from a quadratic form expansion for an operator U, and (G, I, O) is a unit weight geometry, the unitary operation which is performed is U.

Run-time analysis. Let $n = |V(G)|$, $m = |E(G)|$, and $k = |O| \geqslant |I|$. Determining whether a geometry has a gflow for the special case where all measurements are performed in the XY-plane, and constructing the gflow $(g, \preccurlyeq)$ if so, can be done in time $O(n^4)$ by Algorithm 2 of [96]. Having a gflow $(g, \preccurlyeq)$, the measurement procedure in (4.30) is well-defined and requires $O(n^3)$ time to construct, dominated by the product over $v \in O^{\mathsf{c}}$ of the correction operations for $u \in g(v)$ and $w \in \mathrm{odd}(g(v))$, where the latter require time $O(m) \subseteq O(n^2)$ to compute. Bringing this measurement procedure into a standard form by the standardization procedure of Section 2.2.5 also requires time $O(n^3)$; the resulting measurement pattern contains $n - k$ measurement operations and up to k correction operations, each of which involves $O(n)$ measurement dependencies. We may therefore conclude:

Theorem 4-2. *For a unitary embedding U given as a quadratic form expansion, with unit weight geometry (G, I, O) such that $|V(G)| = n$, there is an $O(n^4)$ algorithm which either determines that (G, I, O) has no gflow, or constructs pattern in the* OneWay($\mathbb{R}$) *model whose default measurement angles are given by the square terms of the quadratic form and consisting of $O(n^2)$ operations and which performs the transformation U.*

As we note in Theorem 4-2, the one-way measurement pattern constructed by this technique is in the model OneWay($\mathbb{A}$), where $\mathbb{A} \subseteq \mathbb{R}$ contains the coefficients for the square terms of the quadratic form. We may strengthen this to obtain an algorithm producing one-way patterns in a fixed-precision model of measurement-based quantum computing using any set $\mathbb{A}$ sufficient for generating single-qubit unitaries, such as $\mathbb{A} = \frac{\pi}{4}\mathbb{Z}$. A measurement in the XY-plane with respect to the basis $|\pm_\alpha\rangle$ can be transformed into a Z-axis measurement by a $J(-\alpha)$ rotation (1.62): more generally, we have

$$\mathsf{M}_v^{\alpha;\beta} \quad = \quad \mathsf{M}_v^{\mathsf{z}}\ \mathsf{J}_v^{(-1)^{1+\beta}\cdot\alpha}\ . \tag{4.31}$$

By the universality of the Jacz($\frac{\pi}{4}\mathbb{Z}$) gate set defined on page 27, and by the Solovay-Kitaev Theorem, we may approximate the unitary operator $J(\alpha) \in \mathsf{U}(2)$ to any precision $\varepsilon > 0$ by some product of N operations of the form $J(\theta_j)$ for $\theta_j \in \frac{\pi}{4}\mathbb{Z}$, where $N \in \mathrm{polylog}(1/\varepsilon)$:

$$\Big\| J(\alpha) \ - \ J(\theta_{N-1}) \cdots J(\theta_1) J(\theta_0) \Big\|_\infty < \ \tfrac{\varepsilon}{2}\ . \tag{4.32}$$

This approximation can be found in time $\mathrm{polylog}(1/\varepsilon)$ time by the Solovay-Kitaev theorem. In order to represent the conditional change of sign, note that

$$J(-\alpha) \quad = \quad HR_z(-\alpha) \quad = \quad HXR_z(\alpha)X \quad = \quad ZJ(\alpha)X\ . \tag{4.33a}$$

Then, we may approximate the effect of a measurement M_v^α for arbitrary $\alpha \in \mathbb{R}$ by a (non-standardized) measurement pattern of the form

$$\begin{aligned}\mathsf{M}_v^{\alpha;\beta} &\cong \mathsf{M}_{v'}^{\mathsf{z}}\,\mathsf{J}_{v/v'}^{(-1)^{1+\beta}\cdot\alpha} \\ &\approx_\varepsilon \mathsf{M}_{v'}^{\mathsf{z}}\;\mathsf{Z}_{v'}^{1+\beta}\;\mathfrak{J}_{v'/a_{N-1}}^{\theta_{N-1}}\;\cdots\;\mathfrak{J}_{a_2/a_1}^{\theta_2}\;\mathfrak{J}_{a_1/v}^{\theta_1}\;\mathsf{X}_v^{1+\beta} \\ &= \mathsf{M}_{v'}^{\mathsf{z}}\;\mathfrak{J}_{v'/a_{N-1}}^{\theta_{N-1}}\;\cdots\;\mathfrak{J}_{a_2/a_1}^{\theta_2}\;\mathfrak{J}_{a_1/v}^{\theta_1}\;\mathsf{X}_v^{1+\beta}\;, \end{aligned} \tag{4.34}$$

using the $\mathfrak{J}_{u/v}^\theta$ patterns defined in (2.24); we introduce the auxiliary qubits $a_1, \ldots, a_{N-1}$ in order to approximate $\mathfrak{J}_{v/v'}^\alpha$ to precision ε using the approximation of (4.32), and substituting the measurement result $\mathsf{s}[v']$ throughout the rest of the measurement pattern wherever $\mathsf{s}[v]$ occurs.

To obtain an approximation of the CPTP map $\rho \longmapsto U\rho U^\dagger$ with precision ε, it suffices to approximate each of the $n-k$ measurement operations with precision $\frac{\varepsilon}{n-k}$ by the development of (1.50). We then obtain the following corollary:

Corollary 4-2a. *For a unitary embedding U given as a quadratic form expansion, with unit weight geometry (G, I, O) such that $|V(G)| = n$, for each $\varepsilon > 0$ and $\delta > 0$ there exists an algorithm running in time $O(n^4 + n\log(n/\varepsilon)^{3+\delta})$ which either determines that (G, I, O) has no gflow, or constructs a pattern in the* OneWay$(\frac{\pi}{4}\mathbb{Z})$ *model consisting of $O(n^2 + n\log(n/\varepsilon)^{3+\delta})$ operations and which approximates the CPTP map $\rho \longmapsto U\rho U^\dagger$ to precision ε.*

Geometries of fractional weight with flows

Geometries with fractional edges correspond, as we remarked earlier, to extended one-way measurement procedures with include entangling operations of the form $\mathsf{E}_{u,v}^t(\rho) = \wedge Z^t\,\rho\,\wedge Z^{-t}$. The state space of the system is then not necessarily a stabilizer code, in which case the approach given above for gflows then do not directly apply.

As we noted in Section 4.1.4, we may attempt to treat such quadratic form expansions by applying the substitutions of (4.26) in order to obtain an equivalent expansion which only involved cross-term angles in $\{0, \pi\}$. However, this approach still fails to produce a problem instance approachable by the technique of gflows, as it produces a quadratic form with an extra auxiliary index a_1 which has no square term. It is easy to see that the absence of an a_1^2 term with a generic coefficient in this expansion is crucial, by considering the corresponding formula where the quadratic form includes a θa_1^2 term:

$$\begin{aligned}&\frac{1}{2}\sum_{\mathbf{a}\in\{0,1\}^2} \mathrm{e}^{i\left[\pi(x_u a_1 + x_v a_1 + a_1 a_2) + \theta a_1^2 + \phi a_2^2\right]} \\ &\quad= \frac{1}{2}\left[\mathrm{e}^{i[0]} + \mathrm{e}^{i[\pi(x_u+x_v)+\theta]} + \mathrm{e}^{i[\phi]} + \mathrm{e}^{i[\pi(x_u+x_v+1)+\theta+\phi]}\right] \\ &\quad= \frac{\mathrm{e}^{i\phi/2}}{2}\left[\left(\mathrm{e}^{-i\phi/2} + \mathrm{e}^{i\phi/2}\right) + \mathrm{e}^{i\theta}(-1)^{x_u\oplus x_v}\left(\mathrm{e}^{-i\phi/2} - \mathrm{e}^{i\phi/2}\right)\right] \\ &\quad= \mathrm{e}^{i\phi/2}\left[\cos(\tfrac{\phi}{2}) - i\,\mathrm{e}^{i\theta}(-1)^{x_u\oplus x_v}\sin(\tfrac{\phi}{2})\right]. \end{aligned} \tag{4.35}$$

This expression is not a complex unit for all values of $x_u, x_v \in \{0,1\}$ and angles $\phi \in (-\pi, \pi]$; in particular, for $\phi = \pi/2$, we may see that

$$\frac{1}{2} \sum_{\mathbf{a} \in \{0,1\}^2} \mathrm{e}^{i\left[\pi(x_u a_1 + x_v a_1 + a_1 a_2) + \frac{\pi}{2} a_1^2 + \frac{\pi}{2} a_2^2\right]} = \left\{ \begin{array}{ll} 1+i\,, & \text{if } x_u \oplus x_v = 0 \\ 0\,, & \text{if } x_u \oplus x_v = 1 \end{array} \right\}. \tag{4.36}$$

This is a very different expression from the expression $\mathrm{e}^{i\phi(x_u \oplus x_v)}$ in the case $\theta = 0$: thus, even if the result is a unitary in the latter case, it is by no means clear that quadratic form expansions with the same geometry should be "generically" unitary, regardless of the coefficients of their square terms. (As an extreme example, where $u, v \in I \cap O$, the resulting operation would be singular for $\theta = \phi = \pi/2$, even though the operator may be unitary for $\theta = 0$.) However, this genericity is necessary for the geometry to be a "yes" instance of the **GMPI**.

We can attempt to resolve this by noting that the substitutions in (4.26) are a result of an extended measurement-based procedure which allows arbitrary measurements in the YZ plane: measurements on the two auxiliary qubits a_1 and a_2 of (4.26), where a_1 is measured with an X-measurement and a_2 with an arbitrary XY-plane measurement, are equivalent to an arbitrary YZ-plane measurement on a single auxiliary qubit, as illustrated in the middle panel of Figure 4-1. In particular, as we noted in Section 3.6.1, this construction has an eflow, which is a special case of a gflow where $g(v)$ are singleton sets for each $v \in O^c$. We may then treat such quadratic form expansions by performing a transformation of the underlying *geometry* (G, I, O), to yield a unit-weight geometry (G', I, O) with additional information indicating sets of vertices T to be measured in the YZ plane, and try to find an eflow for (G', I, O, T). We will use a similar technique in Section 4.2.2 to synthesize one-way measurement patterns for certain cases involving auxiliary vertices of degree 1.

For a unitary U which is specified by a quadratic form expansion, whose geometry has fractional weight edges but which can be transformed into an equivalent unit-weight geometry (G', I, O) and set T such that (G', I, O, T) has an eflow $(f, \preccurlyeq)$, we may apply the same techniques as outlined above for gflows to efficiently arrive at an implementation (or approximation) of U. In this case, the vertices of T can be identified with the edges of fractional weight in G; conversely, any qubit $v \in V(G) \setminus O$ lies outside of T, and so the edges $v\, f(v)$ in G are of unit weight. By the comments at the end of Section 3.6.1, the star geometry rooted at vertices $v \in T$ correspond to operations $\wedge Z^t$ in a unitary circuit via the simplified RBB construction, with the edges of the form $v\, f(v)$ representing $J(\theta)$ gates as in the analysis of flows in terms of the DKP construction.

There is, however, a simpler approach to synthesizing an implementation for U by observing that the above also essentially describes an extension of the semantic map `SemanticDKP` constructed in Section 3.5, where we ignore the absence of information about the adaptation of measurements, and allow non-flow edges to have fractional weight to represent fractional powers of $\wedge Z$ in the resulting unitary circuit. First, if necessary, we transform the quadratic form expansion into a form where each $v \in O$ has $\theta_{vv} = 0$, using the technique described on page 169: it is easy to verify that the flow function can be extended to the new vertex set arising from this transformation, so we may without loss of generality suppose that the input geometry has $\theta_{vv} = 0$ for $v \in O$. Then, we may straightforwardly interpret the quadratic form expansion as a unitary circuit, by

noting that it exactly represents the matrix formula produced by making explicit the summations in a stable index tensor product:

(*i*) the edges vw for $w = f(v)$ correspond to a pair of deprecated/advanced tensor indices in an operation $J(\theta_{vv})\left[\frac{w}{v}\right]$ operation;

(*ii*) any non-flow edge vw corresponds to a pair of stable tensor indices in a $\wedge Z\left[{}^{v,w}\right]$ operation.

In the above, we still obtain a unitary circuit by replacing non-flow edges with edges of fractional weight; these correspond to $\wedge Z^t\left[{}^{v,w}\right]$ operations for fractional powers t. This motivates an extension of the definition of flows to fractional-weight geometries:

Definition 4-4. Suppose (G, I, O) is a fractional-weight geometry. A flow for (G, I, O) is then an ordered pair $(f, \preccurlyeq)$ satisfying properties (3.3a) – (3.3c), such that for all $ab \in E(G)$ with $W_G(ab) < 1$, we also have $f(a) \neq b$ and $f(b) \neq a$ (*i.e.*, all flow-edges in G have unit weight).

If a fractional-weight geometry (G, I, O) has a flow, and if the square terms for $v \in O$ are all zero, we may synthesize a circuit from the quadratic form expansion for U using the description above.

We may verify the correctness of this circuit construction by inducting on the number of edges of fractional weight. For the base case, if (G, I, O) has a known flow $(f, \preccurlyeq)$ and no fractional-weight edges, we may synthesize a circuit for U using the algorithm `SemanticDKP` (using the flow $(f, \preccurlyeq)$ rather than searching for one using the procedure `FindFlow`). The resulting circuit has a gate $J(\theta_{vv})\left[\frac{x_{f(v)}}{x_v}\right]$ for each $v \in O^{\mathsf{c}}$ and a gate $\wedge Z\left[{}^{x_v,x_w}\right]$ for each non-flow edge $vw \in E(G)$, which can be expressed as

$$C \;=\; \left(\prod_{v\in O^{\mathsf{c}}} J(\theta_{vv})\left[\frac{x_{f(v)}}{x_v}\right]\right)\left(\prod_{\substack{vw\in E(G)\\ v\neq f(w)\\ w\neq f(v)}} \wedge Z\left[{}^{x_v,x_w}\right]\right) \tag{4.37}$$

in stable index notation. We may then induct for geometries with fractional edge-weights if we can show we can decompose the geometry into ones with fewer fractional edge-weights.

For an arbitrary fractional edge $ab \in E(G)$ and each each $z \in O$, we may define $m(ab, z)$ to be the maximal vertex $v \in V(G)$ in the ordering $\preccurlyeq$ subject to z being in the orbit of v under f (that is, $z = f^{\ell}(v)$ for some $\ell \geqslant 0$), such that at least one of $v \preccurlyeq a$ or $v \preccurlyeq b$ holds. For a set $S \subseteq V(G)$, let $G[S]$ represent the subgraph of G induced by S (*i.e.* by deleting all vertices in G not in S). Then, define the following subgraphs of G, and corresponding geometries:

- Let V_2 be the set of vertices $m(ab, z)$ for each $z \in O^{\mathsf{c}}$: it is easy to show that $a, b \in V_2$. Let $G_2 = G[V_2]$, and let $\mathcal{G}_2 = (G_2, V_2, V_2)$.
- Let V_1 be the set of vertices $u \in V(G)$ such that $u \preccurlyeq v$ for some $v \in V_2$; let $G_1 = G[V_1] \setminus \{uv \,|\, u, v \in V_2\}$; and let $\mathcal{G}_1 = (G_1, I, V_2)$.

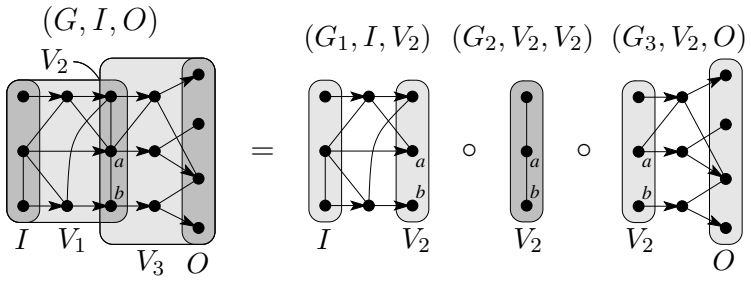


FIGURE 4-2. Illustration of the decomposion of a quadratic form expansion about an edge ab, expressed in terms of geometries. (Again, the composition of geometries $(G_3, V_2, O) \circ (G_2, V_2, V_2) \circ (G_1, I, V_2)$ is illustrated in the opposite order in the diagram above.) V_2 is a set of maximal vertices under the constraint of being bounded from above, by the vertices a and b, in a partial order $\preccurlyeq$ associated with a flow for the fractional-edge geometry (G, I, O). Arrows represent the action of the corresponding flow function f.

- Let V_3 be the set of vertices $u \in V(G)$ such that $u \succcurlyeq v$ for some $v \in V_2$; let $G_3 = G[V_3] \setminus \{uv \,|\, u, v \in V_2\}$; and let $\mathcal{G}_3 = (G_3, V_2, O)$.

This decomposes the geometry (G, I, O) into three (possibly still fractional-weight) geometries with flows, as illustrated in Figure 4-2.

Let Q_1 be a quadratic form on $\{0,1\}^{V_1}$ consisting of the terms $x_u x_v$ of Q for $u \in V_1$ or $v \in V_1$, but not both; Q_2 be a quadratic form on $\{0,1\}^{V_2}$ consisting of the terms $x_u x_v$ of Q for *distinct* $u, v \in V_2$; and similarly let Q_3 be defined on $\{0,1\}^{V_3}$, and consist of the remaining terms of Q. Then Q_1, Q_2, and Q_3 define quadratic form expansions for some operations U_1, U_2, and U_3 (respectively) with geometries $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$ (respectively).

- U_2 in particular will be a product of operations $\wedge Z^{W_G(uv)}$ for distinct $u, v \in V_2$, as it is a quadratic form expansion whose input and output indices coincide. Then U_2 can be represented as a circuit with a wire for each $u \in V_2$, with fractional controlled-Z gates $\wedge Z^{W_G(uv)}$ for each edge $uv \in E(G)$.

- Both $\mathcal{G}_1$ and $\mathcal{G}_3$ have flows, and fewer fractional edges than (G, I, O). By induction, U_1 and U_3 are also unitary embeddings, and have stable index expressions including terms $J(\theta_{vv})\left[\frac{f(x_v)}{x_v}\right]$ for each $v \in V(G_j) \setminus O_j$ and $\wedge Z^{W_G(uv)}\left[x_u, x_v\right]$ for each non-flow edge $uv \in E(G_j)$, for $j \in \{1, 3\}$.

Because $Q_1(\mathbf{x}_{V_1}) + Q_2(\mathbf{x}_{V_2}) + Q_3(\mathbf{x}_{V_3}) = Q(\mathbf{x})$ for all $\mathbf{x} \in \{0,1\}^V$ by construction, the composite operation $U_3 U_2 U_1$ can differ from U by at most a scalar factor by Lemma 4-1. We then obtain a composite circuit

$$C \;=\; \left(\prod_{v \in O^{\mathsf{c}}} J(\theta_{vv})\left[\frac{x_{f(v)}}{x_v}\right] \right) \left(\prod_{\substack{vw \in E(G) \\ v \neq f(w) \\ w \neq f(v)}} \wedge Z^{W_G(vw)}[x_v, x_w] \right) \tag{4.38}$$

performing the operation U, generalizing the construction given for unit-weight geometries described in (4.37). A direct way to verify that this circuit is correct is to expand

the tensor product in terms of the coefficients of each operator $J(\theta)$ and $\wedge Z^t$:

$$J(\theta_{vv})\left[\begin{smallmatrix} x_w \\ x_v \end{smallmatrix}\right] = \frac{1}{\sqrt{2}}\,\mathrm{e}^{i(-\theta_{vv}/2 + \theta_{vv}x_v + \pi x_v x_w)}$$
$$\propto \frac{1}{\sqrt{2}}\,\mathrm{e}^{i(\theta_{vv}x_v^2 + \pi x_v x_w)}\,, \tag{4.39a}$$

$$\wedge Z[x_v, x_w] = (-1)^{x_v x_w}\,; \tag{4.39b}$$

by summing over indices which are both advanced and deprecated, we may then verify that the coefficients of the unitary operation performed by C are the same as those of U (up to a product of the scalar factors $\mathrm{e}^{-i\theta_{vv}/2}$ in the proportionality of (4.39a) above).

Run-time analysis. Let $n = |V(G)|$, $m = |E(G)|$, and $k = |O|$. Determining whether a geometry has a flow, and constructing the flow $(f, \preccurlyeq)$ if so, can be done in time $O(kn)$ by Algorithm 1 of [96]. For each edge uv, we may check whether one of $W_G(uv) = 1$ or $\big[u \neq f(v)$ and $v \neq f(u)\big]$ holds: if all edges satisfy this constraint, the unitary circuit described above is well-defined, and may be constructed in time $O(m)$ by iterating through the edges of $V(G)$. By the extremal result of Theorem 3-7, any geometry with a flow has $m \leqslant kn - \binom{k+1}{2}$: by eliminating geometries with more edges than this, we may reduce the total running time of this algorithm to $O(kn)$.

In the case $|I| = |O|$, a flow function f is unique if it exists by Theorems 3-6 and 3-11 (pages 93 and 99, respectively); so in this case, if (G, I, O) has a flow when we neglect the edge-weights of G, but there is an edge $v\,f(v)$ of fractional weight, there is no flow for (G, I, O) considered as a fractional-weight geometry. Finally, note that there is an operation in the circuit for each edge in the geometry: if the original quadratic form expansion had non-zero coefficients θ_{vv} for $v \in O$, the *expanded* geometry (G', I, O') obtained by applying the technique on page 169 will have at most $m + 2k \leqslant kn - \binom{k+1}{2} + 2k \leqslant kn + 1$ edges (where $m = |E(G)|$ is the number of edges of the *original* geometry), by the upper bound of Theorem 3-7. Therefore, we have the following result:

Theorem 4-3. *For a transformation $U \in \mathsf{U}(2^k)$ given as a quadratic form expansion with geometry (G, I, O) such that $|V(G)| = n$ and $|E(G)| = m$, there is an $O(kn)$ algorithm which either determines that (G, I, O) has no flow, or constructs a stable index expression representing a unitary circuit consisting of at most $m+2k \leqslant kn+1$ operations which performs U.*

In order to obtain a circuit using a finite elementary gate set, we may again approximate each gate $J(\theta_{vv})$ and each gate $\wedge Z^t$ to precision $\frac{\varepsilon}{m}$ by a product of N gates in the $\textsc{Jacz}(\frac{\pi}{4})$ model, where $N \in \mathrm{polylog}(m/\varepsilon)$ for arbitrary $\varepsilon > 0$, by the Solovay-Kitaev Theorem; and by the analysis of (1.50), this suffices to approximate U to precision ε. We then obtain:

Corollary 4-3a. *For a transformation $U \in \mathsf{U}(2^k)$ given as a quadratic form expansion with geometry (G, I, O) such that $|V(G)| = n$ and $|E(G)| = m$, for each $\varepsilon > 0$ and $\delta > 0$ there exists an algorithm running in time $O(kn \log(kn/\varepsilon)^{3+\delta})$ which either determines that (G, I, O) has no flow, or constructs a stable index*

expression representing a unitary circuit in the $\textsc{Jacz}(\frac{\pi}{4}\mathbb{Z})$ *model, which consists of* $O(kn\log(kn/\varepsilon)^{3+\delta})$ *operations and approximates* U *to precision* ε.

The quantum Fourier transform as an example

For the most part, quadratic form expansions have occurred in the literature as analytical tools rather than as the preferred means of representing unitary transformations of interest. While there is a possible source in path integrals, as we remarked on page 161, it is not yet known whether expressions resulting from discretizing path integrals would yield tuples (G, I, O, λ) with generalized flows. At the same time, the study of flows and gflows has thus far been essentially restricted to their potential as tools to analyze procedures in the one-way model, in an attempt to characterize procedures which perform unitaries under certain constraints (as in the **GMPI**) or provide automatically verifiable semantics for measurement-based computations (as in Chapter 3). Therefore, there are few known "natural" examples of a unitary whose definition lends itself to the techniques of this section.

However, as we remarked in Section 4.1.2, there is one very prominent example of a unitary transformation which is naturally defined as a quadratic form expansion: the quantum Fourier transform $\mathcal{F}_{2^k}$ over $\mathbb{Z}_{2^k}$. Recall that $\mathcal{F}_{2^k}$ has an expansion with the quadratic form

$$Q(\mathbf{x},\mathbf{y}) \;=\; \sum_{h=0}^{k-1}\sum_{j=0}^{k-h-1} \frac{2^{(h+j)}}{2^{k-1}}\,\pi\, x_j\, y_h\;, \tag{4.40}$$

where the vectors $\mathbf{x}$ and $\mathbf{y}$ correspond to the qubits of the input system I and the output system O, respectively. The (fractional-edge) geometry $\mathcal{G}$ arising from this quadratic form expansion is illustrated[4] in Figure 4-3 for the case $n = 5$.

We may show that the fractional-weight geometry $\mathcal{G}$ has a flow without having to perform a flow-finding algorithm, as follows. Because the only edges which have unit weight in $\mathcal{G}$ are those of the form $x_j y_{k-j-1}$ for each $0 \leqslant j < k$, these must be the flow edges if $\mathcal{G}$ has a flow. To show that the function $f(x_j) = y_{k-j-1}$ is a flow-function, consider the influence relation $\lhd$ induced by f (Definition 3-4, page 89). By construction, the graph of the geometry $\mathcal{G}$ is a bipartite graph, with partitions $\{x_j\}_{j=0}^{k-1}$ and $\{y_h\}_{h=0}^{k-1}$; then, for each vertex x_j, aside from $f(x_j)$, the only vertices w such that $x_j \lhd w$ are the other vertices x_ℓ adjacent to y_{k-j-1}. The set of vertices adjacent to a given vertex y_h are the vertices x_ℓ for $0 \leqslant \ell \leqslant k-h-1$, by (4.40). Thus, the vertices x_ℓ such that $x_j \lhd x_\ell$ satisfy $\ell \neq j$ and $0 \leqslant \ell < k-(k-j-1)-1$, which holds if and only if $0 \leqslant \ell < j$. Then, every walk in the influence digraph $\mathfrak{I}_f$ (Definition 3-6, page 90) either ends at a vertex y_h for some h, or visits a sequence of vertices $x_{s_1}x_{s_2}\cdots$ for some monotonically decreasing sequence $\{s_j\}_{j\geqslant 1}$ of indices. In particular, $\mathfrak{I}_f$ contains no circuits; thus, by Lemma 3-3 (page 90), the geometry $\mathcal{G}$ has a fractional-edge flow.

Illustrated in Figure 4-3 alongside the presentation of $\mathcal{G}$ is a representation of the circuit that may be constructed from $\mathcal{G}$ together with the flow-function f. Note that

[4] The problem of finding suitable graphical presentations of graphs is an open research problem in computational geometry; the presentation of the geometry in Figure 4-3 was chosen by hand to best exhibit the symmetries of the geometry.

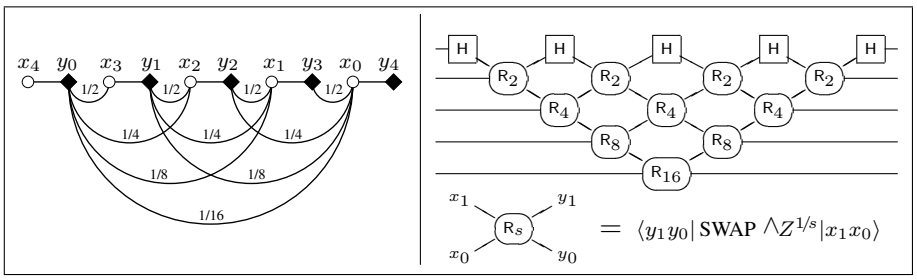


FIGURE 4-3. The geometry for the quadratic form expansion of the QFT for $\mathbb{Z}_{32}$, and the corresponding circuit due to [66]. In the geometry (on the left), input vertices are labelled by circles, output vertices by lozenges, and fractional edges are labelled with their edge-weights.

the static index tensor representation of unitary circuits in the JACZ($\mathbb{R}$) model does not impose any *a priori* restrictions on the layout or interactions of the qubits (in particular, it does not explicitly represent SWAP gates) — partly because the gate-set JACZ($\mathbb{R}$) itself places no restrictions on interactions between qubits (and doesn't contain SWAP gates). However, by

(a) representing the input and output wires in the conventional way, with vertical position increasing in order of the significance of each bit;

(b) identifying the edges of unit weight in the geometry with $H = J(0)$ operations[5] in the unitary circuit, and arranging the other elements of the circuit diagram about those gates; and

(c) allowing the circuit wires to cross each other, and run in directions other than horizontally (in particular, producing a crossing for each fractional edge illustrated),

we are led to a presentation of this circuit as above, where we have collected the wire-crossings and fractional $\wedge Z$ operations into a family of compound gates as illustrated.

The circuit illustrated in Figure 4-3 was first described by Fowler, Devitt, and Hollenberg in [66], and is presented in a similar graphical format in [113]: therefore, the above presentation is neither new nor an independent discovery. This example is meant only to illustrate simple heuristics which, together with the techniques of the preceding sections, can at least reproduce what is known for the quantum Fourier transform, and in a model which places spatial restrictions on multi-qubit operations (unlike the PHASES($\mathbb{A}$) or JACZ($\mathbb{A}$) models).

4.2.2 Geometries with simple modifications admitting eflows

As we noted on pages 174 – 175, there are instances of quadratic form expansions which represent unitary operations, and where this unitarity may depend crucially on the value of some of the coefficients θ_{vv} for $v \in O^{\mathsf{c}}$. However, such quadratic form expansions can nevertheless be related to generalized flows, if we can identify it with a postselection

[5] The presence of Hadamards in the circuit is due to the lack of any square terms in the quadratic form, corresponding to measurement angles of 0 in the post-selection description of the unitary operation; this in turn leads to $J(0)$ gates in the constructed circuit.

procedure in which constrained postselections in the XY plane can be regarded as allowing the postselection of states in some other plane (such as the YZ plane) to be simulated.

In a one-way measurement pattern, a qubit $v \in O^c \cap I^c$ with degree 1 may represent (considered as a simulation of some element of a unitary circuit) either the introduction of an auxiliary qubit prepared in the $|+\rangle$ state which undergoes further transformations, or a terminal measurement on some qubit which has undergone some transformations prior to measurement. In the latter case, the preceding transformations may be interpreted as a change of reference frame prior to measurement, in order to simulate measurement in some other plane by an XY-plane measurement. In particular, to simulate a YZ-plane measurement with an XY-plane measurement, it suffices (as we noted on page 76) to perform a Hadamard transformation prior to the XY-plane measurement.

It is easy to verify that the only single-qubit rotations of the form $J(\theta)$ which do this are for $\theta \in \pi\mathbb{Z}$: measuring an observable $\mathcal{O}^{\mathsf{XY},\alpha} = \cos(\alpha)X + \sin(\alpha)Y$ as defined in (3.52) after performing a $J(\theta) = HR_{\mathsf{z}}(\theta)$ rotation is equivalent to measuring the observable

$$J(\theta)^\dagger \, \mathcal{O}^{\mathsf{XY},\alpha} \, J(\theta) \quad = \quad \cos(\alpha)\, Z \quad - \quad \sin(\alpha)\, R_{\mathsf{z}}(\theta)^\dagger Y R_{\mathsf{z}}(\theta) \,, \tag{4.41}$$

which is of the form $\mathcal{O}^{\mathsf{YZ},\alpha'}$ only for θ a multiple of π. Considering a geometry of unit weight (possibly obtained by performing the substitution described in (4.26)), any vertex $w \in V(G)$ of degree 1 which is adjacent to a vertex $v \in V(G)$ with $\theta_{vv} \in \{0, \pi\}$ may then be considered to be expressing a postselection procedure which selects for some state $|{+}\mathsf{yz}_\alpha\rangle_v$ as defined in (2.45). Furthermore, from the fact that

$$J(\alpha)J(0)J(\beta)J(\gamma) \quad = \quad HR_{\mathsf{z}}(\alpha)R_{\mathsf{z}}(\beta)J(\gamma) \quad = \quad J(\alpha+\beta)J(\gamma) \,, \tag{4.42a}$$

$$J(\alpha)J(\pi)J(\beta)J(\gamma) \quad = \quad HR_{\mathsf{z}}(\alpha)XR_{\mathsf{z}}(\beta)HR_{\mathsf{z}}(\gamma) \quad = \quad J(\alpha-\beta)J(\gamma+\pi) \,, \tag{4.42b}$$

we may extend this to arbitrary paths of odd length: if G contains a path of the form $v_0v_1v_2\cdots v_{N-1}v_N$ for N odd such that v_N has degree 1, v_j has degree 2 for $1 \leqslant j < N$, and $v_j \in \pi\mathbb{Z}$ for j even, the corresponding postselection procedure may be regarded as simulating the postselection of some state in the YZ plane state on the qubit v_0. More generally, we may consider *arbitrary* products of operators $J(\theta)$ which map measurement observables of interest (*e.g.*, observables of the form $\mathcal{O}^{\mathsf{YZ},\alpha}$ or $\mathcal{O}^{\mathsf{XZ},\alpha}$) to XY-plane observables.

By detecting when chains of vertices in a geometry represent postselections in either the YZ or XZ plane, we can obtain smaller geometries and a measurement-plane function λ which may represent "*yes*" instances of **GMPI** (where we extend the domain of that problem in the natural way to include measurement operations in each of the XY, YZ, and XZ planes). Thus, for any algorithm for finding gflows for non-constant maps λ, this yields another approach to synthesizing measurement-based patterns from quadratic form expansions, by contracting paths which may be regarded as representing the simulation of a postselection of observables in the YZ, or XZ planes.

Currently, the only efficient algorithm for obtaining gflows for (G, I, O, λ) for λ not a constant function is Algorithm 3-10, which determines whether or not a tuple (G, I, O, T) has an eflow. As we noted on page 154, this is equivalent to determining whether the tuple (G, I, O, λ) has a gflow, for λ given by

$$\lambda(v) \quad = \quad \left\{ \begin{array}{ll} \mathsf{YZ}\,, & \text{if } v \in T \\ \mathsf{XY}\,, & \text{otherwise} \end{array} \right\} \,, \tag{4.43}$$

and where we require that $g(v)$ be a singleton set for each $v \in O^{\mathsf{c}}$. We may then attempt to map a quadratic form expansion provided as input to a unit-weight geometry (G, I, O) and a set of vertices $T \subseteq V(G)$ such that (G, I, O, T) has an eflow.

Treating fractional-weight geometries

As we have noted before, by applying the substitution of (4.26), we may transform any geometry with fractional edges, corresponding to coefficients $\theta_{uv} \notin \pi\mathbb{Z}$ into a geometry with unit edge-weights. As noted in Figure 4-1, this will yield a certain number of pairs of qubits which represent postselections in the YZ plane: this should be immediately apparent from the fact that the substitution amounts to a decomposition of $\wedge Z^t$ into a pattern of the form $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\pi t/2}$, which itself has an eflow.

In the case where the original geometry has a flow (in the sense defined for fractional-edge geometries in Definition 4-4), rather than attempting to obtain a unit-weight geometry with an eflow, it may prove more convenient (*e.g.* if the objective is to obtain a circuit representation of a unitary) to follow the analysis of pages 175 – 178.

Transformations of quadratic form expansions to find eflows

In order to consider whether a unit-weight quadratic form expansion can be mapped to a one-way measurement pattern using eflows, we may first simplify the expansion by taking advantage of identities of the same form as (4.42) above. Consider once more a quadratic form expansion

$$C \sum_{\mathbf{x}\in\{0,1\}^V} \mathrm{e}^{iQ(\mathbf{x})} \; |\mathbf{x}_O\rangle\langle\mathbf{x}_I| \qquad \text{where} \quad Q(\mathbf{x}) \;=\; \sum_{\{u,v\}\subseteq V} \theta_{uv} x_u x_v \;, \tag{4.44}$$

and suppose there are indices $a, b, c, v \in V \setminus (I \cup O)$ such that $Q(\mathbf{x})$ contains the terms

$$\begin{aligned} &\pi x_a x_b + \pi x_b x_c + \pi x_c x_v \\ &\quad + \theta_{aa} x_a^2 + \theta_{bb} x_b^2 + \theta_{cc} x_c^2 + \theta_{vv} x_v^2 \;, \end{aligned} \tag{4.45}$$

and no other terms in x_b, x_c, or x_v (though it may contain many other terms in x_a). In the geometry induced by this quadratic form expansion, v has degree 1, and b and c have degree 2. If $\theta_{cc} \in \pi\mathbb{Z}$, we may then apply one of the following equalities:

$$\begin{aligned} &\sum_{b,c,v} \exp\left(i\left[\pi x_a x_b + \pi x_b x_c + \pi x_c x_v + \theta_{aa} x_a^2 + \theta_{bb} x_b^2 + 0\cdot x_c^2 + \theta_{vv} x_v^2\right]\right) \\ &\quad = \sum_{b,v} \left(\mathrm{e}^{i\left[\pi x_a x_b + \theta_{aa} x_a^2 + \theta_{bb} x_b^2 + \theta_{vv} x_v^2\right]} \;+\; (-1)^{x_b \oplus x_v} \mathrm{e}^{i\left[\pi x_a x_b + \theta_{aa} x_a^2 + \theta_{bb} x_b^2 + \theta_{vv} x_v^2\right]}\right) \\ &\quad = \sum_{b,v} 2\,\delta_{b,v}\, \mathrm{e}^{i\left[\pi x_a x_b + \theta_{aa} x_a^2 + \theta_{bb} x_b^2 + \theta_{vv} x_v^2\right]} \\ &\quad = 2\sum_{v} \mathrm{e}^{i\left[\pi x_a x_v + \theta_{aa} x_a^2 + (\theta_{bb} + \theta_{vv}) x_v^2\right]} \;; \end{aligned} \tag{4.46a}$$

$$\begin{aligned}
&\sum_{b,c,v} \exp\left(i\left[\pi x_a x_b + \pi x_b x_c + \pi x_c x_v + \theta_{aa}x_a^2 + \theta_{bb}x_b^2 + \pi\cdot x_c^2 + \theta_{vv}x_v^2\right]\right) \\
&\quad= \sum_{b,v}\left(\mathrm{e}^{i\left[\pi x_a x_b+\theta_{aa}x_a^2+\theta_{bb}x_b^2+\theta_{vv}x_v^2\right]} - (-1)^{x_b\oplus x_v}\,\mathrm{e}^{i\left[\pi x_a x_b+\theta_{aa}x_a^2+\theta_{bb}x_b^2+\theta_{vv}x_v^2\right]}\right) \\
&\quad= \sum_{b,v} 2\,\delta_{b,(1-v)}\,\mathrm{e}^{i\left[\pi x_a x_b+\theta_{aa}x_a^2+\theta_{bb}x_b^2+\theta_{vv}x_v^2\right]} \\
&\quad= 2\sum_{v}\mathrm{e}^{i\left[\pi x_a(1-x_v)+\theta_{aa}x_a^2+\theta_{bb}(1-2x_v+x_v^2)+\theta_{vv}x_v^2\right]} \\
&\quad= 2\sum_{v}\mathrm{e}^{i\left[\pi x_a x_v+(\pi+\theta_{aa})x_a^2+(-\theta_{bb}+\theta_{vv})x_v^2\right]}\,. \qquad (4.46b)
\end{aligned}$$

This yields a quadratic form expansion with two fewer indices, and where v is still an index of degree 1. We may repeat this substitution as long as v remains adjacent to a vertex c' of degree 2, which is itself adjacent to a vertex of degree 2 and such that $\theta_{c'c'} \in \pi\mathbb{Z}$. Any such chain will of vertices in the quadratic form will then be reduced to a vertex v which is adjacent to some qubit a which either has $\theta_{aa} \notin \pi\mathbb{Z}$, or which is involved in more than two cross-terms of the original quadratic form.

Performing the substitutions above can only help to obtain a tuple (G', I, O, T) which can be analyzed using eflows. By removing additional indices, we reduce the complexity of the quadratic form expansion, and also (in the corresponding instance of **MPI**) reduce the number of qubits whose particular measurement angles may affect whether the transformation performed by the positive branch is an isometry. For a given vertex v of degree 1 in the geometry induced by the quadratic form expansion, provided both the quadratic form and also a representation of the geometry in terms of adjacency lists, determining whether one of the substitutions of (4.46) can be applied can be determined in constant time. Simplifying the quadratic form expansion provided as input as much as possible, and updating the coefficients of the square terms, is therefore a relatively inexpensive procedure. We may assume for the remainder of this section that the geometry (G, I, O) induced by the quadratic form provided as input has unit weight, and that for any vertex $v \in V(G) \setminus (I \cup O)$ with degree 1 that is adjacent to a vertex $a \in O^{\mathsf{c}}$, either the degree of a is at least 3 or $\theta_{aa} \notin \pi\mathbb{Z}$.

Provided a quadratic form expansion whose geometry (G, I, O) satisfies the above constraints, define $\mathbf{a} = (\alpha_v)_{v\in O^{\mathsf{c}}}$ given by $\alpha_v = -\theta_{vv}$. Consider the set S of vertices v of degree 1: for such vertices v and for the unique neighbor $a \sim v$, as we noted in the analysis of (4.41), the postselection of the states $|+_{\alpha_a}\rangle$ and $|+_{\alpha_v}\rangle$ is equivalent to post-selecting some state $|+\mathsf{yz}_{\alpha'}\rangle$ if and only if $\alpha_a \in \pi\mathbb{Z}$. Let $S' \subseteq S$ be the subset of those degree 1

vertices whose unique neighbors a have $\theta_{aa} \in \pi\mathbb{Z}$: then, using the relations

$$\begin{aligned} \langle +_{\alpha_v}|_v \langle +|_a \wedge Z_{av} \, |+\rangle_v &\propto \langle 0|_v \langle +|_a \, |+\rangle_v \; + \; \mathrm{e}^{-i\alpha_v} \langle 1|_v \langle -|_a \, |+\rangle_v \\ &= \frac{1}{\sqrt{2}} \langle +\mathsf{X}|_a \; + \; \frac{\mathrm{e}^{-i\alpha_v}}{\sqrt{2}} \langle -\mathsf{X}| \\ &= \left\langle +\mathsf{YZ}_{(-\alpha_v)} \right|_a \, , \end{aligned} \tag{4.47a}$$

$$\begin{aligned} \langle +_{\alpha_v}|_v \langle -|_a \wedge Z_{av} \, |+\rangle_v &\propto \langle 0|_v \langle -|_a \, |+\rangle_v \; + \; \mathrm{e}^{-i\alpha_v} \langle 1|_v \langle +|_a \, |+\rangle_v \\ &\propto \frac{1}{\sqrt{2}} \langle +\mathsf{X}|_a \; + \; \frac{\mathrm{e}^{i\alpha_v}}{\sqrt{2}} \langle -\mathsf{X}| \\ &= \langle +\mathsf{YZ}_{\alpha_v}|_a \, , \end{aligned} \tag{4.47b}$$

we may infer that the instance of the **MPI** represented by the tuple $(G, I, O, \mathbf{a})$ is equivalent to an instance $(G', I, O, T, \mathbf{a}')$, defined as follows:

(*i*) $G' = G \setminus S'$;

(*ii*) T consists of the neighbors a of the elements of $v \in S'$ in the graph G, which are to be measured in the YZ plane; and

(*iii*) $\mathbf{a}' = (\alpha'_v)_{v \in V(G') \setminus O}$, where $\alpha'_a = -\mathrm{e}^{i\alpha_a}\alpha_v$ for vertices $a \in T$ which are adjacent in G to some vertex $v \in S'$, and $\alpha'_w = \alpha_w$ otherwise.

(Again, this is an instance of an *extension* of **MPI**, in which measurements in the YZ plane are permitted.) We may then test whether the triple (G', I, O, T) has an eflow using Algorithm 3-10.

As in the case of gflows described on page 172, an eflow leads immediately to a solution to (the original version of) **MPI** for the unit weight geometries (G, I, O) which we have assumed we are given as input. Algorithm 3-10 represents the partial order $\preccurlyeq$ of an eflow $(f, \preccurlyeq)$ on (G', I, O, T) in terms of a layer function $\ell : V(G') \longrightarrow \mathbb{N}$ such that $v \preccurlyeq w \iff [\, v = w \text{ or } \ell(v) > \ell(w) \,]$. We may obtain a linear order $\leqslant$ extending $\preccurlyeq$ by constructing level sets $\mathcal{V}^{\prec}_j$ for each $j \in \mathbb{N}$, inserting each vertex $v \in V(G')$ into the set $\mathcal{V}^{\prec}_{\ell(v)}$. By the remarks on page 172, this allows us to obtain a linear ordering of $V(G)$ extending $\preccurlyeq$. For qubits $v \in V(G') \setminus T$, we may select for the result $\mathsf{s}[v] = 0$ using the pattern

$$\mathfrak{P}_{\mathfrak{s}_v} \;=\; \mathsf{X}^{\mathsf{s}[v]}_{f(v)} \left[\prod_{\substack{w \sim f(v) \\ w \neq v}} \mathsf{Z}^{\mathsf{s}[v]}_w \right] \mathsf{M}^{\alpha_v}_v \; ; \tag{4.48a}$$

for qubits $a \in T$, using the approach described in the analysis of (2.53), we may select for the result $\mathsf{s}[a] = 0$ of a YZ-plane measurement in the $\left|\pm\mathsf{yz}_{\alpha'_a}\right\rangle$ basis in the extended instance of **GMPI**, by performing the pattern

$$\mathfrak{P}_{\mathfrak{s}_a} \;=\; \left[\prod_{\substack{u \sim a \\ u \neq v}} \mathsf{Z}^{\mathsf{s}[v]}_u \right] \mathsf{M}^{\alpha_v;\mathsf{s}[a]}_v \, \mathsf{M}^{\alpha_a}_a \; , \tag{4.48b}$$

where $v \in V(G) \setminus V(G')$ is the corresponding vertex of degree 1 adjacent to a. Then, the following CPTP map performs a unitary embedding for any choice of angles $\mathbf{a} = (\alpha_v)_{v \in O^c}$ such that $\alpha_a \in \pi\mathbb{Z}$ for $a \in T$:

$$\left(\prod_{v \in V(G') \setminus O}^{\geqslant} \mathfrak{P}_{\mathfrak{s}_v}\right)\left(\prod_{uv \in E(G)} \mathsf{E}_{u,v}\right)\left(\prod_{v \in I^c} \mathsf{N}_v^{\mathsf{X}}\right), \tag{4.49}$$

where the left-most product is ordered left-to-right according to the order $\geqslant$. By the analysis of Section 4.1.4, if $(G, I, O, \mathbf{a})$ is an instance of the **MPI** derived from a quadratic form expansion for an operator U, the unitary operation which is performed is U.

Run-time analysis. Let $n = |V(G)|$, $m = |E(G)|$, and $k = |O| \geqslant |I|$. Determining whether a geometry has an eflow can be done by Algorithm 3-10 in time $O(kn+m)$ by the analysis on page 151. Having an eflow $(f, \preccurlyeq)$, the measurement procedure in (4.49) is well-defined and requires $O(m)$ time to construct, dominated by the product over $v \in V(G') \setminus O$ of the patterns $\mathfrak{P}_{\mathfrak{s}_v}$, each of which have complexity at most $O(\deg_G(v))$. Bringing this measurement procedure into a standard form by the standardization procedure of Section 2.2.5 (also requires time $O(m)$, by a similar analysis as applies for patterns constructed by the DKP construction. The resulting measurement pattern contains $n-k$ measurement operations and up to k correction operations, each of which involves $O(n)$ measurement dependencies. Finally, each substitution in (4.26) and (4.46) requires $O(1)$ time to perform provided a graph-based representation of the geometry underlying the original quadratic form expansion; the former expansion can be applied at most m times (once for each edge), and the latter at most $m + n$ times (one for each vertex in the geometry, after the addition of vertices due to fractional edges) after spending $O(n)$ time finding all vertices of degree 1. We may therefore conclude:

Theorem 4-4. *For a unitary embedding U given as a quadratic form expansion such that $|V(G)| = n$, $|E(G)| = m$, and $|O| = k$, there is an $O(kn+m)$ algorithm which determines whether or not the quadratic form expansion cannot be transformed into a set of default measurement angles $\mathbf{a}'$ and a tuple (G', I, O, T) which has an eflow. If so, the algorithm also constructs a pattern in the* OneWay($\mathbb{R}$) *model consisting of $O(kn + m)$ operations which performs the transformation U.*

As for the analysis in terms of gflows on page 173, we may strengthen this result to obtain an algorithm producing one-way patterns in a model with a finite gate set such as $\mathbb{A} = \frac{\pi}{4}\mathbb{Z}$: given that $m \geqslant n - k$ for a geometry with an eflow (from which we may infer that there are $O(m)$ measurement operations in the exact measurement pattern we must be approximated), using the same techniques as in the case for gflows, we then obtain the following corollary:

Corollary 4-4a. *For a unitary embedding U given as a quadratic form expansion such that $|V(G)| = n$, $|E(G)| = m$, and $|O| = k$, for each $\varepsilon > 0$ and $\delta > 0$ there exists an algorithm running in time $O(kn + m\log(m/\varepsilon)^{3+\delta})$ algorithm which determines whether or not the quadratic form expansion cannot be transformed into a set of default measurement angles $\mathbf{a}'$ and a tuple (G', I, O, T) which has an eflow. If so,*

the algorithm also constructs a pattern in the OneWay($\frac{\pi}{4}\mathbb{Z}$) *model consisting of* $O(kn+m\log(m/\varepsilon)^{3+\delta})$ *operations which approximates the CPTP map* $\rho \longmapsto U\rho U^\dagger$ *to precision* ε.

4.2.3 Synthesizing measurement patterns for the Clifford group

If a quadratic form expansion has a geometry whose edges all have unit weight, and its' other coefficients are multiples of $\pi/2$, then it corresponds to the positive branch of a one-way measurement pattern which measures only X or Y observables. A measurement pattern of this sort, if it performs a unitary operation, performs a Clifford group operation, as we may show (using the stabilizer formalism) that it transforms the Pauli group to itself.

An algorithm of Aaronson and Gottesman [2] can produce a Clifford circuit of size $O(n^2/\log(n))$ in classical deterministic time $O(n^3/\log(n))$ for a Clifford group operation U acting on n qubits, from a description of how U transforms Pauli operators by conjugation. By converting the circuit into a measurement-based algorithm, and performing the graph transformations of [74] to remove auxiliary qubits, we may obtain a pattern of at most $3n$ qubits[6] in time $O(n^4/\log(n))$. Building on the results of [48], we show how to classically compute such a minimal pattern in time $O(n^3/\log(n))$ by solving **MPI** for a quadratic form expansion for U.

Obtaining a quadratic form expansion

We now outline the relevant results of [48]. Define the following notation for bit-flip and phase-flip operators on a qubit t out of a collection $\{1,\ldots,n\}$:

$$P_t \;=\; X_t\,, \qquad\qquad P_{n+t} \;=\; Z_t\,. \tag{4.50}$$

Let $\mathrm{diag}(M) \in \mathbb{Z}_2^m$ represent the vector of the diagonal elements of any square boolean matrix M; and let $\mathbf{d}(M) = \mathrm{diag}\left(M^\top \left[\begin{smallmatrix} 0 & \mathbb{1}_n \\ 0 & 0 \end{smallmatrix}\right] M\right) \in \mathbb{Z}_2^{2n}$ for a $2n \times 2n$ matrix M over $\mathbb{Z}_2$. Then, we may represent an n qubit Clifford operation U by a $2n \times 2n$ boolean matrix C and a vector $\mathbf{h} \in \{0,1\}^{2n}$, whose coefficients are jointly given by

$$UP_tU^\dagger \;=\; i^{d_t(C)}\big(-1\big)^{h_t} \bigotimes_{j=1}^{n} \left[Z_j^{C_{(n+j)t}} X_j^{C_{jt}}\right] \tag{4.51}$$

for each $1 \leqslant t \leqslant 2n$. (Note that the factor of $i^{d_t(C)}$ is only necessary to ensure that the image of P_t is Hermitian, and does not serve as a constraint on the value of C.) We will call an ordered pair $(C, \mathbf{h})$ a *Leuven tableau*[7] for a Clifford group element U if it satisfies (4.51).

[6] In [25], Clifford operations on n qubits are described as having minimal patterns for are described as requiring at most $2n$ qubits; however, this only holds up to local Clifford operations on the output qubits. These local Clifford operations may be used to adapt subsequent measurements, if the output state is to be operated on by more measurement based procedures, or simply measured to obtain a classical result; however, in the context of a decomposition of unitary transformations in *e.g.* the OneWay($\frac{\pi}{2}\mathbb{Z}$) model, these local Clifford operations should be explicitly described.

[7] This terminology is motivated by the similarity between the block matrix $\left[\, C^\top \,\middle|\, \mathbf{h} \,\right]$ and *destabilizer tableaux*, as defined in [2].

Provided a Leuven tableau $(C, \mathbf{h})$ for a Clifford group operation U, [48] provides a matrix formula for U which we may obtain for U. We outline the derivation of this formula as follows. Decompose C as a block matrix $C = \left[\begin{smallmatrix} E & F \\ G & H \end{smallmatrix}\right]$ with $n \times n$ blocks, and then find invertible matrices $\tilde{R}_1, \tilde{R}_2$ over $\mathbb{Z}_2$ such that $\tilde{R}_1{}^{-1} G \tilde{R}_2 = \left[\begin{smallmatrix} 0 & 0 \\ 0 & \mathbb{1}_r \end{smallmatrix}\right]$ for some $r < n$ (using *e.g.* the decomposition algorithm of [108] to obtain $\tilde{R}_1$ and $\tilde{R}_2$ in terms of $O(n^2/\log(n))$ elementary row operations). Then, define the matrices

$$\begin{bmatrix} \tilde{E}_{11} & \tilde{E}_{12} \\ \tilde{E}_{21} & \tilde{E}_{22} \end{bmatrix} = \tilde{R}_1^\top E \tilde{R}_2 \,, \qquad R_1 = \tilde{R}_1 \,, \qquad R_2 = \begin{bmatrix} \tilde{E}_{11}{}^{-1} & 0 \\ 0 & \mathbb{1}_r \end{bmatrix}^\top \tilde{R}_2^\top \,, \tag{4.52}$$

where $\tilde{E}_{11}$ is taken to be a block of size $(n-r) \times (n-r)$. We may then obtain the block matrices

$$\begin{bmatrix} \mathbb{1}_{n-r} & E_{12} & F_{11} & F_{12} \\ E_{21} & E_{22} & F_{21} & F_{22} \\ 0 & 0 & H_{11} & H_{12} \\ 0 & \mathbb{1}_r & H_{21} & H_{22} \end{bmatrix} = \begin{bmatrix} R_1^\top & 0 \\ 0 & R_1{}^{-1} \end{bmatrix} C \begin{bmatrix} R_2^\top & 0 \\ 0 & R_2{}^{-1} \end{bmatrix} , \tag{4.53}$$

and use these to construct the $n \times n$ boolean matrices

$$M_{br} = \begin{bmatrix} F_{11} + E_{12} H_{21} & E_{12} \\ E_{12}^\top & E_{22} \end{bmatrix} , \qquad M_{bc} = \begin{bmatrix} 0 & H_{21}^\top \\ H_{21} & H_{22} \end{bmatrix} . \tag{4.54}$$

Next, define

$$\begin{aligned} \mathbf{d}_{br} &= \operatorname{diag}(M_{br}) \,, & \mathbf{d}_{bc} &= \operatorname{diag}(M_{bc}) \,, \\ L_{br} &= \operatorname{Lower}\big(M_{br} + \mathbf{d}_{br} \mathbf{d}_{br}^\top\big) \,, & L_{bc} &= \operatorname{Lower}\big(M_{bc} + \mathbf{d}_{bc} \mathbf{d}_{bc}^\top\big) \,, \end{aligned} \tag{4.55}$$

where $\operatorname{Lower}(M)$ is the strictly lower-triangular part of a square matrix M (with all other coefficients set to 0). Finally, define $\Pi_r = \left[\begin{smallmatrix} 0 & 0 \\ 0 & \mathbb{1}_r \end{smallmatrix}\right]$ and $\Pi_r^\perp = \mathbb{1}_n - \Pi_r$ for the sake of brevity, and let[8]

$$\mathbf{t} = \begin{bmatrix} \mathbb{1}_n & 0 \end{bmatrix} \mathbf{h} + \operatorname{diag}\Big(\big[R_2{}^{-1} \Pi_r\big] L_{br} \big[R_2{}^{-1} \Pi_r\big]^\top \Big) \,, \tag{4.56a}$$

$$\begin{aligned} \mathbf{h}_{bc} = \begin{bmatrix} 0 & R_2^{-\top} \end{bmatrix} \mathbf{h} + R_2^{-\top} \operatorname{diag}\Big(R_2^\top \Big[& L_{bc} + \Pi_r M_{bc} \\ & + \Big(\Pi_r^\perp + \Pi_r M_{bc}\Big) L_{br} \Big(\Pi_r^\perp + M_{bc} \Pi_r\Big) \Big] R_2 \Big) \,. \end{aligned} \tag{4.56b}$$

Then Theorem 6 of [48] states that the unitary operation U for the Clifford operation characterized by $(C, \mathbf{h})$ is given by the matrix formula

$$U = \frac{1}{\sqrt{2^r}} \sum_{\substack{\mathbf{x}_b \in \{0,1\}^{n-r} \\ \mathbf{x}_c, \mathbf{x}_r \in \{0,1\}^r}} \Big[(-1)^{\left(\mathbf{x}_{br}^\top L_{br} \mathbf{x}_{br} + \mathbf{x}_r^\top \mathbf{x}_c + \mathbf{x}_{bc}^\top L_{bc} \mathbf{x}_{bc} + \mathbf{h}_{bc}^\top \mathbf{x}_{bc}\right)} \times (-i)^{\left(\mathbf{d}_{br}^\top \mathbf{x}_{br} + \mathbf{d}_{bc}^\top \mathbf{x}_{bc}\right)} \, \big| R_1 \mathbf{x}_{br} \big\rangle \big\langle R_2{}^{-1} \mathbf{x}_{bc} + \mathbf{t} \big| \Big] , \tag{4.57}$$

where $\mathbf{x}_{br} = \left[\begin{smallmatrix} \mathbf{x}_b \\ \mathbf{x}_r \end{smallmatrix}\right]$ and $\mathbf{x}_{bc} = \left[\begin{smallmatrix} \mathbf{x}_b \\ \mathbf{x}_c \end{smallmatrix}\right]$ are n bit boolean vectors.

The formula in (4.57) shows strong similarities to a quadratic form expansion. In particular, consider disjoint sets of indices V_b, V_r, and V_c, with $|V_b| = n - r$ and $|V_r| =$

[8] These formulae for $\mathbf{t}$ and $\mathbf{h}_{bc}$ may be obtained by repeated application of Theorem 2 of [48] to the data specified by Theorem 6.

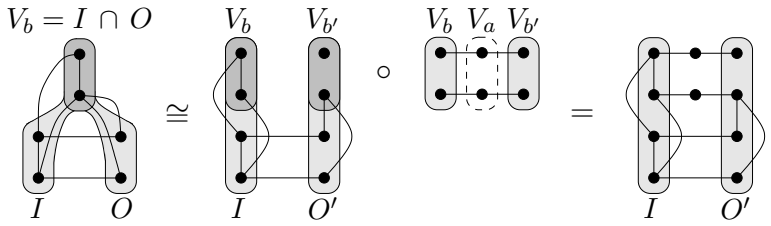


FIGURE 4-4. Illustration of an equivalence of two quadratic form expansions, via their geometries. On the left is a geometry whose inputs and output intersect; on the right is a geometry from an equivalent quadratic form expansion, constructed so that the input and output indices are disjoint.

$|V_c| = r$. Let $V = V_b \cup V_c \cup V_r$, $I = V_b \cup V_c$, and $O = V_b \cup V_r$, and define the following notation for $\mathbf{x} \in \{0,1\}^V$:

$$\mathbf{x}_I \;=\; \begin{bmatrix}\mathbf{x}_b\\ \mathbf{x}_c\end{bmatrix} \;=\; \begin{bmatrix}\mathbf{x}_{V_b}\\ \mathbf{x}_{V_c}\end{bmatrix} \in \{0,1\}^I \;, \qquad \mathbf{x}_O \;=\; \begin{bmatrix}\mathbf{x}_b\\ \mathbf{x}_r\end{bmatrix} \;=\; \begin{bmatrix}\mathbf{x}_{V_b}\\ \mathbf{x}_{V_r}\end{bmatrix} \in \{0,1\}^O \;, \tag{4.58}$$

$$\begin{aligned} Q(\mathbf{x}) \;=\; \pi\Big(\mathbf{x}_O^\top L_{br}\mathbf{x}_O \,+\, \mathbf{x}_O^\top \Pi_r \mathbf{x}_I \,+\, \mathbf{x}_I^\top L_{bc}\mathbf{x}_I \,+\, \mathbf{x}_I^\top \mathbf{h}_{bc}\mathbf{h}_{bc}^\top \mathbf{x}_I\Big) \\ -\; \tfrac{\pi}{2}\Big(\mathbf{x}_O^\top \mathbf{d}_{br}\mathbf{d}_{br}^\top \mathbf{x}_O \,+\, \mathbf{x}_I^\top \mathbf{d}_{bc}\mathbf{d}_{bc}^\top \mathbf{x}_I\Big) \;. \end{aligned} \tag{4.59}$$

Then, (4.57) is equivalent to

$$U \;=\; \tfrac{1}{\sqrt{2^r}} \sum_{\mathbf{x}\in\{0,1\}^V} \mathrm{e}^{iQ(\mathbf{x})} \,\big|R_1 \mathbf{x}_O\big\rangle\big\langle R_2{}^{-1}\mathbf{x}_I + \mathbf{t}\big| \;, \tag{4.60}$$

which is essentially a quadratic form expansion sandwiched between two networks of controlled-not and X gates. To obtain a simple quadratic form expansion, we would like to perform a change of variables on $\mathbf{x}_I$ and $\mathbf{x}_O$; but this cannot be done as I and O intersect at V_b, and the changes of variables do not necessarily respect the partitioning of I and O with respect to this intersection. However, we may add auxiliary variables in order to produce an expansion with disjoint input and output indices. Recall the expansion of the identity of (4.25),

$$\mathbb{1}_2 \;=\; \sum_{\mathbf{x}\in\{0,1\}^2} \delta_{x_1,x_2}\,|x_2\rangle\langle x_1| \;=\; \tfrac{1}{2}\sum_{\mathbf{x}\in\{0,1\}^3} (-1)^{x_1x_3+x_2x_3}\,|x_2\rangle\langle x_1| \;, \tag{4.61}$$

where $\delta_{x,y}$ is the Kronecker delta. Let V_a and $V_{b'}$ be disjoint copies of V_b, and set $V' = V \cup V_a \cup V_{b'}$ and $O' = V_{b'} \cup V_r$. Writing $\mathbf{x}_a$ and $\mathbf{x}_{b'}$ for the restriction of $\mathbf{x} \in \{0,1\}^{V'}$ to V_a and $V_{b'}$, we then define

$$\mathbf{x}_I \;=\; \begin{bmatrix}\mathbf{x}_b\\ \mathbf{x}_c\end{bmatrix} \in \{0,1\}^I \;, \qquad \mathbf{x}_{O'} \;=\; \begin{bmatrix}\mathbf{x}_{b'}\\ \mathbf{x}_r\end{bmatrix} \in \{0,1\}^{O'} \;, \tag{4.62}$$

$$\begin{aligned} Q'(\mathbf{x}_I\,,\,\mathbf{x}_a\,,\,\mathbf{x}_{O'}) \;=\; \pi\Big(\mathbf{x}_{O'}^\top L_{br}\mathbf{x}_{O'} \,+\, \mathbf{x}_{O'}^\top \Pi_r \mathbf{x}_I \,+\, \mathbf{x}_I^\top L_{bc}\mathbf{x}_I \,+\, \mathbf{h}_{bc}^\top \mathbf{x}_I\Big) \\ +\; \pi\mathbf{x}_I^\top \begin{bmatrix}\mathbb{1}_{n-r}\\ 0\end{bmatrix}\mathbf{x}_a \,+\, \pi\mathbf{x}_{O'}^\top \begin{bmatrix}\mathbb{1}_{n-r}\\ 0\end{bmatrix}\mathbf{x}_a \,-\, \tfrac{\pi}{2}\Big(\mathbf{d}_{br}^\top \mathbf{x}_{O'} \,+\, \mathbf{d}_{bc}^\top \mathbf{x}_I\Big) \;. \end{aligned} \tag{4.63}$$

Note that the difference between the expressions for Q' and Q is essentially that all instances of $\mathbf{x}_O$ have been replaced with $\mathbf{x}_{O'}$ (which is independent from $\mathbf{x}_I$), and the presence of the terms involving $\mathbf{x}_a$. We therefore have

$$\begin{aligned}\sum_{\mathbf{x}\in\{0,1\}^V} & \mathrm{e}^{iQ(\mathbf{x})} \left|R_1\mathbf{x}_O\right\rangle\!\left\langle {R_2}^{-1}\mathbf{x}_I + \mathbf{t}\right| \\ &= \sum_{\mathbf{x}_I,\,\mathbf{x}_{O'}} \delta_{\mathbf{x}_b,\mathbf{x}_{b'}}\, \mathrm{e}^{iQ'(\mathbf{x}_I,\,\mathbf{0},\,\mathbf{x}_{O'})} \left|R_1\mathbf{x}_{O'}\right\rangle\!\left\langle {R_2}^{-1}\mathbf{x}_I + \mathbf{t}\right| \\ &= \frac{1}{2^{n-r}} \sum_{\mathbf{x}\in\{0,1\}^{V'}} \mathrm{e}^{iQ'(\mathbf{x}_I,\,\mathbf{x}_a,\,\mathbf{x}_{O'})} \left|R_1\mathbf{x}_{O'}\right\rangle\!\left\langle {R_2}^{-1}\mathbf{x}_I + \mathbf{t}\right| . \end{aligned} \tag{4.64}$$

This transformation is illustrated in Figure 4-4 as a transformation of the geometries underlying Q and Q'. Substituting the final expression of (4.64) into (4.60) and performing the appropriate change of variables, we have

$$U \;=\; \frac{\sqrt{2^r}}{2^n} \sum_{\mathbf{x}\in\{0,1\}^{V'}} \mathrm{e}^{iQ'(R_2(\mathbf{x}_I+\mathbf{t}),\,\mathbf{x}_a,\,R_1^{-1}\mathbf{x}_{O'})} \left|\mathbf{x}_{O'}\right\rangle\!\left\langle\mathbf{x}_I\right| \;: \tag{4.65}$$

expanding the expression in the exponent, it is easy to see that this is a quadratic form expansion for U.

Interpolating the measurement pattern

Note that the quadratic form of the expansion in (4.65) has only angles θ_{uv} which are multiples of $\pi/2$, with $\theta_{uv} \in \{0, \pi\}$ for $u \neq v$. This then represents the positive branch of a one-way measurement pattern using only X- and Y-axis measurements, and having only $n-r$ auxiliary vertices. From the analysis of the stabilizer formalism in Section 1.5.3, the fact that the result of the post-selection procedure is in particular a *unitary* transformation implies that each individual post-selection induces a unitary transformation; then, we can then obtain a correction procedure to obtain a complete measurement pattern for U via the stabilizer formalism.

First, we may produce a complete measurement procedure in which we perform correction operations after each measurement. After applying the preparation and entangling operations $\mathsf{E}_G\,\mathsf{N}_{I^\mathsf{c}}$ before the measurements, the state-space is stabilized by the group $\mathcal{S}_{G,I} = \langle K_v\rangle_{v\in I^\mathsf{c}}$. Because post-selecting the measurement results $\mathsf{s}[v]$ for $v \in O^\mathsf{c}$ performs an isometry of the encoded space, there must be a list of generators $(S_v)_{v\in O^\mathsf{c}}$ for $\mathcal{S}_{G,I}$ such that, for some ordering $v_1, v_2, \ldots$ of the qubits of $V(G)$,

(*i*) S_{v_j} anticommutes with the measurement observable for the measurement on v_j, where we define

$$\mathcal{O}_{v_j} \;=\; \left\{ \begin{array}{ll} X\,, & \text{if } \theta_{v_j v_j} \in \pi\mathbb{Z} \\ Y\,, & \text{if } \theta_{v_j v_j} \in \pi\mathbb{Z} + \pi/2 \end{array} \right\} ; \tag{4.66}$$

(*ii*) S_{v_j} commutes with the measurement observable $\mathcal{O}_{v_h}$ for any $h > j$.

We may produce such a list of generators for any given enumeration $(v_j)_{j=1}^n$ of the elements of $V(G)$ in polynomial time, essentially by row-reduction.

We arrange the generators K_v for $v \in O^{\mathsf{c}}$ in rows in arbitrary order, with the columns denoting the qubits on which each operator acts. We then decompose each operator K_v into a product of operators $\mathcal{O}_w$ for $w \in O^{\mathsf{c}}$ and X_w for $w \in O$ (which we call the *commuting* part of K_v) and a product of Z_w operators for $w \in V(G)$ (which we call the *anticommuting* part of K_v). It is obvious that the anticommuting part of each operator K_v will anticommute with any $\mathcal{O}_w$ for a qubit $w \in O^{\mathsf{c}}$ on which it acts, and similarly the commuting part of K_w commutes with every observable $\mathcal{O}_w$. We may use these components to form a stabilizer tableau, as illustrated in Figure 4-5, with the "anticommuting" Z part of the operator in the leading columns, and the "commuting" part in the trailing columns. Row transformations of this matrix then produce different generators of $\mathcal{S}_{G,I}$, where the commutation relations with the observables $\mathcal{O}_w$ for $w \in O^{\mathsf{c}}$ can be easily read from the Z component part of the table. By putting this table into reduced row-echelon form, we may then obtain a collection of generators $\{S_v\}_{v \in O^{\mathsf{c}}}$ for $\mathcal{S}_{G,I}$ which not only satisfy the criteria above, but such that S_v anticommutes with $\mathcal{O}_w$ only for $v = w$. As each correction operation only acts on the qubit $v \in O^{\mathsf{c}}$ whose measurement it depends upon (and which we are considering to be measured destructively), these generators S_v also represent the final correction operations for the measurement pattern: at the end of the measurement sequence, it suffices to perform the correction $S_v^{\mathbf{s}[v]}$ for each $v \in O^{\mathsf{c}}$ to obtain a unitary transformation.

Run-time analysis. For a Clifford operation $U \in \mathsf{U}(2^n)$ given as a Leuven tableau $(C, \mathbf{h})$, the quadratic form of (4.60) can be found in time $O(n^3/\log(n))$, which is dominated by the time required to compute R_1 and R_2, using the techniques of [108].[9] From this, the quadratic form expansion of (4.65) can be obtained by a constant number of matrix multiplications, and by inverting R_1 and R_2 (which may be computed alongside R_1 and R_2 themselves). The boolean row-reductions required to compute the correcting generators S_v for $v \in O^{\mathsf{c}}$ also requires time $O(n^3/\log(n))$ by using a modified version of the algorithm of [108] to do simple row-reduction.[10] The resulting measurement pattern contains at most $3n$ qubits, and therefore requires $O(n^2)$ entangling operations, $O(n)$ measurement operations, and $O(n)$ correction operations each of complexity at most $O(n)$, yielding a total size of $O(n^2)$. Therefore, we have:

Theorem 4-5. *For an n-qubit Clifford group operation U given in the form of a Leuven tableau, there is an $O(n^3/\log(n))$ algorithm which produces a pattern of size $O(n^2)$ in the* OneWay$(\frac{\pi}{2})$ *model which performs U.*

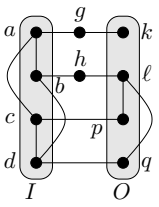
Minimality of patterns synthesized from quadratic form expansions

Hein, Eisert, and Briegel present techniques in [74] for the elimination of Pauli observable measurements in graph states by (a) transformations of the underlying geometry, and

[9] The algorithm of [108] obtains an LU decomposition for matrix over $\mathbb{Z}_2$: while the result is described as applying to invertible matrices, it in fact can be applied to arbitrary boolean matrices.

[10] It is sufficient to modify the algorithm of [108] to eliminate ones above the diagonal as well as below, and to forego the column-reduction phase (consisting of the transposition and all operations afterwards).

(a)



(b)

Table of stabilizer generators K_v

qubit ordering	g	d	h	b	a	c	k	ℓ	p	q
list of $\mathcal{O}_v$	Y	X	Y	X	Y	Y	n/a	n/a	n/a	n/a
$K_g =$	X	·	·	·	Z	·	Z	·	·	·
$K_h =$	·	·	X	Z	·	·	·	Z	·	·
$K_k =$	Z	·	·	·	·	·	X	·	·	·
$K_\ell =$	·	·	Z	·	·	·	·	X	Z	Z
$K_p =$	·	·	·	·	·	Z	·	Z	X	·
$K_q =$	·	Z	·	·	·	·	·	Z	·	X

(c)

Stabilizer Tableau — operator components indicated by headings in the second row

g	d	h	b	a	c	k	ℓ	p	q	g	d	h	b	a	c	k	ℓ	p	q
Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Y	X	Y	X	Y	Y	X	X	X	X
Z	·	·	·	Z	·	Z	·	·	·	Y	·	·	·	·	·	·	·	·	·
·	·	Z	Z	·	·	·	Z	·	·	·	·	Y	·	·	·	·	·	·	·
·	·	·	·	Z	·	·	·	·	·	·	·	·	·	·	·	X	·	·	·
·	·	Z	·	·	·	·	·	Z	Z	·	·	·	·	·	·	·	X	·	·
·	·	·	·	·	Z	·	Z	·	·	·	·	·	·	·	·	·	·	X	·
·	Z	·	·	·	·	·	Z	·	·	·	·	·	·	·	·	·	·	·	X

(d)

Reduced Stabilizer Tableau — obtained by row-reduction

g	d	h	b	a	c	k	ℓ	p	q	g	d	h	b	a	c	k	ℓ	p	q
Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Y	X	Y	X	Y	Y	X	X	X	X
Z	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	X	·	·	·
·	Z	·	·	·	·	·	Z	·	·	·	·	·	·	·	·	·	·	·	X
·	·	Z	·	·	·	·	·	Z	Z	·	·	·	·	·	·	·	X	·	·
·	·	·	Z	·	·	·	Z	Z	Z	·	·	Y	·	·	·	·	X	·	·
·	·	·	·	Z	·	Z	·	·	·	Y	·	·	·	·	·	X	·	·	·
·	·	·	·	·	Z	·	Z	·	·	·	·	·	·	·	·	·	·	X	·

(e)

Table of correcting generators S_v

qubit ordering	g	d	h	b	a	c	k	ℓ	p	q	
list of $\mathcal{O}_v$	Y	X	Y	X	Y	Y	n/a	n/a	n/a	n/a	
$S_g =$	Z	·	·	·	·	·	X	·	·	·	$= K_k$
$S_d =$	·	Z	·	·	·	·	·	Z	·	X	$= K_q$
$S_h =$	·	·	Z	·	·	·	·	X	Z	Z	$= K_\ell$
$S_b =$	·	·	Y	Z	·	·	·	Y	Z	Z	$= K_\ell K_h$
$S_a =$	Y	·	·	·	Z	·	X	·	·	·	$= K_g K_k$
$S_c =$	·	·	·	·	·	Z	·	Z	X	·	$= K_p$

FIGURE 4-5. Illustration of an algorithm for obtaining the correction operations for a postselection-based procedure for a unitary U using measurement observables X and Y. (a) The input geometry. (b) The stabilizer generators K_v arising from the geometry, as well as an arbitrary measurement order of the qubits of O^{c} and a specification of the measurement observables $\mathcal{O}_v$; "dots" represent identity operations $\mathbb{1}_2$. (c) The same generators presented in a "stabilizer tableau", in which each generator is decomposed into a Z component (which anticommutes with every measurement observable) and a non-Z component (chosen to commute with the observable $\mathcal{O}_v$ for each qubit $v \in O^{\mathsf{c}}$). Note that this tableau is essentially an $(n - |I|) \times 2n$ boolean matrix. (d) The reduced row-echelon form arising from the previous table. For each $j \geqslant 1$, the j^{th} row represents an operator which anticommutes with the j^{th} measurement observable, and commutes with every other measurement observable. (e) A more explicit representation of the preceding table.

(b) local unitary transformations of the qubits, which amount to changes in the observable to be performed for some of the measurements. Paraphrasing [74] and extending it to open graphical encodings, these transformations are as follows:

Lemma 4-6 [74, Proposition 1]**.** *Let G be a graph: throughout the following, let $\sim$ denote adjacency in G. Define a unitary embedding*

$$\mathcal{E}_{G,I} = \left(\prod_{u\sim v} \wedge Z_{uv}\right)\left(\bigotimes_{v\in I^{\mathsf{c}}} |+\rangle_v\right) \tag{4.67}$$

performing an open graphical encoding. For a vertex $u \in V(G)$, let $G*u$ represent the local complementation *of G about u, which is the graph on $V(G)$ where*

$$vw \in E(G*u) \iff \left\{ \begin{array}{ll} vw \in E(G)\,, & \text{if } v \not\sim u \text{ or } w \not\sim u \\ vw \notin E(G)\,, & \text{if } v \sim u \text{ and } w \sim u \end{array} \right\}. \tag{4.68}$$

If an X_a, Y_a, or Z_a observable measurement is performed on some qubit $a \in I^{\mathsf{c}}$ for a joint state of the form $\mathcal{E}_{G,I}|\psi\rangle$ for any $|\psi\rangle \in \mathcal{H}_I$, then the resulting pure state, conditioned on the outcome ± 1, are as follows:

(*i*) *for a Z_a observable measurement, the post-measurement state is*

$$|\pm\mathsf{z}\rangle_a \otimes U_{a,\pm\mathsf{z}}\,\mathcal{E}_{G'_{\mathsf{z}},I}\,|\psi\rangle \tag{4.69a}$$

where $G'_{\mathsf{z}} = G \smallsetminus a$, $U_{a,+\mathsf{z}} = \mathbb{1}_{V(G)}$, and $U_{a,-\mathsf{z}} = \prod_{v\sim a} Z_v$;

(*ii*) *for a Y_a observable measurement, the post-measurement state is*

$$|\pm\mathsf{y}\rangle_a \otimes U_{a,\pm\mathsf{y}}\,\mathcal{E}_{G'_{\mathsf{y}},I}\,|\psi\rangle \tag{4.69b}$$

where $G'_{\mathsf{y}} = (G*a) \smallsetminus a$, and $U_{a,\pm\mathsf{y}} = \prod_{v\sim a} R_{\mathsf{z}}(\pm\pi/2)_v$;

(*iii*) *for an X_a observable measurement, for an arbitrarily chosen $b \in I^{\mathsf{c}}$ adjacent to a, the post-measurement state is*

$$|\pm\mathsf{x}\rangle_a \otimes U_{a,\pm\mathsf{x}}\,\mathcal{E}_{G'_{\mathsf{x}},I}\,|\psi\rangle \tag{4.69c}$$

where $G'_{\mathsf{x}} = (G*a*b*a) \smallsetminus a$, and where we define

$$U_{a,+\mathsf{x}} = R_{\mathsf{y}}(-\pi/2)_b \prod_{\substack{v\sim a \\ v\not\sim b_0 \\ v\neq b_0}} Z_v\,; \qquad U_{a,-\mathsf{x}} = R_{\mathsf{y}}(\pi/2)_b \prod_{\substack{v\sim b_0 \\ v\not\sim a \\ v\neq a}} Z_v\,. \tag{4.69d}$$

It can be easily verified that [74, Proposition 1] can be extended as above, as the analysis of that article depends only on the presence of the stabilizer generators K_v for $v \in I^{\mathsf{c}}$.[11]

[11] The input subsystem I does not contribute any generators K_v for $v \in I$, precisely because the input subsystem is not in general stabilized by any non-trivial Pauli operators. Despite this, local complementations still affect adjacency relations which are entirely restricted to the input subsystem. This may be verified by noting that an open graphical encoding can be characterized by $2^{|I|}$ stabilizer groups in parallel, with each group describing a scenario where each qubit $v \in I$ is stabilized by X_v or Z_v prior to encoding. Collectively, these groups characterize any unitary transformations performed on the state, by linearity. For each such group, any qubit $v \in I$ which is initially stabilized by X_v does contribute a generator K_v to the stabilizer group in that case; the generators of such groups are transformed in the same way as in graph states.

The connection with local complementations of graphs (which is not made explicit in the presentation of Proposition 1) can be verified by application of Proposition 8 in the same article.

The results of [74] would seem to be pertinent for the measurement patterns produced from quadratic form expansions, as described by the constructions above. This suggests that we may be able to apply them to obtain measurement patterns involving fewer qubits than the ones produced by our construction. Despite this, we may show that the measurement patterns produced by the techniques of this section cannot be reduced using Lemma 4-6.

Let A denote the set of auxiliary qubits, corresponding to the indices of the vector $\mathbf{x}_a$ adjoined in (4.64). In the resulting measurement pattern, these are all to be measured with the observable X, as there is no square term in the quadratic form Q' for any of these indices. As well, they are adjacent only to the input/output vertices (corresponding to the indices of $\mathbf{x}_I$ and $\mathbf{x}_O$). These properties still hold in the final quadratic form expansions after the final change of variables, which do not act on the indices of A. To eliminate a vertex $v \in A$ using Lemma 4-6 in the open graphical encoding of the resulting one-way pattern, we must identify an output qubit $b' \in O$ adjacent to v, and transform the graph G of the underlying geometry by to the graph $G'_{\mathsf{x}} = (G * v * b' * v) \setminus a$. This would result in a geometry where b' has the former neighbors of v in G (excepting itself): in particular, it will only be adjacent to one other qubit $b \in I$. As well, a $R_{\mathsf{y}}(\pm\pi/2)$ rotation must be applied to b after the entangling procedure. This rotation is essentially unavoidable, in the following sense:

- Because $b' \in O$, it will not be measured, and so the $R_{\mathsf{y}}(\pm\pi/2)$ operation cannot be absorbed into a measurement.
- Because b' is not adjacent to any elements of $V(G'_{\mathsf{x}}) \setminus I$, the only stabilizer generator of the output space of the encoding $\mathcal{E}_{G'_{\mathsf{x}},I}$ is $K'_{b'} = X_{b'}Z_b$, which does not commute with $R_{\mathsf{y}}(\pm\pi/2)$. Then, the $R_{\mathsf{y}}(\pm\pi/2)$ is not equivalent (in the open graphical encoding) to any operator which does not act on b'.

Then, to represent the result of the transformation described by Lemma 4-6 on a qubit $v \in I^{\mathsf{c}}$ which is to be measured, we must perform a local Clifford operation outside of the Pauli group to one of the qubits of the resulting encoding. In the OneWay($\frac{\pi}{2}\mathbb{Z}$) model, this local unitary will be represented by a pattern involving at least one additional qubit; thus, the number of qubits in the total pattern are not decreased.

Given a Clifford operation U with a one-way measurement pattern synthesized from a quadratic form expansion, the techniques of [74] may be applied if U is to be composed with some other one-way pattern, or if the output subsystem is to be measured to produce a final classical result; however, as a representation of a unitary transformation, the above analysis shows that the number of qubits in the measurement pattern cannot be reduced using those techniques.

Comparison with the algorithm of Aaronson and Gottesman

As we noted on page 186, an approximately optimal Clifford circuit for a Clifford group operation (consisting of $O(n^2/\log(n))$ gates) can be found in time $O(n^3/\log(n))$. This

may be done from a Leuven tableau by transforming it into a destabilizer tableau (another representation of transformations of Pauli operators by U which differs trivially from a Leuven tableau), and then applying the algorithm of [2]. To obtain a measurement pattern from such a circuit by composing the patterns for each gate, opportunistically removing vertices using the result of Lemma 4-6 (with each removal taking $O(n^2)$ time), leads instead to an $O(n^4/\log(n))$ time algorithm. Thus, by Theorem 4-5, we may synthesize a minimal one-way measurement pattern more efficiently via quadratic form expansions than from the circuits produced by the algorithm of [2].

The upper bound in Theorem 4-5 on the complexity of the measurement pattern in $O(n^2)$, in contrast with the complexity of $O(n^2/\log(n))$ achieved by the circuits of [2]. It is an open question whether the complexity of the patterns produced by the algorithm outlined above can be bounded more strongly than $O(n^2)$. It is not obvious, however, that the number of edges in the entanglement graph or (nearly equivalently) the complexity of the final correction operations should saturate theupper bound of $O(n^2)$, or that the one-way measurement model itself imposes enough structural constraints to cause an asymptotically significant inflation in the optimal representation of a unitary transformation. I therefore conjecture that the complexity of the measurement patterns synthesized by the algorithm above can also be improved to $O(n^2/\log(n))$, as in the optimal bound for Clifford circuits.

4.3 Further Remarks

In this Chapter, we have focused largely on how we can transform or construct quadratic form expansions in such a way as to obtain an instance of **MPI**, or a related problem such as **GMPI**, which we are able to efficiently solve. Before closing, we will remark on some of the ways in which the work in this Chapter may be extended.

4.3.1 Reductions from more complex operator formulae

As we remarked on page 162, path integrals are a potential "natural" source of quadratic form expansions, for which we may wish to obtain one-way measurement patterns. In the case of path integrals, the most natural analogue of the quadratic form is often the integral of the action along a trajectory through space; however, these trajectories usually manifest as a polynomial of degree higher than 2 for non-trivial systems. Direct attempts to discretize a path integral may then lead to formulae similar to quadratic form expansions, except where the quadratic form is substituted by a higher degree polynomial. As well, natural approaches to efficiently estimating the path integral of a physical system may involve attributing different weights to the discretized paths, corresponding to different amounts of constructive/destructive interference of the amplitudes of similar paths which differ from the trajectories of minimum action. We outline here how variations on quadratic form expansions, involving higher degree polynomials and amplitudes which are non-uniform across the path variables $\mathbf{x} \in \{0,1\}^n$, may be reduced to quadratic form expansions.

Higher degree polynomials governing phases

In order to treat matrix formulae with higher degree polynomials governing the phase contributions of the terms in the sum, consider a generalization of the substitution described by (4.26) to more than two bits. Considering the construction of the pattern $\mathfrak{Z}\mathfrak{z}\cdots\mathfrak{z}^{\theta}$ described in (2.53), it is possible to show that

$$\mathrm{e}^{i\phi(x_1\oplus\cdots\oplus x_d)} \;=\; \frac{1}{2}\sum_{\mathbf{a}\in\{0,1\}^2} \mathrm{e}^{i\left[\pi(x_1a_1+\cdots+x_da_1+a_1a_2)+\phi a_2^2\right]} \tag{4.70}$$

for any $d \geqslant 1$. We may easily prove by induction that

$$x_1 \oplus \cdots \oplus x_d \;=\; \sum_{\substack{S\subseteq\{1,\ldots,d\}\\ S\neq\varnothing}} \prod_{j\in S} (-2)^{|S|-1} x_j \;; \tag{4.71}$$

equivalently, we have

$$\prod_{j=1}^{d} x_j \;=\; \left(\frac{-1}{2}\right)^{d-1} \left[-(x_1 \oplus \cdots \oplus x_d) \;+ \sum_{\substack{S\subset\{1,\ldots,d\}\\ S\neq\varnothing}} \prod_{j\in S} (-2)^{|S|-1} x_j \right] , \tag{4.72}$$

which we can apply recursively to products of the bits x_1 through x_d to obtain the formula

$$\prod_{j=1}^{d} x_j \;=\; -\left(\frac{-1}{2}\right)^{d-1} \sum_{\substack{S'\subseteq\{1,\ldots,d\}\\ S'\neq\varnothing}} \left(\bigoplus_{j\in S} x_j \right) . \tag{4.73}$$

Thus a monomial of degree d may be expanded into a sum of $2^d - 1$ parities; we may then use the substitution of (4.70) to express any factor of the form $\mathrm{e}^{i\theta x_1\cdots x_d}$ in terms of a sum of $2^{d+1} - 2$ auxiliary indices, with two such indices for each sum modulo 2 to be computed. If the degree of the terms of the action of a physical system is bounded by some constant, this then leads to a constant factor overhead in the size of a quadratic form expansion, in order to represent a similar expression whose "paths" have phases given by higher degree polynomials.

As a result of the analysis above, the degree of the action in the dynamical variables of a physical system need not be an overriding concern in the formulation of a quadratic form expansion, provided that the degree is bounded above by a small constant. A comparison between this and techniques for "decoupling" higher degree terms in path integrals in the physics literature (such as the Hubbard–Stratonivich transformation: see [8, 73] for expositions) may provide insights into a high-level physical interpretation of this substitution.

Unequal amplitude contributions across paths

Quadratic form expansions are a normalized sum of contributions of operators $|\mathbf{x}_O\rangle\langle\mathbf{x}_I|$ for $\mathbf{x} \in \{0,1\}^n$, each of which has one singular value of 1 and the rest equal to zero. That is to say, as a sum over paths, each path is given equal weight; only the phases

differ. However, it is natural to consider operator sums where terms may have different weights for different boolean strings $\mathbf{x}$. One reason may be that, in order to discretize a path integral, contributions of many continuous paths are attributed to each discrete path string $\mathbf{x}$, and these will in general result in contributions with different amplitudes. Therefore, matrix expressions in which each coefficient is a sum of complex numbers with different moduli (instead of complex units), normalized over all matrix coefficients, may be of interest.

In case the amplitudes of the paths depend uniformly on the value of a single bit x_v in the path string $\mathbf{x}$ (*i.e.* if all paths with $x_v = 0$ have the same weight, and similarly for $x_v = 1$), we may represent this bias in the amplitudes simply by means of a choice of measurement basis for v. Considering the relationship between quadratic form expansions and instances of **MPI** shown in Section 4.1.4, we may attribute the uniform modulus of the paths in that case to the fact that every measurement is performed with a state which is a uniform-weight superposition of the standard basis: the projection of the states $|0\rangle$ and $|1\rangle$ onto $|+_\alpha\rangle$ for any α has the same modulus. In order to achieve a systematic bias towards either $x_v = 0$ or $x_v = 1$, it suffices to consider a measurement of v where the preferred measurement result is similarly biased towards the corresponding standard basis state.

More generally, we may express the bias in the amplitudes of the paths in terms of a boolean polynomial on $\mathbf{x}$. To achieve the necessary biases, we may consider substitutions of the sort shown in (4.36). The non-uniformity of the sum which presented problems for an analysis in terms of **GMPI** in Section 4.2.1, here allows us to express amplitudes for paths in terms of the parity of two path variables. By using parities of boolean expressions, we may produce biases depending on the value variable $\mathbf{x}$ consisting of a monomial of arbitrary degree, using the techniques of the preceding page. By summing over such monomial variations, we may obtain arbitrary dependencies of path amplitudes on the path variables. Again, if the dependency of the amplitudes on the path variables is of bounded degree, this yields a constant factor increase in the complexity of the resulting matrix expression, and ultimately yields a sum with uniform weights across all terms.

4.3.2 Techniques for simulating postselection via error correction

The results of this chapter have revolved about rather special conditions on the structure of a quadratic form expansion, under which the simulation of postselection is essentially possible generically due to restrictions on the graphical encoding performed by the preparation and entangling phases (in the case of "*yes*" instances of different versions of the **GMPI**) or of the default measurement angles (in the case of Clifford group unitaries). In order to achieve a deeper understanding of how postselection-based procedures may be extended to measurement-based patterns, it would be useful to arrive at characterization of sequences of measurement observables on the one hand, and graphical encodings on the other, such that postselection of the $+1$ result of each measurement in sequence performs an isometry.

In [101], necessary and conditions are laid out for conditions under which a CPTP map (such as a measurement operation) performs a reversible operation on a given subspace of the possible states of a physical system. Considering the special case of single-qubit measurements on open graphical encodings, and supplemented by an explicit account of

the evolution of the state-space under such measurements, may plausibly lead to a fuller understanding of **MPI**.

4.4 Review and open Problems

In this chapter, we have considered the possibility of developing new techniques for developing quantum algorithms via the one-way measurement model by presenting techniques for transforming quadratic form expansions, and for synthesizing realizable procedures for unitary transformations by considering solvable instances of **MPI**. We have also presented quadratic form expansions themselves as interesting expressions in their own right, predicting that they are likely to recur as an important form in which unitary operations may be expressed, and suggesting in particular that they may be obtained by discretizing path integral formulations of propagators for physical systems.

The major open problem of this chapter which may be clearly addressed is whether a meaningful link between path integrals and quadratic form expansions is possible, and whether the **MPI** could be solved for typical instances which would be generated in this way. Affirmative answers for both would represent the most likely way in which the results of this chapter (or extensions of them for more general instances of the **MPI**) could see concrete application.

From a purely theoretical point of view, a subject which has not been addressed in this chapter is the question of when a quadratic form expansion represents a unitary transformation. Also of potential interest is what transformation is performed on the input subsystem by the postselection procedure described by a quadratic form expansion, when that operator is not a unitary transformation. Answers to these questions may also shed light on the one-way measurement based model, as a potentially useful model of computation.

Finally, note that the construction of the one-way measurement patterns produced for Clifford group operations in Section 4.2.3 does not hint at any obvious circuit structure. In particular, while the depth of the measurement pattern is constant, an asymptotically optimal circuit for a Clifford group operation has size $O(n^2/\log(n))$, and therefore depth similar to $n/\log(n)$. Is it possible to arrive at a semantic map for Clifford patterns in some sense similar to that which was done for the DKP and simplified RBB constructions in Chapter 3?

Chapter 5

Further research directions

This thesis has been concerned with the one-way measurement model as a means of representing unitary transformations. This has largely revolved around *flows* on one hand, as a combinatorial structure arising out of a convenient representation of unitary circuits with certain "simple" gate sets, and generalizations of flows, which arise from links between flows and the techniques of the stabilizer formalism. From this analysis, we have obtained efficient algorithms for recognizing certain one-way measurement patterns which perform unitary transformations, and translating them into unitary circuit models. These techniques then proved useful for describing how we might be able to automatically obtain realizable procedures for unitary transformations which are specified by a certain type of operator formula which has recurred in the literature, and which are similar in form to a path integral formulation of propagators for a physical system.

The most natural (if perhaps ambitious) research direction from a mathematical standpoint is to characterize those one-way measurement patterns which perform unitary transformations. This problem seems to demand an extension of the techniques of the stabilizer formalism. The most likely necessary ingredient is one which is contained in the analysis of certain measurement patterns in [113]: given an open graphical encoding stabilized by some subset of the Pauli group, we may obtain Hamiltonians which have a non-trivial kernel, in which the state of the system lies. This allows us to consider the continuous group of unitary operators which also stabilize the state, which may be derived from continuous-time evolution of these Hamiltonians. Judiciously chosen operators of this form can be used to transform measurement observables into Pauli operators whose measurement can be treated by the stabilizer formalism. Connecting these rotation operations to the conditions of [101] for reversibility of quantum operations may yield an analysis of sequences of measurements, for which a characterization may be obtained. Solving this problem will require a characterization of how stabilizer codes transform after successive information-preserving single-qubit measurements; this amounts to finding a more general class of codes than stabilizer codes, which is closed under such operations.

After the problem of characterizing measurement patterns performing unitary transformations, the next natural problem is to recognize instances of **MPI** which may be "interpolated" to satisfy such a characterization (where we extend **MPI** to admit arbitrary "default measurement observables" in place of the vector of default measurement angles). A successful analysis of this problem would represent essentially complete low-level knowledge of the one-way measurement model.

While the above problems may have interesting practical dividends, the immediate question from the point of view of application is where one may expect to obtain instances of **MPI**, in order to have something to apply these techniques to. I have tried to motivate quadratic form expansions as a recurring form of operator expression in the quantum computing literature, but this does not describe a programme for actually obtaining such expressions. The natural question is then whether or not it is possible to transform path integrals for physical systems of interest into quadratic form expansions, for which we may solve **MPI** and thereby obtain efficient algorithms for simulating those systems. At the same time, an exploration of a possible connection between quadratic form expansions and path integrals may itself yield interesting theoretical insights into the one-way measurement model, or into quantum computation in general.

References

Many of the articles below are freely avaliable on the Cornell E-print Archive: such articles are indicated by the notation [arXiv:⟨*article index*⟩] in their bibliographical entries. For such articles, an abstract and links to Postscript/PDF versions of the article can be found at the URL [www.arXiv.org/abs/⟨*article index*⟩] corresponding to the article's index.

After each bibliographical citation, the list of numbers indicate the page numbers on which the article is cited.

[1] S. Aaronson. Quantum computing, postselection, and probabilistic-polynomial time. In *Proceedings of the Royal Society of London*, volume 461 of *A*, pages 3472 – 3482, 2005. [arXiv:quant-ph/0412187]. 158

[2] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70, 2004. [arXiv:quant-ph/0406196]. xv, 48, 186, 194

[3] S. Aaronson and J. Watrous. Closed timelike curves make quantum and classical computing equivalent. Pre-print: [arXiv:0808.2669], 2008. 154

[4] D. Aharonov, A. Kitaev, and N. Nisan. Quantum circuits with mixed states. In *STOC '98: Proceedings of the thirtieth annual ACM Symposium on Theory of Computing*, pages 20–30, New York, NY, USA, 1998. ACM. [arXiv:quant-ph/9806029]. 19, 20

[5] Dorit Aharonov, Vaughan Jones, and Zeph Landau. A polynomial quantum algorithm for approximating the jones polynomial. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 427–436. ACM, 2006. [arXiv:quant-ph/0511096]. 49

[6] Andris Ambainis and Oded Regev. An elementary proof of the Quantum Adiabatic Theorem. Pre-print: [arXiv:quant-ph/0411152], 2004. 49

[7] Daniel Arovas, J. R. Schrieffer, and Frank Wilczek. Fractional statistics and the quantum hall effect. *Phys. Rev. Lett.*, 53(7):722–723, Aug 1984. 49

[8] A. Auerbach. *Interacting electrons and quantum magnetism*. Graduate texts in contemporary physics. Springer-Verlag, 1994. 195

[9] H. Barnum, C. M. Caves, J. Finkelstein, C. A. Fuchs, and R. Schack. Quantum probability from decision theory? In *Proceedings of the Royal Society of London*, volume 456 of *A*, pages 1175–1182, 2000. [arXiv:quant-ph/9907024]. 14

[10] Sean D. Barrett and Pieter Kok. Efficient high-fidelity quantum computation using matter qubits and linear optics. *Physical Review A*, 71(6):060310, 2005. 55

[11] Stephen D. Bartlett and Terry Rudolph. Simple nearest-neighbor two-body Hamiltonian system for which the ground state is a universal resource for quantum computation. *Physical Review A*, 74(4):040302, 2006. 55

[12] P. A. Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22(5):563–591, may 1980. 1

[13] P. A. Benioff. Quantum mechanical Hamiltonian models of discrete processes. *Journal of Mathematical Physics*, 22(3):495–507, 1981. 1

[14] P. A. Benioff. Quantum mechanical Hamiltonian models of discrete processes that erase their histories: applications to Turing machines. *Internationl Journal of Theoretical Physics*, 21:177–202, 1982. 1

[15] Simon C Benjamin, Daniel E Browne, Joe Fitzsimons, and John J L Morton. Brokered graph-state quantum computation. *New Journal of Physics*, 8(8):141, 2006. 55

[16] Charles H. Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K. Wootters. Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Phys. Rev. Lett.*, 70(13):1895–1899, Mar 1993. 50, 51

[17] E. Bernstein. *Quantum complexity theory*. PhD thesis, 1997. PhD thesis. 40, 41

[18] E. Bernstein and U. Vazirani. Quantum complexity theory. In *Proceedings of the 25th ACM Symposium on the Theory of Computing*, pages 11–20, 1993. xi, 1

[19] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal of Computing*, 5(26):1411–1473, 1997. 1, 40, 41

[20] D. Bohm. A suggested interpretation of the quantum theory in terms of "hidden" variables. I. *Physical Review*, 85(2):166–179, 1952. 3

[21] T. Böhme, F. Göring, and J. Harant. Menger's theorem. *Journal of Graph Theory*, 37(1):35–36, 2001. 98, 102

[22] M. Born and V. Fock. Beweis des adiabatensatzes. *Zeitschrift für Physik A: Hadrons and Nuclei*, 51(3–4):165–180, 1928. 49

[23] P. Boykin, T. Mor, M. Pulver, V. Roychowdhury, and F. Vatan. On universal and fault-tolerant quantum computing. Pre-print: [arXiv:quant-ph/9906054], 1999. 25

[24] A. Broadbent and E.Kashefi. Parallelizing quantum circuits. [arXiv:0704.1736], 2007. xiv, 92, 126

[25] D. E. Browne and H. J. Briegel. One-way quantum computation — a tutorial introduction, 2006. [arXiv:quant-ph/0603226]. xv, 71, 75, 186

[26] D. E. Browne, E. Kashefi, M. Mhalla, and S. Perdrix. Generalized flow and determinism in measurement-based quantum computation. *New Journal of Physics*, 9(8):250, 2007. [arXiv:quant-ph/0702212]. 57, 62, 146, 152

[27] Daniel E. Browne and Terry Rudolph. Resource-efficient linear optical quantum computation. *Physical Review Letters*, 95(1):010501, 2005. 54, 56

[28] G. Buntrock, U. Hertrampf, C. Meinel, and C. Damm. Structure and importance of logspace-MOD-classes. In *STACS 91: Proceedings of the 8^{th}annual Symposium on Theoretical Aspects of Computer Science*, pages 360–371, New York, NY, USA, 1991. Springer-Verlag New York, Inc. 48

[29] Andrew M. Childs. Universal computation by quantum walk. Pre-print: [arXiv:0806.1972], 2008. 49

[30] Andrew M. Childs, Debbie W. Leung, and Michael A. Nielsen. Unified derivations of measurement-based schemes for quantum computation. *Physical Review A*, 71(3):032318, 2005. [arXiv:quant-ph/0404132]. 52, 142

[31] J. Clark and D. A. Holton. *A first look at graph theory*. World Scientific Publishing, Singapore, 1991. 102

[32] S. R. Clark, C. Moura Alves, and D. Jaksch. Efficient generation of graph states for quantum computation. *New Journal of Physics*, 7:124, 2005. 55

[33] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 2001. 2nd edition. 91, 99, 113, 114

[34] W. Van Dam, M. Mosca, and U. Vazirani. How powerful is adiabatic quantum computation? In *FOCS '01: Proceedings of the 42^{nd} IEEE symposium on Foundations of Computer Science*, page 279, Washington, DC, USA, 2001. IEEE Computer Society. 49

[35] Carsten Damm. Problems complete for $\oplus$L. In *Proceedings of the 6th International Meeting of Young Computer Scientists on Aspects and Prospects of Theoretical Computer Science*, pages 130–137, London, UK, 1990. Springer-Verlag. 48

[36] V. Danos and E. Kashefi. Private communication, June 2005. 85

[37] V. Danos and E. Kashefi. Determinism in the one-way model. [arXiv:quant-ph/0506062v1], 2005. 62, 125, 146

[38] V. Danos and E. Kashefi. Determinism in the one-way model. *Physical Review A*, 74:052310, 2006. [arXiv:quant-ph/0506062]. xiv, 57, 85, 86, 131, 155

[39] V. Danos, E. Kashefi, and P. Panangaden. Robust and parsimonious realisations of unitaries in the one-way model. *Physical Review A*, 72:064301, 2006. 36

[40] V. Danos, E. Kashefi, and P. Panangaden. The measurement calculus. *J. ACM*, 54(2):8, 2007. [arXiv:0704.1263]. 52, 57, 59, 64, 66, 67, 68

[41] C. M. Dawson, H. L. Haselgrove, A. P. Hines, D. Mortimer, M. A. Nielsen, , and T. J. Osborne. Quantum computing and polynomial equations over $\mathbb{Z}_2$. *Quantum Information & Computation*, 5(2):102, 2004. [arXiv:quant-ph/0408129]. 32, 161, 162, 163, 164

[42] C. M. Dawson and M. A. Nielsen. The Solovay–Kitaev algorithm. *Quantum Information and Computation*, 6(1):81–95, 2006. 23

[43] Christopher M. Dawson, Henry L. Haselgrove, and Michael A. Nielsen. Noise thresholds for optical cluster-state quantum computation. *Physical Review A*, 73(5):052306, 2006. 56

[44] N. de Beaudrap. Characterizing & constructing flows in the one-way measurement model in terms of disjoint $I - O$ paths. Preprint, [arXiv:quant-ph/0603072], 2006. 79, 80, 140

[45] N. de Beaudrap. Finding flows in the one-way measurement model. *Physical Review A*, 77:022328, 2008. [arXiv:quant-ph/0611284]. 79, 80, 126

[46] N. de Beaudrap, V. Danos, E. Kashefi, and M. Roetteler. Quadratic form expansions for unitaries. In Y. Kawano and M. Mosca, editors, *3rd Workshop on Theory of Quantum Computation, Communication, and Cryptography; University of Tokyo, Tokyo, Japan*, volume 5106 of *Lecture Notes in Computer Science*, page 29. Springer, 2008. [arXiv:0801.2461]. 159

[47] N. de Beaudrap and M. Pei. An extremal result for geometries in the one-way measurement model. *Quantum Information & Computation*, 8(5):430–437, 2008. [arXiv:quant-ph/0702229]. 79, 80, 151

[48] J. Dehaene and B. De Moor. Clifford group, stabilizer states, and linear and quadratic operations over GF(2). *Physical Review A*, 68:042318, 2003. [arXiv:quant-ph/0304125]. xv, 160, 186, 187

[49] M. Van den Nest, W. Dür, A. Miyake, and H. J. Briegel. Fundamentals of universality in one-way quantum computation. *New Journal of Physics*, 9(6):204–254, 2007. [arXiv:quant-ph/0702116]. 18, 19, 57

[50] D. Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. In *Proceedings of the Royal Society of London*, volume 400 of *A*, page 97, 1985. xi, 1

[51] D. Deutsch. Quantum computers. *Computer Bulletin*, 3(2):24, June 1987.

[52] D. Deutsch. Quantum computational networks. In *Proceedings of the Royal Society of London*, volume 425 of *A*, page 467, 1989.

[53] D. Deutsch. Quantum mechanics near closed timelike lines. *Physical Review D*, 44(10):3197–3217, 1991. 154

[54] R. Diestel. *Graph Theory*. Springer-Verlag, Heidelberg, New York, 3rd edition, 2005. PDF version freely available on the author's website, at [www.math.uni-hamburg.de/home/diestel/books/graph.theory]. 80, 87, 98

[55] D. P. DiVincenzo and P. W. Shor. Fault-tolerant error correction with efficient quantum codes. *Phys. Rev. Lett.*, 77(15):3260–3263, 1996. [arXiv:quant-ph/9605031].

[56] A. Einstein. The foundation of the general theory of relativity. *Annalen Phys.*, 49:769–822, 1916. 29

[57] A. Einstein, B. Podolsky, and N. Rosen. Can Quantum-Mechanical description of physical reality be considered complete? *Physical Review*, 47(10):777–780, 1935. 3

[58] E. Farhi, J. Goldstone, and S. Gutmann. A quantum algorithm for the hamiltonian NAND tree. Pre-print: [arXiv:quant-ph/0702144], 2007. 49, 79

[59] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Quantum computation by adiabatic evolution. Pre-print: [arXiv:quant-ph/0001106], 2000. 49, 79

[60] Edward Farhi and Sam Gutmann. Quantum computation and decision trees. *Phys. Rev. A*, 58(2):915–928, Aug 1998. 49

[61] R. P. Feynman. Space-time approach to non-relativistic quantum mechanics. *Rev. Mod. Phys.*, 20(2):367–387, Apr 1948. 162

[62] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6 and 7):467–488, 1982. xi, 1

[63] R.P. Feynman. *The Feynman Lectures on Computation*. Perseus Publishing, Cambridge, Massachusetts, 1996. T.Hey and R.W. Allen (eds.).

[64] R. P. Feynmann and A. R. Hibbs. *Quantum Mechanics and Path Integrals*. McGraw–Hill, New York, 1965. 162

[65] R. W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962. 91

[66] A. G. Fowler, S. J. Devitt, and L. C. L. Hollenberg. Implementation of Shor's algorithm on a linear nearest neighbour qubit array. *Quantum Information & Computation*, 4:237–251, 2004. [arXiv:quant-ph/0402196]. 161, 180

[67] G. C. Ghirardi, A. Rimini, and T. Weber. Unified dynamics for microscopic and macroscopic systems. *Physical Review D*, 34(2):470–491, 1986. 14

[68] D. Gottesman. *Stabilizer codes and quantum error correction*. PhD in Physics, California Institute of Technology, 1200 East California Blvd, Pasadena CA, 91125, 1997. [arXiv:quant-ph/9705052]. xv, 42, 45

[69] D. Gottesman. Theory of fault-tolerant quantum computation. *Phys. Rev. A*, 57(1):127–137, 1998. [arXiv:quant-ph/9702029]. 42, 48

[70] D. Gottesman and I. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402, 1999. Preprint version, entitled "Quantum Teleportation is a Universal Computational Primitive" available as [arXiv:quant-ph/9908010v1]. 50, 51, 52, 54, 79

[71] M. Grassl, A. Klappenecker, and M. Roetteler. Graphs, quadratic forms, and quantum codes. *Proceedings of the 2002 IEEE International Symposium on Information Theory*, page 45, 2002. [arXiv:quant-ph/0703112]. 160

[72] D. Gross and J. Eisert. Novel schemes for measurement-based quantum computation. *Physical Review Letters*, 98(22):220503, 2007. 57

[73] J. E. Gubernatis. Monte Carlo simulations of fermion systems: the determinant method. In *Monte-Carlo methods and applications in Neutronics, Photonics, and Statistical Physics*, Lecture Notes in Physics, pages 212–221. Springer-Verlag, 1985. 195

[74] M. Hein, J. Eisert, and H. J. Briegel. Multiparty entanglement in graph states. *Phys. Rev. A*, 69(6):062311, 2004. [arXiv:quant-ph/0307130]. xv, xvi, 77, 186, 190, 192, 193

[75] C. K. Hong, Z. Y. Ou, and L. Mandel. Measurement of subpicosecond time intervals between two photons by interference. *Phys. Rev. Lett.*, 59(18):2044–2046, Nov 1987. 54

[76] R. Jozsa. On the simulation of quantum circuits. [arXiv:quant-ph/0603163], 2006. 50

[77] Samuel J. Lomonaco Jr. and Louis H. Kauffman. Topological quantum computing and the jones polynomial. Pre-print: [arXiv:quant-ph/0605004], 2006. 49

[78] E. Kashefi. Private communication, November 2007. 154

[79] Alastair Kay, Jiannis K. Pachos, and Charles S. Adams. Graph-state preparation and quantum computation with global addressing of optical lattices. *Physical Review A*, 73(2):022310, 2006. 55

[80] K. Kieling, T. Rudolph, and J. Eisert. Percolation, renormalization, and quantum computing with nondeterministic gates. *Physical Review Letters*, 99(13):130501, 2007. 56

[81] A. Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997. 23

[82] A. Y. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303:2–30, 2003. 49

[83] A. Y. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and quantum computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 2002. 18, 19, 20, 22, 23

[84] E. Knill, R. Laflamme, A. Ashikhmin, H. Barnum, L. Viola, and W. H. Zurek. Introduction to quantum error correction. *Los Alamos Science*, (27):188–221, 2002. [arXiv:quant-ph/0207170]. 56

[85] E. Knill, R. Laflamme, and G. J. Milburn. A scheme for efficient quantum computation with linear optics. *Nature*, 409:46–52, 2001. 54

[86] Pieter Kok. Lecture notes on optical quantum computing. Pre-print: [arXiv:0705.4193], 2007. 54

[87] D. Kretschmann, D. Schlingemann, and R. F. Werner. The information-disturbance tradeoff and the continuity of Stinespring's representation. *IEEE Transactions on Information Theory*, 54(4):1708–1717, 2008. [arXiv:quant-ph/0605009]. 21

[88] R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(3):183, 1961. xi, 1

[89] R. Landauer. Information is physical. *Physics Today*, pages 23–29, 1991. xi

[90] D. W. Leung. Two-qubit projective measurements are universal for quantum computation. Pre-print: [arXiv:quant-ph/0111122], 2001. 52, 57

[91] Daniel Lidar and K. Birgitta Whaley. Decoherence-free subspaces and subsystems. In *Irreversible Quantum Dynamics*, Lecture Notes in Physics, pages 83 – 120, 2003. 56

[92] Yuan Liang Lim, Sean D. Barrett, Almut Beige, Pieter Kok, and Leong Chuan Kwek. Repeat-until-success quantum computing using stationary and flying qubits. *Physical Review A*, 73(1):012304, 2006. 54, 55

[93] Yuan Liang Lim, Almut Beige, and Leong Chuan Kwek. Repeat-until-success linear optics distributed quantum computing. *Physical Review Letters*, 95(3):030505, 2005. 54

[94] G.W. Makey. *The Mathematical Foundations of Quantum Mechanics*. W.A. Benjamin, New York — Amsterdam, 1963.

[95] Nicolas C. Menicucci, Steven T. Flammia, and Olivier Pfister. One-way quantum computing in the optical frequency comb. *Physical Review Letters*, 101(13):130501, 2008. 55

[96] M. Mhalla and S. Perdrix. Finding optimal flows efficiently. In *ICALP'08, the 35th International Colloquium on Automata, Languages and Programming*, 2008. To appear. See also [arXiv:0709.2670]. xiv, 62, 147, 148, 149, 150, 155, 172, 173, 178

[97] Chetan Nayak, Steven H. Simon, Ady Stern, Michael Freedman, and Sankar Das Sarma. Non-abelian anyons and topological quantum computation. *Reviews of Modern Physics*, 80(3):1083, 2008. 49

[98] J. W. Negele and H. Orland. *Quantum Many-Particle Systems*. Frontiers in Physics. 1988.

[99] M. A. Nielsen. Conditions for a class of entanglement transformations. *Phys. Rev. Lett.*, 83(2):436–439, 1999. 38

[100] M. A. Nielsen. Quantum computation by measurement and quantum memory. *Physics Letters A*, 308:96–100, 2003. [arXiv:quant-ph/0108020]. 52

[101] M. A. Nielsen, C. M. Caves, B. Schumacher, and H. Barnum. Information theoretic approach to quantum error correction and reversible measurement. In *Proceedings of the Royal Society of London*, volume 454 of *A*, pages 277–304, 1998. [arXiv:quant-ph/9706064]. 78, 196, 198

[102] M. A. Nielsen and I. L. Chuang. Errata list for *Quantum Computation and Quantum Information*. Available online at [web.squint.org/qci/errata/errata/errata.html]; see [103]. 26, 207

[103] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000. See errata [102]. xv, 11, 22, 25, 26, 207

[104] Michael A. Nielsen. Optical quantum computation using cluster states. *Phys. Rev. Lett.*, 93(4):040503, Jul 2004. 54

[105] Michael A. Nielsen. Cluster-state quantum computation. *Reports on Mathematical Physics*, 57(1):147 – 161, 2006. [arXiv:quant-ph/0504097]. 55

[106] E. Nuutila. Efficient transitive closure computation in large digraphs. In *Acta Polytechnica Scandinavica, Mathematics and Computing in Engineering Series #74*, Helsinki, 1995. Ph.D. Thesis. Postscript chapters freely available on the author's website, at [www.cs.hut.fi/~enu/thesis.html]. 91, 114

[107] C.H. Papadimitriou. *Computational Complexity*. Addision-Wesley, 1994.

[108] K. N. Patel, I. L. Markov, and J. P. Hayes. Optimal synthesis of linear reversible circuits. *Quantum Information & Computation*, 8:282–294, 2008. [arXiv:quant-ph/0302002]. 130, 187, 190

[109] K. R. Popper. *Quantum Theory and the Schism in Physics: from the Postscript to the Logic of Scientific Discovery*. Rowman and Littlefield, Totowa, New Jersey, 1982. 14

[110] R. Prevedel, M. S. Tame, A. Stefanov, M. Paternostro, M. S. Kim, and A. Zeilinger. Experimental demonstration of decoherence-free one-way information transfer. *Physical Review Letters*, 99(25):250503, 2007. 56

[111] R. Raussendorf and H. J. Briegel. Quantum computing via measurements only. [arXiv:quant-ph/0010033], 2000. 207

[112] R. Raussendorf and H. J. Briegel. A one-way quantum computer. *Physical Review Letters*, 86(5188), 2001. See also [111] for a pre-print version. xii, 52, 53, 54, 57, 58, 61, 73, 79, 85

[113] R. Raussendorf, D. E. Browne, and H. J. Briegel. Measurement-based quantum computation on cluster states. *Physical Review A*, 68(2):022312, Aug 2003. [arXiv:quant-ph/0301052]. viii, 52, 69, 73, 74, 85, 140, 142, 144, 180, 198

[114] R. Raussendorf, J. Harrington, and K. Goyal. Topological fault-tolerance in cluster state quantum computation. *New Journal of Physics*, 9(6):199, 2007. [arXiv:quant-ph/0703143]. 56

[115] Peter P. Rohde, Timothy C. Ralph, and William J. Munro. Error tolerance and tradeoffs in loss- and failure-tolerant quantum computing schemes. *Physical Review A*, 75(1):010302, 2007. [arXiv:quant-ph/0603130]. 56

[116] D. Schlingemann. Cluster states, algorithms and graphs. *Quantum Information & Computation*, 4(4):287, 2004. [arXiv:quant-ph/0305170]. xv, 160, 161

[117] D. Schlingemann and R. F. Werner. Quantum error-correcting codes associated with graphs. *Physical Review A*, 65(1):012308, 2001. 160

[118] E. Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *Physical Review*, 28(6):1049–1070, Dec 1926. 13

[119] L. S. Schulman. *Techniques and Application of Path Integration*. Wiley-Interscience, New York, 1981. 162

[120] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35thAnnual Symposium on Foundations of Computer Science*, pages 124–134. IEEE Computer Society Press, 1994. xi

[121] D.R. Simon. On the power of quantum computation. *SIAM J. Computing*, 26:1474, 1997. xi

[122] K. Simon. An improved algorithm for transitive closure on acyclic digraphs. *Theoretical Computer Science*, 58:325–346, 1988. 114

[123] S. Sippu and E. Soisalon-Soinenen. *Languages and Parsing*, volume I of *Parsing Theory*. Springer Verlag, Berlin, 1988. 114

[124] W. F. Stinespring. Positive functions on C*-algebras. *Proceedings of the American Mathematical Society*, 6(2):211–216, 1955. 21

[125] M. S. Tame, M. Paternostro, and M. S. Kim. One-way quantum computing in a decoherence-free subspace. *New Journal of Physics*, 9(6):201, 2007. 56

[126] Tetsufumi Tanamoto, Yu xi Liu, Shinobu Fujita, Xuedong Hu, and Franco Nori. Producing cluster states in charge qubits and flux qubits. *Physical Review Letters*, 97(23):230501, 2006. 55

[127] R. E. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972. 91, 114

[128] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. In *Proceedings of the London Mathematical Society*, 2, pages 230–265, 1937. xi, 1

[129] Michael Varnava, Daniel E. Browne, and Terry Rudolph. Loss tolerance in one-way quantum computation via counterfactual error correction. *Physical Review Letters*, 97(12):120501, 2006. [arXiv:quant-ph/0507036]. 56

[130] J. von Neumann. *Mathematische Grundlagen der Quantenmechanik*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen mit besonderer Berücksichtigung der Anwendungsgebiete. Springer, Berlin, 1932. See [131] for the English translation. 1

[131] J. von Neumann. *Mathematical Foundations of Quantum Mechanics*. Number 2 in Investigations in physics. Princeton University Press, Princeton, 1955. Translated by R. Beyer. 3, 9, 13, 14, 208

[132] D. Wallace. Worlds in the Everett interpretation. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 33:637–661, 2002. [arXiv:quant-ph/0103092]. 14

[133] J. Watrous. Theory of Quantum Information. Lecture notes for the Spring 2007 course *Advanced Research Topics* (CS 798) at the University of Waterloo. xv, 6, 21

[134] John Watrous. Quantum simulations of classical random walks and undirected graph connectivity. Pre-print: [arXiv:cs/9812012], 1998. 49

[135] Andrew Chi-Chih Yao. Quantum circuit complexity. In *34th annual symposium on Foundations of Computer Science*, pages 352–361, Palo Alto, California, USA, 1993. 1

[136] W. H. Zurek. Quantum Darwinism and Envariance. [arXiv:quant-ph/0308163], 2003. 14