Homology, Hopf Algebras and Quantum Code Surgery

Alexander Cowtan
Wolfson College

Department of Computer Science
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Trinity 2024

Abstract

This thesis is a study of quantum error-correction codes from an algebraic perspective. We concern ourselves not only with quantum codes but also protocols to perform logical quantum computation using such codes. We derive new methods of performing fault-tolerant quantum computation, rooted in abstract algebra and category theory. We also generalise known constructions of quantum codes and rigorously formalise existing constructions.

At its core, the main question this thesis asks is: what *is* lattice surgery? For quantum computer scientists, the easiest answer is that it is a method of performing quantum computation with a famous family of quantum error-correction codes, called *surface codes*. The method is extremely efficient and preserves the tolerance of the codes to errors throughout.

Upon further inspection, however, lattice surgery has connections to some interesting abstract mathematics. It functions, at the basic level, by taking patches of code, each of which protects some logical information, and 'glues' them together, or 'tears' them apart, yielding operations on the encoded information. This gluing bears similarity to *connected sums* in topology, a geometric modification on topological spaces whereby we join two manifolds together using a local submanifold. On the encoded information, lattice surgery yields operations which can be seen as multiplication and comultiplication of the Hopf algebra $\mathbb{C}\mathbb{Z}_2$ or its dual, where gluing gives multiplication and tearing gives comultiplication. Given these facts, we would like to understand lattice surgery purely in these algebraic terms.

We can think of surface codes, in some sense, as being the combination of the cellulation of a manifold and an algebra $\mathbb{C}\mathbb{Z}_2$. This understanding naturally leads us to generalisations of lattice surgery. Loosening the definition leads us, in the first case, to abandon cellulations of manifolds and get quantum error-correction codes which are homological but not based explicitly on manifolds. Happily, surgery still works in this wider setting, in much the same way. In the second case, we can consider Kitaev's quantum double models, which are described by $\mathbb{C}G$ for some finite group G, and can be defined by other Hopf algebras more generally. Lattice surgery according to (co)multiplication of Hopf algebras can still be performed in this case, albeit with several caveats. On the way, we find and clarify many algebraic intricacies of such models. We now describe these two directions in more detail.

In the first direction, we define code maps between Calderbank-Shor-Steane (CSS) codes using maps between chain complexes, and generalise the technique of lattice surgery to such codes. We describe how to 'merge' and 'split' along a shared $\overline{X}$ or $\overline{Z}$ operator between arbitrary CSS codes in an error-corrected manner, so long as conditions concerning gauge-fixing and systolic distance are satisfied. To do this, we introduce a formalism based on colimits from category theory. As well as describing a surgery operation, this gives a general recipe for new codes. We prove that such merges and splits on quantum Low-Density Parity Check (qLDPC) codes yield codes which are themselves qLDPC. We then present open-source software, called SSIP (Safe Surgery by Identifying Pushouts), which automates the procedure of finding and performing valid code surgeries. We demonstrate on qLDPC codes, which are not topological codes in general, and are of interest for near-term fault-tolerant quantum computing. Such qLDPC codes include lift-connected surface codes, generalised bicycle codes and bivariate bicycle codes. We show empirically that various logical measurements can be performed cheaply by surgery without sacrificing the high code distance.

In the second direction, we then move to the Kitaev quantum double model. We approach this topic in a formal algebraic manner, emphasising the quantum double $D(G)$ symmetry for G a finite group. We use the description of quasiparticles as irreducible representations and combine this with the $D(G)$-bimodule properties of open ribbon excitation spaces to show how open ribbons can be used to teleport information between sites. We show how our constructions generalise to $D(H)$ models based on a finite-dimensional Hopf algebra H, including site actions of $D(H)$ and partial results on ribbon equivariance even when the Hopf algebra is not semisimple. We take a diversion to prove that lattice surgery can be performed using any Kitaev model on a patch, where G is finite Abelian. We relate the surgery procedures to the qudit ZX-calculus, a graphical language for reasoning about qudit quantum computing.

Returning to the Kitaev model with non-Abelian G, we then provide a systematic treatment of boundaries based on subgroups $K \subseteq G$ with the Kitaev model in the bulk. The boundary sites are representations of a $*$-subalgebra Ξ and we explicate its structure as a strong $*$-quasi-Hopf algebra dependent on a choice of transversal R. We provide decomposition formulae for irreducible representations of $D(G)$ pulled back to Ξ. We also provide explicitly the monoidal equivalence of the category of Ξ-modules and the category of G-graded K-bimodules and use this to prove that different choices of R are related by Drinfeld cochain twists. Examples include $S_{n-1} \subset S_n$ and an example related to the octonions where Ξ is also a Hopf quasigroup. As an application of our treatment, we study patches with boundaries based on $K = G$ horizontally and $K = \{e\}$ vertically and give a partial description of how these could be used in a quantum computer using lattice surgery.

Acknowledgements

I first owe my supervisor, Aleks Kissinger. I greatly appreciate his guidance, encouragement and warmth, and for many useful discussions about the ZX-calculus and quantum error-correction, which informed much of this thesis.

I am also indebted to Shahn Majid, who taught me much about quantum groups during the difficult lockdown period and wrote sections of the papers which became the latter half of this thesis. Shahn's patience, rigour and keen eye for detail were invaluable.

The work on chain complexes is in large part due to the input of Simon Burton. I am inspired by his ideas, passion and breadth of knowledge of stabiliser codes.

The fourth key influence on this thesis is Ross Duncan, who set me on this path, whether he intended to or not. There are too many ways in which I learned from Ross to put into words, and I am beyond grateful. If I could do it all again, I would in a heartbeat, with the possible exception of getting a bit lost in a Boston snowstorm.

In the Quantum Group, I will miss fruitful discussions and good times with many colleagues, such as John van de Wetering, Lia Yeh, Will Simmons, Nihil Shah, Stefano Gogioso, Matty Hoban, Nick Ormrod, Nicola Pinzani, Tein van der Lugt, Amin Karamlou, Razin Shaikh, Jan Głowacki and Richie Yeung. In Quantinuum, I am grateful for the company and friendship of Seyon Sivarajah, Silas Dilkes, Nathan Fitzpatrick, Alec Edgington, Dan Mills, Steven Herbet, Ben Criger, John Children, Sherilyn Wright, Carys Harvey and many more.

Much of the follow-on work from this thesis was sparked by thought-provoking discussions at the Fault-Tolerant Quantum Technologies 2024 conference in Benasque. I would like to thank Nikolas Breuckmann, Armando Quintavelle, Michael Vasmer and Christophe Vuillot for organising, and the attendees who asked such interesting questions.

Outside of my profession, special thanks go to Ben Johnson, Kostja Junglas, Rage and others in Oxford, and to all my friends elsewhere, for bringing light and joy to my life.

I benefited from the immense financial generosity of Simon Harrison, through the Wolfson Harrison UK Research Council Quantum Foundation Scholarship.

To Rory Green and the rest of my family, thank you. Without your boundless love and encouragement I would never have made it this far.

Contents

Chapter 1

Introduction

We would like to construct large quantum computers which are capable of running quantum algorithms at large problem sizes, a project which is well under way at the time of writing [AAA24, R-ABB24]. Many of these quantum algorithms are substantially faster than their best-known classical counterparts, to the extent that a variety of problems at sufficiently large sizes are essentially impossible to solve classically yet are, in theory, highly tractable with quantum computers [Mon16]. These problems range from cryptography to quantum chemistry, and solving them would have important scientific and societal implications.

While the study of such quantum algorithms is a large and fascinating field in its own right, we concern ourselves here with the construction of quantum computers.

The greatest obstacle to the deployment of large quantum computers is the presence of *quantum errors*. These are not due to the mistakes of a *user* of a quantum computer, but instead emerge naturally because of the *environment*, or because of precision errors in device calibration. Storing and manipulating quantum data is necessarily a noisy affair, and quantum components will interact in undesirable ways. Despite this, it is still theoretically possible to build a large quantum computer which is *fault-tolerant*, meaning it can accommodate quantum errors up to a certain threshold.

The most promising method for accommodating for quantum errors is via *quantum error correction*, which works by using redundancy of quantum components. That is, there are many interconnected quantum components, and should errors arise in some limited number of them, they can be detected and corrected for by some error correction protocol. Such protocols are often dictated by *quantum error correction codes*, or 'quantum codes' for short. In such codes, a smaller amount of logical data is stored within a larger amount of physical data. This is analogous to classical error correction, used extensively in data transmission over unreliable or noisy channels [MS83]. As one might expect, the quantum version is substantially more complicated due to basic principles of quantum mechanics such as the no-cloning theorem.

In this thesis, we are not only concerned with storing quantum data safely in memory using quantum codes, but also performing computation in a fully error-corrected fashion. We approach this problem abstractly, in an algebraic fashion. We use tools from homology, Hopf algebras and category theory.

We will give a rigorous but brief introduction to quantum errors, before moving on to the algebraic notions required to understand the thesis. Unfortunately for reasons of brevity we do not include an introduction to quantum mechanics or quantum information, but there are excellent textbooks on these topics with different foci [NC10, CK17].

We will study two broad classes of quantum codes: those which can be described using homological algebra, and those which can be described by Hopf algebras. We have divided the thesis into two halves correspondingly.

Part A: Homological codes

Part B: Hopf algebraic codes

There is not a precise delineation between the two, as we will use some basic Hopf algebra notions when defining surgery between homological codes, and also some homology for proofs about Hopf algebraic codes.

1.1 Attribution

This thesis is composed of the following Chapters, which are each based on a publication or preprint:

- Chapter 2: Based on [CB24], a paper with Dr. Simon Burton, published in Quantum 8 (2024). In Section 2.2.3 we elaborate on the relationship between code maps at the logical and physical levels. This is not in the published work, and we thank Clément Poirson, Robert I. Booth and Christophe Vuillot for bringing this omission to our attention.
- Chapter 3: Based on [Cow24], a solo-author preprint. In Section 3.5.1 we provide an extension, which is not in the preprint, on 'basis-changing' logical measurements. We thank Zhiyang He for helpful discussions in this direction.
- Chapter 4: Based on [CM22], a paper with Prof. Shahn Majid, published in the Journal of Mathematical Physics 63 (2022).
- Chapter 5: Based on [Cow22], a solo-author paper, accepted to Quantum Physics & Logic 2022.
- Chapter 6: Based on [CM23], a paper with Prof. Shahn Majid, published in the Journal of Mathematical Physics 64 (2023).

The information contained in the introduction is relevant to both halves of the thesis, and so we have included it here. All of the introductory material is well-known, and not our original work. We give examples and some intuition but only occasional proofs.

1.2 Quantum error correction

While we cover the basic notions of quantum error correction here, we will later abstract away from inspecting the low-level behaviour of individual errors, and instead study the high-level algebraic structure of quantum codes. Nevertheless, this low-level behaviour is fundamentally important to the topic.

We skim over these notions briskly. For much more comprehensive introductory treatments see [LB13, Got24].

To start with, we can consider the state space of a pure quantum system to be a complex Hilbert space $\mathcal{H}$. A state $|\psi\rangle$ is then a vector in $\mathcal{H}$, which at times we require to be normalised, and typically we are unconcerned with global phase, so $|\psi\rangle \sim |\phi\rangle \iff |\psi\rangle = \alpha\,|\phi\rangle$ for some $\alpha \in \mathbb{C}^\times$. In quantum computing, $\mathcal{H}$ is typically (but not always) taken to be finite-dimensional, so $\mathcal{H} \cong \mathbb{C}^d$ for some $d \in \mathbb{N}$. For example, a qubit is $\mathbb{C}^2$. In this thesis we only deal with the finite-dimensional case. A pure quantum map is then a linear map $L : \mathcal{H}_A \to \mathcal{H}_B$ between Hilbert spaces. Throughout we use the term 'operator' interchangeably with 'linear map'.

To treat quantum errors, however, we must consider *mixed* quantum systems, that is systems with states which are a statistical mixture of pure quantum states. These systems are represented by a subset of $\mathcal{B}(\mathcal{H})$, the space of linear maps on $\mathcal{H}$. The subset we use is the set of *density operators*, which are positive semi-definite, self-adjoint operators. Recall that a positive semi-definite operator's eigenvalues are real and positive. A density operator ρ can always be represented as

$$\rho = \sum_j p_j \left|\psi_j\right\rangle \left\langle\psi_j\right| ,$$

for $p_j \in \mathbb{R}$. We can view $\left|\psi_j\right\rangle \left\langle\psi_j\right|$ as a kind of 'doubling' of the pure quantum system, appearing in the statistical mixture ρ with probability p_j. This doubling conveniently erases the distinction between global phases. See [CK17] for a diagrammatic treatment of this doubling.

A *quantum channel* is a completely positive trace-preserving (CPTP) map Φ between density operators. Every CPTP map can be expressed as

$$\Phi(\rho) = \sum_i A_i \rho A_i^\dagger .$$

This is the Kraus form of a quantum channel, where we have decomposed the channel into Kraus operators A_i, such that $\sum_i A_i A_i^\dagger = \mathrm{id}$. Each Kraus operator occurs with probability $\mathrm{Tr}\left(A_i \rho A_i^\dagger\right)$. Assuming we would like to put in a quantum state and receive that same state afterwards, we can consider each Kraus operator which is not equal to the identity (up to a scalar factor) as being a possible error. Different factorisations of Φ give decompositions into different errors, which are related by unitaries so are equivalent in a suitable sense.

One might want to relax the definition of quantum errors to account for the fact that we may not want to apply the identity channel. We may wish to apply a quantum logic gate but end up inadvertently with a different one.

The nomenclature would imply that the quantum channel is something through which we send quantum information between two locations, but this is not generally the case. If we wish to hold quantum data in memory in a single location, it too will undergo a process determined by the quantum channel.

Example 1.2.1. [Got24, Sec. 1.2.3] Let the quantum system be a qubit, i.e. $\mathbb{C}^2$. We can have the *depolarising channel* on the qubit. Recalling that a mixed qubit state is a density operator $\rho \in \mathcal{B}(\mathbb{C}^2)$, the unbiased depolarising channel is

$$\Phi_p(\rho) = (1-p)\rho + \frac{p}{3} X\rho X + \frac{p}{3} Y\rho Y + \frac{p}{3} Z\rho Z$$

with X, Y, Z being the Pauli matrices.

This channel has equal probability of an X, Y or Z error. If $p = 3/4$, the channel deterministically sends any density operator to the maximally mixed state id_2, up to normalisation.

In that example, the individual Kraus operators were Paulis, which are unitary and self-adjoint; errors are not required to have these properties in general.

The core of quantum error correction is to take a quantum channel, say $\mathcal{E}$, representing the errors on our system, and implement a *recovery channel*, i.e. CPTP map, $\mathcal{R}$ such that $(\mathcal{R} \circ \mathcal{E})(\rho) = \rho$ for relevant states ρ. When this is possible is determined by the *Knill-Laflamme quantum error-correction criterion.*

Theorem 1.2.2. [KL00] Let $L \subset \mathcal{H}$ be a subspace of a larger, physical space. Let $\mathcal{E}$ be a quantum channel with errors $\{A_a\}$, and let L have a basis $\{|i\rangle\}$. Then there exists a recovery operation $\mathcal{R}$ such that $(\mathcal{R} \circ \mathcal{E})(\rho_L) = \rho_L$ if and only if

$$\langle i|A_a^\dagger A_b|j\rangle = c_{ab}\langle i|j\rangle$$

where ρ_L has support only on the subspace, and c_{ab} are coefficients uniquely determined by A_a and A_b.

Incidentally, this defines quantum error-correction codes for us. A code is the subspace L, which we call the *logical space*, along with its inclusion into $\mathcal{H}$ and the quantum channel $\mathcal{E}$ it is subjected to. Happily, this lets us specify a quantum error-correction code without reference to the recovery operation; we only need know that it exists.

We sometimes refer to the logical space as the codespace, and a state in this space as a codeword. It is also common to define a quantum error-correction code without reference to the error set, just the logical space, and the error set can be inferred.

In addition to relaxing the definition of a quantum error-correction code, we may want to give additional information. For example, we commonly associate a set of measurements, described by self-adjoint operators, such that the codespace L is the mutual $+1$ eigenspace of these measurements. We may wish to assign a tensor decomposition of the codespace into components; for example if $\dim L = d^k$, we can assign an isomorphism $L \cong (\mathbb{C}^d)^{\otimes k}$. The implementation of such codes on a gate-based quantum computer also requires a compilation into *quantum circuits*, i.e. a composition of quantum gates.

There is one last helpful consequence of the quantum error-correction criterion. If a code can correct for the errors $\{A_a\}$, then it can also correct arbitrary linear sums thereof. This linearity is the fundamental reason why quantum error-correction works: despite the set of possible errors being continuous, it can be discretised into finite generators.

It is common to consider physical spaces which are tensor products of components, so that $\mathcal{H} = (\mathbb{C}^d)^{\otimes n}$ for some qudits of dimension d. In this case, we say that the *weight* of an operator A in $\mathcal{B}(\mathcal{H})$ is the number of components on which A acts nontrivially.

Example 1.2.3. Let $\mathcal{H} = (\mathbb{C}^2)^{\otimes 5}$ and $A = X \otimes Y \otimes I \otimes I \otimes Z$. Then A has weight 3.

Definition 1.2.4. The *distance* of a quantum error-correction code is the lowest weight error A such that

$$\langle i|A|j\rangle \neq c(A)\langle i|j\rangle$$

for some $c(A)$ which depends only on A, and where i, j run over all basis elements of L.

The distance is denoted d_Q or just d, and is distinct from the qudit dimension d. The two are not to be confused, and throughout we will be careful to make the distinction clear from context.

Definition 1.2.5. (Stabiliser code)
Let $(\mathbb{C}^2)^{\otimes n}$ be the physical Hilbert space. We call the $\mathbb{C}^2$ components the *data qubits*. Define a logical subspace as follows. Let $\mathscr{S} \subset \mathscr{P}^n$ be a subgroup of the n-qubit Pauli group on $(\mathbb{C}^2)^{\otimes n}$. Let $L \subset (\mathbb{C}^2)^{\otimes n}$ be the space of states such that

$$U|\psi\rangle = |\psi\rangle, \quad \forall U \in \mathscr{S}, |\psi\rangle \in L,$$

so L is *stabilised* by $\mathscr{S}$.

For this space to have non-zero dimension, $\mathscr{S}$ must be abelian and not contain $-I$. As $\mathscr{S}$ is a group we can define a generating set $\mathcal{G}$, which is not unique or minimal in general. Elements of $\mathcal{G}$ are then tensor products of Pauli operators, which are self-adjoint; these become the measurements on our code, such that L is the mutual $+1$ eigenspace of $\mathcal{G}$.

Lemma 1.2.6. [Got24, Sec. 3.3.3]
Let r be the number of independent generators of $\mathscr{S}$, so $|\mathscr{S}| = 2^r$. Then

$$\dim L = 2^k,$$

where $k = n - r$.

Hence there are k *logical qubits* in the logical space L.

Qubit stabiliser codes are ubiquitous in quantum error-correction, for their practical applicability, algebraic simplicity, wide variety and ease to simulate using the stabiliser formalism [AG04]. They are straightforward to generalise to qudits, depending slightly on the dimension [Got24, Sec. 8], and requiring some bookkeeping as the natural generalisations of Paulis to qudits are not self-adjoint.

Because the Paulis form a basis of $\mathcal{B}(\mathbb{C}^2)$, any error on a qubit can be decomposed into a linear combination of Paulis. The quantum error-correction criterion then implies that for qubit codes we only need to consider correction of Paulis, and tensor products thereof. Stabiliser codes are precisely those codes which detect Pauli errors, so are the most natural quantum codes on qubits.

Definition 1.2.7. We say that the *parameters* of a stabiliser code are $[\![n, k, d]\!]$, with n the number of data qubits, k the number of logical qubits and d the code distance.

These do not uniquely characterise a code; there will generally be several different codes with the same parameters.

In Part A we devote our attention to a subclass of stabiliser codes, called Calderbank-Shor-Steane codes. In Part B we tackle quantum codes which come from condensed matter and Hopf algebras; these are not generally stabiliser codes.

There is a great deal more to be said about quantum codes which we have not touched on. If we wish to compute with a gate-based quantum computer then the codes must be compiled into quantum circuits. The real question at this lower level of abstraction is then how tolerant the *circuits* are to errors, which is related to but not the same as how tolerant the *codes* are to errors. As the compiled quantum circuits perform a combination of component initialisation, entanglement between quantum components, and then measurement, typically using ancillae, i.e. extra components, environmental errors can occur at every step, including the measurements. Other sources of error which do not come from the environment are *coherent errors*, unitary errors which occur due to faulty control of the device. For example, we could intend to apply a $R_Z(\theta)$ gate to a physical qubit but instead apply an $R_Z(\theta + \epsilon)$ gate, for some small angle ϵ. Precise calibration of devices is vital to prevent such errors, and calibration of the many parameters which go into device control is extremely difficult. Lastly, one more source of error is *leakage*, when the size of physical system considered $\mathcal{H}$ is actually inadequate due to coupling to the external world, and the evolution of the computation is described by some operator which includes components outside of the physical system we have considered. Thus, amplitudes may drop such that the state is no longer normalised when considered only in $\mathcal{H}$. We do not consider such detailed and accurate error models in this thesis.

Classical computation is required in tandem with the quantum computer to *decode* measurement results and return likely candidates for errors, which can then be corrected. Such decoders can be designed for specific code families [Hig22, PK21, Del14].

A code which is compiled down to the hardware level only satisfies the properties of a *quantum memory.* For this to be useful we must then be able to perform operations with the quantum memory in a way which does not destroy the carefully implemented error-correction. Several quantum codes admit *transversal gates* [Got24, Sec. 11], which perform separable unitary operations on physical quantum components to apply logical unitary operations to the encoded data, in a manner which does not spread errors uncontrollably throughout the code. Unfortunately, this is always insufficient for universal quantum computation [EK09], so for the quantum computers to be useful we must modify the physical and logical spaces throughout computation in a non-unitary manner.

This problem of performing modifications of the spaces while maintaining error-correction is central to this thesis, and we will see different approaches to this in later sections.

1.3 Category theory

Category theory is a large and algebraically dense subject. Quoting Leinster's textbook [Lei14], "category theory takes a bird's eye view of mathematics". It is less about particular mathematical objects and more about how objects related to each other. Composition of functions, operators etc. are crucial.

In this thesis, we use relatively basic category theory to study composition of quantum codes in Part A, as well as to take a bird's eye view of quasiparticles in Part B. Here we pick out a select few aspects of category theory which we will use. For more introductory material, see e.g. [Lei14, Mac78, HV19].

Definition 1.3.1. A category $\mathscr{C}$ contains the following data:

- a collection of objects $\mathrm{Obj}(\mathscr{C})$,
- for each pair of objects $A, B \in \mathrm{Obj}(\mathscr{C})$ a collection of morphisms $\mathrm{Hom}(A, B)$ from A to B, such that morphisms from $\mathrm{Hom}(B, C)$ and $\mathrm{Hom}(A, B)$ compose associatively,
- an identity morphism $\mathrm{id}_A \in \mathrm{Hom}(A, A)$.

Associativity means that for any morphisms $f \in \mathrm{Hom}(A, B)$, $g \in \mathrm{Hom}(B, C)$, $h \in \mathrm{Hom}(C, D)$, we have

$$(h \circ g) \circ f = h \circ (g \circ f)$$

and the identity morphism satisfies $f \circ \mathrm{id}_A = f = \mathrm{id}_B \circ f$ for any $f \in \mathrm{Hom}(A, B)$.

We say ‘collection’ rather than ‘set’ of objects and morphisms to avoid size problems, *à la* Russell’s paradox and similar. These are irrelevant to our work here, so we do not dwell on it.

Definition 1.3.2. An *isomorphism* $f : A \to B$ satisfies $g \circ f = \mathrm{id}_A$ and $f \circ g = \mathrm{id}_B$ for some $g : B \to A$.

Example 1.3.3. Some basic examples of categories are:

- `Set`, with sets as objects and functions as morphisms,
- `Grp`, with groups as objects and group homomorphisms as morphisms,
- $\mathtt{Vect}_{\Bbbk}$, with vector spaces over a field $\Bbbk$ as objects and linear maps as morphisms,
- `CPM`, with density operators as objects and completely-positive maps as morphisms.

All these examples are ‘`Set`-like’, in that their objects are sets with possible extra structure, and morphisms are functions which are compatible with this structure. While not all categories are of this form, the ones we use in this thesis are. These are called *concrete categories.*

Definition 1.3.4. Let $\mathscr{C}$ and $\mathscr{D}$ be categories. A functor $F : \mathscr{C} \to \mathscr{D}$ has the following data:

- a function on objects $F : \mathrm{Obj}(\mathscr{C}) \to \mathscr{D}$,
- a function $\mathrm{Hom}(A, B) \to \mathrm{Hom}(F(A), F(B))$, written $F(f)$ for $f \in \mathrm{Hom}(A, B)$, such that $F(g \circ f) = F(g) \circ F(f)$ and $F(\mathrm{id}_A) = \mathrm{id}_{F(A)}$.

For example, there are forgetful functors $G : \mathtt{Grp} \to \mathtt{Set}$ and $F : \mathtt{Vect}_{\Bbbk} \to \mathtt{Grp}$, which forget the relevant extra structure of the domain. So, to every group we can associate its set of group elements. To every vector space we can associate its group of vectors under linear composition. Such functors also compose, so we have $G \circ F : \mathtt{Vect}_{\Bbbk} \to \mathtt{Set}$.

Consequently we have the definition of an *isomorphic* functor. That is, for every category $\mathscr{C}$ we have the identity functor $\mathrm{id}_{\mathscr{C}}$, acting as identity on objects and morphisms. Isomorphisms satisfy $G \circ F = \mathrm{id}_{\mathscr{C}}$, $F \circ G = \mathrm{id}_{\mathscr{D}}$ for functors $F : \mathscr{C} \to \mathscr{D}$, $G : \mathscr{D} \to \mathscr{C}$.

This implies another category, that of `Cat`, with (small) categories as objects and functors as morphisms. `Cat` is in fact a 2-category or bicategory, that is a ‘higher’ version of category, as functors also have morphisms between them, called *natural transformations*. We do not deal with higher categories in detail here, but we will use natural transformations.

Definition 1.3.5. Given two functors $F, G : \mathscr{C} \to \mathscr{D}$, a *natural transformation* $\zeta : F \Rightarrow G$ is the assignment of a morphism between objects $F(A) \to G(A)$ for every $A \in \mathrm{Obj}(\mathscr{C})$, such that the following diagram commutes:

$$\begin{tikzcd}
F(A) \arrow[r, "\zeta_A"] \arrow[d, "F(f)"] & G(A) \arrow[d, "G(f)"] \\
F(B) \arrow[r, "\zeta_B"] & G(B)
\end{tikzcd}$$

for every morphism f in $\mathscr{C}$.

By “commuting” we simply mean that all paths around the diagram are equal, so $\zeta_B \circ F(f) = G(f) \circ \zeta_A$ in this case.

If every ζ_A is an isomorphism then ζ is said to be a *natural isomorphism*.

Intuitively, if functors are maps between categories which preserve internal categorical structure, then natural transformations are maps between functors which preserve internal *functorial* structure.

We can use natural transformations to define successively weaker versions of an isomorphism between categories. The first of these is an *equivalence*.

Definition 1.3.6. An equivalence of categories $\mathscr{C}$, $\mathscr{D}$ is a pair of functors $F : \mathscr{C} \to \mathscr{D}$, $G : \mathscr{D} \to \mathscr{C}$, such that $G \circ F \cong \mathrm{id}_{\mathscr{C}}$ and $F \circ G \cong \mathrm{id}_{\mathscr{D}}$.

When the natural isomorphisms are identities this reduces to an isomorphism of categories again. The next weaker version is an *adjunction*.

Definition 1.3.7. An adjunction of categories $\mathscr{C}$, $\mathscr{D}$ is a pair of functors $F : \mathscr{C} \to \mathscr{D}$, $G : \mathscr{D} \to \mathscr{C}$, such that there is a pair of natural transformations $\eta : \mathrm{id}_{\mathscr{C}} \to G \circ F$, $\varepsilon : F \circ G \to \mathrm{id}_{\mathscr{D}}$ satisfying

$$\begin{tikzcd}
F \arrow[r, "F\eta"] \arrow[dr, "\mathrm{id}_F"'] & FGF \arrow[d, "\varepsilon F"] & G \arrow[r, "\eta G"] \arrow[dr, "\mathrm{id}_G"'] & GFG \arrow[d, "G\varepsilon"] \\
 & F & & G
\end{tikzcd}$$

We say that $F \dashv G$. F is left adjoint, and G is right adjoint.

Example 1.3.8. Free and forgetful functors tend to form an adjunction, with the free functor being the left adjoint and forgetful the right. If G is the forgetful functor `Grp` $\to$ `Set`, then F is the free functor `Set` $\to$ `Grp` sending each set S to its free group $F(S)$.

1.3.1 Universal properties

Universal constructions and properties are useful when we would like to have an object which is canonical in some sense. That is, an object which is the only one (up to isomorphism) which satisfies some conditions. Examples include taking subsets, subgroups and subspaces, as well as quotients and ‘gluings’ of algebraic structures. There are many different but equivalent ways of defining universal properties [Lei14]. We use the most straightforward, by way of (co)cones.

Let $\mathscr{A}$ and $\mathscr{I}$ be categories. A functor $D : \mathscr{I} \to \mathscr{A}$ is called a diagram in $\mathscr{A}$ of shape $\mathscr{I}$.[1]

Definition 1.3.9. (Cone) Let $D : \mathscr{I} \to \mathscr{A}$ be a diagram in $\mathscr{A}$. A cone on D is an object $A \in \mathscr{A}$ and a family $(A \to D(I))_{I \in \mathscr{I}}$ of morphisms in $\mathscr{A}$, such that for all morphisms $u : I \to J$ in $\mathscr{I}$ the following triangle commutes:

$$\begin{tikzcd}
 & D(I) \arrow[dd, "D(u)"] \\
A \arrow[ur, "f_I"] \arrow[dr, "f_J"'] & \\
 & D(J)
\end{tikzcd}$$

[1]Strictly speaking, we require $\mathscr{I}$ to be *small* to avoid size problems but in this thesis all (co)limits we use are finite anyway.

Definition 1.3.10. (Limit) A limit of D is a cone $(p_I : L \to D(I))_{I\in\mathscr{I}}$ such that for any cone on D there is a unique morphism $\overline{f} : A \to L$ satisfying $p_I \circ \overline{f} = f_I$ for all $I \in \mathscr{I}$.

$$\begin{tikzcd}
& & D(I) \arrow[dd, "D(u)"] \\
A \arrow[urr, "f_I"] \arrow[r, dotted, "\overline{f}"] \arrow[drr, "f_J"'] & L \arrow[ur, "p_I"'] \arrow[dr, "p_J"] & \\
& & D(J)
\end{tikzcd}$$

That is, a limit is a cone which satisfies the *universal property* described above, so a limit is called a *universal construction.* The dotted arrow is called a *mediating map.* The intuition in concrete categories is that a limit is the 'minimal' object L in $\mathscr{A}$ which satisfies the commutation relations of the cone, and any other object A which satisfies those relations is 'at least as large' and factors through L. This is only defined up to isomorphism, as A and L could have unique morphisms $g : A \to L$ and $h : L \to A$ such that $h \circ g = \mathrm{id}_A$ and $g \circ h = \mathrm{id}_L$, and hence both A and L could be described as 'the' limit.

Example 1.3.11. A categorical product in $\mathscr{A}$ is the limit of the diagram $D : \mathscr{I} \to \mathscr{A}$, where $\mathscr{I}$ has only two objects and no non-identity morphisms.

This coincides with the cartesian product $\times$ of sets and groups in `Set` and `Grp`, and the direct sum $\oplus$ of vector spaces in $\mathtt{Vect}_{\Bbbk}$. The maps p_I and p_J are the projections onto the component sets, groups and spaces.

We can also take the product $\times$ of (small) categories in `Cat`, which is just the product of the sets of objects, extended to also take products of morphisms in the same way.

There are many more examples of limits, including terminal objects, equalisers and pullbacks. Generally, not all limits necessarily exist in an arbitrary category.

A finite limit is one in which $\mathscr{I}$ is a finite category, having a finite number of objects and morphisms. The categorical product is a finite limit. In this work all our limits are finite.

Definition 1.3.12. (Cocone) Let $D : \mathscr{I} \to \mathscr{A}$ be a diagram in $\mathscr{A}$. A cocone on D is an object $A \in \mathscr{A}$ and a family $(D(I) \to A)_{I\in\mathscr{I}}$ of morphisms in $\mathscr{A}$, such that for all morphisms $u : I \to J$ in $\mathscr{I}$ the following triangle commutes:

$$\begin{tikzcd}
D(I) \arrow[dd, "D(u)"'] \arrow[dr, "f_I"] & \\
& A \\
D(J) \arrow[ur, "f_J"'] &
\end{tikzcd}$$

Definition 1.3.13. (Colimit) A colimit of D is a cocone $(p_I : D(I) \to L)_{I\in\mathscr{I}}$ such that for any cocone on D there is a unique morphism $\overline{f} : L \to A$ satisfying $\overline{f} \circ p_I = f_I$ for all

$I \in \mathscr{I}$.

$$\begin{tikzcd}
D(I) \arrow[dd, "D(u)"'] \arrow[dr, "f_I"] \arrow[drr, "p_I"] & & \\
& A \arrow[r, dotted, "\overline{f}"] & L \\
D(J) \arrow[ur, "f_J"] \arrow[urr, "p_J"'] & &
\end{tikzcd}$$

Colimits are dual to limits in a sense which can be made formal, but we do not go into that here. The intuition of colimits in concrete categories is that a colimit is the 'maximal' object L in $\mathscr{A}$ which satisfies the commutation relations of the cocone, and any other object A which satisfies those relations is 'at least as small' and factors through L. Like limits, colimits are defined only up to isomorphism.

Example 1.3.14. A categorical coproduct in $\mathscr{A}$ is the colimit of the diagram $D : \mathscr{I} \to \mathscr{A}$, where $\mathscr{I}$ has only two objects and no non-identity morphisms.

This coincides with the disjoint union $\sqcup$ of sets in `Set`, the free product $*$ of groups in `Grp`, and the direct sum $\oplus$ of vector spaces in $\texttt{Vect}_{\Bbbk}$; note that the product and coproduct of vector spaces coincide. Other examples of colimits include initial objects, coequalisers and pushouts. We shall meet these later on.

1.3.2 Monoidal categories

Apart from the usage of category theory to give algebraic structures which are canonical, we can take another perspective. In this perspective we rely less on universality and instead focus on *parallel*, in addition to sequential, composition of objects and morphisms. This perspective starts with *monoidal categories*. For further introductory material on monoidal categories for quantum theory, see [HV19, CK17]. In some cases, monoidal category theory can be more easily understood using string diagrams. We will see this in the context of the ZX-calculus in Section 1.5.

Definition 1.3.15. (Monoidal category) A monoidal category is a category $\mathscr{C}$ equipped with:

- a *monoidal product* functor $\otimes : \mathscr{C} \times \mathscr{C} \to \mathscr{C}$,
- a distinguished object $1 \in \mathrm{Obj}(\mathscr{C})$ called the *monoidal unit*,
- an *associator* natural isomorphism α with components

$$\alpha_{A,B,C} : (A \otimes B) \otimes C \to A \otimes (B \otimes C),$$

- *left* and *right unitor* natural isomorphisms λ, ρ with components

$$\lambda_A : 1 \otimes A \to A \qquad \rho_A : A \otimes I \to A,$$

such that the triangle and pentagon equations are satisfied for all objects A, B, C, D in $\mathrm{Obj}(\mathscr{C})$:

$$\begin{tikzcd}
(A \otimes I) \otimes B \arrow[rr, "\alpha_{A,I,B}"] \arrow[dr, "\rho_A \otimes \mathrm{id}_B"'] & & A \otimes (I \otimes B) \arrow[dl, "\mathrm{id}_A \otimes \lambda_B"] \\
& A \otimes B &
\end{tikzcd}$$

$$\begin{tikzcd}
 & (A \otimes B) \otimes (C \otimes D) \arrow[dr, "\alpha_{A\otimes B,C,D}"] & \\
A \otimes (B \otimes (C \otimes D)) \arrow[ur, "\alpha_{A,B,C\otimes D}"] \arrow[d, "\mathrm{id}_A \otimes \alpha_{B,C,D}"'] & & ((A \otimes B) \otimes C) \otimes D \\
A \otimes ((B \otimes C) \otimes D) \arrow[rr, "\alpha_{A,B\otimes C,D}"] & & (A \otimes (B \otimes C)) \otimes D \arrow[u, "\alpha_{A,B,C} \otimes \mathrm{id}_D"']
\end{tikzcd}$$

As the associator and unitors are natural isomorphisms they must satisfy Definition 1.3.5 appropriately.

Example 1.3.16. Monoidal structures are common in categories. For example,

- `Set` and `Grp` are monoidal w.r.t. the cartesian product $\times$,
- $\mathtt{Vect}_{\Bbbk}$ and `CPM` are both monoidal w.r.t. to the direct sum $\oplus$ and the tensor product $\otimes$.

Definition 1.3.17. A monoidal category is strict if all the natural isomorphisms are identity transformations, meaning that all arrows in the equations above are equalities.

It is commonly sufficient to *think* of monoidal categories as being strict, even if they technically are not. Formally, every monoidal category is monoidally equivalent to some strict monoidal category [Mac78, Sec. VII], a well-known result which for concrete categories can be made even stronger [Sch01]. Sometimes we do care about the actual associators and unitors, however.

Definition 1.3.18. A *braided monoidal category* is equipped with a natural isomorphism

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

such that the following conditions hold:

$$\begin{tikzcd}
 & A \otimes (B \otimes C) \arrow[r, "\sigma_{A,B\otimes C}"] & (B \otimes C) \otimes A \arrow[dr, "\alpha_{B,C,A}"] & \\
(A \otimes B) \otimes C \arrow[ur, "\alpha_{A,B,C}"] \arrow[dr, "\sigma_{A,B} \otimes \mathrm{id}_C"'] & & & B \otimes (C \otimes A) \\
 & (B \otimes A) \otimes C \arrow[r, "\alpha_{B,A,C}"] & B \otimes (A \otimes C) \arrow[ur, "\mathrm{id}_B \otimes \sigma_{A,C}"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
 & (A \otimes B) \otimes C \arrow[r, "\sigma_{A\otimes B,C}"] & C \otimes (A \otimes B) \arrow[dr, "\alpha^{-1}_{C,A,B}"'] & \\
A \otimes (B \otimes C) \arrow[ur, "\alpha^{-1}_{A,B,C}"'] \arrow[dr, "\mathrm{id}_A \otimes \sigma_{B,C}"'] & & & (C \otimes A) \otimes B \\
 & A \otimes (C \otimes B) \arrow[r, "\alpha^{-1}_{A,C,B}"'] & (A \otimes C) \otimes B \arrow[ur, "\sigma_{A,C} \otimes \mathrm{id}_B"'] &
\end{tikzcd}$$

That is, a braided monoidal category has a braiding σ which is compatible with associativity, such that we can braid objects in both directions either 'all at once' or 'one at a time'. This is more-or-less the categorical version of the Yang-Baxter equation for braid groups.

Definition 1.3.19. If $\sigma_{B,A} \circ \sigma_{A,B} = \mathrm{id}_{A\otimes B}$ for all $A, B \in \mathrm{Obj}(\mathscr{C})$ then $\mathscr{C}$ is a *symmetric monoidal category.*

Example 1.3.20. All of our previous examples, `Set`, `Grp`, $\mathtt{Vect}_{\Bbbk}$ and `CPM` are symmetric monoidal categories.

The category of $\mathbb{Z}_3$-graded vector spaces has a non-symmetric braided monoidal structure, where the braiding is given by a scalar obtained by multiplication of degrees of elements. That is,

$$\sigma_{A,B} : x \otimes y \mapsto \omega^{|x||y|} y \otimes x$$

for elements $x \in A, y \in B$, where $\omega = e^{2\pi i/3}$ is the primitive 3rd root of unity.

We can also ask for a functor to preserve the monoidal structure, and for monoidal equivalences, but we only need these for the very last section of the thesis. See [Mac78, Sec. VII] for the relevant definitions.

The last piece we need is duality.

Definition 1.3.21. Given an object X in a monoidal category, an object X^* is the *left dual* of X (and X is the right dual of X^*) if there exist morphisms $\eta : 1 \to X \otimes X^*$ and $\varepsilon : X^* \otimes X \to 1$ such that the following diagrams commute:

$$\begin{tikzcd}
1 \otimes X \arrow[rr, "\lambda"] \arrow[d, "\eta\otimes\mathrm{id}"] & & X \\
(X \otimes X^*) \otimes X \arrow[r, "\alpha"] & X \otimes (X^* \otimes X) \arrow[r, "\mathrm{id}\otimes\varepsilon"] & X \otimes 1 \arrow[u, "\rho"']
\end{tikzcd}$$

$$\begin{tikzcd}
X^* \otimes 1 \arrow[rr, "\rho"] \arrow[d, "\mathrm{id}\otimes\eta"] & & X^* \\
X^* \otimes (X \otimes X^*) \arrow[r, "\alpha^{-1}"] & (X^* \otimes X) \otimes X^* \arrow[r, "\varepsilon\otimes\mathrm{id}"] & 1 \otimes X^* \arrow[u, "\lambda"']
\end{tikzcd}$$

In practice, all our categories have objects being finite-dimensional vector spaces with extra structure, so they have all duals – such categories are called 'rigid' or 'autonomous' categories – and moreover we do not have to worry about which of X and X^* is left or right dual; that they are dual to each other is enough. Using the duals of objects, we can also take the dual of a morphism $\phi : A \to B$, which will become $\phi^* : B^* \to A^*$. In particular,

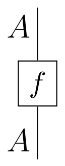
$$\phi^* = (\varepsilon_B \otimes \mathrm{id}_{A^*}) \circ (\mathrm{id}_{B^*} \otimes \phi \otimes \mathrm{id}_{A^*}) \circ (\mathrm{id}_{B^*} \otimes \eta_A),$$

which will be clearer in terms of string diagrams.

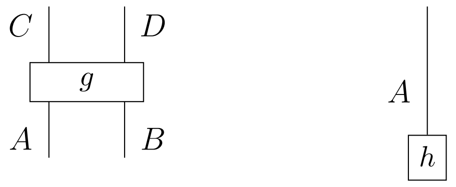
1.3.3 String diagrams

It can be instructive and at times helpful in calculations to use *string diagrams* to describe monoidal categories. String diagrams date back to Penrose diagrams [Pen63] and were used for proofs in quantum algebra as far back as the mid 90s [Maj95, Ch. 9]. Many of the ways in which they are used today, see [Wet12], are direct descendants of these early works. As we shall see, string diagrams look quite different to the commutative diagrams earlier.

With string diagrams, an object in a monoidal category is represented by a ‘string’; here we use the convention going from bottom to top. We may label the string with the name of the object, say A, or leave it implicit. A morphism $f : A \to A$ is a box on the string.



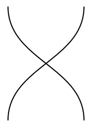
We can also have morphisms between different objects, say $g : A \otimes B \to C \otimes D$, or $h : 1 \to A$, where 1 is the monoidal unit:



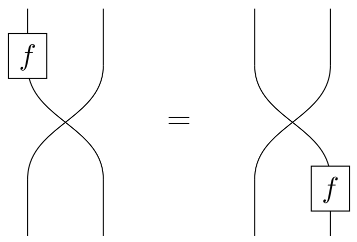
The morphism $h : 1 \to A$ can then be identified with a *state* in A; equally, a morphism $A \to 1$ can be identified with an *effect* on A. Objects and morphisms in a tensor product are just placed in adjacency, so that the above diagram represents

$$g \otimes h : A \otimes B \otimes 1 \to C \otimes D \otimes A.$$

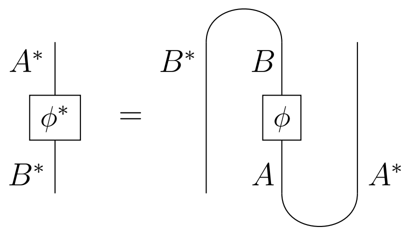
In a symmetric monoidal category, we allow for wire crossings.



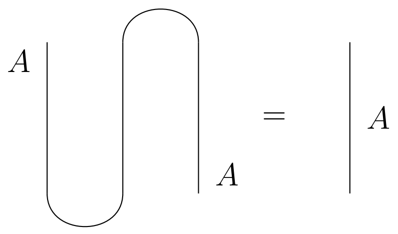
The properties of symmetric monoidal categories, such as the interchange law, and functoriality of the monoidal product, are inherent in the diagrams, as we can slide boxes along wires, and past crossings.



We can also use diagrams for nontrivially braided monoidal categories, but in this thesis we only use string diagrams in the symmetric context. We can also incorporate duals into string diagrams, with the dual of a morphism $\phi : A \to B$ being $\phi^* : B^* \to A^*$



where we have 'cups' and 'caps' as the duality morphisms $\eta_A : 1 \to A \otimes A^*$ and $\varepsilon : B^* \otimes B \to 1$ respectively. These are presented as bends in the wires, rather than boxes, to highlight their special 'yanking' property



which encapsulates the first commutative diagram of Definition 1.3.21. The second is the flipped version.

That these diagrams are sound, and hence can be used to reason formally about monoidal categories, is a consequence of work by Mac Lane [Mac78] and Joyal & Street [JS91]. Readers familiar with quantum computing will see that quantum circuits live in a symmetric monoidal category, albeit traditionally going from left to right, and quantum gates are morphisms on qubits, or more generally qudits. Of course, the quantum circuit model is somewhat limited, in that gates traditionally take the form of either unitaries, such as the Paulis, Hadamard, CNOT, CZ etc., or single-qubit preparations and measurements. In this thesis, we only use string diagrams in a rudimentary manner, but use the ZX-calculus, which 'natively' captures isometries and Kraus operators.

There are other categories we make use of in this thesis which are braided or symmetric monoidal, such as categories of chain complexes, and representation categories of certain algebras, but we do not use string diagrams for these.

1.4 Homological algebra

Homology arose out of the study of topology, and in that context can be thought of as a way of assigning invariants to topological spaces. Homological algebra has extended far beyond that, however, influencing many other areas of algebra [Wei84]. In this thesis, as with category theory, we only require elementary homological algebra, and only describe the aspects relevant to us. We treat it purely in terms of linear algebra over $\mathbb{F}_2$, while homology theory in general can be applied to any Abelian category.

Let $\mathtt{Mat}_{\mathbb{F}_2}$ be the category which has as objects linearised finite sets over $\mathbb{F}_2$, so each object is a finite-dimensional vector space V equipped with a specified basis $\tilde{V}$ such that $V = \mathbb{F}_2\tilde{V}$. Each element of $\tilde{V}$ corresponds to an entry for vectors in V. Importantly, we require the vector spaces to have this form on the nose for later applications. Throughout, we will call these *based vector spaces.* A morphism $f : V \to W$ in $\mathtt{Mat}_{\mathbb{F}_2}$ is a $\dim W \times \dim V$ matrix valued in $\mathbb{F}_2$.

Let $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ be the category of bounded chain complexes in $\mathtt{Mat}_{\mathbb{F}_2}$. We now recap some of the basic properties of this category. A chain complex $C_\bullet$ has the following form:

$$\begin{tikzcd} \cdots \arrow[r] & C_{n+1} \arrow[r, "\partial_{n+1}"] & C_n \arrow[r, "\partial_n"] & C_{n-1} \arrow[r] & \cdots \end{tikzcd}$$

where each component C_i is a based vector space and $n \in \mathbb{Z}$ is called the degree of the component in $C_\bullet$. $C_\bullet$ has $\mathbb{F}_2$-matrices as differentials $\partial_{n+1} : C_{n+1} \to C_n$ such that $\partial_n \circ \partial_{n+1} = 0 \pmod 2$, $\forall n \in \mathbb{Z}$. To disambiguate differentials between chain complexes we will use $\partial_n^{C_\bullet} := \partial_n \in C_\bullet$ when necessary.

All our chain complexes are bounded, meaning there is some $k \in \mathbb{Z}$ such that $C_{n>k} = 0$ and $l \in \mathbb{Z}$ such that $C_{n<l} = 0$, i.e. it is bounded above and below. We call $k - l$ the length of $C_\bullet$ for k and l the smallest and largest possible values respectively.

Definition 1.4.1. Given a chain complex $C_\bullet$ we let

$$Z_n(C_\bullet) = \ker(\partial_n); \quad B_n(C_\bullet) = \operatorname{im}(\partial_{n+1})$$

and call Z_n, B_n the n-cycles and n-boundaries. We also define a quotient $H_n(C_\bullet) = Z_n(C_\bullet)/B_n(C_\bullet)$, and call H_n the nth homology space of $C_\bullet$.

Recall that $\dim(\ker(\partial_n)) = \operatorname{null}(\partial_n) = \dim C_n - \operatorname{rank}(\partial_n)$. Throughout we sometimes use $\ker(f)$ of a matrix f to mean the kernel object, i.e. subspace, and sometimes the kernel morphism, i.e. inclusion map. It should be clear from context which is meant.

Example 1.4.2. Let $\Gamma = (V, E)$ be a finite simple undirected graph. We can form the incidence chain complex $C_\bullet$ of Γ, which has $C_0 = \mathbb{F}_2 V$, $C_1 = \mathbb{F}_2 E$. All other components are zero. The sole nonzero differential ∂_1 is the incidence matrix of Γ, with $(\partial_1)_{ij} = 1$ if the jth edge is attached to the ith vertex, and 0 otherwise. $H_1(C_\bullet)$ is determined by the graph homology of Γ [Wei84].

Definition 1.4.3. A morphism $f_\bullet : C_\bullet \to D_\bullet$ in $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ is called a chain map, and consists of a collection of matrices $\{f_i : C_i \to D_i\}_{i\in\mathbb{Z}}$ such that each resultant square of maps commutes:

$$\begin{tikzcd}
\cdots \arrow[r] & C_{n+1} \arrow[r, "\partial^{C_\bullet}_{n+1}"] \arrow[d, "f_{n+1}"] & C_n \arrow[r, "\partial^{C_\bullet}_{n}"] \arrow[d, "f_n"] & C_{n-1} \arrow[r] \arrow[d, "f_{n-1}"] & \cdots \\
\cdots \arrow[r] & D_{n+1} \arrow[r, "\partial^{D_\bullet}_{n+1}"] & D_n \arrow[r, "\partial^{D_\bullet}_{n}"] & D_{n-1} \arrow[r] & \cdots
\end{tikzcd}$$

As we specified *bounded* chain complexes only a finite number of the f_i matrices will be non-zero. A chain map $f_\bullet$ is an isomorphism in $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ iff all f_i are invertible, in which case one can think of the isomorphism as being a 'change of basis' for all components, which thus transforms the differential matrices appropriately. Every pair of chain complexes has at least two chain maps, the zero chain maps, between them, given by a collection of entirely zero matrices either way.

Lemma 1.4.4. A chain map at a component $f_n : C_n \to D_n$ restricts and then lifts to a matrix $H_n(f_\bullet) : H_n(C_\bullet) \to H_n(D_\bullet)$.

Proof. It is easy to check that f_n induces matrices from $Z_n(C_\bullet) \to Z_n(D_\bullet)$ and the same for B_n. ■

This lemma is equivalent to saying that $H_n(-)$ is a functor from $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2}) \to \texttt{Mat}_{\mathbb{F}_2}$.

$\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ has several known categorical properties which will be useful to us. One way to see a chain complex $C_\bullet$ in $\texttt{Mat}_{\mathbb{F}_2}$ is as a $\mathbb{Z}$-graded $\mathbb{F}_2$-vector space, with a specified basis and a distinguished map $\partial : C_\bullet \to C_\bullet$ with components $\partial_i : C_{i+1} \to C_i$, such that $\partial \circ \partial = 0$. Many of the properties of $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ are inherited directly from those of $\mathbb{Z}$-graded $\mathbb{F}_2$-vector spaces.

$\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ is an additive category, i.e. its morphisms can add together, and it has all finite biproducts, i.e. direct sums. These have components

$$(C \oplus D)_n = C_n \oplus D_n$$

and the same for differentials. This is both a categorical product and coproduct. Homology preserves direct sums: given chain complexes $C_\bullet$ and $D_\bullet$,

$$H_n((C \oplus D)_\bullet) \cong H_n(C_\bullet) \oplus H_n(D_\bullet)$$

This is obvious, considering the blocks of each differential in $(C \oplus D)_\bullet$.

Lemma 1.4.5. The category of chain complexes is Abelian.

Proof. Recall that an Abelian category is an additive category such that:

1. Every morphism has a kernel and cokernel.
2. Every monomorphism is the kernel of its cokernel.
3. Every epimorphism is the cokernel of its kernel.

Recall that $\mathtt{Mat}_{\mathbb{F}_2}$ has all kernels and cokernels, i.e. subspaces and quotient spaces. Then given a chain map $f : C_\bullet \to D_\bullet$ we define $\ker(f)$ with

$$\begin{tikzcd}
\cdots \arrow[r, dotted] & K_{n+1} \arrow[r, dotted, "\partial^{K_\bullet}_{n+1}"] \arrow[d, "\ker(f_{n+1})"'] & K_n \arrow[r, dotted] \arrow[d, "\ker(f_n)"] & \cdots \\
\cdots \arrow[r] & C_{n+1} \arrow[r, "\partial^{C_\bullet}_{n+1}"] \arrow[d, "f_{n+1}"'] & C_n \arrow[r] \arrow[d, "f_n"] & \cdots \\
\cdots \arrow[r] & D_{n+1} \arrow[r, "\partial^{D_\bullet}_{n+1}"] & D_n \arrow[r] & \cdots
\end{tikzcd}$$

where $\partial^{K_\bullet}_n$ always exists and is uniquely defined, because

$$f_n \circ \partial^{C_\bullet}_{n+1} \circ \ker(f_{n+1}) = \partial^{D_\bullet}_{n+1} \circ f_{n+1} \circ \ker(f_{n+1}) = 0$$

and so by the universal property of $\ker(f_n)$ there is a unique matrix $\partial^{K_\bullet}_{n+1} : K_{n+1} \to K_n$. These satisfy $\partial^{K_\bullet}_{n+1} \circ \partial^{K_\bullet}_{n+1} = 0$ as

$$\ker(f_n) \circ \partial^{K_\bullet}_{n+1} \circ \partial^{K_\bullet}_{n+2} = \partial^{C_\bullet}_{n+1} \circ \partial^{C_\bullet}_{n+2} \circ \ker(f_{n+2}) = 0$$

and then kernels are monic. $K_n = \{v \in C_n \mid f_n(v) = 0\}$ by the definition of kernels in $\mathtt{Mat}_{\mathbb{F}_2}$. Given the correct choice of basis, $\partial^{K_\bullet}_n$ is thus just $\partial^{C_\bullet}_n \circ \ker(f_n)$ as a matrix but without the all-zero rows which map into C_n/K_n.

That $\ker(f)$ is a genuine kernel in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ is straightforward to check but we do not give further details.

The reversed argument applies for cokernels, giving quotient complexes $D_\bullet/\mathrm{im}(f)$ with components $D_n/\mathrm{im}(f_n)$ etc.

The other two conditions, that every monomorphism is the kernel of its cokernel and every epimorphism is the cokernel of its kernel, follow using the fact that they hold degree-wise in $\mathtt{Mat}_{\mathbb{F}_2}$. ■

Remark 1.4.6. As $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ is additive, equalisers and coequalisers can be seen as special cases of kernels and cokernels by defining $\mathrm{eq}(f,g) = \ker(f-g)$ and $\mathrm{coeq}(f,g) = \mathrm{coker}(f-g)$, for $f, g : C_\bullet \to D_\bullet$. For the chain complex part $E_\bullet$ of an equaliser we have components $E_n = \{c \mid f(c) = g(c)\} \subseteq C_n$. For the chain complex part $F_\bullet$ of a coequaliser, we have components $F_n = D_n/f(c) \sim g(c)$, for $c \in C_n$.

Corollary 1.4.7. An immediate consequence of $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ being Abelian is that it has all finite limits and colimits [Mac78, Sec. VIII].

$\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ also has a monoidal structure.

Definition 1.4.8. [Wei84, Sec. 2.7] Let $C_\bullet, D_\bullet$ be chain complexes in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$. Define $(C \otimes D)_\bullet$ with components

$$(C \otimes D)_n = \bigoplus_{i+j=n} C_i \otimes D_j$$

where the latter tensor product is the normal tensor product in $\mathtt{Mat}_{\mathbb{F}_2}$. Differentials between components are given for $\partial_l^{(C\otimes D)}$ by matrices

$$\begin{pmatrix} \partial_i^C \otimes I \\ I \otimes \partial_j^D \end{pmatrix}$$

for a given i, j, then stacked horizontally for each term $i, j \mid i + j = l$, and vertically for each $i', j' \mid i' + j' = l - 1$. One can check that $\partial_{l-1}^{(C\otimes D)} \circ \partial_l^{(C\otimes D)} = 0 \pmod 2$, as desired.

Also define the object $\mathbf{1}_\bullet \in \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ as

$$\mathbf{1}_\bullet = \cdots \longrightarrow 0 \longrightarrow \mathbf{1}_0 \longrightarrow 0 \longrightarrow \cdots$$

where $\mathbf{1}_0 = \mathbb{F}_2$, and all other $\mathbf{1}_i$ are 0.

One can check that $(C \otimes D)_\bullet$ is a monoidal product $\otimes$ in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, which follows from associativity and distributivity of $\oplus$ and $\otimes$ in $\mathtt{Mat}_{\mathbb{F}_2}$. For the unit, observe that

$$(C \otimes \mathbf{1})_n = C_n \otimes 1 = C_n; \quad \partial_n^{(C\otimes\mathbf{1})_\bullet} = \begin{pmatrix} \partial_n^{C_\bullet} \otimes \mathrm{id}_{\mathbf{1}_0} \\ \mathrm{id}_{C_n} \otimes \partial_0^{\mathbf{1}_\bullet} \end{pmatrix} = \partial_n^{C_\bullet},$$

where in the last matrix all other contributions are zero.

Example 1.4.9. Consider two chain complexes of length 1:

$$C_\bullet = \cdots \longrightarrow 0 \longrightarrow C_1 \xrightarrow{\partial_1^{C_\bullet}} C_0 \longrightarrow 0 \longrightarrow \cdots$$

$$D_\bullet = \cdots \longrightarrow 0 \longrightarrow D_1 \xrightarrow{\partial_1^{D_\bullet}} D_0 \longrightarrow 0 \longrightarrow \cdots$$

In this case we have

$$(C \otimes D)_0 = C_0 \otimes D_0; \quad (C \otimes D)_1 = (C_1 \otimes D_0) \oplus (C_0 \otimes D_1); \quad (C \otimes D)_2 = C_1 \otimes D_1$$

for nonzero components, and

$$\partial_1^{(C\otimes D)_\bullet} = \begin{pmatrix} \mathrm{id}_{C_0} \otimes \partial_1^{D_\bullet} & \partial_1^{C_\bullet} \otimes \mathrm{id}_{D_0} \end{pmatrix}; \quad \partial_2^{(C\otimes D)_\bullet} = \begin{pmatrix} \partial_1^{C_\bullet} \otimes \mathrm{id}_{D_1} \\ \mathrm{id}_{C_1} \otimes \partial_1^{D_\bullet} \end{pmatrix}$$

for nonzero differentials. Then

$$\partial_1^{(C\otimes D)_\bullet} \circ \partial_2^{(C\otimes D)_\bullet} = \partial_1^{C_\bullet} \otimes \partial_1^{D_\bullet} + \partial_1^{C_\bullet} \otimes \partial_1^{D_\bullet} = 0 \pmod 2$$

as the matrix partitions factor upon multiplication.

This example illustrates an interesting property of $\otimes$ in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$: both $C_\bullet, D_\bullet$ have only one nonzero differential, but $(C \otimes D)_\bullet$ has two. It is easy to see that given two complexes of lengths s, t the tensor product will have length $s + t$.

Lemma 1.4.10. [Wei84]

$$H_n((C \otimes D)_\bullet) \cong \bigoplus_{i+j=n} H_i(C_\bullet) \otimes H_j(D_\bullet)$$

That is, the homology subspaces factor through the tensor product conveniently. This is also called the Künneth formula. The manner in which the homology factors through does not make $H_n(-)$ a monoidal functor with respect to the tensor product.

A co-chain complex is similar to a chain complex, with some notational differences.

Definition 1.4.11. Given a chain complex $C_\bullet$, where all the components are finite-dimensional, the co-chain complex $C^\bullet$ has components

$$C^n = C_n$$

and differentials

$$\delta^{n-1} = \partial_n^\intercal.$$

We also have co-cycles $Z^n = \ker(\delta^n)$, co-boundaries $B^n = \operatorname{im}(\delta^{n-1})$ and co-homology spaces $H^n = Z^n/B^n$. It is easy to show that $H_n(C_\bullet) \cong H^n(C^\bullet)$.

A co-chain map is defined similarly to a chain map:

$$\begin{tikzcd}
\cdots & C^{n+1} \arrow[l] & C^n \arrow[l, "\delta^n_{C^\bullet}"'] & C^{n-1} \arrow[l, "\delta^{n-1}_{C^\bullet}"'] & \cdots \arrow[l] \\
\cdots & D^{n+1} \arrow[l] \arrow[u, "f^{n-1}"] & D^n \arrow[l, "\delta^n_{D^\bullet}"] \arrow[u, "f^n"] & D^{n-1} \arrow[l, "\delta^{n-1}_{D^\bullet}"] \arrow[u, "f^{n+1}"] & \cdots \arrow[l]
\end{tikzcd}$$

and these are morphisms in the category $\mathtt{Coch}(\mathtt{Mat}_{\mathbb{F}_2})$. There is a functor $H^n(-)$: $\mathtt{Coch}(\mathtt{Mat}_{\mathbb{F}_2}) \to \mathtt{Mat}_{\mathbb{F}_2}$. Given a chain map $f_\bullet$, we automatically also have the cochain map $f^\bullet$, with components $f^n = f_n^\intercal$.

A bounded cochain complex, where the components are all finite-dimensional, can be thought of as the 'dual' to a chain complex. If we took the categorical dual from Definition 1.3.21 of a chain complex then we would get the cochain complex but indexed differently.

1.5 Hopf algebras

Hopf algebras can be thought of as generalisations of groups, and so are sometimes called 'quantum groups' [Maj95]. They have a variety of applications in areas like quantum field theory [HHTBP92], quantum gravity [PST06], quantum computing [Kit03] and models of noncommutative differential geometry [BM20]. We use Hopf algebras to study merges and splits of codes, as well as various properties of quantum double models. The representation theory of Hopf algebras is particularly useful in that case.

To start, recall that an *algebra* is a monoid in the category of $\mathbb{k}$-vector spaces. That is, if A is an algebra we have an associative multiplication $\cdot : A \otimes A \to A$ and a unit $1_A \in A$ such that $1_A \cdot a = a = a \cdot 1_A$ for every $a \in A$. We normally drop the $\cdot$ so that multiplication is $ab := a \cdot b$. Also, recall that the state $1_A \in A$ can be seen instead as a map $\mu : \Bbbk \to A$. A *coalgebra* is then the dual structure, with an associative comultiplication $\Delta : A \to A \otimes A$ and counit $\epsilon : A \to \Bbbk$, such that $(\epsilon \otimes \mathrm{id})\Delta(a) = (\mathrm{id} \otimes \epsilon)\Delta(a) = a$ for every $a \in A$. All maps are linear in the field $\Bbbk$, which for our purposes we can always think of as being $\mathbb{C}$.

Definition 1.5.1. A *bialgebra* is a vector space which is both an algebra and coalgebra, such that multiplication and comultiplication commute. That is,

$$\Delta(ab) = \Delta(a)\Delta(b).$$

In addition, we require that the unit and counit are coalgebra and algebra morphisms respectively, i.e.

$$\Delta(1_A) = 1_A \otimes 1_A; \quad \epsilon(ab) = \epsilon(a)\epsilon(b); \quad \epsilon(1_A) = 1.$$

It can be useful to express the comultiplication of a coalgebra using *Sweedler notation.* Here, we have $\Delta(a) := a_1 \otimes a_2$, where a sum is implied. In full, we would have $\Delta(a) = \sum_i \Delta(a^i) = \sum_i (a^i)_1 \otimes (a^i)_2$ for basis elements a^i of a, and where each $(a^i)_1$ and $(a^i)_2$ could also contain sums, but this is unnecessarily cumbersome. We shuffle indices around to handle coassociativity, e.g.

$$(\Delta \otimes \mathrm{id})\Delta(a) = a_{11} \otimes a_{12} \otimes a_2 = a_1 \otimes a_2 \otimes a_3 = a_1 \otimes a_{21} \otimes a_{22} = (\mathrm{id} \otimes \Delta)\Delta(a).$$

Definition 1.5.2. A Hopf algebra is a bialgebra with an *antipode* map $S : A \to A$, such that

$$a_1 S(a_2) = S(a_1)a_2 = 1_A \epsilon(a),$$

for every $a \in A$.

The antipode takes the place of the inverse in a group. The definition of a Hopf algebra can alternatively be seen in the commuting diagram:

$$\begin{tikzcd}
A \otimes A \arrow[rr, "S\otimes\mathrm{id}"] & & A \otimes A \arrow[d, "\cdot"] \\
A \arrow[u, "\Delta"] \arrow[d, "\Delta"'] \arrow[r, "\epsilon"] & \Bbbk \arrow[r, "\mu"] & A \\
A \otimes A \arrow[rr, "\mathrm{id}\otimes S"'] & & A \otimes A \arrow[u, "\cdot"']
\end{tikzcd}$$

The other definitions, of algebras, coalgebras and bialgebras, can also be couched in terms of commuting diagrams, but this is not how we will use them in practice so we stick with traditional algebraic notation, and occasionally string diagrams when dealing with the ZX-calculus.

We give a dictionary relating the morphisms of a Hopf algebra A in algebraic notation to their string diagram form in Table 1.1. Rather than the boxes from Section 1.3.3 we have chosen red and green circles to represent the morphisms, as they are the same depiction used in the ZX-calculus, where they are called *spiders.* We shall meet the ZX-calculus presently.

Example 1.5.3. Let G be a finite group. Then $\mathbb{C}G$ is a *group algebra* with basis elements labelled by elements of G, and multiplication just given by group multiplication linearised over $\mathbb{C}$. The other morphisms are as follows:

$$\Delta g = g \otimes g; \quad \epsilon(g) = 1; \quad Sg = g^{-1}$$

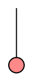
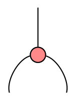
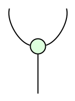
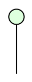
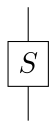
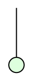
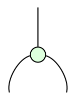
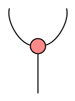
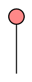
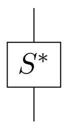
Algebra notation	String diagram
1_A	
$a \otimes b \mapsto ab$	
$a \mapsto a_1 \otimes a_2$	
$a \mapsto \epsilon_A(a)$	
S	
1_{A^*}	
$a^* \otimes b^* \mapsto a^* b^*$	
$a^* \mapsto a_1^* \otimes a_2^*$	
$a^* \mapsto \epsilon_{A^*}(a^*)$	
S^*	

Table 1.1: Dictionary of algebraic notation, incl. Sweedler notation, to string diagrams.

Example 1.5.4. Let G be a finite group. Then $\mathbb{C}(G)$ is the *function algebra* on G, with basis elements labelled by delta functions δ_g. Multiplication is $\delta_g\delta_h = \delta_{g,h}\delta_g$, and the unit is $\frac{1}{|G|}\sum_g \delta_g$. For the coalgebra and antipode we have

$$\Delta\delta_g = \sum_h \delta_h \otimes \delta_{h^{-1}g}; \quad \epsilon(\delta_g) = \delta_{g,e}; \quad S\delta_g = \delta_{g^{-1}}$$

Definition 1.5.5. The dual of a finite-dimensional Hopf algebra A takes every morphism f of the algebra and converts it to its dual f^*, which is now a morphism of A^*.

Thus every unit $1_A \in A$ becomes a counit ϵ on A^*, every multiplication in A becomes comultiplication in A^* and so on, using the 'cups' and 'caps' of Definition 1.3.21.

Example 1.5.6. $\mathbb{C}(G)$ is the *dual* of $\mathbb{C}G$, where $\eta = \sum_g g \otimes \delta_g$ and $\epsilon(\delta_g \otimes h) = \delta_{g,h}$.

Definition 1.5.7. An *integral* in a Hopf algebra is an element Λ such that $\Lambda h = \Lambda\epsilon(h) = h\Lambda$.

Remark 1.5.8. One may wish to consider left integrals, which satisfy the first equality, separately from right integrals, which satisfy the second. This distinction is not important for our purposes, as all semisimple Hopf algebras, including group algebras, are unimodular. The only exception in this thesis is the non-semisimple Hopf algebras in Section 4.3.

$\mathbb{C}G$ has the integral $\frac{1}{|G|}\sum_g g \in \mathbb{C}G$, while $\mathbb{C}(G)$ has the integral $\delta_e \in \mathbb{C}(G)$.

Recall that a complex left representation of a group G is a vector space V equipped with an action $\triangleright : G \otimes V \to V$, such that $e\triangleright v = v$ and $gh\triangleright v = g\triangleright(h\triangleright v)$, for every $v \in V$. As generalisations of groups, Hopf algebras also have representations.

Definition 1.5.9. Let A be an algebra. A left representation, or left module, of A is a vector space V with a left action $\triangleright$, such that $1_A\triangleright v = v$ and $ab\triangleright v = a\triangleright(b\triangleright v)$.

It is well known that if A is a bialgebra then we have tensor products of representations, where $\triangleright : A \otimes V \otimes W \to V \otimes W$ is defined by $a\triangleright(V \otimes W) = a_1\triangleright V \otimes a_2\triangleright W$. Similarly, if A is Hopf then we have duals of representations, with $(h\triangleright f)(v) = f((S(h))\triangleright v)$.

As the representation theory of Hopf algebras is quite a dense subject, we do not describe it here, despite relying on it in Part B. Instead, we recommend [Maj95, Ch. 9]. We will also end up working with *quasi-triangular* and *quasi-Hopf* algebras, which are Hopf algebras with relaxed cocommutativity and coassociativity conditions on the coalgebra part respectively, for which we recommend [Maj95, Ch. 2]. In short, the representation category of a quasi-triangular Hopf algebra is non-trivially braided, and the representation category of a quasi-Hopf algebra has non-trivial monoidal associators.

The ZX-calculus

The ZX-calculus is a formal diagrammatic calculus for qubit quantum computing [Wet12], although it has since been extended in many directions, such as to qudits in various ways [Wan21]. For our purposes, the qubit ZX-calculus can be thought of as an instantiation of the simplest Hopf algebras, $\mathbb{C}\mathbb{Z}_2$ and $\mathbb{C}(\mathbb{Z}_2)$, sitting on the same vector space. Before explaining further, let us take a moment to consider these algebras. First, observe that $S_{\mathbb{C}\mathbb{Z}_2} = S_{\mathbb{C}(\mathbb{Z}_2)} = \mathrm{id}$. Next, as $\mathbb{Z}_2$ is Abelian, $\mathbb{C}\mathbb{Z}_2$ enjoys a Fourier isomorphism which is just the Hadamard gate, i.e. the matrix

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

which is also an isomorphism between $\mathbb{C}\mathbb{Z}_2$ and $\mathbb{C}(\mathbb{Z}_2)$. To emphasise, $\mathbb{C}\mathbb{Z}_2$ and $\mathbb{C}(\mathbb{Z}_2)$ are not only dual but also isomorphic. More complicated algebras with duals do not enjoy such an isomorphism; if an algebra is semi-simple the Fourier isomorphism then becomes a Peter-Weyl-like isomorphism [PW27], defined by the representation theory of the algebra.

As a consequence, if we let

$$\delta_e = \frac{1}{\sqrt{2}}(e+g); \quad \delta_g = \frac{1}{\sqrt{2}}(e-g)$$

and consider the morphisms from each algebra, we have, say, the comultiplication of $\mathbb{C}\mathbb{Z}_2$ – but by the Fourier isomorphism this is the same as comultiplication of $\mathbb{C}(\mathbb{Z}_2)$ in the other basis. We would like to have expand our selection of morphisms available. To this end, we instead set

$$|0\rangle = \delta_e = e; \quad |1\rangle = \delta_g = g$$

i.e. use the isomorphism of vector spaces (but not of algebras) to put $\mathbb{C}\mathbb{Z}_2$ and $\mathbb{C}(\mathbb{Z}_2)$ on the same qubit space, with the same basis. For comultiplication we then have

$$\Delta_{\mathbb{C}\mathbb{Z}_2} : |i\rangle \mapsto |i\rangle \otimes |i\rangle; \quad \Delta_{\mathbb{C}(\mathbb{Z}_2)} : |i\rangle \mapsto \sum_j |j\rangle \otimes |i-j\rangle$$

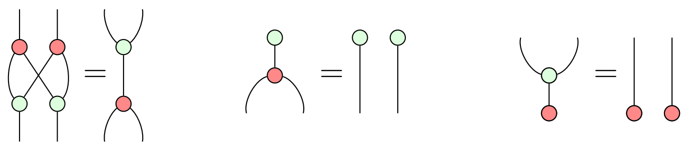
which are different. Of course, if we consider the X basis $\{|+\rangle, |-\rangle\}$, then the comultiplications swap, and $|i\rangle \mapsto |i\rangle \otimes |i\rangle$ gives

$$|+\rangle \mapsto \frac{1}{\sqrt{2}}(|+\rangle \otimes |+\rangle + |-\rangle \otimes |-\rangle)$$

$$|-\rangle \mapsto \frac{1}{\sqrt{2}}(|+\rangle \otimes |-\rangle + |-\rangle \otimes |+\rangle),$$

the comultiplication of $\mathbb{C}(\mathbb{Z}_2)$, where the factor of $\frac{1}{\sqrt{2}}$ comes from the Fourier transform.

We now have access to all the Hopf algebra morphisms from both algebras. When viewed as diagrams the algebraic structure entails *rewrite rules*, whereby a diagram can be rewritten by using, say, $\Delta(ab) = \Delta(a)\Delta(b)$ (a bialgebra rule). The bialgebra rules would then be expressed as



In addition to those coming from Hopf algebras, it transpires that all finite-dimensional Hopf algebras including $\mathbb{C}\mathbb{Z}_2$ and $\mathbb{C}(\mathbb{Z}_2)$ are automatically Frobenius algebras [Par71], so those rewrite rules are included as well. We do not include all of the rewrite rules here, but see [Wet12]. Substantial effort has gone into proving that the ZX-calculus is [Bac16, PSW24]:

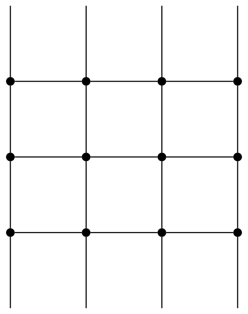
- sound – every valid rewrite on a ZX-diagram using the calculus is an equality in terms of linear maps between qubits,
- universal – every linear map between qubits has a presentation as a ZX-diagram, and

- complete – every equality between different diagrammatic presentations of the same linear map can be found by valid rewrites.

Universality is only possible given the phase group; with only the morphisms given in Table 1.1, and even with the addition of other Pauli basis elements of $\mathbb{C}^2$, one cannot represent every linear map between qubits.

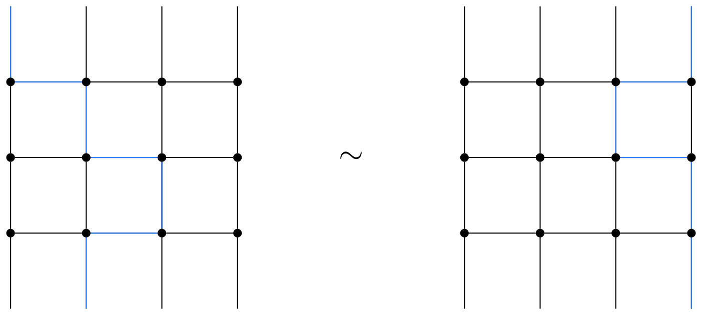
1.6 Lattice surgery

Having taken a detour through various aspects of algebra, we now return to quantum error-correction. A famous family of quantum stabiliser codes is *surface codes* [FMMC12]. Such codes are defined by tessellating surfaces. A *patch* of surface code has the following presentation[2]:

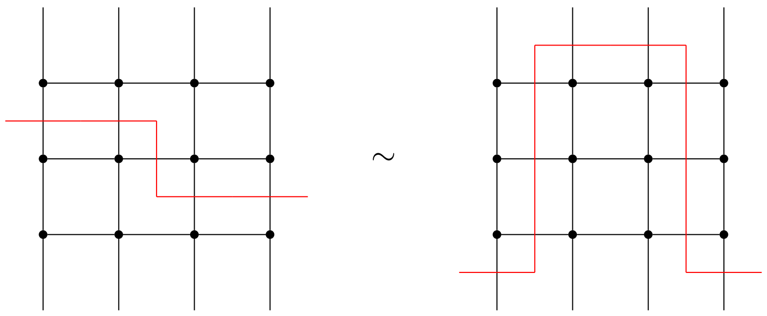


To each edge we associate a qubit. To each interior vertex – that is, vertex not on a boundary – we associated a stabiliser generator, of the form $X \otimes X \otimes X \otimes X$, with support on its incident edges. To each interior face we also associate a stabiliser generator, of the form $Z \otimes Z \otimes Z \otimes Z$, with support on its incident edges. To boundary vertices and faces, on the left, right, top and bottom, we associate generators of the form $X \otimes X \otimes X$ and $Z \otimes Z \otimes Z$ respectively. One can check that these all commute.

Logical operators take the form of 'strings' extending from top to bottom ($\overline{Z}$), passing through vertices, or left to right ($\overline{X}$), passing through faces. In particular, any string which extends from top to bottom is equivalent, by applying Z stabilisers, to any other string which extends from top to bottom, assuming the two strings each touch the top and bottom only once. The same is true for strings from left to right. We illustrate both cases below,



[2]This is the 'unrotated' surface code. The 'rotated' surface code has a slightly different definition [BM-D07B].



where blue lines are Z Paulis and red lines X Paulis.

We assert that these are the only logical Pauli operators available, and the $\overline{X}$ operators anticommute with the $\overline{Z}$ operators. As a consequence, the code has $k = 1$, that is one logical qubit. The shortest length string which extends from left to right or top to bottom is length 4, and so this code has distance $d = 4$. There are 25 edges, so the code has parameters $[\![25, 1, 4]\!]$.

Now, readers familiar with cell and chain complexes may recognise that, as generators made of only Xs are assigned to edges incident to vertices, and the same for Zs but for faces, the commutation rules imply that the code can be described by a chain complex over $\mathbb{F}_2$. This is a well-known insight [BM-D07A, BE21B] and there is a much wider class of codes which do not have to be defined on lattices, but can be similarly described using homology. These codes will be the focus of Part A.

Alternatively, readers familiar with Hopf algebras may see that the X-type Paulis can be seen as an action of $\mathbb{C}\mathbb{Z}_2$. Specifically we have $\triangleright : \mathbb{C}\mathbb{Z}_2 \otimes (\mathbb{C}\mathbb{Z}_2)^{\otimes 4} \to (\mathbb{C}\mathbb{Z}_2)^{\otimes 4}$, identifying $\mathbb{C}^2 \cong \mathbb{C}\mathbb{Z}_2$ as vector spaces, sending $|0\rangle \mapsto e$, $|1\rangle \mapsto g$, where g is the non-identity element in $\mathbb{Z}_2$. The action is then

$$g\triangleright(h^1 \otimes h^2 \otimes h^3 \otimes h^4) = gh^1 \otimes gh^2 \otimes gh^3 \otimes gh^4,$$

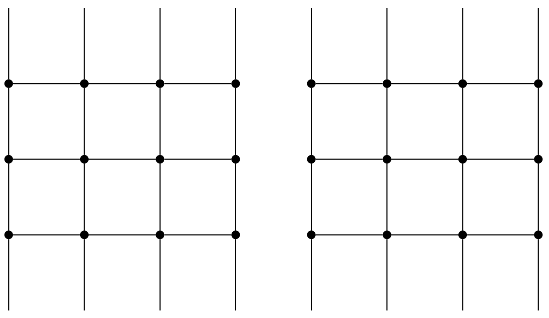
that is a tensor product of regular representations, on the edges incident to a vertex. We have similar for the Z-type Paulis on edges incident to faces, but where the action is of the element $\delta_e - \delta_g \in \mathbb{C}(\mathbb{Z}_2)$, as a consequence of the Fourier isomorphism. It turns out that in addition the logical space satisfies

$$L \cong \mathbb{C}^2 \cong \mathbb{C}\mathbb{Z}_2 \cong \mathbb{C}(\mathbb{Z}_2)$$

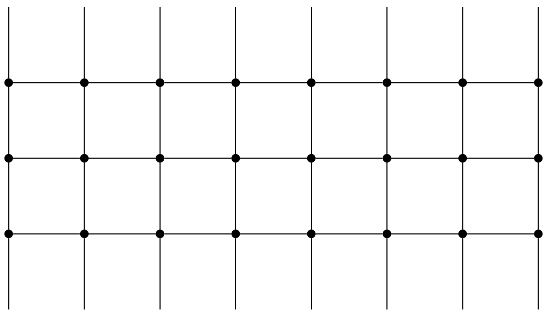
as vector spaces. It is equally well-known that this instance belongs to a much wider class of codes which are defined on lattices, but use actions of more interesting algebras [Kit03, Meu17]. These in turn are the focus of Part B.

Before we discuss lattice surgery, we assert that patches of surface code may be prepared in the logical $|+\rangle$ or $|0\rangle$ states by initialising the data qubits in that same state and then measuring all stabilisers. This procedure is fault-tolerant [FMMC12]. Ditto for single-qubit logical measurements, where instead one measures out all qubits in the X or Z basis. One can check that these are correct by considering strings across the lattice.

To perform lattice surgery as per [HFDM12], we either take a patch and split it into two, or take two patches and merge them into one. A $\overline{Z}$-merge (or 'smooth' merge) is as follows. Take two adjacent patches of surface code,

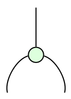


then initialise a new intermediate section between them,



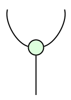
where all new qubits (edges) are initialised in the $|+\rangle$ state. Then immediately being measuring the stabilisers, including all the new faces. Despite only initialising new qubits and stabilisers (and modifying some X stabilisers), this performs a $\overline{Z} \otimes \overline{Z}$ measurement on the logical qubits, quotienting down to one logical qubit afterwards. The logical measurement outcome is dictated by the parity of the new Z stabilisers.

One can reason about the logical operation performed by considering the blue strings from top to bottom: previously, they belonged to two separate equivalence classes, but afterwards they belong to the same one. The $+1$ and -1 logical measurement outcomes can be computed straightforwardly, and are dependent on the *parity* of the measurement outcomes of the new Z stabilisers corresponding to new faces. From the perspective of homology, we have taken an *injection* of chain complexes into the new chain complex, but a *surjection* on the homology space at degree 1. From the perspective of Hopf algebras, we have only added new copies of $\mathbb{C}\mathbb{Z}_2$ with appropriate representations, but in the $+1$ outcome case yielded a logical operation of the form



on the two initial logical qubits [dBH20], that is the map $|i\rangle \otimes |j\rangle \mapsto \delta_{i,j}|i\rangle$. This is multiplication in $\mathbb{C}(\mathbb{Z}_2)$.

A ‘smooth’ split is the adjoint of this procedure. Given one initial patch we measure out a layer of edges, from top to bottom, and split the patch in twain. This yields the logical operation

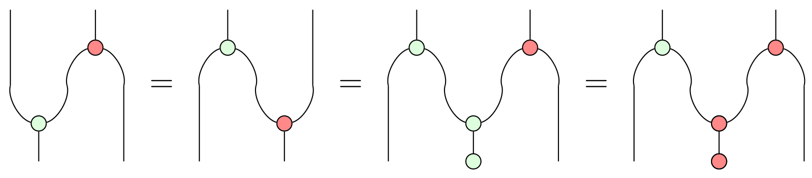


that is $|i\rangle \mapsto |i\rangle \otimes |i\rangle$. This is performing a *surjection* of chain complexes, but an *injection* on the homology space. It is also the comultiplication of $\mathbb{C}\mathbb{Z}_2$. Similar rules dictate $\overline{X}$-merges (‘rough’ merges) and splits.

One can use these operations to build up unitary gates, such as the CNOT, which has the map on basis states

$$|i\rangle \otimes |j\rangle \mapsto |i\rangle \otimes |i+j\rangle$$

by observing that



for example, where the first two diagrams are well-known to be equal to a CNOT [dBH20]. Alternatively, one can use merges to inject magic states from patches of surface code upon which magic states have been *distilled* or *cultivated* [BK05, GSJ24].

For a more in-depth look at this simple case of lattice surgery but for d-dimensional qudits see Chapter 5. There are many more Pauli measurements one can perform using surgery, for which see [Lit19, CKBB22], but we focus on those with a nice algebraic correspondence.

Part A

Homological codes

Chapter 2

CSS code surgery as a universal construction

2.1 Introduction

In this Chapter we construct generalisations of lattice surgery to Calderbank-Shor-Steane (CSS) codes. There are several equivalent ways of defining CSS codes, but for our purposes we shall describe them as codes which are all *homological* in a suitable sense [BE21B, BM-D07A].

This means that we can study CSS codes using the tools of homological algebra [Wei84]. This approach has recently seen much success, for example in the construction of so-called good quantum low-density parity check (LDPC) code families using a lifted product of chain complexes [PK22A]. Such code families have an encoding rate k/n of logical to physical qubits which is constant in the code size, while maintaining a linear code distance d, a substantial asymptotic improvement over simpler examples such as the toric code. The main caveat is, informally, that the connectivity between physical qubits is non-local. This complicates the architecture of the system, and also complicates the protocols for performing logical gates.

There have been several recent works on protocols for logical gates in CSS codes [KP21, CKBB22, BB24, QWV23, HJY23, ZSP23], of varying generality. Here, we build on this work by defining surgery, in the abstract, using arbitrary CSS codes which form a categorical span, although the practical implementation of such surgery has several important caveats. The idea is that merging two codes works by identifying a common structure in each code and quotienting it out. CSS code surgery is particularly convenient when the CSS codes are *compatible*, in the sense that they have at least one identical $\overline{Z}$ or $\overline{X}$ logical operator. In this case, the common structure being quotiented out is the logical operator. In order to formalise this, we use the category of chain complexes $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$.

Remark 2.1.1. The protocols defined in [CKBB22] are also generalisations of surgery to CSS codes. Here, our perspective is more algebraic, which we will see can allow us to prove certain things in a very general manner. Our surgeries are also different, and are in a certain sense a more direct generalisation of the lattice surgery defined in [HFDM12]. On the other hand, the constructions of [CKBB22] are somewhat more general, in that they allow arbitrary Pauli product measurements, in principle. For example, one can measure the operator $\overline{X} \otimes \overline{Y} \otimes \overline{Z}$. Such arbitrary Pauli product measurements take us out of CSS codes, and so cannot be described using the homological formalism.

Remark 2.1.2. Since this Chapter and its corresponding paper [CB24] was published, a series of preprints studying the same topic have appeared on the arXiv. First, Ref. [CHRY24] showed that the qubit overhead can be reduced greatly by gauge-fixing and considering the expansion properties of logical operators. Next, Ref. [ZL24] gave a construction for parallelising the scheme of [CKBB22] by 'branching' and taking more subtle measurement 'stickers'. Lastly, Refs. [IGND24, WY24] further developed the use of expander graphs for the ancilla patches, vastly reducing overhead for single-qubit measurements. In thesis we do not make use of expanders or expansion properties of graphs.

We start by giving a recap of classical linear binary codes and qubit CSS codes using chain complexes. We then define code maps between CSS codes using morphisms between chain complexes. These are maps which send X-checks to X-checks and Z-checks to Z-checks in a coherent way, and have a convenient presentation as phase-free ZX diagrams, which we prove in Proposition 2.2.12.

We believe that code maps crop up throughout the CSS code literature. We see 3 primary use-cases for code maps:

1. Encoders/decoders [D-CP10, Del14, HWH21].
2. Constructing new codes.
3. Designing fault-tolerant logical operations [HJY23].

We define CSS code merges as a colimit – specifically, a coequaliser/pushout – in the category of chain complexes. Not only does the construction describe a surgery operation, but it also gives a general recipe for new codes. An application of our treatment is the description of certain classes of code surgery whereby the codes are merged or split along a $\overline{Z}$ or $\overline{X}$ operator. This is closely related to the notion of 'welding' in [Mic14], and generalises the cases for 2D topological codes given in [HFDM12, NFB17][1]. We prove that merging two LDPC codes in such a manner still yields an LDPC code. We give a series of examples, including the specific case of lattice surgery between surface codes. Lastly, we discuss how to apply such protocols in practice. We prove that when two technical conditions are satisfied then code surgery can be performed while maintaining the error-correcting properties of the code, allowing us to perform logical parity measurements on codes.

2.2 Quantum codes

Here we introduce classes of both classical and quantum codes as chain complexes. We give easy examples such as the surface and toric codes. Up until Section 2.2.3, this part is also well-known, although we describe the relationship between Z and X operators in greater detail than we have found elsewhere.

2.2.1 Codes as chain complexes

Binary linear classical codes which encode k bits using n bits can be described by a $m \times n$ parity check $\mathbb{F}_2$-matrix P. The parity check matrix P, when applied to any codeword of length n, gives $Pc = 0$, and thus $k = \dim\ker(P)$; if the result is non-zero then an error has been detected, and under certain assumptions can be corrected. The distance d of a binary linear classical code is the minimum Hamming weight of its nonzero codewords, and one

[1]It is, however, different from surgery with colour codes [LR-A14] and rotated surface codes [BM-D07B].

characterisation of codes is by the parameters $[n, k, d]$. We may trivially view a binary linear classical code as a length 1 chain complex, with indices chosen for convenience:

$$C_\bullet = C_1 \xrightarrow{\partial_1} C_0$$

where $C_1 = \mathbb{F}_2^n$, $C_0 = \mathbb{F}_2^m$, and $\partial_1 = P$, the chosen $m \times n$ parity check matrix. Then we have $k = \dim H_1(C_\bullet) = \dim Z_1(C_\bullet)$, where $Z_1(C_\bullet)$ is the codespace.

Example 2.2.1. Let $C_\bullet$ be a $[3, 1, 3]$ repetition code, encoding 1 bit into 3 bits. In this case, let

$$P = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

Example 2.2.2. Let $C_\bullet$ be the $[7, 4, 3]$ Hamming code. Then let

$$P = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

We now move on to quantum codes. Qubit Calderbank-Shor-Steane (CSS) codes are a type of stabiliser quantum code [CS96], for which see Definition 1.2.5. Let $\mathscr{P}_n = \mathscr{P}^{\otimes n}$ be the Pauli group over n qubits. Stabiliser codes start by specifying an Abelian subgroup $\mathscr{S} \subset \mathscr{P}_n$, called a stabiliser subgroup, such that the codespace $\mathscr{H}$ is the mutual $+1$ eigenspace of all operators in $\mathscr{S}$. That is,

$$U |\psi\rangle = |\psi\rangle \quad \forall U \in \mathscr{S}, |\psi\rangle \in \mathscr{H}$$

We then specify a generating set of $\mathscr{S}$, of size m. For CSS codes, this generating set has as elements tensor product strings of either $\{I, X\}$ or $\{I, Z\}$ Pauli terms, with no scalars other than 1. One can define two parity check $\mathbb{F}_2$-matrices P_X, P_Z, for the Xs and Zs, which together define a particular code. Each column in P_X and P_Z represents a physical qubit, and each row a measurement/stabiliser generator. P_X and P_Z thus map Z and X operators on physical qubits respectively to sets of measurement outcomes, with a 1 outcome if the operators anticommute with a given stabiliser generator, and 0 otherwise; these outcomes are also called *syndromes*. P_X is a $m_X \times n$ matrix, and P_Z is $m_Z \times n$, with m_X, m_Z marking the division of the generating set into Xs and Zs respectively, satisfying $m = m_X + m_Z$. We do not require the generating set to be minimal, and hence P_X and P_Z need not be full rank.

Definition 2.2.3. We say that w^Z is the maximal weight of all Z-type generators and w^X the same for the X-type generators. These are the highest weight rows of P_Z and P_X respectively. Similarly, we say that q^Z, q^X is the maximal number of Z, X generators sharing a single qubit. These are the highest weight columns of P_Z and P_X.

CSS codes are described by parameters $[\![n, k, d]\!]$, with k the number of encoded qubits and d the code distance, which we define presently.

That the stabilisers must commute is equivalent to the requirement that $P_X P_Z^\intercal = P_Z P_X^\intercal = 0$. We may therefore view these matrices as differentials in a length 2 chain complex:

$$C_\bullet = C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$$

where $\partial_2 = P_Z^\intercal$ and $\partial_1 = P_X$, or the other way round ($\partial_2 = P_X^\intercal, \partial_1 = P_Z$) if desired, but we use the former for consistency with the literature. The quantum code then has $C_1 = \mathbb{F}_2^n$, and thus:

$$C_\bullet = \mathbb{F}_2^{m_Z} \xrightarrow{P_Z^\intercal} \mathbb{F}_2^n \xrightarrow{P_X} \mathbb{F}_2^{m_X}$$

The code also has $k = \dim\ H_1(C_\bullet)$. To see this, observe first that C_1 represents the space of Z Paulis on the set of physical qubits, with a vector being a Pauli string e.g. $v = \begin{pmatrix}1 & 0 & 1\end{pmatrix}^\intercal \rightsquigarrow Z \otimes I \otimes Z$. Each vector in $H_1(C_\bullet)$ can be interpreted as an equivalence class $[v]$ of Z operators on the set of physical qubits, modulo Z operators which arise as Z stabilisers. That this vector is in $Z_1(C_\bullet)$ means that the Z operators commute with all X stabilisers, and when the vector is not in $[0] = B_1(C_\bullet)$ it means that the Z operators act nontrivially on the logical space. A basis of $H_1(C_\bullet)$ constitutes a choice of individual logical Paulis $\overline{Z}$, that is a tensor product decomposition of the space of logical Z operators, and we set $\overline{Z}_1 = \overline{Z} \otimes \overline{I} \cdots \otimes \overline{I}$ on *logical* qubits, $\overline{Z}_2 = \overline{I} \otimes \overline{Z} \cdots \otimes \overline{I}$ etc. There is a logical qubit for every logical Z, hence $k = \dim H_1(C_\bullet)$.

To get the logical X operators, consider the cochain complex $C^\bullet$. The vectors in $H^1(C^\bullet)$ then correspond to $\overline{X}$ operators in the same manner. As $H_i(C_\bullet) \cong H^i(C^\bullet)$ there must be an $\overline{X}$ operator for every $\overline{Z}$ operator and vice versa.

Lemma 2.2.4. A choice of basis $\{[v]_i\}_{i \leq k}$ for $H_1(C_\bullet)$ implies a choice of basis $\{[w]_j\}_{j \leq k}$ for $H^1(C^\bullet)$.

Proof. First, recall that we have the nondegenerate bilinear form

$$\cdot : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2; \quad u \cdot v = u^\intercal v$$

which is equivalent to $\cdot : C_1 \times C^1 \to \mathbb{F}_2$; computationally, this tells us whether a Z operator commutes or anticommutes with an X operator. Now, let $u \in Z_1(C_\bullet)$ be a (possibly trivial) logical Z operator, and $v \in B^1(C^\bullet)$ be a product of X stabilisers. Then $P_X u = 0$, and $v = P_X^\intercal w$ for some $w \in C^2$. Thus $u \cdot v = u^\intercal v = u^\intercal P_X^\intercal w = (P_X u)^\intercal w = 0$, and so products of X stabilisers commute with logical Z operators. The same applies for Z stabilisers and logical X operators.

As a consequence, $v \cdot w = (v + s) \cdot (w + t)$ for any $v \in Z_1(C_\bullet)$, $w \in Z^1(C^\bullet)$, $s \in B_1(C_\bullet)$, $t \in B^1(C^\bullet)$, and so we may define $[v] \cdot [w] = v \cdot w$ for any $[v] \in H_1(C_\bullet)$, $[w] \in H^1(C^\bullet)$ with representatives v, w. The duality pairing of C_1, C^1 thus lifts to $H_1(C_\bullet), H^1(C^\bullet)$, and a choice of basis $\{[v]_i\}_{i \leq k}$ for $H_1(C_\bullet)$ implies a choice of basis of $H^1(C^\bullet)$, determined uniquely by $[v]_i \cdot [w]_j = \delta_{i,j}$. ■

The above lemma ensures that picking a tensor product decomposition of logical Z operators also entails the same tensor product decomposition of logical X operators, so that $\overline{X}_i \overline{Z}_j = (-1)^{\delta_{i,j}} \overline{Z}_j \overline{X}_i$, for operators on the ith and jth logical qubits.

Let

$$d^Z = \min_{v \in Z_1(C_\bullet) \setminus B_1(C_\bullet)} |v|; \quad d^X = \min_{w \in Z^1(C^\bullet) \setminus B^1(C^\bullet)} |w|$$

where $|\cdot|$ is the Hamming weight of a vector, then the code distance $d = \min(d^Z, d^X)$. d^Z and d^X are called the systolic and cosystolic distances, and represent the lowest weight nontrivial Z and X logical operators respectively.

As all the data required for a CSS code is contained within the chain complex $C_\bullet$ – and potentially a choice of basis of $H_1(C_\bullet)$ – we could define a CSS code as just the single chain complex, but it will be convenient to have direct access to the cochain complex as well.

Definition 2.2.5. A CSS code is a pair $(C_\bullet, C^\bullet)$, with $C_\bullet$ a length 2 chain complex centred at degree 1, so we have:

$$C_\bullet = C_2 \xrightarrow{P_Z^\intercal} C_1 \xrightarrow{P_X} C_0 \ ; \qquad C^\bullet = C^0 \xrightarrow{P_X^\intercal} C^1 \xrightarrow{P_Z} C^2$$

We call the first of the pair the Z-type complex, as vectors in C_1 correspond to Z-operators, and the second the X-type complex. A *based* CSS code additionally has a choice of basis for $H_1(C_\bullet)$, and hence for $H^1(C^\bullet)$.

Employing the direct sum $(C \oplus D)_\bullet$ of chain complexes we have the CSS code $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$, which means the CSS codes $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ perform in parallel on disjoint sets of qubits, without any interaction. The Z and X operators will then be the tensor product of operators in each.

In summary, there is a bijection between length 1 chain complexes in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ and binary linear classical codes, and between length 2 chain complexes in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ and CSS codes. There are CSS codes for higher dimensional qudits, but for simplicity we stick to qubits.

Rather than just individual codes we tend to be interested in families of codes, where n, k, d scale with the size of code in the family. Of particular practical interest are quantum *low density parity check* (LDPC) CSS codes, which are families of codes where all w^Z, w^X, q^Z and q^X in the family are bounded from above by a constant. Equivalently, this means the Hamming weight of each column and row in each differential is bounded by that constant.

Definition 2.2.6. A subsystem CSS code is a CSS code where some of the logical qubits are not used for storing logical data. These qubits are instead called *gauge qubits* [KLP05]. In this case the first homology space divides into $H_1(C_\bullet) = \mathcal{L} \oplus \mathcal{G}$, with $\mathcal{L}$ being the space used for storing data and $\mathcal{G}$ the space of gauge qubits.

Importantly, we can at times relegate logical qubits to be gauge qubits instead in order to increase the distance of the code. The nontrivial $\overline{Z}$ logicals which act on the logical data must have some support in $\mathcal{L}$, not just $\mathcal{G}$. These belong to the set $H_1(C_\bullet)\backslash\mathcal{G}$ and are called dressed logical operators. Dressed logicals can still have support in $\mathcal{G}$. The same applies to $H^1(C^\bullet)$.

The distance d of a subsystem CSS code is therefore the smallest weight of the dressed logical operators.

2.2.2 Basic quantum codes

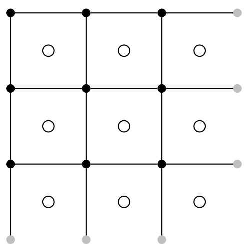
Example 2.2.7. Let $(C_\bullet, C^\bullet)$ be the $[\![9,1,3]\!]$ Shor code, so we have $C_2 = \mathbb{F}_2^2$, $C_1 = \mathbb{F}_2^9$, $C_0 = \mathbb{F}_2^6$. The parity check matrices are given by

$$P_X = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}; \quad P_Z = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

We then have $\dim Z_1(C_\bullet) = \dim C_1 - \mathrm{rank}(P_X) = 9 - 2 = 7$ and $\dim B_1(C_\bullet) = \mathrm{rank}(P_Z^\intercal) = 6$. Thus $k = \dim H_1(C_\bullet) = 1$. There is a single nonzero equivalence class $[v] \in H_1(C_\bullet)$, with a representative $v = \begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\end{pmatrix}^\intercal$. Similarly there is the nonzero vector $w = \begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\end{pmatrix}^\intercal$, which is a representative of $[w] \in H_1(C_\bullet)$. Hence, we have two logical operators $\overline{Z} = \bigotimes_i^9 Z_i$, $\overline{X} = \bigotimes_i^9 X_i$ with Z_i on the ith qubit and the same for X_i. We equally have, say, $\overline{Z} = Z_1 \otimes Z_4 \otimes Z_7$ and $\overline{X} = X_1 \otimes X_2 \otimes X_3$ in the same equivalence classes as those above, $[v]$ and $[w]$.

We now consider two examples which come from square lattices. This can be done much more generally. In Appendix 1 we formalise categorically the procedure of acquiring chain complexes – and therefore CSS codes – from square lattices, which are a certain type of cell complex.

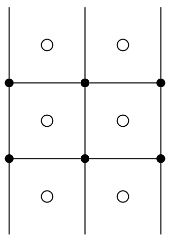
Example 2.2.8. Consider the following square lattice:



Edges in the lattice are qubits, so $n = 18$, the 9 X-checks are associated with vertices and the 9 Z-checks are associated with faces, which are indicated by white circles. Grey vertices indicate periodic boundary conditions, so the lattice can be embedded on a torus. This is an instance of the standard toric code [Kit03].

The abstracted categorical homology from before is now the homology of the tessellated torus, with cycles, boundaries etc. having their usual meanings. $k = \dim H_1(C_\bullet) = 2$, and (co)systolic distances are the lengths of the essential cycles of the torus.

Example 2.2.9. Now consider a different square lattice:



This represents a patch of surface code $(D_\bullet, D^\bullet)$, where we have two smooth sides, on the left and right, and two rough sides to the patch, on the top and bottom. There are 'dangling' edges at the top and bottom, which do not terminate at vertices. We have

$$\dim D_2 = \dim D_0 = 6; \quad n = \dim D_1 = 13; \quad k = \dim H_1(D_\bullet) = 1$$

The systolic distance is 3, the length of the shortest path from the top to bottom boundary, and the cosystolic distance 3, the same but from left to right.

2.2.3 Code maps

One may wish to convert one code into another, making a series of changes to the set of stabiliser generators to be measured, and potentially also to the physical qubits. The motivation behind such protocols is typically to perform logical operations which are not available natively to the code; not only might the target code have other logical operations, but the protocol is itself a map between logical spaces when chosen carefully. An example of a change to the measurements and qubits is code deformation. We do not formalise code deformation here, as that has some specific connotations [VLC19]. Instead we define a related notion, called a *code map*, which has some overlap. A code map is also related to, but not the same as, the 'homomorphic gadgets' from [HJY23].

Definition 2.2.10. A $\overline{Z}$-preserving code map $\mathcal{F}_{\overline{Z}}$ from a CSS code $(C_\bullet, C^\bullet)$ to $(D_\bullet, D^\bullet)$ is a paired chain map and cochain map $(f_\bullet, f^\bullet)$, for $f_\bullet : C_\bullet \to D_\bullet$ and $f^\bullet : D^\bullet \to C^\bullet$.

$$\begin{tikzcd} {} \arrow[d, Rightarrow, "\mathcal{F}_{\overline{Z}}"'] & (C_\bullet, C^\bullet) \arrow[d, shift right, "f_\bullet"'] \\ {} & (D_\bullet, D^\bullet) \arrow[u, shift right, "f^\bullet"'] \end{tikzcd}$$

Note that the cochain map is strictly speaking redundant, as all the data is contained in a single chain map $f_\bullet$, but as with CSS codes it will be handy to keep both around.

Let us unpack this definition. $\mathcal{F}_{\overline{Z}}$ first maps Z-operators in C_1 to Z-operators in D_1, using f_1. It may map a single Z on a qubit to a tensor product of Zs, or to I. It then has a map f_2 on Z generators, and another f_0 on X checks. Recalling Definition 1.4.3, we have:

$$\begin{tikzcd} C_2 \arrow[r, "\partial_2^{C_\bullet}"] \arrow[d, "f_2"'] \arrow[dr, phantom, "\mathrm{I}"] & C_1 \arrow[r, "\partial_1^{C_\bullet}"] \arrow[d, "f_1"] \arrow[dr, phantom, "\mathrm{II}"] & C_0 \arrow[d, "f_0"] \\ D_2 \arrow[r, "\partial_2^{D_\bullet}"'] & D_1 \arrow[r, "\partial_1^{D_\bullet}"'] & D_0 \end{tikzcd}$$

With two commuting squares labelled I and II. I stipulates that applying products of Z stabiliser generators on the code and then performing the code map should be equivalent to performing the code map and then applying products of Z stabiliser generators, i.e. $f_1 \circ \partial_2^{C_\bullet} = \partial_2^{D_\bullet} \circ f_2$. II stipulates that performing the X measurements and then mapping the code should be equivalent to mapping the code and then performing X measurements, so there is a consistent mapping between all measurement outcomes, i.e. $f_0 \circ \partial_1^{C_\bullet} = \partial_1^{D_\bullet} \circ f_1$.

Then there is the cochain map $f^\bullet$. This has the component $f^1 = f_1^\intercal : D^1 \to C^1$, which maps an X-operator in D^1 back to an X-operator in C^1. Similarly for f^0 and $f_2^\intercal$, each of which come with commuting squares which are just the transposed conditions of those in $f_\bullet$, so they say nothing new. This is not surprising, as all the data for $f^\bullet$ is given by $f_\bullet$ already.

We now show that this definition entails some elementary properties. For a start, Lemma 1.4.4 implies that a code map gives a map from a $\overline{Z}$ operator in $H_1(C_\bullet)$ to $\overline{Z}$s in $H_1(D_\bullet)$; this can also map to a product of logical $\overline{Z}$s, and in particular map $\overline{Z}$ to zero i.e. $\overline{I}$, but it must not map a $\overline{Z}$ to an operator which can be detected by the X stabiliser measurements. Hence $(f_\bullet, f^\bullet)$ preserves the fact that any $\overline{Z}$ is an undetectable operator on the codespace. A similar requirement holds for $\overline{X}$ operators, but this time the condition is inverted. Every $\overline{X}$ in $H^1(D^\bullet)$ must have a map only to logical operators in $H^1(C^\bullet)$, but the other way is not guaranteed.

Let n_C and n_D be the number of physical qubits in codes $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ respectively. We may interpret $\mathcal{F}_{\overline{Z}}$ as a $\mathbb{C}$-linear map M in `FHilb`, the category of Hilbert spaces. This $\mathbb{C}$-linear map has the property that $MU_Z = U'_Z M$, where U_Z is a tensor product of Z Paulis on n_C qubits and U'_Z is a tensor product of Z Paulis on n_D qubits.

In particular, given any U_Z we have a specified U'_Z. The same is not true the other way round, as the map f_1 is not necessarily injective or surjective. Similarly, $MU_X = U'_X M$. This time, however, given any unique U'_X on n_D qubits we have a specified U_X but vice versa is not guaranteed, depending on $f_1^\intercal$.

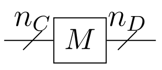
As a consequence, the linear map M is *stabiliser*, in the sense that it maps Paulis to Paulis, but not *unitary* in general. M is unitary iff f_1 is invertible.

If M is not even an isometry, it cannot be performed deterministically, and the code map must include measurements on physical qubits. There will in general be Kraus operators corresponding to different measurement outcomes which will determine whether the code map has been implemented as desired; for now we assume that M is performed deterministically, and leave this complication for Section 2.4. Similarly, while the code map can be interpreted as a circuit between two codes, we do not claim that such a circuit can be performed fault-tolerantly in general.

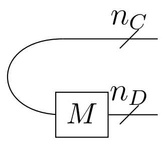
Remark 2.2.11. For the following proposition, and at various points throughout the rest of this Chapter, we will use the ZX-calculus, a formal graphical language for reasoning about computation with qubits. See Section 1.5 for a short introduction, or see Sections 1-3 of [Wet12]. Our use of ZX diagrams is unsophisticated, and primarily for convenience.

Proposition 2.2.12. Let $\mathcal{F}_{\overline{Z}}$ be a $\overline{Z}$-preserving code map between codes $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ with qubit counts n_C and n_D. The interpretation of $\mathcal{F}_{\overline{Z}}$ as a $\mathbb{C}$-linear map M in `FHilb` has a presentation as a circuit with gates drawn from $\{\mathrm{CNOT}, |+\rangle, \langle 0|\}$.

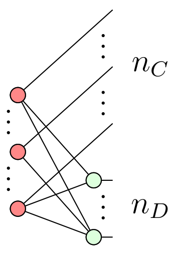
Proof. We start with the linear map $M : (\mathbb{C}^2)^{\otimes n_C} \to (\mathbb{C}^2)^{\otimes n_D}$:



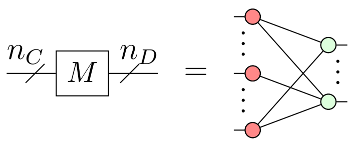
By employing the partial transpose in the computational basis we convert it into the state

$|\psi\rangle =$ 

i.e. inserting n_C Bell pairs. By the definition of f_1 we know that this has an independent stabiliser, with one Z and $n_C - 1$ Is followed by some n_D-fold tensor product of Z and I, for each of the n_C qubits. From $f_1^\intercal$ it also has an independent stabiliser, with some n_C-fold tensor product of X and I followed by $n_D - 1$ Is and one X, for each of the n_D qubits. $|\psi\rangle$ is therefore a stabiliser state. Further, from Theorem 5.1 of [Kis22] it has a presentation as a 'phase-free ZX diagram', of the form

$|\psi\rangle =$ 

where the top n_C qubits do not have a green spider. We perform the partial transpose again to convert the state $|\psi\rangle$ back into the map M, which has the form



Any ZX diagram of this form can be expressed as a matrix over $\mathbb{F}_2$, mapping X-basis states from $(\mathbb{C}^2)^{\otimes n_C}$ to $(\mathbb{C}^2)^{\otimes n_D}$. The example above, ignoring the ellipses, has the matrix

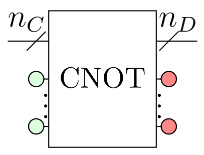
$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

which is equal to f_1; the point of the above rigmarole is thus to say that f_1 is precisely a linear map between X-basis states, which one can check easily. One can explicitly calculate M as a matrix in the X-basis in `FHilb`. For the first column, we compute $f_1 \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, so $M \left|+\right\rangle \left|+\right\rangle \left|+\right\rangle = \left|+\right\rangle \left|+\right\rangle$. Overall, we have

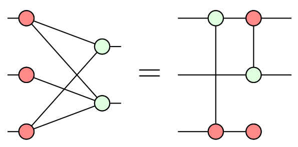
$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

which again is in the X-basis, not the (computational) Z-basis.

Returning to f_1, we can perform Gaussian elimination, performing row operations, which produce CNOTs on the r.h.s. of the diagram in the manner of [KM-v20], until the matrix is in reduced row echelon form. We then perform column operations producing CNOTs on the l.h.s. of the diagram, until the matrix has at most one 1 in each row and column. This can be performed using the leading coefficients to remove all other 1s in that row. The final matrix just represents a permutation of qubits with some states and effects. An empty column corresponds to a $\langle 0|$ effect, and an empty row a $\left|+\right\rangle$ state. We thus end up with a presentation of M in the form



On our example, this is then



which one can check maps $Z \otimes I \otimes I \mapsto Z \otimes Z$ etc. ■

As a consequence $\bar{M} = M$, i.e. the conjugate of M is just M.

Corollary 2.2.13. If $n_C = 0$ then the map M is actually a stabiliser state of the form $M = \left|+\right\rangle^{\otimes n_D}$. When $n_D = 0$ then $M = \langle 0|^{\otimes n_C}$.

Proof. When $n_C = 0$ we see that M has exactly n_D independent stabilisers with 1 X and $n_D - 1$ Is, for each qubit to put X on. The flipped argument applies when $n_D = 0$. ■

Definition 2.2.14. An $\overline{X}$-preserving code map $\mathcal{F}_{\overline{X}}$ from a CSS code $(D_\bullet, D^\bullet)$ to $(C_\bullet, C^\bullet)$ is a paired chain map and cochain map $(f_\bullet, f^\bullet)$, for $f_\bullet : C_\bullet \to D_\bullet$ and $f^\bullet : D^\bullet \to C^\bullet$.

$$\begin{tikzcd} \bullet \arrow[d, phantom] & (C_\bullet, C^\bullet) \arrow[d, shift right, "f_\bullet"'] \\ \bullet \arrow[u, Rightarrow, "\mathcal{F}_{\overline{X}}"] & (D_\bullet, D^\bullet) \arrow[u, shift right, "f^\bullet"'] \end{tikzcd}$$

So $\mathcal{F}_{\overline{X}}$ is just mapping in the other direction to $\mathcal{F}_{\overline{Z}}$ from before, and we say that $\mathcal{F}_{\overline{X}}$ is *opposite* to $\mathcal{F}_{\overline{Z}}$. In this case, when we interpret $\mathcal{F}_{\overline{X}}$ as a $\mathbb{C}$-linear map L, it has the property that $LU_X = U'_X L$ and that any U_X gives a specified U'_X, and $LU_Z = U'_Z L$, but that any U'_Z gives a specified U_Z but not vice versa.

By inspecting the stabilisers we see that, for $\mathcal{F}_{\overline{Z}}$ with interpretation M and $\mathcal{F}_{\overline{X}}$ with interpretation L, $L = M^\dagger = M^\intercal$.

Corollary 2.2.15. Let $\mathcal{F}_{\overline{X}}$ be an $\overline{X}$-preserving code map between codes $(D_\bullet, D^\bullet)$ and $(C_\bullet, C^\bullet)$ with qubit counts n_D and n_C. The interpretation of $\mathcal{F}_{\overline{X}}$ as a $\mathbb{C}$-linear map L in `FHilb` has a presentation as a circuit with gates drawn from $\{\text{CNOT}, |0\rangle, \langle +|\}$.

Corollary 2.2.16. If $n_D = 0$ then $L = |0\rangle^{\otimes n_C}$, and if $n_C = 0$ then $L = \langle +|^{\otimes n_D}$.

Corollary 2.2.17. The restrictions of $\mathcal{F}_{\overline{Z}}$ and $\mathcal{F}_{\overline{X}}$ to use only $H_1(f_\bullet)$ and $H^1(f^\bullet)$ also have interpretations as $\mathbb{C}$-linear maps on logical qubits in the same way, and Proposition 2.2.12 and Corollary 2.2.15 also apply to such interpretations.

Lemma 2.2.18. Let $\Omega : (\mathbb{C}^2)^{\otimes k_C} \to (\mathbb{C}^2)^{\otimes k_D}$ and $M : (\mathbb{C}^2)^{\otimes n_C} \to (\mathbb{C}^2)^{\otimes n_D}$ be the interpretations of the $\overline{Z}$-preserving maps

$$(H_1(f_\bullet), H^1(f^\bullet)) : (H_1(C_\bullet), H^1(C^\bullet)) \Rightarrow (H_1(D_\bullet), H^1(D^\bullet))$$

and

$$\mathcal{F}_{\overline{Z}} : (C_\bullet, C^\bullet) \Rightarrow (D_\bullet, D^\bullet)$$

in `FHilb` respectively. Recall that there are encoding embeddings for any CSS codes, $E_C : (\mathbb{C}^2)^{\otimes k_C} \hookrightarrow (\mathbb{C}^2)^{\otimes n_C}$ and $E_D : (\mathbb{C}^2)^{\otimes k_D} \hookrightarrow (\mathbb{C}^2)^{\otimes n_D}$. Then there is a commuting square,

$$\begin{tikzcd}
(\mathbb{C}^2)^{\otimes k_C} \arrow[r, hook, "E_C"] \arrow[d, "\Omega"'] & (\mathbb{C}^2)^{\otimes n_C} \arrow[d, "P_{S_D}\circ M"] \\
(\mathbb{C}^2)^{\otimes k_D} \arrow[r, hook, "E_D"] & (\mathbb{C}^2)^{\otimes n_D}
\end{tikzcd}$$

where P_{S_D} is the projector onto the $+1$ eigenspace of the stabilisers of $(D_\bullet, D^\bullet)$.

Proof. Consider an arbitrary logical computational basis state $|\overline{\psi}\rangle \in (\mathbb{C}^2)^{\otimes k_C}$, where $\overline{\psi} \in \{0,1\}^{k_C}$. This is stabilised by k_C $\overline{Z}$ logicals, where the signs are $+$ or $-$ depending on the basis element 0 or 1. Mapping into $(\mathbb{C}^2)^{\otimes n_C}$, $E_C |\overline{\psi}\rangle$ is stabilised by representatives of the same $\overline{Z}$s, with the appropriate signs, as well as S_C, the set of stabilisers in $(C_\bullet, C^\bullet)$.

$\Omega |\overline{\psi}\rangle$ is stabilised by the $\overline{Z}$ and $\overline{X}$ logicals in $(\mathbb{C}^2)^{\otimes k_D}$ defined by the map $(H_1(f_\bullet), H^1(f^\bullet))$. The state $E_D \circ \Omega |\overline{\psi}\rangle$ is stabilised by those $\overline{Z}$ and $\overline{X}$ logicals and *all* stabilisers in S_D. Meanwhile, $M \circ E_C |\overline{\psi}\rangle$ is stabilised by the $\overline{Z}$ and $\overline{X}$ logicals determined by $(H_1(f_\bullet), H^1(f^\bullet))$[2], and other stabilisers in the image of (f_1, f^1).

Thus $M \circ E_C |\overline{\psi}\rangle \neq E_D \circ \Omega |\overline{\psi}\rangle$ in general, as there may be additional stabilisers in S_D which are not in the image of (f_1, f^1). However,

$$P_{S_D} \circ M \circ E_C |\overline{\psi}\rangle = E_D \circ \Omega |\overline{\psi}\rangle,$$

as projecting into the eigenspace adds the missing stabilisers. Note that $M \circ E_C |\overline{\psi}\rangle$ must still be a stabiliser state, as M is a stabiliser map, so the addition of these missing

[2]The -1 sign on any $\overline{Z}$ logicals does not matter for the maps E_C or M as they are over $\mathbb{C}$.

stabilisers must replace existing stabilisers of $M \circ E_C \left|\overline{\psi}\right\rangle$ which are not in S_D. In fact, one does not require the projector of *all* the independent stabilisers of $(D_\bullet, D^\bullet)$, merely those missing from $\mathrm{im}(M)$.

As the set of logical computational basis states spans the logical space $(\mathbb{C}^2)^{\otimes k_C}$, we thus have

$$P_{S_D} \circ M \circ E_C = E_D \circ \Omega.$$

■

This fits with the computational interpretation, as in code deformation the map on physical Hilbert spaces can initially place the initial state outside the logical space of the deformed code, which then must be projected back inside by performing stabiliser checks. A dual result to Lemma 2.2.18 applies to $\overline{X}$-preserving code maps, as expected.

While our definitions in this section are for chain complexes of length 2, in principle one can map between any two codes with an arbitrary number of meta-checks, or between a classical code and quantum code, which could be interpreted as 'switching on/off' either X or Z stabiliser measurements.

Code maps are related to code deformations, but we are aware of code deformation protocols which do not appear to fit in the model of chain maps described. For example, when moving defects around on the surface code for the purpose of, say, defect braiding [FMMC12], neither $\overline{Z}$ nor $\overline{X}$ operators are preserved in the sense we give here.

2.3 CSS code surgery

To understand code surgery we require some additional chain complex technology, namely tensor products and colimits.

2.3.1 Tensor products of classical codes

Here we recap tensor products [AC19] of classical codes from the perspective of homological algebra. This can be deduced from Definition 1.4.8, but this particular case deserves further inspection. Let $C_\bullet$ and $D_\bullet$ be two linear binary codes

$$C_\bullet = C_1 \xrightarrow{A} C_0 \ ; \qquad D_\bullet = D_1 \xrightarrow{B} D_0$$

where A and B are parity matrices. The dual codes $C^\bullet$, $D^\bullet$ are the codes obtained by transposing the parity-check matrices. The codes have parameters $[n_A, k_A, d_A]$ and $[n_B, k_B, d_B]$, and their dual codes have parameters $[n_A^\intercal, k_A^\intercal, d_A^\intercal]$ and $[n_B^\intercal, k_B^\intercal, d_B^\intercal]$. Explicitly,

$$n_A = \dim C_1; \quad k_A = \dim\ker(A); \quad n_A^\intercal = \dim C_0; \quad k_A^\intercal = \dim\ker(A^\intercal) = \dim C_0/\mathrm{im}(A),$$

so n_A, k_A are the length and dimension of the code $C_\bullet$, and $n_A^\intercal, k_A^\intercal$ are the length and dimension of the dual code $C^\bullet$. We will also use the respective distances $d_A, d_A^\intercal$. Similar definitions apply to B.

The tensor product quantum code is given by the chain complex:

$$(C \otimes D)_\bullet = C_1 \otimes D_1 \xrightarrow{\partial_2} C_0 \otimes D_1 \oplus C_1 \otimes D_0 \xrightarrow{\partial_1} C_0 \otimes D_0$$

where by convention we say that $\partial_2 = P_Z^\intercal$ and $\partial_1 = P_X$, and

$$P_Z = \begin{pmatrix} A^\intercal \otimes \mathrm{id}_{D_1} & \mathrm{id}_{C_1} \otimes B^\intercal \end{pmatrix}; \quad P_X = \begin{pmatrix} \mathrm{id}_{C_0} \otimes B & A \otimes \mathrm{id}_{D_0} \end{pmatrix}$$

We will use some straightforward facts about this code. It has parameters

$$[\![n_A^\intercal n_B + n_A n_B^\intercal, k_A^\intercal k_B + k_A k_B^\intercal, \min(d_A, d_B, d_A^\intercal, d_B^\intercal)]\!].$$

That $n_{(C\otimes D)} = n_A^\intercal n_B + n_A n_B^\intercal$ is obvious from $\dim(C\otimes D)_1$. $k_{(C\otimes D)} = \dim H_1((C\otimes D)_\bullet) = \dim \ker(P_X)/\mathrm{im}(P_Z^\intercal)$, which can be found using the Künneth formula [Wei84]:

$$H_1((C\otimes D)_\bullet) \cong H_0(C_\bullet)\otimes H_1(D_\bullet) \oplus H_1(C_\bullet)\otimes H_0(D_\bullet)$$

In particular we have the decomposition

$$\ker(P_X) = \ker(P_X)/\mathrm{im}(P_Z^\intercal) \oplus \mathrm{im}(P_Z^\intercal) = C_0/\mathrm{im}(A)\otimes\ker(B)\oplus\ker(A)\otimes D_0/\mathrm{im}(B)\oplus\mathrm{im}(P_Z^\intercal). \tag{2.1}$$

The distance can be obtained easily using this expression for $\ker(P_X)$ and a similar one for $\ker(P_Z)$:

$$\ker(P_Z) = \ker(A^\intercal)\otimes D_1/\mathrm{im}(B^\intercal)\oplus C_1/\mathrm{im}(A^\intercal)\otimes\ker(B^\intercal)\oplus\mathrm{im}(P_X^\intercal) \tag{2.2}$$

These equations are only required by the Künneth formula to hold up to isomorphism, but one can check using a simple counting argument that they hold on the nose.

Toric and surface codes, see Example 2.2.8 and Example 2.2.9 are basic examples of tensor product codes, where the input classical codes are repetition codes.

Considering families of tensor product codes built from families of classical codes, the products will be qLDPC iff the classical codes are also LDPC, and the parameters above mean that the scaling have $d \in \mathcal{O}(\sqrt{n})$ and $k \in \mathcal{O}(n)$. These bounds are saturated by quantum expander codes [LTZ15], using the fact that hypergraph product codes [TZ14] are tensor product codes with one of the classical codes dualised.

Tensor product codes are important to us because we will use them to 'glue' other codes together, and for performing single-qubit logical measurements.

2.3.2 Colimits in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$

Coproducts, pushouts and coequalisers are directly relevant for our applications. Coproducts are just direct sums, so we describe pushouts and coequalisers here.

Definition 2.3.1. The pushout of chain maps $f_\bullet : A_\bullet \to C_\bullet$ and $g_\bullet : A_\bullet \to D_\bullet$ gives the chain complex $Q_\bullet$, where each component is the pushout Q_n of f_n and g_n. The differentials $\partial_n^{Q_\bullet}$ are given by the unique mediating map from each component's pushout. Specifically, if we have the pushout

$$\begin{tikzcd} A_\bullet \arrow[r, "g_\bullet"] \arrow[d, "f_\bullet"'] & D_\bullet \arrow[d, "l_\bullet"] \\ C_\bullet \arrow[r, "k_\bullet"'] \arrow[ur, phantom, "\ulcorner", very near end] & Q_\bullet \end{tikzcd}$$

then for degrees $n, n+1$ we have

$$\begin{tikzcd}
A_n \arrow[rrr, "g_n"] \arrow[ddd, "f_n"'] & & & D_n \arrow[ddd, "l_n"] \\
& A_{n+1} \arrow[ul, "\partial^{A_\bullet}_{n+1}"] \arrow[r, "g_{n+1}"] \arrow[d, "f_{n+1}"'] & D_{n+1} \arrow[ur, "\partial^{D_\bullet}_{n+1}"'] \arrow[d, "l_{n+1}"] \arrow[dl, phantom, "\ulcorner"] & \\
& C_{n+1} \arrow[dl, "\partial^{C_\bullet}_{n+1}"] \arrow[r, "k_{n+1}"'] & Q_{n+1} \arrow[dr, dotted, "\partial^{Q_\bullet}_{n+1}"] & \\
C_n \arrow[rrr, "k_n"'] & & & Q_n
\end{tikzcd}$$

where

$$Q_n = (C \oplus D)_n / f_n \sim g_n; \quad k_n(c) = [c] \in Q_n; \quad l_n(d) = [d] \in Q_n.$$

with $[c]$ being the equivalence class in Q_n having c as a representative, and the same for $[d]$. As $k_n \circ \partial^{C_\bullet}_{n+1} \circ f_{n+1} = l_n \circ \partial^{D_\bullet}_{n+1} \circ g_{n+1}$ and the inner square is a pushout in $\mathtt{Mat}_{\mathbb{F}_2}$, there is a unique matrix $\partial^{Q_\bullet}_{n+1}$. The differentials satisfy $\partial^{Q_\bullet}_n \circ \partial^{Q_\bullet}_{n+1} = 0$, and one can additionally check that this is indeed a pushout in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ by considering the universal property at each component.

Definition 2.3.2. The coequaliser of chain maps $C_\bullet \overset{f}{\underset{g}{\rightrightarrows}} D_\bullet$ is a chain complex $E_\bullet$ and chain map $\mathrm{coeq}(f,g)_\bullet : D_\bullet \to E_\bullet$, which we will just call $\mathrm{coeq}_\bullet$. We have $E_n = D_n / f_n \sim g_n$ and $\mathrm{coeq}_n(d) = [d]$.

Doing some minor diagram chasing one can check that this is indeed a coequaliser in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$.

Remark 2.3.3. We can view the pushout

$$\begin{tikzcd}
A_\bullet \arrow[r, "g_\bullet"] \arrow[d, "f_\bullet"'] & D_\bullet \arrow[d, "l_\bullet"] \arrow[dl, phantom, "\ulcorner"] \\
C_\bullet \arrow[r, "k_\bullet"'] & Q_\bullet
\end{tikzcd}$$

as the coequaliser of $A_\bullet \overset{\tau_\bullet \circ f_\bullet}{\underset{\omega_\bullet \circ g_\bullet}{\rightrightarrows}} (C \oplus D)_\bullet$ for the inclusion maps $C_\bullet \xhookrightarrow{\tau_\bullet} (C \oplus D)_\bullet$, $D_\bullet \xhookrightarrow{\omega_\bullet} (C \oplus D)_\bullet$. The difference is that the pair of chain maps $k_\bullet, l_\bullet$ have been replaced with the single map $\mathrm{coeq}_\bullet$, so we have

$$A_\bullet \overset{\tau_\bullet \circ f_\bullet}{\underset{\omega_\bullet \circ g_\bullet}{\rightrightarrows}} (C \oplus D)_\bullet \xrightarrow{\mathrm{coeq}_\bullet} Q_\bullet$$

We can view coequalisers as instances of pushouts as well, doing a sort of reverse of the procedure above.

As with all colimits, those above are defined by the category theory only up to isomorphism. Because we are working over a field, the isomorphism class of a chain complex $Q_\bullet$ is completely determined by the dimensions of the underlying vector spaces $\{\dim Q_i\}_i$ and its *Betti numbers*, which is the set $\{\dim H_i(Q_\bullet)\}_i$ of dimensions of the homology spaces.

This is a homological version of the rank-nullity theorem. These are very large iso-classes, and we require more fine-grained control over which chain complexes are chosen by the colimits.

One way to choose a specific pushout of chain maps is via an explicit definition of the coequaliser of two based linear maps. For this we need not just a basis for our vector spaces, but an ordered basis. Using these coequalisers we can then construct pushouts of linear maps and their universal arrows, which is used in turn to define the pushout of chain maps. For the coequaliser of linear maps $r, s : V \to W$, we may take $s = 0$ by linearity. Then we let r^+ be the reflexive generalised inverse of r, which always exists by [WD98], and see that $P = I - rr^+$ is a projector $P : W \to W$ that coequalizes $r, 0 : V \to W$. To make this a universal projector we need to row-reduce the matrix of P, i.e. put P into row echelon form and remove all-zero rows, which is where we use the order on the basis of W. This row-reduced matrix will then have full rank and will be a universal coequaliser.

Alternatively, there is a straightforward way to choose the representatives we want from these iso-classes when the chain maps $f_\bullet, g_\bullet$ in the span are *basis-preserving*.

Definition 2.3.4. We say that a chain map is *basis-preserving* when every matrix at each component maps basis elements to basis elements.

This does not require that the map is either monic or epic, and is a property associated only with based vector spaces, as it evidently does not arise with abstract vector spaces. We can equivalently see this property as the case when the chain map is a collection of functions between the underlying sets $\{\tilde{f}_i : \tilde{C}_i \to \tilde{D}_i\}_{i \in \mathbb{Z}}$. We can also ask for co-chain maps to be basis-preserving, with the same definition.

Lemma 2.3.5. Let the chain maps $f_\bullet, g_\bullet$ be basis-preserving. Then there is always a choice of pushout such that the chain maps $k_\bullet, l_\bullet$ are also basis-preserving.

Proof. Let $Q_n = (C \oplus D)_n / f_n \sim g_n$. Then, for any basis element $a \in A_n$, the elements $f_n(a)$ and $g_n(a)$ are mapped by k_n and l_n respectively to the same basis element in Q_n. For basis elements in C_n, D_n which are not in the images of f_n and g_n, k_n and l_n will map them to distinct basis elements in Q_n. So this choice of representative of the isomorphism class of pushouts has $k_\bullet$, $l_\bullet$ being basis-preserving. ■

We can think of the pushout at each component as being a pushout in the category `Set`, freely promoted to $\texttt{Mat}_{\mathbb{F}_2}$. The differentials are then defined by the universal property. All of the choices of pushout such that $k_\bullet$, $l_\bullet$ are basis-preserving are equivalent up to a relabelling of basis elements in each component, hence (coherent) row and column permutation of the differential matrices. Computationally, this means that the choice of pushout is defined up to relabelling qubits, Z-checks and X-checks, and hence properties like code distance and being LDPC are well-defined for such pushouts. This will be important later.

Lemma 2.3.5 also applies to coequalisers, following Remark 2.3.3.

Lemma 2.3.6. $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ is an Abelian category, and thus is finitely complete and cocomplete, meaning that it has all finite limits and colimits.

This is reiterating Lemma 1.4.5 from the introduction. This lemma does not mean that every span has a *basis-preserving* pushout, but whenever there is a basis-preserving span there is a basis-preserving pushout. The same applies to coequalisers. From now on every example of a span we use will be basis-preserving, so we assume that the pushouts and coequalisers are also basis-preserving and will no longer mention this property.

2.3.3 Generic code surgery

We now give a general set of definitions for surgery between arbitrary compatible CSS codes; the condition for compatibility is very weak here. Working at this level of generality means that we cannot prove very much about the output codes or relevant logical maps. As a consequence, we will then focus on particular surgeries which make use of 'gluing' or 'tearing' along logical $\overline{Z}$ or $\overline{X}$ operators in Section 2.3.4.

Definition 2.3.7. Let $(C_\bullet, C^\bullet)$, $(D_\bullet, D^\bullet)$ and $(A_\bullet, A^\bullet)$ be CSS codes, such that there is a basis-preserving span of chain complexes

$$\begin{tikzcd} A_\bullet \arrow[r, "g_\bullet"] \arrow[d, "f_\bullet"'] & D_\bullet \\ C_\bullet & \end{tikzcd}$$

The Z-type merged code of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $f_\bullet, g_\bullet$ is the code $(Q_\bullet, Q^\bullet)$ such that $Q_\bullet$ is the pushout of the above diagram.

Recall from Remark 2.3.3 that we can view any pushout as a coequaliser. We thus have

$$\begin{tikzcd} A_\bullet \arrow[r, shift left, "\iota_C \circ f_\bullet"] \arrow[r, shift right, "\iota_D \circ g_\bullet"'] & (C \oplus D)_\bullet \arrow[r, "\mathrm{coeq}_\bullet"] & Q_\bullet \end{tikzcd}$$

and we call $\mathrm{coeq}_\bullet$ the Z-merge chain map. We can bundle this up into a Z-merge code map:

$$\mathcal{F}_{\overline{Z}} \Big\Downarrow \quad \begin{tikzcd} ((C \oplus D)_\bullet, (C \oplus D)^\bullet) \arrow[d, shift right, "\mathrm{coeq}_\bullet"'] \\ (Q_\bullet, Q^\bullet) \arrow[u, shift right, "\mathrm{coeq}^\bullet"'] \end{tikzcd} \tag{2.3}$$

We then call $\mathrm{coeq}^\bullet : Q^\bullet \to (C \oplus D)^\bullet$ an X-split cochain map, and hence we have an X-split code map too:

$$\mathcal{F}_{\overline{X}} \Big\Uparrow \quad \begin{tikzcd} ((C \oplus D)_\bullet, (C \oplus D)^\bullet) \arrow[d, shift right, "\mathrm{coeq}_\bullet"'] \\ (Q_\bullet, Q^\bullet) \arrow[u, shift right, "\mathrm{coeq}^\bullet"'] \end{tikzcd} \tag{2.4}$$

Definition 2.3.8. Let $(C_\bullet, C^\bullet)$, $(D_\bullet, D^\bullet)$ and $(A_\bullet, A^\bullet)$ be CSS codes, such that there is a span of cochain complexes

$$\begin{tikzcd} A^\bullet \arrow[r, "g^\bullet"] \arrow[d, "f^\bullet"'] & D^\bullet \\ C^\bullet & \end{tikzcd}$$

the X-type merged code of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $f^\bullet, g^\bullet$ is the code $(Q_\bullet, Q^\bullet)$ such that $Q^\bullet$ is the pushout of the above diagram.

We have an X-merge cochain map and thus X-merge code map using the coequaliser picture, so

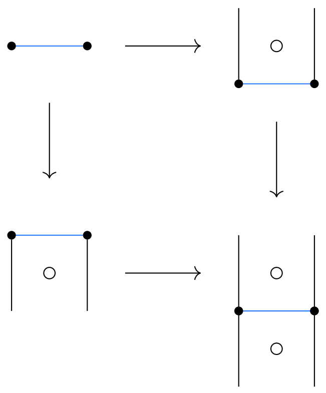
$$\mathcal{E}_{\overline{X}} \Big\Uparrow \quad \begin{tikzcd} (Q_\bullet, Q^\bullet) \arrow[d, shift right, "\mathrm{coeq}_\bullet"'] \\ ((C \oplus D)_\bullet, (C \oplus D)^\bullet) \arrow[u, shift right, "\mathrm{coeq}^\bullet"'] \end{tikzcd}$$

We also have a Z-split chain map and the Z-split code map $\mathcal{E}_{\overline{Z}}$ by taking the opposite.

$$\begin{tikzcd}
 & (Q_\bullet, Q^\bullet) \arrow[d, shift right, "\mathrm{coeq}_\bullet"'] \\
 & ((C\oplus D)_\bullet, (C\oplus D)^\bullet) \arrow[u, shift right, "\mathrm{coeq}^\bullet"']
\end{tikzcd} \qquad \mathcal{E}_{\overline{Z}} \Downarrow$$

This is rather abstract, so let's see a small concrete example.

Example 2.3.9. Consider the following pushout of square lattices:



We have not properly formalised pushouts of square lattices in the main body for brevity, but we do so in Appendix 1. Informally, we are just 'gluing along' the graph in the top left corner, where the edges to be glued are coloured in blue.

We can consider this pushout to be in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ [3], giving the pushout:

$$\begin{tikzcd}
A_\bullet \arrow[r, "g_\bullet"] \arrow[d, "f_\bullet"'] & D_\bullet \arrow[d, "q_\bullet"] \\
C_\bullet \arrow[r, "p_\bullet"'] \arrow[ur, phantom, "\ulcorner"] & Q_\bullet
\end{tikzcd}$$

with

$$A_\bullet = \mathbb{F}_2 \xrightarrow{\partial_1^{A_\bullet}} \mathbb{F}_2^2$$

$$C_\bullet = \mathbb{F}_2 \xrightarrow{\partial_2^{C_\bullet}} \mathbb{F}_2^3 \xrightarrow{\partial_1^{C_\bullet}} \mathbb{F}_2^2$$

$$D_\bullet = \mathbb{F}_2 \xrightarrow{\partial_2^{D_\bullet}} \mathbb{F}_2^3 \xrightarrow{\partial_1^{D_\bullet}} \mathbb{F}_2^2$$

and

$$\partial_1^{A_\bullet} = \begin{pmatrix}1\\1\end{pmatrix}; \quad \partial_2^{C_\bullet} = \partial_2^{D_\bullet} = \begin{pmatrix}1\\1\\1\end{pmatrix}; \quad \partial_1^{C_\bullet} = \begin{pmatrix}1 & 1 & 0\\0 & 1 & 1\end{pmatrix}; \quad \partial_1^{D_\bullet} = \begin{pmatrix}1 & 0 & 1\\0 & 1 & 1\end{pmatrix}.$$

One can see from the cell complexes that we have

$$Q_\bullet = \mathbb{F}_2^2 \xrightarrow{\partial_2^{Q_\bullet}} \mathbb{F}_2^5 \xrightarrow{\partial_1^{Q_\bullet}} \mathbb{F}_2^2$$

[3]Categorically, this is because there is a cocontinuous functor from the appropriate category of square lattices to $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$.

with

$$\partial_2^{Q_\bullet} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} ; \quad \partial_1^{Q_\bullet} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

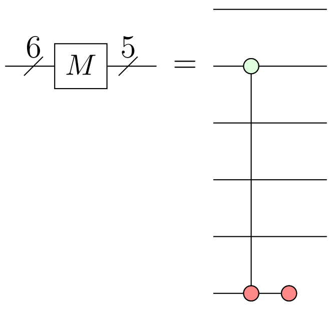
Rather than compute the pushout maps, let us instead give the coequaliser $\mathrm{coeq}_\bullet$:

$$\begin{tikzcd}
(C \oplus D)_2 \arrow[r, "\partial_2^{(C\oplus D)_\bullet}"] \arrow[d, "\mathrm{coeq}_2"] & (C \oplus D)_1 \arrow[r, "\partial_1^{(C\oplus D)_\bullet}"] \arrow[d, "\mathrm{coeq}_1"] & (C \oplus D)_0 \arrow[d, "\mathrm{coeq}_0"] \\
Q_2 \arrow[r, "\partial_2^{Q_\bullet}"] & Q_1 \arrow[r, "\partial_1^{Q_\bullet}"] & Q_0
\end{tikzcd}$$

We immediately see that $\mathrm{coeq}_2 = \mathrm{id}$. For the other two surjections we have

$$\mathrm{coeq}_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} ; \quad \mathrm{coeq}_0 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

Finally we interpret all the chain complexes in this pushout as being the Z-type complexes of CSS codes $(A_\bullet, A^\bullet)$, $(C_\bullet, C^\bullet)$ etc. Thus we have a Z-merge code map $\mathcal{F}_{\overline{Z}}$, with an interpretation M as a $\mathbb{C}$-linear map, using coeq_1 and $\mathrm{coeq}_1^\intercal$. We refrain from writing out the full 32-by-64 matrix, but as a ZX-diagram using gates from $\{\mathrm{CNOT}, |+\rangle, \langle 0|\}$ we have simply



We know from Lemma 1.4.4 that this map must restrict to a map on logical qubits. However, easy calculations show that $H_1((C \oplus D)_\bullet) = 0$, while $H_1(Q_\bullet) = 1$. That is, in the code $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ there are no logical qubits – there are still operators which show up as errors and some which don't, but all of those which don't are products of Z or X stabiliser generators. By Corollary 2.2.13 and Corollary 2.2.17 the logical map in `FHilb` is then just $|+\rangle$. This trivially preserves both $\overline{Z}$ and $\overline{X}$ operators, although its opposite code map $\mathcal{F}_{\overline{X}}$ does not preserve $\overline{Z}$ operators.

This example was very simple, but the idea extends in a general way. To convey how general this notion of CSS code surgery is, consider the balanced product codes from [BE21A, PK22A, PK22B]. The balanced product of codes is by definition a coequaliser in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, and so we can convert it into a pushout using routine category theory. The coequaliser is

$$\begin{tikzcd}
(C \otimes A \otimes D)_\bullet \arrow[r, shift left, "f_\bullet"] \arrow[r, shift right, "g_\bullet"'] & (C \otimes D)_\bullet \arrow[r, "\mathrm{coeq}_\bullet"] & (C \otimes_A D)_\bullet
\end{tikzcd}$$

where $g_\bullet$ and $f_\bullet$ represent left and right actions of $A_\bullet$ respectively; in all the cases from [BE21A, PK22A, PK22B] these actions are basis-preserving. We have not explicitly defined the tensor product of chain complexes in the main body for brevity, but see Definition 1.4.8. Then to this coequaliser we can associate a pushout,

$$\begin{tikzcd}
((C \otimes A \otimes D) \oplus (C \otimes D))_\bullet \arrow[r, "(g_\bullet \mid \mathrm{id}_\bullet)"] \arrow[d, "(f_\bullet \mid \mathrm{id}_\bullet)"'] \arrow[dr, phantom, "\ulcorner", very near end] & (C \otimes D)_\bullet \arrow[d, "l_\bullet"] \\
(C \otimes D)_\bullet \arrow[r, "k_\bullet"'] & (C \otimes_A D)_\bullet
\end{tikzcd}$$

where one can check that the universal property is the same in both cases. Thus we can think of a balanced product as a merge of tensor product codes, with the apex being two adjacent tensor product codes. As the maps in the span are evidently not monic, the merge is of a distinctly different sort from Example 2.3.9, and also the $\overline{Z}$- and $\overline{X}$-merges we will describe in Section 2.3.4.

It would be convenient if we could guarantee some properties of pushouts in general; for example, if the pushout of LDPC codes was also LDPC, or if the homologies were always preserved. Unfortunately, the definition is general enough that neither of these are true. We discuss this in slightly greater detail in Appendix 2, but the gist is that we need to stipulate some additional conditions to guarantee bounds on these quantities.

2.3.4 Surgery along a logical operator

The procedure of merging here is closely related to that of 'welding' in [Mic14]. Our focus is not just on the resultant codes, but the maps on physical and logical data. On codes generated from square lattices, the merges here will correspond to a pushout along a 'string' through the lattice.

Definition 2.3.10. Let $C_\bullet = C_2 \xrightarrow{\partial_1} C_1 \xrightarrow{\partial_1} C_0$ be a length 2 chain complex. Let $v \in C_1$ be a vector such that $v \in \ker(\partial_1^{C_\bullet})\backslash \mathrm{im}(\partial_2^{C_\bullet})$. We now construct the *logical operator subcomplex* $V_\bullet$. This has:

$$\tilde{V}_1 = \mathrm{supp}\ v; \quad \partial_1^{V_\bullet} = \partial_1^{C_\bullet} \restriction_{\mathrm{supp}\ v}; \quad \tilde{V}_0 = \bigcup_{u \in \mathrm{im}(\partial_1^{V_\bullet})} \mathrm{supp}\ u$$

where supp v is the set of basis vectors in the support of v, and $\partial_i \restriction_S$ is the restriction of a differential to a subset S of its domain. All other components and differentials of $V_\bullet$ are zero.

There is a monic $f_\bullet : V_\bullet \hookrightarrow C_\bullet$ given by the inclusion maps of $V_1 \subseteq C_1$ etc.

Definition 2.3.11. Let $V_\bullet$ be a logical operator subcomplex of two chain complexes

$$C_\bullet = C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$$

and

$$D_\bullet = D_2 \xrightarrow{\partial_2} D_1 \xrightarrow{\partial_1} D_0$$

simultaneously, so there is a basis-preserving monic span

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & D_\bullet \\ C_\bullet & \end{tikzcd}$$

This monic span has the pushout

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & D_\bullet \arrow[d, "q_\bullet"] \\ C_\bullet \arrow[r, "p_\bullet"'] & Q_\bullet \arrow[ul, phantom, "\ulcorner", very near start] \end{tikzcd}$$

with components

$$Q_2 = C_2 \oplus D_2; \quad Q_1 = C_1 \oplus D_1/(\text{supp }\ v \sim \text{supp }\ w); \quad Q_0 = C_0 \oplus D_0/(\tilde{f}_0 \sim \tilde{g}_0),$$

where w is the logical operator associated to $\mathrm{im}(g_1) \in D_1$.

The construction here is inspired by [CKBB22].

Definition 2.3.12. Let $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ be CSS codes. Let $(V_\bullet, V^\bullet)$ be a CSS code such that $V_\bullet$ is a logical operator subcomplex of $C_\bullet$ and $D_\bullet$; this means that $(V_\bullet, V^\bullet)$ can be seen as merely a classical code, as $V_2 = 0$. Then the Z-type merged CSS code $(Q_\bullet, Q^\bullet)$ is called the $\overline{Z}$- *merged code* of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $(V_\bullet, V^\bullet)$.

Definition 2.3.13. (Irreducible) Let $V_\bullet$ be a logical operator subcomplex such that for the inclusion maps $f_\bullet$ and $g_\bullet$, $\mathrm{im}(f_1)$ and $\mathrm{im}(g_1)$ contain only one operator in $Z_1(C_\bullet)$, $Z_1(D_\bullet)$ respectively, with those operators being v, w. Then we say that these operators are irreducible, as they contain no other logicals in their support, and the pushout satisfies the irreducibility property.

The intuition here, following [CKBB22], is that it is convenient when the logical operators we glue along do not themselves contain any nontrivial logical operators belonging to a different logical qubit; if they do, the gluing procedure may yield a more complicated output code, as we could be merging along multiple logical operators simultaneously. In Appendix 3 we demonstrate that it is possible for this condition to not be satisfied, using a patch of octagonal surface code. Additionally, we do not want the gluing procedure to send any logical $\overline{Z}$ operators to stabilisers.

We would like to study not only the resultant code given some $\overline{Z}$-merge, but also the map on the logical space. We can freely switch between pushouts and coequalisers. Recall the Z-merge code map $\mathcal{F}_{\overline{Z}}$ from Equation 2.3. We call this a $\overline{Z}$-merge code map when the merge is along a $\overline{Z}$-operator as above, and from now on we assume that all merges are irreducible unless otherwise stated.

Lemma 2.3.14. Let $(Q_\bullet, Q^\bullet)$ be a irreducible $\overline{Z}$-merged code with parameters $[\![n_Q, k_Q, d_Q]\!]$, and let $[\![n_C, k_C, d_C]\!]$, $[\![n_D, k_D, d_D]\!]$ be the parameters of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ respectively. Let $n_V = \dim V_0$. Then

$$n_Q = n_C + n_D - n_V; \quad k_Q \geq k_C + k_D - 1$$

Further, let $\{[u]_i\}$ and $\{[v]_j\}$ be the bases for $H_1(C_\bullet)$ and $H_1(D_\bullet)$ respectively, and say w.l.o.g. that $u \in [u]_1$ and $v \in [v]_1$ are the vectors quotiented by the pushout. Then $H_1(Q_\bullet)$

has a basis $\{[w]_l\}$ for $l \leq k_Q$, where $[w]_1 = [u]_1 = [v]_1$, $[w]_l = [u]_l$ when $1 < l \leq k_C$ and $[w]_l = [v]_{l-k_C+1}$ for $k_C < l \leq k_C + k_D - 1$.

Proof. n_Q is immediate by the definition. Given $u \in [u]_1$ and $v \in [v]_1$, any other representatives $y \in [u]_1, x \in [v]_1$ belong to the same equivalence class in $H_1(Q_\bullet)$, as $y \sim u \sim v \sim x$.

All other equivalence classes remain distinct, as they would be in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$.

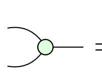
However, it is possible to introduce new equivalence classes, without a preimage in $H_1((C \oplus D)_\bullet)$. Despite $\mathrm{coeq}_\bullet$ being surjective, the lift $H_1(\mathrm{coeq}_\bullet)$ is not always surjective, as the restriction of coeq_1 to $\ker(\partial_1^{(C\oplus D)_\bullet})$ is not always surjective. ■

This last case is subtle, and rarely occurs with small codes or topological codes. We present an explicit example in Appendix 5. Should it be useful, we can swap to subsystem codes after the merge, and not store any data in the new logical qubits, relegating them to gauge qubits, as done in [CKBB22].

Lemma 2.3.15. Let the $\overline{Z}$-merge code map

$$\begin{tikzcd} {} \arrow[d, Rightarrow, "\mathcal{F}_{\overline{Z}}"'] & ((C \oplus D)_\bullet, (C \oplus D)^\bullet) \arrow[d, shift right, "\mathrm{coeq}_\bullet"'] \\ {} & (Q_\bullet, Q^\bullet) \arrow[u, shift right, "\mathrm{coeq}^\bullet"'] \end{tikzcd}$$

of an irreducible $\overline{Z}$-merged code have its interpretation M as a $\mathbb{C}$-linear map. Then M acts as



$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

on each pair of qubits in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ which are equivalent in $(Q_\bullet, Q^\bullet)$ and M acts as identity on all other qubits.

Proof. M must have the following maps on Paulis on each pair of qubits being merged:

$$Z \otimes I \mapsto Z; \quad I \otimes Z \mapsto Z; \quad X \otimes X \mapsto X$$

which uniquely defines the matrix above. In other words we have $|00\rangle \mapsto |0\rangle$, $|11\rangle \mapsto |1\rangle$, $|01\rangle \mapsto |0\rangle$, $|10\rangle \mapsto |0\rangle$ etc, which has the convenient presentation as the ZX diagram on the left above. ■

Lemma 2.3.16. Let $(Q_\bullet, Q^\bullet)$ be an irreducible $\overline{Z}$-merged code of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $(V_\bullet, V^\bullet)$. Call $f = H_1(\mathrm{coeq}_\bullet)$. Then

$$f([u]_i + [v]_j) = [w]_l$$

where $[w]_l$ was defined in Lemma 2.3.14.

This is obvious by considering the surjection in question and using Lemma 2.3.14. It essentially says that on the pair of logical operators in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ which are being quotiented together, $\mathcal{F}_{\overline{Z}}$ acts as:

$$\overline{Z} \otimes \overline{I} \mapsto \overline{Z}; \quad \overline{I} \otimes \overline{Z} \mapsto \overline{Z}; \quad \overline{X} \otimes \overline{X} \mapsto \overline{X}$$

where the map on Xs is inferred from the dual. In the case where new logical qubits are introduced, as described in Lemma 2.3.14, it can be easily checked that these are initialised in the logical $|+\rangle$ state, as they are not in the image of the $H_1(\mathrm{coeq}_\bullet)$.

Lemma 2.3.17. Let the merged code have no new logical qubits, i.e. $k_Q = k_C + k_D - 1$. Then,

$$d^X_Q \geq \min(d^X_C, d^X_D)$$

Proof. By considering the code map $\mathcal{F}_{\overline{Z}}$, we see that any $\overline{X}$ logical operator u in $(Q_\bullet, Q^\bullet)$ has a preimage w which is also an $\overline{X}$ logical operator in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$, s.t. $|w| \leq |u|$. This is because coeq$^\bullet$ can be restricted to $f^\intercal = H^1(\mathrm{coeq}^\bullet)$, and any logical in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ has sublogicals in $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$. ■

The proof fails when the merged code has new logical qubits, as there can be a $\overline{X}$ logical operator u whose image under coeq$^\bullet$ is in $[0]$, which gives no bound on the weight of u.

Remark 2.3.18. Note that we do not in general have a lower bound on d^Z_Q in terms of d^Z_C and d^Z_D. We can see this from the discussion in Section 2.2.3. Given the code map $\mathcal{F}_{\overline{Z}}$, the chain map $f_1 : (C \oplus D)_1 \to Q_1$ restricts to $H_1(f)$, but this does not preclude there being other vectors in $(C \oplus D)_1 \backslash \ker \partial_1^{(C\oplus D)_\bullet}$ which are mapped into one of the equivalence classes in $H_1(Q_\bullet)$. In computational terms, while we cannot have detectable X operators in the initial codes which are mapped to logicals by the code map $\mathcal{F}_{\overline{Z}}$, this is unfortunately possible with detectable Z operators. We illustrate this with an example in Appendix 4.

We now show that, if we consider two codes to be merged as instances of LDPC families, their combined $\overline{Z}$-merged code code is also LDPC. Recall Definition 2.2.3.

Lemma 2.3.19. (LDPC) Say our input codes $(C_\bullet, C^\bullet)$, $(D_\bullet, D^\bullet)$ have maximal weights of generators labelled w^Z_C, w^X_C and w^Z_D, w^X_D respectively. Let $(Q_\bullet, Q^\bullet)$ be an irreducible $\overline{Z}$-merged code of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $(V_\bullet, V^\bullet)$. Then

$$w^Z_Q = \max(w^Z_C, w^Z_D); \quad w^X_Q < w^X_C + w^X_D.$$

Similarly, letting the input codes have maximal number of shared generators on a single qubit q^Z_C, q^X_C and q^Z_D, q^X_D we have

$$q^Z_Q \leq q^Z_C + q^Z_D; \quad q^X_Q = \max(q^X_C, q^X_D)$$

Proof. None of the Z-type generators are quotiented by a $\overline{Z}$-merge map, so $w^Z_Q = w^Z_{(C\oplus D)} = \max(w^Z_C, w^Z_D)$. For the X-type generators, in the worst case the two generators which are made to be equivalent by the merge are the highest weight ones. For these generators to appear in V_0 they must have at least two qubits in each of their support which is in V_1, and thus these qubits are merged together, so $w^X_Q < w^X_C + w^X_D$.

Next, using again the fact that none of the Z-type generators are quotiented, a single qubit could in the worst case be the result of merging two qubits in $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ which each have the maximal number of shared Z-type generators, so $q^Z_Q \leq q^Z_C + q^Z_D$. For the X case, if a qubit is in V_1 then all X-type generators it is in the support of must appear in V_0. Therefore, when any two qubits are merged all of their X-type generators are also merged. Thus $q^X_Q = q^X_{(C\oplus D)} = \max(q^X_C, q^X_D)$. ■

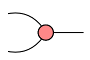
Note that as w^Z, w^X and q^Z, q^X are at worst additive in those of the input codes, the $\overline{Z}$-merge of two LDPC codes is still LDPC, assuming the pushout is still well-defined using matching $\overline{Z}$ operators for each member of the code families. Next, we dualise everything, and talk about $\overline{X}$-merges.

Definition 2.3.20. Let $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ be CSS codes. Let $(V_\bullet, V^\bullet)$ be a CSS code such that $V^\bullet$ is a logical operator subcomplex of $C^\bullet$ and $D^\bullet$, and $Q^\bullet$ is the merged complex along $V^\bullet$. Then the CSS code $(Q_\bullet, Q^\bullet)$ is called the $\overline{X}$- *merged code* of $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ along $(V_\bullet, V^\bullet)$.

In this case we glue along an $\overline{X}$ logical operator instead. The notions of irreducibility, Lemma 2.3.14 and Lemma 2.3.19 carry over by transposing appropriately.

An $\overline{X}$-merge map $\mathcal{E}_{\overline{X}}$ can be defined similarly, and a similar result as Lemma 2.3.15 applies to irreducible $\overline{X}$-merged codes.

Lemma 2.3.21. Let the $\overline{X}$-merge code map of an irreducible $\overline{X}$-merged code have its interpretation L as a $\mathbb{C}$-linear map. Then L acts as



$$= \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 0 & 0 & 1\\ 0 & 1 & 1 & 0\end{pmatrix}$$

on each pair of qubits in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ which are equivalent in $(Q_\bullet, Q^\bullet)$, i.e. $|{+}{+}\rangle \mapsto |{+}\rangle$, $|{-}{-}\rangle \mapsto |{-}\rangle$, and L acts as identity on all other qubits.

Proof. This time, L must have the maps

$$X \otimes I \mapsto X; \quad I \otimes X \mapsto X; \quad Z \otimes Z \mapsto Z$$

■

Similarly, the maps on logical operators are

$$\overline{X} \otimes \overline{I} \mapsto \overline{X}; \quad \overline{I} \otimes \overline{X} \mapsto \overline{X}; \quad \overline{Z} \otimes \overline{Z} \mapsto \overline{Z}$$

and, when new logical qubits are generated, they are initialised in the $|0\rangle$ state.

Having discussed $\overline{Z}$- and $\overline{X}$-merged codes, we briefly mention splits. These are just the opposite code maps to $\mathcal{F}_{\overline{Z}}$ and $\mathcal{E}_{\overline{X}}$. In both cases, all the mappings are determined entirely by Lemma 2.3.14 by taking transposes or adjoints when appropriate.

Remark 2.3.22. In practice, when the CSS codes in question hold multiple logical qubits it may be preferable to merge/split along multiple disjoint $\overline{Z}$ or $\overline{X}$ operators at the same time. Such a protocol is entirely viable within our framework, and requires only minor tweaks to the above results. The same is true should one wish to merge/split along operators within the same code.

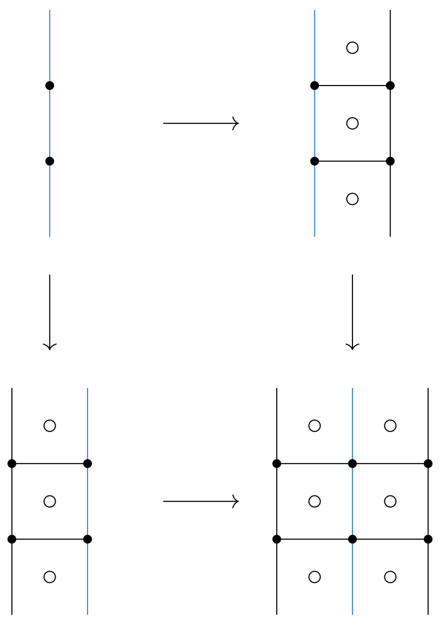
We now look at a short series of examples.

2.3.5 Examples of surgery

Lattice surgery

Lattice surgery is the prototypical instance of CSS code surgery. It starts with patches of surface code and then employs irreducible splits and merges to perform non-unitary logical operations [HFDM12]. The presentation we give of lattice surgery is idiosyncratic, in the sense that we perform the merges on physical edges/qubits, whereas the standard method is to introduce additional edges between patches to join them together. We remedy this in Section 2.4.

Consider the pushout of cell complexes below:



As before, we informally consider this to be 'gluing along' the graph in the top left, but for completeness it is formalised in Appendix 1. By considering the pushout to be in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, we have:

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & D_\bullet \arrow[d, "q_\bullet"] \\ C_\bullet \arrow[r, "p_\bullet"'] \arrow[ur, phantom, "\ulcorner", very near end] & Q_\bullet \end{tikzcd}$$

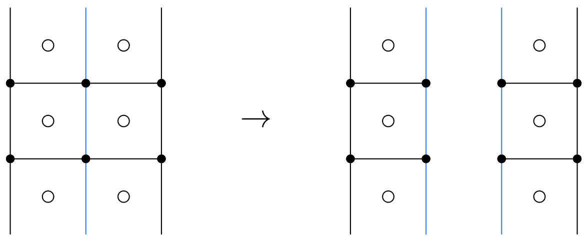
Letting $\mathrm{coeq}_\bullet : (C \oplus D)_\bullet \to Q_\bullet$ be the relevant coequaliser map, we see that $\mathcal{F}_{\overline{Z}} = (\mathrm{coeq}_\bullet, \mathrm{coeq}^\bullet)$ constitutes an irreducible $\overline{Z}$-merge map. In particular, observe that $\mathcal{F}_{\overline{Z}}$ sends the logical operators:

$$\overline{Z} \otimes \overline{I} \mapsto \overline{Z}$$
$$\overline{I} \otimes \overline{Z} \mapsto \overline{Z}$$
$$\overline{X} \otimes \overline{X} \mapsto \overline{X}$$

as predicted by Lemma 2.3.16.

The first two give $H_1(\mathrm{coeq}_\bullet) = \begin{pmatrix} 1 & 1 \end{pmatrix}$ and the last $H^1(\mathrm{coeq}^\bullet) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. $\mathcal{F}_{\overline{Z}}$ is evidently $\overline{Z}$-preserving but not $\overline{X}$-preserving, as $\overline{X} \otimes \overline{I}$ is taken to an operation which is detected by the Z stabilisers. Observe that we end up with a greater cosystolic distance of $(Q_\bullet, Q^\bullet)$ than we started with in $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$.

If we instead consider the pair $(\mathrm{coeq}_\bullet, \mathrm{coeq}^\bullet)$ as an $\overline{X}$-preserving code map $\mathcal{F}_{\overline{X}}$, then it is an irreducible $\overline{X}$-split map. In terms of cell complexes we would have [4]

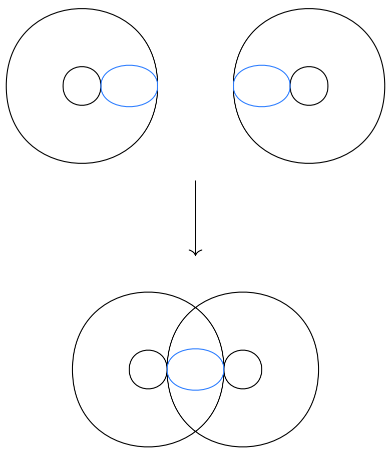


[4]Pedantically, this is a morphism in the opposite category of cell complexes $\mathtt{OACC}^{\mathrm{op}}$.

We similarly have an irreducible $\overline{X}$-merge map and irreducible $\overline{Z}$-split map with the obvious forms by dualising appropriately.

Remark 2.3.23. While it is convenient to choose logical operators along patch boundaries to glue along, so that the complexes can all be embedded on the 2D plane, this is not necessary. One could intersect two patches along any matching operator.

Recall the toric code from Example 2.2.8. We can merge two copies of the code along a logical $\overline{Z}$ operator, which corresponds to an essential cycle of each torus. The resultant code will then look like two tori intersecting, depending somewhat on the choices of essential cycle:



The $\overline{Z}$-merge map on logical qubits will be the same as for patches.

Shor code surgery

Of course, the pushout we take does not have to come from square lattices. Let $C_\bullet$ and $D_\bullet$ be two copies of Shor codes from Example 2.2.7.[5] We can perform merges between them. We give two examples. First, for a $\overline{Z}$-merge, we take the logical $\overline{Z}$ operator $\overline{Z} = \bigotimes_i^8 Z_i$ and apply Definition 2.3.10 to get the logical operator subcomplex:

$$V_\bullet = V_1 \xrightarrow{P_X} V_0$$

with $V_1 = \mathbb{F}_2^9$, $V_0 = \mathbb{F}_2^2$, and all other components zero. This is just $C_\bullet$ from Example 2.2.7 truncated to be length 1, as this logical $\overline{Z}$ operator has support on all physical qubits; this logical is not irreducible. The monic chain map $f_\bullet$ given by inclusion into the Shor code is just

$$\begin{tikzcd} 0 \arrow[r, "0"] \arrow[d, "0"] & V_1 \arrow[r, "P_X"] \arrow[d, "\mathrm{id}"] & V_0 \arrow[d, "\mathrm{id}"] \\ C_2 \arrow[r, "P_Z^\intercal"] & C_1 \arrow[r, "P_X"] & C_0 \end{tikzcd}$$

and the same for $g_\bullet$. The pushout of

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & D_\bullet \\ C_\bullet & \end{tikzcd}$$

[5]The Shor code can be constructed as a cellulation of the projective plane, so it is actually not wholly dissimilar from the lattice codes [FM01].

will then be

$$Q_\bullet = \mathbb{F}_2^{12} \xrightarrow{\partial_2^{Q_\bullet}} \mathbb{F}_2^{9} \xrightarrow{\partial_1^{Q_\bullet}} \mathbb{F}_2^{2}$$

where $\partial_1^{Q_\bullet} = P_X$ and $\partial_2^{Q_\bullet} = \left(P_Z^\intercal | P_Z^\intercal\right)$. We have ended up with virtually the same code as the Shor code, except that we have a duplicate for every Z-type generator, i.e. every measurement of Z stabilisers is performed twice and the result noted separately. While this example is very simple, it highlights that the result of a merge can have somewhat subtle features, such as duplicating measurements, which the two input codes do not. This merge did not use irreducible logical operators.

For our second case, we use a different (but equivalent) logical operator, $\overline{Z} = Z_1 \otimes Z_4 \otimes Z_7$. We still glue two copies of the Shor code, but now we have $V_1 = \mathbb{F}_2^3$, $V_0 = \mathbb{F}_2^2$ and $\partial_1^{V_\bullet} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$. That is, our logical operator subcomplex is just the repetition code from Example 2.2.1. The logical is irreducible. We then have

$$\begin{tikzcd}
0 \arrow[r, "0"] \arrow[d, "0"] & V_1 \arrow[r, "\partial_1^{V_\bullet}"] \arrow[d, "f_1"] & V_0 \arrow[d, "\mathrm{id}"] \\
C_2 \arrow[r, "P_Z^\intercal"] & C_1 \arrow[r, "P_X"] & C_0
\end{tikzcd}$$

where

$$f_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

and the same for g_1, forming again a monic span of chain complexes. The resultant $\overline{Z}$-merged code is then

$$Q_\bullet = \mathbb{F}_2^{12} \xrightarrow{\partial_2^{Q_\bullet}} \mathbb{F}_2^{15} \xrightarrow{\partial_1^{Q_\bullet}} \mathbb{F}_2^{2}$$

and the large matrices $\partial_2^{Q_\bullet}$ and $\partial_1^{Q_\bullet}$ are easily obtained by quotienting out rows and columns from $\partial_2^{C_\bullet} \oplus \partial_2^{D_\bullet}$ and $\partial_1^{C_\bullet} \oplus \partial_1^{D_\bullet}$.

2.4 Error-corrected logical operations

We now describe how our abstract formalism leads to a general set of error-corrected logical operations for CSS codes. We consider this to be a good application of the homological algebraic formalism, as we suspect these logical operations would be challenging to derive without the machinery of $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$. [6] So far in our description of code maps there are two main assumptions baked in: that one can perform linear maps between CSS codes (a) deterministically and (b) while maintaining error-correction, both of which are desired for performing quantum computation.

[6]An alternative approach could be to use Tanner graphs.

For assumption (a), we can only implement code maps which are interpreted as an isometry deterministically. If they are not, instead we must perform measurements on physical qubits. Recall from Proposition 2.2.12 that every code map has an interpretation constructed from CNOTs and some additional states and effects taken from $\{|+\rangle, \langle 0|\}$ for a $\overline{Z}$-preserving code map or $\{\langle +|, |0\rangle\}$ for an $\overline{X}$-preserving code map. This means that in order to implement the code map non-deterministically, one need only apply CNOTs and measure some qubits in the Z-basis (for a $\overline{Z}$-preserving code map) or the X-basis ($\overline{X}$-preserving code map). Of course, should we acquire the undesired measurement result, we induce errors in our code map. There is no protocol for correcting these errors in all generality. For assumption (b), there is no protocol for performing arbitrary CNOT circuits on physical qubits in a code fault-tolerantly. However, when performing CSS code surgery which is an irreducible $\overline{Z}$- or $\overline{X}$-merge, we have a protocol which addresses both (a) and (b).

2.4.1 Procedure summary

Our procedure for performing an error-corrected $\overline{Z} \otimes \overline{Z}$ measurement is as follows:

1. Find a matching $\overline{Z}$ logical operator which belongs to both initial codes, in the sense of Definition 2.3.10.
2. Verify that this logical operator satisfies the irreducibility property of Definition 2.3.13 in both codes.
3. Verify that the merge is bounded below, in the sense of Definition 2.4.8 below.
4. Perform the merge as described in Proposition 2.4.10.

We do not know how difficult it will be in general to perform the verification in steps (2) and (3) for codes (or families of codes) of interest.

2.4.2 Full description of procedure

We will first describe gauge fixing. Luckily this does not become an additional condition, as we will show it coincides precisely with irreducibility. For reasons of brevity we do not describe the connection between lattice surgery and gauge fixing, but refer the interested reader to [VLC19]. Briefly, we will consider the whole system to be a subsystem code, and fix the gauges of the $\overline{Z}$ operators we are gluing along.

Definition 2.4.1. Let $C_\bullet$ be a chain complex and u be a representative of the equivalence class $[u] \in H_1(C_\bullet)$, which is a basis vector of $H_1(C_\bullet)$. Let x be a vector in C_1 such that $|x| = 1$ and $x \cdot u = 1$. We say that x is a qubit in the support of u. Recall from Lemma 2.2.4 that u has a unique paired basis vector $[v] \in H^1(C^\bullet)$ such that $[u] \cdot [v] = 1$. It is possible to *safely correct* a qubit x when there is a vector $v \in [v]$ such that $x \cdot v = 1$ and $y \cdot v = 0$ for all other qubits y in the support of u. We say that u is *gauge-fixable* when it is possible to safely correct all qubits in the support of u.

Example 2.4.2. Consider the Shor code from Example 2.2.7 and Section 2.3.5. The $\overline{Z}$ operator

$$v = \begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\end{pmatrix}^\intercal$$

has qubits in its support for which it is not possible to safely correct, as there are only 4 representatives of the nonzero equivalence class $[w] \in H^1(C^\bullet)$ but 9 qubits for which being able to safely correct is necessary. However, it is possible to safely correct all qubits in the support of the $\overline{Z}$ operator

$$u = \begin{pmatrix}1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0\end{pmatrix}^\intercal,$$

where $u \in [v]$, with the fixing operators:

$$\begin{pmatrix}1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}^\intercal; \quad \begin{pmatrix}0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0\end{pmatrix}^\intercal; \quad \begin{pmatrix}0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1\end{pmatrix}^\intercal$$

The same definition of gauge-fixability applies if we exchange X and Z appropriately.

Lemma 2.4.3. Every irreducible logical operator is also gauge-fixable, and vice versa.

Proof. See Appendix 6. ■

Next we will require the tensor product of chain complexes, for which see Definition 1.4.8.

Definition 2.4.4. Let $V_\bullet = \; V_1 \longrightarrow V_0$ and $P_\bullet = \; P_1 \xrightarrow{\begin{pmatrix}1\\1\end{pmatrix}} P_0$ be length 1 chain complexes. Then we can make the tensor product chain complex $W_\bullet = (P \otimes V)_\bullet$. Explicitly,

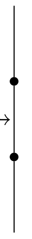
$$W_\bullet = \; W_2 \longrightarrow W_1 \longrightarrow W_0$$

with

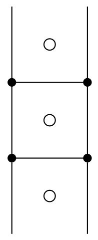
$$W_2 = P_1 \otimes V_1 = V_1; \quad W_1 = (P_0 \otimes V_1) \oplus (P_1 \otimes V_0) = (\mathbb{F}_2^2 \otimes V_1) \oplus V_0; \quad W_0 = P_0 \otimes V_0 = \mathbb{F}_2^2 \otimes V_0$$

Also, $\partial_2^{W_\bullet} = \begin{pmatrix}\mathrm{id}_{V_1}\\ \mathrm{id}_{V_1}\\ \partial_1^{V_\bullet}\end{pmatrix}$ and $\partial_1^{W_\bullet} = \begin{pmatrix}\mathrm{id}_{\mathbb{F}_2^2} \otimes \partial_1^{V_\bullet} & \partial_1^{P_\bullet} \otimes \mathrm{id}_{V_0}\end{pmatrix} = \begin{pmatrix}\partial_1^{V_\bullet} & 0 & \mathrm{id}_{V_0}\\ 0 & \partial_1^{V_\bullet} & \mathrm{id}_{V_0}\end{pmatrix}$.

In the case where $V_\bullet$ is a string along a patch of surface code, say of the form:

$V_\bullet \rightsquigarrow$ 

then $W_\bullet$ will be of the form

$W_\bullet \rightsquigarrow$ 

as a square lattice, see Definition 1.8. We can see this as the 'intermediate section' used to perform lattice surgery.

Lemma 2.4.5. Let $V_\bullet$ be a $\overline{Z}$ logical operator subcomplex of a chain complex $C_\bullet$, and let $V_\bullet$ satisfy the irreducibility property from Definition 2.3.13. Then

$$w^X_W = w^X_C + 1; \quad w^Z_W = q^X_C + 2; \quad q^X_W = \max(q^X_C, 2); \quad q^Z_W = w^X_C$$

and $\dim H_1(W_\bullet) = \dim H_1(V_\bullet) = 1$.

Proof. Observe that $\partial^{V_\bullet}_1$ has maximum row weight w^X_C and column weight q^X_C. Then inspect the matrices $\partial^{W_\bullet}_2$ and $\partial^{W_\bullet}_1$ from Definition 2.2.3. For $\dim H_1(W_\bullet)$, we use the Künneth formula, for which see Lemma 1.4.10, which in this case says $H_1((P \otimes V)_\bullet) = (H_0(P_\bullet) \otimes H_1(V_\bullet)) \oplus (H_1(P_\bullet) \otimes H_0(V_\bullet))$. We then have

$$\dim H_0(P_\bullet) = 1; \quad \dim H_1(P_\bullet) = 0; \quad \dim H_0(V_\bullet) = 0; \quad \dim H_1(V_\bullet) = 1$$

where the last comes from the fact that $V_0 = \mathrm{im}(\partial^{V_\bullet}_1)$, using Definition 2.3.10. $\dim H_1(V_\bullet) = 1$ as $B_1(V_\bullet) = 0$ and $Z_1(V_\bullet) = 1$. ■

Definition 2.4.6. Let $V_\bullet$ be a simultaneous $\overline{Z}$ logical operator subcomplex of both $C_\bullet$ and $D_\bullet$, satisfying the irreducibility property. Then define the 'sandwiched code' $(T_\bullet, T^\bullet)$, with $T_\bullet$ as the pushout of a pushout:

$$\begin{tikzcd}
 & V_\bullet \arrow[r, hook] \arrow[d, hook] & C_\bullet \arrow[d] \\
V_\bullet \arrow[r, hook] \arrow[d, hook] & W_\bullet \arrow[r] & R_\bullet \arrow[d] \arrow[ul, phantom, "\ulcorner", very near start] \\
D_\bullet \arrow[rr] & & T_\bullet \arrow[ul, phantom, "\ulcorner", very near start]
\end{tikzcd}$$

where the middle term is $W_\bullet = (P \otimes V)_\bullet$ from Definition 2.4.4 above, and the two inclusion maps $V_\bullet \hookrightarrow W_\bullet$ map V_1 into each of the copies of V_1 in W_1, and the same for V_0. All maps in the pushouts are basis-preserving, and one can check that they are all monic.

Colloquially, we are gluing first one side of the code $W_\bullet$ to $C_\bullet$, and then the other side to $D_\bullet$. [7]

Lemma 2.4.7. The 'sandwiched code' $(T_\bullet, T^\bullet)$ has

$$n_T = n_C + n_D + r; \quad k_T \geq k_C + k_D - 1$$

and

$$w^X_T \leq w^X_{C\oplus D} + 1; \quad w^Z_T \leq \max(w^Z_{C\oplus D}, q^X_{C\oplus D} + 2); \quad q^Z_T \leq q^Z_{C\oplus D} + w^X_{C\oplus D}; \quad q^X_T = \max(q^X_{C\oplus D}, 2).$$

If $k_T = k_C + k_D - 1$ then $d^X_T \geq \min(d^X_C, d^X_D)$.

Proof. For n_T, just apply Lemma 2.3.14 twice. For k_T, use Lemma 2.4.5 and apply Lemma 2.3.14 twice.

For d^X_T, we first show that $(R_\bullet, R^\bullet)$ has $d^X_R \geq d^X_C$. Every $\overline{X}$ operator in $(W_\bullet, W^\bullet)$ must anticommute with the $\overline{Z}$ operator used to construct $V_\bullet$, and thus must have support on those qubits. In addition, it must have a matched $\overline{X}$ operator in $(C_\bullet, C^\bullet)$, which also has

[7] We could equally do it the other way, in which case the two pushouts would be flipped, but this does not change $T_\bullet$.

support on those qubits. As the only other $\overline{X}$ operators in $(R_\bullet, R^\bullet)$ are those in $(C_\bullet, C^\bullet)$ which are unaffected by the merge, having no support on the qubits being merged, $d^X_R \geq d^X_C$. Then, if $k_T = k_C + k_D - 1$, $d^X_T \geq \min(d^X_C, d^X_D)$ using Lemma 2.3.17.

For w^X_T, the pushouts will glue each X type stabiliser generator in $W_\bullet$ into those in $C_\bullet$ and $D_\bullet$ in such a way that they will have exactly one extra qubit in the support, by the product construction of $W_\bullet$; we can see this from $\partial^{W_\bullet}_1$ in Definition 2.4.4, as there is exactly a single 1 which is not part of the $\partial^{V_\bullet}_1$ in any given row of the matrix.

For w^Z_T, q^Z_T and q^X_T we just use Lemma 2.4.5 and apply Lemma 2.3.19 twice. ■

The intuition here is that rather than gluing two codes $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ together directly along a logical operator, we have made a low distance hypergraph code $(W_\bullet, W^\bullet)$ and used that to sandwich the codes. A consequence of the above lemma is that this 'sandwiching' procedure maps LDPC codes to LDPC codes. Importantly, under suitable conditions the two pushouts let us perform a code map on logical qubits in an error-corrected manner.

Definition 2.4.8. Let $(T_\bullet, T^\bullet)$ have no logical $\overline{Z}$ operators with weight lower than $d_{C\oplus D}$. Then we say that the merged code has *distance bounded below.*

Remark 2.4.9. Note that the only Z operators which can lower the distance are those with support on the logical $\overline{Z}$ which is used to construct $V_\bullet$, as all others will be unchanged by the quotient. The condition for a merge to have distance bounded below is quite a tricky one, as we do not know of a way to check this easily. Because of Lemma 2.4.7, this problem is isolated to $\overline{Z}$ operators, as the distance is guaranteed to be bounded below for $\overline{X}$ operators.

Proposition 2.4.10. Let $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ be CSS codes which share an irreducible $\overline{Z}$ operator on m physical qubits and r X-type stabiliser generators each; let the relevant logical qubits be i and j, and let $V_\bullet$ be the logical operator subcomplex of $C_\bullet$ and $D_\bullet$ such that the codes admit an irreducible $\overline{Z}$-merge. Further, let d be the code distance of $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$, and let the merged code $(T_\bullet, T^\bullet)$ have distance bounded below. Then there is an error-corrected procedure with distance d for implementing a $\overline{Z} \otimes \overline{Z}$ measurement on the pair i, j of logical qubits, which gives the code $(T_\bullet, T^\bullet)$. This procedure requires r auxiliary clean qubits and an additional m Z-type stabiliser generators.

Proof. We aim to go from the code $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ to $(T_\bullet, T^\bullet)$. The code map we apply to physical qubits is as follows. We call the physical qubits in the support of the logical operators to be glued together the *participating* qubits. We initialise a fresh qubit in the $|+\rangle$ state for each pairing of X-measurements on the two logical operators of qubits i and j, that is for each qubit in $(W_\bullet, W^\bullet)$ which is not glued to a qubit in $(C_\bullet, C^\bullet)$ or $(D_\bullet, D^\bullet)$.

We now modify the stabilisers to get to $(T_\bullet, T^\bullet)$. To start, change the X stabiliser generators with support on the participating qubits to have one additional fresh qubit each, so that each pairing of X-measurements shares one fresh qubit. We add a new Z stabiliser generator with weight $a + 2$ for each participating qubit in one of the logical operators to be glued, where a is the number of X type generators of which that physical qubit is in the support. One can see this using Definition 2.4.4, as on the middle code $(W_\bullet, W^\bullet)$ we have

$$P_Z = (\partial^{W_\bullet}_2)^\intercal = \begin{pmatrix} \mathrm{id}_{\mathbb{F}_2^m} & \mathrm{id}_{\mathbb{F}_2^m} & (\partial^{V_\bullet}_1)^\intercal \end{pmatrix}$$

We then measure d rounds of all stabilisers. All of the qubits in the domain of the last block of P_Z above are those which were initialised to $|+\rangle$. The only other qubits which contribute to the new Z stabiliser generators are those on either side of the sandwiched code, i.e. those along the $\overline{Z}$ logical operators of qubits i and j. Each of the physical qubits in the support of these logical operators is measured exactly once by the new Z stabiliser generators, and they are measured in pairs, one from each side; therefore performing these measurements and recording the total product is equivalent to measuring $\overline{Z} \otimes \overline{Z}$. We will now check this, and verify that it maintains error-correction.

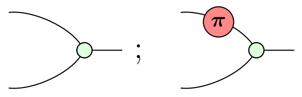
Let the outcome of a new Z-type measurement be $c_\lambda \in \{1, -1\}$, and the overall outcome $c_L = \prod_{\lambda \leq m} c_\lambda$. Whenever $c_\lambda = -1$ we apply the gauge fixing operator $X_\lambda = \bigotimes_{(i \in v \ | \ v_i = 1)} X_i$ for the specified $v \in C^0$ (or one could choose a gauge fixing operator using D^0 instead). We let $X_{c_L} = \prod_{(\lambda \ | \ c_\lambda = -1)} X_\lambda$. On participating physical qubits, the merge is then

$$X_{c_L} \prod_\lambda \frac{I + c_\lambda Z}{2} = \prod_\lambda \frac{I + Z}{2} X_{c_L}$$

where we abuse notation somewhat to let I and Z here refer to tensor products thereof. As each X_λ belongs to the same equivalence class of logical $\overline{X}$ operators in $H_1(C_\bullet)$, if $c_L = 1$ then X_{c_L} acts as identity on the logical space; if $c_L = -1$ then X_{c_L} acts as $\overline{X}$ on logical qubit i in the code before merging. One can then see that these two branches are precisely the branches of the logical $\overline{Z} \otimes \overline{Z}$ measurement. As the measurements were performed using d rounds of stabilisers, and the gauge fixing operators each have support on at least d qubits, the overall procedure is error-protected with code distance d.

We also check that the procedure is insensitive to errors in the initialisation of fresh qubits. If a qubit is initialised instead to $|-\rangle$, or equivalently suffers a Z error, then the new Z stabiliser measurements are insensitive to this change, and it will just show up at the X measurements on either side of the fresh qubit. If it suffers some other error, say sending it to $|1\rangle$, then each new stabiliser measurement with that qubit in its support may have its result flipped. By construction of $V_\bullet$, each fresh qubit is in the support of an even number of new Z stabiliser measurements, and so initialising the fresh qubits incorrectly will not change c_L. ■

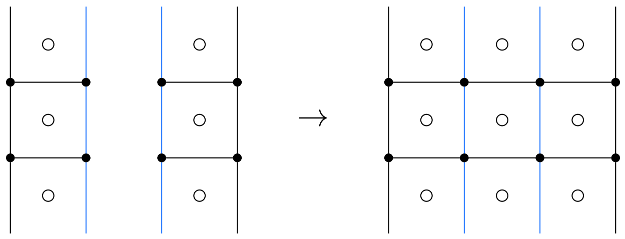
As ZX diagrams, the branches are:



on logical qubits i and j, and all other logical qubits in the code are acted on as identity. We can freely choose which logical qubit may have the red π spider, as it will differ only up to a red π – i.e. a logical $\overline{X}$ – on the output logical qubit. In practice, depending on the code there will typically be cheaper ways of fixing the gauges than using an $\overline{X}$ logical operator for each -1 outcome, as there could be an $\overline{X}$ logical operator which has support on multiple of the qubits belonging to new stabilisers. Moreover, one can just update the Pauli frame rather than apply any actual $\overline{X}$ logical operators. The ability to do so is necessary, however, so that the -1 outcome is well-defined.

The protocol obviates the problem of performing the code map on *physical* qubits deterministically, as the only non-isometric transformations we perform are measurements of stabiliser generators. However, the code map on *logical* qubits is still not isometric, hence we have a logical measurement.

For the prototypical example of lattice surgery we then have:

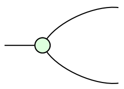


We also look at a less obvious example, that of error-corrected surgery of the Shor code, in Appendix 7.

By dualising appropriately one can perform an $\overline{X}$-merge by sandwiching in a similar manner. We can also do the 'inverse' of the merge operation:

Corollary 2.4.11. Let $(T_\bullet, T^\bullet)$ be a CSS code formed by sandwiching codes $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$ together along a $\overline{Z}$ operator. Then there is an error-corrected procedure to implement a code map on logical qubits $\mathcal{E}_{\overline{X}}$ from $(T_\bullet, T^\bullet)$ to $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$.

Proof. As the initial code is already a sandwiched code we can just take the opposite of sandwiching. We delete the qubits belonging to the intermediate code $(W_\bullet, W^\bullet)$ but not $(C_\bullet, C^\bullet)$ or $(D_\bullet, D^\bullet)$ by measuring them out in the X-basis. The code map $\mathcal{E}_{\overline{X}}$ on participating logical qubits is



by following precisely the same logic as for traditional lattice surgery [HFDM12]. ■

Again, by dualising appropriately we get the last split operation.

Given a procedure for making $\overline{Z} \otimes \overline{Z}$ and $\overline{X} \otimes \overline{X}$ logical measurements and the isometries from splits, one can easily construct a logical CNOT between suitable CSS codes following, say, [dBH20] and observing that the same ZX diagrammatic arguments apply. Augmented with some Clifford single-qubit gates and non-stabiliser logical states one can then perform universal computation. As opposed to some other methods of performing entangling gates with CSS codes, e.g. transversal 2-qubit gates, the schemes above require only the m qubits from the respective $\overline{Z}$ or $\overline{X}$ operators to participate, and we expect $m \ll n$ for practical codes. Unlike that of [CKBB22], our method does not require a large ancillary hypergraph product code, which can have significantly worse encoding rate and code distance scaling than the LDPC codes holding data – the tradeoff is that we cannot generally prove that the code distance will be maintained. Our method does not require the code to be 'self-ZX-dual' in the sense of [BB24], and unlike [HJY23] our method does not require the code to be defined on any kind of manifold.

2.5 Conclusions and further work

The pushouts we gave along logical operators are the most obvious cases. By taking pushouts of more interesting spans other maps on logical data can be obtained, although by Proposition 2.2.12 and Corollary 2.2.17 all code maps as we defined them are limited and do not allow for universal quantum computation on their own; we also do not know whether other pushouts would allow the maps on logical data to be performed fault-tolerantly.

In this Chapter we assumed that the two codes being ‘glued’ are different codes, but the same principles apply if we have only one code we would like to perform internal surgery on. In this case, the correct universal construction to use should be a coequaliser. We meet this case in Chapter 3. It should be possible to extend the definitions of $\overline{X}$- and $\overline{Z}$-merges straightforwardly to include metachecks [Cam19], by specifying that the logical operator subcomplex $V_\bullet$ now runs from V_1 to V_{-1}, so it has X-checks and then metachecks on X-checks, but we have not proved how this affects metachecks in the merged code.

There are several ways in which our constructions could be generalised to other codes. The obvious generalisation is to qudit CSS codes. For qudits of prime dimension q, everything should generalise fairly straightforwardly using a different finite field $\mathbb{F}_q$ but in this case the cell complexes will require additional data in the form of an orientation on edges, as is familiar for qudit surface codes. When q is not prime, one formalism for CSS codes with dimension q is chain complexes in $\mathbb{Z}_q$-$\mathtt{FFMod}$, the category of free finite modules over the ring $\mathbb{Z}_q$ [Nov24]. As $\mathbb{Z}_q$ is not generally a domain this complicates the homological algebra.

Second, if we wish to upgrade to more general stabiliser codes we can no longer use chain complexes. The differential composition $P_X P_Z^\intercal$ is a special case of the symplectic product $\omega(M,N) = M\omega N^\intercal$ for $\omega = \begin{pmatrix} 0_n & I_n \\ -I_n & 0_n \end{pmatrix}$ [Haa16], but by generalising to such a product we lose the separation of Z and X stabilisers to form a pair of differentials. It is unclear what the appropriate notion of a quotient along an $\overline{X}$ or $\overline{Z}$ operator is for such codes.

Chapter 3

SSIP: automated surgery with quantum LDPC codes

3.1 Introduction

There are several desiderata for logical operations on codes. They should:

- Yield universality – commonly in conjunction with state injection.
- Be individually addressable on logical qubits.
- Be parallelisable.
- Not add significant overhead to the quantum memory – in terms of qubit count, stabiliser weight, reduction in threshold etc.

We argue that generalised surgery can satisfy these desiderata. Here we present `SSIP`, software which automates the procedure of identifying and performing CSS code surgery. While we focus on the homological formalism in [CB24], the software is also capable of performing some of the surgeries in [CKBB22] by converting the protocols defined using Tanner graphs into chain complexes. `SSIP` has been extensively tested and benchmarked, and we find that it is fast (and correct) on small-to-medium sized codes, while using lower resource requirements than previously estimated [BCGMRY24].

The layout of this Chapter is as follows. We start by explaining how `SSIP` determines if two logical operators from different codeblocks can be merged together, yielding a logical parity measurement. This is an external code merge. `SSIP` does not explicitly handle code splits, the adjoint operation to merges, because once a merge has been found we have all the data required for its corresponding split. Upon performing a merge, `SSIP` can optionally compute substantial additional data, such as which new ancillae data qubits, stabilisers and logical qubits are introduced. We illustrate first with some very small codes, including mildly interesting cases where we merge a triorthogonal code into other quantum memories, allowing for magic state injection without distillation. All examples can be found in the Github repository `https://github.com/CQCL/SSIP`. We then give results for a variety of external merges with lift-connected surface codes [ORM24], generalised bicycle codes [KP13], and bivariate bicycle codes [BCGMRY24].

We perform logical single-qubit and parity measurements in the X and Z bases. After a merge is performed, we must ensure that the code distance is preserved; for small codes this is straightforward to calculate using naive methods, such as enumerating over all logical operators, but for codes with blocklengths in the hundreds of qubits such methods would take too long. We use `QDistRnd` [PSK22] to upper bound the code distances. Where possible we use the Satisfiability Modulo Theories (SMT) solver Z3 [dMB08] to give explicit distances.

We demonstrate the developed techniques on the $[\![144, 12, 12]\!]$ gross code from [BCGMRY24], which belongs to a family of bivariate bicycle codes. We find that we can measure any logical qubit simultaneously in either the X or Z basis while maintaining $d = 12$, when viewed as a subsystem code, using at most an additional 78 data qubits, and 72 syndrome qubits, so 150 total ancillae. This is substantially lower than the 1380 additional qubits described in [BCGMRY24, Sec. 9.4]. These results on the gross code are reliant on the upper bound from `QDistRnd` being tight.

We then describe how one can perform internal merges within a CSS codeblock. Happily, the procedure is very similar to external merges, which we previously described in [CB24], with almost the same prerequisites. We find that we can perform pairwise parity measurements between many (but not all) of the logical qubits in the gross code in either basis using a total of at most 150 extra qubits, maintaining code distance.

Importantly, there are several things which `SSIP` does *not* do. For fault-tolerance, we must give circuits for syndrome measurements and other operations on the codes, and then use an accurate error model to establish a (pseudo-)threshold [KLZ96]. Without circuits, one can attempt to approximate the error tolerance of the code using phenomenological noise. `SSIP` does neither of these things; it does not include methods for constructing quantum circuits or modelling noise in any way. Additionally, codes should come with good decoders, allowing us to extract a likely error from the outcomes of syndrome measurements [PK21, WB24, RWBC20]. `SSIP` does not perform any decoding. Lastly, quantum architectures commonly have geometric constraints, which put conditions on the Tanner graphs of any implemented CSS codes. `SSIP` has no notion of geometric constraints or architectures. For simplicity, it is solely concerned with code parameters and figures of merit, such as code distances and stabiliser weights.

3.1.1 Related work

Cohen et al [CKBB22] first published generalisations of lattice surgery to arbitrary CSS codes. Our work is directly inspired by theirs, although our approach is homological rather than using Tanner graphs. Our surgeries, when performing logical parity measurements and their adjoint splits, are also different. This makes performing an apples-to-apples comparison between the two difficult, but we do present some comparisons to their approach.

There are several different open-source repositories for reasoning about quantum CSS and LDPC codes [Sab, Per23, Rof22]. These have different foci, and to the best of our knowledge none are designed to reason about surgery.

Separately, LaSsynth has recently been developed for synthesising lattice surgeries [TNG24]. This has a different scope to the present work. LaSsynth takes a desired quantum routine and synthesises it into a sequence of lattice surgery operations, encoding the synthesis problem as a SAT instance to exhaustively optimise the resources. It is designed exclusively for surface codes. In a similar vein see [Wat24]. `SSIP` cannot compile quantum routines, beyond a specified merge/split or sequence of merges/splits. It could be fruitful to attempt to optimise resources when performing surgery with more elaborate codes than surface codes in a similar manner.

Shortly after the preprint for this Chapter appeared on arXiv, a preprint appeared which has some crossover with the present work [CHRY24]. In this later preprint, the authors prove that by gauge-fixing the new logicals present in merged codes one can prove bounds on the size of the ancilla patch, conditional on the expansion properties of a certain graph. As an application the authors focus on the gross code of [BCGMRY24], while we benchmark a variety of different codes. Later works then further reduced the overheads [IGND24, WY24].

3.1.2 General software description

The software is called Safe Surgery by Identifying Pushouts (`SSIP`) because its core function is to find pushouts, and other colimits, between codes in order to perform surgery. This surgery is 'safe' in the sense that it is guaranteed to perform logical measurements on the logical qubits involved in the merge, without affecting logical data elsewhere in the code. It is also safe in the sense that the distance can be checked afterwards, although this becomes challenging for codes at high blocklengths and distances. As a consequence, `SSIP` is best suited to codes with blocklengths in the low hundreds, i.e. for near-term fault-tolerant computing.

`SSIP` is written in Python for ease of use, with occasional function calls to a library in GAP [PSK22] for code distance estimates. `SSIP` is available from its Github repository `https://github.com/CQCL/SSIP` or alternatively by calling `pip install ssip`, and is fully open-source, released with a permissive MIT license. Documentation can be found at `https://cqcl.github.io/SSIP/api-docs/`.

For simplicity, `SSIP` is entirely procedural. The only new classes defined in `SSIP` are structs[1], such as the `CSScode`, which merely contains the two parity-check matrices of a CSS code, stored as `numpy` arrays. The codebase then operates by performing numerics in functions, passing around the `CSScode` and other elementary data structures.

There are four main purposes of `SSIP`:

- Determine whether, given suitable data, a code merge is possible (and hence its adjoint split).
- Perform code merges and return merged codes.
- Calculate additional data about the merge, such as the new stabilisers, data qubits and logical qubits introduced.
- Calculate code parameters, either as CSS codes or as subsystem codes, once merges have been performed.

All of these are extremely tedious to compute by hand, and so software is required for practically relevant codes.

We will describe how `SSIP` performs all of these steps, but in order to explain our algorithms and results we must give some algebraic background on CSS codes and surgery.

3.2 The CSS code-homology correspondence

First, recall the definition of CSS codes in terms of chain complexes from Chapter 2. In a slight departure, we define CSS codes only in terms of a single chain complex, rather than the chain complex and its dual cochain complex, to lighten notation.

[1]Python does not have structs, but as of Python 3.7 it has dataclasses, which are close to structs.

Recall that a CSS code is called *ω-limited* when the weights of rows and columns in both parity-check matrices are bounded above by ω. It is common to consider infinite families of codes of increasing size. If every member of the family is ω-limited for some finite ω then the family is called quantum Low-Density Parity Check (qLDPC). A similar definition applies to classical LDPC codes.

Throughout, we will refer to the ω of a code as being the maximum column or row weight of its parity-check matrices.

We will also use subsystem codes in this Chapter, for which recall Definition 2.2.6.

3.2.1 Code distance

We take a brief aside to discuss the calculation of minimum distance for CSS codes. There are at least 5 ways to perform this calculation for CSS codes, without relying on the codes being 2D surface codes using e.g. [BVCKT17, App. B].

1. Enumerate over all logicals in the code and find the one(s) with the lowest weight. The compute time of this will generally scale exponentially in n, and so it can only reasonably be used for small codes. Evidently the compute time is insensitive to d, as every operator is checked regardless.

2. Start by searching for any weight 1 logicals and then increment the weight until a logical is found. This will also generally scale poorly but will perform better for codes with low d, even if n is high.

3. Use `QDistRnd` to give an upper bound on the code distance [PSK22]. To calculate d_Z `QDistRnd` constructs a generator matrix whose rows are a basis of $\ker(P_X)$, then randomly permutes columns, performs Gaussian elimination, and un-permutes the columns, leaving a random set of rows in $\ker(P_X)$. Rows not in $\mathrm{im}(P_Z^\intercal)$ are then considered for their lowest weight. The permutation is applied many times, improving the upper bound on d_Z. The same can then be done for d_X.

4. Interpret the minimum distance problem as a binary programming problem. This is a somewhat less common problem than the mixed integer programming problem, so in certain cases one can convert the former into the latter to make use of mixed integer programming solvers [LAR11, Sec. C 1.].

5. Perform distance verification with ensembles of codes with related properties [DKP17].

`SSIP` makes use of the first three methods. In our results, for small codes or codes for which we already know the distance is modest, we use (1.) and (2.). For some codes we upper bound the distance using (3.) first and then verify that the bound is tight using (2.). In `SSIP` we offload the computation of finding logicals in (2.) to Z3 [dMB08], which is written in C++ and so significantly faster than it would be to find logicals in Python. For the largest codes, in the hundreds of qubits, we merely estimate the distance using (3.). While there are no guarantees, we find that empirically `QDistRnd` is accurate compared to exact results given a large enough number of information sets, say 10^4 for codes in the low hundreds of qubits.

We will also calculate the distance of subsystem CSS codes. For computations with methods (1.) and (2.) nothing much changes, we just add conditions to the logicals to consider. For method (3.), we check in Appendix 8 that any black box method for calculating the code distance of a CSS code can be adapted to subsystem CSS codes, and so by changing the input given to `QDistRnd` we can also use (3.) to upper bound subsystem CSS code distances.

We do not use methods (4.) and (5.) in our results. To the best of our knowledge, the conversion in (4.) to mixed integer programming requires the codes to have regular stabiliser weights, which merged codes will generally not have. The methods in (5.) also require the codes to have a certain structure.

3.2.2 Lifted products

Lifted products are a mild generalisation of tensor products, and we will make use of lifted products extensively in our set of examples. Unlike tensor products, they are not a necessary ingredient of our constructions, but many lifted product codes have good parameters – in both the formal and informal senses [PK22A, PK22B, PK21, KP13, ORM24, BCGMRY24, LP24, SHR24] – and so are a useful class of codes on which to demonstrate our methods.

Recall that a chain complex is well-defined over any ring R. Differentials are R-module homomorphisms. We assume that the components of the chain complexes are all free R-modules of finite rank.

Then, fix R to be a commutative subring of $\mathcal{M}_\ell(\mathbb{F}_2)$, the ring of ℓ-by-ℓ matrices over $\mathbb{F}_2$, with a specified basis. The tensor product of chain complexes is also well-defined for the ring R. Taking two chain complexes over R and making the tensor product $(C \underset{R}{\otimes} D)_\bullet$, this tensor product is also a valid chain complex when replacing each entry of the differentials in R with its corresponding matrix over $\mathbb{F}_2$, and considering the whole chain complex over $\mathbb{F}_2$. This is the lifted product. Explicitly, we have

$$(C \underset{R}{\otimes} D)_\bullet = \; C_1 \underset{R}{\otimes} D_1 \longrightarrow C_0 \underset{R}{\otimes} D_1 \oplus C_1 \underset{R}{\otimes} D_0 \longrightarrow C_0 \underset{R}{\otimes} D_0$$

when the input chain complexes $C_\bullet$ and $D_\bullet$ are length 1 chain complexes over R, that is two classical codes with a free, coherent R-action. We do not generally know *a priori* what the code parameters k and d of the lifted product code $(C \underset{R}{\otimes} D)_\bullet$ will be when viewed over $\mathbb{F}_2$, a marked difference from the tensor product. The straightforward facts derived from the Künneth formula only apply to the complex viewed over R, and do not easily translate to $\mathbb{F}_2$.

When $R = \mathbb{F}_2$, the lifted product coincides with the tensor product. Lifted products are special cases of balanced products [BE21B] where the actions are free. A common ring to use for generating codes is $\mathscr{C}_\ell$, the ring of ℓ-by-ℓ circulant matrices. This is guaranteed to be commutative, so the tensor product is defined. Helpfully, $\mathscr{C}_\ell \cong \mathbb{F}_2^{\langle \ell \rangle}$, where $\mathbb{F}_2^{\langle \ell \rangle} := \mathbb{F}_2[x]/(x^\ell - 1)$ is the ring of polynomials over $\mathbb{F}_2$ modulo $x^\ell - 1$, by sending the mth shift matrix to x^m. This means that any circulant matrix can be denoted concisely by its corresponding polynomial.

`SSIP` can generate a variety of different lifted products, but there are several families of lifted product codes which one would have to generate elsewhere and import. Given that a `CSScode` in `SSIP` is just a pair of `numpy` arrays, this is straightforward.

3.2.3 CSS code surgery

We recap the surgery from Chapter 2, but also extend some definitions for single qubit measurements and larger ancilla patches.

The category $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ has as objects chain complexes over $\mathbb{F}_2$. Morphisms are chain maps, matrices between components at the same degree, such that the matrices are coherent in the sense that we have the following commuting squares:

$$\begin{tikzcd}
\cdots \arrow[r] & C_{n+1} \arrow[r, "\partial^{C_\bullet}_{n+1}"] \arrow[d, "f_{n+1}"] & C_n \arrow[r, "\partial^{C_\bullet}_{n}"] \arrow[d, "f_n"] & C_{n-1} \arrow[r] \arrow[d, "f_{n-1}"] & \cdots \\
\cdots \arrow[r] & D_{n+1} \arrow[r, "\partial^{D_\bullet}_{n+1}"] & D_n \arrow[r, "\partial^{D_\bullet}_{n}"] & D_{n-1} \arrow[r] & \cdots
\end{tikzcd}$$

Definition 3.2.1. (Basis-preserving) We say that a chain map is basis-preserving when each matrix sends basis elements to basis elements, i.e. they are functions on basis elements.

We can use universal properties in $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$ to construct new codes from old ones; in particular, we can perform surgery between CSS codes by using pushouts and coequalisers. We give a quick recap of this procedure here. See [CB24] for a more detailed explanation.

We assume that we are performing $\overline{Z}$-parity measurements, which merge codes in a manner which uses $\texttt{Ch}(\texttt{Mat}_{\mathbb{F}_2})$; $\overline{X}$-parity measurements can be inferred by duality, using the category of cochain complexes instead. Similarly, splits of codes can be inferred by reversing the procedure.

Definition 3.2.2. Let $v \in \ker(P_X)\backslash\mathrm{im}(P_Z^\intercal)$ be such that no other vector in $\ker(P_X)$ is contained in the support of v. Then we call v an irreducible logical operator.

Definition 3.2.3. (Logical operator subcomplex) Given a logical $\overline{Z}$ operator $v \in \ker(P_X)\backslash\mathrm{im}(P_Z^\intercal)$, we can construct a chain complex which represents this operator and its stabilisers, in a suitable sense.

$V_\bullet = V_1 \to V_0$, where:

$$\tilde{V}_1 = \text{supp } v; \quad \partial_1^{V_\bullet} = \partial_1^{C_\bullet} \restriction_{\text{supp } v}; \quad \tilde{V}_0 = \bigcup_{u \in \mathrm{im}(\partial_1^{V_\bullet})} \text{supp } u$$

where supp v is the set of basis vectors in the support of v, and $\partial_i \restriction_S$ is the restriction of a differential to a subset S of its domain. $V_\bullet$ is called a logical operator subcomplex.

We have a suitable dualised definition for a logical $\overline{X}$ operator in $\ker(P_Z)\backslash\mathrm{im}(P_X^\intercal)$. We will make repeated use of logical operator subcomplexes throughout. Observe that this subcomplex only has one non-zero differential, i.e. it can be considered a classical code.

External merges

Given an irreducible logical operator v in a code we can construct its logical operator subcomplex $V_\bullet$. Then, if we have a monic span:

$$\begin{tikzcd}
V_\bullet \arrow[r, hook, "f_\bullet"] \arrow[d, hook, "g_\bullet"'] & C_\bullet \\
D_\bullet &
\end{tikzcd}$$

where both chain maps are basis-preserving, and $V_\bullet$ is a logical operator subcomplex in both codes $C_\bullet$ and $D_\bullet$, then the logical operator is 'present' in both codes in a suitable sense, and we can perform an external merge which performs a parity measurement on the two logical qubits.

We do this by first generating a new tensor product code.

Definition 3.2.4. Let $P_\bullet = \mathbb{F}_2^r \to \mathbb{F}_2^{r+1}$ be the classical code with parity-check matrix

$$\partial_1^P = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

i.e. $P_\bullet$ is the incidence matrix of the path graph $\mathcal{P}_{r+1}$. We call r the depth.

We can see that $\dim\ker(\partial_1^P) = 0$, and $\dim\ker((\partial_1^P)^\intercal) = 1$, with the non-zero codeword 1_{r+1}.

Let $W_\bullet = (P \otimes V)_\bullet$ be the new tensor product code. Explicitly it is the chain complex

$$\mathbb{F}_2^r \otimes V_1 \to \mathbb{F}_2^{r+1} \otimes V_1 \oplus \mathbb{F}_2^r \otimes V_0 \to \mathbb{F}_2^{r+1} \otimes V_0$$

with differentials

$$\partial_2^W = \begin{pmatrix} \mathrm{id}_{V_1} & 0 & 0 & \cdots \\ \mathrm{id}_{V_1} & \mathrm{id}_{V_1} & 0 & \cdots \\ 0 & \mathrm{id}_{V_1} & \mathrm{id}_{V_1} & \cdots \\ 0 & 0 & \mathrm{id}_{V_1} & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ \partial_1^V & 0 & 0 & \cdots \\ 0 & \partial_1^V & 0 & \cdots \\ 0 & 0 & \partial_1^V & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}, \quad \partial_1^W = \begin{pmatrix} \partial_1^V & 0 & 0 & 0 & \cdots & \mathrm{id}_{V_0} & 0 & 0 & \cdots \\ 0 & \partial_1^V & 0 & 0 & \cdots & \mathrm{id}_{V_0} & \mathrm{id}_{V_0} & 0 & \cdots \\ 0 & 0 & \partial_1^V & 0 & \cdots & 0 & \mathrm{id}_{V_0} & \mathrm{id}_{V_0} & \cdots \\ 0 & 0 & 0 & \partial_1^V & \cdots & 0 & 0 & \mathrm{id}_{V_0} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

We can then make the composition of two pushouts,

$$\begin{tikzcd} & V_\bullet \arrow[r, hook] \arrow[d, hook] & C_\bullet \arrow[d] \\ V_\bullet \arrow[r, hook] \arrow[d, hook] & W_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner"] & R_\bullet \arrow[d] \\ D_\bullet \arrow[rr] & & T_\bullet \arrow[ul, phantom, "\ulcorner"] \end{tikzcd}$$

The diagram is drawn with two different instances of $V_\bullet$ so that the diagram commutes.

The first inclusion of $V_\bullet$ into $W_\bullet$ sends V_1 into the 1st copy of V_1 in W_1 and the same for V_0 in W_0. The second inclusion of $V_\bullet$ sends V_1 into the $(r+1)$th copy of V_1 in W_1 and the same for V_0 in W_0. The inclusions of $V_\bullet$ into $C_\bullet$ and $D_\bullet$ are just inherited from the monic span above. As all these inclusions are basis-preserving, the code $T_\bullet$ can be uniquely defined up to relabelling of basis elements [CB24, Lemma 5.4], so all weight-related notions such as code distance, being ω-limited etc. are canonical.

In this way we make the merged code $T_\bullet$ from the initial codes $C_\bullet$ and $D_\bullet$, where two logical operators in $C_\bullet$ and $D_\bullet$ respectively have been quotiented into the same equivalence class, performing a $\overline{Z} \otimes \overline{Z}$ measurement. This is done purely by initialising new qubits and stabilisers, so can be done in a fully error-corrected fashion, assuming the merge retains the code distance.

This is not generally guaranteed, so we must check it separately. It is also possible to introduce new logical qubits when doing this merge, for reasons described in [CKBB22, Sec. C]. These new logicals are often of low weight, so it can be useful to switch to a subsystem code [KLP05], labelling the newly introduced logicals as gauge qubits. As we shall demonstrate, this frequently lets us increase the minimum distance of the merged code, which is now the minimum dressed distance of the subsystem code. When the depth $r = d$, the minimum distance of the codes beforehand, we assert that an external merge always maintains the code distance, when viewed as a subsystem code. For brevity we do not prove this here, but claim that it can be done by converting to the Tanner graph formalism and using similar arguments as in [CKBB22, Sec. IV] pertaining to 'cleaning' [BT09].

Increasing the depth r of the code $P_\bullet$ will increase the size of $W_\bullet$ and hence the number of new data qubits and stabilisers added to the code. We would like to do this if a low depth results in a low distance.

Single-qubit measurements

In [CKBB22] new tensor product codes are also adjoined to the initial codes to perform logical single-qubit measurements. We will now convert this protocol into the homological picture. They also use their framework to perform logical multi-qubit Pauli measurements. We omit these as they are harder to view in the homological picture, although for those measurements which still yield CSS codes we assert that it can be done.

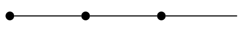
For single-qubit measurements, we only need one pushout. Again, say we are performing a $\overline{Z}$ measurement. Given an irreducible logical operator, we will make a new tensor product code.

Definition 3.2.5. Let $S_\bullet = \mathbb{F}_2^r \to \mathbb{F}_2^r$ be the classical code with parity-check matrix

$$\partial_1^S = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

with 1s on diagonal elements, and 1s on the entries below the diagonal, apart from the bottom-right diagonal entry which has no entry below it.

This is the incidence matrix of a 'truncated' path graph, where the last vertex has been removed but its dangling incident edge remains. For example, if $r = 3$ then the graph is



with

$$\partial_1^S = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Observe that $\ker(\partial_1^S) = \ker((\partial_1^S)^\intercal = 0$, i.e. the classical code and its dual have no codespace, for any r.

Given an irreducible logical operator subcomplex for a CSS code $C_\bullet$, we can then make the code $(S \otimes V)_\bullet$. Similar to with external merges above, we have an inclusion $V_\bullet \hookrightarrow (S \otimes V)_\bullet$, where V_1 is sent to the first copy of V_1 in $(S \otimes V)_1$, and V_0 is sent to the first copy of V_0 in $(S \otimes V)_0$.

We then have a basis-preserving monic span

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & (S \otimes V)_\bullet \\ C_\bullet & \end{tikzcd}$$

and so we can construct a new code by a single pushout

$$\begin{tikzcd} V_\bullet \arrow[r, hook] \arrow[d, hook] & (S \otimes V)_\bullet \arrow[d] \\ C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner", very near end] & R_\bullet \end{tikzcd} .$$

This time, $R_\bullet$ is the final code we are left with. We have initialised new qubits and stabilisers as dictated by $(S \otimes V)_\bullet$. As $(S \otimes V)_\bullet$ has no logical qubits, by the Künneth formula, we have quotiented the $\overline{Z}$ logical operator v, which was used to construct $V_\bullet$ and so $(S \otimes V)_\bullet$, into the $[0]$ equivalence class. In other words, measuring the stabilisers of the new code will also perform a $\overline{Z}$ measurement on that logical qubit. As before, it is possible to incidentally introduce new logical qubits in the process.

If $r = d$, the distance of the initial code $C_\bullet$, then $R_\bullet$ will always have minimum dressed distance d when viewed as a subsystem code [CKBB22, Thm. 1], setting new logical qubits to be gauge qubits. If $r < d$ then this can still be the case, but it is not guaranteed. We will show in later sections that it is common to be able to perform such logical single-qubit measurements without requiring high depth r.

Internal merges

We can also perform surgery within a single codeblock $C_\bullet$, taking two logical $\overline{Z}$ operators from different logical qubits and merging them together. As before, we start with an irreducible logical operator subcomplex. This time, however, we have the diagram

$$\begin{tikzcd} V_\bullet \arrow[r, hook, shift left, "f_\bullet"] \arrow[r, hook, shift right, "g_\bullet"'] & C_\bullet \end{tikzcd}$$

where $f_\bullet$ and $g_\bullet$ are basis preserving, and $\mathrm{im}(f_\bullet) \cap \mathrm{im}(g_\bullet) = 0$, i.e. there are no data qubits in the two logical operators which overlap, and the same for X stabilisers.

Remark 3.2.6. It is possible to relax this condition of no overlap, by observing that when there is overlap we can just take v being the logical operator $\overline{Z} \otimes \overline{Z}$. If v is irreducible, we can do 'single-qubit surgery' but for the two qubits, gluing in a single tensor product patch with $V_\bullet$ the logical operator subcomplex of v.

We can then construct the merged code using the same tensor product code $W_\bullet = (P \otimes V)_\bullet$ from Section 3.2.3. This time, the merged code is the result of two coequalisers:

$$\begin{tikzcd}
V_\bullet \arrow[r, hook, shift left] \arrow[r, hook, shift right] & (W \oplus C)_\bullet \arrow[r] & R_\bullet \arrow[r] & T_\bullet \\
& V_\bullet \arrow[ur, hook, shift left] \arrow[ur, hook, shift right] & &
\end{tikzcd}$$

As with external merges, the diagram is drawn with two separate instances of $V_\bullet$ so that the diagram commutes.

The first two inclusions on the left take $V_\bullet$ and map it into $W_\bullet$ and $C_\bullet$ respectively. The $C_\bullet$ inclusion is $f_\bullet$, and the $W_\bullet$ inclusion takes V_1 and V_0 to their first copies in W_1 and W_0, as with internal merges.

The second two inclusions of $V_\bullet$ into $R_\bullet$ are as follows. One is $g_\bullet$ composed with the inclusion $C_\bullet \to R_\bullet$, and the second is the $W_\bullet$ inclusion taking V_1 and V_0 to their $(r+1)$th copies in W_1 and W_0, composed with the inclusion $W_\bullet \to R_\bullet$.

The intuition is we glue first one side of $W_\bullet$ into $C_\bullet$ based on the irreducible logical operator, then the same thing with the other side. It may be instructive to instead view the two coequalisers as a single pushout as follows:

$$\begin{tikzcd}
(V \oplus V)_\bullet \arrow[r, hook] \arrow[d, hook] & W_\bullet \arrow[d] \\
C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner", very near end] & T_\bullet
\end{tikzcd}$$

where the same data is contained in the universal construction. The inclusion $(V \oplus V)_\bullet \hookrightarrow W_\bullet$ takes one $V_\bullet$ to the first copy in $W_\bullet$, and the second $V_\bullet$ to the $(r+1)$th copy. The inclusion $(V \oplus V)_\bullet \hookrightarrow C_\bullet$ maps each $V_\bullet$ to the chosen logical operators to merge.

In `SSIP`, the merged code is constructed using the two coequalisers diagram, so we stick with this picture.

Of course, we can view any external merge as an internal merge by setting $C_\bullet = (D \oplus E)_\bullet$ for some pair of codeblocks $D_\bullet$, $E_\bullet$. As for external merges, when the depth $r = d$, the minimum distance of the codes beforehand, we assert that an internal merge always maintains the code distance, when viewed as a subsystem code.

3.3 Automated external surgery

We can now explain how `SSIP` applies these universal constructions to perform surgery and extract useful data from merged codes. We start with external surgery, present results for external surgeries, then move on to internal surgery.

The basic data given to Algorithm 1 for performing external surgery is as follows:

- The parity-check matrices of the two codes $C_\bullet$, $D_\bullet$ to be merged.
- The two irreducible logicals $u \in C_1$ and $v \in D_1$ we would like to merge.
- The basis (Z or X) to perform the merge in.
- The desired depth r of the merge.

The codes are entered as `CSScode` objects, while the logicals are vectors, and the depth is an unsigned integer. Verifying that a vector is an irreducible logical is straightforward linear algebra and is efficient to calculate, so we do not include this in the algorithm. We assume that the chosen basis is Z; as always, the X version can be obtained by dualising to cochain complexes.

Algorithm 1 External merge calculation

```
$RM_1 \leftarrow$ RestrictedMatrix$(u, \partial_1^C)$
$RM_2 \leftarrow$ RestrictedMatrix$(v, \partial_1^D)$
Span $\leftarrow$ FindMonicSpan$(RM_1, RM_2)$
**if** Span is None **then**
    **return** None
$V_\bullet \leftarrow RM_1$
$P_\bullet \leftarrow$ ConstructP$(r)$
$W_\bullet \leftarrow (P \otimes V)_\bullet$
NewSpan1 $\leftarrow$ LHSspan(Span, $W_\bullet$)
NewSpan2 $\leftarrow$ RHSspan(Span, $W_\bullet$)
$R_\bullet \leftarrow$ Pushout$(V_\bullet, W_\bullet, C_\bullet,$ NewSpan1$)$
$T_\bullet \leftarrow$ Pushout$(V_\bullet, R_\bullet, D_\bullet,$ NewSpan2$)$
**return** $T_\bullet$
```

Let us explain this algorithm in more detail. `RestrictedMatrix` simply takes a vector u in C_1 and the differential $\partial_1 : C_1 \to C_0$ and calculates $R_1 = \partial_1 \restriction_{\mathrm{supp}\ u}$, by removing columns with no support in u, and then removing any all-zero rows.

`FindMonicSpan` is more interesting. There exists a basis-preserving monic span

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "f_\bullet"] \arrow[d, hook, "g_\bullet"'] & C_\bullet \\ D_\bullet & \end{tikzcd}$$

if (but not only if) there are permutation matrices M, N such that $R_1 = MR_2N$. That is, we have two injections $U_\bullet \hookrightarrow C_\bullet$ and $V_\bullet \hookrightarrow D_\bullet$, and we wish to find a basis-preserving isomorphism $U_\bullet \cong V_\bullet$ such that we have an injection $V_\bullet \cong U_\bullet \hookrightarrow C_\bullet$. The basis-preserving isomorphism is given precisely by the permutation matrices M and N, which dictate where basis elements of V_1 and V_0 are mapped to. The isomorphism explicitly is

$$\begin{tikzcd} V_1 \arrow[r] \arrow[d, "\sim"] & V_0 \arrow[d, "\sim"] \\ U_1 \arrow[r] & U_0 \end{tikzcd}$$

Finding such permutation matrices is the hypergraph isomorphism problem. This in turn can be expressed as a graph isomorphism problem between bipartite graphs [ADK15], at the cost of some increased space. The graph isomorphism problem is neither known to be poly-time nor NP-complete, but in practice is very fast to solve using VF2 [CFSV04]. `SSIP` uses NetworkX [HSS08] to represent the graphs and call VF2.

Remark 3.3.1. We do not need a hypergraph isomorphism, only a hypergraph inclusion, to construct a basis-preserving monic span. However, isomorphism is necessary for $V_\bullet$ to be a logical operator subcomplex, see Definition 3.2.3, in both codes. We rely on this property to perform logical parity measurements so throughout we assume our monic spans are constructed by hypergraph isomorphisms, and hence the two logicals to be merged are identical up to relabelling of qubits and checks.

There may not be a graph isomorphism, in which case we do not find a basis-preserving monic span. In this case Algorithm 1 returns `None`. On the other hand, there may be many graph isomorphisms. In this case Algorithm 1 just uses the first one found for simplicity. At times we may know the monic span *a priori*, in which case this step can be skipped.

Once the monic span has been found the algorithm then performs the two pushouts. Taking a pushout of a basis-preserving monic span with a logical operator subcomplex at the apex is straightforward. Take the first pushout:

$$\begin{tikzcd}
V_\bullet \arrow[r, hook] \arrow[d, hook] & C_\bullet \arrow[d] \\
W_\bullet \arrow[r] & R_\bullet \arrow[ul, phantom, "\ulcorner", very near start]
\end{tikzcd}$$

We can expand this into components:

$$\begin{tikzcd}
V_0 \arrow[rrrrr, "f_0"] \arrow[ddddd, "l_0"'] & & & & & C_0 \arrow[ddddd] \\
& V_1 \arrow[ul, "\partial_1^V"] \arrow[rrr, "f_1"] \arrow[ddd, "l_1"] & & & C_1 \arrow[ur, "\partial_1^C"] \arrow[ddd] & \\
& & 0 \arrow[ul, "0"] \arrow[r, "0"] \arrow[d, "0"'] & C_2 \arrow[ur, "\partial_2^C"] \arrow[d] & & \\
& & W_2 \arrow[r] \arrow[dl, "\partial_2^W"'] & R_2 \arrow[dr, dotted, "\partial_2^R"] \arrow[ul, phantom, "\ulcorner", very near start] & & \\
& W_1 \arrow[rrr] \arrow[dl, "\partial_1^W"'] & & & R_1 \arrow[dr, dotted, "\partial_1^R"] & \\
W_0 \arrow[rrrrr] & & & & & R_0
\end{tikzcd}$$

The pushout at degree 2 is just $R_2 = W_2 \oplus C_2$. At degree 1 we have $R_1 = W_1 \oplus C_1/\mathrm{im}(l_1) \sim \mathrm{im}(f_1)$. To construct ∂_2^R we therefore start with $\partial_2^W \oplus \partial_2^C$ and add the rows corresponding to quotiented basis elements in R_1 together. That is, if e_i is an entry in V_1, take the entries $l_1(e_i)$ and $f_1(e_i)$ and add those rows together in $\partial_2^W \oplus \partial_2^C$. All rows to be added together have disjoint support, so the addition of rows is unambiguous.

At degree 0 we have $R_0 = W_0 \oplus C_0/\mathrm{im}(l_0) \sim \mathrm{im}(f_0)$. To construct ∂_1^R start with $\partial_1^W \oplus \partial_1^C$ then add rows corresponding to quotiented basis elements in R_0 together. Then, take the bitwise OR (logical inclusive) of columns corresponding to quotiented basis elements in R_1 together.

We verify in Appendix 9 that these differentials are the mediating maps given by the universal properties of pushouts, and that the above diagram commutes. The second pushout to acquire $T_\bullet$ follows in the same fashion.

Optionally, Algorithm 1 can calculate some additional data to inform the user what the effect of the merge has been. This is wrapped up into a `MergeResult` object. In addition to the output `CSScode`, this object contains:

- The inclusion matrix $C_1 \oplus D_1 \hookrightarrow T_1$, i.e. the map on qubits from the initial codes to the merged code.
- The row indices for any new Z stabilisers initialised in P_Z.
- The row indices for any new X stabilisers initialised in P_X.
- The indices of any new qubits initialised.
- A basis for any new $\overline{Z}$ logical operators introduced.
- A basis for any new $\overline{X}$ logical operators introduced.
- A basis for the $k_C + k_D - 1$ $\overline{Z}$ logical operators inherited from $C_\bullet$ and $D_\bullet$, which we call 'old' $\overline{Z}$ logical operators, as opposed to the 'new' $\overline{Z}$ logical operators which can be incidentally introduced during a merge.
- A basis for the 'old' $\overline{X}$ logical operators.

We know the inclusion matrix immediately from the pushouts; the same is true for the indices of new stabilisers and qubits. Calculating the new and old logical operators is done by calculating logicals in the initial codes and multiplying through by the inclusion matrix, then taking the appropriate quotient to find the new logicals.

Overall, aside from the graph isomorphism problem all of the subroutines in this section have at worst $\mathcal{O}(n^3)$ runtime, with the worst complexity being Gaussian elimination, which is required for the additional data. We find in practice that graph isomorphism is not a bottleneck using VF2. For codes with hundreds of qubits Algorithm 1 runs in a few seconds or at most minutes on a Mac laptop. Given that most of the time is spent doing linear algebra in Python, should the runtime become problematic then implementation in a faster language such as C should let Algorithm 1 run in seconds for codes with many thousands of qubits, until the graph isomorphism problem becomes challenging.

In practice, there is some hidden complexity here. Given two arbitrary CSS codes with no additional knowledge of the code structure, the problem to solve is not just whether, given two irreducible logical operators, we can perform an external merge. We would have to work out which irreducible logical operators are available, and so could be paired up to merge. Assuming we have no additional knowledge this will be a formidable problem in general: the number of logical operators will typically scale exponentially with the blocklength of the codes, and the number of possible pairings of logicals between the codes will explode combinatorially. Even if running Algorithm 1 is extremely fast, the combinatorial explosion makes exhaustively finding all monic spans between two large codes implausible.

Thus for codes of high blocklength we would like to know in advance the structure of the available irreducible logical operators. Fortunately, modern qLDPC codes are not random, and in fact tend to be highly structured, such as lifted product codes. In [BCGMRY24, ES24] this structure is leveraged to find irreducible $\overline{Z}$ and $\overline{X}$ logicals for every qubit.

3.3.1 Small examples

We warm up to our results on external surgery with some small $d = 3$ codes with $k = 1$ each. We will use combinations of the

- $[\![9,1,3]\!]$ Shor code [Sho95],
- $[\![15,1,3]\!]$ Quantum Reed-Muller (QRM) code [KLZ96],
- $[\![7,1,3]\!]$ Steane code [Ste96],
- $[\![9,1,3]\!]$ rotated surface code [BM-D07B], and
- $[\![13,1,3]\!]$ unrotated surface code [Kit03],

which we call our small example set.

In addition to all having distance 3, these codes have the following property: there exists a weight 3 $\overline{Z}$ logical operator v with the restricted matrix

$$\partial_1^C \restriction_{\text{supp } v} \sim \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

where $\sim$ means up to permutation of rows and columns. The restricted matrix is the parity-check matrix of a repetition code; this will always be true for an irreducible logical operator, as the only non-zero element in $\ker(\partial_1^{C_\bullet} \restriction_{\text{supp } v})$ must be the all-1s vector, and every minimum-weight logical operator must be irreducible.

The above restricted matrix has $d-1$ rows (once all-zero rows have been removed). We will always have a monic span with $V_\bullet$ having the differential above. Thus we can always do external surgery between any two of these codes, and the merged codes with $r=1$ will have 2 additional data qubits when compared to the disjoint initial codes, i.e. $\dim T_1 = \dim C_1 + \dim D_1 + 2$. The same applies for $\delta_C^1 \restriction_{\text{supp } v}$ for $\overline{X}$ logicals instead, with the exception of the QRM code which has $d_X = 7$ so has no weight 3 $\overline{X}$ logicals.

Of course, not every restricted matrix of a weight d logical has $d-1$ rows for any other $d=3$ CSS code, as there may be redundant checks on that logical. An example for which Algorithm 1 would fail to find any monic spans for $\overline{Z}$ logicals with the codes in our small example set, despite having distance 3, is the $[\![18,2,3]\!]$ toric code. The restricted matrix for a weight 3 $\overline{Z}$ logical in the toric code is

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

up to permutation, i.e. there is an extra X-check; the dual applies for a weight 3 X logical, which will have an extra Z-check.

As it turns out, we can do surgery between any two of the codes in our example set in the Z basis with depth $r=1$ while maintaining $d=3$ in the merged codes. All of the merged codes have 1 logical qubit. For the X basis the same applies with the exception of the QRM code.

The only remaining figure of merit is ω, the maximum weight of any column or row. We show in Figure 3.1 that we increase ω by at most 1 when compared to the codes beforehand. We do not claim that this is optimal – we can obviously do X merges between distance 3 surface codes without increasing ω, but this will depend on the choice of logical used.

Interestingly, the QRM code is triorthogonal [BH12], meaning that it admits a transversal logical T gate. This means that one can use `SSIP` to generate merges between a triorthogonal code and some other code to inject T states. One candidate for the other code is the surface code, with which one can easily perform Cliffords [BKLW17]. The QRM

$\omega_{\text{after}} - \omega_{\text{before}}$	Shor	QRM	Steane	Rotated surface	Surface
Shor	1, 0	1	1, 0	1, 0	1, 0
QRM		1	1	1	1
Steane			1, 1	1, 1	1, 1
Rotated surface				1, 1	1, 1
Surface					0, 1

Figure 3.1: $\omega_{\text{after}} - \omega_{\text{before}}$ for merges between weight 3 logicals. Values for Z merges are shown first and values for X merges second, when weight 3 logicals exist.

code is a small example of a 3D colour code [Bom15], and the merging protocol scales to arbitrary distance d 3D colour codes and surface codes, using only $d-1$ additional qubits to perform the injection. 3D colour codes and surface codes are perhaps not the best candidates for fault-tolerant computation for reasons of threshold and code parameter scaling compared to other qLDPC codes, but the principle is interesting, and different from Bombin's code-switching protocol [Bom15].

It is unsurprising that one can do surgery with our small example set. They can all be seen as topological codes. Other than those already mentioned, the Shor code is a tessellation of $\mathbb{R}P^2$ [FM01] and the Steane code is a 2D colour code [BM-D06]. In fact, surgery between 2D colour codes and surface codes has already been described in [NFB17].

We could also perform logical single-qubit measurements with our small example set using the method in Section 3.2.3, but this is uninteresting as every code in our set has $k = 1$ so measuring the logical qubit in the Z or X basis can be done by measuring out every data qubit in the Z or X basis. Similarly, it does not make sense to do internal surgery with $k = 1$ codeblocks.

3.3.2 Lift-connected surface codes

Lift-connected surface (LCS) codes [ORM24] are lifted product codes where the commutative matrix subring is $\mathscr{C}_\ell$, the ring of ℓ-by-ℓ circulant matrices. Intuitively, one can think of LCS codes as disjoint surface codes which are then connected by some stabilisers. They are interesting in part because their parameters can outperform those of surface codes even at low blocklengths. They also perform comparably to surface codes against phenomenological noise, and can be implemented with 3D local connectivity.

LCS codes are constructed using two variables: ℓ, the size of the circulant matrices, and L, the length of the 'base' code over $\mathscr{C}_\ell$.[2]

Let $P^{(0)} = \mathrm{id}_\ell$ and $P^{(1)}$ be the first right cyclic shift of id_ℓ, so for example

$$P^{(1)} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

when $\ell = 3$.

[2]In this Chapter we have swapped round the variable labelling compared to the original LCS paper [ORM24] in order to conform to the notation in [PK21].

Then, construct the L-by-$L+1$ matrix

$$B = \begin{pmatrix} P^{(0)} & P^{(0)} + P^{(1)} & 0 & \cdots & 0 & 0 \\ 0 & P^{(0)} & P^{(0)} + P^{(1)} & \cdots & 0 & 0 \\ \vdots & & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & P^{(0)} & P^{(0)} + P^{(1)} \end{pmatrix}$$

and let $A = B^\intercal$. Take the tensor product over $\mathscr{C}_\ell$, recalling from Section 2.3.1 that this gives

$$P_Z = \begin{pmatrix} B \underset{\mathscr{C}_\ell}{\otimes} \mathrm{id}_{L+1} & \mathrm{id}_L \underset{\mathscr{C}_\ell}{\otimes} A \end{pmatrix}; \quad P_X = \begin{pmatrix} \mathrm{id}_{L+1} \underset{\mathscr{C}_\ell}{\otimes} B & A \underset{\mathscr{C}_\ell}{\otimes} \mathrm{id}_L \end{pmatrix}$$

which we then view over $\mathbb{F}_2$.

LCS codes have parameters $[\![((L+1)^2 + L^2)\ell, \ell, \min(\ell, 2L+1)]\!]$. Strictly speaking, the parameter $d = \min(\ell, 2L+1)$ has not been proven, just conjectured with empirical evidence to support it. Under this conjecture, LCS codes have distance scaling linearly in n until $\ell = 2L+1$. This is shown to be true for LCS codes under a certain size [ORM24, Sec. III A]. LCS codes are also ω-limited with $\omega = 6$, so they are qLDPC codes.

Individual merges

We test individual merges between LCS codes, without considering parallelisation. We take the set of LCS codes with $L \in \{1, 2, 3\}$ and $\ell \in \{L+2, L+3, L+4\}$, except we truncate at $\ell = 6$. Our smallest initial code has $L = 1$, $\ell = 3$ and parameters $[\![15, 3, 3]\!]$. Our largest initial code has $L = 3$, $\ell = 6$ and parameters $[\![150, 6, 6]\!]$. These blocklengths are chosen such that our results are reproducible in a few hours on a personal computer, and so that the initial codes are close to having the best possible parameters for LCS codes, see [ORM24, Fig. 4].

Our method is simple. For each code $C_\bullet$, we find an arbitrary tensor product decomposition of the logical space, i.e. a basis of the homology space $H_1(C_\bullet)$, and its consistent basis for the cohomology space $H^1(C^\bullet)$. We then pick out representative logicals for each one, which we call u_i for $u_i \in [u_i]$, recalling that the basis set is $\{[u_i]\}_{u \in I}$, such that the logicals are irreducible. In principle, should we not find an irreducible logical for a given qubit we would leave that qubit out, but we successfully find irreducible logicals for all qubits in our benchmarking set.

We then test by taking two identical copies of $C_\bullet$ and merging them along their shared irreducible logical u_i. Evidently this is guaranteed to give a monic span as the two codes are identical. After the merge, there may be additional logical qubits introduced. We relegate these to being gauge qubits and make the merged code a subsystem CSS code.

For every merge, we compute three figures of merit: (1) the depth r required for the merges to leave the code distance unchanged, so $d = \min(\ell, 2L+1)$ when viewed as a subsystem code, (2) the total number of additional data qubits required as a proportion of the total length of the original codes $n_{\text{ancilla}}/n_{\text{initial}}$, (3) the maximum weight of any row or column in the parity-check matrices ω. In Figure 3.2 we present the mean average of these values over each of the $\overline{X}$ and $\overline{Z}$ logicals for a given pair of codes. For example, the $[\![15, 3, 3]\!]$ code has 3 logical qubits, so we perform 3 different X-merges and average out the values of r, $n_{\text{ancilla}}/n_{\text{initial}}$ and ω for the 3 different merged codes. For more fine-grained results, where we show the results of each merge rather than just their averages, see Appendix 11. The scripts for running all benchmarks can be found

L	ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
1	3	1	0.16	6
1	4	1	0.14	6
1	5	1	0.12	6
2	4	1	0.1	7
2	5	2	0.36	7
2	6	2	0.3	7
3	5	2.6	0.37	7
3	6	2.8	0.31	7

L	ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
1	3	1	0.15	6
1	4	1	0.125	6
1	5	1	0.112	6
2	4	1.75	0.25	7
2	5	2.6	0.34	7
2	6	3	0.37	7
3	5	2.4	0.27	7
3	6	3	0.31	7

Figure 3.2: Figures of merit for individual X and Z merges between LCS codes.

at `https://github.com/CQCL/SSIP/benchmarks`. We explicitly calculated all subsystem code distances using Z3, as the distance is low enough for this method to be practical.

Reading through the tables of Figure 3.2, we can see first that the average required depth r increases as the size of the initial codes increases. This is not surprising, as the larger the initial code the more likely it is that performing a low depth merge with another code will incidentally add a logical, which does not belong wholly to the new logical qubits introduced, with a lower distance. Regardless, the depth required remains low, with the maximum required for any of the merges being 4. Similarly, as the size of the initial codes increases so does the typical proportion of new qubits required for the merges. The intermediate code being added is a tensor product code, which itself has vanishing rate and poorer distance scaling when compared to the underlying quantum memories. Thus when we have to increase the depth r we are adding more of a 'worse' code, slightly inhibiting the efficiency of the merges.

There are cases at larger sizes where the merges can be performed at $r = 1$, and in these cases $n_{\text{ancilla}}/n_{\text{initial}}$ is extremely low. For instance, for $L = 3, \ell = 5$, which makes $n_C = 125$, so $n_{\text{initial}} = 250$, there is a $\overline{Z}$ logical which gives an $r = 1$ merge at $n_{\text{ancilla}}/n_{\text{initial}} = 0.072$, a marginal overhead of 18 ancillary data qubits compared to the overall blocksize of 250. We expect that a greater understanding of the structure of LCS codes would yield more logicals which admit low depth merges, but our rudimentary technique only finds these occasionally.

Lastly, ω remains virtually constant throughout, at just above $\omega = 6$ for the initial LCS codes. In [CB24, Lem. 5.18] we showed that merges of qLDPC codes remain qLDPC, and for LCS codes the row and column weights barely increase.

In Figure 3.3 we also compare to both lattice surgery and a naive application of

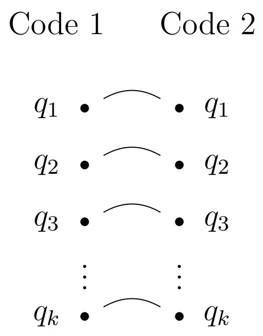
[CKBB22] to performing the same merges in the Z basis. That is, for lattice surgery we generate $2k$ (unrotated) surface code patches with the same d as the LCS codes, then perform a single merge between two patches and record the total overhead. For [CKBB22] we use LCS codes, but then for merging we initialise large $r = d$ tensor product ancillae codes which are connected appropriately.

As mentioned in Section 3.1.1, comparisons between these different procedures will generally be apples-to-oranges. The results here are for generating codeblocks and then performing a single merge between them. The advantage of surface codes is that one can easily parallelise lattice surgery, while the same is not true of our protocol. For example, constructing 12 surface code patches, as in the $L = 3$, $\ell = 6$ comparison case, and then performing a single merge between two of them is quite a contrived scenario. Similarly, for the approach of [CKBB22] one can do Pauli measurements using any combination of logical qubits, while ours is more restricted. We also used the default procedure, where the ancilla blocks of [CKBB22] have high depth, where in reality one could perhaps get away with reducing the overhead while maintaining distance.

With those caveats out of the way, the main take away of Figure 3.3 is that for individual merges at low blocklength, our homological approach requires much less overhead than the other two methods. A common feature of both our procedure and that of [CKBB22] is that, while the initial quantum memories have better parameters than those of surface codes, some of the overhead is instead offloaded onto the ancillae used for surgery. Surface codes require very small numbers of additional qubits.

Parallel merges

Now we present results for performing a logical merge between *every* pair of logical qubits in the two codes simultaneously, as shown in the schematic below:



We use the same logicals as before. This time, our approach is as follows: take $r = 1$ for a merge between logical qubits of the same index. Should this result in a merged code with lower d than the initial codes, when viewed as a subsystem CSS code, then increment r to 2 for *every* merge, and so on until $d = \min(\ell, 2L + 1)$. This is to avoid having to explore the search space of different possible depths for each merge.

We follow the same procedure of using Z3 to calculate subsystem code distances. Results are presented in Figure 3.4.

Here we see the cost of parallelisation. Not only does the overhead in terms of ancillae data qubits increase, so too does ω. This is more-or-less unavoidable with efficient codes: the logicals being used to perform merges are likely to have overlap on some data qubits and stabilisers, so the new tensor product codes will increase stabiliser weights and the number of stabilisers some data qubits are in the support of. Despite this, the depths are encouraging. The largest LCS codes used have 6 logical qubits, but despite this only

L	ℓ	n_{initial}	$\langle n_{\text{ancilla}}\rangle$	$\langle n_{\text{total}}\rangle$
1	3	30	4.5	34.5
		78	2	80
		30	49	79
1	4	40	5.6	45.6
		104	2	106
		40	55	95
1	5	50	5.6	55.6
		130	2	132
		50	60.6	110.6
2	4	104	26	130
		200	3	203
		104	142	246
2	5	130	44.2	174.2
		410	4	414
		130	199	329
2	6	156	57.7	213.7
		492	4	496
		156	209	365
3	5	250	67.5	317.5
		410	4	414
		250	316.6	566.6
3	6	300	93	393
		732	5	737
		300	404.3	704.3

Figure 3.3: Comparison of LCS code individual Z-merges to surface codes and [CKBB22]. The first row in each box is our homological approach using Algorithm 1. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to LCS codes.

L	ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	1	0.47	8
1	4	1	0.55	9
1	5	1	0.62	10
2	4	2	1.27	10
2	5	3	2.51	11
2	6	3	2.6	12
3	5	2	1.34	11
3	6	2	1.37	12

L	ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	1	0.43	8
1	4	1	0.5	9
1	5	1	0.56	10
2	4	2	1.15	9
2	5	3	2.08	10
2	6	2	1.33	11
3	5	1	0.34	11
3	6	2	1.1	11

Figure 3.4: Figures of merit for parallel X and Z merges between LCS codes.

a merge depth of $r \leq 2$ was required for their parallel merges. We again compare the overhead required to that of surface codes with the same k and d as the LCS codes, and a naive application of Cohen et al. [CKBB22] in Figure 3.5.

While still comparing favourably in terms of overall qubit overhead to surface codes, the advantage is significantly weakened. This is because surface codes make it extremely easy to parallelise merges. We expect substantial gains could be made by considering the logicals used more carefully, and lowering the level of parallelisation somewhat without restricting ourselves to individual merges.

Naive application of [CKBB22] performs very poorly by contrast. This is because the quantum memory is not yet large enough for its efficiency as a qLDPC code to outweigh the large ancilla requirements when compared to surface codes, and because we can 'get away with' having low depth merges in our homological approach.

Individual single-qubit measurements

We now retrace our steps for the same benchmarking set but performing single-qubit logical measurements instead, following Section 3.2.3. Recall that we are claiming no novelty in our approach here, it is merely that of [CKBB22] translated into chain complexes. It is still interesting, however, because in [CKBB22, Table. 1] the results given are just estimates at high depth. We show that it is possible to perform these single-qubit measurements while incurring lower overhead.

We show figures of merit for single-qubit measurements in Figure 3.6. Overall, on the LCS benchmarking set they tend to be more expensive than individual external merges, both as a fraction of the initial blocklength and the raw number of ancilla qubits. Similarly, in Figure 3.7 we see that while low depth measurements in this manner still compare

L	ℓ	n_{initial}	n_{ancilla}	n_{total}
1	3	30	13	43
		78	6	84
		30	147	177
1	4	40	20	60
		104	6	110
		40	220	260
1	5	50	28	78
		130	6	136
		50	303	353
2	4	104	120	224
		200	12	212
		104	568	672
2	5	130	271	401
		410	20	430
		130	995	1125
2	6	156	207	363
		492	24	416
		156	1254	1410
3	5	250	91	341
		410	20	430
		250	1583	1833
3	6	300	331	631
		732	30	762
		300	2426	2726

Figure 3.5: Comparison of LCS code parallel Z-merges to surface codes and [CKBB22]. The first row in each box is our homological approach using Algorithm 1. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to LCS codes.

L	ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
1	3	1.67	0.8	6
1	4	1.75	0.75	6
1	5	1.8	0.7	6
2	4	1.5	0.5	7
2	5	2.6	1.03	7
2	6	2.5	0.82	7
3	5	3.8	1.12	7
3	6	3.3	0.81	7

L	ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
1	3	1.67	0.7	6
1	4	1.75	0.75	6
1	5	1.8	0.64	6
2	4	1.2	0.3	7
2	5	3.4	0.96	7
2	6	4	1.02	7
3	5	2.8	0.7	7
3	6	2.83	0.62	7

Figure 3.6: Figures of merit for individual single-qubit logical X and Z measurements with LCS codes.

favourably to those of surface codes, the difference is much less pronounced, and again one should bear in mind that surface codes favour parallelisation much better.

Parallel single-qubit measurements

We could consider performing single-qubit measurements in the same basis on *every* logical qubit in parallel, but this would be a completely contrived benchmark, as this can always be done in a CSS code by measuring out the existing data qubits, rather than adding new data qubits. Instead, we consider the following: take the first half (rounded down) of the logical qubits and perform logical single-qubit measurements on these in parallel. The half of the logical qubits is chosen arbitrarily. We show figures of merit for doing this in Figure 3.8, then show comparisons to surface codes and a naive application of [CKBB22] in Figure 3.9.

In Appendix 10 we rerun this entire benchmarking procedure for generalised bicycle (GB) codes [PK21, KP13]. We find a similar story there, but GB codes are even more amenable to surgery and compare extremely favourably compared to surface codes and the approach of [CKBB22].

3.3.3 The gross code

Bivariate bicycle (BB) codes [BCGMRY24] are lifted products over the ring $\mathscr{C}_\ell \otimes \mathscr{C}_m$, that is the tensor product over rings of circulant matrices of different dimensions, but viewed over $\mathbb{F}_2$.

L	ℓ	n_{initial}	$\langle n_{\text{ancilla}} \rangle$	$\langle n_{\text{total}} \rangle$
1	3	15	10.5	25.5
		39	0	39
		15	23	38
1	4	20	15	35
		52	0	52
		20	26	46
1	5	25	16	41
		65	0	65
		25	28.8	53.8
2	4	52	15.6	77.6
		100	0	100
		52	69	121
2	5	65	62.4	127.4
		205	0	205
		65	97	162
2	6	78	79.6	157.7
		246	0	246
		78	102	180
3	5	125	87.5	212.5
		205	0	205
		125	155.8	280.8
3	6	150	93	243
		366	0	366
		150	199	349

Figure 3.7: Comparison of LCS code individual single-qubit logical Z-measurements to surface codes and a naive application of [CKBB22]. The first row uses the method described in Section 3.2.3. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to LCS codes.

L	ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	2	1.0	6
1	4	2	1.65	7
1	5	2	1.56	7
2	4	1	0.48	8
2	5	2	0.96	8
2	6	3	2.24	9
3	5	4	2.36	8
3	6	3	2.19	9

L	ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	2	1.07	6
1	4	2	1.85	7
1	5	2	1.6	7
2	4	1	0.42	8
2	5	2	0.98	8
2	6	3	2.11	9
3	5	3	0.97	8
3	6	2	0.9	9

Figure 3.8: Figures of merit for parallel single-qubit logical X and Z measurements with LCS codes.

Recalling that $\mathscr{C}_\ell \cong \mathbb{F}_2^{\langle \ell \rangle}$, it is immediate that

$$\mathscr{C}_\ell \otimes \mathscr{C}_m \cong \mathbb{F}_2^{\langle \ell \rangle} \otimes \mathbb{F}_2^{\langle m \rangle},$$

the ring of polynomials over two variables x and y modulo $x^\ell - 1$ and $y^m - 1$. The bijection sends

$$P_\ell^{(1)} \otimes \mathrm{id}_m \mapsto x; \quad \mathrm{id}_\ell \otimes P_m^{(1)} \mapsto y,$$

and $x^\ell = y^m = 1$. Recall that $P_\ell^{(1)}$ is the right cyclic shift matrix of id_ℓ, as in Section 3.3.2.

Then, let

$$A = A_1 + A_2 + A_3; \quad B = B_1 + B_2 + B_3$$

where each matrix A_i and B_j is a power of x or y, interpreted in $\mathscr{C}_\ell \otimes \mathscr{C}_m$. Bivariate bicycle (BB) codes have $P_X = \begin{pmatrix} A & B \end{pmatrix}$ and $P_Z = \begin{pmatrix} B^\intercal & A^\intercal \end{pmatrix}$ for some matrices A and B. Therefore each BB code is uniquely defined by a pair of polynomials in $\mathbb{F}_2^{\langle \ell \rangle} \otimes \mathbb{F}_2^{\langle m \rangle}$, each of which is the sum of three monomials in a single variable. Observe that $n = 2\ell m$ for any BB code. Call the first and second block of ℓm data qubits the 'unprimed' and 'primed' blocks respectively.

The example we focus on in this work is the "gross code", a $[\![144, 12, 12]\!]$ code with $\ell = 12$, $m = 6$, $A = x^3 + y + y^2$ and $B = y^3 + x + x^2$. This code has a high error threshold under circuit-level noise, its Tanner graph can be decomposed into two planar subgraphs, which is important for planar architectures, and it is ω-limited with $\omega = 6$.

As we are only using one code here, the benchmarking we perform will be a bit more exhaustive. For this, we use another useful feature of BB codes: we can calculate a basis of the logical space using the algebraic structure of the codes. In the case of the gross

L	ℓ	n_{initial}	n_{ancilla}	n_{total}
1	3	15	16	31
		39	0	39
		15	27	42
1	4	20	37	57
		52	0	52
		20	62	82
1	5	25	40	65
		65	0	65
		25	67	92
2	4	52	21.8	73.8
		100	0	100
		52	145	197
2	5	65	64	129
		205	0	205
		65	190	255
2	6	78	164.6	242.6
		246	0	246
		78	294	372
3	5	125	121.3	246.3
		205	0	205
		125	217	342
3	6	150	135	285
		366	0	366
		150	491	641

Figure 3.9: Comparison of LCS code parallel single-qubit logical Z-measurements to surface codes and a naive application of [CKBB22]. The first row uses the method described in Section 3.2.3. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to LCS codes.

code this gives us an immediate set of logical Paulis with weight 12, one $\overline{Z}$ and one $\overline{X}$ for each logical qubit. As $d = 12$, these logicals are also irreducible. We forgo further details but see [BCGMRY24, Sec. 9.1]. We use these logicals for all our benchmarking in this section. These logicals split into primed and unprimed sets, with the (un)primed set having support only in the (un)primed block.

First we find that there is a basis-preserving monic span between every pair of $\overline{Z}$ logicals in the primed block; the same applies to every pair of logicals in the unprimed block, and also to $\overline{X}$ logicals. Therefore, given two copies of the gross code we can perform individual external merges between any of the logical qubits which belong to the same block, in either basis. As $d = 12$ for the gross code, checking preservation of distance in merged codes is out of reach of the Z3 algorithm in reasonable compute time, so we again rely on `QDistRnd`. The upshot is that these individual merges can each be done with a depth $r = 1$, requiring only 18 additional data qubits and increasing ω to 7, leaving the code distance as a subsystem code at 12 assuming the bound from `QDistRnd` is tight.

Furthermore, we find that we can do parallel merges on all 12 logical qubits between two copies of the gross code in either basis using only $r = 1$, requiring a total of $18 \times 12 = 216$ extra data qubits. This raises ω to 12.

Similarly we can study individual single-qubit logical measurements. We find that single-qubit logical X measurements on the unprimed block require a depth of $r = 3$, and so 78 extra data qubits. They also introduce 72 new stabiliser generators, so a total of 150 new qubits including syndrome qubits. X measurements on the primed block require only a depth of $r = 1$, 18 extra data qubits, and 12 new generators so a total of 30 new qubits. These all raise ω to 7. These overheads are far below that required by performing such measurements naively, as it was predicted in [BCGMRY24, Sec. 9.4] that these measurements would each require a total of 1380 extra qubits when including syndrome qubits (although the authors did expect this value to be optimised significantly). The flipped version applies to single-qubit logical Z measurements: those in the unprimed block require a depth of $r = 1$, and a total of 30 new qubits. Those in the primed block require $r = 3$ and 150 total new qubits.

Additionally, we can perform parallel single-qubit logical measurements. We can measure every logical qubit in the unprimed block in the X basis with $r = 3$ using 468 new data qubits and 432 new syndrome qubits, so 900 ancillae in total. This increases ω to 8. The same applies to the primed block in the Z basis.

We can measure every logical qubit in the primed block in the X basis with $r = 1$ using 108 new data qubits and 72 new syndrome qubits, so 190 ancillae in total. This increases ω to 11. The same applies to the unprimed block in the Z basis.

3.4 Automated internal surgery

As mentioned earlier we can also use `SSIP` to perform internal surgery, that is surgery between logicals in the same codeblock. This is performed in a similar manner to external surgery but with some minor differences. The basic data given to Algorithm 2 is as follows:

- The parity-check matrices of $C_\bullet$, the code within which internal surgery will be performed.
- The two irreducible logicals $u, v \in C_1$ we would like to merge.
- The basis (Z or X) to perform the merge in.

- The desired depth r of the merge.

Algorithm 2 Internal merge calculation

```
$RM_1 \leftarrow \text{RestrictedMatrix}(u, \partial_1^C)$
$RM_2 \leftarrow \text{RestrictedMatrix}(v, \partial_1^C)$
if $|RM_1 \cap RM_2| \neq 0$ then
    return None
Diagram $\leftarrow$ FindDiagram($RM_1, RM_2$)
if Diagram is None then
    return None
$V_\bullet \leftarrow RM_1$
$P_\bullet \leftarrow \text{ConstructP}(r)$
$W_\bullet \leftarrow (P \otimes V)_\bullet$
NewDiagram1 $\leftarrow$ LHSdiagram(Diagram, $W_\bullet$)
$R_\bullet \leftarrow \text{Coequaliser}(V_\bullet, (W \oplus C)_\bullet, \text{NewDiagram1})$
NewDiagram2 $\leftarrow$ RHSdiagram(Diagram, $W_\bullet$)
$T_\bullet \leftarrow \text{Coequaliser}(V_\bullet, R_\bullet, \text{NewDiagram2})$
return $T_\bullet$
```

The first thing Algorithm 2 does is calculate the restrictions of P_X to the support of the logicals u and v. It then finds $|RM_1 \cap RM_2|$, by which we mean the set of data qubits which have overlapping support, and the same for stabiliser generators. If there is any overlap on either of these, Algorithm 2 rejects the merge and outputs `None`. The algorithm then proceeds similarly to Algorithm 1: it computes a hypergraph isomorphism between the restricted matrices, then computes a tensor product code to merge the two logicals together. Finally, it computes the two coequalisers and returns the merged code $T_\bullet$. See Appendix 9 for this computation. Optionally, Algorithm 2 can return a `MergeResult` object, which contains the same additional data as in Algorithm 1.

3.4.1 The gross code

We return to the $[\![144, 12, 12]\!]$ gross code to conduct benchmarking on internal merges. While we are guaranteed to have a diagram of the form

$$\begin{tikzcd} V_\bullet \arrow[r, hook, shift left, "f_\bullet"] \arrow[r, hook, shift right, "g_\bullet"'] & C_\bullet \end{tikzcd}$$

whenever u and v are in the set of irreducible logicals given in [BCGMRY24, Sec. 9.1] and belong to the same block (primed or unprimed), these will commonly have some overlap in data qubits or stabilisers. Therefore we cannot perform internal merges using any arbitrary pair of logical qubits in the same block, even if they have logicals of the same shape. We emphasise that as stated in Remark 3.2.6, one can in this case do a single-qubit logical measurement but in the homology basis where $\overline{Z} \otimes \overline{Z}$ is a single $\overline{Z}$, assuming that one can find a logical $\overline{Z} \otimes \overline{Z}$ operator which is irreducible, and where the component $\overline{Z}$ operators have some support overlap. We do not test out that case here.

Remark 3.4.1. Of course, each logical qubit has many irreducible logicals associated to it, and so we could try to find pairs which do or do not overlap. This is a large search space so we just stick with the irreducible logicals given in [BCGMRY24, Sec. 9.1].

r	0	1	2	3	4	5	6	7	8	9	10	11
0	-	2	2	2	3	-	-	-	-	-	-	-
1	2	-	-	2	3	2	-	-	-	-	-	-
2	2	-	-	3	2	2	-	-	-	-	-	-
3	2	2	3	-	-	2	-	-	-	-	-	-
4	3	3	2	-	-	2	-	-	-	-	-	-
5	-	2	2	2	2	-	-	-	-	-	-	-
6	-	-	-	-	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-	-	-	-	-	-

r	0	1	2	3	4	5	6	7	8	9	10	11
0	-	-	-	-	-	1	-	-	-	-	-	-
1	-	-	-	-	-	1	-	-	-	-	-	-
2	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	-	-	-	1	-	-	-	-	-	-
4	-	-	-	-	-	-	-	-	-	-	-	-
5	1	1	-	1	-	-	-	-	-	-	-	-
6	-	-	-	-	-	-	-	3	-	2	2	2
7	-	-	-	-	-	-	3	-	3	2	2	-
8	-	-	-	-	-	-	-	3	-	2	3	2
9	-	-	-	-	-	-	2	2	2	-	-	2
10	-	-	-	-	-	-	2	2	3	-	-	3
11	-	-	-	-	-	-	2	-	2	2	3	-

Figure 3.10: Depths required for individual internal X and Z merges between logical qubits i and j in the gross code. Dashed entries have no internal merge for the logical operators chosen.

In Figure 3.10 we show results for internal merges in the X and Z basis. All of the possible merges increase ω to 7. At depth $r = 1$ we use 18 ancilla data qubits for a merge. For $r = 2$ we use 48, and for $r = 3$ we use 78.

Of the possible 15 different internal merges one could do within a primed or unprimed block, we find that 12 different X merges can be done in the unprimed block, as the logicals have no overlap, while none can be done in the primed block. For Z merges, only 3 can be done in the unprimed block, while 12 can be done in the primed block.

3.5 Future directions

In order for surgeries identified with `SSIP` to be useful in practice we must tackle the problems which `SSIP` does not handle, as stated in the introduction, namely: establishing (pseudo-)thresholds for codes throughout the surgery process, along with circuits for the syndrome measurements, and decoders which function throughout.

There are recently developed classes of lifted product qLDPC codes which we have not tested `SSIP` on, for which it could be interesting to do so [LP24, SHR24]. Beyond these, it would be very interesting to consider the design of efficient qLDPC CSS codes which, in addition to other useful properties such as low depth syndrome circuits, also admit surgery between and within codeblocks with low overhead of additional qubits, while provably retaining high code distance. This would allow us to avoid the problems of (a) trying to find suitable logicals to construct merges, which results in a combinatorial explosion when done naively, and (b) calculating distance after merges, which is always going to be difficult without additional *a priori* knowledge of the code's structure.

Arguably the most interesting use-case for surgery is merging different classes of codes, such that we can achieve universality using the different logical operations available natively to the codes. To that end it is an interesting question to consider large codes which are triorthogonal or otherwise admit transversal logical non-Clifford gates, and which have low thresholds and favourable code parameters. Given such large codes, performing code merges could allow us to cheaply teleport magic states into quantum memories which admit Clifford operations, or vice versa, thereby circumventing Eastin-Knill [EK09] without requiring magic-state distillation or cultivation [GSJ24]. Previous resource estimates using code-switching protocols with the 3D colour code have indicated that distillation is superior [BKS21], but code-switching is somewhat different to magic state injection by surgery, and those limitations may not apply when using different triorthogonal codes.

3.5.1 Basis-changing ancillae

In a different direction, while the chain complex formalism is perhaps more sophisticated than *ad hoc* constructions with topological codes, the actual tensor product codes we are using to perform merges are quite primitive; they are the obvious generalisation of the small codes used to merge patches in lattice surgery. There is no reason why there should not be more sophisticated ancilla codes which could be initialised to merge logicals, in certain cases going beyond just parity measurements to many-qubit measurements in certain codes, but which do not suffer from the higher overhead of ancillae used for many-qubit measurements in [CKBB22].

To that end, in this subsection we present an extended description of *basis-changing* the ancilla patch for a logical measurement. We thank Zhiyang He for helpful discussions on this topic, and Aleks Kissinger for pointing out the right inverse of the repetition code parity-check matrix which led to Lemma 3.5.5.

The first observation is that when making the pushout for a single-qubit $\overline{Z}$ measurement with an irreducible $\overline{Z}$ logical,

$$\begin{tikzcd} V_\bullet \arrow[r, hook] \arrow[d, hook] & (S \otimes V)_\bullet \arrow[d] \\ C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner", very near end] & R_\bullet \end{tikzcd}$$

we can relax the basis-preservation condition of Definition 2.3.4. One way to do this is to consider the chain complex $U_\bullet = U_1 \to U_0$, where ∂_1^U is the 'ideal' parity-check matrix of the repetition code, and $\dim U_1 = \dim V_1$. So $U_\bullet$ has the same number of data qubits as the logical operator subcomplex $V_\bullet$, but fewer X-checks (or the same number, if $V_\bullet$ has no redundant checks). Explicitly,

$$\partial_1^U = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}.$$

Then, there is an injection $\iota_\bullet : U_\bullet \hookrightarrow V_\bullet$, which explicitly is

$$\begin{tikzcd} U_1 \arrow[r, "\sim"] \arrow[d, "\partial_1^U"] & V_1 \arrow[d, "\partial_1^V"] \\ U_0 \arrow[r, hook] & V_0 \end{tikzcd}$$

where, letting $\dim U_1 = \dim V_1 = n$, $\dim \mathrm{im}(\partial_1^U) = \dim \mathrm{im}(\partial_1^V) = n-1$. The last equality holds as $V_\bullet$ is irreducible. We can choose the injection such that ι_1 is an equality. Now, ι_0 is an isomorphism between U_0 and $\mathrm{im}(\partial_1^V)$, hence $\mathrm{im}(\iota_0^\intercal \restriction \mathrm{im}(\partial_1^V)) = \mathrm{im}(\partial_1^U) = U_0$. But this map is not basis-preserving: it sends single basis elements in U_0 to multiple in V_0. Nevertheless, we can define the following pushout:

$$\begin{tikzcd} U_\bullet \arrow[r, hook, "g"] \arrow[d, hook, "\iota"] & (U \otimes S)_\bullet \arrow[dd] \\ V_\bullet \arrow[d, hook, "f"] & \\ C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner", very near end] & R_\bullet \end{tikzcd}$$

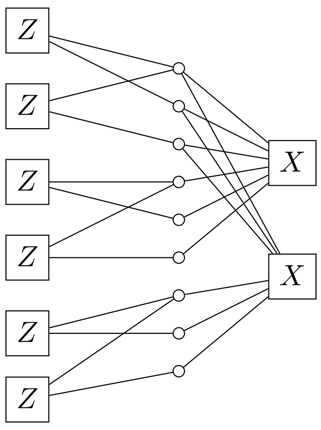
where, as $\dim U_0 \leq \dim V_0$, the tensor product code $(U \otimes S)_\bullet$ is smaller, in both the number of data qubits and the number of stabilisers, than $(S \otimes V)_\bullet$ would be. However, because ι is not basis-preserving, the code $R_\bullet$ is not uniquely defined by Lemma 2.3.5.

We can fix this in a simple manner, but to do so we have to inspect microscopic behaviour of the code, and to do so it is helpful to use *Tanner graphs.*

Definition 3.5.1. The Tanner graph of a chain complex $C_\bullet \in \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ is an undirected graph $G(V, E)$ where V is the set of all basis elements in $C_\bullet$, and there is an edge $e \in E$ between two vertices u, v iff $v \in \partial(u)$ or $u \in \partial(v)$ as basis elements.

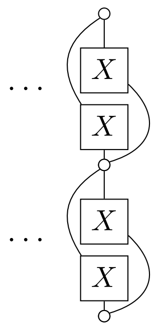
Tanner graphs are used extensively in other approaches to code surgery [CKBB22, CHRY24].

Example 3.5.2. This is the Tanner graph of the $[\![9,1,3]\!]$ Shor code.

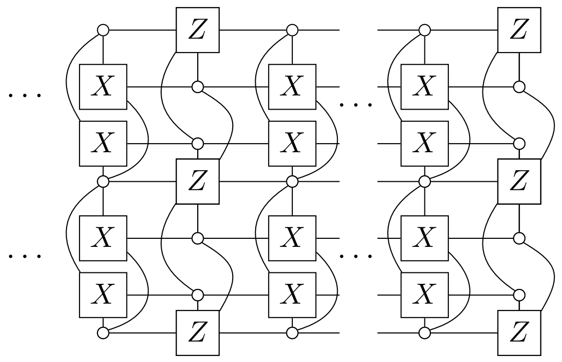


X and Z-checks are labelled, while qubits are left as circles.

Let us use an example of a logical $\overline{Z}$ operator which has redundant X-checks, in some arbitrary CSS code.

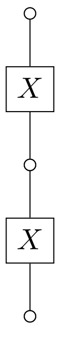


Every vertex in the Tanner graph of this subcomplex $V_\bullet$ has edges extending into the rest of the code $C_\bullet$, indicated by ellipses. Ordinarily, a logical measurement using this operator would look like:

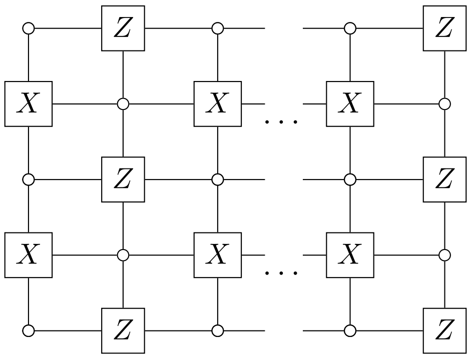


where we have glued in the tensor product code $(S \otimes V)_\bullet$.

If instead we start with $U_\bullet$, which has the Tanner graph



then $(S \otimes U)_\bullet$ has the Tanner graph

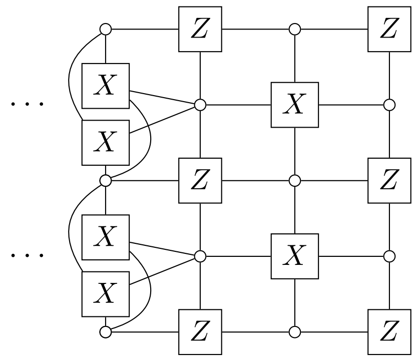


Finding a suitable pushout code $R_\bullet$ can then be done using the injection ι. In this case, we have

$$\partial_1^V = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}; \quad \partial_1^U = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

and so $\iota_0 = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$.

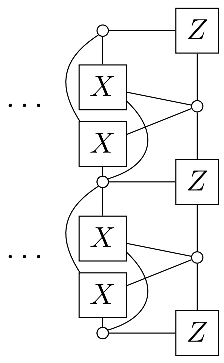
We can then glue $(S \otimes U)_\bullet$ into $C_\bullet$, setting $f_0 \circ \iota_0(e_i) \sim g_0(e_i)$ for each basis element $e_i \in U_0$. The part of $R_\bullet$ we are interested in can now be described by the Tanner graph



where qubits in the left-most slice of the tensor product code $(S \otimes U)_\bullet$ have now been quotiented with qubits as normal, but the X-checks are quotiented with *multiple* X-checks, as shown by the extra edges. The incidence matrix of the subgraph connecting the X-checks to the qubits in the next slice is exactly ι_0. In this way, we have traded away qubits for extra connectivity into the ancilla layer. There is another side-effect: we have not introduced any new logical qubits. If we had glued in $(S \otimes V)_\bullet$ then the redundant checks would have the consequence of introducing new logicals in the intermediate slices; as we have glued in a smaller tensor product code and eliminated the redundancy in intermediate layers we have also eliminated any new logicals. This is a general property of a basis change to the repetition code.

Proposition 3.5.3. A single-qubit logical measurement which changes basis to the repetition code tensor product $(S \otimes U)_\bullet$ has no additional logical qubits.

Proof. We first show it for the first slice, then the proof extends iteratively. For the first slice we have:



There are no new logicals in the old code $C_\bullet$. There is evidently not a new $\overline{X}$ logical wholly contained on the second slice in the Tanner graph above, as there are too many Z-checks. Phrased differently, any new $\overline{X}$ logical wholly in that slice must be in $\ker((\partial_1^U)^\intercal)$, but $\dim\ker((\partial_1^U)^\intercal) = 0$. If there are new $\overline{X}$ logicals, then, they must have support on both the first and second slices.

If a qubit in the first slice has an X Pauli on it, to make it a logical it must also have support on some qubits in the second slice. In particular, let u be the set of X Paulis on the first slice which are applied as part of a logical. Then to satisfy the stabilisers in the second slice it must also have support on qubits defined by $v = \partial_1^U u$, as we have 'switched on' $|u|$ Z stabilisers in slice two, which must be switched off by X Paulis on the qubits in that slice. Now, $v = \iota_0^\intercal \circ \partial_1^V u$. This is because $\iota_0^\intercal \restriction \mathrm{im}(\partial_1^V) = \mathrm{im}(\partial_1^U) = U_0$ by construction.

Applying the set of stabilisers $\partial_1^V u$ will therefore remove the $\overline{X}$ logical from both slices entirely, moving the logical into the old code. As there can be no new logicals wholly contained in the old code, there are no new $\overline{X}$ logicals; hence there are no new logical qubits.

Further slices work similarly, but more easily as there is now a 1-to-1 map between qubits in the second slice and checks in the third slice. Hence, even if there are m slices, there are no new logical qubits. ■

This is a different approach to the gauge-fixing of [CHRY24], which eliminates the new logicals by adding stabilisers.

Corollary 3.5.4. A single-qubit logical measurement which changes basis to the repetition code tensor product $(S \otimes U)_\bullet$ preserves the X-distance of $C_\bullet$.

Proof. First, by using the flexibility of pushouts, we change from the pushout

$$\begin{tikzcd} U_\bullet \arrow[r, hook, "g"] \arrow[d, hook, "f\circ\iota"] & (U \otimes S)_\bullet \arrow[d] \\ C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner"] & R_\bullet \end{tikzcd}$$

to

$$\begin{tikzcd} V_\bullet \arrow[r, hook] \arrow[d, hook] & D_\bullet \arrow[d] \\ C_\bullet \arrow[r] \arrow[ur, phantom, "\ulcorner"] & R_\bullet \end{tikzcd}$$

where $D_\bullet$ has the Tanner graph shown above, i.e. it already includes the basis change, and thus all morphisms are basis-preserving.

Then, we extend Lemma 2.3.17 to the case where one of the logicals is actually a stabiliser. In this case, we have the coequaliser

$$\begin{tikzcd} V_\bullet \arrow[r, shift left, "\iota_C \circ f_\bullet"] \arrow[r, shift right, "\iota_D \circ g_\bullet"'] & (C \oplus D)_\bullet \arrow[r, "\mathrm{coeq}_\bullet"] & R_\bullet \end{tikzcd}$$

The cochain map coeq$^\bullet$ then maps $\overline{X}$ operators from $R^\bullet \to (C \oplus D)^\bullet$, such that all nontrivial operators are mapped to nontrivial operators, as the only element of $H^1(R^\bullet)$ mapped to $[0] \in H^1((C \oplus D)^\bullet)$ is $[0] \in H^1(R^\bullet)$. Then, take any $\overline{X}$ in $R^\bullet$ and map it using $H^1(\mathrm{coeq}^\bullet)$. On $D^\bullet$ this will be mapped to a stabiliser, as it has no logical qubits, but on $C^\bullet$ it must be mapped to a nontrivial logical. The map to $C^\bullet$ is 1-to-1 on those qubits, so the logical in $C^\bullet$ must have weight at least as small as the original $\overline{X}$-logical, so the distance is preserved by the logical measurement.

Note that this does not apply when new logicals are introduced for the same reason as Lemma 2.3.17. ■

We now comment on the LDPC property.

Lemma 3.5.5. If $C_\bullet$ is ω-limited then the basis-changing logical measurement code $R_\bullet$ is not generally limited.

Proof. If the original code is ω-limited, then ∂_1^V is also ω-limited. If we take any bit e_i in V_1 and map it to V_0 then the result $\partial_1^V e_i$ has weight at most ω. Take the same bit and, using the equality $U_1 = V_1$, map it to U_0. $|\partial_1^U e_i| \leq 2$. Then the map ι_0 takes $\partial_1^U e_i$ and maps it to $\partial_1^V e_i$, so the column weight of ι_0 is at most ω.

For the row weight, however, we must be careful. $\iota_0 \circ \partial_1^U = \partial_1^V$, and ∂_1^U has a right inverse $(\partial_1^U)^R$ of the form

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 1 \\ 0 & 1 & 1 & \cdots & 1 & 1 \\ 0 & 0 & 1 & \cdots & 1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

and so $\iota_0 = \partial_1^V \circ (\partial_1^U)^R$. As $(\partial_1^U)^R$ is not limited in column weight, ι_0 is not limited in row weight.

Because ι_0 determines the connectivity of the first slice X-checks with the second slice qubits, the code $R_\bullet$ is not limited by the initial weights, and so is not generally LDPC. ■

Evidently, there are cases where $R_\bullet$ is LDPC, for example if $V_\bullet = U_\bullet$, as then $\iota_0 = \mathrm{id}_{n-1}$. For the structure of ∂_1^V to maintain the LDPC property there must be enough collisions to zero out columns of $(\partial_1^U)^R$. It would be interesting to know what class of classical codes this is.

It is interesting that both this basis-change method and the gauge-fixing approach of [CHRY24] can eliminate new logicals, but cannot guarantee that the LDPC property will be preserved in general.

We can attempt to fix this by decomposing the basis change over multiple layers, instead of changing to the repetition code immediately. As this becomes substantially more complicated, we leave this for future work. Similarly, we will generalise to a basis changes to other codes – not just the ideal repetition code – in order to ‘convert’ from one logical to another to perform parity measurements in a single ancilla patch without requiring the basis-preserving monic span of Definition 2.3.11. Lastly, we aim to generalise to logicals with differing numbers of qubits, so that the injection ι is nontrivial at degree 1 as well.

Part B

Hopf algebraic codes

Chapter 4

Quantum double aspects of Kitaev models

The idea of fault-tolerant quantum computing using topological methods has been around for some years now, notably the Kitaev model in the original work[Kit03] and important sequels such as [BM-D08, BSW11, BMCA13, Meu17]. Here we add to this growing body of literature with a renewed focus on the quantum double $D(G)$ Hopf algebra symmetry implicit in the original Kitaev model, where G is a finite group. The model here is built on a suitable oriented graph but for our purposes we focus on a fixed oriented square lattice. The Hilbert space $\mathcal{H}$ of the system is then the tensor product over all arrows of a vector space with basis G at every arrow. Every site, by which we mean a choice of a face and vertex on it, carries a representation of the quantum group $D(G)$. In general 'quasiparticles' in the model are defined as irreducible representations of this quantum group and we explain how these can be detected using certain projection operators $P_{\mathcal{C},\pi}$, where $\mathcal{C}$ is a conjugacy class in G and π is an irreducible representation of the isotropy group. We then study quasiparticles at the end-points s_0, s_1 of an open ribbon ξ, again taking a $D(G)$ approach to the ribbon operator commutation relations. Most of these results are in Section 4.2 but a more sophisticated view of ribbon operators as left and right module maps $F_\xi : D(G)^* \to \mathrm{End}(\mathcal{H})$ is deferred to Section 4.3 as a warm up for the generalisation there.

Of particular interest in this Chapter is the space $\mathcal{L}(s_0, s_1)$ of states created from a local vacuum by all possible ribbon operations F_ξ for a fixed ξ. This was a key ingredient in [Kit03] and its independence as a subspace of $\mathcal{H}$ on deformations of the ribbon expresses the topological nature of the model. Our results here build on ideas in [BSW11] whereby this space carries the left action of $D(G)$ at s_0 and another action, which we view as a right action, at s_1. The space is then isomorphic to $D(G)$ itself as a bimodule under left and right multiplication and hence subject to its Peter-Weyl decomposition as a direct sum of $\mathrm{End}(V_{\mathcal{C},\pi})$ over all quasiparticle irrep spaces $V_{\mathcal{C},\pi}$. We use this to create a state $|\mathrm{Bell};\xi\rangle \in \mathcal{L}(s_0, s_1)$ and show that this can be used to teleport information between s_0, s_1. We illustrate the theory further as well as give more details and examples of quantum computations for $D(S_3)$ in Section 4.2.5, where S_3 is the group of permutations on 3 elements. Likewise, the theory simplifies but carries some of the same structures in the toric case $D(\mathbb{Z}_n)$ for which the ribbon theory is in Section 4.2.4.

The Chapter begins with a preliminary warm up Section 4.1 which sets up the basic ideas as this easier level of $D(\mathbb{Z}_n)$ but from the point of view of this as $\mathbb{Z}_n \times \mathbb{Z}_n$ with a certain factorisable quastriangular structure in the sense of Drinfeld[Dri87]. The body of the Chapter concludes in Section 4.3 with some partial results going the other way to $D(H)$, where H is a finite-dimensional Hopf algebra. The Kitaev theory at this level but with H semisimple so that (over $\mathbb{C}$) we have $S^2 = \mathrm{id}$ was introduced in [BMCA13] where it was was shown that one has a $D(H)$ action at every site, but without explicitly considering ribbons. The latter, however, are special cases of 'holonomy maps' in the follow-up work [Meu17], again in the semisimple case. This work focusses more on the topological and 'gauge theory' aspects rather than ribbon operators specifically, thus at the very least we aim in the semisimple case for a much more explicit treatment of what is already known in some form. Thus, our main result of the section on ribbon operators as left and right module maps $D(H)^* \to \mathrm{End}(\mathcal{H})$ is similar to [Meu17, Thm 8.1] except that that applies to a special class of holonomy operators that explicitly do not include ribbon ones, and our proofs are much more explicit. For example, the equivariance of the smallest open ribbons (which are base for our induction) is proven in Figures 4.8, 4.9 by Hopf algebra calculations.

The bottom line, however, is that the theory is known to generalise well to H semisimple and the most novel aspect of Section 4.3 is that we do as much as we can without assuming this. Computationally speaking, the H non-semisimple case loses the interpretation of the integral actions as check operators which are measured to detect unwanted excitations. In addition, ribbon operators on the vacuum are no longer in general equivalent up to isotopy. For this reason, the logical space does not enjoy the same 'topological protection' as the semisimple case. On the other hand, we find that there is no problem with a $D(H)$ action at every site, but for dual-triangle operators and ribbon operators involving them, we need two versions ${}^{(\pm)}L$ depending one whether we use $S^{\pm 1}$ at the relevant incoming arrow. That means that the same ribbon operator is not a module map from both the left and the right at the same time. This obstruction can also be put on the faces and is not a deal breaker, but requires more study for a fully worked out theory. For example, in the quasitriangular Hopf algebra case the two are equivalent by conjugation, $S = uS^{-1}(\ )u^{-1}$ for Drinfeld's element $u \in H$ in [Dri87]. Thus, this aspect of Section 4.3 should be seen as first steps in a fully general Kitaev theory.

In fact such a more general theory is needed in order to apply to quantum groups such as $u_q(sl_2)$ at roots of unity, which in turn would be needed to connect up to ideas for quantum computing based on modular tensor categories associated to such non-semisimple quantum groups. For example, the Fibonacci anyons surveyed in [TTWL08] are based on $u_q(sl_2)$ at $q^5 = 1$. The double $D(u_q(sl_2))$ here also underlies the Turaev-Viro invariant of 3-manifolds and hence this should certainly be a source of topological stability if the Kitaev model can be extended to such cases. If so, it would then be related closely to $2+1$ quantum gravity with point sources, which is a viable theory and another reason to expect that this is ultimately possible. There are many other obstacles also, however, to such a programme, some of which are discussed in the final Section 4.4. We also discuss there other issues for topological quantum computing and possible links with ZX-calculus.

Remark 4.0.1. While completing the writing of other parts of the Chapter, there appeared the preprint [YCC22] which covers some of the same ground as Section 4 with regard to the ribbon operators in the semisimple case where $S^2 = \mathrm{id}$. Our approach is different and is, moreover, directed to exposing the issues for the general non-semisimple case.

4.1 Preliminaries: $D(\mathbb{Z}_n)$ model

Let $\mathbb{C}\mathbb{Z}_n$ denote the group Hopf algebra with generator h where $h^n = 1$ and $\Delta h = h \otimes h$, $\epsilon h = 1$, $Sh = h^{-1}$ for the coproduct, counit and antipode. Let $\mathbb{C}(\mathbb{Z}_n)$ be the Hopf algebra of functions on $\mathbb{Z}_n$ with a basis of δ-functions on $\mathbb{Z}_n = 0, 1, \cdots, n-1$ and $\Delta\delta_i = \sum_j \delta_j \otimes \delta_{i-j}$, $\epsilon(\delta_i) = \delta_{i,0}$, $S\delta_i = \delta_{-i}$ for the Hopf algebra structure. The normalised integrals in these Hopf algebras are

$$\Lambda = \frac{1}{n}\sum_i h^i \in \mathbb{C}\mathbb{Z}_n, \quad \Lambda^* = \delta_0 \in \mathbb{C}(\mathbb{Z}_n).$$

The quantum double $D(\mathbb{Z}_n) = \mathbb{C}(\mathbb{Z}_n) \otimes \mathbb{C}\mathbb{Z}_n \cong \mathbb{C}\mathbb{Z}_n \otimes \mathbb{C}\mathbb{Z}_n \cong \mathbb{C}.\mathbb{Z}_n \times \mathbb{Z}_n$ as Hopf algebras, since the groups are Abelian, and since (over $\mathbb{C}$) $\mathbb{C}\mathbb{Z}_n \cong \mathbb{C}(\mathbb{Z}_n)$ by the Fourier isomorphism

$$g \mapsto \sum_i q^i \delta_i, \quad \delta_i \mapsto \frac{1}{n}\sum_k q^{-ik} g^k, \tag{4.1}$$

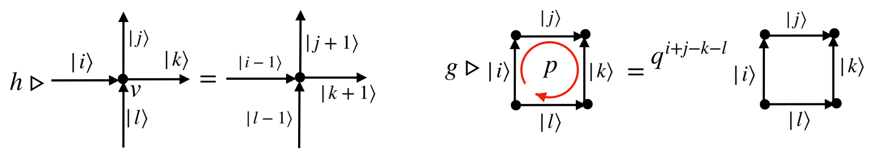
where q is a primitive nth root of unity. Now the double is $\mathbb{C}\mathbb{Z}_n \otimes \mathbb{C}\mathbb{Z}_n$ with generators g, h respectively for the two copies, commuting and obeying $h^n = g^n = 1$. Under this isomorphism, the general $D(G)$ theory in Section 4.2 looks much simpler and we therefore treat this case first as a model for the later sections.

Denoting the generators of the two copies of $\mathbb{C}\mathbb{Z}_n$ in this form of the double by g, h respectively, the $D(\mathbb{Z}_n)$ quasitriangular structure is

$$\mathcal{R} = \frac{1}{n}\sum_{i,j} q^{-ij} g^i \otimes h^j,$$

where we will see $\mathcal{R} = \sum_j \delta_j \otimes h^j$ for $D(\mathbb{Z}_n)$ according to 4.2 given later.

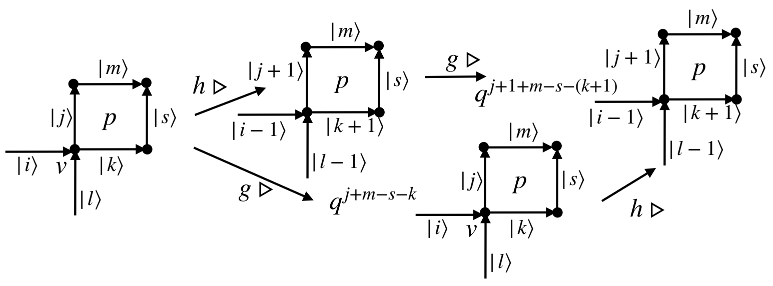
Now let $\Sigma = \Sigma(V, E, P)$ be a square lattice viewed as a directed graph with its usual (cartesian) orientation. The Hilbert space will be a tensor product of vector spaces with one copy of $\mathbb{C}\mathbb{Z}_n$ at each arrow $e \in E$. We denote the basis of each copy by $|i\rangle$. Next, for each vertex $v \in V$ and each face $p \in P$ we define an action of $\mathbb{Z}_n$ which acts on the vector spaces around the vertex or around the face, and trivially elsewhere, according to



These are built from four-fold copies of the operator X and its adjoint and of Z and its adjoint, where $X|i\rangle = |i+1\rangle$ and $Z|i\rangle = q^i|i\rangle$ obey $ZX = qXZ$. Here $h\triangleright$ subtracts in the case of arrows pointing towards the vertex and $g\triangleright$ has k, l entering negatively in the exponent because these are contra to a *clockwise* flow around the face in our conventions. These combine to an action of $\mathbb{Z}_n \times \mathbb{Z}_n$ at every 'site' (v, p) defined as a vertex v and an adjacent face p (the exact placement of v in relation to p is not relevant in an Abelian group model such as this).

Lemma 4.1.1. *For every site (v, p), the operators $g\triangleright$ and $h\triangleright$ commute and give a representation of $\mathbb{Z}_n \times \mathbb{Z}_n$ on the Hilbert space $\mathcal{H}$.*

Proof. This is a direct calculation acting on the 6 relevant vector spaces, of which two are in common to the two actions, see



■

The same applies trivially if v and p are not adjacent (as they have no arrow in common). Thus, we can in fact consider $h\triangleright$ determined by a vertex v and $g\triangleright$ determined by a face p independently. With this in mind, we define

$$A(v) = \Lambda\triangleright = \frac{1}{n}\sum_m (h\triangleright)^m, \quad B(p) = \Lambda^*\triangleright = \frac{1}{n}\sum_m (g\triangleright)^m$$

where now $\Lambda^* = n^{-1}\sum_i g^i$ according to (5.1), and these necessarily commute. In fact it is easy to see that

$$A(v)^2 = A(v), \quad B(p)^2 = B(p), \quad [A(v), A(v')] = [B(p), B(p')] = [A(v), B(p)] = 0.$$

We then define the Hamiltonian

$$H = \sum_v (1 - A(v)) + \sum_p (1 - B(p)) = -(\sum_v A(v) + \sum_p B(p)) + \text{const.}$$

and define the set of vacuum states

$$\mathcal{H}_{vac} = \{|\xi\rangle \in \mathcal{H} \mid A(v)|\xi\rangle = B(p)|\xi\rangle = |\xi\rangle, \forall v, p\}.$$

Vacuum states are 'topologically protected' from errors which are sufficiently local, which we will make precise later.

Next, the irreducible representations of the double in this form are

$$\pi_{ij}(g) = q^i, \quad \pi_{ij}(h) = q^j,$$

which as all 1-dimensional. We denote these by

$$1 = \pi_{00}, \quad e_i = \pi_{0i}, \quad m_i = \pi_{i0}, \quad \epsilon_{ij} = \pi_{ij}, \quad i, j \in 1, \cdots, n-1$$

with braiding

$$\Psi_{1,*} = \Psi_{*,1} = \Psi_{e_i,e_j} = \Psi_{m_i,m_j} = \Psi_{e_i,m_j} = \Psi_{\epsilon_{jk},m_i} = \Psi_{e_i,\epsilon_{jk}} = 1,$$

$$\Psi_{m_i,e_j} = q^{ij}, \quad \Psi_{m_i,\epsilon_{jk}} = q^{ik}, \quad \Psi_{\epsilon_{jk},e_i} = q^{ij}, \quad \Psi_{\epsilon_{ij},\epsilon_{kl}} = q^{il}$$

where $\Psi_{u,v} = \mathcal{R}^{(2)}\triangleright v \otimes \mathcal{R}^{(1)}\triangleright u = \frac{1}{n}\sum_{i,j} q^{-ij} h^j\triangleright v \otimes g^i\triangleright u$. Next, we define projectors associated to π_{ij} namely

$$P_{ij} = \frac{1}{n^2}\sum_{kl} (\mathrm{Tr}_{\pi_{ij}} g^{-k}h^{-l}) g^k h^l = P^g_i P^h_j, \quad P^g_i = \frac{1}{n}\sum_k q^{-ik} g^k$$

in the group algebra of $\mathbb{Z}_n \times \mathbb{Z}_n$, built from projectors in each $\mathbb{Z}_n$ (here P^h_j is defined in the same way on the other copy). The projectors on one copy obey $P^g_i P^g_j = \delta_{ij} P^g_i$ and $\sum_i P^g_i = 1$ and similarly for P^h_j, so that

$$P_{ij} P_{i'j'} = \delta_{ii'} \delta_{jj'} P_{ij}, \quad \sum_{i,j} P_{ij} = 1.$$

At every vertex v, every face p and every site (v,p), we have specific projection operators on $\mathcal{H}$ given by

$$P_i(p) = P^g_i {\triangleright}, \quad P_j(v) = P^h_j {\triangleright}, \quad P_{ij}(v,p) = P_{ij} {\triangleright}$$

for the actions above on the relevant arrows. We consider these orthogonal projectors as measurement outcomes dictating, for $i, j \neq 0$:

- $P_i(p)$ – *there is a quasiparticle of type* m_i *occupying face* p
- $P_j(v)$ – *there is a quasiparticle of type* e_j *occupying vertex* v
- $P_{ij}(v,p)$ – *there is a quasiparticle of type* ϵ_{ij} *occupying site* (v,p)

which, combined, make two projective measurements at a site (v,p), as the outcomes for m_i and e_j are independent. The corresponding quantum mechanical observables are the self-adjoint operators $O_p = \sum_i r_i P_i(p)$ and $O_v = \sum_j t_j P_j(v)$, where each $r_i \in \mathbb{R}$ is distinct and the same for each t_j. In particular we acquire the outcome $P_{00}(v,p)$ when there is a trivial representation quasiparticle at (v,p), which is equivalent to the absence of the above excitations, i.e. we regard it as a local vacuum. Note also that

$$P_0(v) = A(v), \quad P_0(p) = B(p), \quad P_{00}(v,p) = A(v)B(p)$$

which gives the meaning of these. Thus $A(v)$ specifies that there is no excitation at the vertex independently of the face, etc.

Lemma 4.1.2. *Let $|\psi\rangle \in \mathcal{H}$. For all $i, j \in \mathbb{Z}_n$:*

1. *$P_i(p)|\psi\rangle = |\psi\rangle$ if and only if $g{\triangleright}|\psi\rangle = q^i|\psi\rangle$ for the four arrows around p.*
2. *$P_j(v)|\psi\rangle = |\psi\rangle$ if and only if $h{\triangleright}|\psi\rangle = q^j|\psi\rangle$ for the four arrows around v.*
3. *$P_{ij}(v,p)|\psi\rangle = |\psi\rangle$ if and only if $g{\triangleright}|\psi\rangle = q^i|\psi\rangle$ and $h{\triangleright}|\psi\rangle = q^j|\psi\rangle$ for the six arrows at the site.*
4. *$|\mathrm{vac}\rangle \in \mathcal{H}_{vac}$ if and only if $P_{ij}(v,p)|\mathrm{vac}\rangle = 0$ for all $(i,j) \neq (0,0)$ and at all sites (v,p).*

On a ‘closed plane’, which we can consider to be a plane where we ignore boundary effects, there is a unique vacuum state (up to normalisation):

$$|\mathrm{vac}\rangle = \prod_{v \in V} A(v) \bigotimes_E |0\rangle$$

where 0 is the group identity of $\mathbb{Z}_n$.

Proof. (1) $P_i(p)$ acts on the four-arrow state $|i_1\rangle\otimes|i_2\rangle\otimes|i_3\rangle\otimes|i_4\rangle$ in order around the face by $\frac{1}{n}\sum_k q^{-ia}q^{a(i_1+i_2-i_3-i_4)}=\delta_{i,i_1+i_2-i_3-i_4}$. So invariant states are linear combinations of ones with $i_1+i_2-i_3-i_4=i$ going around the face. These are precisely the local states where $g{\triangleright}|\psi\rangle=q^i|\psi\rangle$.

(2) Linear combinations of $|i_i\rangle\otimes|i_2\rangle\otimes|i_3\rangle\otimes|i_4\rangle$ in order around the vertex that are invariant under $P_j(v)$ are of the form

$$|\psi\rangle=\sum_b q^{-jb}|i_1-b\rangle\otimes|i_2+b\rangle\otimes|i_3+b\rangle\otimes|i_4-b\rangle$$

and these are also the local states where

$$h{\triangleright}|\psi\rangle=\sum_b q^{-jb}|i_1-b-1\rangle\otimes|i_2+b+1\rangle\otimes|i_3+b+1\rangle\otimes|i_4-b-1\rangle=q^j|\psi\rangle$$

(3) Considering the site (v,p) with p to the upper right of v as before, the joint eigenvectors for (1) and (2) are of the form

$$|\psi\rangle=\sum_b q^{-jb}|i_1-b\rangle\otimes|i_2+b\rangle\otimes|i_3+b\rangle\otimes|i_4-b\rangle\otimes|i_5\rangle\otimes|i_6\rangle;\quad i_2+i_5-i_6-i_3=i$$

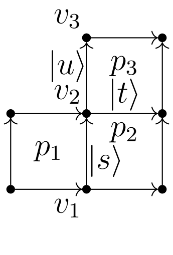
where we take them in order round the vertex then around the face. These are also the local states where $g{\triangleright}|\psi\rangle=q^i|\psi\rangle$ and $h{\triangleright}|\psi\rangle=q^j|\psi\rangle$.

(4) We just note that $P_0(v)=A(v)$, $P_0(p)=B(p)$ so $P_{00}(v,p)=A(v)B(p)$. So if $|\psi\rangle\in\mathcal{H}_{vac}$ then $P_{00}|\psi\rangle=|\psi\rangle$ i.e. there are no excitations, at every site (v,p). Moreover, for $(i,j)\neq(0,0)$, $P_{ij}|\psi\rangle=P_{ij}P_{00}|\psi\rangle=0$ by the projector orthogonalilty, again at every site. Conversely, if $P_{ij}|\psi\rangle=0$ for all $(i,j)\neq(0,0)$ at (v,p) then $P_{00}|\psi\rangle=|\psi>$ as $\sum_{ij}P_{ij}=1$ while $A(v)|\psi\rangle=\sum_i P_{i0}|\psi\rangle=P_{00}|\psi\rangle=|\psi\rangle$ and similarly for $B(p)$. If this is true at every site then $|\psi\rangle\in H_{vac}$.

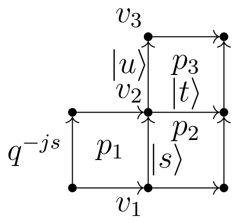
Note that (4) is the same as saying that if the system is in a vacuum state there is no excitation at any site. We can see this directly as $h{\triangleright}\circ\Lambda{\triangleright}=\Lambda{\triangleright}$ at a given vertex. So if $|\mathrm{vac}\rangle$ is in $\mathcal{H}_{vac}$ as defined above then $h{\triangleright}|\mathrm{vac}\rangle=h{\triangleright}A(v)|\mathrm{vac}\rangle=h\Lambda{\triangleright}|\mathrm{vac}\rangle=\Lambda{\triangleright}|\mathrm{vac}\rangle=|\mathrm{vac}\rangle$. Similarly for $g{\triangleright}|\mathrm{vac}\rangle=|\mathrm{vac}\rangle$. This agrees with the analysis above. The vacuum state $|\mathrm{vac}\rangle$ may be verified by directly checking the definition of $\mathcal{H}_{vac}$. We will see later that this state is unique in $\mathcal{H}_{vac}$ as a special case of Corollary 4.2.3.

4.1.1 Quasiparticle creation and transportation

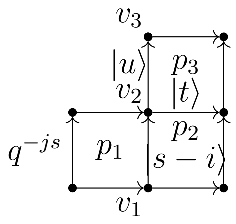
We now consider concretely how to create quasiparticles on the lattice. Assume the system has state $|\mathrm{vac}\rangle\in\mathcal{H}_{vac}$. Consider the arrow between vertices v_2 and v_1 on the boundary of faces p_1 and p_2,



For some $j \in \mathbb{Z}_n$, consider Z^{-j} acting on $|s\rangle$, which we denote Z_s^{-j} and takes $|s\rangle \mapsto q^{-sj}|s\rangle$:

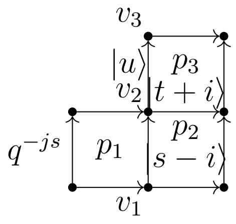


Then, $h\triangleright_{v_1} Z_s^{-j}|\text{vac}\rangle = q^j Z_s^{-j} h\triangleright_{v_1}|\text{vac}\rangle = q^j Z_s^{-j}|\text{vac}\rangle$ and similarly $h\triangleright_{v_2} Z_s^{-j}|\text{vac}\rangle = q^{-j} Z_s^{-j}|\text{vac}\rangle$, which is easy to check using commutation relations. By Lemma 4.1.2, all neighbouring sites (v_1, p_a) and (v_2, p_a) are occupied by m_j and m_{-j}, where p_a is any neighbouring face. Let X^{-i} further act on $Z_s^{-j}|s\rangle$ alone, for some $i \in \mathbb{Z}_n$:

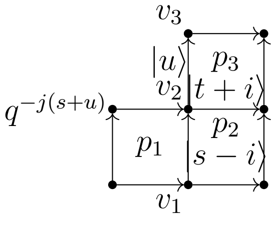


Now, $g\triangleright_{p_1}(X^{-i}Z^{-j})_s|\text{vac}\rangle = q^i (X^{-i}Z^{-j})_s g\triangleright_{p_1}|\text{vac}\rangle = q^i (X^{-i}Z^{-j})_s|\text{vac}\rangle$ and $g\triangleright_{p_2}(X^{-i}Z^{-j})_s|\text{vac}\rangle = q^{-i}(X^{-i}Z^{-j})_s|\text{vac}\rangle$. All neighbouring sites (v_b, p_1) and (v_b, p_2) are now occupied by a quasiparticle e_i and e_{-i} respectively, where v_b is any neighbouring vertex. In particular, (v_1, p_1) is occupied by $\pi_{i,j}$, while (v_2, p_2) is occupied by $\pi_{-i,-j}$.

Quasiparticles may be moved on the surface by X and Z edge operations. We next apply X^i to $|t\rangle$:



Now, $g\triangleright_{p_2} X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle = X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle$ (being careful about edge orientation). Site (v_2, p_2) is now only occupied by m_{-j}. However, the previously unoccupied site (v_3, p_3) is now occupied by e_{-i}, as $g\triangleright_{p_3} X_t^i|\vec{p}_3\rangle = q^i X_t^i|\vec{p}_3\rangle$. Now further apply Z^{-j} acting on $|u\rangle$:



$h\triangleright_{v_2} Z_u^{-j} \otimes X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle = Z_u^{-j} \otimes X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle$, and so site (v_2, p_2) is now unoccupied. Site (v_3, p_3) is occupied by $\pi_{-i,-j}$, as $h\triangleright_{v_3} Z_u^{-j} \otimes X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle = q^{-j} Z_u^{-j} \otimes X_t^i \otimes (X^{-i}Z^{-j})_s|\text{vac}\rangle$. This explanation of creation and transport is quite *ad hoc*. In fact, the above operators are specific instances of ribbon operators, which we describe in Section 4.2. We delay discussing braiding until then, as it is clearer in terms of ribbons.

4.2 $D(G)$ models and example of $D(S_3)$

The models described in this Section are the primary subject of Kitaev's original paper [Kit03], and while some of the results here have been described in some form either there or elsewhere, see [BSW11, BM-D08], we aim to be explicit and formal in the presentation. In addition, we believe that our account of how to utilise the Peter-Weyl isomorphism for ribbon operators is novel at least in its level of detail, as is the description of a generalised quantum teleportation-like protocol.

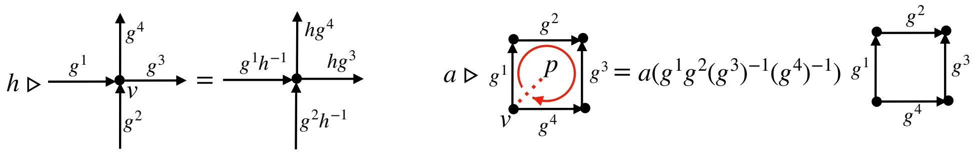
Let G be a finite group with identity $e \in G$. We recall that the group Hopf algebra $\mathbb{C}G$ base basis G with product extended linearly and $\Delta h = h \otimes h$, $\epsilon h = 1$ and $Sh = h^{-1}$ for the Hopf algebra structure. Its dual Hopf algebra $\mathbb{C}(G)$ of functions on G has basis of δ-functions $\{\delta_g\}$ with $\Delta\delta_g = \sum_h \delta_h \otimes \delta_{h^{-1}g}$, $\epsilon\delta_g = \delta_{g,e}$ and $S\delta_g = \delta_{g^{-1}}$ for the Hopf algebra structure. The normalised integrals are

$$\Lambda = \frac{1}{|G|}\sum_{h\in G} h \in \mathbb{C}G, \quad \Lambda^* = \delta_e \in \mathbb{C}(G).$$

For the Drinfeld double we have $D(G) = \mathbb{C}(G){>\!\!\!\triangleleft}\mathbb{C}G$, see [Maj95], with $\mathbb{C}G$ and $\mathbb{C}(G)$ subalgebras and the cross relations $h\delta_g = \delta_{hgh^{-1}}h$ (a semidirect product). We will often prefer to refer to $D(G)$ explicitly on the tensor product vector space, then for example the cross relation appears explicitly as $(1\otimes h)(\delta_g \otimes 1) = (\delta_{hgh^{-1}} \otimes 1)(1\otimes h) = \delta_{hgh^{-1}} \otimes h$ and antipode as $S(\delta_g \otimes h) = \delta_{h^{-1}g^{-1}h} \otimes h^{-1}$. There is also a quasitriangular structure which in the subalgebra notation is

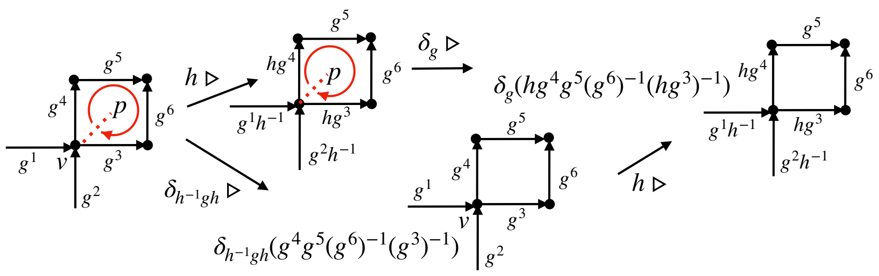
$$\mathcal{R} = \sum_{h\in G} \delta_h \otimes h \in D(G)\otimes D(G). \tag{4.2}$$

More relevant to us is the representation on Hilbert space $\mathcal{H}$, which now is a tensor product of $\mathbb{C}G$ at each arrow. As before, this is associated to a pair (v,p) (a 'site') where v is a vertex on the boundary of the face p. What is different from the Abelian group case in Section 4.1 is that now for the $a\triangleright$ action on $\mathcal{H}$ we have to pay attention to the exact placement of v in relation to p by drawing dashed line (the 'cilium') between v and the interior of p and taking the group elements in order around the face according to



Lemma 4.2.1. $h\triangleright$ and $a\triangleright$ for all $h\in G$ and $a\in\mathbb{C}(G)$ define a representation of $D(G)$ on $\mathcal{H}$ associated to each site (v,p).

Proof. This follows from the definitions and a check acting on the six affected arrow spaces, see



■

We next define

$$A(v) = \Lambda\triangleright = \frac{1}{|G|}\sum_{h\in G} h\triangleright, \quad B(p) = \Lambda^*\triangleright = \delta_e\triangleright$$

where $\delta_e(g^1g^2g^3g^4) = 1$ iff $g^1g^2g^3g^4 = e$ which is iff $(g^4)^{-1} = g^1g^2g^3$ which is iff $g^4g^1g^2g^3 = e$. Hence $\delta_e(g^1g^2g^3g^4) = \delta_e(g^4g^1g^2g^3)$ is invariant under cyclic rotations, hence $\Lambda^*\triangleright$ computed at site (v,p) does not depend on the location of v on the boundary of p. Moreover,

$$A(v)B(p) = |G|^{-1}\sum_h h\delta_e\triangleright = |G|^{-1}\sum_h \delta_{heh^{-1}}h\triangleright = |G|^{-1}\sum_h \delta_e h\triangleright = B(p)A(v)$$

if v is a vertex on the boundary of p by Lemma 4.2.1, and more trivially if not. We also have the rest of

$$A(v)^2 = A(v), \quad B(p)^2 = B(p), \quad [A(v), A(v')] = [B(p), B(p')] = [A(v), B(p)] = 0$$

for all $v \neq v'$ and $p \neq p'$, as easily checked. We then define

$$H = \sum_v (1 - A(v)) + \sum_p (1 - B(p))$$

and the space of vacuum states

$$\mathcal{H}_{vac} = \{|\psi\rangle \in \mathcal{H} \mid A(v)|\psi\rangle = B(p)|\psi\rangle = |\psi\rangle, \quad \forall v, p\}.$$

4.2.1 Vacuum space

The vacuum space degeneracy depends on the surface topology. Here and throughout the Chapter, we describe everything very concretely using a square lattice for convenience. While this is obviously possible for a plane, more general surfaces may not admit such a tiling. Precisely, the only 2-dimensional closed orientable surface which admits a (4, 4) tessellation is the torus, which follows from [EEK82, Thm 1]. However, the following well-known theorem, and results throughout this Chapter, apply for other (ciliated, ribbon) graphs embedded into a closed orientable surface. We avoid getting into the weeds on the subject of topological graph theory, but observe that while the lattice will primarily be square, in some places there will have to be irregular faces or vertices. Face and vertex operators generalise straightforwardly to such irregularities.

Theorem 4.2.2. Let Σ be a closed, orientable surface. Then

$$\dim(\mathcal{H}_{vac}) = |\mathrm{Hom}(\pi_1(\Sigma), G)/G|.$$

where the G-action on any $\phi \in \mathrm{Hom}(\pi_1(\Sigma), G)$ is $\phi \mapsto \{h\phi h^{-1} \mid h \in G\}$.

For completeness we prove this in the Appendix 12, mostly following [CDH20], and in the process presenting an orthogonal basis for $\mathcal{H}_{vac}$. This implies, in particular:

Corollary 4.2.3. Let Σ be planar, with no boundaries. Then the vacuum state $|\mathrm{vac}\rangle$ is unique up to normalisation, and

$$|\mathrm{vac}\rangle = \prod_{v\in V} A(v) \bigotimes_E e$$

where e is the group identity of G and $\otimes$ is over the arrows.

Proof. We have assumed that $\pi_1(\Sigma) = \{e\}$ and clearly $\mathrm{Hom}(\{e\}, G) = \{e\}$, $\{e\}/G = \{e\}$. Hence, the vacuum is unique. To find it, define $g := \bigotimes_E e$, and observe that $B(p)g = g$ for all $p \in P$, so $g \in S$. Since $B(p)$ commutes with every $A(v)$ commute with, it follows that $B(p)|\mathrm{vac}\rangle = |\mathrm{vac}\rangle$. Moreover, applying $A(v)$ for a fixed v to $|\mathrm{vac}\rangle$, this combines with $A(v)$ in the product to give $A(v)$ again, hence $A(v)|\mathrm{vac}\rangle = |\mathrm{vac}\rangle$. Hence, we have constructed the vacuum state. ■

We specify that the plane has no boundaries for Corollary 4.2.3 because Theorem 4.2.2 holds only for *closed* surfaces; the 'plane' can then be thought of as an infinite sphere. The treatment of boundaries requires adding more algebraic structure to the model, and in general splits vacuum degeneracy [BSW11]. It is also obvious that if Σ is a closed orientable surface and G is Abelian so that the G-action by conjugation is trivial, then

$$\dim(\mathcal{H}_{vac}) = |\mathrm{Hom}(\pi_1(\Sigma), G)|.$$

The Kitaev model may be used to perform fault-tolerant quantum computation – indeed, the $D(G)$ model corresponds to a class of quantum error-correcting codes in the sense of [KL00], according to [CDH20]. If we consider the vacuum to be the logical space of a quantum computer and by following the proof of Theorem 4.2.2, we observe that the only non-trivial operators in $\mathrm{End}(\mathcal{H}_{vac})$ are non-contractible closed loops on the lattice. Operators which do not form closed paths take the system out of $\mathcal{H}_{vac}$, and introduce excitations. In particular, considering the quantum computer to be operating in a noisy environment, errors on the lattice which introduce unwanted excitations may be detected using the projectors $A(v), B(p)$ and corrected. Undetectable errors must therefore be sufficiently non-local as to form undetectable non-trivial holonomies; we thus refer to the logical state of the computer as being 'topologically protected'.

To run algorithms of practical interest, the model must be capable of supporting a large Hilbert space, but Corollary 4.2.3 tells us that a boundary-less plane is only capable of supporting a single vacuum state. There are therefore 3 methods of encoding data in Kitaev models:

1. Build the lattice Σ as a torus with k holes, which can encode data in the degenerate vacuum state using $\pi_1(\Sigma)$.

2. Incorporate *gapped boundaries* or *topological defects* into the lattice, which are compatible (in some suitable sense) with the algebra of $D(G)$ and allow for additional vacuum states [BSW11, BM-D08].

3. Use excited states to encode data. This method requires that G be non-Abelian, as the $D(G)$ model does not admit degenerate excited states on the plane when G is Abelian without the addition of topological features such as boundaries [Kit03].

4.2.2 Quasiparticles and projection operators to detect them

We now return to the underlying algebra of the Kitaev model. The 'quasiparticles' in the theory are labelled by irreducible representations of $D(G)$. A couple of standard but not generally irreducible right representations of $D(G)$ on $\mathbb{C}G$ itself are

$$\text{(i)}\quad g{\triangleleft}h = gh, \quad g{\triangleleft}\delta_h = g\delta_{h,e}; \quad \text{(ii)}\quad g{\triangleleft}h = h^{-1}gh, \quad g{\triangleleft}\delta_h = g\delta_{g,h}.$$

More generally, as a semidirect product, irreducible representations of $D(G)$ are given by standard theory as labelled by pairs $(\mathcal{C},\pi)$ consisting of an orbit under the action (i.e. by a conjugacy class $\mathcal{C}\subset G$ in the present case) and an irrep π of the isotropy subgroup C_G of a fixed element $r_{\mathcal{C}}\in\mathcal{C}$ (in our case its centraliser i.e. $n\in G$ such that $nr_{\mathcal{C}}=r_{\mathcal{C}}n$), the choice of which does not change the group up to isomorphism but does change how it sits inside G. Here $\mathcal{C}$ is called the 'magnetic charge' and π is called the 'electric charge'. Special cases corresponding to e_i and m_i respectively in the $D(\mathbb{Z}_n)$ case are

$$\text{chargeons}\quad (\{e\},\pi),\quad \delta_g h{\triangleright} w=\delta_{g,e}\pi(h)w;\quad \text{fluxions}\quad (\mathcal{C},1),\quad \delta_g h{\triangleright} c=\delta_{g,hch^{-1}}hch^{-1}$$

acting on the representation space V_π of π as an irrep of G, and the span $\mathbb{C}\mathcal{C}$ of the conjugacy class, respectively. The braiding of two fluxions or a fluxion with a chargeon, for example, are

$$\Psi(f\otimes f')=\sum_g g{\triangleright} f'\otimes\delta_g{\triangleright} f=ff'f^{-1}\otimes f,\quad \Psi(f\otimes w)=\sum_g g{\triangleright} w\otimes\delta_g{\triangleright} f=\pi(f)w\otimes f.$$

The irrep associated to general $(\mathcal{C},\pi)$ can be described as follows[Maj04]. First, fix a map

$$q:\mathcal{C}\to G,\quad q_c r_{\mathcal{C}} q_c^{-1}=c,\quad \forall c\in\mathcal{C}, \tag{4.3}$$

and define from this a 'cocycle' $\zeta:\mathcal{C}\times G\to C_G$ respectively defined and characterised by

$$\zeta_c(g)=q^{-1}_{gcg^{-1}}gq_c;\quad \zeta_c(gh)=\zeta_{hch^{-1}}(g)\zeta_c(h)$$

for all $c\in\mathcal{C}$ and $g,h\in G$. The quantum double action on $\mathbb{C}\mathcal{C}\otimes V_\pi$ is then

$$\delta_g h{\triangleright}(c\otimes w)=\delta_{g,hch^{-1}}hch^{-1}\otimes\pi(\zeta_c(h))w. \tag{4.4}$$

This is irreducible and although the formulae depend on the choice of q, different choices give isomorphic representations. In particular, we can right multiply q_c by any element $n_c\in C_G$, and using this freedom we can suppose that

$$q_{r_{\mathcal{C}}}=e \tag{4.5}$$

which, in particular, ensures that (e,π) recovers the chargeon representation rather than an equivalent conjugate of it. Also note G is partitioned into the right cosets of C_G with the quotient space G/C_G identified with $\mathcal{C}$ by its action on $r_{\mathcal{C}}$. This implies that every element $g\in G$ can be uniquely factorised as $g=q_c n$ for some $c\in\mathcal{C}$ and $n\in C_G$.

We now describe the projectors[Maj04] that detect the presence of such quasiparticles, focussing first on the electric/chargeon sector. Then for each irrep π, such quasiparticles will be detected by measuring an observable $O=\sum_\pi r_\pi P_\pi{\triangleright} v$, where $r_\pi\in\mathbb{R}$ are all distinct, and v is a vertex; P_π is a central projection element (central idempotent) in the group algebra $\mathbb{C}G$ given by

$$P_\pi=\frac{\dim V_\pi}{|G|}\sum_g(\mathrm{Tr}_\pi\, g^{-1})g \tag{4.6}$$

These obey $P_\pi P_{\pi'}=\delta_{\pi,\pi'}P_\pi$ by the orthogonality of characters on finite groups, as well as $\sum_\pi P_\pi=1$ and $P_1=\Lambda$. Centrality is immediate by changing the variable g and symmetry of the trace. For reference, the orthogonality relations for characters on any finite group are

$$\sum_{h\in G}\mathrm{Tr}_\pi(h^{-1})\,\mathrm{Tr}_{\pi'}(hg)=\delta_{\pi,\pi'}\frac{|G|}{\dim(V_\pi)}\,\mathrm{Tr}_\pi(g) \tag{4.7}$$

$$\sum_{\pi\in\hat{G}} \mathrm{Tr}_\pi(g^{-1})\,\mathrm{Tr}_\pi(h) = \delta_{\mathcal{C}_g,\mathcal{C}_h}|C_G(g)| \tag{4.8}$$

for all $h, g \in G$ and $\pi, \pi' \in \hat{G}$ the set of irreps up to equivalence. Here $\mathcal{C}_g$ denotes the conjugacy class containing g. We likewise have a projection element $\chi_\mathcal{C}$ in $\mathbb{C}(G)$ defined as the characteristic function of $\mathcal{C}$ and $P_\mathcal{C}(v) = \chi_\mathcal{C}\triangleright v$ for all $v \in \mathcal{H}$ acting at any site. The general case is

$$P_{\mathcal{C},\pi} = \sum_{c\in\mathcal{C}} \delta_c \otimes q_c P_\pi q_c^{-1} = \frac{\dim V_\pi}{|C_G|}\sum_{c\in\mathcal{C}}\sum_{n\in C_G} \mathrm{Tr}_\pi(n^{-1})\delta_c \otimes q_c n q_c^{-1} \tag{4.9}$$

where $P_\pi \in \mathbb{C}C_G$ is for π as a representation of C_G, and associated site projection operators $P_{\mathcal{C},\pi}(v) = P_{\mathcal{C},\pi}\triangleright v$ for $v \in \mathcal{H}$ and action at a site. Here $\dim(V_\pi)/|C_G| = \dim(V_{\mathcal{C},\pi})/|G|$. Also note that

$$P_{e,1} = \Lambda^*\otimes\Lambda, \quad P_{e,\pi} = \Lambda^*\otimes P_\pi, \quad P_{\mathcal{C},1} = \sum_{c\in\mathcal{C}} \delta_c \otimes q_c \Lambda_{C_G} q_c^{-1},$$

the last of which in the Abelian case is $\delta_c\otimes\Lambda$ and recovers the chargeon and fluxion projections to the extent possible. We can also define for a fixed $\mathcal{C}$,

$$\sum_{\pi\in\hat{C}_G} P_{\mathcal{C},\pi} = \sum_{c\in\mathcal{C}} \delta_c \otimes q_c (\sum_{\pi\in\hat{C}_G} P_\pi) q_c^{-1} = \chi_\mathcal{C}\otimes 1.$$

What we can not do in the nonAbelian case is sum over $\mathcal{C}$ for a fixed nontrivial π as these depend on $\mathcal{C}$, so we do not have a formula like $\sum_\mathcal{C} P_{\mathcal{C},\pi} = 1\otimes P_\pi$.

Lemma 4.2.4. In $D(G)$, the $P_{\mathcal{C},\pi}$ are central and form a complete orthogonal set of projections,

$$P_{\mathcal{C},\pi}P_{\mathcal{C}',\pi'} = \delta_{\mathcal{C},\mathcal{C}'}\delta_{\pi,\pi'}P_{\mathcal{C},\pi}, \quad \sum_{\mathcal{C},\pi} P_{\mathcal{C},\pi} = 1 \tag{4.10}$$

Proof. This is due to [Maj04], but for completeness we now provide more explicit proofs than given there. Thus,

$$\begin{aligned} P_{\mathcal{C},\pi}P_{\mathcal{C}',\pi'} &= \sum_{c\in\mathcal{C},d\in\mathcal{C}'} (\delta_c\otimes q_c P_\pi q_c^{-1})(\delta_d \otimes q'_d P_{\pi'} {q'}_d^{-1}) \\ &= \frac{\dim V_\pi}{|C_G|}\sum_{c\in\mathcal{C},d\in\mathcal{C}'} \delta_c \sum_{n\in C_G} \mathrm{Tr}_\pi(n^{-1})\delta_{c,q_c n q_c^{-1} d q_c n^{-1} q_c^{-1}} \otimes q_c n q_c^{-1} q'_d P_\pi {q'}_d^{-1} \\ &= \frac{\dim V_\pi}{|C_G|}\sum_{c\in\mathcal{C},d\in\mathcal{C}'} \delta_c \sum_{n\in C_G} \mathrm{Tr}_\pi(n^{-1})\delta_{c,d} \otimes q_c n q_c^{-1} q'_d P_{\pi'} {q'}_d^{-1} \\ &= \delta_{\mathcal{C},\mathcal{C}'}\sum_{c\in\mathcal{C}} \delta_c\otimes q_c P_\pi P_{\pi'} q_c^{-1} = \delta_{\mathcal{C},\mathcal{C}'}\delta_{\pi,\pi'}\sum_{c\in\mathcal{C}}\delta_c \otimes q_c P_{\pi'} q_c^{-1} = \delta_{\mathcal{C},\mathcal{C}'}\delta_{\pi,\pi'}P_{\mathcal{C},\pi} \end{aligned}$$

where $C_G = C_G(r_\mathcal{C})$ and $c = q_c n q_c^{-1} d q_c n^{-1} q_c^{-1}$ iff $d = q_c n^{-1} q_c^{-1} c q_c n q_c^{-1} = q_c n^{-1} r_\mathcal{C} n q_c^{-1} = q_c r_\mathcal{C} q_c^{-1} = c$. Note that if $\mathcal{C} = \mathcal{C}'$, which is needed for $c = d$, then $q = q'$ are the same function and we can cancel $q_c {q'}_d^{-1}$ in this case. We also have

$$\sum_{\mathcal{C},\pi} P_{\mathcal{C},\pi} = \sum_{\mathcal{C}}\sum_{c\in\mathcal{C}} \delta_c \otimes q_c \left(\sum_\pi P_\pi\right) q_c^{-1} = \sum_\mathcal{C}\sum_{c\in\mathcal{C}}\delta_c\otimes 1 = \sum_\mathcal{C}\chi_\mathcal{C}\otimes 1 = 1\otimes 1$$

where we sum over irreps π of C_G for each $\mathcal{C}$. For centrality,

$$\begin{aligned}P_{\mathcal{C},\pi}(\delta_h\otimes g) &= \frac{\dim V_\pi}{|C_G|}\sum_{c\in\mathcal{C}}\sum_{n\in C_G}\delta_c\,\mathrm{Tr}_\pi(n^{-1})\delta_{c,q_cnq_c^{-1}hq_cn^{-1}q_c^{-1}}\otimes q_cnq_c^{-1}g\\ &= \frac{\dim V_\pi}{|C_G|}\delta_h\chi_{\mathcal{C}}(h)\otimes\sum_{n\in C_G}\mathrm{Tr}_\pi(n^{-1})q_hnq_h^{-1}g=\chi_{\mathcal{C}}(h)\delta_h\otimes q_hP_\pi q_h^{-1}g\\ (\delta_h\otimes g)P_{\mathcal{C},\pi} &= (\delta_h\otimes g)\sum_{c\in\mathcal{C}}\delta_c\otimes P_\pi q_c^{-1}=\sum_c\delta_h\delta_{gcg^{-1},h}\otimes gq_cP_\pi q_c^{-1}=\chi_{\mathcal{C}}(h)\delta_h\otimes gq_{g^{-1}dg}P_\pi q_{g^{-1}dg}^{-1}\end{aligned}$$

where for the second equality $c=q_cnq_c^{-1}hq_cn^{-1}q_c^{-1}$ iff $c=h$ by the same calculation as above. But $q_h^{-1}gq_{g^{-1}hg}r_{\mathcal{C}}q_{g^{-1}hg}^{-1}g^{-1}q_h=q_h^{-1}gg^{-1}hgg^{-1}q_h=r_{\mathcal{C}}$ so $q_h^{-1}gq_{g^{-1}hg}\in C_G$ and therefore commutes with P_π. ∎

The origin of these projection operators is the Peter-Weyl decomposition which applies to group algebras and other semisimple Hopf algebras including $D(G)$. We look at the group algebra case first in some detail. Thus, for $\mathbb{C}G$, there is an isomorphism $\mathbb{C}G\cong\oplus_\pi$ $\mathrm{End}(V_\pi)$ where the map to each component is to send $g\mapsto\pi(g)_{ij}e_i\otimes f^j$ where e_i is a basis of V_π and f^j is a dual basis. Here, $e_i\otimes f^j$ is the elementary matrix with 1 at the i,j row/column if we identify $\mathrm{End}(V_\pi)=M_{\dim(V_\pi)}(\mathbb{C})$. We check conventions: if $v=v^ie_i$ then $\pi(g)v=v^i\pi_{kj}e_k\langle f^j,e_i\rangle=e_k\pi_{ki}v^i$ so that $\pi(g)$ acts by matrix multiplication on (v^i) as a column vector. In the converse direction we define

$$\Phi_{\mathbb{C}G}:\oplus_\pi\mathrm{End}(V_\pi)\to\mathbb{C}G,\quad\Phi(e_i\otimes f^j)=\frac{\dim V_\pi}{|G|}\sum_{g\in G}\pi(g^{-1})_{ji}g$$

which we see obeys $\Phi(e_i\otimes f^i)=P_\pi$. One can check that the map Φ is an isomorphism of bimodules where $\mathbb{C}G$ acts on itself from the left and the right and acts on $\mathrm{End}(V_\pi)=V_\pi\otimes V_\pi^*$ on the left by π and on the right by its adjoint. Here $h\triangleright e_i=e_k\pi(h)_{ki}$ and $f^j\triangleleft h=\pi(h)_{jk}f^k$ (the dual basis elements transform the same way as vectors) and $\Phi_{\mathbb{C}G}$ is necessarily surjective as the image of $\sum_\pi\sum_i e_i\otimes f^i=\sum P_\pi=1$, given that is is a bimodule map. Moreover, under π', the element $\Phi(e_i\otimes f^j)$ maps to

$$\frac{\dim V_\pi}{|G|}\sum_{g\in G}\pi(g^{-1})_{ji}\pi'(g)_{kl}e_k\otimes f^l=\delta_{\pi,\pi'}e_i\otimes f^j\tag{4.11}$$

as required for the inverse in one direction, which proves that $\Phi_{\mathbb{C}G}$ is injective. The equality (4.11) used here is equivalent to a stronger version of the orthogonality relations for matrix entries of unitary irreducible representations over $\mathbb{C}$, of which (4.7) is a consequence. This also implies that $\pi'(P_\pi)=\mathrm{id}\delta_{\pi,\pi'}$ and hence that

$$P_\pi\triangleright e_i=e_k\pi(P_\pi)_{ki}=e_i,\quad f^j\triangleleft P_\pi=\pi(P_\pi)_{jk}f^k=f^j\tag{4.12}$$

if $e_i\in V_\pi$ and $f_j\in V_\pi^*$ respectively, or zero if these are in one of the other components. By the equivariance, these actions are equivalent to the projectors P_π acting by left or right multiplication, hence $P_\pi\mathbb{C}G=(\mathbb{C}G)P_\pi\cong\mathrm{End}(V_\pi)$ via Φ.

We now similarly let $D(G)$ act on $\mathrm{End}(V_{\mathcal{C},\pi})=V_{\mathcal{C},\pi}\otimes V_{\mathcal{C},\pi}^*$ from the left and right by the given left representation and its adjoint as a right one. It also acts on itself by left and right multiplication.

Theorem 4.2.5. Taking a basis $\{c\otimes e_i\}$ of the $D(G)$ representation $V_{\mathcal{C},\pi}$, with dual basis $\{\delta_d\otimes f^j\}$, the map $\Phi:\oplus_{\mathcal{C},\pi}\mathrm{End}(V_{\mathcal{C},\pi})\to D(G)$ given on $\mathrm{End}(V_{\mathcal{C},\pi})$ by

$$\Phi(c\otimes e_i\otimes\delta_d\otimes f^j)=\delta_c\otimes q_c\Phi_{\mathbb{C}C_G}(e_i\otimes f^j)q_d^{-1}=\frac{\dim V_\pi}{|C_G|}\sum_{n\in C_G}\pi(n^{-1})_{ji}\delta_c\otimes q_cnq_d^{-1}$$

is an isomorphism of bimodules.

Proof. Using the action (4.4) of $D(G)$ on $V_{\mathcal{C},\pi}$ in basis terms

$$(\delta_h\otimes g)\triangleright(c\otimes e_i)=\delta_{h,gcg^{-1}}gcg^{-1}\otimes\pi(q^{-1}_{gcg^{-1}}gq_c)_{ki}e_k,\tag{4.13}$$

the left module property of Φ is

$$\begin{aligned}\Phi((\delta_h\otimes g)\triangleright(c\otimes e_i)\otimes\delta_d\otimes f^j)&=\Phi(\delta_{h,gcg^{-1}}gcg^{-1}\otimes\pi(q^{-1}_{gcg^{-1}}gq_c)_{ki}e_k\otimes\delta_d\otimes f^j)\\&=\delta_{h,gcg^{-1}}\frac{\dim V_\pi}{|C_G|}\sum_{n\in C_G}\pi(n^{-1})_{jk}\pi(q^{-1}_{gcg^{-1}}gq_c)_{ki}\delta_{gcg^{-1}}\otimes q_{gcg^{-1}}nq_d^{-1}\\&=\delta_{h,gcg^{-1}}\frac{\dim V_\pi}{|C_G|}\sum_{n'\in C_G}\pi(n'^{-1})_{ki}\delta_{gcg^{-1}}\otimes gq_cn'q_d^{-1}\\&=(\delta_h\otimes g)\Phi(c\otimes e_i\otimes\delta_d\otimes f^j)\end{aligned}$$

where $n'=q_c^{-1}g^{-1}q_{gcg^{-1}}n$. We check that $n'r_{\mathcal{C}}n'^{-1}=q_c^{-1}g^{-1}q_{gcg^{-1}}r_{\mathcal{C}}q^{-1}_{gcg^{-1}}gq_c=q_c^{-1}g^{-1}(gcg^{-1})gq_c=r_{\mathcal{C}}$ so $n'\in C_G$ in our change of variables. For the other side we first use

$$\begin{aligned}\langle(\delta_d\otimes f^j)\triangleleft(\delta_h\otimes g),c\otimes e_i\rangle&:=\langle\delta_d\otimes f^j,(\delta_h\otimes g)\triangleright(c\otimes e_i)\rangle\\&=\langle\delta_g\otimes f^j,\delta_{h,gcg^{-1}}gcg^{-1}\otimes\pi(q^{-1}_{gcg^{-1}}gq_c)_{ki}e_k\rangle=\delta_{d,h}\delta_{d,gcg^{-1}}\pi(q^{-1}_{gcg^{-1}}gq_c)_{ji}\\&=\delta_{d,h}\langle\delta_{g^{-1}dg}\otimes\pi(q_d^{-1}gq_{g^{-1}dg})_{jk}f^k,c\otimes e_i\rangle\end{aligned}$$

for all c, e_i, from which we find the dual action

$$(\delta_d\otimes f^j)\triangleleft(\delta_h\otimes g)=\delta_{h,d}\delta_{g^{-1}dg}\otimes\pi(q_d^{-1}gq_{g^{-1}dg})_{jk}f^k\tag{4.14}$$

We then proceed similarly for the right module property

$$\begin{aligned}\Phi(c\otimes e_i\otimes(\delta_d\otimes f^j)\triangleleft(\delta_h\otimes g))&=\Phi(c\otimes e_i\otimes\delta_{h,d}\delta_{g^{-1}dg}\otimes\pi(q_d^{-1}gq_{g^{-1}dg})_{jk}f^k)\\&=\delta_{h,d}\frac{\dim V_\pi}{|C_G|}\sum_{n\in C_G}\delta_c\otimes\pi(q_d^{-1}gq_{g^{-1}dg})_{jk}\pi(n^{-1})_{ji}q_cnq^{-1}_{g^{-1}dg}\\&=\delta_{h,d}\frac{\dim V_\pi}{|C_G|}\sum_{n'\in C_G}\delta_c\otimes\pi(n'^{-1})_{ji}q_cn'q_d^{-1}g\\&=\Phi(c\otimes e_i\otimes\delta_d\otimes f^j)(\delta_h\otimes g)\end{aligned}$$

where $n'=nq^{-1}_{g^{-1}dg}g^{-1}q_d$ and one can check that this is in C_G. For the last line to identify the product in $D(G)$, we need for any $n\in C_G$ that $q_cnq_d^{-1}hq_dn^{-1}q_c^{-1}=c$ if and only if $h=d$.

We now check that Φ is inverse to the composite of the representations $(\mathcal{C},\pi)$ as maps $D(G)\to\mathrm{End}(V_{\mathcal{C},\pi})$. It is already surjective as it is a bimodule map and $\Phi(\sum_{\mathcal{C},\pi}\sum_{c,i}c\otimes e_i\otimes\delta_c\otimes f^i)=$

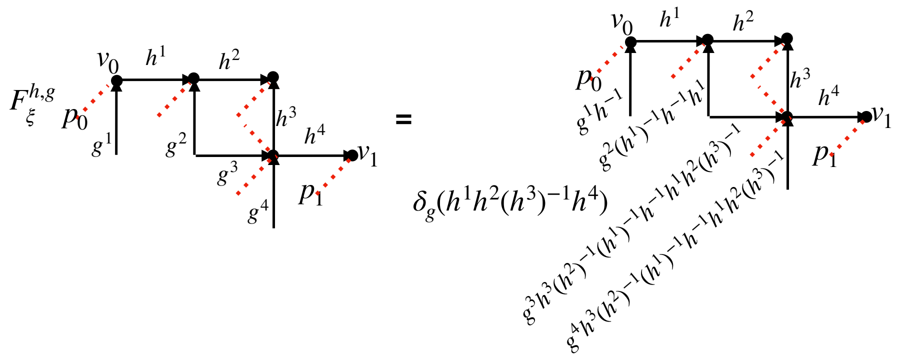


Figure 4.1: Example of a ribbon operator for a ribbon ξ starting at $s_0 = (v_0, p_0)$ to $s_1 = (v_1, p_1)$.

$\sum_{\mathcal{C},\pi} P_{\mathcal{C},\pi} = 1 \in D(G)$. Therefore it suffices to check that applying the representation (4.13) undoes Φ. Focussing on the block $\mathrm{End}(V_{\mathcal{C},\pi})$ and acting with its image on $c' \otimes e_{i'} \in V_{\mathcal{C}',\pi'}$,

$$\begin{aligned}
\Phi(c \otimes e_i \otimes \delta_d \otimes f^j) \triangleright (c' \otimes e_{i'}) &= \frac{\dim V_\pi}{|C_G|} \sum_{n \in C_G} \pi(n^{-1})_{ji} (\delta_c \otimes q_c n q_d^{-1}) \triangleright (c' \otimes e_{i'}) \\
&= \frac{\dim V_\pi}{|C_G|} \sum_{n \in C_G} \pi(n^{-1})_{ji} \delta_{c, q_c n q_d^{-1} c' q_d n^{-1} q_c^{-1}} c \otimes \pi'(q_c^{-1} q_c n q_d^{-1} q_{c'})_{j'i'} e_{j'} \\
&= \delta_{\mathcal{C},\mathcal{C}'} \delta_{d,c'} c \otimes \frac{\dim V_\pi}{|C_G|} \sum_{n \in C_G} \pi(n^{-1})_{ji} \pi'(n)_{j'i'} e_{j'} \\
&= \delta_{\mathcal{C},\mathcal{C}'} \delta_{\pi,\pi'} \delta_{d,c'} \delta_{j,i'} c \otimes e_i
\end{aligned}$$

as required. Here, $c = q_c n q_d^{-1} c' q_d n^{-1} q_c^{-1}$ iff $n^{-1} r_{\mathcal{C}} = q_d^{-1} c' q_d n^{-1}$ which is iff $q_d r_{\mathcal{C}} q_d^{-1} = c'$ which is iff $c = c'$. This is zero unless $\mathcal{C} = \mathcal{C}'$ also. We then used the full orthogonality (4.11) for the group C_G.

By general arguments as in the group case, it follows that $P_{\mathcal{C},\pi}$ acts as the identity on $V_{\mathcal{C},\pi}$ and $V^*_{\mathcal{C},\pi}$ (and zero on other components). One can also check this explicitly, for example,

$$\begin{aligned}
P_{\mathcal{C},\pi} \triangleright (c \otimes e_i) &= \frac{\dim V_\pi}{|C_G|} \sum_{d \in \mathcal{C}} \sum_{n \in C_G} \mathrm{Tr}_\pi(n^{-1}) \delta_{d, q_d n q_d^{-1} c q_d n^{-1} q_d^{-1}} d \otimes e_j \pi(\zeta_c(q_d n q_d^{-1})_{ji} \\
&= \frac{\dim V_\pi}{|C_G|} \sum_{n \in C_G} \mathrm{Tr}_\pi(n^{-1}) c \otimes e_j \pi(n)_{ji} = c \otimes e_i
\end{aligned}$$

since $d = q_d n q_d^{-1} c q_d n^{-1} q_d^{-1}$ iff $c = q_d n q_d^{-1} d q_d n^{-1} q_d^{-1} = q_d n r_c n^{-1} q_d^{-1} = q_d r_{\mathcal{C}} q_d^{-1} = d$. We used the strong orthogonality relations. Likewise for $f^v \triangleleft P_{\mathcal{C},\pi} = f^v$. ■

We see that, while Φ clearly sends the identity element or 'maximally entangled state' of $V_{\mathcal{C},\pi} \otimes V^*_{\mathcal{C},\pi}$ to $P_{\mathcal{C},\pi}$, it also implies a basis of all of $D(G)$ broken down into irreps $(\mathcal{C}, \pi)$ and elements $\Phi(c \otimes e_i \otimes \delta_d \otimes f^j)$ for each block. We will need this result for the discussion of ribbon teleportation.

4.2.3 $D(G)$ ribbon operators

To discuss the physics further, one needs the notion of a ribbon operator. By definition, a ribbon ξ is a strip of face width that connects two sites $s_0 = (v_0, p_0)$ to $s_1 = (v_1, p_1)$ by a sequence of sites (shown dashed) as for example in Figure 4.1. We call a ribbon *closed* if its endpoints are at the same site, and *open* if the endpoints are at disjoint sites with no intersection. Note that there exist ribbons which are neither open nor closed, which end at the same vertex but with different faces, say, but we are not concerned with this case. In Figure 4.1 we also show an associated ribbon operator $F_\xi^{h,g}$ acting on the spaces associated to the participating arrows and trivially elsewhere. The ribbon has an edge along which we transport h from the initial vertex by conjugation along the path, at each vertex of which we apply the conjugated h in the manner of a vertex operation but only to the cross arrow that comes anticlockwise from the dashed site marker. It follows that if we concatenate ribbon ξ' following on from ribbon ξ then we have the first of

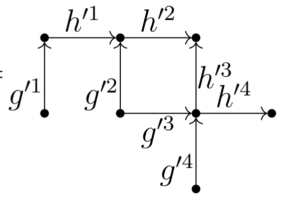
$$F_{\xi'\circ\xi}^{h,g} = \sum_{f\in G} F_{\xi'}^{f^{-1}hf, f^{-1}g} \circ F_\xi^{h,f}; \quad F_\xi^{h,g} \circ F_\xi^{h',g'} = \delta_{g,g'} F_\xi^{hh',g}, \tag{4.15}$$

where we see the coproduct $\Delta\delta_g$ of $\mathbb{C}(G)$. The latter implies the adjointness

$$(F_\xi^{h,g})^\dagger = F_\xi^{h^{-1},g} \tag{4.16}$$

with respect to the inner product of $\mathcal{H}$.

Example 4.2.6. Let the state on the l.h.s. of Figure 4.1 be $|\psi\rangle$, and take the inner product with another state:

$|\psi'\rangle :=$ 

$$\begin{aligned}\langle\psi'|(F_\xi^{h,g}|\psi\rangle) &= \delta_g(h^1h^2(h^3)^{-1}h^4)\delta_{h'^1}(h^1)\delta_{h'^2}(h^2)\delta_{h'^3}(h^3)\delta_{h'^4}(h^4)\delta_{g'^1}(g^1h^{-1})\\ &\quad \delta_{g'^2}(g^2(h^1)^{-1}h^{-1}h^1)\delta_{g'^3}(g^3h^3(h^2)^{-1}(h^1)^{-1}h^{-1}h^1h^2(h^3)^{-1})\\ &\quad \delta_{g'^4}(g^4h^3(h^2)^{-1}(h^1)^{-1}h^{-1}h^1h^2(h^3)^{-1})\\ &\quad = \delta_g(h'^1h'^2(h'^3)^{-1}h'^4)\delta_{h'^1}(h^1)\delta_{h'^2}(h^2)\delta_{h'^3}(h^3)\delta_{h'^4}(h^4)\delta_{g^1}(g'^1h)\\ &\quad \delta_{g^2}(g'^2(h^1)^{-1}hh^1)\delta_{g^3}(g'^3h^3(h^2)^{-1}(h^1)^{-1}hh^1h^2(h^3)^{-1})\\ &\quad \delta_{g^4}(g'^4h^3(h^2)^{-1}(h^1)^{-1}hh^1h^2(h^3)^{-1})\\ &\quad = (\langle\psi'|F_\xi^{h^{-1},g})|\psi\rangle\end{aligned}$$

and by (4.15), $(F_\xi^{h,g})^\dagger F_\xi^{h,g} = F_\xi^{h,g}(F_\xi^{h,g})^\dagger = F_\xi^{e,g}$.

Ribbon operators of the form $F_\xi^{e,g}$ produce only a scalar $\delta_g(\cdots)$ when applied to a lattice state. It is easy to see that

$$[F_\xi^{e,g}, F_{\xi'}^{e,g'}] = 0 \tag{4.17}$$

for all $g, g' \in G$ and ribbons ξ, ξ'.

Another important property of ribbon operators is that closed, contractible ribbons admit a trivial action of the corresponding ribbon operator on a vacuum state.

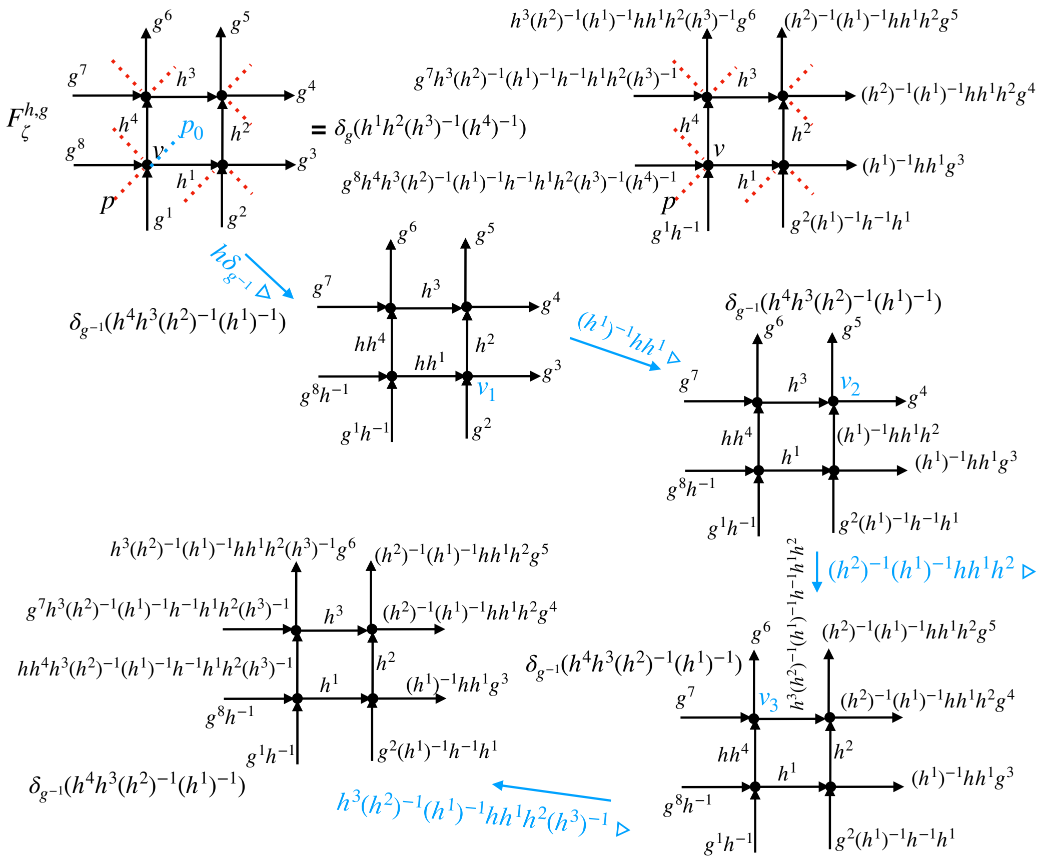


Figure 4.2: (a) Example of a circular ribbon starting at (v, p) and going anticlockwise, and (b) proof that this acts trivially on a vacuum state

Example 4.2.7. An example of a closed ribbon operator $F^{h,g}_\zeta$ from site (v, p) going anticlockwise back to itself is shown in Figure 4.2. We compare this with the following sequence of operations (i) $\delta_g\triangleright$ at site (v, p_0), (ii) $h\triangleright$ at v, (iii) $(h^1)^{-1}hh^1\triangleright$ at v_1 (iv) $(h^2)^{-1}(h^1)^{-1}hh^1h^2\triangleright$ at v_2, and (v) $h^3(h^2)^{-1}(h^1)^{-1}hh^1h^2(h^3)^{-1}\triangleright$ at v_3. The final results differ only on the initial arrows (h^4, g^8) where the ribbon sends these to $(h^4, g^8h^{-1}hg^{-1}h^{-1}g)$ (given the δ_g) while the sequence by contrast sends these to $(hg^{-1}h^{-1}gh^4, g^8h^{-1})$. Thus, the two act the same as long as the state they act on forces $\delta_{g,e}$. This is true for a vacuum state where $\delta_g\triangleright|\mathrm{vac}\rangle = \delta_g\Lambda^*\triangleright|\mathrm{vac}\rangle = \delta_{g,e}\Lambda^*|\mathrm{vac}\rangle = \delta_{g,e}|\mathrm{vac}\rangle$ and where $F^{h,g}_\zeta$ can be viewed as starting with $\delta_g\triangleright$, as does our sequence. We have $h\triangleright|\mathrm{vac}\rangle = h\Lambda\triangleright|\mathrm{vac}\rangle = \Lambda\triangleright|\mathrm{vac}\rangle = |\mathrm{vac}\rangle$ similarly, hence the action of the sequence (i)-(v) on the vacuum is $\delta_{g,e}|\mathrm{vac}\rangle$. We conclude that $F^{h,g}_\zeta|\mathrm{vac}\rangle = \delta_{g,e}|\mathrm{vac}\rangle$.

Lemma 4.2.8. Let ξ be a ribbon between sites $s_0 = (v_0, p_0)$ and $s_1 = (v_1, p_1)$. Then

$$[F^{h,g}_\xi, f\triangleright_v] = 0, \quad [F^{h,g}_\xi, \delta_e\triangleright_p] = 0,$$

for all $v \notin \{v_0, v_1\}$ and $p \notin \{p_0, p_1\}$.

$$f\triangleright_{s_0} \circ F^{h,g}_\xi = F^{fhf^{-1},fg}_\xi \circ f\triangleright_{s_0}, \quad \delta_f\triangleright_{s_0} \circ F^{h,g}_\xi = F^{h,g}_\xi \circ \delta_{h^{-1}f}\triangleright_{s_0},$$

$$f\triangleright_{s_1} \circ F^{h,g}_\xi = F^{h,gf^{-1}}_\xi \circ f\triangleright_{s_1}, \quad \delta_f\triangleright_{s_1} \circ F^{h,g}_\xi = F^{h,g}_\xi \circ \delta_{fg^{-1}hg}\triangleright_{s_1}$$

for all ribbons where s_0, s_1 are disjoint, i.e. when s_0 and s_1 share neither vertices or faces.

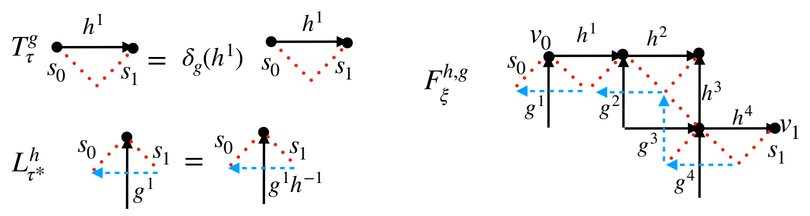
Proof. We refer to the example in Figure 4.1 to be concrete, but the arguments are general. (1) Commutation of the ribbon with $f\triangleright$ at sites across from the main path is automatic because the ribbon acts on the states on the cross arrows ($g^1, \dots, g^4$ in the example) like a vertex operator on the main path, which has an opposite relative orientation to a vertex at the other end of the relevant cross arrow. Hence the two actions are from opposite sides and commute. $f\triangleright$ between h^1, h^2 changes these to $h^1 f^{-1}, fh^2$ in the illustration which does not change the product when it comes to parts of a subsequent ribbon operator at later vertices. It also changes g^2 to $g^2 f^{-1}$. When we then apply the ribbon operator this changes to $g^2 f^{-1}(h^1 f^{-1})^{-1} h^{-1} h^1 f^{-1} = g^2 (h^1)^{-1} h^{-1} h^1 f^{-1}$ which is what we get if we apply the ribbon first and then $f\triangleright$ at this vertex. The same cancellation applies at other vertices on the main path other than the endpoints.

(2) The action of $\delta_f\triangleright$ at a face depends on the cyclic order determined by the vertex part of the site; commutation only holds in general if we chose this correctly or if we restrict to δ_e as stated (this disagrees with [BSW11]). For faces on the other side of the main path $\delta_f\triangleright$ has the form of $\delta_f(...h_i...)$ where the ... are states on arrows unaffected by the ribbon. The ribbon op does not change h_i so commutes with $\delta_f\triangleright$. The other relevant faces are those in the body of the ribbon itself and we look at all three in detail. (i) The face bounded by g^1, h^1, g^2 and an unkown x has $\delta_f\triangleright = \delta_f(g^1 h^1 (g^2)^{-1} x^{-1})$ in some cyclic order. But if we apply the ribbon first then $\delta_f\triangleright = \delta_f(g^1 h^{-1} h^1 (g^2 (h^1)^{-1} h^{-1} h^1)^{-1} x^{-1})$ in the same cyclic order, which we see is the same unless we started at g^2. (ii) The face bounded by g^2, h^2, h^3, g^3 has $\delta_f\triangleright = \delta_f(g^2 h^2 (h^3)^{-1} (g^3)^{-1})$ in some cyclic order. But if we apply the ribbon first then $\delta_f\triangleright = \delta_f(g^2 (h^1)^{-1} h^{-1} h^1 h^2 h^3 (g^3 h^3 (h^2)^{-1} (h^1)^{-1} h^{-1} h^1 h^2 (h^3)^{-1})^{-1})$ in the same order which again cancels (but only for the order shown). (iii) The face bounded by g^3, g^4 and unknowns x, y say has $\delta_f\triangleright = \delta_f(g^3 (g^4)^{-1} x^{-1} y)$, say, in some cyclic order. If we apply the ribbon first then g^3 is replaced by $g^3 w$ for a certain expression w but so is g^4, so $\delta_f\triangleright$ is the same as long as we do not start at g^4.

(3) We have four remaining cases and again we refer to Figure 4.1 to be concrete. (i) $f\triangleright$ at vertex v_0 sends (h^1, g^1) to $(fh^1, g^1 f^{-1})$ (the other two arrows are also changed but this commutes with the ribbon operation). Applying $F_\xi^{fhf^{-1}, fg}$ changes this to $(fh^1, g^1 f^{-1}(fhf^{-1})^{-1})$ with a factor $\delta_{fg}(fh^1 \cdots)$. If we apply $F_\xi^{h,g}$ first then we have $(h^1, g^1 h^{-1})$ and a factor $\delta_g(h^1 \cdots)$ and applying $f\triangleright$ turns the former to $(fh^1, g^1 h^{-1} f^{-1})$, which is the same. (ii) Similarly, $f\triangleright$ at vertex v_1 sends h^4 to $h^4 f^{-1}$ (the action on other, unmarked, arrows commutes with the ribbon operator). The ribbon operator $F_\xi^{h, gf^{-1}}$ then gives a factor $\delta_{gf^{-1}}(\cdots h^4 f^{-1})$. If we apply the ribbon first, we have $\delta_g(\cdots h^4)$ and then $f\triangleright$ gives the same as before. (iii) $\delta_{h^{-1}f}\triangleright$ at v_0 gives factor $\delta_{h^{-1}f}((g^1)^{-1}...)$ (for three unmarked arrows around the rest of the face) and the ribbon then gives a factor sends gives a factor $\delta_g(h^1 \cdots)$. It also acts on g^1. If we apply the ribbon first, then this gives $\delta_g(h^1 \cdots)$ and changes g^1 to $g^1 h^{-1}$. Then applying $\delta_f\triangleright$ gives a factor $\delta_f((g^1 h^{-1})^{-1} \cdots)$, which is the same. (iv) $\delta_{fg^{-1}hg}\triangleright$ at v_1 gives a factor $\delta_{fg^{-1}hg}(\cdots g^4 h^4)$ for two unmarked arrows at the start of the face). The ribbon then imposes $\delta_g(zh^4)$ where z is the product along the ribbon main path up to h^4 (in our case, $h^1 h^2 (h^3)^{-1}$). If we apply the ribbon first then g^4 gets changed to $g^4 z^{-1} h^{-1} z$ and then $\delta_f\triangleright$ gives $\delta_f(\cdots g^4 z^{-1} h^{-1} z h^4)$, which is the same, given the $\delta_g(zh^4)$ factor. ∎

This means that $F_\xi^{h,g}$ commutes with all terms of the Hamiltonian except those at s_0, s_1, where the nontrivial commutation relations will be used to create a quasiparticle at s_0 and its antiparticle at s_1. In this sense, a ribbon operator is a generalisation of the creation operators discussed in Section 4.1. We briefly consider so-called triangle operators, as they will be useful in future proofs.

Definition 4.2.9. The direct-triangle and dual-triangle operators T^g_τ and $L^h_{\tau^*}$ respectively are defined by



We also show how a ribbon can be built as a sequence of triangle operations.

Here the dual lattice inherits an orientation by anticlockwise rotation of the unique arrow that crosses a dual arrow. If the flow around either triangle is clockwise then h or g enters as shown, otherwise their opposite version much as before. For the dual triangle, this is the same as the arrow pointing inwards towards the vertex. Triangle operations can be viewed as atomic instances of ribbon operators, where the start and end are adjacent sites, namely the associated ribbons are $F^{h,g}_\tau = T^g_\tau$ and $F^{h,g}_{\tau^*} = \delta_{g,e}L^h_{\tau^*}$ respectively and convolving these via (4.15) gives the composite $F^{h,g}_\xi$. However, triangle operators are not open ribbons due to s_0, s_1 not being disjoint, so they have different commutation relations from those in Lemma 4.2.8, which we study in full later. It is clear that we have have algebras $\mathcal{A}_\tau := \mathrm{span}\{T^g_\tau \mid g \in G\} \cong \mathbb{C}(G)$ and $\mathcal{A}_{\tau^*} := \mathrm{span}\{L^h_{\tau^*} \mid h \in G\} \cong \mathbb{C}G$ in view of the composition rules

$$T^g_\tau \circ T^{g'}_\tau = \delta_{g,g'}T^g_\tau, \quad L^h_{\tau^*} \circ L^{h'}_{\tau^*} = L^{hh'}_{\tau^*}.$$

Proposition 4.2.10. Let $|\mathrm{vac}\rangle$ be a vacuum vector on a plane Σ. Let ξ be a ribbon between fixed sites $s_0 := (v_0, p_0), s_1 := (v_1, p_1)$ and

$$|\psi^{h,g}\rangle := F^{h,g}_\xi|\mathrm{vac}\rangle.$$

(1) $|\psi^{h,g}\rangle$ is independent of the choice of ribbon between fixed sites s_0, s_1.
(2) The space

$$\mathcal{L}(s_0, s_1) := \{|\psi\rangle \in \mathcal{H} \mid A(v)|\psi\rangle = B(p)|\psi\rangle = |\psi\rangle, \quad \forall v \notin \{v_0, v_1\}, p \notin \{p_0, p_1\}\}$$

is spanned by $\{|\psi^{h,g}\rangle \mid h, g \in G\}$.

(3) When sites s_0 and s_1 are disjoint, $\{|\psi^{h,g}\rangle \mid h, g \in G\}$ is an orthogonal basis of $\mathcal{L}(s_0, s_1)$. We call this the 'group basis' of $\mathcal{L}(s_0, s_1)$.

(4) $\mathcal{L}(s_0, s_1) \subset \mathcal{H}$ inherits actions at disjoint sites s_0, s_1,

$$f \triangleright_{s_0} |\psi^{h,g}\rangle = |\psi^{fhf^{-1},fg}\rangle, \quad \delta_f \triangleright_{s_0} |\psi^{h,g}\rangle = \delta_{f,h}|\psi^{h,g}\rangle$$

$$f \triangleright_{s_1} |\psi^{h,g}\rangle = |\psi^{h,gf^{-1}}\rangle, \quad \delta_f \triangleright_{s_1} |\psi^{h,g}\rangle = \delta_{f,g^{-1}h^{-1}g}|\psi^{h,g}\rangle$$

isomorphic to the left and right regular representation of $D(G)$ by $|\psi^{h,g}\rangle \mapsto \delta_h g$.

Proof. (1) Acting on the vacuum, a contractible, closed ribbon acts trivially as we have illustrated. Then if ξ, ξ' are two ribbons between the same sites, we regard the composite of the reverse of ξ' with ξ as a contractible, closed ribbon, as Σ is a plane. We then use equation (4.15).

(2) We leave this proof to Appendix 13, as it is lengthy and similar in some respects to [BM-D08].

(3) This proof can be found in [BSW11], but we include it to clarify that it applies only when s_0, s_1 are disjoint. Thus,

$$\langle\psi^{h,g}|\psi^{h',g'}\rangle = \langle\mathrm{vac}|(F_\xi^{h,g})^\dagger F_\xi^{h',g'}|\mathrm{vac}\rangle = \langle\mathrm{vac}|F_\xi^{h^{-1},g}F_\xi^{h',g'}|\mathrm{vac}\rangle = \delta_{g,g'}\langle\mathrm{vac}|F_\xi^{h^{-1}h',g}|\mathrm{vac}\rangle$$

and, if s_0, s_1 are disjoint,

$$\langle\mathrm{vac}|F_\xi^{h,g}|\mathrm{vac}\rangle = \langle\mathrm{vac}|(\delta_e{\triangleright}_{p_1})^\dagger F_\xi^{h,g}|\mathrm{vac}\rangle = \langle\mathrm{vac}|\delta_e{\triangleright}_{p_1}F_\xi^{h,g}|\mathrm{vac}\rangle = \langle\mathrm{vac}|F_\xi^{h,g}\delta_{g^{-1}hg}{\triangleright}_{p_1}|\mathrm{vac}\rangle$$

so that $\langle\mathrm{vac}|F_\xi^{h,g}|\mathrm{vac}\rangle = 0$ if $h \neq e$. When $h = e$,

$$\langle\mathrm{vac}|F_\xi^{e,g}|\mathrm{vac}\rangle = \langle\mathrm{vac}|(k{\triangleright}_{v_1})^\dagger F_\xi^{e,g}|\mathrm{vac}\rangle = \langle\mathrm{vac}|F_\xi^{e,gk}k^{-1}{\triangleright}|\mathrm{vac}\rangle = \langle\mathrm{vac}|F_\xi^{e,gk}|\mathrm{vac}\rangle$$

for every k, from which we deduce that $\langle\mathrm{vac}|F_\xi^{e,g}|\mathrm{vac}\rangle$ is independent of g. Since $\sum_{g\in G}F_\xi^{e,g} = \mathrm{id}$, it follows that $\langle\mathrm{vac}|F_\xi^{h,g}|\mathrm{vac}\rangle = \frac{\delta_{h,e}}{|G|}$ and hence that

$$\langle\psi^{h,g}|\psi^{h',g'}\rangle = \delta_{g,g'}\langle\mathrm{vac}|F_\xi^{h^{-1}h',g}|\mathrm{vac}\rangle \ = \frac{1}{|G|}\delta_{h,h'}\delta_{g,g'}.$$

Combined with (2), $\{|\psi^{h,g}\rangle \mid h, g \in G\}$ is then an orthogonal basis of $\mathcal{L}(s_0, s_1)$.

If s_0, s_1 are not disjoint then Lemma 4.2.8 no longer applies, and the commutation relations are different. For example, if s_0 and s_1 are joined by a direct triangle τ then $F_\tau^{h,g} = T_\tau^g$ so $\{|\psi^{h,g}\rangle \mid h, g \in G\}$ are no longer orthogonal.

(4) This follows from the commutation relations in Lemma 4.2.8 at s_0 and s_1 using $f{\triangleright}|\mathrm{vac}\rangle = |\mathrm{vac}\rangle$ and $\delta_f{\triangleright}|\mathrm{vac}\rangle = \delta_{f,e}|\mathrm{vac}\rangle$ replacing f as modified by the commutation relations. Making the identification with $D(G)$ we compare the s_0 action with the left regular representation $\delta_f{\triangleright}(\delta_h g) = \delta_f\delta_h g = \delta_{f,h}\delta_h g$ and $f{\triangleright}(\delta_h g) = f\delta_h g = \delta_{fhf^{-1}}fg$ using the $D(G)$ commutation relations. The right regular representation is made into a left action via the antipode, so $\delta_f{\triangleright}(\delta_h g) = \delta_h g\delta_{f^{-1}} = \delta_h\delta_{gf^{-1}g^{-1}}g = \delta_{f,g^{-1}h^{-1}g}g$ and $f{\triangleright}(\delta_h g) = \delta_h gf^{-1}$. These match the stated $D(G)$ actions at the end sites. ■

Remark 4.2.11. The above Proposition 4.2.10 is known in the literature, albeit in different forms, see [BSW11], and is included to be precise in our set up. It assumes that we begin with a vacuum state $|\mathrm{vac}\rangle$ on Σ. It is immediate, however, that the same arguments apply for a state $|\vartheta\rangle$ which is merely *locally* vacuum – that is, $B(p)|\vartheta\rangle = A(v)|\vartheta\rangle = |\vartheta\rangle$ for v, p at sites along the ribbon path and in the region between if we change the ribbon path. Thus, (1) now becomes more precisely that $|\vartheta^{h,g}\rangle := F_\xi^{h,g}|\vartheta\rangle$ is invariant under choice of ribbons ξ and ξ' between fixed sites s_0, s_1 iff the composite of ξ with reversed ξ' forms a closed, contractible ribbon ξ'', and where $A(v)|\vartheta\rangle = B(p)|\vartheta\rangle = |\vartheta\rangle$ for all p and v adjacent to ξ'' and in the region enclosed by ξ''. The intuition is that the ribbons may be smoothly deformed into one another, and thus leave the state invariant by previous arguments. The subspace $\mathcal{L}'(s_0, s_1)$ is then defined in the natural way, ignoring excitations outwith the local neighbourhood of consideration, and actions are inherited on the sites s_0, s_1 in the identical manner to $\mathcal{L}(s_0, s_1)$. This locality of the Hamiltonian H allows us to create quasiparticles at distance without being concerned about the compounding effects: they may be considered entirely separately. While we don't refer to it explicitly, this remark applies to the corollaries and applications throughout the Chapter in this context.

The last part of Proposition 4.2.10 implies a new basis of $\mathcal{L}(s_0, s_1)$ in terms of the quasiparticle content at the two ends.

Corollary 4.2.12. Let ξ be an open ribbon from s_0 to s_1. Then $\mathcal{L}(s_0,s_1)$ has an alternative 'quasiparticle basis' consisting for each irrep $\mathcal{C},\pi$ of $D(G)$ of the elements

$$|u,v;\mathcal{C},\pi\rangle=\frac{\dim V_\pi}{|C_G|}F'^{\mathcal{C},\pi;u,v}_\xi|\mathrm{vac}\rangle;\quad F'^{\mathcal{C},\pi;u,v}_\xi:=\sum_{n\in C_G}\pi(n^{-1})_{ji}F_\xi^{c,q_cnq_d^{-1}}$$

where $u=(c,i)$ and $v=(d,j)$ with $c,d\in\mathcal{C}$ and $i,j=1,\cdots\dim V_\pi$.

Proof. Here $|u,v;\mathcal{C},\pi\rangle=\tilde\Phi(e_u\otimes f^v)$ by which we mean $\Phi(e_u\otimes e^v)$ in Theorem 4.2.5, where $e_u=c\otimes e^i$ and $f^v=\delta_d\otimes f^j$ are basis elements of $V_{\mathcal{C},\pi}$ and $V^*_{\mathcal{C},\pi}$ respectively, then identified with an element of $\mathcal{L}(s_0,s_1)$ by the inverse of the last part of Proposition 4.2.10. ■

These states behave for the left site action $\triangleright_{s_0}$ on $\mathcal{L}(s_0,s_1)$ according to a quasiparticle state labelled by basis element e_u and for the right action at s_1 according to an anti-quasiparticle state labelled by the dual basis element f^v. Recall that we view the left site action $\triangleright_{s_1}$ as a right one via the antipode S of $D(G)$. The ribbon operators $F'^{\mathcal{C},\pi;u,v}_\xi$ that create these states from the vacuum are also of interest in their own right and it is claimed in [BM-D08] that they form a basis of the space of operators that commute with almost all $A(v)$ and $B(p)$ in the same way that $\mathcal{L}(s_0,s_1)$ is defined.

Corollary 4.2.13. If $|\psi\rangle\in\mathcal{L}(s_0,s_1)$ and we detect in it a quasiparticle of type $\mathcal{C},\pi$ at s_0 by nonzero projection $P_{\mathcal{C},\pi}\triangleright_{s_0}|\psi\rangle$ then

$$P_{\mathcal{C},\pi}\triangleright_{s_0}|\psi\rangle=|\psi\rangle\triangleleft_{s_1}P_{\mathcal{C},\pi},$$

hence we also automatically detect it at s_1, and vice-versa. In particular, the state

$$|\mathrm{Bell},\xi\rangle=\sum_{h\in G}F_\xi^{h,e}|\mathrm{vac}\rangle$$

has a nonzero projection $P_{\mathcal{C},\pi}\triangleright_{s_0}|\mathrm{Bell},\xi\rangle=|\mathrm{Bell},\xi\rangle\triangleleft_{s_1}P_{\mathcal{C},\pi}\neq0$ for all $\mathcal{C},\pi$.

Proof. This is essentially a block version of teleportation. A general state is highly entangled in a superposition between the different particle types,

$$|\psi\rangle=\sum_{\mathcal{C},\pi}\sum_{u,v}\phi(\mathcal{C},\pi,u,v)\tilde\Phi(e_u\otimes f^v)$$

where, as above, $\tilde\Phi:\mathrm{End}(V_{\mathcal{C},\pi})\to\mathcal{L}(s_0,s_1)$ denotes Φ combined with the inverse of the identification in Proposition 4.2.10. Here $\{e_u\}$ are a basis of $V_{\mathcal{C},\pi}$ and $\{f^u\}$ a dual basis. Applying $P_{\mathcal{C}',\pi'}\triangleright_{s_0}$ and $\triangleleft_{s_1}P_{\mathcal{C}',\pi'}$ becomes via the bimodule properties respectively $P_{\mathcal{C}',\pi'}\triangleright e_u$ and $f^v\triangleleft P_{\mathcal{C}',\pi'}$. But these projections are zero unless $(\mathcal{C}',\pi')=(\mathcal{C},\pi)$, in which case they act as the identity, as in the proof of Theorem 4.2.5. Hence

$$P_{\mathcal{C},\pi}\triangleright_{s_0}|\psi\rangle=\sum_{u,v}\phi(\mathcal{C},\pi,u,v)\tilde\Phi(e_u\otimes f^v)=|\psi\rangle\triangleleft_{s_1}P_{\mathcal{C},\pi}.$$

In particular,

$$P_{\mathcal{C}',\pi'}\triangleright_{s_0}|u,v;\mathcal{C},\pi,\xi\rangle=|u,v;\mathcal{C},\pi,\xi\rangle\triangleleft_{s_1}P_{\mathcal{C}',\pi'}=\delta_{\mathcal{C}',\mathcal{C}}\delta_{\pi',\pi}|u,v;\mathcal{C},\pi,\xi\rangle.\qquad(4.18)$$

For the 'block Bell state', we consider $1_{D(G)} = \sum_{h\in G} \delta_h \otimes e$ which map to $\sum_h |\psi^{h,e}\rangle$ according to the last part of Proposition 4.2.10. On the other hand, this is $\sum_{\mathcal{C},\pi} P_{\mathcal{C},\pi}$ by Lemma 4.2.4 and hence each term is the image under Φ of $\sum e_u \otimes f^u$ in Theorem 4.2.5. Thus,

$$|\mathrm{Bell};\xi\rangle = \sum_{\mathcal{C},\pi}\sum_u \tilde{\Phi}(e_u\otimes f^u) = \sum_{\mathcal{C},\pi}\sum_u |u,u;\mathcal{C},\pi\rangle$$

and from (4.18) we have

$$P_{\mathcal{C},\pi}{\triangleright_{s_0}}|\mathrm{Bell},\xi\rangle = |\mathrm{Bell};\mathcal{C},\pi,\xi\rangle = |\mathrm{Bell},\xi\rangle{\triangleleft_{s_1}}P_{\mathcal{C},\pi} \neq 0$$

where

$$|\mathrm{Bell};\mathcal{C},\pi,\xi\rangle = \sum_u \tilde{\Phi}(e_u\otimes f^u) = \sum_u |u,u;\mathcal{C},\pi\rangle$$

is the claimed nonzero state projected out from $|\mathrm{Bell};\xi\rangle$. ∎

We recall that in teleportation one has an entangled 'Bell state', $\sum_i |v_i\rangle \otimes\langle v_i|$ for a basis and dual basis of a Hilbert space, and if we apply from the left a projection $|v_1\rangle\langle v_1|$ say then the state collapses to $|v_1\rangle\otimes\langle v_1|$ so that the right factor is an eigenstate if we apply $|v_1\rangle\langle v_1|$ from the right. Equivalently, if we evaluate against any $\langle\psi|$ on the left then the result is $\langle\psi|$ in the right factor. We see a similar phenomenon with the block $V_{\mathcal{C},\pi}\otimes V^*_{\mathcal{C},\pi}$ in place of $|v_i\rangle\otimes\langle v_i|$. In the $D(\mathbb{Z}_n)$ case discussed below, each block will be 1-dimensional so that we are then a bit closer to the standard case.

We can also potentially look inside each block, i.e. for each fixed $\mathcal{C},\pi$, regard $|\mathrm{Bell};\mathcal{C},\pi,\xi\rangle \in \mathcal{L}(s_0,s_1)$ as a 'mini Bell state' that can similarly transport a single particle state across the ribbon. We saw in the proof above that this is $\sum_u \Phi(e_u\otimes f^u) = P_{\mathcal{C},\pi}$ mapped over to this space by the inverse of the identification in the last part of Proposition 4.2.10. We can also write

$$|\mathrm{Bell};\mathcal{C},\pi,\xi\rangle = \frac{\dim V_\pi}{|C_G|} W_\xi^{\mathcal{C},\pi}|\mathrm{vac}\rangle;\quad W_\xi^{\mathcal{C},\pi} := \sum_u F_\xi^{\prime\mathcal{C},\pi;u,u}. \tag{4.19}$$

so that the 'ribbon trace operator' $W_\xi^{\mathcal{C},\pi}$ has the physical interpretation of creating a maximally entangled quasiparticle/anti-quasiparticle pair (the mini Bell state) of only the specified type $\mathcal{C},\pi$. The issue for teleportation of a single quasiparticle state vector using a such mini Bell state would be how, in a quantum computer, to create a single particle state or its dual and evaluate it against e_u at s_0 or against f^u at s_1.

Lemma 4.2.14. Let $\xi : s_0 \to s_1$ and $\xi' : s_1 \to s_2$ be open ribbons. Then

$$F_{\xi'\circ\xi}^{\prime\mathcal{C},\pi;u,v} = \sum_w F_{\xi'}^{\prime\mathcal{C},\pi;w,v}\circ F_\xi^{\prime\mathcal{C},\pi;u,w}$$

Proof. We have using (4.15),

$$\begin{aligned}
F_{\xi'\circ\xi}^{\prime\mathcal{C},\pi;(c,i),(d,j)} &= \sum_{n\in C_G}\pi(n^{-1})_{ji}F_{\xi'\circ\xi}^{c,q_c n q_d^{-1}}\\
&= \sum_{f\in G}\sum_{n\in C_G}\pi(n^{-1})_{ji}F_{\xi'}^{f^{-1}cf,f^{-1}q_c n q_d^{-1}}\circ F_\xi^{c,f}\\
&= \sum_{b\in\mathcal{C}}\sum_k\sum_{m,n\in C_G}\pi((m^{-1}n)^{-1})_{jk}\pi(m^{-1})_{ki}F_{\xi'}^{b,q_b m^{-1} n q_d^{-1}}F_\xi^{c,q_c m q_d^{-1}}
\end{aligned}$$

where we uniquely factorised $f^{-1}q_c = q_b m^{-1}$ in terms of some $b \in \mathcal{C}$ and $m \in C_G$. We then change variables to $n' = m^{-1}n$ and recognise the answer with $w = (b, k)$. ■

This reflects that F'_ξ are a kind of (nonAbelian) Fourier transform of the original F_ξ with convolution as in (4.15) becoming multiplication. Invertibility of Fourier transform implies that the space spanned by such operators is the same as the space spanned by the original F_ξ, now organised according to the quasiparticle type. In addition, we have:

Lemma 4.2.15.

$$W_\xi^{e,\pi} \circ W_\xi^{e,\pi'} = W_\xi^{e,\pi\otimes\pi'}$$

Proof. Using (4.19), Corollary 4.2.12 and (4.15) in that order,

$$\begin{aligned}
W_\xi^{e,\pi} \circ W_\xi^{e,\pi'} &= \sum_{n,n'\in G} \mathrm{Tr}_\pi(n^{-1}) F_\xi^{e,n} \mathrm{Tr}_{\pi'}(n'^{-1}) F_\xi^{e,n'} \\
&= \sum_{n,n'\in G} \mathrm{Tr}_\pi(n^{-1}) \mathrm{Tr}_{\pi'}(n'^{-1}) \delta_{n,n'} F_\xi^{e,n} \\
&= \sum_{n} \mathrm{Tr}_{\pi\otimes\pi'}(n^{-1}) F_\xi^{e,n}
\end{aligned}$$

which we recognise as stated. ■

We will also need the following.

Lemma 4.2.16. Let $\xi : s_0 \to s_1$ be an open ribbon. Then $W_\xi^{\prime\mathcal{C},\pi\dagger} = W_\xi^{\prime\mathcal{C}^*,\pi^*}$ where π^* is the conjugate unitary representation of C_G and $\mathcal{C}^* = \mathcal{C}^{-1}$ equipped with $r_{\mathcal{C}^*} = r_{\mathcal{C}}^{-1}$ and $q : \mathcal{C}^{-1} \to G$ given by $q_{c^{-1}} = q_c$ for all $c \in \mathcal{C}$.

Proof. Here

$$F_\xi^{\prime\mathcal{C},\pi;(c,i),(d,j)\dagger} = \sum_{n\in C_G} \overline{\pi(n^{-1})_{ji}} F_\xi^{c^{-1},q_c n q_d^{-1}} = \sum_{n\in C_G} \pi^*(n^{-1})_{ij} F_\xi^{c^{-1},q_c n q_d^{-1}} = F_\xi^{\prime\mathcal{C}^{-1},\pi^*;(c^{-1},j),(d^{-1},i)}$$

where $\mathcal{C}^*$ is the $\mathcal{C}^{-1}$ with the basepoint and q function data as stated and the same C_G. We now take the trace by summing over $c = d$ and $i = j$. ■

Note that it could be that $\mathcal{C} = \mathcal{C}^{-1}$ as a set but is not $\mathcal{C}^*$ due to a different base point (this happens for the order two conjugacy class of S_3).

The last ingredient we need for applications is a generalisation of the space $\mathcal{L}(s_0, s_1)$. If $s_0, s_1, \cdots, s_n$ are $n + 1$ sites, define the subspace

$$\mathcal{L}(s_0, s_1, \cdots, s_n) := \{|\psi\rangle \in \mathcal{H} \mid A(v)|\psi\rangle = B(p)|\psi\rangle = |\psi\rangle, \forall v \notin \{v_0, v_1, \cdots, v_n\}, p \notin \{p_0, p_1, \cdots, p_n\}\}.$$

Lemma 4.2.17. Given a $D(G)$ model on a borderless planar lattice Σ, let $s_0, s_1, \cdots, s_n$ be $n + 1$ disjoint sites. Then

$$\dim(\mathcal{L}(s_0, s_1, \cdots, s_n)) = |G|^{2n}$$

with an orthogonal basis

$$\{|\psi^{\{h^1,h^2,\cdots,h^n\},\{g^1,g^2,\cdots,g^n\}}\rangle \mid h^1, h^2, \cdots, h^n, g^1, g^2, \cdots, g^n \in G\}$$

generalising the group basis of $\mathcal{L}(s_0, s_1)$. There is another orthogonal basis that is the equivalent generalisation of the quasiparticle basis.

Proof. As we saw in the proof of Proposition 4.2.10, the only operations which take the vacuum to states with excitations only at any two sites, say s_0, s_1, are ribbon operators along a ribbon $\xi : s_0 \to s_1$. Now, let A be a complete graph, with sites $s_0, s_1, \cdots, s_n$ as vertices. We then have a contribution to $\dim(\mathcal{L}(s_0, s_1, \cdots, s_n))$ of $|G|^2 = \dim(\mathcal{L}(s_0, s_1))$ from an edge in A between s_0, s_1, corresponding to some ribbon $\xi : s_0 \to s_1$; we can give this the group basis with labels $\{h^1, g^1 \mid h^1, g^1 \in G\}$ or the equivalent quasiparticle basis. Edges in A between other vertices/sites contribute similarly, so for example there are another $|G|^2$ orthogonal ribbon operators along the ribbon $\xi' : s_1 \to s_2$, which multiplies with the initial $|G|^2$ from ξ. However, by Lemma 4.2.8, if we have already counted the operators along ribbons $\xi : s_0 \to s_1$ and $\xi' : s_1 \to s_2$ then any ribbon operator $F^{h,g}_{\xi''}$ for $\xi'' : s_0 \to s_2$ has a decomposition into ribbons along ξ and ξ' iff ξ'' is isotopic to $\xi' \circ \xi$. Therefore the only edges which contribute are between vertices which have no alternative path along previously visited edges. In particular, we define T as any maximally spanning tree on A. Then $\dim(\mathcal{L}(s_0, s_1, \cdots, s_n))$ receives contributions from exactly the n edges in T, and we may for example give the group basis with labels $\{h^i, g^i \mid h^i, g^i \in G\}$ from each edge. ■

We note that while the dimensions multiply, it is not true that $\mathcal{L}(s_0, s_1, \cdots, s_n)$ can be presented as $\mathcal{L}(s_0, s_1) \otimes \cdots \otimes \mathcal{L}(s_{n-1}, s_1)$ where the tensor product is along each edge in T. This is because, for example, ribbon operators $F^{h,g}_{\xi}$ and $F^{h',g'}_{\xi'}$ meet at the endpoint s_1 and need not commute. On the other hand, if we have some disjoint subsets of $\{s_0, \cdots, s_{n-1}\}$ then the tensor product of the logical spaces associated to each subset form a subspace. For example

$$\mathcal{L}(s_0, s_1) \otimes \mathcal{L}(s_2, s_3) \subset \mathcal{L}(s_0, s_1, s_2, s_3)$$

by sending $F^{h,g}_{\xi}|\text{vac}\rangle \otimes F^{h',g'}_{\xi'}|\text{vac}\rangle \mapsto F^{h,g}_{\xi} \circ F^{h',g'}_{\xi'}|\text{vac}\rangle$.

4.2.4 Reduction to Abelian model for $G = \mathbb{Z}_n$

In this section, we verify that everything above reduces correctly to the Abelian case already covered in Section 4.1 via the Fourier correspondence (5.1). Here $D(\mathbb{Z}_n) = \mathbb{C}(\mathbb{Z}_n) \otimes \mathbb{C}\mathbb{Z}_n \cong \mathbb{C}.\mathbb{Z}_n \times \mathbb{Z}_n = \mathbb{C}[g,h]/\langle g^n - 1, h^n - 1\rangle$ and we recall that we set $q = e^{\frac{2\pi\imath}{n}}$. Clearly, at a face

$$g \triangleright = \sum_m q^m \delta_m \triangleright = \sum_m q^m \delta_{m,i+j-k-l} = q^{i+j-k-l}$$

if the state around the face is $|i\rangle, |j\rangle, |k\rangle, |l\rangle$ with orientations as displayed before. This no longer depends on the starting point. Moreover $h\triangleright$ around a vertex is the action of $1 \in \mathbb{Z}_n$ so acts as before. This clearly gives gives $A(v)$ as before and $B(p) = \delta_0 \triangleright = \frac{1}{n}\sum_k g^k \triangleright$ as before.

The vacuum degeneracy of the Abelian model is straightforward to calculate.

Lemma 4.2.18. Let Σ be a closed, orientable surface, and let $G = \mathbb{Z}_n$. Then

$$\dim(\mathcal{H}_{vac}) = n^{2k},$$

where k is the genus of Σ.

Proof. The fundamental group $\pi_1(\Sigma) \cong \mathbb{Z}^{2k}$. $\mathbb{Z}^{2k}$ is a $2k$-biproduct of $\mathbb{Z}$ in the category of groups, so $\text{Hom}(\mathbb{Z}^{2k}, \mathbb{Z}_n) \cong \text{Hom}(\mathbb{Z}, \mathbb{Z}_n)^{2k}$. Now, $|\text{Hom}(\mathbb{Z}, \mathbb{Z}_n)| = n$, so $|\text{Hom}(\mathbb{Z}^{2k}, \mathbb{Z}_n)| = n^{2k}$. The G-action is trivial, so we are done. ■

For representations, the conjugacy classes are singletons $\{i\}$, say, with isotropy group all of $\mathbb{Z}_n$, with irrep π_j say. The carrier space is 1-dimensional and the irrep is

$$g{\triangleright}\{i\} = \sum_m q^m \delta_{m,i}\{i\} = q^i\{i\}, \quad h{\triangleright}\{i\} = q^j\{i\}$$

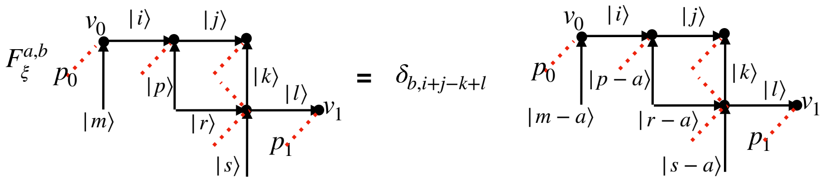
as employed before. Projectors simplify to those from Section 4.1. Thus,

$$P_j = \frac{\dim V_\pi}{|G|}\sum_g (\mathrm{Tr}_j\, g^{-1})g = \frac{1}{n}\sum_k q^{-jk}g^k$$

$$P_{\{i\},j} = \delta_i \otimes P_j = \delta_i \otimes \frac{1}{n}\sum_{l\in G} q^{-jl}g^l \cong P_i \otimes P_j = \frac{1}{n^2}\sum_{k,l} q^{-(ik+jl)}h^k g^l = P_{ij}$$

by Fourier isomorphism between $\mathbb{C}\mathbb{Z}_n$ and $\mathbb{C}(\mathbb{Z}_n)$.

The ribbon operators are now labelled as $F_\xi^{a,b}$ say, where $a, b \in \mathbb{Z}_n$ and have a simpler form. For example,



for the ribbon in Figure 4.1. Concatenation of ribbon operators simplifies to:

$$F_{\xi'\circ\xi}^{a,b} = \sum_{f\in G} F_{\xi'}^{a,b-f} \circ F_\xi^{a,f}. \tag{4.20}$$

The commutation relations in Lemma 4.2.8 simplify to

$$h{\triangleright}_{s_0} \circ F_\xi^{a,b} = F_\xi^{a,b+1} \circ h{\triangleright}_{s_0}, \quad g{\triangleright}_{s_0} F_\xi^{a,b} = q^a F_\xi^{a,b} g{\triangleright}_{s_0}$$

$$h{\triangleright}_{s_1} \circ F_\xi^{a,b} = F_\xi^{a,b-1} \circ h{\triangleright}_{s_0}, \quad g{\triangleright}_{s_0} F_\xi^{a,b} = q^{-a} F_\xi^{a,b} g{\triangleright}_{s_0}$$

i.e. these 'q-commute'. For example,

$$g{\triangleright}_{s_0} \circ F_\xi^{a,b} = \sum_m q^m \delta_m {\triangleright}_{s_0} \circ F_\xi^{a,b} = \sum_m q^m F^{a,b} \delta_{m-a} {\triangleright}_{s_0} = \frac{1}{n}\sum_{m,k} q^m F_\xi^{a,b} q^{-(m-q)k} g^k {\triangleright}_{s_0}.$$

The sum over m forces $k = 1$ which then gives the answer stated. Consequently, on states $|\psi^{a,b}\rangle = F_\xi^{a,b}|\mathrm{vac}\rangle$ we just have that

$$h{\triangleright}_{s_0}|\psi^{a,b}\rangle = |\psi^{a,b+1}\rangle, \quad g{\triangleright}_{s_0}|\psi^{a,b}\rangle = q^a|\psi^{a,b}\rangle$$

which commute, and similarly at s_1.

For the quasiparticle basis, the relevant ribbon operator and its adjoint are

$$F_\xi^{\prime i,j} := \sum_k q^{-jk} F_\xi^{i,k}, \quad F_\xi^{\prime i,j\dagger} = F_\xi^{\prime -i,-j}$$

where we omit u, v as these are trivial and the i, j play the role of $\mathcal{C}, \pi$ respectively in the construction of P_{ij} above. We see that F'_ξ is just a Fourier transform in the second argument, which takes convolution to multiplication so that (4.20) becomes

$$F_{\xi'\circ\xi}^{\prime i,j} = \sum_k q^{-j(k-l)-jl} \sum_l F_{\xi'}^{i,k-l} \circ F_\xi^{i,l} = F_{\xi'}^{\prime i,j} \circ F_\xi^{\prime i,j}. \tag{4.21}$$

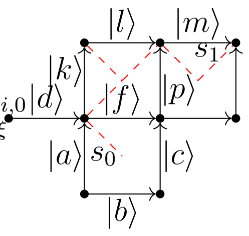
Also, since there are no u, v indices, $W_\xi^{i,j} = F_\xi^{\prime i,j}$. The following three subsections show how the above might be used in practice to perform operations relevant to quantum computation.

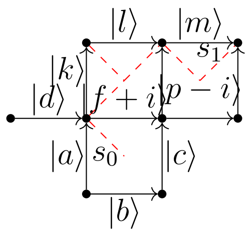
Abelian Bell state and teleportation

According to our general theory and our calculations above, the state which is maximally entangled between the particle types is

$$|\mathrm{Bell};\xi\rangle=\sum_{i,j}\frac{1}{n}F'^{i,j}_{\xi}|\mathrm{vac}\rangle=\frac{1}{n}\sum_{i,j,k}q^{-jk}|\psi^{i,k}\rangle=\sum_{i}F^{i,0}_{\xi}|\mathrm{vac}\rangle=\sum_{i}|\psi^{i,0}\rangle,$$

where the second-to-last step is the Fourier transform. Here $\langle\mathrm{Bell};\xi|\mathrm{Bell};\xi\rangle=n$. For a concrete example, consider

$|\mathrm{Bell};\xi\rangle=\sum_i F^{i,0}_{\xi}$ 

$=\delta_0(k+l+m)\sum_i$ 

for a ribbon $\xi:s_0\to s_1$, where $s_0=(v_0,p_0)$, $s_1=(v_1,p_1)$ and a generic term in $|\mathrm{vac}\rangle$ with relevant arrow values $|a\rangle\otimes\cdots\otimes|p\rangle$ as shown. We also know that

$$|i,j\rangle:=|\mathrm{Bell};i,j,\xi\rangle=P_{ij}{\triangleright}_{s_0}|\mathrm{Bell};\xi\rangle=|\mathrm{Bell};\xi\rangle{\triangleleft}_{s_1}P_{ij}=\frac{1}{n}W^{i,j}_{\xi}|\mathrm{vac}\rangle$$

are the 'mini-Bell' states associated to each i,j. In our case (as there are no u,v indices) these have no internal substructure as an entangled sum of internal states and we just regard them as a basis of $\mathcal{L}(s_0,s_1)$ as we vary i,j. To illustrate how this goes explicitly, consider $P_{ij}=\frac{1}{n^2}\sum_{x,y}q^{-ix-jy}g^xh^y$ as above, acting at s_0, say. Then, renaming the dummy index i in $|\mathrm{Bell};\xi\rangle$ as z,

$$\begin{aligned}P_{ij}{\triangleright}_{s_0}|\mathrm{Bell};\xi\rangle&=\frac{1}{n^2}\sum_{x,y,z}q^{-ix-jy}\delta_{0,k+l+m}q^{x(f+z+y-c-b+a-y)}\\&\qquad|f+z+y\rangle\otimes|k+y\rangle\otimes|d-y\rangle\otimes|a-y\rangle\otimes|p-z\rangle\\&=\frac{1}{n}\sum_{y}q^{-jy}\delta_{0,k+l+m}|i+c+b-a+y\rangle\otimes|k+y\rangle\otimes|d-y\rangle\otimes|a-y\rangle\otimes|p-i+f-c-b+a\rangle\\&=\frac{1}{n}\sum_{y}q^{-jy}\delta_{0,k+l+m}|i+f+y\rangle\otimes|k+y\rangle\otimes|d-y\rangle\otimes|a-y\rangle\otimes|p-i\rangle\\&=\frac{1}{n}\sum_{y}q^{-jy}\delta_{y,k+l+m}|f+i\rangle\otimes|k\rangle\otimes|d\rangle\otimes|a\rangle\otimes|p-i\rangle\\&=\frac{1}{n}\sum_{y}q^{-jy}F^{i,y}_{\xi}|\mathrm{vac}\rangle=\frac{1}{n}W^{i,j}_{\xi}|\mathrm{vac}\rangle\end{aligned}$$

for the affected arrows located in line with the previous diagram. The sum over x forced the value $z=i-f+c+b-a$ for the second equality. We then used that $\delta_0{\triangleright}_{s_0}$ acts as the identity on the vacuum so that $f+c+b-a=0$ around the face p_0 for the third equality. Likewise, we use that $h^{-y}{\triangleright}_{s_0}$ is the identity on the vacuum for the fourth (this shifts the original values f,k,d,a around the vertex v_0 by $\mp y$). We then recognise the action of $W^{i,j}_{\xi}$ as expected. Similarly for ${\triangleleft}_{s_1}P_{ij}$.

We now explain our teleportation point of view in this Abelian case. Here $\{|i,j\rangle\}$ are a basis of $\mathcal{L}(s_0,s_1)$ and we have seen that $P_{ij}(s_0)$ applied to $|\mathrm{Bell};\xi\rangle$ collapses the ribbon state to $|i,j\rangle$, and a quasi-particle of type (i,j) now occupies s_0. This is a local operation at s_0 but the resulting state when measured at $P_{-i,-j}(s_1) = \triangleleft_{s_1} P_{ij}$ is also an eigenstate and detects a quasiparticle of type $(-i,-j)$ locally at s_1. Here the right action of P_{ij} is the left action of $S(P_{ij})$, where S is the antipode of the group algebra of $\mathbb{Z}_n \times \mathbb{Z}_n$. Although the details are not the same as usual quantum teleportation, we follow the same principle of using a maximally entangled state to transfer information along the length of an extended object, our case the ribbon. We can use this to transmit any state vector in the vector space spanned by the particle types, i.e. a vector $\vec{\psi} = (\psi_{ij})$. We set up a state $|\mathrm{Bell};\xi\rangle$ and apply the operator $\sum_{i,j} \psi_{ij} P_{ij}(s_0)$ to it locally at s_0. This results in

$$|\psi\rangle := \sum_{i,j} \psi_{ij} P_{ij}(s_0)|\mathrm{Bell};\xi\rangle = \sum_{i,j} \psi_{ij}|i,j\rangle \in \mathcal{L}(s_0,s_1)$$

as a ribbon state that encodes our vector $\vec{\psi}$. We can then read off the latter at the other end by applying the operator $P_{-i,-j}(s_1)$ locally at s_1, where

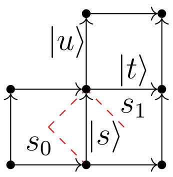
$$P_{-i,-j}(s_1)|\psi\rangle = |\psi\rangle \triangleleft_{s_1} P_{ij} = P_{ij} \triangleright_{s_0} |\psi\rangle = \sum_{k,l} \psi_{kl} P_{ij}(s_0)|k,l\rangle = \psi_{ij} P_{i,j}(s_0)|\mathrm{Bell};\xi\rangle$$

This is the component of $|\psi\rangle$ that contains the ψ_{ij} coefficient. It is also equal to $\psi_{ij} P_{-i,-j}(s_1)|\mathrm{Bell};\xi\rangle$ making it clear that we can extract the coefficient by local operations at s_1.

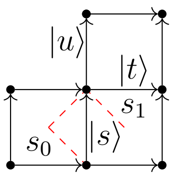
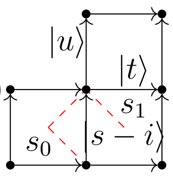
While such a teleportation scheme is possible when the projectors P_{ij} can be applied to the lattice, in reality such projectors can only be applied probabilistically, by performing measurements. In particular, assuming that application of projectors can be performed deterministically in general has grave complexity-theoretic consequences, such as allowing NP-complete problems to always be solved in polynomial time on a quantum computer [Aar05]. This means that, while the above scheme is illustrative of the entanglement between sites s_0 and s_1, it is unclear how to leverage this property to be computationally useful when the superposition may only be collapsed by a measurement.

Quasiparticle creation and transportation redux

Next, we show how to create and transport quasiparticles using the $W_\xi^{i,j}$ operators, which are equal to $F_\xi^{\prime i,j}$ in the $\mathbb{Z}_n$ case, and relate this to the *ad hoc* description of Section 4.1.1. This pertains to the following lattice in the vacuum state:

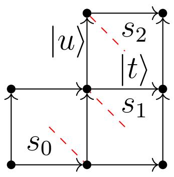


with $\xi : s_0 \to s_1$ as shown. We apply

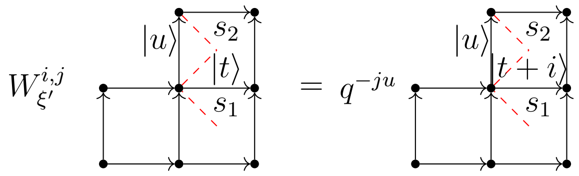
$W_\xi^{i,j}$  $= \sum_k q^{-jk}\delta_k(s)$ 

We see that only $|s\rangle$ is affected and $\sum_k q^{-jk}\delta_k(s) = q^{-js}$. In terms of our X, Z operations, we have $W^{i,j}_\xi|s\rangle = X^{-i}Z^{-j}|s\rangle$. Recall from Section 4.1.1 that the effect of this is that particles $\pi_{i,j}$ and $\pi_{-i,-j}$ appear at sites s_0 and s_1 respectively, which we tested using projectors. In other words, we have the mini-Bell state $|\mathrm{Bell}; i, j, \xi\rangle$.

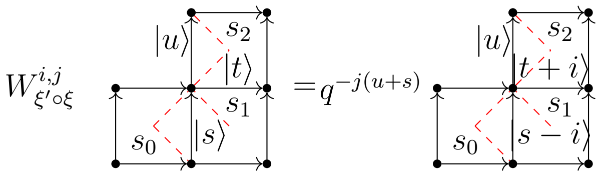
Next, we consider a further site s_2



then the further effect of a operator $W^{i,j}_{\xi'}$ for an open ribbon $\xi' : s_1 \to s_2$ is



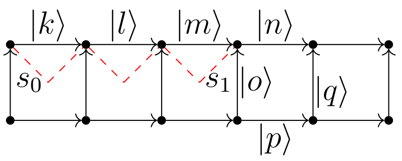
We see that $W^{i,j}_{\xi'} : |t\rangle \otimes |u\rangle \mapsto X^i|t\rangle \otimes Z^{-j}|u\rangle$ while leaving the other states unchanged. We saw in Section 4.1.1 that quasiparticles $\pi_{i,j}$ and $\pi_{-i,-j}$ now occupy sites s_0 and s_2 respectively. So the effect of this second ribbon operator is to transport the $\pi_{-i,-j}$ excitation from s_1 to s_2. We also know from (4.21) that these two ribbon operations compose to $W^{i,j}_{\xi'\circ\xi}$ along the composite ribbon, so we create the state $|\mathrm{Bell}; i, j, \xi' \circ \xi\rangle$. In other words, creation at sites s_0 and s_1 followed by transport from s_1 to s_2 is equal to creation at sites s_0 and s_2. The combined operation is



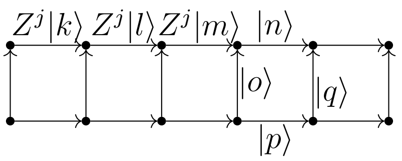
which we see affects only the states $|s\rangle \otimes |t\rangle \otimes |u\rangle$ along the ribbon and has the particle content at the ends as previously analysed.

Quasiparticle braiding

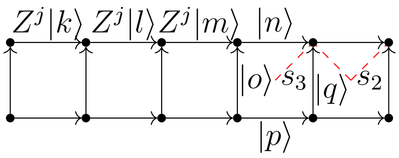
This section gives an example of braiding on the lattice, and relates it to the braiding of irreducible representations of $D(G)$ given at the start of Section 4.1. We do not prove explicitly that all such lattice braidings correspond to braids in the representation category, but the broad arguments are easy to see. Let $\xi : s_0 \to s_1$ be the following ribbon acting on a vacuum state $|\mathrm{vac}\rangle$,



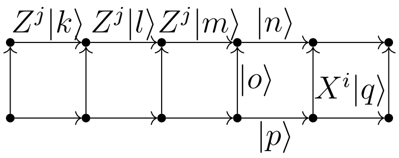
where we have labelled the relevant edges $|k\rangle$ to $|q\rangle$ as shown. $W^{0,-j}_\xi$ creates quasiparticles e_{-j}, e_j at s_0, s_1, and takes the vacuum to



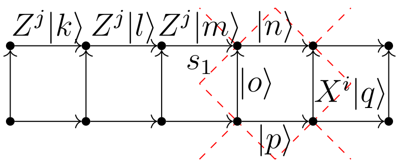
Also consider another ribbon operator $W_{\xi'}^{-i,0}$ for $\xi' : s_2 \to s_3$, creating m_{-i}, m_i quasiparticles at s_2, s_3 according to



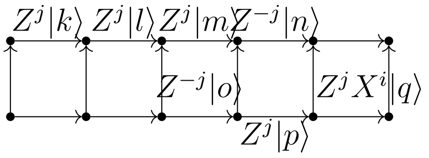
The combined effect of these is the state $|\psi\rangle := W_{\xi'}^{-i,0} \otimes W_{\xi}^{0,-j}|\text{vac}\rangle$



Now let ξ'' be a ribbon rotating anti-clockwise from s_1 back to s_1 around the face of s_3, according to



Acting on $|\psi\rangle$, we move the e_j quasiparticle at s_1 around the m_i at s_3 using $W_{\xi''}^{0,-j}$ and resulting in



Now use $Z^jX^i = q^{ij}X^iZ^j$ on $|q\rangle$ so that the $Z^{\pm j}$ operators that make up $W_{\xi''}^{0,-j}$ act on $|\text{vac}\rangle$. But the latter is a face operator $g^{-j}\triangleright$ around the face of s_3 and acts trivially on the vacuum. Hence the effect of $W_{\xi''}^{0,-j}$ on $|\psi\rangle$ is to send

$$|\psi\rangle \mapsto q^{ij}|\psi\rangle$$

as expected for the braiding of m_i with e_j. To visualise this braiding, we should think in terms of worldlines to take account of the temporal aspect: we first create the quasiparticles, and then transport one around the other. We identify the map above with

$$\Psi_{m_i,e_j} \circ \Psi_{e_j,m_i} : e_j \otimes m_i \to e_j \otimes m_i$$

and have braided the worldline of the e_j quasiparticle around that of the m_i quasiparticle and back again.

While this is only one instance of braiding, any ribbon operator on the plane which forms a closed loop around another occupied site will admit a similar braiding, as the same argument from above applies but taking a product of vertex and face operators, rather than just $g^{-j}\triangleright$ in this example. We assert that the state at the occupied site will always admit commutation relations such that the appropriate phase factor is produced.

4.2.5 Details for $D(S_3)$ and applications

S_3 is the smallest nonAbelian group. We let S_3 be generated by $u = (12)$, $v = (23)$ with relations $u^2 = v^2 = e$ and $uvu = vuv$ $(= w = (13))$. This has three irreducible representations:

$$1, \quad \sigma = \text{sign}, \quad \tau; \quad \sigma\otimes\sigma = 1, \quad \sigma\otimes\tau = \tau\otimes\sigma = \tau, \quad \tau\otimes\tau = 1\oplus\sigma\oplus\tau$$

where τ is the only 2-dimensional one and sign $= -1$ on u, v, w and $+1$ otherwise. The irreps of $D(S_3)$ are given by pairs $(\mathcal{C}, \pi)$, where $\mathcal{C}$ is a conjugacy class in S_3 and π is an irrep of the centraliser of a distinguished element $r_{\mathcal{C}}$ in $\mathcal{C}$, i.e. an isotropy subgroup, and we also need to fix q_c for each $c \in \mathcal{C}$ such that $c = q_c r_{\mathcal{C}} q_c^{-1}$. We take these as follows:

1. The trivial conjugacy class $\mathcal{C} = \{e\}$, $r_{\mathcal{C}} = q_c = e$ and $C_G = S_3$, giving exactly 3 chargeons $(\{e\}, 1)$, $(\{e\}, \sigma)$ and $(\{e\}, \tau)$ as $D(S_3)$ irreps.
2. $\mathcal{C} = \{u, v, w\}$, $r_{\mathcal{C}} = u$, $q_u = e, q_v = w, q_w = v$ and $C_G = \mathbb{Z}_2 = \{e, u\}$, giving $(\{u, v, w\}, 1)$ and $(\{u, v, w\}, -1)$ as 2 irreps of $D(S_3)$, where we indicate the representation $\pi_{-1}(u) = -1$ of C_G.
3. $\mathcal{C} = \{uv, vu\}$, with $r_{\mathcal{C}} = uv$, $q_{uv} = e$, $q_{vu} = v$ and $C_G = \mathbb{Z}_3 = \{e, uv, vu\}$, giving $(\{uv, vu\}, 1)$, $(\{uv, vu\}, \omega)$, $(\{uv, vu\}, \omega^*)$ as 3 irreps of $D(S_3)$, where $\omega = e^{\frac{2\pi\imath}{3}}$ and we indicate irreps $\pi_\omega(uv) = \omega$ and $\pi_{\omega^*}(uv) = \omega^{-1}$ of C_G.

Thus, there are 8 irreps of $D(S_3)$. To describe the projectors, we denote the conjugacy class $\mathcal{C}$ by its chosen element $r_{\mathcal{C}}$ as shorthand, for example $P_{u,\pi} := P_{\{u,v,w\},\pi}$. The chargeons have projectors

$$P_1 = \frac{1}{6}\sum_g g, \quad P_\sigma = \frac{1}{6}(e - u - v - w + uv + vu), \quad P_\tau = \frac{1}{6}(2e - uv - vu)$$

in $\mathbb{C}S_3$ with actual $D(S_3)$ projectors $P_{e,1} = \delta_e \otimes P_1, P_{e,\sigma} = \delta_e \otimes P_\sigma, P_{e,\tau} = \delta_e \otimes P_\tau$. The fluxion projectors are

$$\begin{aligned} P_{u,1} &= \sum_{c\in\mathcal{C}} \delta_c \otimes q_c \Lambda_{C_G} q_c^{-1} = \frac{1}{2}(\delta_u \otimes (e+u) + \delta_v \otimes (e+v) + \delta_w \otimes (e+w)) \\ P_{uv,1} &= \frac{1}{3}(\delta_{uv} + \delta_{vu})(e + uv + vu) \end{aligned}$$

along with $P_{e,1}$ from before which can be viewed as either. The remaining projectors after a short computation are

$$\begin{aligned} P_{u,-1} &= \frac{1}{2}(\delta_u \otimes (e-u) + \delta_v \otimes (e-v) + \delta_w \otimes (e-w)) \\ P_{uv,\omega} &= \frac{1}{3}(\delta_{uv} \otimes (e + \omega^{-1}uv + \omega vu) + \delta_{vu} \otimes (e + \omega uv + \omega^{-1}vu)) \\ P_{uv,\omega^{-1}} &= \frac{1}{3}(\delta_{uv} \otimes (e + \omega uv + \omega^{-1}vu) + \delta_{vu} \otimes (e + \omega^{-1}uv + \omega vu)) \end{aligned}$$

On a lattice Σ where each edge has an associated state in $\mathbb{C}S_3$, $\mathcal{L}(s_0, s_1)$ has the quasiparticle basis $|u, v; \mathcal{C}, \pi\rangle$ from Corollary 4.2.12, where unlike the $\mathbb{Z}_n$ case $u = (c, i), v = (d, j)$ can have different i, j as not all irreps are 1-dimensional. To avoid a clash with group elements of S_3, we will refer to the pairs $(c,i),(d,j)$ directly. We again refer to $\mathcal{C}$ by its representative. Then in our case, the ribbon operators required to create these bases from vacuum for each chargeon are

$$F'^{e,1}_\xi=\sum_{n\in S_3}F^{e,n}_\xi=\mathrm{id},\quad F'^{e,\sigma}_\xi=\sum_{n\in S_3}\mathrm{sign}(n)F^{e,n}_\xi,\quad F'^{e,\tau;i,j}_\xi=\sum_{n\in S_3}\tau(n^{-1})_{ji}F^{e,n}_\xi$$

The last of these is the only case with i,j indices as the other π are 1-dimensional. Similarly, for fluxions:

$$F'^{u,1;c,d}_\xi=F^{c,q_cq_d^{-1}}_\xi+F^{c,q_cuq_d^{-1}}_\xi,\quad F'^{uv,1;c,d}_\xi=F^{c,q_cq_d^{-1}}_\xi+F^{c,q_cuvq_d^{-1}}_\xi+F^{c,q_cvuq_d^{-1}}_\xi$$

where in the first case have indices $c,d\in\{u,v,w\}$ and in the second case $c,d\in\{uv,vu\}$. The remaining quasiparticle basis operators are

$$\begin{aligned}F'^{u,-1;c,d}_\xi&=F^{c,q_cq_d^{-1}}_\xi-F^{c,q_cuq_d^{-1}}_\xi\\F'^{uv,\omega;c,d}_\xi&=F^{c,q_cq_d^{-1}}_\xi+\omega^{-1}F^{c,q_cuvq_d^{-1}}_\xi+\omega F^{c,q_cvuq_d^{-1}}_\xi\\F'^{uv,\omega^*;c,d}_\xi&=F^{c,q_cq_d^{-1}}_\xi+\omega F^{c,q_cuvq_d^{-1}}_\xi+\omega^{-1}F^{c,q_cvuq_d^{-1}}_\xi\end{aligned}$$

with corresponding indices as before.

We will mainly need the traces $W^{\mathcal{C},\pi}_\xi$ of these defined in (4.19). Up to normalisation, these are just the $P_{\mathcal{C},\pi}$ already computed but converted to ribbon operators according to the last part of Proposition 4.2.10. For chargeons these come out as

$$W^{e,1}_\xi=\mathrm{id},\quad W^{e,\sigma}_\xi=\sum_{n\in S_3}\mathrm{sign}(n)F^{e,n}_\xi,\quad W^{e,\tau}_\xi=\sum_j F'^{e,\tau;j,j}_\xi=2F^{e,e}_\xi-F^{e,uv}_\xi-F^{e,vu}_\xi.$$

For fluxions we have

$$W^{u,1}_\xi=\sum_{c\in\{u,v,w\}}F^{c,e}_\xi+F^{c,c}_\xi,\quad W^{uv,1}_\xi=\sum_{c\in\{uv,vu\}}F^{c,e}_\xi+F^{c,uv}_\xi+F^{c,vu}_\xi$$

and the other ones are

$$\begin{aligned}W^{u,-1}_\xi&=\sum_{c\in\{u,v,w\}}F^{c,e}_\xi-F^{c,c}_\xi\\W^{uv,\omega}_\xi&=F^{uv,e}_\xi+F^{vu,e}_\xi+\omega(F^{uv,vu}_\xi+F^{vu,uv}_\xi)+\omega^{-1}(F^{uv,uv}_\xi+F^{vu,vu}_\xi)\\W^{uv,\omega^*}_\xi&=F^{uv,e}_\xi+F^{vu,e}_\xi+\omega(F^{uv,uv}_\xi+F^{vu,vu}_\xi)+\omega^{-1}(F^{uv,vu}_\xi+F^{vu,uv}_\xi)\end{aligned}$$

Note that the $\mathcal{C}=\{uv,vu\}$ class is self-inverse but its elements are not self-inverse, so $\mathcal{C}^*$ is the same class $\mathcal{C}$ but with $r_{\mathcal{C}^*}=vu$ and $q^*_{uv}=q_{vu}=v,q^*_{vu}=q_{uv}=e$. Hence Lemma 4.2.16 says that

$$W^{uv,\omega\dagger}_\xi=W^{vu,\omega^*}_\xi=\sum_n\pi_{\omega^*}(n^{-1})(F^{uv,vnv^{-1}}+F^{vu,n})=W^{uv,\omega}_\xi$$

so this works out as self-adjoint (as one can also check directly). Similarly for W^{uv,ω^*}_ξ, and more obviously for the other cases.

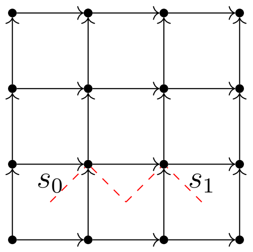
The maximally entangled state is then

$$\begin{aligned}|\mathrm{Bell};\xi\rangle &= \Big(\frac{1}{6}(W_\xi^{e,1} + W_\xi^{e,\sigma} + 2W_\xi^{e,\tau}) + \frac{1}{2}(W_\xi^{u,1} + W_\xi^{u,-1}) \\ &\qquad + \frac{1}{3}(W_\xi^{uv,1} + W_\xi^{uv,\omega} + W_\xi^{uv,\omega^*})\Big)|\mathrm{vac}\rangle \\ &= \sum_{h\in S_3} F_\xi^{h,e}|\mathrm{vac}\rangle\end{aligned}$$

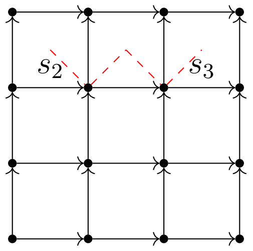
as required by Corollary 4.2.13, the first expression being as a sum of 8 mini Bell states.

Protected qubit system using S_3 ribbons

Here, we provide a concrete construction of a protected logical qubit within the $D(S_3)$ Kitaev model, elaborating on ideas in [WLP09]. Let Σ be a lattice in the vacuum state. Let ξ be a ribbon between sites $s_0 := (v_0, p_0)$ and $s_1 := (v_1, p_1)$.

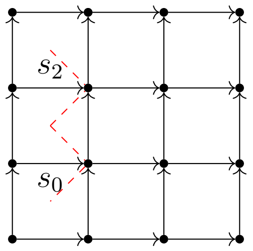


This particular choice of ribbon and sites is just for illustrative purposes; any open ribbon will do. We focus initially on the chargeon sector with $W_\xi^\tau := W_\xi^{e,\tau}$. If we apply this to the vacuum the lattice is now occupied by quasiparticles π and π^* at sites s_0 and s_1. Next, let $\xi' : s_2 \to s_3$ be another ribbon and apply the ribbon operator $W_{\xi'}^\tau$



We call this state $|0_L\rangle := W_{\xi'}^\tau \circ W_\xi^\tau|\mathrm{vac}\rangle$ for reasons which will become clear. $|0_L\rangle \in \mathcal{L}(s_0, s_1, s_2, s_3)$, and now τ quasiparticles occupy the lattice at sites s_0, s_1, s_2, s_3, which is obvious as $P_{\tau\triangleright s_i} W_{\xi'}^\tau \circ W_\xi^\tau|\mathrm{vac}\rangle = W_{\xi'}^\tau \circ W_\xi^\tau|\mathrm{vac}\rangle$ for all i.

Next, let $\xi'' : s_0 \to s_2$ connect across as



and apply the ribbon operator $W_{\xi''}^\sigma$ to $|0_L\rangle$, defining $|1_L\rangle := W_{\xi''}^\sigma|0_L\rangle$. We claim that $1_L\rangle$

still has only τ excitations at s_0, s_1, s_2, s_3. We check this by expanding P_τ and $W^\sigma_{\xi''}$:

$$\begin{aligned}
P_\tau\triangleright_{s_0} W^\sigma_{\xi''} &= \frac{1}{6}(2e\triangleright_{s_0} - uv\triangleright_{s_0} - vu\triangleright_{s_0})(F^{e,e}_{\xi''} - F^{e,u}_{\xi''} - F^{e,v}_{\xi''} - F^{e,w}_{\xi''} + F^{e,uv}_{\xi''} + F^{e,vu}_{\xi''}) \\
&= \frac{1}{6}(2(F^{e,e}_{\xi''} - F^{e,u}_{\xi''} - F^{e,v}_{\xi''} - F^{e,w}_{\xi''} + F^{e,uv}_{\xi''} + F^{e,vu}_{\xi''})e\triangleright_{s_0} \\
&- (F^{e,vu}_{\xi''} - F^{e,w}_{\xi''} - F^{e,u}_{\xi''} - F^{e,v}_{\xi''} + F^{e,e}_{\xi''} + F^{e,vu}_{\xi''})uv\triangleright_{s_0} \\
&- (F^{e,uv}_{\xi''} - F^{e,v}_{\xi''} - F^{e,w}_{\xi''} - F^{e,u}_{\xi''} + F^{e,vu}_{\xi''} + F^{e,e}_{\xi''})vu\triangleright_{s_0}) \\
&= (F^{e,e}_{\xi''} - F^{e,u}_{\xi''} - F^{e,v}_{\xi''} - F^{e,w}_{\xi''} + F^{e,uv}_{\xi''} + F^{e,vu}_{\xi''})\frac{1}{6}(2e\triangleright_{s_0} - uv\triangleright_{s_0} - vu\triangleright_{s_0}) \\
&= W^\sigma_{\xi''} \circ P_\tau\triangleright_{s_0}
\end{aligned}$$

by Lemma 4.2.8. Therefore

$$\begin{aligned}
P_\tau\triangleright_{s_0}|1_L\rangle &= P_\tau\triangleright_{s_0} W^\sigma_{\xi''} \circ W^\tau_{\xi'} \circ W^\tau_{\xi}|\mathrm{vac}\rangle \\
&= W^\sigma_{\xi''} \circ P_\tau\triangleright_{s_0} W^\tau_{\xi'} \circ W^\tau_{\xi}|\mathrm{vac}\rangle \\
&= W^\sigma_{\xi''} \circ P_\tau\triangleright_{s_0}|0_L\rangle = W^\sigma_{\xi''}|0_L\rangle = |1_L\rangle
\end{aligned}$$

and an identical calculation applies at s_1.

The states $|0_L\rangle$ and $|1_L\rangle$ are therefore indistinguishable by local projectors, as the orthogonality of projectors shown in Lemma 4.2.4 means that for all $P_{\mathcal{C},\pi}$, $P_{\mathcal{C},\pi}\triangleright_{s_i}|0_L\rangle = P_{\mathcal{C},\pi}\triangleright_{s_i}|1_L\rangle = 0$, $\forall s_i$ iff $\mathcal{C}, \pi \neq e, \tau$, and $P_{e,\tau}\triangleright_{s_i}|0_L\rangle = |0_L\rangle$, $P_{e,\tau}\triangleright_{s_i}|1_L\rangle = |1_L\rangle$. A physical explanation is that the σ quasiparticles generated by $W^\sigma_{\xi''}$ at sites s_0, s_2 'fuse' with the extant τ quasiparticles, as we have $\sigma \otimes \tau = \tau$. Now, $|0_L\rangle$ and $|1_L\rangle$ are orthogonal since

$$\begin{aligned}
\langle 0_L|1_L\rangle &= \langle \mathrm{vac}|W^{\tau\dagger}_{\xi} \circ W^{\tau\dagger}_{\xi'} \circ W^\sigma_{\xi''} \circ W^\tau_{\xi'} \circ W^\tau_{\xi}|\mathrm{vac}\rangle \\
&= \langle \mathrm{vac}|W^\sigma_{\xi''} \circ W^\tau_{\xi} \circ W^\tau_{\xi'} \circ W^\tau_{\xi'} \circ W^\tau_{\xi}|\mathrm{vac}\rangle \\
&= \langle \mathrm{vac}|W^\sigma_{\xi''} \circ W^{\tau\otimes\tau}_{\xi'} \circ W^{\tau\otimes\tau}_{\xi}|\mathrm{vac}\rangle
\end{aligned}$$

by (4.17), Lemma 4.2.15 and Lemma 4.2.16. By the arguments of Lemma 4.2.17, $W^{\tau\otimes\tau}_{\xi'} W^{\tau\otimes\tau}_{\xi}|\mathrm{vac}\rangle$ has no support in $\mathcal{L}(s_0, s_2)$, while $W^\sigma_{\xi''}|\mathrm{vac}\rangle$ has no support in $\mathcal{L}(s_0, s_1)$ or $\mathcal{L}(s_2, s_3)$, and so

$$\langle 0_L|1_L\rangle = 0.$$

Thus, $\mathcal{H}_L := \mathrm{span}(\{|0_L\rangle, |1_L\rangle\})$ is a 2-dimensional subspace of $\mathcal{L}(s_0, s_1, s_2, s_3)$ that is degenerate under H. We call $\mathcal{H}_L$ a *logical qubit* on the lattice. By similar arguments as for the vacuum in Section 4.2.1, any state in $\mathcal{H}_L$ is 'topologically protected'; local errors leave the state unaffected. In this case, the two types of errors which are undetectable and affect $\mathcal{H}_L$ are (a) loops enclosing at least one occupied site and (b) ribbon operators extending between occupied sites. Therefore, quasiparticles should be placed at distant locations to minimise errors.

We then identify $W^\sigma_{\xi''}$ with X_L, the logical X gate, which is justified as

$$W^\sigma_{\xi''} \circ W^\sigma_{\xi''} = W^{\sigma\otimes\sigma}_{\xi''} = W^1_{\xi''} = \mathrm{id}$$

by Lemma 4.2.15. Therefore, X_L is involutive as desired for any implementation of a qubit computation within the model, for example by ZX-calculus based on $\mathbb{C}\mathbb{Z}_2$ as a quasiFrobenius algebra. Clearly, we can obtain any X_L basis rotation on the logical qubit by exponentiation. In [WLP09] it is argued that we can in fact acquire universal quantum computation by an implementation of the logical Hadamard, entangling gates and measurements. For completeness, we outline some aspects of these further steps in Appendix 14.

4.3 Aspects of general $D(H)$ models

The Kitaev model is known to generalise with $\mathbb{C}G$ replaced by any finite-dimensional Hopf algebra H with antipode S obeying $S^2 = \mathrm{id}$ (which over $\mathbb{C}$ or another field of characteristic zero is equivalent to H semisimple or cosemisimple). Although less well studied, that one can obtain topological invariants as a version of the Turaev-Viro invariant was shown in [BK12, Meu17]. That one has an action of the Drinfeld quantum double $D(H)$ [Dri87] at each site is more immediate and was first noted in [BMCA13]. We just replace the group action $g\triangleright$ by $h\triangleright$ acting in the tensor product representation with factors in order going around the vertex as in Figure 4.3, which now depends on where p is located. We likewise replace the action of δ_g by $a\triangleright$ for $a \in H^*$ and likewise just take the tensor product action around the face in the order depending on where v is located. We use the Hopf algebra regular and coregular representations

$$h\triangleright g = hg \quad \text{or} \quad h\triangleright g = gSh; \quad a\triangleright g = a(g_1)g_2 \quad \text{or} \quad a\triangleright g = a(Sg_2)g_1 \tag{4.22}$$

with the first choice if the arrow is outbound for the vertex /in the same direction as the rotation around the face. Here $\Delta g = g_1 \otimes g_2$ (sum understood) denotes the coproduct $\Delta : H \to H \otimes H$ and $a\triangleright$ is a right action of H^* viewed as a left action of H^{*op}. The antipode $S : H \to H$ is characterised by $(Sh_1)h_2 = h_1 Sh_2 = 1\epsilon(h)$ for all $h \in H$, where $\epsilon \in H^*$ is the counit. We refer to [Maj95] for more details.

We have also used better conventions for $D(H)$, namely the double cross product construction introduced by the 2nd author in [Maj90]. Here $D(H) \cong H^{*op} \bowtie H$, where H left acts on H^* and H^* left acts on H by the coadjoint actions

$$h\triangleright a = a_2 \langle h, (Sa_1)a_3 \rangle\rangle, \quad h\triangleleft a = h_2 \langle a, (Sh_1)h_3 \rangle$$

with the left action of H^* viewed as a right action of H^{*op}. The numerical suffices denote iterated coproducts (sums understood) and $\langle\ ,\ \rangle$ is the duality pairing or evaluation. These then form a matched pair of Hopf algebras[Maj90] and give the Drinfeld double explicitly as [Maj95, Thm 7.1.1],

$$(a \otimes h)(b \otimes g) = b_2 a \otimes h_2 g \langle Sh_1, b_1 \rangle \langle h_3, b_3 \rangle, \quad \Delta(a \otimes h) = a_1 \otimes h_1 \otimes a_2 \otimes h_2.$$

$$S(a \otimes h) = S^{-1} a_2 \otimes Sh_2 \langle h_1, a_1 \rangle \langle Sh_3, a_3 \rangle, \quad \mathcal{R} = \sum_a f^a \otimes 1 \otimes 1 \otimes e_a,$$

where we also give the factorisable quasitriangular structure. Here $\{e_a\}$ is a basis of H and $\{f^a\}$ is a dual basis. We will also sometimes employ a subalgebra notation where h, a are viewed in $D(H)$ with cross relations $ha = a_2 h_2 \langle Sh_1, a_1 \rangle \langle h_3, a_3 \rangle$ and $\mathcal{R} = \sum_a f^a \otimes e_a$. While this much is clear, explicit properties of ribbon operators have not been much studied as far as we can tell even for $S^2 = \mathrm{id}$, and we do so here. Moreover, we will explore how much can be done without this semisimplicity assumption.

From Hopf algebra theory, we will particularly need that every finite-dimensional Hopf algebra H has, uniquely up to normalisation, a left integral element $\Lambda \in H$ such that $h\Lambda = \epsilon(h)\Lambda$ and a right-invariant integral map $\int \in H^*$ such that $(\int h_1)h_2 = 1 \int h$ for all h. Ditto with left-right swapped. In the semisimple case in characteristic zero the integrals can be normalised so that $\epsilon(\Lambda) = \int 1 = 1$, are both left and right integrals at the same time, and obey $\int hg = \int gh$ and $\Delta\Lambda = \mathrm{flip}\Delta\Lambda$ (here $\int = \mathrm{Tr}_H / \dim H$ is the normalised trace in the left regular representation), see [Sch95] for an account (the general theory underlying this goes back to the work of Larson and Radford). If we denote irreps of H by (π, V_π) then analogously to the group case, one has a complete orthogonal set of central idempotents P_π given by

$$P_\pi = \dim(V_\pi)\Lambda_1 \operatorname{Tr}_\pi(S\Lambda_2) \tag{4.23}$$

whereby $P_\pi H = HP_\pi \cong \mathrm{End}V_\pi$. Note that $\sum_\pi P_\pi = \Lambda_1 \int S\Lambda_2 \dim H = 1$ as part of the Frobenius structure where Λ is currently normalised so that $\int \Lambda = 1/\dim H$ compared to usual normalisation in [Sch95, Maj21]. We omit the proof but part of the theory is the orthogonality relation $\operatorname{Tr}_{H^*}(\chi_\pi \chi_{\pi'}) = \delta_{\pi,\pi'} \dim H$ for normalised characters $\chi_\pi = \operatorname{Tr}_\pi / \dim V_\pi$. Moreover, in this case of H semisimple, $D(H)$ is also, with integrals $\Lambda_D = \int \otimes \Lambda$ and $\int_D = \Lambda \otimes \int$. Hence the same result applied to $D(H)$ tells us that $D(H) \cong \oplus_{\tilde\pi}$ $\mathrm{End}(V_{\tilde\pi})$ now for irreps $(\tilde\pi, V_{\tilde\pi})$ of $D(H)$. Hence our ideas about Bell states and ribbon teleportation still apply in this case, with quasiparticles detected by projectors $P_{\tilde\pi}$.

Also note that a representation of $D(H)$ can also be described as a H-crossed [Maj95, Maj02] or 'Drinfeld-Yetter' module consisting of a left action or representation π of H and a compatible *left coaction* of H Δ_L (this is equivalent to a compatible right action of H^* or left action of H^{*op} on the same vector space, these being two subalgebras from which $D(H)$ is built). If $V_{\tilde\pi}$ has basis $\{e_i\}$ then the structures for a crossed module are $\Delta_L e_i = \rho_{ij} \otimes e_j$ where $a \triangleright e_i = \langle a, \rho_{ij}\rangle e_j$ is the corresponding action, and $h \triangleright e_i = \pi(h)_{ki} e_k$ as usual. We sum over the repeated k and $\rho_{ij} \in H$ is required to obey

$$\Delta\rho_{ij} = \rho_{ik} \otimes \rho_{kj}, \quad \epsilon\rho_{ij} = \delta_{ij}, \quad h_1\rho_{ik}\pi(h_2)_{jk} = \pi(h_1)_{ki}\rho_{kj}h_2$$

for all $h \in H$, again summing over k.

Lemma 4.3.1. Let $S^2 = \mathrm{id}$ and $\tilde\pi$ an irrep of $D(H)$ with π, ρ its associated crossed module data with respect to a basis $\{e_i\}$ as above. Then

$$P_{\tilde\pi} = \dim(V_{\tilde\pi}) \sum_{a,i,j} f^a \otimes \Lambda_1 \pi(S\Lambda_2)_{ij} (\int e_a S\rho_{ij}).$$

Moreover, when specialised to $D(G)$, we recover the projectors $P_{\mathcal{C},\pi}$ in (4.9).

Proof. The general formula for the tensor product integral becomes

$$P_{\tilde\pi} = \dim(V_{\tilde\pi}) \int_1 \otimes \Lambda_1 \operatorname{Tr}_{\tilde\pi}((S \int_2) S\Lambda_2) = \dim(V_{\tilde\pi}) f^a \int_1 e_a \otimes \Lambda_1 \langle S \int_2, \rho_{kj}\rangle \langle f^i, e_j\rangle \pi(S\Lambda_2)_{ki}$$

which becomes as stated using Hopf algebra duality. In the $D(G)$ case, we let $\tilde\pi = (\mathcal{C}, \pi)$ as before and go back to (4.23). Then

$$\begin{aligned} P_{\tilde\pi} &= \frac{\dim(V_{\mathcal{C},\pi})}{|G|} \sum_{h,g\in G} (\delta_h \otimes g) \operatorname{Tr}_{\tilde\pi}(S(\delta_{h^{-1}} \otimes g)) \\ &= \frac{\dim V_\pi}{|C_G|} \sum_{h,g\in G} \sum_{c\in\mathcal{C},i} (\delta_h \otimes g) \langle \delta_c \otimes f^i, (\delta_{g^{-1}hg} \otimes g^{-1}) \triangleright (c \otimes e_i)\rangle \\ &= \frac{\dim V_\pi}{|C_G|} \sum_{h,g\in G} \sum_{c\in\mathcal{C}} (\delta_h \otimes g) \delta_{g^{-1}hg, g^{-1}cg} \delta_{c,g^{-1}cg} \operatorname{Tr}_\pi(q^{-1}_{g^{-1}cg} g^{-1} q_c) = P_{\mathcal{C},\pi} \end{aligned}$$

where $\triangleright$ is the action (4.13). We view the restrictions setting $h = c$ and $g \in C_G(c)$. Changing variables to $n = q_c g q_c^{-1}$, this is equivalent to a sum over $n \in C_G(r_{\mathcal{C}})$ as usual and n^{-1} in the trace. ■

How exactly to construct and classify irreps $\tilde{\pi}$, however, depends on the structure of $D(H)$, which is no longer generally a semidirect product. This therefore has to be handled on a case by case basis before one can do practical quantum computations.

Example 4.3.2. Let $G = G_+.G_-$ be a finite group that factorises into two subgroups $G_\pm$, neither of which need be normal and $H = \mathbb{C}(G_-)\blacktriangleright\!\!\triangleleft\mathbb{C}G_+$ the associated bicrossproduct (or 'bismash product') quantum group[Tak81, Maj90, Maj95], which is semisimple. It is shown in [BGM96] that $D(H)\cong D(G)_F$, where the latter is a Drinfeld twist of $D(G)$ by a 2-cocycle

$$F = \sum_{g\in G} 1 \otimes g_- \otimes \delta_{g^{-1}} \otimes 1 \in D(G)\otimes D(G)$$

in the sense of [Maj95]. We write $g = g_+g_-$ for the unique factorisation of any element of G. Explicitly, $D(G)_F$ has the same algebra as $D(G)$ but a conjugated coproduct

$$\Delta(\delta_g \otimes h) = F(\Delta_{D(G)}(\delta_g \otimes h))F^{-1} = \sum_{f\in G} \delta_{f_-gff_-^{-1}} \otimes f_-h(h^{-1}fh)_-^{-1} \otimes \delta_{f^{-1}} \otimes h$$

after a short computation. The nontrivial isomorphism with $D(H)$ in [BGM96] is needed to identify the H and H^{*op} subalgebras but where this is not required, we can work directly with this twisted description. In particular, irreps of $D(G)_F$ are the same as those of $D(G)$ (since the algebra is not changed) and can be identified with irreps of $D(H)$ by the isomorphism. The braided tensor category is different from but monoidally equivalent to that of $D(G)$.

We will be concerned more with the formalism with explicit models, such as based on this construction, deferred to a sequel. We see, however, that there are plenty of examples. Note that $G\rtimes G$ by Ad is an example with one subgroup normal, so $H = D(G)$ is covered by this analysis and $D(D(G))\cong D(G\rtimes G)_F$.

4.3.1 $D(H)$ site operators.

By working with the above cleaner form of the Drinfeld double, our modest new observation in this section is that the same format for the Kitaev model works in the general case without assuming $S^2 = \mathrm{id}$ provided we use additional information from the lattice geometry to distinguish the four cases (a)-(d) in Figure 4.3 which follow the same rules as above but sometimes specify to use S^{-1}. We focus on the case of a square lattice with its standard orientation as this is most relevant to computer science, rather than on a general ciliated ribbon graph.

Theorem 4.3.3. If (v, p) is a site in the lattice then the actions for the form in Figure 4.3, where we act as shown and by the identity on other arrows, is a representation of $D(H)$ provided we use (as shown) S^{-1} if the first arrow going around the vertex is inward and S if the last arrow is inward. We can freely choose S or S^{-1} if the inward arrow is in one of the intermediate places.

Proof. We have to check the relations $a_2h_2\langle Sh_1, a_1\rangle\langle h_3, a_3\rangle = ha$ for all $h \in H$ and $a \in H^*$. The proof of the hardest case (c) is in Figure 4.4 and for the cancellation for the 3rd equality, we see that we need S^{-1} when the first vertex going around is inward and S when the last vertex is, as is the case here. The other cases are similar but slightly easier as unconstrained on the choice of S where there is no inward arrow in one or both of these positions. ■

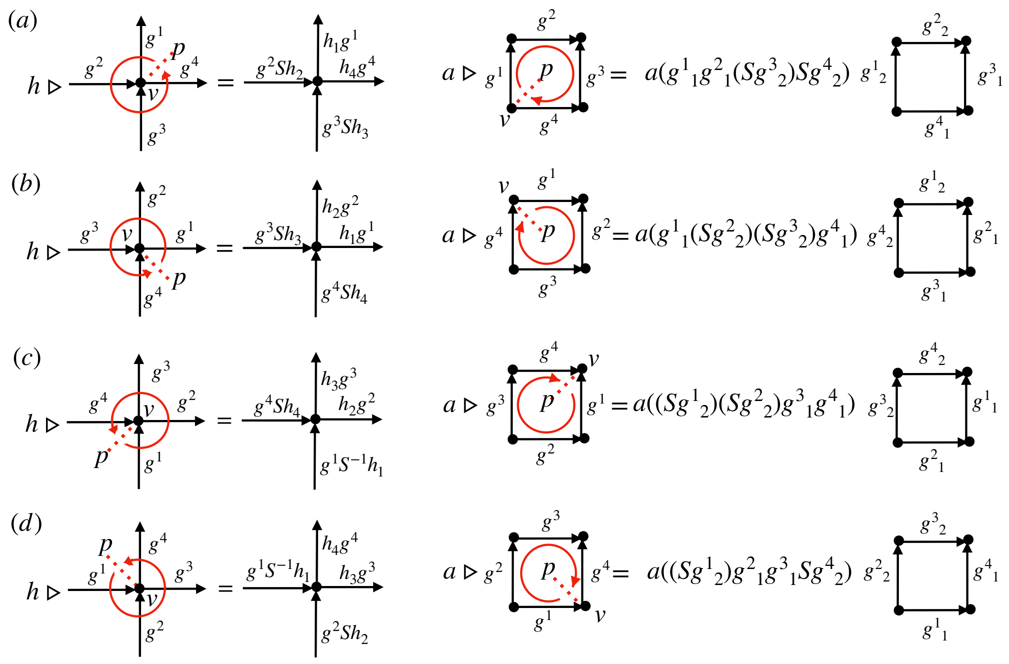

(a)
$h \triangleright$
$= a(g^1{}_1 g^2{}_1 (Sg^3{}_2) Sg^4{}_2)$
(b)
$= a(g^1{}_1 (Sg^2{}_2)(Sg^3{}_2) g^4{}_1)$
(c)
$= a((Sg^1{}_2)(Sg^2{}_2) g^3{}_1 g^4{}_1)$
(d)
$= a((Sg^1{}_2) g^2{}_1 g^3{}_1 Sg^4{}_2)$

Figure 4.3: Kitaev model representation of $D(H)$ at a site (v, p) for general not necessarily semisimple Hopf algebras. Most instances of the antipode S can be equally S^{-1} but if we use S in all the $a\triangleright$ then we have to use S^{-1} in $h\triangleright$ if this occurs in the first arrow encountered in going around the vertex.

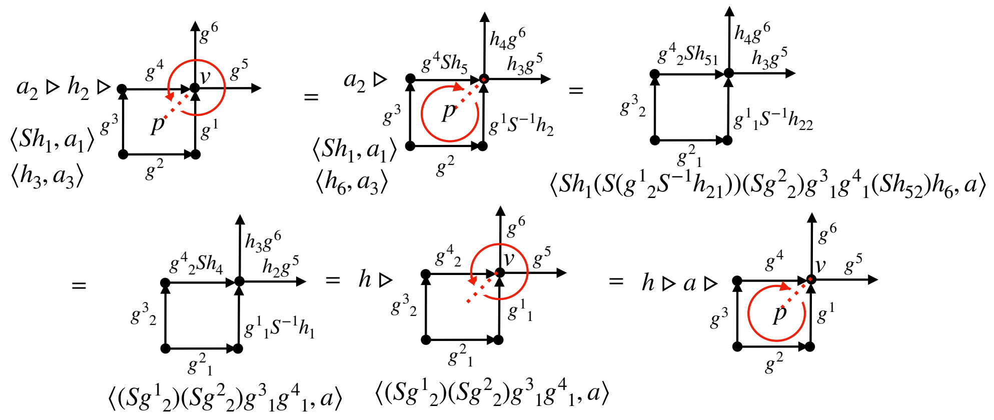

$a_2 \triangleright h_2 \triangleright$
$\langle Sh_1, a_1 \rangle$
$\langle h_3, a_3 \rangle$
$\langle h_6, a_3 \rangle$
$\langle Sh_1(S(g^1{}_2 S^{-1} h_{21}))(Sg^2{}_2) g^3{}_1 g^4{}_1 (Sh_{52}) h_6, a \rangle$
$\langle (Sg^1{}_2)(Sg^2{}_2) g^3{}_1 g^4{}_1, a \rangle$
$h \triangleright a \triangleright$

Figure 4.4: Proof that case (c) of Figure 4.3 works in Theorem 4.3.3.

We could equally well decide to always use S for $h\triangleright$ and use S^{-1} in $a\triangleright$ if the contraflowing is in first position going around the face and S if it is in last position. (This is the same as above in the dual lattice and faces and arrows interchanged and with the roles of H, H^* interchanged.) *We see that in the non-involutive S case there is still some freedom in the choice of S or S^{-1}, which we need to fix by what we want to do with these $D(H)$-representations.*

We also know that our finite dimensional Hopf algebra has up to scale a unique right integral $\Lambda \in H$ and a unique right-invariant integral $\int : H \to k$ so we can proceed to define operators

$$A(v,p) = \Lambda\triangleright, \quad B(v,p) = \int \triangleright$$

on $\mathcal{H}$. It is striking that exactly this integral data is also key to a Frobenius Hopf algebra interacting pair for ZX calculus based on H, see [CD19, Maj21] at this level of generality. Clearly

$$A(v,p)^2 = \epsilon(\Lambda)A(v,p), \quad B(v,p)^2 = (\int 1)B(v,p)$$

but without further assumptions, both operators depend on both parts of the site. One can also check that

$$[A(v,p), A(v',p')] = 0, \quad [B(v,p), B(v',p')] = 0$$

for all v, v', p, p' with in the first case $v \neq v'$ and in the second case $p \neq p'$. The first is because if, in the worst case, the vertices are adjacent then the common arrow is pointing in for one vertex and out for the other, hence the element g in the middle gets multiplied by something on the left and something on the right, which does not depend on the order by associativity. Similarly for two faces with an arrow in common. We do *not* in general have that $[A(v,p), B(v,p)] = 0$.

For the Hamiltonian, there are two possible approaches. (i) we could we fix the vertex of all site to be at the bottom left of the face (Case (a) in Figure 4.3). Thus if v is a vertex, we define p_v as the face to its upper right. Then

$$H = \sum_v (1 - A(v,p_v) + 1 - B(v,p_v))$$

makes sense. (ii) Alternatively, motivated by [Meu17] we can define

$$H_K = \prod_{(v,p)} A(v,p)B(v,p)$$

The $D(G)$ model admits a Hamiltonian which is necessarily frustration-free, meaning that any vacuum state is also the lowest energy state of any given local term. This condition is broken by general $D(H)$ models. Let $A(v_1,p_1)$ be a local term. First, consider the Hamiltonian from (i), ignoring the additive constant:

$$\begin{aligned} A(v_1,p_{v_1})|\mathrm{vac}\rangle &= -A(v_1,p_{v_1})\sum_v (A(v,p_v) + B(v,p_v))|\mathrm{vac}\rangle \\ &= -(\epsilon(\Lambda)A(v_1,p_{v_1}) + A(v_1,p_{v_1})B(v_1,p_{v_1}) + \sum_{v\neq v_1}(A(v,p_v) + B(v,p_v)))|\mathrm{vac}\rangle \end{aligned}$$

So in general, $A(v_1, p_{v_1})|\mathrm{vac}\rangle \neq |\mathrm{vac}\rangle$. Next, consider the Hamiltonian from (ii):

$$A(v_1, p_{v_1})|\mathrm{vac}\rangle = A(v_1, p_{v_1}) \prod_{(v,p)} A(v,p)B(v,p)|\mathrm{vac}\rangle = \epsilon(\Lambda)|\mathrm{vac}\rangle$$

The fact that the integral actions are no longer idempotent also breaks the interpretation of these actions as 'check operators' to be measured and detect unwanted excitations. In these more general models, it is unclear what error-correcting capabilities still exist on the lattice, or whether there are alternative methods of preserving fault-tolerance. They don't appear to fit under the umbrella of 'surface codes' in the usual sense. We still preserve some locality as a feature of the model, in the sense that a locally vacuum state can be defined for example in case (ii) as being the image of $\prod_{(v\in V_R, p\in P_R)} A(v,p)B(v,p)$, where V_R and P_R are sets of vertices and faces in some region R.

In the semisimple case where $S^2 = \mathrm{id}$, we have already noted that $\Delta\Lambda = \mathrm{flip}\Delta\Lambda$ and that the integrals can be normalised so that $\epsilon(\Lambda) = \int 1 = 1$. This implies that $A(v,p) = A(v)$ independently of p and $B(v,p) = B(p)$ independently of v, are projectors and, using the commutation relations of $D(H)$ and Theorem 4.3.3 that $[A(v), B(p)] = 0$. Therefore, we recover both the frustration-free property of H and the interpretation of the lattice as a fault-tolerant quantum memory. In this case, it is claimed in [Meu17] that

$$H_K = \prod_v A(v) \circ \prod_p B(p), \quad \mathcal{H}_{vac} = \mathrm{image}(H_K)$$

results in the latter 'protected space' being a topological invariant of the surface obtained by gluing discs on the faces of a ribbon graph. This motivates the definition above. It is also claimed in [Meu17] that a particle at (v,p) corresponds to a defect where we leave out the site (v,p) in the product.

While the $D(G)$ model on a lattice Σ allows for the convenient expression in Theorem 4.2.2 for $\dim(\mathcal{H}_{vac})$ in terms of the fundamental group $\pi_1(\Sigma)$, the proof of this relies on the invertibility of group elements and the invariance under orientation of $\delta_e \triangleright$. The topological content for $D(H)$ models can similarly be related to holonomy as in [Meu17] but is more complicated. The topological content in the $D(H)$ model in the non-semisimple case is less clear and will be more indirect. For example, reversal of orientation cannot be expressed simply via the antipode as this no longer squares to the identity.

4.3.2 $D(H)$ triangle and ribbon operators

Canonical representations of $D(H)$ that we will need are left and right actions of $D(H)$ on H, [Maj95, Ex. 7.1.8]

$$h\triangleright g = h_1 g S h_2, \quad a\triangleright g = a(g_1)g_2; \quad g\triangleleft h = (Sh_1)gh_2, \quad g\triangleleft a = g_1 a(g_2) \tag{4.24}$$

and left and right actions of $D(H)$ on H^*,

$$h\triangleright b = \langle Sh, b_1\rangle b_2, \quad a\triangleright b = (S^{-1}a_2)ba_1; \quad b\triangleleft h = b_1\langle Sh, b_2\rangle, \quad b\triangleleft a = a_2 b S^{-1}a_1 \tag{4.25}$$

which is essentially the same construction with the roles of H, H^{*op} swapped. Moreover, in the quasitriangular case, there is a braided monoidal functor ${}_H\mathcal{M} \to {}_{D(H)}\mathcal{M}$, see [Maj95, Maj02], which can be used to obtain a class of nice representations of $D(H)$ from irreps of H.

Also note that if D is any Hopf algebra, for example $D = D(H)$, it acts on its dual as a module algebra by the left and right coregular representations

$$d\triangleright\phi = \langle Sd, \phi_1\rangle\phi_2, \quad \phi\triangleleft d = \phi_1\langle Sd, \phi_2\rangle \tag{4.26}$$

for all $d \in D$ and $\phi \in D^*$. These already feature in (4.25) for the action of H. Also, if D acts from the left (say) on a vector space $\mathcal{H}$ by an action $\triangleright$ then it acts on the linear operators $\mathrm{End}(\mathcal{H})$ as a module algebra from both the left and the right [Maj95].

$$(d\triangleright L)(\psi) = d_1\triangleright L(Sd_2\triangleright\psi), \quad (L\triangleleft d)(\psi) = Sd_1\triangleright L(d_2\triangleright\psi) \tag{4.27}$$

for all d in the Hopf algebra and $L \in \mathrm{End}(\mathcal{H})$. We will use (4.27) with different site actions of $D(H)$ in Theorem 4.3.3 for example $\triangleright_{s_0}$ and $\triangleright_{s_1}$ for the two halves. These commute if s_0, s_1 are far enough apart. In the case of $D(H)$, its dual is $H \otimes H^*$ as an algebra and has the coproduct

$$\Delta_{D(H)^*}(h \otimes a) = \sum_{a,b} h_2 \otimes f^a a_1 f^b \otimes Se_a h_1 e_b \otimes a_2. \tag{4.28}$$

Next, we define Hopf algebra triangle and ribbon operators at least in the case $S^2 = \mathrm{id}$. Before attempting this, we need to better understand the $D(G)$ case, both ribbon covariance properties and the construction of a ribbon a sequence of triangle and dual triangle ops as defined in Definition 4.2.9.

Lemma 4.3.4. For $D(G)$ and with left and right actions on $\mathrm{End}(\mathcal{H})$ induced as in (4.27) by the initial and final site actions:

1. If τ^* is a dual triangle, the triangle operator $L^h_{\tau*} = \sum_g F^{h\otimes\delta_g}_{\tau^*}$ is a left and right module map $L_{\tau^*} : \mathbb{C}G \to \mathrm{End}(\mathcal{H})$, where $D(G)$ acts as in (4.24) by
$$(\delta_a \otimes b)\triangleright h = \delta_{a,bhb^{-1}} bhb^{-1}, \quad h\triangleleft(\delta_a \otimes b) = b^{-1}hb\,\delta_{a,h}.$$
2. If τ is a direct triangle, the triangle operator $T^{\delta_g}_\tau = F^{h\otimes\delta_g}_\tau$ for any h is a left and right module map $T_\tau : \mathbb{C}(G) \to \mathrm{End}(\mathcal{H})$, where $D(G)$ acts as in (4.25) by
$$(\delta_a \otimes b)\triangleright\delta_g = \delta_{a,e}\delta_{bg}, \quad \delta_g\triangleleft(\delta_a \otimes b) = \delta_{a,e}\delta_{gb}.$$
3. If ξ is an open ribbon then $\tilde{F}^{h\otimes\delta_g}_\xi := F^{h^{-1},g}_\xi$ is a left and right module map $\tilde{F} : D(G)^* \to \mathrm{End}(\mathcal{H})$, where $D(G)$ acts by (4.26). Moreover, if ξ, ξ' are composeable ribbons then
$$\tilde{F}^{h\otimes\delta_g}_{\xi'\circ\xi} = \tilde{F}^{(h\otimes\delta_g)_2}_{\xi'} \circ \tilde{F}^{(h\otimes\delta_g)_1}_\xi$$
using the coproduct (4.28) of $D(G)^*$. This also applies to $F^{h\otimes\delta_g}_\xi = F^{h,g}_\xi$.

Proof. (1) The relations we find for dual triangles are

$$(\delta_a \otimes b)\triangleright_{s_0}\circ\tilde{F}^{b^{-1}hb\otimes\delta_g} = \tilde{F}^{h\otimes\delta_g}\circ(\delta_{ha} \otimes b)\triangleright_{s_0}, \quad (\delta_{ah} \otimes b)\triangleright_{s_1}\circ\tilde{F}^{b^{-1}hb\otimes\delta_g} = \tilde{F}^{h\otimes\delta_g}\circ(\delta_a \otimes b)\triangleright_{s_1}$$

which we interpret as stated.

(2) The relations we find for direct triangles are

$$(\delta_a \otimes b)\triangleright_{s_0} \circ \tilde{F}^{h\otimes\delta_g} = \tilde{F}^{h\otimes\delta_{bg}} \circ (\delta_a \otimes b)\triangleright_{s_0}, \quad \tilde{F}^{h\otimes\delta_g} \circ (\delta_a \otimes b)\triangleright_{s_1} = (\delta_a \otimes b)\triangleright_{s_1} \circ \tilde{F}^{h\otimes\delta_{gb}}$$

which we interpret as stated. These same commutation rules hold for the action $\triangleright_t$ at any site t that has the same vertex as s_0, s_1 respectively, while the action at other sites commutes with the triangle operator.

(3) Here $D(G)^* = \mathbb{C}G \otimes \mathbb{C}(G)$ as an algebra while its coproduct dual to the product of $D(G)$ is

$$\Delta_{D(G)^*}(h \otimes \delta_g) = \sum_{f \in G} h \otimes \delta_f \otimes f^{-1}hf \otimes \delta_{f^{-1}g}$$

Then the composition rule in equation (4.15) for F_ξ is already in the form stated. The same then applies $\tilde{F}_\xi$ as $S \otimes \mathrm{id}$ is clearly a coalgebra map. For equivariance, we have from Lemma 4.2.8,

$$\begin{aligned}
&\langle S(\delta_a \otimes b)_1, (h \otimes \delta_g)_1 \rangle \tilde{F}_\xi^{(h \otimes \delta_g)_{(2)}} \circ (\delta_a \otimes b)_1 \triangleright_{s_0} \\
&= \sum_{x,f} \langle S(\delta_{x^{-1}} \otimes b), h \otimes \delta_f \rangle \tilde{F}_\xi^{f^{-1}hf \otimes \delta_{f^{-1}g}} \circ (\delta_{xa} \otimes b)_1 \triangleright_{s_0} \\
&= \sum_{x,f} \langle \delta_{b^{-1}xb} \otimes b^{-1}, h \otimes \delta_f \rangle \tilde{F}_\xi^{f^{-1}hf \otimes \delta_{f^{-1}g}} \circ (\delta_{xa} \otimes b)_1 \triangleright_{s_0} \\
&= \tilde{F}_\xi^{bhb^{-1} \otimes \delta_{fg}} \circ (\delta_{bhb^{-1}a} \otimes b) \triangleright_{s_0} = \delta_a \triangleright \tilde{F}_\xi^{bhb^{-1} \otimes \delta_{bg}} \circ b \triangleright_{s_0} \\
&= (\delta_a \otimes b) \triangleright_{s_0} \tilde{F}_\xi^{h \otimes \delta_g}
\end{aligned}$$

where $f = b^{-1}$ and $x = bhb^{-1}$. We used the commutation relations from Lemma 4.2.8. Similarly for the other side,

$$\begin{aligned}
&(\delta_a \otimes b)_1 \triangleright_{s_1} \circ \tilde{F}_\xi^{(h \otimes \delta_g)_1} \langle S(\delta_a \otimes b)_{(2)}, (h \otimes \delta_g)_{(2)} \rangle \\
&\quad = \sum_{x,f} (\delta_{ax} \otimes b) \triangleright_{s_1} \circ \tilde{F}_\xi^{h \otimes \delta_f} \langle S(\delta_{x^{-1}} \otimes b), f^{-1}hf \otimes \delta_{f^{-1}g} \rangle \\
&\quad = \sum_{x,f} (\delta_{ax} \otimes b) \triangleright_{s_1} \circ \tilde{F}_\xi^{h \otimes \delta_f} \langle \delta_{b^{-1}xb} \otimes b^{-1}, f^{-1}hf \otimes \delta_{f^{-1}g} \rangle \\
&\quad = (\delta_{ag^{-1}hg} \otimes b) \triangleright_{s_1} \circ \tilde{F}_\xi^{h \otimes \delta_{gb}} = \delta_{ag^{-1}hg} \triangleright_{s_1} \circ \tilde{F}_\xi^{h \otimes \delta_g} \circ b \triangleright_{s_1} = \tilde{F}_\xi^{h \otimes \delta_g} \circ (\delta_a \otimes b) \triangleright_{s_1}
\end{aligned}$$

where $gb = f$ and $x = bf^{-1}hfb^{-1} = g^{-1}hg$. ■

The additional commutation relations for T_τ mentioned in the proof can best be said as the action on it as an operator in $\mathrm{End}(\mathcal{H})$,

$$(\delta_a \otimes b) \triangleright_t (T^{\delta_g}) = T^{(\delta_a \otimes b) \triangleright_t \delta_g}; \quad (\delta_a \otimes b) \triangleright_t \delta_g = \delta_{a,e} \begin{cases} \delta_{bg} \\ \delta_{gb^{-1}} \\ \delta_g \end{cases}$$

where we act as per L^b at vertex t on g regarded effectively as living on the arrow of the direct triangle, i.e. bg if the arrow in relation to the vertex of t is outgoing, gb^{-1} if incoming and g otherwise. We can then derive the left and right module properties for ribbon operators by iterating those for triangle operators. To illustrate this, consider

$$\tilde{F}_{\tau_2 \circ \tau_1^*}^{h \otimes \delta_g} = T_{\tau_2}^{\delta_g} \circ L_{\tau_1^*}^{h^{-1}} = \sum_f \tilde{F}_{\tau_2}^{f^{-1}hf \otimes \delta_{f^{-1}g}} \circ \tilde{F}_{\tau_1^*}^{h \otimes \delta_f} = (\tilde{F}_{\tau_2} \circ \tilde{F}_{\tau_1^*})(h \otimes \delta_g)$$

where $\tau_1^* : s_0 \to s_1$ and $\tau_2 : s_1 \to s_2$ and $\tilde{F}_{\tau}^{h\otimes\delta_g} = T_{\tau}^{\delta_g}$ and $\tilde{F}_{\tau^*}^{h\otimes\delta_g} = \delta_{g,e} L_{\tau^*}^{h^{-1}}$ are the associated ribbon operators which we convolve as in part (3) of the lemma. Using the first expression and the triangle operator left module properties

$$\begin{aligned}(\delta_a\otimes b){\triangleright}_{s_0}(\tilde{F}^{h\otimes\delta_g}_{\tau_2\circ\tau_1^*}) &= (\delta_a\otimes b)_1{\triangleright}(T^{\delta_g}_{\tau_2})\circ(\delta_a\otimes b)_2{\triangleright}(L^{h^{-1}}_{\tau_1^*})\\ &= \sum_x T_{\tau_2}^{(\delta_{ax^{-1}}\otimes b){\triangleright}_{s_0}\delta_g}\circ L_{\tau_1^*}^{(\delta_x\otimes b){\triangleright} h^{-1}}\\ &= T_{\tau_2}^{\delta_{bg}}\circ L_{\tau_1^*}^{\delta_{a,bh^{-1}b^{-1}}bh^{-1}b^{-1}} = \delta_{a^{-1},bhb^{-1}}\tilde{F}^{bhb^{-1}\otimes\delta_{bg}}_{\tau_2\circ\tau_1^*} = \tilde{F}^{(\delta_a\otimes b){\triangleright}(h\otimes\delta_g)}_{\tau_2\circ\tau_1^*}\end{aligned}$$

where from (4.26),

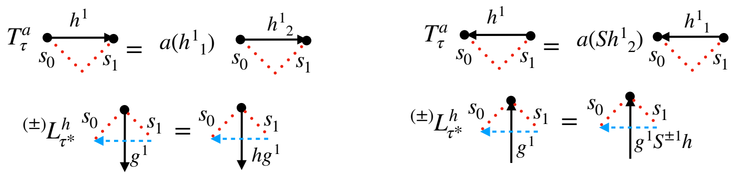
$$(\delta_a\otimes b){\triangleright}(h\otimes\delta_g) = \sum_f\langle\delta_{b^{-1}a^{-1}b}\otimes b^{-1}, h\otimes\delta_f\rangle f^{-1}hf\otimes\delta_{f^{-1}g} = \delta_{h,b^{-1}a^{-1}b}bhb^{-1}\otimes\delta_{bg}$$

The similar calculation for $(\tilde{F}^{h\otimes\delta_g}_{\tau_2\circ\tau_1^*}){\triangleleft}_{s_2}(\delta_a\otimes b)$ is not so easy as ${\triangleleft}_{s_2}$ does not enjoy simple commutation relations with $L^{h^{-1}}$. There is a similar story for

$$\tilde{F}^{h\otimes\delta_g}_{\tau_2^*\circ\tau_1} = L^{g^{-1}h^{-1}g}_{\tau_2^*}\circ T^{\delta_g}_{\tau_1} = \sum_f \tilde{F}^{f^{-1}hf\otimes\delta_{f^{-1}g}}_{\tau_2^*}\circ\tilde{F}^{h\otimes\delta_f}_{\tau_1} = (\tilde{F}_{\tau_2^*}\circ\tilde{F}_{\tau_1})(h\otimes\delta_g)$$

as the other smallest open ribbon.

Now proceeding to the Hopf algebra case, we define triangle operations in the obvious manner as partial vertex and face operators, with left multiplication by h or right multiplication by $S^{\pm1}h$ for dual triangles, depending on orientation,



Recall that we have chosen to use S throughout the face operations but S^{-1} if the first arrow was inward in a vertex operation. As a result, we need both versions ${}^{(\pm)}L^h_{\tau^*}$ to express both left and right covariance. We could equally well have put this complication on the T^a_τ side.

Lemma 4.3.5. Let H be a finite-dimensional Hopf algebra, $D(H)$ act on H, H^* as in (4.24) and (4.25) and act on End($\mathcal{H}$) as in (4.27) from the left induced by ${\triangleright}_{s_0}$ and from the right induced by ${\triangleright}_{s_1}$.

1. For dual triangles, ${}^{(-)}L_{\tau^*} : H \to \mathrm{End}(\mathcal{H})$ is a left module map and ${}^{(+)}L_{\tau^*} : H \to \mathrm{End}(\mathcal{H})$ is right module map.

2. The direct triangle operator $T_\tau : H^* \to \mathrm{End}(\mathcal{H})$ is a left and right module map.

Proof. This is shown in Figure 4.5 and Figure 4.6 for sample orientations where $S^{\pm}$ appears in the dual triangle operation. The other orientations are similar with less work. We use the definitions and the actions of $D(H)$ on H and H^*. ■

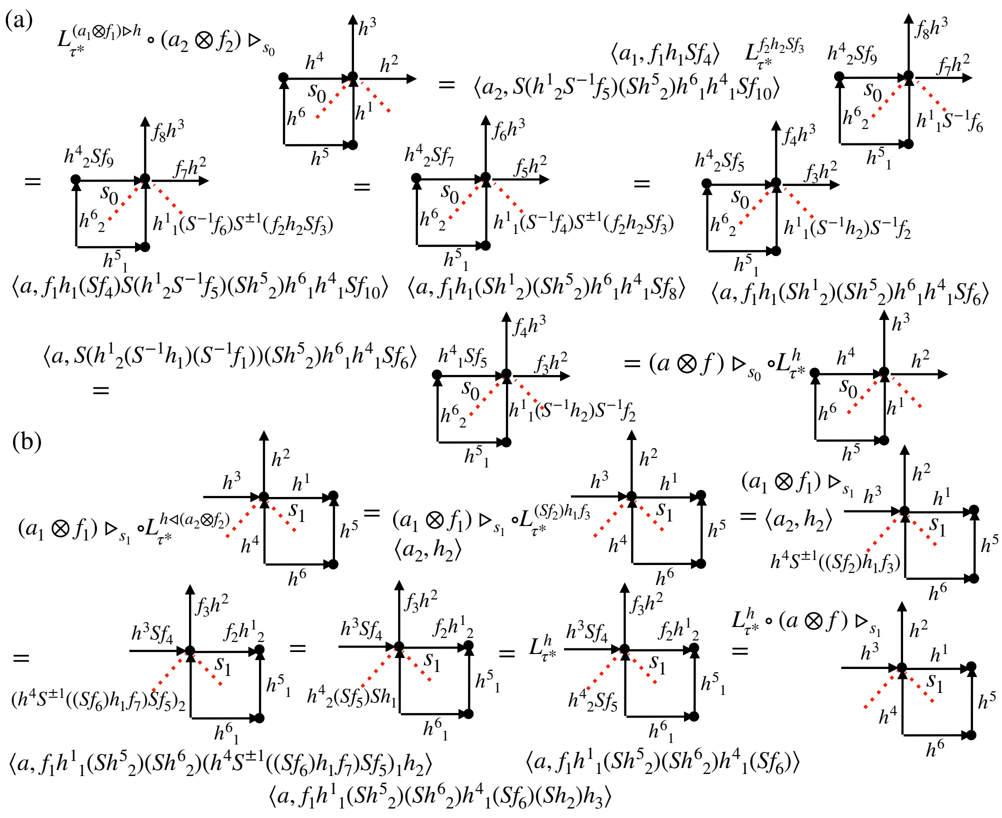


Figure 4.5: (a) Proof of left covariance of dual triangle operator needing the $(-)$ version for the 3rd equality. (b) Proof of right covariance needing the $(+)$ version. They coincide when $S^2 = \mathrm{id}$.

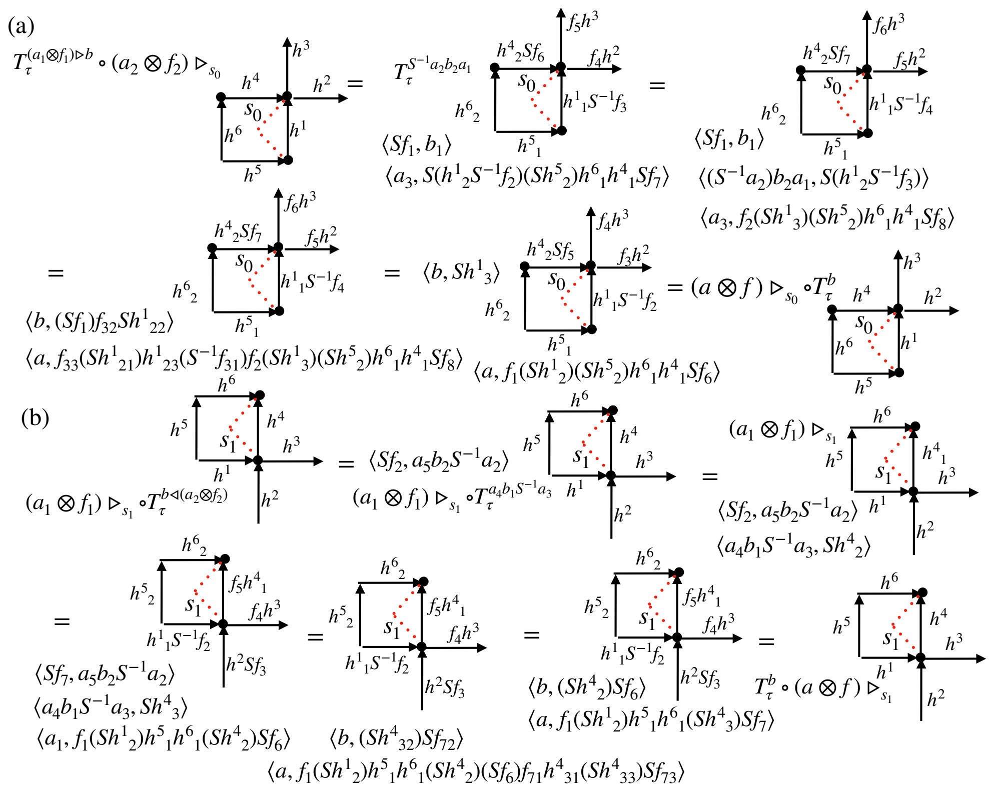


Figure 4.6: Proof of (a) left covariance and (b) right covariance of direct triangle operator.

Next, we define ribbon operators $F_\xi^{h\otimes a}$ associated to a ribbon ξ by convolution-composition of triangle operations, where $h \in H$ and $a \in H^*$. They are a special case of the 'holonomy' maps defined in [Meu17] but even so, it is nontrivial to write them out explicitly in our case and in our notations. The first step it to view triangle operators as ribbon operators by

$$\tilde{F}_\tau^{h\otimes a} = \epsilon(h)T_\tau^a, \quad {}^{(\pm)}\tilde{F}_{\tau^*}^{h\otimes a} = \epsilon(a){}^{(\pm)}L_{\tau^*}^{S^{-1}h}. \tag{4.29}$$

Next, the ribbon operators for two composeable ribbons can be convolution-composed by

$$\tilde{F}_{\xi'\circ\xi}^{\phi} = \tilde{F}_{\xi'}^{\phi_2} \circ \tilde{F}_{\xi}^{\phi_1} \tag{4.30}$$

where now we use the coproduct (4.28) on $\phi \in D(H)^*$. This is an associative operation, so starting with a triangle operation viewed as a ribbon operator and extending to a ribbon by composing a series of these, we correspondingly define the associated ribbon operator by iterating this formula. Because $\epsilon \otimes \mathrm{id}$ and $\mathrm{id} \otimes \epsilon$ are coalgebra maps from $D(H)^*$ to H^*, H^{cop} respectively, the convolution of direct triangle operators viewed as ribbons is the same as convolution of Ts via the coproduct of H^* and (due to the S^{-1}) the convolution of dual triangle operators as ribbons is the same as convolution of Ls via the coproduct of H. It follows that the $D(H)$ site actions in Theorem 4.3.3 can be viewed as ribbon operators, where for our default conventions $a\triangleright = T^{a_4} \circ \cdots \circ T^{a_1}$ going clockwise around a face and $h\triangleright = {}^{(+)}L^{h_4} \circ \dots {}^{(+)}L^{h_2} \circ {}^{(-)}L^{h_1}$ going anticlockwise around a vertex. The sign refers to the use of $S^{\pm 1}$ if applicable and as noted, we can also have different patterns of signs, including in the T's, and still get an action of $D(H)$.

In particular, this means that we no longer have a clear route to topological invariance as we can construct a contractible, closed ribbon equal to $A(v,p)$ (or $B(v,p)$). $A(v,p)|\mathrm{vac}\rangle \neq |\mathrm{vac}\rangle$ in general, so ribbon operators are no longer invariant up to isotopy. As a consequence, it is not clear that $\dim(\mathcal{H}_{vac})$ is a topological invariant in general, albeit it is known to be one in the semisimple case.

Next, we wish to prove a generalisation of Lemma 4.2.8 from Section 4.2. However, we could previously rely on topological invariance to justify claims in our proofs by bending ribbons into a convenient shape and eliminating contractible loops. In the non-semisimple case, we need to specify a new class of ribbon for which our generalisation applies and where these steps are not needed.

Definition 4.3.6. Recall that a ribbon is a sequence of triangles between sites. Let us represent it as a list of sites in order $[s_0, s_1, \cdots, s_n] := [(v_0, p_0), (v_1, p_1), \cdots, (v_n, p_n)]$, which must change either the vertex or face between each site. A *strongly open* ribbon ξ is an open ribbon which satisfies the following condition: any two sites $s_i := (v_i, p_i)$, $s_j := (v_j, p_j)$ inside ξ may have $v_i = v_j$ only if every site in the sequence of sites $[s_{i+1}, \cdots, s_{j-1}]$ between s_i, s_j also has $v_{i+1} = \cdots = v_{j-1}$. Similarly, $p_i = p_j$ is only allowed if $p_{i+1} = \cdots = p_{j-1}$.

The intuition behind this is that the ribbon ξ does not bend 'too quickly' or get 'too close' to itself; equivalently, it is saying that all subribbons of ξ are either on a single vertex/face or are themselves open.

Example 4.3.7. Figure 4.7a shows a strongly open ribbon. While it has rotations at a vertex and a face, it never returns to previously seen vertices or faces. However, Figure 4.7b is an open ribbon which is not strongly open: at the self-crossing of the ribbon, there are sites s_2, s_3 such that $s_2 = s_3$ but they are not sequentially adjacent in the ribbon. Similarly, Figure 4.7c is an open ribbon which does not cross itself but gets 'too close' – sites s_4, s_5 intersect at p_4.

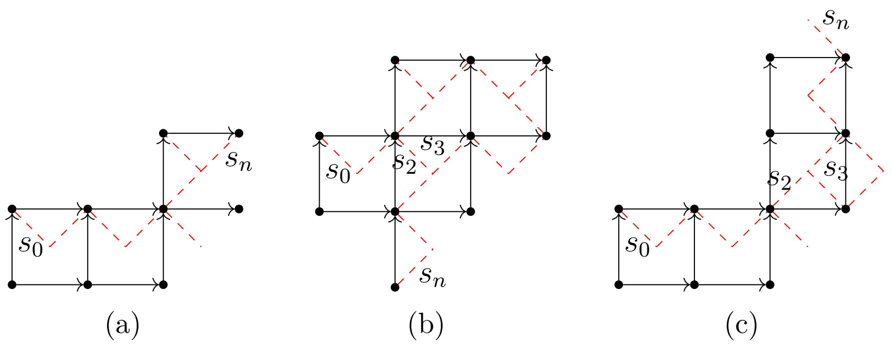
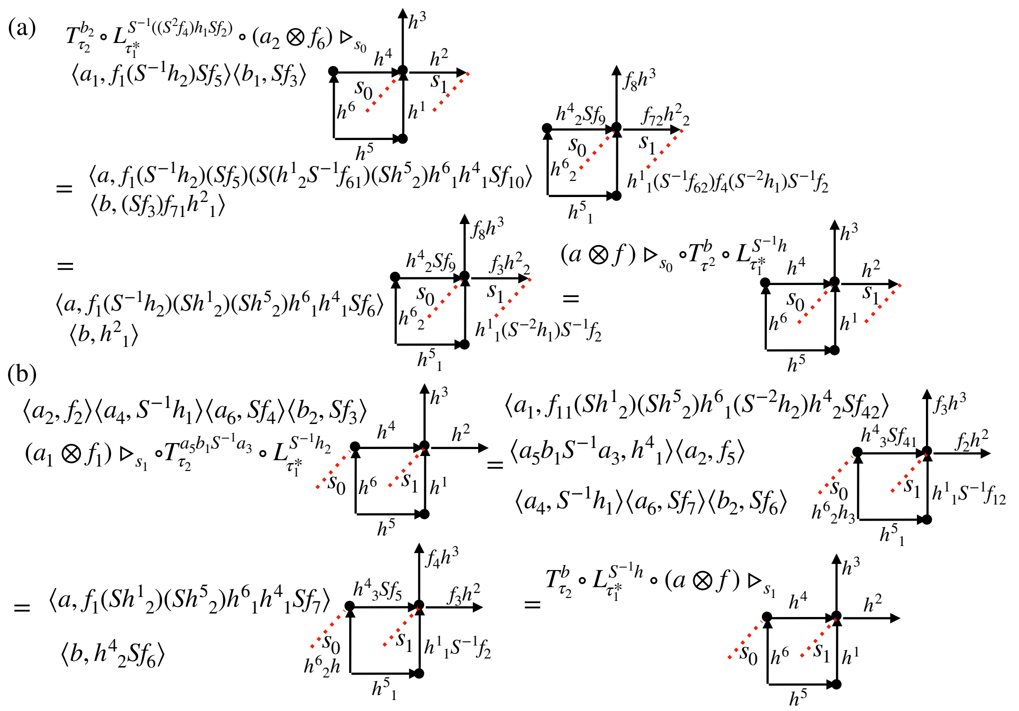


Figure 4.7: (A) is a strongly open ribbon. (B) and (C) are open, but not strongly open.



(a)
$T^{b_2}_{\tau_2} \circ L^{S^{-1}((S^2 f_4) h_1 S f_2)}_{\tau_1^*} \circ (a_2 \otimes f_6) \triangleright_{s_0}$
$\langle a_1, f_1(S^{-1}h_2) S f_5 \rangle \langle b_1, S f_3 \rangle$
h^3 h^4 h^2 s_0 s_1 h^6 h^1 h^5
$= \langle a, f_1(S^{-1}h_2)(Sf_5)(S(h^1{}_2 S^{-1} f_{61})(Sh^5{}_2) h^6{}_1 h^4{}_1 S f_{10} \rangle$
$\langle b, (Sf_3) f_{71} h^2{}_1 \rangle$
$f_8 h^3$ $h^4{}_2 S f_9$ $f_{72} h^2{}_2$ s_0 s_1 $h^6{}_2$ $h^1{}_1 (S^{-1} f_{62}) f_4 (S^{-2} h_1) S^{-1} f_2$ $h^5{}_1$
$=$
$\langle a, f_1(S^{-1}h_2)(Sh^1{}_2)(Sh^5{}_2) h^6{}_1 h^4{}_1 S f_6 \rangle$
$\langle b, h^2{}_1 \rangle$
$f_8 h^3$ $h^4{}_2 S f_9$ $f_3 h^2{}_2$ s_0 s_1 $h^6{}_2$ $h^1{}_1 (S^{-2} h_1) S^{-1} f_2$ $h^5{}_1$
$=$
$(a \otimes f) \triangleright_{s_0} \circ T^b_{\tau^2} \circ L^{S^{-1}h}_{\tau_1^*}$
h^3 h^4 h^2 s_0 s_1 h^6 h^1 h^5
(b)
$\langle a_2, f_2 \rangle \langle a_4, S^{-1} h_1 \rangle \langle a_6, S f_4 \rangle \langle b_2, S f_3 \rangle$
$(a_1 \otimes f_1) \triangleright_{s_1} \circ T^{a_5 b_1 S^{-1} a_3}_{\tau_2} \circ L^{S^{-1} h_2}_{\tau_1^*}$
h^3 h^4 h^2 s_0 h^6 s_1 h^1 h^5
$= \langle a_1, f_{11}(Sh^1{}_2)(Sh^5{}_2) h^6{}_1 (S^{-2} h_2) h^4{}_2 S f_{42} \rangle$
$\langle a_5 b_1 S^{-1} a_3, h^4{}_1 \rangle \langle a_2, f_5 \rangle$
$\langle a_4, S^{-1} h_1 \rangle \langle a_6, S f_7 \rangle \langle b_2, S f_6 \rangle$
$f_3 h^3$ $h^4{}_3 S f_{41}$ $f_2 h^2$ s_0 s_1 $h^6{}_2 h_3$ $h^1{}_1 S^{-1} f_{12}$ $h^5{}_1$
$= \langle a, f_1(Sh^1{}_2)(Sh^5{}_2) h^6{}_1 h^4{}_1 S f_7 \rangle$
$\langle b, h^4{}_2 S f_6 \rangle$
$f_4 h^3$ $h^4{}_3 S f_5$ $f_3 h^2$ s_0 s_1 $h^6{}_2 h$ $h^1{}_1 S^{-1} f_2$ $h^5{}_1$
$= T^b_{\tau_2} \circ L^{S^{-1}h}_{\tau_1^*} \circ (a \otimes f) \triangleright_{s_1}$
h^3 h^4 h^2 s_0 h^6 s_1 h^1 h^5

Figure 4.8: Proof of covariance of the 1st elementary open ribbon (a) from the left using ${}^{(-)}L$ (b) from the right using ${}^{(+)}L$.

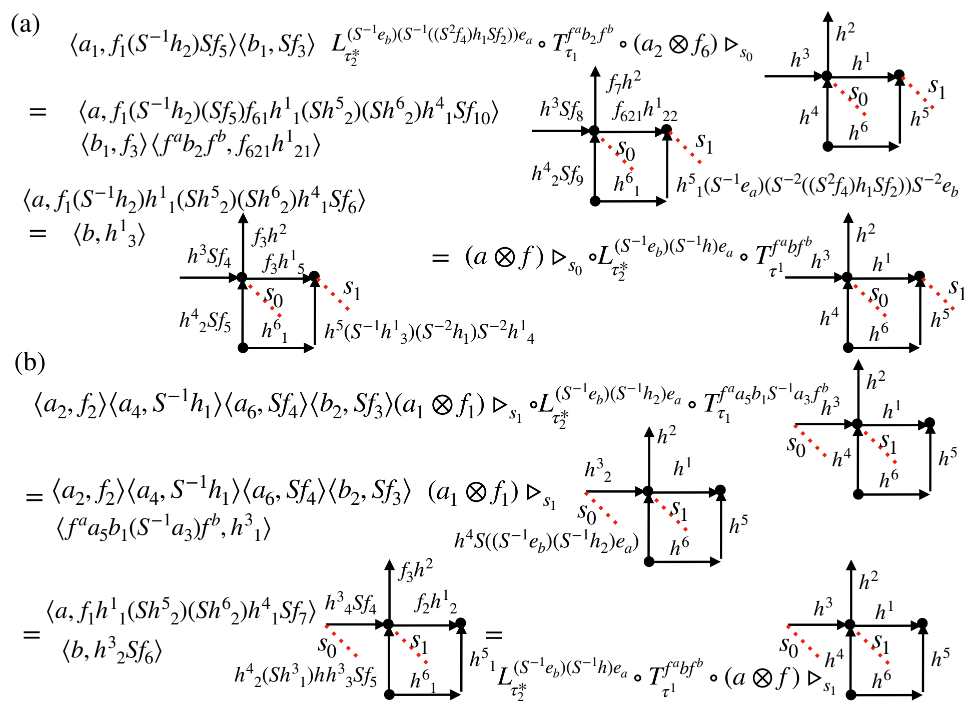


Figure 4.9: Proof of covariance of the 2nd elementary open ribbon (a) from the left using ${}^{(-)}L$ (b) from the right using ${}^{(+)}L$.

Proposition 4.3.8. Let ξ be a strongly open ribbon from site s_0 to site s_1. Then $\tilde F_\xi : D(H)^* \to \mathrm{End}(\mathcal{H})$ defined iteratively is a left and right module map under $D(H)$, where $D(H)$ acts on itself by (4.24). The actions on $\mathrm{End}(\mathcal{H})$ are as before according to $\triangleright_{s_0}$ and $\triangleright_{s_1}$. Moreover, if Λ and Λ^* are cocommutative then $\tilde F_\xi$ commutes with $A(t)$ and $B(t)$ for all sites t disjoint from s_0, s_1.

Proof. The left and right module map properties to be proven are equivalent to

$$d\triangleright_{s_0} \circ \tilde F_\xi^{\phi} = \langle Sd_1, \phi_1\rangle \tilde F_\xi^{\phi_2} \circ d_2\triangleright_{s_0}, \quad \tilde F_\xi^{\phi} \circ d\triangleright_{s_1} = d_1\triangleright_{s_1} \circ \tilde F_\xi^{\phi_1}\langle Sd_2, \phi_2\rangle \tag{4.31}$$

for $d \in D(H), \phi \in D(H)^*$, which using (4.28) and the antipode of $D(H)$ come down to

$$(a\otimes f)\triangleright_{s_0} \circ \tilde F^{h\otimes b} = \langle a_1, f_1(S^{-1}h_2)Sf_5\rangle\langle b_1, Sf_3\rangle \tilde F_\xi^{(S^2 f_4)h_1 Sf_2\otimes b_2} \circ (a_2\otimes f_6)\triangleright_{s_0}$$

$$\tilde F_\xi^{h\otimes b} \circ (a\otimes f)\triangleright_{s_1} = \langle a_2, f_2\rangle\langle a_4, S^{-1}h_1\rangle\langle a_6, Sf_4\rangle\langle b_2, Sf_3\rangle (a_1\otimes f_1)\triangleright_{s_1} \circ \tilde F_\xi^{h_2\otimes a_5 b_1 S^{-1}a_3}$$

(i) The elementary open ribbon operators

$${}^{(\pm)}\tilde F^{h\otimes b}_{\tau_2\circ\tau_1^*} = (\tilde F_{\tau_2}\circ \tilde F_{\tau_1^*})(h\otimes b) = T^b_{\tau_2} \circ {}^{(\pm)}L^{S^{-1}h}_{\tau_1^*}$$

$${}^{(\pm)}\tilde F^{h\otimes b}_{\tau_2^*\circ\tau_1} = (\tilde F_{\tau_2^*}\circ \tilde F_{\tau_1})(h\otimes b) = \sum_{a,b} {}^{(\pm)}L^{(S^{-1}e_b)(S^{-1}h)e_a}_{\tau_2^*} \circ T^{f^a b f^b}_{\tau_1}$$

obey the s_0 (left) module condition for the $(-)$ case and the s_1 (right) module condition for the $(+)$ case. These are shown for a sample orientation in Figures 4.8 and 4.9. In the latter, we use the Hopf algebra duality axioms to identify $\langle f^a, \ \rangle\ \langle f^b, \ \rangle$ and transfer the other sides to e_a, e_b respectively, then apply cancellations.

While these elementary ribbons are the smallest open ribbons, not every open ribbon can be generated iteratively starting from one of these elementary ribbons. Any open ribbon can be generated beginning from a rotation around a vertex or face, followed by extension to further sites. However, we can just replace the first ${}^{\pm}L_{\tau^*}^{S^{-1}h}$ in the equation above with the appropriate convolution of L operators, and the same for the T case. The left and right module properties will be preserved for the $(+)$ and $(-)$ cases respectively.

(ii) We proceed by induction. Let $\xi : s_0 \to s_2$ be a strongly open ribbon. First observe that if $\xi' : s_0 \to s_1$ and $\xi'' : s_1 \to s_2$ and $\tilde{F}_{\xi'}$ is a left module map with respect to its start then

$$(\tilde{F}_{\xi''} \circ \tilde{F}_{\xi'})^{d \triangleright \phi} = \langle Sd, \phi_1 \rangle \tilde{F}_{\xi''}^{\phi_{22}} \circ \tilde{F}_{\xi'}^{\phi_{21}} = \langle Sd, \phi_{11} \rangle \tilde{F}_{\xi''}^{\phi_2} \circ \tilde{F}_{\xi'}^{\phi_{12}} = \tilde{F}_{\xi''}^{\phi_2} \circ d_1 \triangleright_{s_0} \circ \tilde{F}_{\xi'}^{\phi_1} \circ Sd_2 \triangleright_{s_0}.$$

Now, ξ'' is disjoint from s_0 as ξ is strongly open. Hence, $\tilde{F}_{\xi''}$ commutes with $\triangleright_{s_0}$, and so $\tilde{F}_{\xi}$ is a left module map with respect to its start.

Similarly, if $\tilde{F}_{\xi''}$ is a right module map with respect to its end then

$$(\tilde{F}_{\xi''} \circ \tilde{F}_{\xi'})^{\phi \triangleleft d} = \langle Sd, \phi_2 \rangle \tilde{F}_{\xi''}^{\phi_{12}} \circ \tilde{F}_{\xi'}^{\phi_{11}} = \langle Sd, \phi_{22} \rangle \tilde{F}_{\xi''}^{\phi_{21}} \circ \tilde{F}_{\xi'}^{\phi_1} = Sd_1 \triangleright_{s_2} \tilde{F}_{\xi''}^{\phi_2} \circ d_2 \triangleright_{s_2} \circ \tilde{F}_{\xi'}^{\phi_1}.$$

As before, ξ' is disjoint from s_2, as ξ is strongly open. Hence, $\tilde{F}_{\xi'}$ commutes with $\triangleright_{s_2}$ and $\tilde{F}_{\xi}$ is a right module map with respect to its start.

Now suppose that the left and right module property holds for strongly open ribbons up to some number of triangles in length. Let ξ be a strongly open ribbon that is not an elementary one from part (i). Then (a) we write $\xi'' \circ \xi'$ where $\xi'' = \tau$ or τ^* and ξ' is also a strongly open ribbon. In that case our first observation applies and $\tilde{F}_{\xi}$ is a left module map. And (b) we can write it as $\xi'' \circ \xi'$ where $\xi' = \tau$ or τ^* and now ξ'' is a strongly open ribbon. Then $\tilde{F}_{\xi}$ is also a right module map, hence a left and right module map.

(iii) We also note that if $\tilde{F}_{\xi''}$ is a left module map and $\tilde{F}_{\xi'}$ a right one then

$$\begin{aligned} d \triangleright_{s_1} \circ (\tilde{F}_{\xi''} \circ \tilde{F}_{\xi'})^{\phi} &= d \triangleright_{s_1} \circ \tilde{F}_{\xi''}^{\phi_2} \circ \tilde{F}_{\xi'}^{\phi_1} = \langle Sd_1, \phi_{21} \rangle \tilde{F}_{\xi''}^{\phi_{22}} \circ d_2 \triangleright_{s_1} \circ \tilde{F}^{\phi_1} \\ &= \tilde{F}_{\xi''}^{\phi_2} \circ \langle Sd_1, \phi_{12} \rangle d_2 \triangleright_{s_1} \circ \tilde{F}^{\phi_{11}} = \tilde{F}_{\xi''}^{\phi_2} \circ \langle Sd_2, \phi_{12} \rangle d_1 \triangleright_{s_1} \circ \tilde{F}^{\phi_{11}} \\ &= \tilde{F}_{\xi''}^{\phi_2} \circ \tilde{F}_{\xi'}^{\phi_1} \circ d \triangleright_{s_1} = (\tilde{F}_{\xi''} \circ \tilde{F}_{\xi'})^{\phi} \circ d \triangleright_{s_1} \end{aligned}$$

provided for the 4th equality we have d cocommutative in the sense $d_1 \otimes d_2 = d_2 \otimes d_1$. As ξ'' and ξ' are both strongly open, this implies that the action by such elements d at interior sites commute with $\tilde{F}_{\xi}$. ■

Observe that this argument holds because the ribbon ξ is strongly open: all subribbons of ξ are either themselves (strongly) open or are rotations around a vertex/face. A ribbon which is open but not strongly open may have subribbons which are not open but have interior sites disjoint from the endpoints, and therefore the above inductive argument breaks down. In the semisimple case, the condition of *strongly* open in Proposition 4.3.8 can be relaxed to just open, as we have topological invariance and so any subribbons which are not open may be smoothly deformed to their shortest path, which will be a rotation with no interior sites disjoint from the endpoints. We also know in the semisimple case that Λ, Λ^* are cocommutative, so the last part of the proposition applies.

The left and right module property and convolution of strongly open ribbons can also be viewed as follows in terms of $D = D(H)$. Recall that two left actions of D on $\mathcal{H}$, as was the case above using the site actions at the ends of open ribbons, induce left and right D-module structures on $A = \mathrm{End}(\mathcal{H})$ as in (4.27) which are compatible with the product (i.e. A is a left and right D-module algebra). They commute (i.e. make A into bimodule) if the end sites are far enough apart. Also recall that D acts on D^* by (4.26) and on itself by the product, so that D^*, D are D-bimodules. We will use a compact notation where $\check{F} = \check{F}^1 \otimes \check{F}^2$ (summation understood) denotes an element of a tensor product over the field and $\check{F}_{21} = \check{F}^2 \otimes \check{F}^1$.

Lemma 4.3.9. Let D be a finite-dimensional Hopf algebra and A a left and right D-module algebra with actions denoted by dot. We let $\{e_\alpha\}$ be a basis of D and $\{f^\alpha\}$ a dual basis. The following are equivalent.

1. $\tilde{F} : D^* \to A$ is left and right D-module map.
2. $\check{F} := \check{F}^1 \otimes \check{F}^2 = \sum_\alpha S^{-1}e_\alpha \otimes \tilde{F}^{f^\alpha} \in D \otimes A$ obeys $d\check{F} = \check{F}.d$ and $d.\check{F}_{21} = \check{F}_{21}d$ for all $d \in D$, using the product or action of D on the adjacent factor.
3. $(S \otimes \mathrm{id})\check{F}$ is invariant under the left and right tensor product D-actions.

If A is an algebra and $\tilde{F}'', \tilde{F}' : D^* \to A$ are linear maps, their convolution product $(\tilde{F}'' \circ \tilde{F}')^\phi = \tilde{F}''^{\phi_2} \circ \tilde{F}'^{\phi_1}$ is equivalent to the product of the corresponding $\check{F}'', \check{F}'$ in the tensor product algebra. If A is a bimodule and $\tilde{F}$ a bimodule map then $f = \check{F}^1.\check{F}^2, g = \check{F}^2.\check{F}^1 \in A$ are in the bimodule centre.

Proof. This is elementary but we give some details for completeness. First, as linear maps it is obvious that $\tilde{F} : D^* \to A$ is equivalent to an element $e_\alpha \otimes \tilde{F}^\alpha \in D \otimes A$ (summation over repeated labels understood). It is also clear that if $\tilde{F}'', \tilde{F}'$ are two such linear maps then $e_\alpha \otimes (\tilde{F}'' \circ \tilde{F}')^{f^\alpha} = e_\alpha \otimes \tilde{F}''^{f^\alpha{}_2} F'^{f^\alpha{}_1} = e_\beta e_\alpha \otimes \tilde{F}^{f^\alpha} F^{f^\beta}$ which is the product in $D^{op} \otimes A$. The S^{-1} means that the corresponding $\check{F}'', \check{F}'$ multiply in $D \otimes A$.

Moreover, suppose $\tilde{F}$ is a left and write D-module map. We denote the left and right tensor product actions of D on $D \otimes A$ by $\triangleright$, $\triangleleft$. Then

$$\begin{aligned} d \triangleright (e_\alpha \otimes \tilde{F}^{f^\alpha}) &= d_1 e_a \otimes d_2.\tilde{F}^{f^a} = d_1 e_a \otimes \tilde{F}^{d_2 \triangleright f^a} = d_1 e_\alpha \otimes \langle Sd_1, f^\alpha{}_1 \rangle \tilde{F}^{f^\alpha{}_2} \\ &= d_1 e_\alpha e_\beta \otimes \langle Sd_1, f^\alpha \rangle \tilde{F}^{f^\beta} = d_1 (Sd_2) e_\beta \otimes \tilde{F}^{e^\beta} = \epsilon(d)(e_\alpha \otimes F^{f^\alpha}) \end{aligned}$$

and similarly from the other side for the right action $\triangleleft$. This argument can be reversed to prove equivalence of (1) and (3). Similarly,

$$S^{-1}e_\alpha \otimes d.\tilde{F}^\alpha = S^{-1}e_\alpha \otimes \langle Sd, f^\alpha{}_1 \rangle \tilde{F}^{f^\alpha{}_2} = (S^{-1}e_\beta)(S^{-1}e_\alpha) \otimes \langle Sd, f^\alpha \rangle \tilde{F}^{f^\beta} = (S^{-1}e_\beta) d \otimes \tilde{F}^\beta$$
$$S^{-1}e_\alpha \otimes \tilde{F}^\alpha.d = S^{-1}e_\alpha \otimes \langle Sd, f^\alpha{}_2 \rangle \tilde{F}^{f^\alpha{}_1} = (S^{-1}e_\beta)(S^{-1}e_\alpha) \otimes \langle Sd, f^\beta \rangle \tilde{F}^{f^\alpha} = d(S^{-1}e_\alpha) \otimes \tilde{F}^\alpha$$

which can be reversed for the equivalence of (1) and (2). Then in the bimodule case, $d.f = d.(\check{F}^1.\check{F}^2) = \check{F}^1.(\check{F}^2.d) = f.d$ and $g.d = (\check{F}^2.\check{F}^1).d = (d.\check{F}^2).\check{F}^1 = d.g$ for all $d \in D$. ■

This is relevant to us in the case where $\tilde{F} = \tilde{F}_\xi$ for an open ribbon from s_0 to s_1 and $A = \mathrm{End}(\mathcal{H})$ with left and right module structures induced by the site actions on $\mathcal{H}$ at s_0, s_1 of $D = D(H)$. We write ${}_{s_0}A_{s_1}$ for the algebra A with this left and right D-module structure. For example, the associated $f = f_\xi$ and $g = g_\xi$ are in here. Moreover, if $\xi = \xi'' \circ \xi'$ then $\check{F}' \in D \otimes {}_{s_0}A_{s_1}$ and $\check{F}'' \in D \otimes {}_{s_1}A_{s_2}$ while $\check{F}''\check{F}' \in D \otimes {}_{s_0}A_{s_2}$. This gives a functor from the 'ribbon path groupoid' to the category of D-modules $\mathcal{H}$ with morphisms given by elements of $D \otimes A$. The composition of morphisms is given by the tensor product algebra $D \otimes A$ plus an assignment of the left and right D-module structures on A for the result. This is such that the product of ${}_{s_1}A_{s_2}$ and ${}_{s_0}A_{s_1}$ is deemed to lie in ${}_{s_0}A_{s_2}$. The morphisms that arise from open ribbons also obey the centrality properties (2). This is in the spirit of the 'holonomy' point of view in [Meu17].

4.3.3 Quasiparticle spaces for $D(H)$ ribbons

Finally, we fix a vaccum state $|\mathrm{vac}\rangle$ and consider quasiparticle spaces

$$\mathcal{L}_\xi(s_0,s_1)=\{\tilde{F}_\xi^\phi|\mathrm{vac}\rangle \mid \phi\in D(H)^*\}\subset\mathcal{H}$$

much as before, where $\xi : s_0\to s_1$ is a fixed strongly open ribbon. We make $\mathcal{H}$ a left and right $D(H)$-module where d acts from the left by $d\triangleright_{s_0}\psi$ and from the right by $\psi\triangleleft_{s_1}d := Sd\triangleright_{s_1}\psi$. These commute on $\mathcal{H}$ so that we have a bimodule when s_0,s_1 are sufficiently far apart, meaning that p_0 and p_1 do not share an edge, and neither do v_0 and v_1. However the next proposition shows that they always commute when we restrict to $\psi\in\mathcal{L}_\xi(s_0,s_1)$. We moreover dualise the left and right actions to respectively right and left actions on $\mathcal{L}_\xi(s_0,s_1)^*$ which then also form a bimodule. Recall that $D(H)^*$ is always a $D(H)$-bimodule by (4.26) and $D(H)$ a $D(H)$-bimodule by left and right multiplication.

Proposition 4.3.10. Let $\xi : s_0\to s_1$ be a strongly open ribbon and $\mathcal{L}_\xi(s_0,s_1)$ as above. This is a bimodule and

1. $D(H)^*\twoheadrightarrow\mathcal{L}_\xi(s_0,s_1)$ sending $\phi\mapsto\tilde{F}_\xi^\phi|\mathrm{vac}\rangle$ is a bimodule map.
2. $\mathcal{L}_\xi(s_0,s_1)^*\hookrightarrow D(H)$ sending $\langle\Phi|\mapsto\check{F}^1\langle\Phi|\check{F}^2|\mathrm{vac}\rangle$ is a bimodule map.

Proof. If $\Lambda\in H$ and $\Lambda^*\in H^*$ are integral elements then $\Lambda_D:=\Lambda^*\otimes\Lambda\in D(H)$ is an integral element in $D(H)$ and if $|\mathrm{vac}\rangle\in\mathcal{H}_{vac}$ then $\Lambda_D\triangleright|\mathrm{vac}\rangle=|\mathrm{vac}\rangle$ at any site. It follows that if $d\in D=D(H)$ then $d\triangleright|\mathrm{vac}\rangle=d\Lambda_D\triangleright|\mathrm{vac}\rangle=\epsilon(d)\Lambda_D\triangleright|\mathrm{vac}\rangle=\epsilon(d)|\mathrm{vac}\rangle$ as we have seen before. Then

$$d\triangleright_{s_0}\circ\tilde{F}_\xi^\phi|\mathrm{vac}\rangle=\langle Sd_1,\phi_1\rangle\tilde{F}_\xi^{\phi_2}\circ d_2\triangleright_{s_0}|\mathrm{vac}\rangle=\langle Sd,\phi_1\rangle\tilde{F}^{\phi_2}|\mathrm{vac}\rangle=\tilde{F}^{d\triangleright\phi}|\mathrm{vac}\rangle$$
$$Sd\triangleright_{s_1}\circ\tilde{F}^\phi|\mathrm{vac}\rangle=Sd_1\triangleright_{s_1}\tilde{F}_\xi^\phi\circ d_2\triangleright_{s_1}|\mathrm{vac}\rangle=(Sd_1)d_2\triangleright_{s_1}\circ\tilde{F}_\xi^{\phi_1}|\mathrm{vac}\rangle\langle Sd_3,\phi_2\rangle=\tilde{F}_\xi^{\phi\triangleleft d}|\mathrm{vac}\rangle$$

which implies that $\mathcal{L}_\xi(s_0,s_1)$ is a bimodule and proves (1). Moreover, we can unpack the centrality in Lemma 4.3.9 explicitly and apply it as

$$d\check{F}_\xi|\mathrm{vac}\rangle=\check{F}^1\otimes(\check{F}^2)\triangleleft_{s_1}d|\mathrm{vac}\rangle=\check{F}^1\otimes Sd_{(1)}\triangleright_{s_1}\circ\check{F}^2\circ d_2\triangleright_{s_1}|\mathrm{vac}\rangle=\check{F}^1\otimes Sd\triangleright_{s_1}\circ\check{F}^2|\mathrm{vac}\rangle.$$

$$\check{F}^1d\otimes\check{F}^2|\mathrm{vac}\rangle=\check{F}^1\otimes d\triangleright(\check{F}^2)|\mathrm{vac}\rangle=\check{F}^1\otimes d_1\triangleright_{s_0}\circ\check{F}^2\circ Sd_2\triangleright_{s_0}|\mathrm{vac}\rangle=\check{F}^1\otimes d\triangleright_{s_0}\circ\check{F}^2|\mathrm{vac}\rangle$$

so that

$$d\check{F}^1\langle\Phi|\check{F}^2|\mathrm{vac}\rangle=\check{F}^1\langle\Phi|Sd\triangleright_{s_1}\circ\check{F}^2|\mathrm{vac}\rangle=\check{F}^1\langle d\triangleright\Phi|\check{F}^2|\mathrm{vac}\rangle$$
$$\check{F}^1\langle\Phi|\check{F}^2|\mathrm{vac}\rangle d=\check{F}^1\langle\Phi|d\triangleright_{s_0}\circ\check{F}^2|\mathrm{vac}\rangle=\check{F}^1\langle\Phi\triangleleft d|\check{F}^2|\mathrm{vac}\rangle$$

which is (2). Here the left action on $\mathcal{L}_\xi(s_0,s_1)$ dualises to the right action $(\Phi\triangleleft d)(\psi)=\Phi(d\triangleright_{s_0}\psi)$ and the right action on $\mathcal{L}_\xi(s_0,s_1)$ dualises to the left action $(d\triangleright\Phi)(\psi)=\Phi(\psi\triangleleft_{s_1}d)=\Phi(Sd\triangleright_{s_1}\psi)$. ∎

The maps in the proposition are expected to be isomorphisms in line with Proposition 4.2.10 for the $D(G)$ case, but this requires more proof. For example, this follows if $\tilde{F}_\xi^\phi|\mathrm{vac}\rangle=0$ implies that $\phi=0$, which is expected to follow from unitarity properties with respect to a $*$-structure. Likewise, it is expected that $\mathcal{L}_\xi(s_0,s_1)$ is independent of ξ at least in the H semisimple case and characterised in terms of $A(t),B(t)$ in the manner that was done in Proposition 4.2.10.

4.4 Concluding remarks

We have given a self-contained treatment of the Kitaev model for a finite group G, focussed on the quasiparticle content and ribbon equivariance properties expressed in terms of the quantum double $D(G)$. This was largely avoided in works such as [Kit03, BM-D08], while [BSW11] starts to take a quantum double view, and we built on this. As well as a systematic treatment of the core of the theory, we have then demonstrated how quasiparticles could be created and manipulated in practice, with details in the case of $D(S_3)$ of the construction of logical operations and gates. We also showed the existence of a 'Bell state' that exists in the ribbon space $\mathcal{L}(s_0, s_1)$ created by ribbon operations for an open ribbon between s_0, s_1 and which can be used to teleport quasiparticle information between the endpoints. We also illustrated these ideas for the Abelian case of $D(\mathbb{Z}_n)$.

Beyond this practical side, we also looked closely as the obstruction to generalising such models to the 'quantum case' where the group algebra $\mathbb{C}G$ is replaced by a finite-dimensional Hopf algebra H. That this works when $S^2 = \mathrm{id}$ (e.g. the Hopf algebra is semisimple and we work over a field of characteristic zero such as $\mathbb{C}$) is well known as are its link to topological invariants [BK12, Meu17] such as the Turaev-Viro invariant and the Kuperberg invariant[Kup91]. As far as we can tell, ribbon operators at this level have not been studied very explicitly, athough contained in principle in [Meu17] as part of a theory of 'holonomy', following the work of [BMCA13] for the site operations. As well as our own work we have noted [YCC22]. We provided a self-contained treatment of the core properties in the $S^2 = \mathrm{id}$ case but we could also see by giving direct proofs what is involved in the general case. We found that site operation work perfectly well but we must use S^{-1} in certain key places. To be concrete, we put this on the vertex side but this complication can be put in different places leading in fact to a set of possible site operations all forming representations of $D(H)$. Dual triangle and ribbon equivariance properties then become more complicated with ${}^{-}L$ needed in some places for good behaviour with respect to the initial site action s_0. We also noted that the Peter-Weyl decomposition whereby $D(H)$ is a direct sum of endomorphism spaces for the irreps holds when $S^2 = \mathrm{id}$. More generally, one will have some blocks associated to irreps but these will not be the whole story. Hence our ideas on ribbon teleportation will be more complicated in general. Likewise, the actions of the integrals $A(v,p)$ and $B(v,p)$ are no longer projectors in the nonsemisimple case, but square to zero, which considerably changes how the physics should be approached and requires further work. It will also be necessary to look at $*$-structures needed to formulate unitarity at this level, possibly using the notion of flip Hopf $*$-algebras as recently initiated for ZX calculus in [Maj21].

Nevertheless, there are good reasons to persist with the general case, namely in order to link up with 2+1 quantum gravity and the Turaev-Viro invariant of 3-manifolds in a graph version. In quantum gravity, the relevant 3-manifold would be $\mathbb{R} \times \Sigma$ where $\mathbb{R}$ is time and Σ is a surface with marked points, but we would make a discrete approximation of the latter by a (ciliated, ribbon) graph, or in the simplest case a square lattice as here. Since the Turaev-Viro invariant is based on $D(u_q(sl_2))$, the goal would be to have a more Kitaev model point of view in contrast to current Hamiltonian constructions[AGS96]. Going the other way, it would be interesting to try to regard the $D(S_3)$ model as leading to a baby version of quantum gravity in the context of discrete noncommutative geometry[Maj04].

Another longer term motivation for the current work is the need for some kind of compiler or 'functor' from surface code models such as the Kitaev one to ZX-calculus[CD11] as more widely used in quantum computing. The Z,X here are Fourier dual and it would be useful to understand even in the Abelian case of $D(\mathbb{Z}_n) \cong \mathbb{C}\mathbb{Z}_n^2$ how the surface code theory relates to ZX calculus based on $\mathbb{C}\mathbb{Z}_n$ as a quasispecial Frobenius algebra. There are current ideas about this but they appear to require a notion of boundary defects. This and the notion of condensates will both need to be studied more systematically by the methods in the present work, building on current literature such as [BSW11]. We also note that in topology, the Jones invariant and its underlying Chern-Simons theory are based on the quantum group $u_q(sl_2)$ and such invariants are related via surgery on the knot to the Turaev-Viro invariant based on $D(u_q(sl_2))$, suggesting the possibility of a general link between $D(H)$ surface code theory and ZX calculus on H. The latter on general Hopf algebras and braided-Hopf algebras was recently studied in [CD19, Maj21]. These are some directions for further work.

Chapter 5

Qudit lattice surgery

This is a short Chapter, which will give us a warm-up to the more complicated version of boundaries and surgery in Kitaev models more generally. Throughout this Chapter, we let $\mathbb{Z}_d$ be the cyclic group with d elements labelled by integers $0, \cdots, d-1$ with addition as group multiplication. We assume $d \geq 2$, as the $d = 1$ case is trivial. We occasionally ignore normalisation (typically factors of d or $\frac{1}{d}$) when convenient. We have patches of Kitaev models where $G = \mathbb{Z}_d$, and where the boundaries are described in the same manner as in Section 1.6, but the Paulis are qudit versions instead.

One can see that because the underlying group is Abelian, the qudit surface codes are CSS, and we could in principle treat this homologically. However, it leads into the case where the group is nonAbelian in Chapter 6, which cannot be seen as a CSS code. We now give a series of definitions for the $\mathbb{Z}_d$ quantum double model, which recaps those given in Section 4.1 but geared towards the application here.

Definition 5.0.1. Let $\mathbb{C}\mathbb{Z}_d$ be the group Hopf algebra with basis states $|i\rangle$ for $i \in \mathbb{Z}_d$. $\mathbb{C}\mathbb{Z}_d$ has multiplication given by a linear extension of its native group multiplication, so $|i\rangle \otimes |j\rangle \mapsto |i+j\rangle$, and the unit $|0\rangle$. It has comultiplication given by $|i\rangle \mapsto |i\rangle \otimes |i\rangle$, and the counit $|i\rangle \mapsto 1 \in \mathbb{C}$. It has the normalised integral element $\Lambda_{\mathbb{C}\mathbb{Z}_d} = \frac{1}{d}\sum_i |i\rangle$ and the antipode is the group inverse. $\mathbb{C}\mathbb{Z}_d$ is commutative and cocommutative.

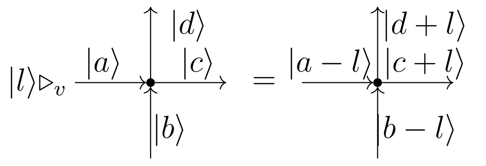
Definition 5.0.2. Let $\mathbb{C}(\mathbb{Z}_d)$ be the function Hopf algebra with basis states $|\delta_i\rangle$ for $i \in \mathbb{Z}_d$. $\mathbb{C}(\mathbb{Z}_d)$ is the dual algebra to $\mathbb{C}\mathbb{Z}_d$. $\mathbb{C}(\mathbb{Z}_d)$ has multiplication $|\delta_i\rangle \otimes |\delta_j\rangle \mapsto \delta_{i,j}|\delta_i\rangle$ and the unit $\sum_i |\delta_i\rangle$. It has comultiplication $|\delta_i\rangle \mapsto \sum_{h\in\mathbb{Z}_d} |\delta_h\rangle \otimes |\delta_{i-h}\rangle$ and counit $|\delta_i\rangle \mapsto \delta_{i,0}$. It has the normalised integral element $\Lambda_{\mathbb{C}(\mathbb{Z}_d)} = |\delta_0\rangle$ and the antipode is also the inverse. $\mathbb{C}(\mathbb{Z}_d)$ is commutative and cocommutative.

Lemma 5.0.3. The algebras are related by the Fourier isomorphism, so $\mathbb{C}(\mathbb{Z}_d) \cong \mathbb{C}\mathbb{Z}_d$ as Hopf algebras. In particular this isomorphism has maps

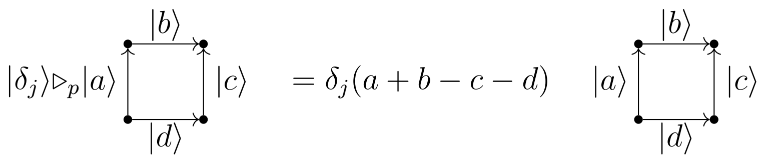
$$|j\rangle \mapsto \sum_k q^{jk}|\delta_k\rangle, \quad |\delta_j\rangle \mapsto \frac{1}{d}\sum_k q^{-jk}|k\rangle, \tag{5.1}$$

where $q = e^{\frac{i2\pi}{d}}$ is a primitive dth root of unity.

Definition 5.0.4. Now let $\Sigma = \Sigma(V, E, P)$ be a square lattice viewed as a directed graph with its usual (cartesian) orientation. The corresponding Hilbert space $\mathcal{H}$ will be a tensor product of vector spaces with one copy of $\mathbb{C}\mathbb{Z}_d$ at each arrow in E, with basis denoted by $\{|i\rangle\}_{i\in\mathbb{Z}_d}$ as before. Next, for each vertex $v \in V$ and each face $p \in P$ we define an action of $\mathbb{C}\mathbb{Z}_d$ and $\mathbb{C}(\mathbb{Z}_d)$, which acts on the vector spaces around the vertex or around the face, and trivially elsewhere, according to



and

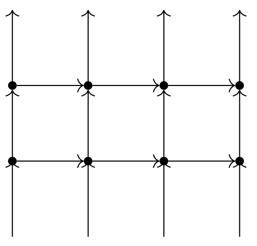


for $|l\rangle \in \mathbb{C}\mathbb{Z}_d$ and $|\delta_j\rangle \in \mathbb{C}(\mathbb{Z}_d)$.

Here $|l\rangle{\triangleright_v}$ subtracts in the case of arrows pointing towards the vertex and $|\delta_j\rangle{\triangleright_p}$ has c, d entering negatively in the δ-function because these are contra to a *clockwise* flow around the face in our conventions. The vertex actions are built from four-fold copies of the operator X and $X^\dagger$, where $X^l|i\rangle = |i+l\rangle$. Consider the face actions of elements $\sum_j q^{mj}|\delta_j\rangle$, i.e. the Fourier transformed basis of $\mathbb{C}(\mathbb{Z}_d)$; these face actions are made up of Z and $Z^\dagger$, where $Z^m|i\rangle = q^{mi}|i\rangle$, and the Z, X obey $ZX = qXZ$.

Stabilisers on the lattice are given by measurements of the $X \otimes X \otimes X^\dagger \otimes X^\dagger$ and $Z \otimes Z \otimes Z^\dagger \otimes Z^\dagger$ operators on vertices and faces respectively; that is, for the vertices we non-deterministically perform one of the d projectors $P_v(j) = \sum_k q^{jk}|k\rangle{\triangleright_v}$ for $j \in \mathbb{Z}_d$, according to each of the d measurement outcomes. Similarly for faces, we perform one of the d projectors $P_p(j) = |\delta_j\rangle{\triangleright_p}$. In practice, this requires additional 'syndrome' qudits at each vertex and face. At each round of measurement, we measure all of the stabilisers on the whole lattice.

For a whole patch, including boundaries, we have



where the boundaries mean that some vertex and face actions are missing incident edges. It is easy to compute that a patch of this shape has $\dim \mathcal{L} = d$, the same dimension as the data qudits. Thus we can confer the bases of $\mathbb{C}\mathbb{Z}_d$ and $\mathbb{C}(\mathbb{Z}_d)$ upon it, setting $|0\rangle_L$ as the state where all data qudits are initialised in the $|0\rangle$ state and so on.

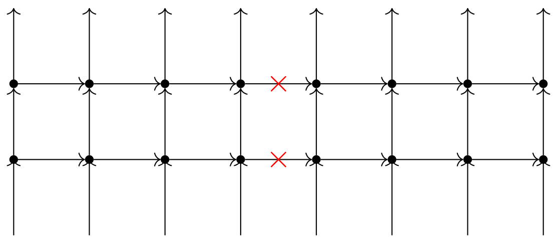
5.1 Lattice surgery

If we have two patches with logical spaces $(\mathcal{H}_{vac})_1$ and $(\mathcal{H}_{vac})_2$ which are disjoint in space then we evidently have a combined logical space $\mathcal{H}_{vac} = (\mathcal{H}_{vac})_1 \otimes (\mathcal{H}_{vac})_2$.

We may start with one patch and 'split' it to convert it into two patches.

5.1.1 Splits

To perform a smooth split, take a patch and measure out a string of intermediate qudits from top to bottom in the $\{|\delta_i\rangle\}$ basis, like so:



Regardless of the measurement results we get, we now have two disjoint patches next to each other. We can see the effect on the logical state by considering an X-type string operator which had been extending across a string ξ from left to right on the original patch. Previously it had been ${}_xF^i_\xi$, say. Now, let $\xi = \xi'' \circ \xi'$, where ξ' extends across the left patch after the split and ξ'' extends across right one. Then ${}_xF^i_\xi = {}_xF^i_{\xi'} \circ {}_xF^i_{\xi''}$; our X^i_L gate on the original logical space is taken to $X^i_L \otimes X^i_L$ on $(\mathcal{H}_{vac})_1 \otimes (\mathcal{H}_{vac})_2$. It is easy to see that this then gives the map:

$$\Delta_s : |i\rangle_L \mapsto |i\rangle_L \otimes |i\rangle_L$$

for $i \in \mathbb{Z}_d$. This is the same regardless of the measurement outcomes on the intermediate qubits we measured out.

To perform a rough split, take a patch and measure out a string of qudits from left to right in the $\{|i\rangle\}$ basis. A similar analysis to before, but for Z^i_L gates, shows that we have

$$\Delta_r : |\delta_i\rangle_L \mapsto |\delta_i\rangle_L \otimes |\delta_i\rangle_L.$$

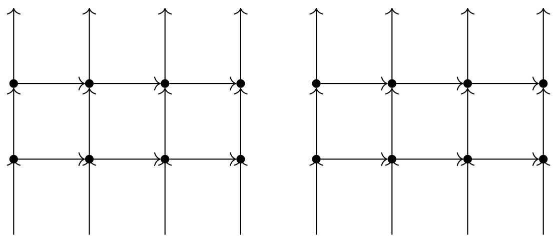
Remark 5.1.1. We now note a subtlety: for both smooth and rough splits we induce a copy in the relevant bases, that is the comultiplication of $\mathbb{C}\mathbb{Z}_d$, rather than the comultiplication of $\mathbb{C}(\mathbb{Z}_d)$ for the rough splits. This is because we are placing both algebras on the same object, using the non-natural isomorphism $V \cong V^*$ for vector spaces V. Thus if we take the rough split map in the other basis we get

$$\Delta_r : |i\rangle_L \mapsto \sum_h |h\rangle_L \otimes |i-h\rangle_L.$$

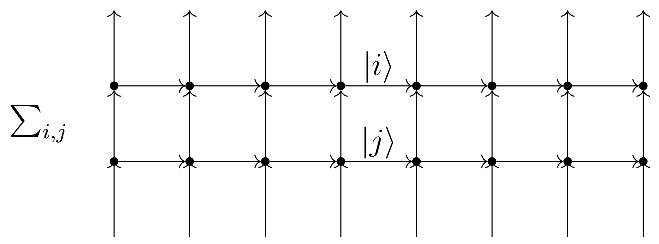
This follows directly from Lemma 5.0.3. The fact that both algebras are placed on the same object allows us to relate the model to the ZX-calculus in Section 5.2.

5.1.2 Merges

To perform a smooth merge, we do the reverse operation. Start with two disjoint patches:



and then initialise between them a string of intermediate qudits, each in the $\sum_i |i\rangle$ state, like so:



Then measure the stabilisers at all sites on the now merged lattice. Now, assuming no errors have occurred all the stabilisers are automatically satisfied everywhere except the measurements which include the new edges. These measurements realise a measurement of $Z_L \otimes Z_L$ on the logical space $(\mathcal{H}_{vac})_1 \otimes (\mathcal{H}_{vac})_2$. We prove this in Appendix 16. With merges, the resultant logical state after merging is also dependent on the measurement outcomes.

Depending on which ‘frame’ we choose we can have two different sets of possible maps from the smooth merge, see [dBH20] for the easier qubit case. Here we choose to adopt the Pauli frame of the second patch. In the Fourier basis we thus have the Kraus operators:

$$\nabla_s : \{|\delta_i\rangle_L \otimes |\delta_j\rangle_L \mapsto q^{in}|\delta_{i+j}\rangle_L\}_{n\in\{0,\cdots,d-1\}}$$

where q^{in} is a factor introduced by the $Z_L \otimes Z_L$ measurement; we have $n \in \{0, \cdots, d-1\}$ for the d different possible measurement outcomes. If we only consider the $n = 0$ case for a moment, one can come to the conclusion that this is the correct map using the Z_L logical operators:

$$\sum_j q^{ij}{}_z F_\xi^{\delta_j} \circ \sum_j q^{kj}{}_z F_\xi^{\delta_j} = \sum_j q^{(i+k)j}{}_z F_\xi^{\delta_j}$$

from earlier, where ξ extends from bottom to top on both original patches. Then when we merge the patches, we get the combined string operator. In the other basis of logical states, the smooth merge gives:

$$\nabla_s : \{|i\rangle_L \otimes |j\rangle_L \mapsto \delta_{i+n,j}|i+n\rangle_L\}_{n\in\{0,\cdots,d-1\}},$$

Remark 5.1.2. It is common in categorical quantum mechanics to consider the so-called multiplicative fragment of quantum mechanics. In this fragment, we may post-select rather than just make measurements according to the traditional postulates. As such, there is a choice of post-selection such that $n = 0$ and we acquire the multiplication of $\mathbb{C}\mathbb{Z}_d$ or $\mathbb{C}(\mathbb{Z}_d)$ depending on basis. While physically we cannot post-select, this is a useful toy model in which algebraic notions may be more conveniently related to quantum mechanical processes.

Considering the same convention of frame, a rough merge gives:

$$\nabla_r : \{|i\rangle_L \otimes |j\rangle_L \mapsto q^{in}|i+j\rangle\}_{n\in\{0,\cdots,d-1\}}$$

by a similar argument, this time performing a measurement of $X_L \otimes X_L$ to merge patches at the top and bottom.

5.1.3 Units and deletion

While we are on the subject of measurements, we can delete a patch by measuring out every qudit associated to its lattice in the Z-basis. If we do so, we obtain the maps

$$\epsilon_r : \{|i\rangle_L \mapsto \delta_{n,i}\}_{n\in\{0,\cdots,d-1\}}.$$

In the $n = 0$ outcome this is precisely the counit of $\mathbb{C}(\mathbb{Z}_d)$. We check this in Appendix 17. If we instead measure out each qudit in the X-basis we get

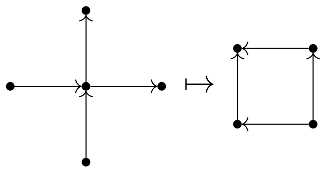
$$\epsilon_s : \{|i\rangle_L \mapsto q^{in}\}_{n\in\{0,\cdots,d-1\}},$$

where we see the counit of $\mathbb{C}\mathbb{Z}_d$.

One can clearly also construct the units of $\mathbb{C}(\mathbb{Z}_d)$ and $\mathbb{C}\mathbb{Z}_d$, being $\eta_s : \sum_i |i\rangle_L$ and $\eta_r : |0\rangle_L$ respectively. The last remaining pieces of the puzzle are the antipode and Fourier transform on the logical space.

5.1.4 Antipode

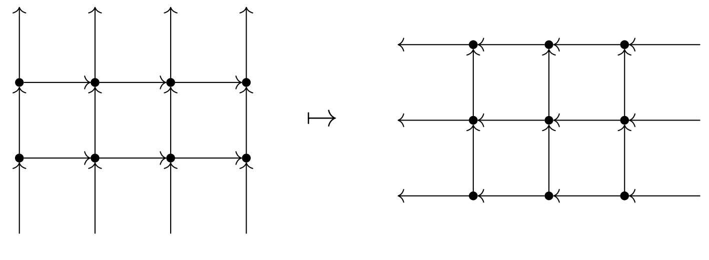
First we demonstrate how to map between the $|0\rangle_L$ and $|\delta_i\rangle_L$ states. If we apply a Fourier transform $H = \sum_{j,k} q^{-jk}|k\rangle\langle j|$ to a qudit in the state $|0\rangle$ we have $H|0\rangle = \sum_i |i\rangle$.[1] As $HX = Z^\dagger H$ (and $XH = HZ$) all $A(v)$ projectors are translated to $B(p)$ projectors by rotating the lattice to exchange vertices with faces



such that the $X, X^\dagger$ match up with $Z^\dagger, Z$ appropriately when considering the clockwise conventions from Def 5.0.4. This is just a conceptual rotation, and there does not need to be any *physical* rotation in space. Thus we have

$$H_L|0\rangle_L = (\bigotimes_E H)\prod_v A(v)\bigotimes_E |0\rangle = \prod_p B(p)\bigotimes_E \sum_i |i\rangle = |\delta_0\rangle_L$$

where $H_L = \bigotimes_E H$ is the logical Fourier transform, and the lattice has been mapped:



H_L also takes X-type string operators to Z-type string operators in the quasiparticle basis but with a sign change, and thus we have

$$H_L|i\rangle_L = H_L X^i|0\rangle_L = Z^{-i}H_L|0\rangle_L = \sum_k q^{-ik}|k\rangle_L = |\delta_i\rangle_L$$

[1]The H stands for Hadamard, which is what the qubit Fourier transform is commonly called. The qudit Fourier transform is not a Hadamard matrix in general.

so it is genuinely a Fourier transform. Applying it twice gives

$$H_L H_L|i\rangle_L = \sum_{k,l} q^{-ik}q^{-kl}|l\rangle_L = \sum_{l} \delta_{l,-i}|l\rangle_L = |-i\rangle_L$$

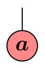
where the lattice is now as though the whole patch has been rotated in space by π by the same argument as before. This is evidently the *logical antipode*, $S_L = H_L H_L$.

This completes the set of fault-tolerant operations we may perform with the $\mathbb{C}\mathbb{Z}_d$ lattice surgery. One can create other states in a non-error corrected manner and then perform state distillation to acquire the correct state with a high probability, but this is beyond the scope of the Chapter and very similar to e.g. [FSG09].

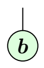
5.2 The ZX-calculus

The ZX-calculus is based on Hopf-Frobenius algebras sitting on the same object. It imports ideas from monoidal category theory to justify its graphical formalism [Sel11]. See [HV19] for an introduction from the categorical point of view. Calculations may be performed by transforming diagrams into one another, and the calculus may be thought of as a tensor network theory equipped with rewriting rules.

Here we present the syntax and semantics of ZX-diagrams for $\mathbb{C}\mathbb{Z}_d$. We are unconcerned with either universality or completeness [Bac16], and give only the necessary generators for our purposes; moreover, we adopt a slightly simplified convention. First, we have generators:

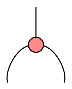


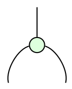
$\rightsquigarrow |a\rangle$



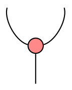
$\rightsquigarrow \sum_i q^{-ib}|i\rangle$

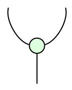
for elements, where the small red and green nodes are called ‘spiders’, and diagrams flow from bottom to top.[2] The labels associated to a spider are called phases. Then we have the multiplication maps,


$\rightsquigarrow |i\rangle \otimes |j\rangle \mapsto |i+j\rangle$

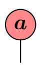

$\rightsquigarrow |i\rangle \otimes |j\rangle \mapsto \delta_{i,j}|i\rangle$

comultiplication,

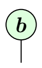

$\rightsquigarrow \quad |i\rangle \mapsto \sum_h |h\rangle \otimes |i-h\rangle$


$\rightsquigarrow \quad |i\rangle \mapsto |i\rangle \otimes |i\rangle$

maps to $\mathbb{C}$,

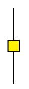


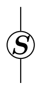
$\rightsquigarrow |i\rangle \mapsto \delta_{i,a}$



$\rightsquigarrow |i\rangle \mapsto q^{ib}$

and Fourier transform[3] plus antipode:


$\rightsquigarrow \quad |j\rangle \mapsto \sum_i q^{-ij}|i\rangle$

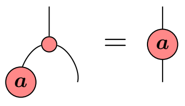


$\rightsquigarrow \quad |i\rangle \mapsto |-i\rangle$

[2]Red and green are dark and light shades in greyscale.

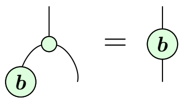
[3]The Hadamard symbol here makes it look like it is vertically reversible, i.e. $H^\dagger = H$, but it is not; this is just a notational flaw.

Now, these generators obey all the normal Hopf rules: associativity of multiplication and comultiplication, unit and counit, bialgebra and antipode laws, but that it is not all. The ZX-calculus makes use of an old result by Pareigis [Par71], which states that all finite-dimensional Hopf algebras on vector spaces automatically give two Frobenius structures, which in the present case correspond to the red and green spiders above. In this case, they are in fact so-called †-special commutative Frobenius algebras (†-SFCAs) [CPV12]. Such algebras have a normal form, such that any connected set of green or red spiders may be combined into a single green or red spider respectively, summing the phases [CD11]. This is called the *spider theorem*. As an easy example, observe that we can define the X^a gate in the ZX-calculus as:



$$\rightsquigarrow |j\rangle \mapsto |j+a\rangle$$

and similarly for a Z^b gate,



$$\rightsquigarrow |j\rangle \mapsto q^{-bj}|j\rangle \, .$$

The Fourier transform then 'changes colour' between green and red spiders. We show these axioms in Appendix 18. For a detailed exposition of the qudit ZX-calculus in greater generality see [Wan21].

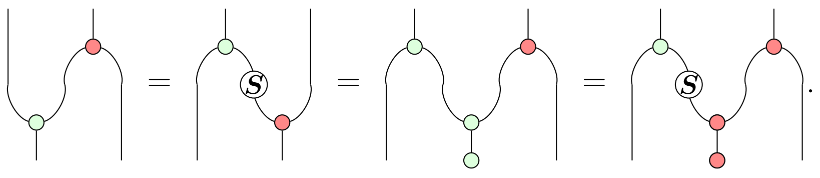
Now, one can immediately see that the generators are automatically (by virtue of the $\mathbb{C}\mathbb{Z}_d$ and $\mathbb{C}(\mathbb{Z}_d)$ structures) in bijection with the lattice surgery operations described previously. The bijection between this fragment of the ZX-calculus and lattice surgery was spotted by de Beaudrap and Horsman in the qubit case [dBH20]; however, their presentation emphasises the Frobenius structures. The algebraic explanation for the lattice surgery properties is all in the Hopf structure: in summary, it is because the string operators are Hopf-like.[4] The Frobenius structures are still useful diagrammatic reasoning tools because of the spider theorem, and also because the two interacting Frobenius algebras correspond to the rough (red spider) and smooth (green spider) operations. There is a convenient 3-dimensional visualisation for this using 'logical blocks', which we defer to Appendix 19. There we also include Table 1, which is a dictionary between lattice operations, ZX-diagrams and linear maps.

5.2.1 Gate synthesis

Using the ZX-calculus we can thus design logical protocols in a straightforward manner. We have already implicitly shown a state injection protocol, being the spider merges for the X^a and Z^b gates above, but we can go further. A common gate in the circuit model is the controlled-X (CX) gate. In qudit quantum computing this is defined as the map

$$CX : |i\rangle \otimes |j\rangle \mapsto |i\rangle \otimes |i+j\rangle$$

which in the ZX-calculus we might represent as, say,



[4]We formalise such operators as module maps in [CM22].

In the first diagram we perform a smooth split followed by a rough merge; in the second we do the opposite. In the third and fourth we first generate a maximally entangled state and then perform a smooth and rough merge on either side. The antipodes are necessary because of a minor complication with duals in the qudit ZX-calculus. Rewrites using the calculus show that these are equal, and conversions into linear maps do indeed yield the CX. We check this in Appendix 20. Note that we implicitly assumed the $n = 0$ measurement outcomes for the merges, but we assert that in this case the protocol works deterministically by applying corrections. This is a generalisation of protocols specified in [dBH20], and the correction arguments are identical.

We can also easily see that the lattice surgery operations are not universal, even with the addition of logical X_L and Z_L gates using string operators. All phases have integer values and so we cannot even achieve all single-qudit gates in the 2nd level of the Clifford hierarchy fault-tolerantly. For example, we cannot construct a $\sqrt{X}_L$ gate with the operations listed here.

With this limitation in mind, in Appendix 21 we discuss the prospects for expanding the scope of the model to other group algebras and to Hopf algebras more generally.

5.3 Conclusion

We have shown that lattice surgery is straightforward to generalise to qudits, assuming an underlying abelian group structure. The particular Abelian group here was the cyclic group $\mathbb{Z}_d$, but as any Abelian group can be factorised into a product of cyclic groups the results extend in the obvious manner. The resultant diagrammatics which can be used to describe computation are elegant, concise and powerful.

Chapter 6

Algebraic aspects of boundaries in quantum double models

The Kitaev model[Kit03] for topologically fault-tolerant quantum computing is defined by the quantum double $D(G)$ of a finite group G. The irreducible representations of this quantum group are quasiparticles corresponding to measurement outcomes at sites on a lattice, and their dynamics correspond to linear maps on the data. The lattice can be any ciliated ribbon graph embedded on a surface [Meu17], although throughout we will assume a square lattice on the plane for convenience. The topological properties of the Kitaev model derive from the 'topological order' in condensed matter terms[KZ22], which is the braided category $\mathcal{Z}(\mathcal{C})$ given by the 'dual' or 'centre' construction'[Maj91] applied to the monoidal category $\mathcal{C} = \mathcal{M}^G$ of G-graded vector spaces. This is then identified with the category ${}_{D(G)}\mathcal{M}$ of $D(G)$-modules for the explicit algebraic treatment. The Kitaev model generalises to replace G by a finite-dimensional semisimple Hopf algebra, as well as aspects that work of a general finite-dimensional Hopf algebra. We refer to [CM22] for details of the relevant algebraic aspects of this theory, which applies in the bulk of the Kitaev model.

In the present sequel, we extend from the bulk theory to a detailed study of a certain quasi-Hopf algebra $\Xi(R, K)$ that similarly governs the quasiparticle states on a boundary as its representations, as in [CCW16]. In physical terms, a gapped boundary of a Kitaev model preserves a finite energy gap between the vacuum and the lowest excited state(s), which is independent of system size. There are two equivalent views of gapped boundaries, as summarised in [JKT22, Sec 3.2]. The first is using a Lagrangian algebra L in ${}_{D(G)}\mathcal{M}$ and then constructing functors ${}_{D(G)}\mathcal{M} \to {}_L\mathcal{M}$ to describe anyon condensation, with ${}_L\mathcal{M}$ defining the boundary phase. One can also use Frobenius algebras and take idempotent completion of a relevant quotient category to acquire the boundary [CCW16]. In the second view, which is the one we take, boundary conditions are defined by module categories of the fusion category $\mathcal{C}$. By definition, a (right) $\mathcal{C}$-module means[Ost03B, KK12] a category $\mathcal{V}$ equipped with a bifunctor $\mathcal{V} \times \mathcal{C} \to \mathcal{V}$ obeying coherence equations which are a polarised version of the properties of $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ (in the same way that a right module of an algebra obeys a polarised version of the axioms for the product). For our purposes, we care about *indecomposable* module categories, that is module categories which are not equivalent to a direct sum of other module categories. Excitations on the boundary with condition $\mathcal{V}$ are then given by functors $F \in \mathrm{End}_{\mathcal{C}}(\mathcal{V})$ that commute with the $\mathcal{C}$ action[KK12], beyond the vacuum state which is the identity functor $\mathrm{id}_{\mathcal{V}}$. More than just the boundary conditions above, we care about these excitations, and so $\mathrm{End}_{\mathcal{C}}(\mathcal{V})$ is the category of interest. Finally, for the Kitaev model, indecomposable module categories for $\mathcal{C} = \mathcal{M}^G$ are classified by subgroups $K \subseteq G$ and cocycles $\alpha \in H^2(K, \mathbb{C}^\times)$ [Ost03A]. We will stick to the trivial α case here and just work with $\mathcal{V} = {}_K\mathcal{M}^G$, the G-graded

K-modules where $x \in K$ itself has grade $|x| = x \in G$. Then the excitations are governed by objects of $\mathrm{End}_{\mathcal{C}}(\mathcal{V}) \simeq {}_K\mathcal{M}^G_K$, the category of G-graded bimodules over K. This is a nontrivially monoidal category and by Tannaka-Krein arguments[Maj92] one can expect a quasi-Hopf algebra $\Xi(R, K)$ such that ${}_K\mathcal{M}^G_K \simeq {}_{\Xi(R,K)}\mathcal{M}$, the modules of $\Xi(R, K)$. Here R is a choice of transversal for $K \subseteq G$ so that every element of G factorises uniquely as RK.

This categorical derivation of $\Xi(R, K)$ is deferred to Section 6.5, while the quasi-Hopf algebra $\Xi(R, K)$ itself and its concrete application to gapped boundaries is the main focus of the Chapter. The algebraic model provides a critical bridge between explicit on the nose formulae that would be needed in any concrete implementation and the abstract categorical picture, which is more qualitative being only defined up to isomorphisms, for example up to equivalence of categories. After recapping the algebraic model for the bulk in Section 6.1 as a warm up, we study the algebra $\Xi(R, K)$ and its physical role for boundary lattice models in Section 6.2. We provide in detail the construction of its irreducible representations, their associated projections and (Proposition 6.2.8) the induction-restriction multiplicities due to an algebra inclusion $i : \Xi(R, K) \hookrightarrow D(G)$. The latter amounts to formulae for the decomposition of bulk quasiparticles into quasiparticles on the boundary, in our case directly from the algebra and not relying on the abstract categorical arguments of [Sch16]. We also demonstrate that these formulae hold explicitly on the lattice. While much of this has been studied previously [BSW11, BM-D08, CCW16], we give detailed proofs of results which have not been formally proven before to the best of our knowledge. We also correct several inaccuracies found in the literature, including in statements given without proof.

On the applications side, Section 6.3 develops the theory of lattice surgery for the Kitaev model, which to our knowledge is the first description of quantum code surgery which goes beyond stabiliser codes [HFDM12]. We give the maps on logical data and find that they are precisely the morphisms of the Hopf algebras $\mathbb{C}G$ and $\mathbb{C}(G)$ on the same space. Interestingly, this leaves open the possibility that there could be a method of universal computation with Kitaev models which does not require anyons and can be performed wholly on the vacuum space, unlike the methods of e.g. [Moc04, Moc03] which use excited states. We leave the problem of determining lattice surgery's computational power to future work.

Section 6.4 then covers the further structure of $\Xi(R, K)$ as a quasi-Hopf algebra, in much more detail than we have found elsewhere and with proofs. The coproduct here is well-known, for example it can be found in an equivalent form in [KM10A, Nat05], but we include its proof for completeness in our conventions and without certain restrictions previously assumed on R. The 'standard' antipode in Theorem 6.4.8 appears to be less well-known but is identical (up to conventions) to the antipode of $\Xi(R, K)$ as a Hopf quasigroup in [KM10A], and appears under slightly more assumptions in [Nat05]. The interaction between the coproduct and the standard antipode, in Proposition 6.4.9, is particularly new and follows from the $*$-quasi-Hopf algebra structure in Appendix 24. The antipode of a quasi-Hopf algebra[Dri87] is not unique but the standard one comes closest to familiar formulae for ordinary Hopf algebras, up to certain conjugations. In physical terms, the coproduct and antipode define the tensor product and dualisation of representations, which in our case are the boundary quasiparticles. We also give an extended series of examples, including one related to the octonions.

In Section 6.5, we give the promised categorical equivalence ${}_K\mathcal{M}^G_K \simeq {}_{\Xi(R,K)}\mathcal{M}$ concretely, deriving the quasi-bialgebra structure of $\Xi(R, K)$ precisely such that this works. Since the left hand side is independent of R, it should be that changing R changes $\Xi(R,K)$ by a Drinfeld cochain twist and we find this cochain, as a main result of the section. This is important as Drinfeld twists do not change the category of modules up to equivalence, so many aspects of the physics do not depend on R. Twisting arguments then imply that we have an antipode more generally for any R. We also look at $\mathcal{V} = {}_K\mathcal{M}^G$ as a module category for $\mathcal{C} = \mathcal{M}^G$. It can be shown further that $\mathcal{Z}({}_\Xi\mathcal{M}) \simeq \mathcal{Z}(\mathcal{M}^G) \simeq {}_{D(G)}\mathcal{M}$ as braided monoidal categories [Kon14], known as the bulk-boundary correspondence. At our algebraic level this means that $D(\Xi)$ is Drinfeld cochain twist equivalent to $D(G)$, using the double of a quasi-Hopf algebra[Maj98]. The algebra inclusion $i : \Xi \hookrightarrow D(G)$ is a part of this, but the full isomorphism here is beyond our scope, albeit given explicitly in [BGM96] in the case where $R \subseteq G$ is a subgroup and hence $\Xi(R,K)$ an ordinary bicrossproduct Hopf algebra.

Section 6.6 provides some concluding remarks, including about generalisations of the boundary theory to models based on other Hopf algebras [BMCA13, JKT22].

6.1 Preliminaries: recap of the Kitaev model in the bulk

We begin with the model in the bulk. This is largely a recap of Chapter 4 and eg. [Kit03, CM22].

6.1.1 Quantum double

Let G be a finite group with identity e, then $\mathbb{C}G$ is the group Hopf algebra with basis G. Multiplication is extended linearly, and $\mathbb{C}G$ has comultiplication $\Delta h = h \otimes h$ and counit $\epsilon h = 1$ on basis elements $h \in G$. The antipode is given by $Sh = h^{-1}$. $\mathbb{C}G$ is a Hopf $*$-algebra with $h^* = h^{-1}$ extended antilinearly. Its dual Hopf algebra $\mathbb{C}(G)$ of functions on G has basis of δ-functions $\{\delta_g\}$ with $\Delta\delta_g = \sum_h \delta_h \otimes \delta_{h^{-1}g}$, $\epsilon\delta_g = \delta_{g,e}$ and $S\delta_g = \delta_{g^{-1}}$ for the Hopf algebra structure, and $\delta_g^* = \delta_g$ for all $g \in G$. The normalised integral elements *in* $\mathbb{C}G$ and $\mathbb{C}(G)$ are

$$\Lambda_{\mathbb{C}G} = \frac{1}{|G|}\sum_{h\in G} h \in \mathbb{C}G, \quad \Lambda_{\mathbb{C}(G)} = \delta_e \in \mathbb{C}(G).$$

The integrals *on* $\mathbb{C}G$ and $\mathbb{C}(G)$ are

$$\int h = \delta_{h,e}, \quad \int \delta_g = 1$$

normalised so that $\int 1 = 1$ for $\mathbb{C}G$ and $\int 1 = |G|$ for $\mathbb{C}(G)$.

For the Drinfeld double we have $D(G) = \mathbb{C}(G){\blacktriangleright\!\!\!\triangleleft}\mathbb{C}G$ as in [Maj95], with $\mathbb{C}G$ and $\mathbb{C}(G)$ sub-Hopf algebras and the cross relations $h\delta_g = \delta_{hgh^{-1}}h$ (a semidirect product). The Hopf algebra antipode is $S(\delta_g h) = \delta_{h^{-1}g^{-1}h}h^{-1}$, and over $\mathbb{C}$ we have a Hopf $*$-algebra with $(\delta_g h)^* = \delta_{h^{-1}gh}h^{-1}$. There is also a quasitriangular structure which in subalgebra notation is

$$\mathcal{R} = \sum_{h\in G} \delta_h \otimes h \in D(G) \otimes D(G). \tag{6.1}$$

If we want to be totally explicit we can build $D(G)$ on either the vector space $\mathbb{C}(G)\otimes\mathbb{C}G$ or on the vector space $\mathbb{C}G\otimes\mathbb{C}(G)$. In fact the latter is more natural but we follow the conventions in [Maj95, CM22] and use the former. Then one can say the above more explicitly as

$$(\delta_g \otimes h)(\delta_f \otimes k) = \delta_g \delta_{hfh^{-1}} \otimes hk = \delta_{g,hfh^{-1}} \delta_g \otimes hk, \quad S(\delta_g \otimes h) = \delta_{h^{-1}g^{-1}h} \otimes h^{-1}$$

etc. for the operations on the underlying vector space.

As a semidirect product, irreducible representations of $D(G)$ are given by standard theory as labelled by pairs $(\mathcal{C}, \pi)$ consisting of an orbit under the action (i.e. by a conjugacy class $\mathcal{C} \subset G$ in this case) and an irrep π of the isotropy subgroup, in our case

$$G^{c_0} = \{n \in G \mid nc_0 n^{-1} = c_0\}$$

of a fixed element $c_0 \in \mathcal{C}$, i.e. the centraliser $C_G(c_0)$. The choice of c_0 does not change the isotropy group up to isomorphism but does change how it sits inside G. We also fix data $q_c \in G$ for each $c \in \mathcal{C}$ such that $c = q_c c_0 q_c^{-1}$ with $q_{c_0} = e$ and define from this a cocycle $\zeta_c(h) = q^{-1}_{hch^{-1}} h q_c$ as a map $\zeta : \mathcal{C} \times G \to G^{c_0}$. The associated irreducible representation is then

$$W_{\mathcal{C},\pi} = \mathbb{C}\mathcal{C} \otimes W_\pi, \quad \delta_g.(c \otimes w) = \delta_{g,c} c \otimes w, \quad h.(c \otimes w) = hch^{-1} \otimes \zeta_c(h).w$$

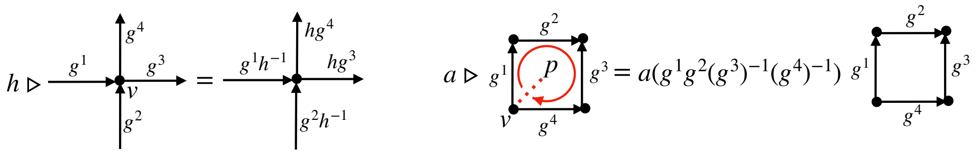
for all $w \in W_\pi$, the carrier space of π. This constructs all irreps of $D(G)$ and, over $\mathbb{C}$, these are unitary in a Hopf $*$-algebra sense if π is unitary. Moreover, $D(G)$ is semisimple and hence has a block decomposition $D(G) \cong \oplus_{\mathcal{C},\pi} \mathrm{End}(W_{\mathcal{C},\pi})$ given by a complete orthogonal set of self-adjoint central idempotents

$$P_{(\mathcal{C},\pi)} = \frac{\dim(W_\pi)}{|G^{c_0}|} \sum_{c \in \mathcal{C}} \sum_{n \in G^{c_0}} \mathrm{Tr}_\pi(n^{-1}) \delta_c \otimes q_c n q_c^{-1}. \qquad (6.2)$$

We refer to [CM22] for more details and proofs. Acting on a state, this will become a projection operator that determines if a quasiparticle of type $\mathcal{C}, \pi$ is present. Chargeons are quasiparticles with $\mathcal{C} = \{e\}$ and π an irrep of G, and fluxions are quasiparticles with $\mathcal{C}$ a conjugacy class and $\pi = 1$, the trivial representation.

6.1.2 Bulk lattice model

Having established the prerequisite algebra, we move on to the lattice model itself. This first part is largely a recap of [Kit03, CM22] and we use the notations of the latter. Let $\Sigma = \Sigma(V, E, P)$ be a square lattice viewed as a directed graph with its usual (cartesian) orientation, vertices V, directed edges E and faces P. The Hilbert space $\mathcal{H}$ will be a tensor product of vector spaces with one copy of $\mathbb{C}G$ at each arrow in E. We have group elements for the basis of each copy. Next, to each adjacent pair of vertex v and face p we associate a site $s = (v, p)$, or equivalently a line (the 'cilium') from p to v. We then define an action of $\mathbb{C}G$ and $\mathbb{C}(G)$ at each site by



Here $h \in \mathbb{C}G$, $a \in \mathbb{C}(G)$ and $g^1, \cdots, g^4$ denote independent elements of G (not powers). Observe that the vertex action is invariant under the location of p relative to its adjacent v, so the red dashed line has been omitted.

Lemma 6.1.1. [Kit03, CM22] $h\triangleright$ and $a\triangleright$ for all $h \in G$ and $a \in \mathbb{C}(G)$ define a representation of $D(G)$ on $\mathcal{H}$ associated to each site (v, p).

We next define

$$A(v) := \Lambda_{\mathbb{C}G}\triangleright = \frac{1}{|G|}\sum_{h\in G} h\triangleright, \quad B(p) := \Lambda_{\mathbb{C}(G)}\triangleright = \delta_e\triangleright$$

where $\delta_e(g^1g^2g^3g^4) = 1$ iff $g^1g^2g^3g^4 = e$, which is iff $(g^4)^{-1} = g^1g^2g^3$, which is iff $g^4g^1g^2g^3 = e$. Hence $\delta_e(g^1g^2g^3g^4) = \delta_e(g^4g^1g^2g^3)$ is invariant under cyclic rotations, hence $\Lambda_{\mathbb{C}(G)}\triangleright$ computed at site (v, p) does not depend on the location of v on the boundary of p. Moreover,

$$A(v)B(p) = |G|^{-1}\sum_h h\delta_e\triangleright = |G|^{-1}\sum_h \delta_{heh^{-1}}h\triangleright = |G|^{-1}\sum_h \delta_e h\triangleright = B(p)A(v)$$

if v is a vertex on the boundary of p by Lemma 6.1.1, and more trivially if not. We also have the rest of

$$A(v)^2 = A(v), \quad B(p)^2 = B(p), \quad [A(v), A(v')] = [B(p), B(p')] = [A(v), B(p)] = 0$$

for all $v \neq v'$ and $p \neq p'$, as easily checked. We then define the Hamiltonian

$$H = \sum_v (1 - A(v)) + \sum_p (1 - B(p))$$

and the space of vacuum states

$$\mathcal{H}_{\text{vac}} = \{|\psi\rangle \in \mathcal{H} \mid A(v)|\psi\rangle = B(p)|\psi\rangle = |\psi\rangle, \quad \forall v, p\}.$$

Quasiparticles in Kitaev models are labelled by representations of $D(G)$ occupying a given site (v, p), which take the system out of the vacuum. Detection of a quasiparticle is via a *projective measurement* of the operator $\sum_{\mathcal{C},\pi} p_{\mathcal{C},\pi}P_{\mathcal{C},\pi}$ acting at each site on the lattice for distinct coefficients $p_{\mathcal{C},\pi} \in \mathbb{R}$. By definition, this is a process which yields the classical value $p_{\mathcal{C},\pi}$ with a probability given by the likelihood of the state prior to the measurement being in the subspace in the image of $P_{\mathcal{C},\pi}$, and in so doing performs the corresponding action of the projector $P_{\mathcal{C},\pi}$ at the site. The projector $P_{e,1}$ corresponds to the vacuum quasiparticle.

In computing terms, this system of measurements encodes a logical Hilbert subspace, which we will always take to be the vacuum space $\mathcal{H}_{\text{vac}}$, within the larger physical Hilbert space given by the lattice; this subspace is dependent on the topology of the surface that the lattice is embedded in, but not the size of the lattice. For example, there is a convenient closed-form expression for the dimension of $\mathcal{H}_{\text{vac}}$ when Σ occupies a closed, orientable surface [CDH20]. Computation can then be performed on states in the logical subspace in a fault-tolerant manner, with unwanted excitations constituting detectable errors.

In the interest of brevity, we forgo a detailed exposition of such measurements, ribbon operators and fault-tolerant quantum computation on the lattice. The interested reader can learn about these in e.g. [Kit03, BM-D08, CCW16, CM22]. We do give a brief recap of ribbon operators, although without much rigour, as these will be useful later.

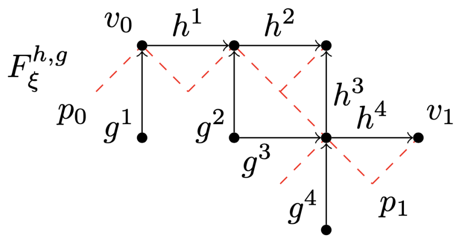


Figure 6.1: Example of a ribbon operator for a ribbon ξ from $s_0 = (v_0, p_0)$ to $s_1 = (v_1, p_1)$.

Definition 6.1.2. A ribbon ξ is a strip of face width that connects two sites $s_0 = (v_0, p_0)$ and $s_1 = (v_1, p_1)$ on the lattice. A ribbon operator $F_\xi^{h,g}$ acts on the vector spaces associated to the edges along the path of the ribbon, as shown in Fig 6.1. We call this basis of ribbon operators labelled by h and g the *group basis*.

Lemma 6.1.3. If ξ' is a ribbon concatenated with ξ, then the associated ribbon operators in the group basis satisfy

$$F_{\xi'\circ\xi}^{h,g} = \sum_{f\in G} F_{\xi'}^{f^{-1}hf, f^{-1}g} \circ F_\xi^{h,f}, \quad F_\xi^{h,g} \circ F_\xi^{h',g'} = \delta_{g,g'} F_\xi^{hh',g}.$$

The first identity shows the role of the comultiplication of $D(G)^*$,

$$\Delta(h\delta_g) = \sum_{f\in G} h\delta_f \otimes f^{-1}hf\delta_{f^{-1}g}.$$

using subalgebra notation, while the second identity implies that

$$(F_\xi^{h,g})^\dagger = F_\xi^{h^{-1},g}.$$

Lemma 6.1.4. [Kit03] Let ξ be a ribbon with the orientation as shown in Figure 6.1 between sites $s_0 = (v_0, p_0)$ and $s_1 = (v_1, p_1)$. Then

$$[F_\xi^{h,g}, f\triangleright_v] = 0, \quad [F_\xi^{h,g}, \delta_e\triangleright_p] = 0,$$

for all $v \notin \{v_0, v_1\}$ and $p \notin \{p_0, p_1\}$.

$$f\triangleright_{s_0} \circ F_\xi^{h,g} = F_\xi^{fhf^{-1},fg} \circ f\triangleright_{s_0}, \quad \delta_f\triangleright_{s_0} \circ F_\xi^{h,g} = F_\xi^{h,g} \circ \delta_{h^{-1}f}\triangleright_{s_0},$$

$$f\triangleright_{s_1} \circ F_\xi^{h,g} = F_\xi^{h,gf^{-1}} \circ f\triangleright_{s_1}, \quad \delta_f\triangleright_{s_1} \circ F_\xi^{h,g} = F_\xi^{h,g} \circ \delta_{fg^{-1}hg}\triangleright_{s_1}$$

for all ribbons where s_0, s_1 are disjoint, i.e. when s_0 and s_1 share neither vertices or faces. The subscript notation $f\triangleright_v$ means the local action of $f \in \mathbb{C}G$ at vertex v, and the dual for $\delta_f\triangleright_s$ at a site s.

We call the above lemma the *equivariance property* of ribbon operators. Such ribbon operators may be deformed according to a sort of discrete isotopy, so long as the endpoints remain the same. We formalised ribbon operators as left and right module maps in [CM22], but skim over any further details here. The physical interpretation of ribbon operators is that they create, move and annihilate quasiparticles.

Lemma 6.1.5. [Kit03] Let s_0, s_1 be two sites on the lattice. The only operators in $\mathrm{End}(\mathcal{H})$ which change the states at these sites, and therefore create quasiparticles and change the distribution of measurement outcomes, but leave the state in vacuum elsewhere, are ribbon operators.

This lemma is somewhat hard to prove rigorously but a proof was sketched in [CM22]. Next, there is an alternate basis for these ribbon operators in which the physical interpretation becomes more obvious. The *quasiparticle basis* has elements

$$F_\xi^{'\mathcal{C},\pi;u,v} = \sum_{n\in G^{c_0}} \pi(n^{-1})_{ji} F_\xi^{c,q_c n q_d^{-1}}, \tag{6.3}$$

where $\mathcal{C}$ is a conjugacy class, π is an irrep of the associated isotropy subgroup G^{c_0} and $u=(c,i)$, $v=(d,j)$ label basis elements of $W_{\mathcal{C},\pi}$ in which $c,d\in\mathcal{C}$ and i,j label a basis of W_π. This amounts to a nonabelian Fourier transform of the space of ribbons (that is, the Peter-Weyl isomorphism of $D(G)$) and has inverse

$$F_\xi^{h,g} = \sum_{\mathcal{C},\pi\in\hat{G^{c_0}}} \sum_{c\in\mathcal{C}} \delta_{h,gcg^{-1}} \sum_{i,j=0}^{\dim(W_\pi)} \pi(q_{gcg^{-1}}^{-1} g q_c)_{ij} F_\xi^{'\mathcal{C},\pi;a,b}, \tag{6.4}$$

where $a=(gcg^{-1},i)$ and $b=(c,j)$. This reduces in the chargeon sector to the special cases

$$F_\xi^{'e,\pi;i,j} = \sum_{n\in G} \pi(n^{-1})_{ji} F_\xi^{e,n} \tag{6.5}$$

and

$$F_\xi^{e,g} = \sum_{\pi\in\hat{G}} \sum_{i,j=0}^{\dim(W_\pi)} \pi(g)_{ij} F_\xi^{'e,\pi;i,j} \tag{6.6}$$

Meanwhile, in the fluxion sector we have

$$F_\xi^{'\mathcal{C},1;c,d} = \sum_{n\in G^{c_0}} F_\xi^{c,q_c n q_d^{-1}} \tag{6.7}$$

but there is no inverse in the fluxion sector. This is because the chargeon sector corresponds to the irreps of $\mathbb{C}G$, itself a semisimple algebra; the fluxion sector has no such correspondence.

If G is Abelian then π are 1-dimensional and we do not have to worry about the indices for the basis of W_π; this then looks like a more usual Fourier transform.

Lemma 6.1.6. If ξ' is a ribbon concatenated with ξ, then the associated ribbon operators in the quasiparticle basis satisfy

$$F_{\xi'\circ\xi}^{'\mathcal{C},\pi;u,v} = \sum_w F_{\xi'}^{'\mathcal{C},\pi;w,v} \circ F_\xi^{'\mathcal{C},\pi;u,w}$$

and are such that the nonabelian Fourier transform takes convolution to multiplication and vice versa, as it does in the abelian case.

In particular, we have the *ribbon trace operators*, $W_\xi^{\mathcal{C},\pi} := \sum_u F_\xi^{'\mathcal{C},\pi;u,u}$. Such ribbon trace operators create exactly quasiparticles of the type $\mathcal{C},\pi$ from the vacuum, meaning that

$$P_{(\mathcal{C},\pi)} \triangleright_{s_0} W_\xi^{\mathcal{C},\pi}|\mathrm{vac}\rangle = W_\xi^{\mathcal{C},\pi}|\mathrm{vac}\rangle = W_\xi^{\mathcal{C},\pi}|\mathrm{vac}\rangle \triangleleft_{s_1} P_{(\mathcal{C},\pi)}.$$

We refer to [CM22] for more details and proofs of the above.

Example 6.1.7. Our go-to example for our expositions will be $G = S_3$ generated by transpositions $u = (12), v = (23)$ with $w = (13) = uvu = vuv$. There are then 8 irreducible representations of $D(S_3)$ according to the choices $\mathcal{C}_0 = \{e\}$, $\mathcal{C}_1 = \{u, v, w\}$, $\mathcal{C}_2 = \{uv, vu\}$ for which we pick representatives $c_0 = e$, $q_e = e$, $c_1 = u$, $q_u = e$, $q_v = w$, $q_w = v$ and $c_2 = uv$ with $q_{uv} = e, q_{vu} = v$ (with the c_i in the role of c_0 in the general theory). Here $G^{c_0} = S_3$ with 3 representations π = trivial, sign and W_2 the 2-dimensional one given by (say) $\pi(u) = \sigma_3, \pi(v) = (\sqrt{3}\sigma_1 - \sigma_3)/2$, $G^{c_1} = \{e, u\} = \mathbb{Z}_2$ with $\pi(u) = \pm 1$ and $G^{c_2} = \{e, uv, vu\} = \mathbb{Z}_3$ with $\pi(uv) = 1, \omega, \omega^2$ for $\omega = e^{\frac{2\pi\imath}{3}}$. See [CM22] for details and calculations of the associated projectors and some $W_\xi^{\mathcal{C},\pi}$ operators.

6.2 Gapped boundaries

While $D(G)$ is the relevant algebra for the bulk of the model, our focus is on the boundaries. For these, we require a different class of algebras.

6.2.1 The boundary subalgebra $\Xi(R, K)$

Let $K \subseteq G$ be a subgroup of a finite group G and $G/K = \{gK \mid g \in G\}$ be the set of left cosets. It is not necessary in this section, but convenient, to fix a representative r for each coset and let $R \subseteq G$ be the set of these, so there is a bijection between R and G/K whereby $r \leftrightarrow rK$. We assume that $e \in R$ and call such a subset (or section of the map $G \to G/K$) a *transversal.* Every element of G factorises uniquely as rx for $r \in R$ and $x \in K$, giving a coordinatisation of G which we will use. Next, as we quotiented by K from the right, we still have an action of K from the left on G/K, which we denote $\triangleright$. By the above bijection, this equivalently means an action $\triangleright : K \times R \to R$ on R which in terms of the factorisation is determined by $xry = (x\triangleright r)y'$, where we refactorise xry in the form RK for some $y' \in K$. There is much more information in this factorisation, as will see in Section 6.4, but this action is all we need for now. Also note that we have chosen to work with left cosets so as to be consistent with the literature [CCW16, BSW11], but one could equally choose a right coset factorisation to build a class of algebras similar to those in [KM10A]. We consider the algebra $\mathbb{C}(G/K)\rtimes\mathbb{C}K$ as the cross product by the above action. Using our coordinatisation, this becomes the following algebra.

Definition 6.2.1. $\Xi(R, K) = \mathbb{C}(R)\rtimes\mathbb{C}K$ is generated by $\mathbb{C}(R)$ and $\mathbb{C}K$ with cross relations $x\delta_r = \delta_{x\triangleright r}x$. Over $\mathbb{C}$, this is a $*$-algebra with $(\delta_r x)^* = x^{-1}\delta_r = \delta_{x^{-1}\triangleright r}x^{-1}$.

If we choose a different transversal R then the algebra does not change up to an isomorphism which maps the δ-functions between the corresponding choices of representative. Of relevance to the applications, we also have:

Lemma 6.2.2. $\Xi(R, K)$ has the 'integral element'

$$\Lambda := \Lambda_{\mathbb{C}(R)} \otimes \Lambda_{\mathbb{C}K} = \delta_e \frac{1}{|K|} \sum_{x \in K} x$$

characterised by $\xi\Lambda = \epsilon(\xi)\Lambda = \Lambda\xi$ for all $\xi \in \Xi$, and $\epsilon(\Lambda) = 1$, where $\epsilon(\delta_s \otimes x) = \delta_{s,e}$ is the counit.

Proof. Let $\xi = \delta_s y$ w.l.o.g. We check that

$$\xi\Lambda = (\delta_s y)(\delta_e \frac{1}{|K|}\sum_{x\in K} x) = \delta_{s,y\triangleright e}\delta_s \frac{1}{|K|}\sum_{x\in K} yx = \delta_{s,e}\delta_e \frac{1}{|K|}\sum_{x\in K} x$$
$$= \epsilon(\xi)\Lambda = \frac{1}{|K|}\sum_{x\in K}\delta_{e,x\triangleright y}\delta_e xy = \frac{1}{|K|}\sum_{x\in K}\delta_{e,y}\delta_e x = \Lambda\xi.$$

And clearly, $\epsilon(\Lambda) = \delta_{e,e}\frac{|K|}{|K|} = 1$. ∎

As a cross product algebra, we can take the same approach as with $D(G)$ to the classification of its irreps:

Lemma 6.2.3. Irreps of $\Xi(R,K)$ are classified by pairs $(\mathcal{O},\rho)$ where $\mathcal{O}\subseteq R$ is an orbit under the action $\triangleright$ and ρ is an irrep of the isotropy group $K^{r_0} := \{x\in K \mid x\triangleright r_0 = r_0\}$. Here we fix a base point $r_0\in\mathcal{O}$ as well as $\kappa:\mathcal{O}\to K$ a choice of lift such that

$$\kappa_r\triangleright r_0 = r,\quad \forall r\in\mathcal{O},\quad \kappa_{r_0} = e.$$

Then
$$V_{\mathcal{O},\rho} = \mathbb{C}\mathcal{O}\otimes V_\rho,\quad \delta_r(s\otimes v) = \delta_{r,s}s\otimes v,\quad x.(s\otimes v) = x\triangleright s\otimes\zeta_s(x).v$$
for $v\in V_\rho$, the carrier space for ρ, and

$$\zeta:\mathcal{O}\times K\to K^{r_0},\quad \zeta_r(x) = \kappa^{-1}_{x\triangleright r}x\kappa_r.$$

Proof. One can check that $\zeta_r(x)$ lives in K^{r_0},

$$\zeta_r(x)\triangleright r_0 = (\kappa^{-1}_{x\triangleright r}x\kappa_r)\triangleright r_0 = \kappa^{-1}_{x\triangleright r}\triangleright(x\triangleright r) = \kappa^{-1}_{x\triangleright r}\triangleright(\kappa_{x\triangleright r}\triangleright r_0) = r_0$$

and the cocycle property

$$\zeta_r(xy) = \kappa^{-1}_{x\triangleright y\triangleright r}x\kappa_{y\triangleright r}\kappa^{-1}_{y\triangleright r}y\kappa_r = \zeta_{y\triangleright r}(x)\zeta_r(y),$$

from which it is easy to see that $V_{\mathcal{O},\rho}$ is a representation,

$$x.(y.(s\otimes v)) = x.(y\triangleright s\otimes\zeta_s(y).v) = x\triangleright(y\triangleright s)\otimes\zeta_{y\triangleright s}(x)\zeta_s(y).v = xy\triangleright s\otimes\zeta_s(xy).v = (xy).(s\otimes v),$$
$$x.(\delta_r.(s\otimes v)) = \delta_{r,s}x\triangleright s\otimes\zeta_s(x).v = \delta_{x\triangleright r,x\triangleright s}x\triangleright s\otimes\zeta_s(x).v = \delta_{x\triangleright r}.(x.(s\otimes v)).$$

That $V_{\mathcal{O},\pi}$ is irreducible is by similar arguments to the construction of irreps of semidirect products groups (such as the Poincaré group), but we provide a short proof directly for our case. Indeed, suppose that $W\subseteq V_{\mathcal{O},\rho}$ is a non-zero subrepresentation under $\Xi(R,K)$. Then W has the form $W = \oplus_{s\in\mathcal{O}}s\otimes W_s$ for some subspaces $W_s\subseteq V_\rho$ and we show that $W_s = V_\rho$ for all s so that $W = V_{\mathcal{O},\rho}$. Let $0\neq w = \sum_{t\in\mathcal{O}}t\otimes v_t\in W$ with at least one component say $v_t\neq 0$ for some t. Let $t = y^{-1}\triangleright s$ for some $y\in K$, since $\mathcal{O}$ is a single orbit, then $y\delta_t.w = y.(t\otimes v_t) = s\otimes\zeta_t(y).v_t = s\otimes v\in W$ since W is closed under the action of $\Xi(R,K)$, for some element $v = \zeta_t(y).v_t\in W_s$ which is nonzero since $\zeta_t(y)\in K^{r_0}$ is a group element (so its action on v_t is invertible). Now consider the set

$$\kappa_s\mathbb{C}K^{r_0}\kappa_s^{-1}.(s\otimes v) = s\otimes\mathbb{C}K^{r_0}.v\subseteq s\otimes W_s\subseteq s\otimes V_\rho$$

since $\kappa_s x\kappa_s^{-1}\triangleright s = \kappa_s x\triangleright r_0 = \kappa_s\triangleright r_0 = s$ for all $x\in K^{r_0}$ and W is closed under the action of $\Xi(R,K)$. Here, $\kappa_s(\ )\kappa_s^{-1}$ is a bijection between K^{r_0} and the isotropy group K^s at s. Now $\mathbb{C}K^{r_0}.v\subseteq V_\rho$ is a nonzero representation of K^{r_0} since we can act further from the left by K^{r_0}, and hence equal to V_ρ as the latter is irreducible. It follows that $W_s = V_\rho$ also since this was in between the two spaces. One can further show that the stated construction does not depend up to isomorphism on the choice of r_0 or κ_r. ∎

In the $*$-algebra case as here, we obtain a unitary representation if ρ is unitary. One can also show that all irreps can be obtained this way. In fact the algebra $\Xi(R,K)$ is semisimple and has a block associated to the $V_{\mathcal{O},\pi}$.

Lemma 6.2.4. $\Xi(R,K)$ has a complete orthogonal set of central idempotents

$$P_{(\mathcal{O},\rho)} = \frac{\dim V_\rho}{|K^{r_0}|}\sum_{r\in\mathcal{O}}\sum_{n\in K^{r_0}} \mathrm{Tr}_\rho(n^{-1})\delta_r\otimes\kappa_r n\kappa_r^{-1}.$$

Proof. The proofs are similar to those for $D(G)$ in [CM22]. That we have a projection is

$$\begin{aligned}P^2_{(\mathcal{O},\rho)} &= \frac{\dim(V_\rho)^2}{|K^{r_0}|^2}\sum_{m,n\in K^{r_0}}\mathrm{Tr}_\rho(m^{-1})\,\mathrm{Tr}_\rho(n^{-1})\sum_{r,s\in\mathcal{O}}(\delta_r\otimes\kappa_r m\kappa_r^{-1})(\delta_s\otimes\kappa_s n\kappa_s^{-1})\\ &= \frac{\dim(V_\rho)^2}{|K^{r_0}|^2}\sum_{m,n\in K^{r_0}}\mathrm{Tr}_\rho(m^{-1})\,\mathrm{Tr}_\rho(n^{-1})\sum_{r,s\in\mathcal{O}}\delta_r\delta_{r,s}\otimes\kappa_r m\kappa_r^{-1}\kappa_s n\kappa_s^{-1}\\ &= \frac{\dim(V_\rho)^2}{|K^{r_0}|^2}\sum_{m,m'\in K^{r_0}}\mathrm{Tr}_\rho(m^{-1})\,\mathrm{Tr}_\rho(mm'^{-1})\sum_{r\in\mathcal{O}}\delta_r\otimes\kappa_r m'\kappa_r^{-1} = P_{(\mathcal{O},\rho)}\end{aligned}$$

where we used $r=\kappa_r m\kappa_r^{-1}{\triangleright}s$ iff $s=\kappa_r m^{-1}\kappa_r^{-1}{\triangleright}r=\kappa_r m^{-1}{\triangleright}r_0=\kappa_r{\triangleright}r_0=r$. We then changed $mn=m'$ as a new variable and used the orthogonality formula for characters on K^{r_0}. Similarly, for different projectors to be orthogonal. The sum of projectors is 1 since

$$\sum_{\mathcal{O},\rho}P_{(\mathcal{O},\rho)} = \sum_{\mathcal{O},r\in\mathcal{C}}\delta_r\otimes\kappa_r\sum_{\rho\in\hat{K^{r_0}}}\left(\frac{\dim V_\rho}{|K^{r_0}|}\sum_{n\in K^{r_0}}\mathrm{Tr}_\rho(n^{-1})n\right)\kappa_r^{-1} = \sum_{\mathcal{O},r\in\mathcal{O}}\delta_r\otimes 1 = 1,$$

where the bracketed expression is the projector P_ρ for ρ in the group algebra of K^{r_0}, and these sum to 1 by the Peter-Weyl decomposition of the latter. ■

Remark 6.2.5. In the previous literature, the irreps have been described using double cosets and representatives thereof [CCW16]. In fact a double coset in ${}_KG_K$ is an orbit for the left action of K on G/K and hence has the form $\mathcal{O}K$ corresponding to an orbit $\mathcal{O}\subset R$ in our approach. We will say more about this later, in Proposition 6.5.3.

An important question for the physics is how representations on the bulk relate to those on the boundary. This is afforded by functors in the two directions. Here we give a direct approach to this issue as follows.

Proposition 6.2.6. There is an inclusion of algebras $i:\Xi(R,K)\hookrightarrow D(G)$

$$i(x)=x,\quad i(\delta_r)=\sum_{x\in K}\delta_{rx}.$$

The pull-back or restriction of a $D(G)$-module W to a Ξ-module $i^*(W)$ is simply for $\xi\in\Xi$ to act by $i(\xi)$. Going the other way, the induction functor sends a Ξ-module V to a $D(G)$-module $D(G)\otimes_\Xi V$, where $\xi\in\Xi$ right acts on $D(G)$ by right multiplication by $i(\xi)$. These two functors are adjoint.

Proof. We just need to check that i respects the relations of Ξ. Thus,

$$i(\delta_r)i(\delta_s) = \sum_{x,y\in K} \delta_{rx}\delta_{sy} = \sum_{x\in K} \delta_{r,s}\delta_{rx} = i(\delta_r\delta_s),$$
$$i(x)i(\delta_r) = \sum_{y\in K} x\delta_{ry} = \sum_{y\in K} \delta_{xryx^{-1}}x = \sum_{y\in K} \delta_{(x\triangleright r)x'yx^{-1}}x = \sum_{y'\in K} \delta_{(x\triangleright r)y'}x = i(\delta_{x\triangleright r}x),$$

as required. For the first line, we used the unique factorisation $G = RK$ to break down the δ-functions. For the second line, we use this in the form $xr = (x\triangleright r)x'$ for some $x' \in K$ and then changed variables from y to $y' = x'yx^{-1}$. The rest follows as for any algebra inclusion. ■

In fact, Ξ is a quasi-bialgebra and at least when $(\ )^R$ is bijective a quasi-Hopf algebra, as we see in Section 6.4. In the latter case, it has a quantum double $D(\Xi)$ which contains Ξ as a sub-quasi Hopf algebra. Moreover, it can be shown that $D(\Xi)$ is a 'Drinfeld cochain twist' of $D(G)$, which implies it has the same algebra as $D(G)$. This is the algebraic level of the bulk-boundary correspondence [Kon14]. We do not need the full isomorphism here, which is beyond our scope since the double of a quasi-Hopf algebra[Maj98] is itself quite complicated to describe explicitly, but this is the abstract reason for the above inclusion. (An explicit proof of this twisting result in the usual Hopf algebra case with R a group is in [BGM96].) Meanwhile, the statement that the two functors in the lemma are adjoint is that

$$\mathrm{hom}_{D(G)}(D(G) \underset{\Xi}{\otimes} V, W)) = \mathrm{hom}_{\Xi}(V, i^*(W))$$

for all Ξ-modules V and all $D(G)$-modules W. These functors do not take irreps to irreps and of particular interest are the multiplicities for the decompositions back into irreps, i.e. if V_i, W_a are respective irreps and $D(G)\otimes_\Xi V_i = \oplus_a n^i{}_a W_a$ then

$$\dim(\mathrm{hom}_{D(G)}(D(G) \underset{\Xi}{\otimes} V_i, W_a)) = \dim(\mathrm{hom}_{\Xi}(V_i, i^*(W_a)))$$

and hence $i^*(W_a) = \oplus_i n^i_a V_i$. It remains to give a formula for these multiplicities; here we were not able to reproduce the formulae of [CCW16, Thm 2.12][CCW17, Thm 2.12] which were stated without proof and referenced back to [Sch16]. Instead, our approach goes via a general lemma. First, recall that a linear map $\int : B \to \mathbb{C}$ is Frobenius if the bilinear form $(b, c) := \int bc$ is nondegenerate, and is symmetric if this bilinear form is symmetric. Also, let $g = g^1 \otimes g^2 \in B \otimes B$ (in a notation with the sum of such terms understood) be the associated 'metric' such that $(\int bg^1)g^2 = b = g^1 \int g^2 b$ for all b (it is the inverse matrix in a basis of the algebra). We say that the Frobenius form is special if the algebra product $\cdot$ obeys $\cdot(g) = 1$. It is well-known that there is a unique symmetric special Frobenius form up to scale, given by the trace in the left regular representation, see [MR21] for a recent study.

Lemma 6.2.7. Let $i : A \hookrightarrow B$ be an inclusion of finite-dimensional semisimple algebras and $\int$ the unique symmetric special Frobenius linear form on B such that $\int 1 = 1$. Let V_i be an irrep of A and W_a an irrep of B. Then the multiplicity V_i in the pull-back $i^*(W_a)$ (which is the same as the multiplicity of W_a in $B \otimes_A V_i$) is given by

$$n^i{}_a = \frac{\dim(B)}{\dim(V_i)\dim(W_a)} \int i(P_i)P_a,$$

where $P_i \in A$ and $P_a \in B$ are the associated central idempotents. Moreover, $i(P_i)P_a = 0$ if and only if $n^i_a = 0$.

Proof. In our case, over $\mathbb{C}$, we know that a finite-dimensional semisimple algebra B is a direct sum of matrix algebras $\mathrm{End}(W_a)$ associated to the irreps W_a of B. Then

$$\begin{aligned}\int i(P_i)P_a &= \frac{1}{\dim(B)}\sum_{\alpha,\beta}\langle f^\alpha\otimes e_\beta, i(P_i)P_a(e_\alpha\otimes f^\beta)\rangle\\ &= \frac{1}{\dim(B)}\sum_{\alpha}\dim(W_a)\langle f^\alpha, i(P_i)e_\alpha\rangle = \frac{\dim(W_a)\dim(V_i)}{\dim(B)}n^i{}_a.\end{aligned}$$

where $\{e_\alpha\}$ is a basis of W_a and $\{f^\beta\}$ is a dual basis, and P_a acts as the identity on $\mathrm{End}(W_a)$ and zero on the other blocks. We then used that if $i^*(W_a)=\oplus_i n^i{}_a V_i$ as A-modules, then $i(P_i)$ just picks out the V_i components where P_i acts as the identity.

For the last part, the forward direction is immediate given the first part of the lemma. For the other direction, suppose $n^i_a=0$ so that $i^*(W_a)=\oplus_j n^j_a V_j$ with $j\neq a$ running over the other irreps of A. Now, we can view $P_a\in W_a\otimes W_a^*$ (as the identity element) and left multiplication by $i(P_i)$ is the same as P_i acting on P_a viewed as an element of $i^*(W_a)\otimes W_a^*$, which is therefore zero. ■

We apply Lemma 6.2.7 in our case of $A=\Xi$ and $B=D(G)$, where

$$\dim(V_i)=|\mathcal{O}|\dim(V_\rho),\quad \dim(W_a)=|\mathcal{C}|\dim(W_\pi)$$

with $i=(\mathcal{C},\rho)$ as described above and $a=(\mathcal{C},\pi)$ as described in Section 6.1.

Proposition 6.2.8. For the inclusion $i:\Xi\hookrightarrow D(G)$ in Proposition 6.2.6, the multiplicities for restriction and induction as above are given by

$$n^{(\mathcal{O},\rho)}_{(\mathcal{C},\pi)}=\frac{|G|}{|\mathcal{O}||\mathcal{C}||K^{r_0}||G^{c_0}|}\sum_{\substack{r\in\mathcal{O},c\in\mathcal{C}\\ r^{-1}c\in K}}|K^{r,c}|\sum_{\tau\in\hat{K^{r,c}}}n_{\tau,\tilde\rho|_{K^{r,c}}}n_{\tau,\tilde\pi|_{K^{r,c}}},\qquad K^{r,c}=K^r\cap G^c,$$

where $\tilde\pi(m)=\pi(q_c^{-1}mq_c)$ and $\tilde\rho(m)=\rho(\kappa_r^{-1}m\kappa_r)$ are the corresponding representation of K^r,G^c decomposing as $K^{r,c}$ representations as

$$\tilde\rho|_{K^{r,c}}\cong\oplus_\tau\, n_{\tau,\tilde\rho|_{K^{r,c}}}\tau,\quad \tilde\pi|_{K^{r,c}}\cong\oplus_\tau\, n_{\tau,\tilde\pi|_{K^{r,c}}}\tau.$$

Proof. We include the projector from Lemma 6.2.4 as

$$i(P_{(\mathcal{O},\rho)})=\frac{\dim(V_\rho)}{|K^{r_0}|}\sum_{r\in\mathcal{O},x\in K}\sum_{m\in K^{r_0}}\mathrm{Tr}_\rho(m^{-1})\delta_{rx}\otimes\kappa_r m\kappa_r^{-1}$$

and multiply this by $P_{(\mathcal{C},\pi)}$ from (6.2). In the latter, we write $c=sy$ for the factorisation of c. Then when we multiply these out, for $(\delta_{rx}\otimes\kappa_r m\kappa_r^{-1})(\delta_c\otimes q_c n q_c^{-1})$ we will need $\kappa_r m\kappa_r^{-1}{\triangleright} s=r$ or equivalently $s=\kappa_r m^{-1}\kappa_r^{-1}{\triangleright} r=r$ so we are actually summing not over c but over $y\in K$ such that $ry\in\mathcal{C}$. Also then x is uniquely determined in terms of y. Hence

$$i(P_{(\mathcal{O},\rho)})P_{(\mathcal{C},\pi)}=\frac{\dim(V_\rho)\dim(W_\pi)}{|K^{r_0}||G^{c_0}|}\sum_{m\in K^{r_0},n\in G^{c_0}}\ \sum_{r\in\mathcal{O},y\in K|ry\in\mathcal{C}}\mathrm{Tr}_\rho(m^{-1})\,\mathrm{Tr}_\pi(n^{-1})\delta_{rx}\otimes\kappa_r m\kappa_r^{-1}q_c n q_c^{-1}.$$

Now we apply the integral of $D(G)$, $\int\delta_g\otimes h=\delta_{h,e}$ which requires

$$n=q_c^{-1}\kappa_r m^{-1}\kappa_r^{-1}q_c$$

and $x = y$ for $n \in G^{c_0}$ given that $c = ry$. We refer to this condition on y as $(\star)$. Remembering that $\int$ is normalised so that $\int 1 = |G|$, we have from the lemma

$$\begin{aligned} n^{(\mathcal{O},\rho)}_{(\mathcal{C},\pi)} &= \frac{|G|}{\dim(V_i)\dim(W_a)} \int i(P_{(\mathcal{O},\rho)})P_{(\mathcal{C},\pi)} \\ &= \frac{|G|}{|\mathcal{O}||\mathcal{C}||K^{r_0}||G^{c_0}|} \sum_{m\in K^{r_0}} \sum_{\substack{r\in\mathcal{O},y\in K\\ (*),ry\in\mathcal{C}}} \mathrm{Tr}_\rho(m^{-1})\,\mathrm{Tr}_\pi(q_{ry}^{-1}\kappa_r m\kappa_r^{-1}q_{ry}) \\ &= \frac{|G|}{|\mathcal{O}||\mathcal{C}||K^{r_0}||G^{c_0}|} \sum_{m\in K^{r_0}} \sum_{\substack{r\in\mathcal{O},c\in\mathcal{C}\\ r^{-1}c\in K}} \sum_{m'\in K^r\cap G^c} \mathrm{Tr}_\rho(\kappa_r^{-1}m'^{-1}\kappa_r)\,\mathrm{Tr}_\pi(q_c^{-1}mq_c), \end{aligned}$$

where we compute in G and view $(\star)$ as $m' := \kappa_r m\kappa_r^{-1} \in G^c$. We then use the group orthogonality formula

$$\sum_{m\in K^{r,c}} \mathrm{Tr}_\tau(m^{-1})\,\mathrm{Tr}_{\tau'}(m) = \delta_{\tau,\tau'}|K^{r,c}|$$

for any irreps τ,τ' of the group

$$K^{r,c} := K^r \cap G^c = \{x \in K \mid x\triangleright r = r, \quad xcx^{-1} = c\}$$

to obtain the formula stated. ■

This simplifies in four (overlapping) special cases as follows.
(i) V_i trivial:

$$n^{(\{e\},1)}_{(\mathcal{C},\pi)} = \frac{|G|}{|\mathcal{C}||K||G^{c_0}|} \sum_{c\in\mathcal{C}\cap K} \sum_{m\in K\cap G^c} \mathrm{Tr}_\pi(q_c^{-1}mq_c) = \frac{|G|}{|\mathcal{C}||K||G^{c_0}|} \sum_{c\in\mathcal{C}\cap K} |K^c| n_{1,\tilde\pi}$$

as $\rho = 1$ implies $\tilde\rho = 1$ and forces $\tau = 1$. Here K^c is the centraliser of $c \in K$. If $n_{1,\tilde\pi}$ is independent of the choice of c then we can simplify this further as

$$n^{(\{e\},1)}_{(\mathcal{C},\pi)} = \frac{|G||(\mathcal{C}\cap K)/K|}{|\mathcal{C}||G^{c_0}|} n_{1,\pi|_{K^{c_0}}}$$

using the orbit-counting lemma, where K acts on $\mathcal{C}\cap K$ by conjugation.
(ii) W_a trivial:

$$n^{(\mathcal{O},\rho)}_{(\{e\},1)} = \frac{|G|}{|\mathcal{O}||K^{r_0}||G|} \sum_{r\in\mathcal{O}\cap K} \sum_{m\in K^{r_0}} \mathrm{Tr}_\rho(m^{-1}) = \begin{cases} 1 & \text{if } \mathcal{O},\rho \text{ trivial} \\ 0 & \text{else} \end{cases}$$

as $\mathcal{O}\cap K = \{e\}$ if $\mathcal{O} = \{e\}$ (but is otherwise empty) and in this case only $r = e$ contributes. This is consistent with the fact that if W_a is the trivial representation of $D(G)$ then its pull back is also trivial and hence contains only the trivial representation of Ξ.
(iii) Fluxion sector:

$$n^{(\mathcal{O},1)}_{(\mathcal{C},1)} = \frac{|G|}{|\mathcal{O}||\mathcal{C}||K^{r_0}||G^{c_0}|} \sum_{\substack{r\in\mathcal{O},c\in\mathcal{C}\\ r^{-1}c\in K}} |K^r \cap G^c|.$$

(iv) Chargeon sector:

$$n^{(\{e\},\rho)}_{(\{e\},\pi)} = n_{\rho,\pi|_K},$$

where ρ,π are arbitrary irreps of K,G respectively and only $r = c = e$ are allowed so $K^{r,c} = K$, and then only $\tau = \rho$ contributes.

Example 6.2.9. (i) We take $G = S_3$, $K = \{e, u\} = \mathbb{Z}_2$, where $u = (12)$. Here G/K consists of

$$G/K = \{\{e, u\}, \{w, uv\}, \{v, vu\}\}$$

and our standard choice of R will be $R = \{e, uv, vu\}$, where we take one from each coset (but any other transversal will have the same irreps and their decompositions). This leads to 3 irreps of $\Xi(R, K)$ as follows. In R, we have two orbits $\mathcal{O}_0 = \{e\}$, $\mathcal{O}_1 = \{uv, vu\}$ and we choose representatives $r_0 = e, \kappa_e = e$, $r_1 = uv, \kappa_{uv} = e, \kappa_{vu} = u$ since $u\triangleright(uv) = vu$ for the two cases (here r_1 was denoted r_0 in the general theory and is the choice for $\mathcal{O}_1$). We also have $u\triangleright(vu) = uv$. Note that it happens that these orbits are also conjugacy classes but this is an accident of S_3 and not true for S_4. We have $K^{r_0} = K = \mathbb{Z}_2$ with representations $\rho(u) = \pm 1$ and $K^{r_1} = \{e\}$ with only the trivial representation.

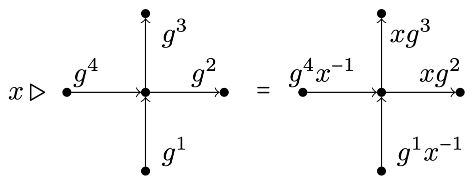
(ii) For $D(S_3)$, we have the 8 irreps in Example 6.1.7 and hence there is a 3×8 table of the $\{n^i{}_a\}$. We can easily compute some of the special cases from the above. For example, the trivial π restricted to K is $\rho = 1$, the sign representation restricted to K is the $\rho = -1$ representation, the W_2 restricted to K is $1 \oplus -1$, which gives the upper left 2×3 submatrix for the chargeon sector. Another 6 entries (four new ones) are given from the fluxion formula. We also have $\mathcal{C}_2 \cap K = \emptyset$ so that the latter part of the first two rows is zero by our first special case formula. For $\mathcal{C}_1, \pm 1$ in the first row, we have $\mathcal{C}_1 \cap K = \{u\}$ with trivial action of K, so just one orbit. This gives us a nontrivial result in the $+1$ case and 0 in the -1 case. The story for $\mathcal{C}_1, \pm 1$ in the second row follows the same derivation, but needs $\tau = -1$ and hence $\pi = -1$ for the nonzero case. In the third row with $\mathcal{C}_2, \pi$, we have $K^r = \{e\}$ so $G' = \{e\}$ and we only have $\tau = 1 = \rho$ as well as $\tilde{\pi} = 1$ independently of π as this is 1-dimensional. So both n factors in the formula in Proposition 6.2.8 are 1. In the sum over r, c, we need $c = r$ so we sum over 2 possibilities, giving a nontrivial result as shown. For $\mathcal{C}_1, \pi$, the first part goes the same way and we similarly have c determined from r in the case of $\mathcal{C}_1, \pi$, so again two contributions in the sum, giving the answer shown independently of π. Finally, for $\mathcal{C}_0, \pi$ we have $r = \{uv, vu\}$ and $c = e$, and can never meet the condition $r^{-1}c \in K$. So these all have 0. Thus, Proposition 6.2.8 in this example tells us:

$n^i{}_a$	$\mathcal{C}_0, 1$	$\mathcal{C}_0, \mathrm{sign}$	$\mathcal{C}_0, W_2$	$\mathcal{C}_1, 1$	$\mathcal{C}_1, -1$	$\mathcal{C}_2, 1$	$\mathcal{C}_2, \omega$	$\mathcal{C}_2, \omega^2$
$\mathcal{O}_0, 1$	1	0	1	1	0	0	0	0
$\mathcal{O}_0, -1$	0	1	1	0	1	0	0	0
$\mathcal{O}_1, 1$	0	0	0	1	1	1	1	1

One can check for consistency that for each W_a, $\dim(W_a)$ is the sum of the dimensions of the V_i that it contains, which determines one row from the other two.

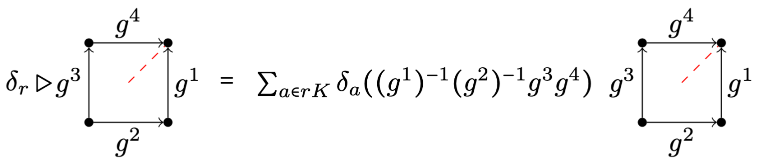
6.2.2 Boundary lattice model

Consider a vertex on the lattice Σ. Fixing a subgroup $K \subseteq G$, we define an action of $\mathbb{C}K$ on $\mathcal{H}$ by



(6.8)

One can see that this is an action as it is a tensor product of representations on each edge, or simply because it is the restriction to K of the vertex action of G in the bulk. Next, we define the action of $\mathbb{C}(R)$ at a face relative to a cilium,

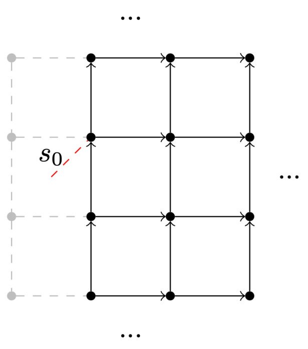

g^4
$\delta_r \triangleright g^3$
g^1
g^2
$= \sum_{a\in rK} \delta_a((g^1)^{-1}(g^2)^{-1}g^3g^4)\ g^3$
g^4
g^1
g^2

(6.9)

with a clockwise rotation. That this is indeed an action is also easy to check explicitly, recalling that either $rK = r'K$ when $r = r'$ or $rK \cap r'K = \emptyset$ otherwise, for any $r, r' \in R$. These actions define a representation of $\Xi(R, K)$, which is just the bulk $D(G)$ action restricted to $\Xi(R, K) \subseteq D(G)$ by the inclusion in Proposition 6.2.6. This says that $x \in K$ acts as in G and $\mathbb{C}(R)$ acts on faces by the $\mathbb{C}(G)$ action after sending $\delta_r \mapsto \sum_{a\in rK} \delta_a$. To connect the above representation to the topic at hand, we now define a boundary. We will consider two different types, rough and smooth boundaries, which were first described in the $D(\mathbb{Z}_2)$ case in [BK98].

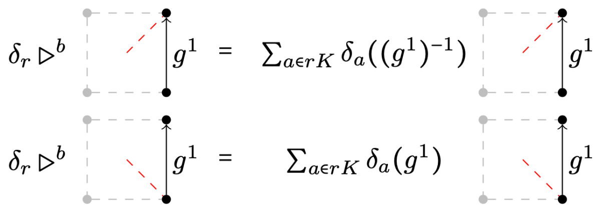
Smooth boundaries

Consider the lattice in the half-plane for simplicity,



where each solid black arrow still carries a copy of $\mathbb{C}G$ and ellipses indicate the lattice extending infinitely. The boundary runs along the left hand side and we refer to the rest of the lattice as the 'bulk'. The grey dashed edges and vertices are there to indicate empty space and the lattice borders the edge with faces; we will call this case a 'smooth' boundary. There is a site s_0 indicated at the boundary.

There is an action of $\mathbb{C}K$ at the boundary vertex associated to s_0, identical to the action of $\mathbb{C}K$ defined above but with the left edge undefined. Similarly, there is an action of $\mathbb{C}(R)$ at the face associated to s_0. However, this is more complicated, as the face has three edges undefined and the action must be defined slightly differently from in the bulk:

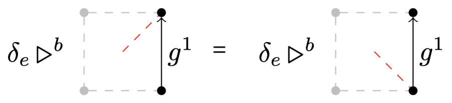

$\delta_r \triangleright^b$
g^1
$=$
$\Sigma_{a\in rK}\,\delta_a((g^1)^{-1})$
g^1
$\delta_r \triangleright^b$
g^1
$=$
$\Sigma_{a\in rK}\,\delta_a(g^1)$
g^1

where the action is given a superscript $\triangleright^b$ to differentiate it from the actions in the bulk. In the first case, we follow the same clockwise rotation rule but skip over the undefined values on the grey edges, but for the second case we go round round anticlockwise. The resulting rule is then according to whether the cilium is associated to the top or bottom of the edge. It is easy to check that this defines a representation of $\Xi(R,K)$ on $\mathcal{H}$ associated to each smooth boundary site, such as s_0, and that the actions of $\mathbb{C}(R)$ have been chosen such that this holds. A similar principle holds for $\triangleright^b$ in other orientations of the boundary.

The integral actions at a boundary vertex v and at a face $s_0 = (v,p)$ of a smooth boundary are then

$$A_1^b(v) := \Lambda_{\mathbb{C}K}\triangleright_v^b = \frac{1}{|K|}\sum_k k\triangleright_v^b, \quad B_1^b(p) := \Lambda_{\mathbb{C}(R)}\triangleright_p^b = \delta_e\triangleright_p^b,$$

where the superscript b and subscript 1 label that these are at a smooth boundary. We have the convenient property that



so both the vertex and face integral actions at a smooth face each depend only on the vertex and face respectively, not the precise cilium, similar to the integral actions.

Remark 6.2.10. There is similarly an action of $\mathbb{C}(G)\rtimes\mathbb{C}K \subseteq D(G)$ on $\mathcal{H}$ at each site in the next layer into the bulk, where the site has the vertex at the boundary but an internal face. We mention this for completeness, and because using this fact it is easy to show that

$$A_1^b(v)B(p) = B(p)A_1^b(v),$$

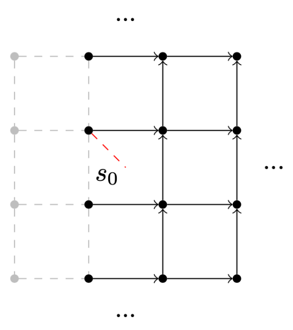
where $B(p)$ is the usual integral action in the bulk.

Remark 6.2.11. In [BSW11] it is claimed that one can similarly introduce actions at smooth boundaries defined not only by R and K but also a 2-cocycle α. This makes some sense categorically, as the module categories of $\mathcal{M}^G$ may also include such a 2-cocycle, which enters by way of a *twisted* group algebra $\mathbb{C}_\alpha K$ [Ost03A]. However, in Figure 6 of [BSW11] one can see that when the cocycle α is introduced all edges on the boundary are assumed to be copies of $\mathbb{C}K$, rather than $\mathbb{C}G$. On closer inspection, it is evident that this means that the action on faces of $\delta_e \in \mathbb{C}(R)$ will always yield 1, and the action of any other basis element of $\mathbb{C}(R)$ will yield 0. Similarly, the action on vertices is defined to still be an action of $\mathbb{C}K$, not $\mathbb{C}_\alpha K$. Thus, the excitations on this boundary are restricted to only the representations of $\mathbb{C}K$, without either $\mathbb{C}(R)$ or α appearing, which appears to defeat the purpose of the definition. It is not obvious to us that a cocycle can be included along these lines in a consistent manner. There are Hamiltonian models which use 'tube algebras' to include cocycles [BD19], but it is unclear how these could be incorporated into a system of measurements defining a quantum computer.

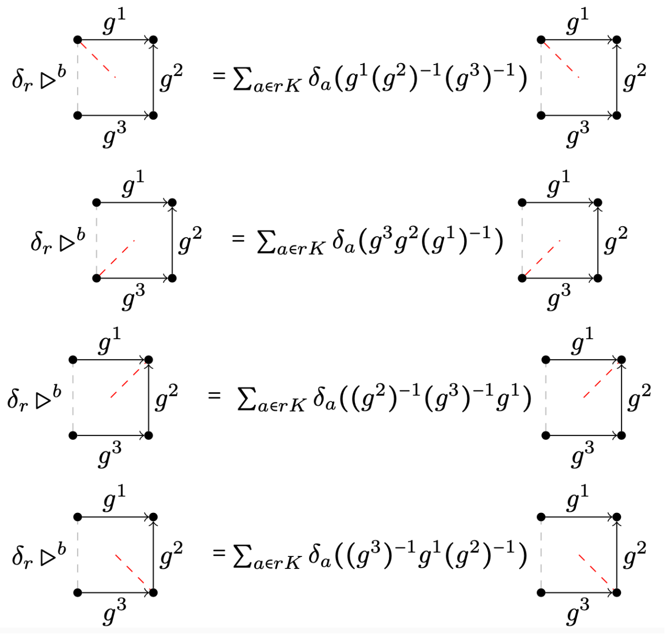
In quantum computational terms, in addition to the measurements in the bulk we now measure the operator $\sum_{\mathcal{O},\rho} p_{\mathcal{O},\rho} P_{(\mathcal{O},\rho)}{}^{\triangleright b}$ for distinct coefficients $p_{\mathcal{O},\rho} \in \mathbb{R}$ at all sites along the boundary.

Rough boundaries

We now consider the half-plane lattice with a different kind of boundary,



This time, there is an action of $\mathbb{C}K$ at the exterior vertex and an action of $\mathbb{C}(R)$ at the face at the boundary with an edge undefined. Again, the former is just the usual action of $\mathbb{C}K$ with three edges undefined, but the action of $\mathbb{C}(R)$ requires more care and is defined as


$\delta_r \triangleright^b$ g^1 g^2 g^3 $= \sum_{a\in rK} \delta_a(g^1(g^2)^{-1}(g^3)^{-1})$ g^1 g^2 g^3
$\delta_r \triangleright^b$ g^1 g^2 g^3 $= \sum_{a\in rK} \delta_a(g^3g^2(g^1)^{-1})$ g^1 g^2 g^3
$\delta_r \triangleright^b$ g^1 g^2 g^3 $= \sum_{a\in rK} \delta_a((g^2)^{-1}(g^3)^{-1}g^1)$ g^1 g^2 g^3
$\delta_r \triangleright^b$ g^1 g^2 g^3 $= \sum_{a\in rK} \delta_a((g^3)^{-1}g^1(g^2)^{-1})$ g^1 g^2 g^3

All but the second action are just clockwise rotations as in the bulk, but with the greyed-out edge missing from the δ-function. The second action goes counterclockwise in order to have an associated representation of $\Xi(R, K)$ at the bottom left. We have similar actions for other orientations of the lattice.

Remark 6.2.12. Although one can check that one has a representation of $\Xi(R, K)$ at each site using these actions and the action of $\mathbb{C}K$ defined before, this requires g_1 and g_2 on opposite sides of the δ-function, and g_1 and g_3 on opposite sides, respectively for the last two actions. This means that there is no way to get $\delta_e{\triangleright}^b$ to always be invariant under choice of site in the face. It is implicitly claimed in [CCW16, Rem 2.9] that $\delta_e{\triangleright}^b$ at a rough boundary can be defined in a way that depends only on the face, but this is not the case.

The integral actions at a boundary vertex v and at a site $s_0 = (v,p)$ of a rough boundary are then

$$A_2^b(v) := \Lambda_{\mathbb{C}K}{\triangleright}_v^b = \frac{1}{|K|}\sum_k k{\triangleright}_v^b, \quad B_2^b(v,p) := \Lambda_{\mathbb{C}(R)}{\triangleright}_{s_0}^b = \delta_e{\triangleright}_{s_0}^b$$

where the superscript b and subscript 2 label that these are at a rough boundary. In computational terms, we measure the operator $\sum_{\mathcal{O},\rho} p_{\mathcal{O},\rho} P_{(\mathcal{O},\rho)}{\triangleright}^b$ at each site along the boundary, as with smooth boundaries.

Unlike the smooth boundary case, there is not an action of, say, $\mathbb{C}(R){\bowtie}\mathbb{C}G$ at each site in the next layer into the bulk, with a boundary face but interior vertex. In particular, we do not have $B_2^b(v,p)A(v) = A(v)B_2^b(v,p)$ in general; this means that the model is not a commuting projector model. When the action at v is restricted to $\mathbb{C}K$ we recover an action of $\Xi(R,K)$ again. Similarly, if $K = \{e\}$, as we will use in Section 6.3 later, then the projectors commute and we recover a consistent definition of the vacuum in terms of projectors.

As with the bulk, the Hamiltonian incorporating the boundaries uses the actions of the integrals. We can accommodate both rough and smooth boundaries into the Hamiltonian. Let V, P be the set of vertices and faces in the bulk, S_1 the set of all sites (v,p) at smooth boundaries, and S_2 the same for rough boundaries. Then

$$\begin{aligned} H = &\sum_{v_i\in V}(1 - A(v_i)) + \sum_{p_i\in P}(1 - B(p_i)) \\ &+ \sum_{s_{b_1}\in S_1}((1 - A_1^b(s_{b_1}) + (1 - B_1^b(s_{b_1}))) + \sum_{s_{b_2}\in S_2}((1 - A_2^b(s_{b_2})) + (1 - B_2^b(s_{b_2})). \end{aligned}$$

If the rough boundaries have $K = \{e\}$, or otherwise $B_2^b(v,p)A(v) = A(v)B_2^b(v,p)$ we can pick out two vacuum states immediately:

$$|\mathrm{vac}_1\rangle := \prod_{v_i,s_{b_1},s_{b_2}} A(v_i)A_1^b(s_{b_1})A_2^b(s_{b_2}) \bigotimes_E e \tag{6.10}$$

and

$$|\mathrm{vac}_2\rangle := \prod_{p_i,s_{b_1},s_{b_2}} B(p_i)B_1^b(s_{b_1})B_2^b(s_{b_2}) \bigotimes_E \sum_{g\in G} g \tag{6.11}$$

where the tensor product runs over all edges in the lattice.

Remark 6.2.13. There is no need for two different boundaries to correspond to the same subgroup K, and the Hamiltonian can be defined accordingly. This principle is necessary when performing quantum computation by braiding 'defects', i.e. finite holes in the lattice, on the Abelian code [FMMC12], and also for the lattice surgery in Section 6.3. We do not write out this Hamiltonian in all its generality here, but its form is obvious.

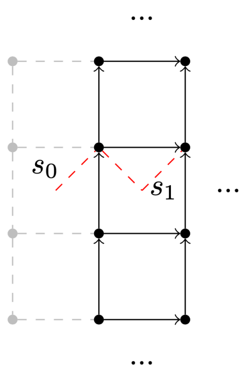
6.2.3 Quasiparticle condensation

Quasiparticles on the boundary correspond to irreps of $\Xi(R,K)$. It is immediate from Section 6.2.1 that when $\mathcal{O} = \{e\}$, we must have $r_0 = e, K^{r_0} = K$. We may choose the trivial representation of K and then we have $P_{e,1} = \Lambda_{\mathbb{C}(R)} \otimes \Lambda_{\mathbb{C}K}$. We say that this particular measurement outcome corresponds to the absence of nontrivial quasiparticles, as the states yielding this outcome are precisely the locally vacuum states with respect to the Hamiltonian. This set of quasiparticles on the boundary will not in general be the same as quasiparticles defined in the bulk, as ${}_{\Xi(R,K)}\mathcal{M} \not\simeq {}_{D(G)}\mathcal{M}$ for all nontrivial G.

Quasiparticles in the bulk can be created from a vacuum and moved using ribbon operators [Kit03], where the ribbon operators are seen as left and right module maps $D(G)^* \to \mathrm{End}(\mathcal{H})$, see [CM22]. Following [CCW16], we could similarly define a different set of ribbon operators for the boundary excitations, which use $\Xi(R,K)^*$ rather than $D(G)^*$. However, these have limited utility. For completeness we cover them in Appendix 22. Instead, for our purposes we will keep using the normal ribbon operators.

Such normal ribbon operators can extend to boundaries, still using Definition 6.1.2, so long as none of the edges involved in the definition are greyed-out. When a ribbon operator ends at a boundary site s, we are not concerned with equivariance with respect to the actions of $\mathbb{C}(G)$ and $\mathbb{C}G$ at s, as in Lemma 6.1.4. Instead we should calculate equivariance with respect to the actions of $\mathbb{C}(R)$ and $\mathbb{C}K$. We will study the matter in more depth in Section 6.4, but note that if $s,t \in R$ then unique factorisation means that $st = (s\cdot t)\tau(s,t)$ for unique elements $s\cdot t \in R$ and $\tau(s,t) \in K$. Similarly, if $y \in K$ and $r \in R$ then unique factorisation $yr = (y{\triangleright}r)(y{\triangleleft}r)$ defines $y{\triangleleft}r$ to be studied later.

Lemma 6.2.14. Let ξ be an open ribbon from s_0 to s_1, where s_0 is located at a smooth boundary, for example:



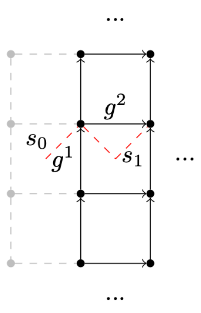
and where ξ begins at the specified orientation in the example, leading from s_0 into the bulk, rather than running along the boundary. Then

$$x{\triangleright}^b_{s_0} \circ F^{h,g}_\xi = F^{xhx^{-1},xg}_\xi \circ x{\triangleright}^b_{s_0}; \qquad \delta_r{\triangleright}^b_{s_0} \circ F^{h,g}_\xi = F^{h,g}_\xi \circ \delta_{s\cdot(y\triangleright r)}{\triangleright}^b_{s_0}$$

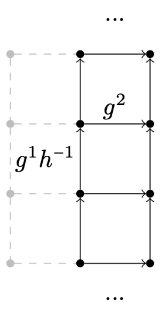
$\forall x \in K, r \in R, h, g \in G$, and where sy is the unique factorisation of h^{-1}.

Proof. The first is just the vertex action of $\mathbb{C}G$ restricted to $\mathbb{C}K$, with an edge greyed-out which does not influence the result. For the second, expand $\delta_r \triangleright^b_{s_0}$ and verify explicitly:

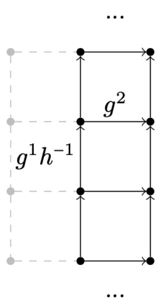
$\delta_r \triangleright^b_{s_0} \circ F^{h,g}_{\xi}$



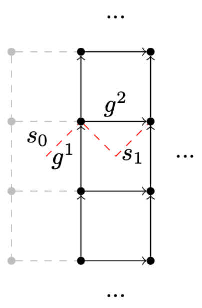
$= \sum_{a \in rK} \delta_a(h(g^1)^{-1})\delta_g(g^2)$



$= \sum_{a' \in h^{-1}rK} \delta_{a'}((g^1)^{-1})\delta_g(g^2)$

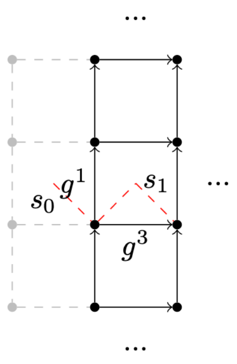


$= F^{h,g}_{\xi} \circ \delta_{s\cdot(y \triangleright r)} \triangleright^b_{s_0}$

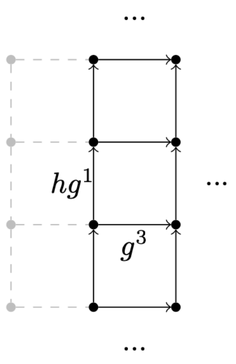


where we see $(s \cdot (y \triangleright r))K = s(y \triangleright r)\tau(s, y \triangleright r)^{-1}K = s(y \triangleright r)K = s(y \triangleright r)(y \triangleleft r)K = syrK = h^{-1}rK$. We check the other site as well:

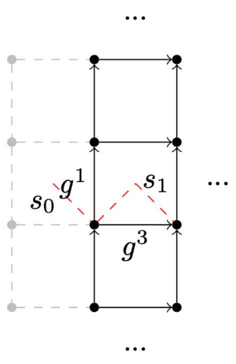
$\delta_r \triangleright^b_{s_0} \circ F^{h,g}_{\xi'}$



$= \sum_{a \in rK} \delta_a(hg^1)\delta_g((g^3)^{-1})$



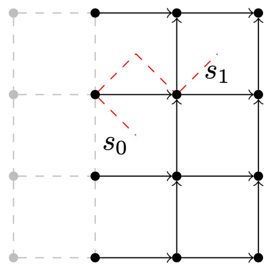
$= F^{h,g}_{\xi'} \circ \delta_{s\cdot(y \triangleright r)} \triangleright^b_{s_0}$



■

Remark 6.2.15. One might be surprised that the equivariance property holds for the latter case when s_0 is attached to the vertex at the bottom of the face, as in this case $\delta_r \triangleright^b_{s_0}$ confers a δ-function in the counterclockwise direction, different from the bulk. This is because the well-known equivariance properties in the bulk [Kit03] are not wholly correct, depending on orientation, as pointed out in [YCC22, Section 3.3]. We accommodated for this by specifying an orientation in Lemma 6.1.4.

Remark 6.2.16. We have a similar situation for a rough boundary, albeit we found only one orientation for which the same equivariance property holds, which is:



In the reverse orientiation, where the ribbon crosses downwards instead, equivariance is similar but with the introduction of an antipode. For other orientations we do not find an equivariance property at all. We do not know of a physical interpretation for this oddity.

As with the bulk, we can define an excitation space using a ribbon between the two endpoints s_0, s_1, although more care must be taken in the definition.

Lemma 6.2.17. Let $|\mathrm{vac}\rangle$ be a vacuum state on a half-plane Σ, where there is one smooth boundary beyond which there are no more edges. Let ξ be a ribbon between two endpoints s_0, s_1 where $s_0 = \{v_0, p_0\}$ is on the boundary and $s_1 = \{v_1, p_1\}$ is in the bulk, such that ξ interacts with the boundary only once, when crossing from s_0 into the bulk; it cannot cross back and forth multiple times. Let $|\psi^{h,g}\rangle := F^{h,g}_\xi |\mathrm{vac}\rangle$, and $\mathcal{T}_\xi(s_0, s_1)$ be the space with basis $|\psi^{h,g}\rangle$.

(1)$|\psi^{h,g}\rangle$ is independent of the choice of ribbon through the bulk between fixed sites s_0, s_1, so long as the ribbon still only interacts with the boundary at the chosen location.

(2)$\mathcal{T}_\xi(s_0, s_1) \subset \mathcal{H}$ inherits actions at disjoint sites s_0, s_1,

$$x \triangleright^b_{s_0} |\psi^{h,g}\rangle = |\psi^{xhx^{-1},xg}\rangle, \quad \delta_r \triangleright^b_{s_0} |\psi^{h,g}\rangle = \delta_{rK,hK} |\psi^{h,g}\rangle$$

$$f \triangleright_{s_1} |\psi^{h,g}\rangle = |\psi^{h,gf^{-1}}\rangle, \quad \delta_f \triangleright_{s_1} |\psi^{h,g}\rangle = \delta_{f,g^{-1}h^{-1}g} |\psi^{h,g}\rangle$$

where we use the module isomorphism $|\psi^{h,g}\rangle \mapsto \delta_h g$ to see the action at s_0 as a representation of $\Xi(R, K)$ on $D(G)$. In particular it is the restriction of the left regular representation of $D(G)$ to $\Xi(R, K)$, with inclusion map i from Lemma 6.2.6. The action at s_1 is the right regular representation of $D(G)$, as in the bulk.

Proof. (1) is the same as the proof in [CM22, Prop.3.10], with the exception that if the ribbon ξ' crosses the boundary multiple times it will incur an additional energy penalty from the Hamiltonian for each crossing, and thus $\mathcal{T}_{\xi'}(s_0, s_1) \neq \mathcal{T}_\xi(s_0, s_1)$ in general.

(2) This follows by the commutation rules in Lemma 6.2.14 and Lemma 4.2.8 respectively, using

$$x \triangleright^b_{s_0} |\mathrm{vac}\rangle = \delta_e \triangleright^b_{s_0} |\mathrm{vac}\rangle = |\mathrm{vac}\rangle; \quad f \triangleright_{s_1} |\mathrm{vac}\rangle = \delta_e \triangleright_{s_1} |\mathrm{vac}\rangle = |\mathrm{vac}\rangle$$

$\forall x \in K, f \in G$. For the hardest case we have

$$\begin{aligned}\delta_r \triangleright^b_{s_0} F^{h,g}|\mathrm{vac}\rangle &= F^{h,g}_\xi \circ \delta_{s\cdot(y\triangleright r)} \triangleright^b_{s_0}|\mathrm{vac}\rangle \\ &= F^{h,g}_\xi \delta_{s\cdot(y\triangleright r)K,K}|\mathrm{vac}\rangle \\ &= F^{h,g}_\xi \delta_{rK,hK}|\mathrm{vac}\rangle.\end{aligned}$$

For the restriction of the action at s_0 to $\Xi(R,K)$, we have that

$$\delta_r \cdot \delta_h g = \delta_{rK,hK}\delta_h g = \sum_{a\in rK} \delta_{a,h}\delta_h g = i(\delta_r)\delta_h g.$$

and $x \cdot \delta_h g = x\delta_h g = i(x)\delta_h g$. ■

In the bulk, the excitation space $\mathcal{L}(s_0, s_1)$ is totally independent of the ribbon ξ [Kit03, CM22], but we do not know of a similar property for $\mathcal{T}_\xi(s_0, s_1)$ when interacting with the boundary without the restrictions stated.

We explained in Section 6.2.1 how representations of $D(G)$ at sites in the bulk relate to those of $\Xi(R,K)$ in the boundary by functors in both directions. Physically, if we apply ribbon trace operators, that is operators of the form $W^{\mathcal{C},\pi}_\xi$, to the vacuum, then in the bulk we create exactly a quasiparticle of type $(\mathcal{C},\pi)$ and $(\mathcal{C}^*,\pi^*)$ at either end. Now let us include a boundary.

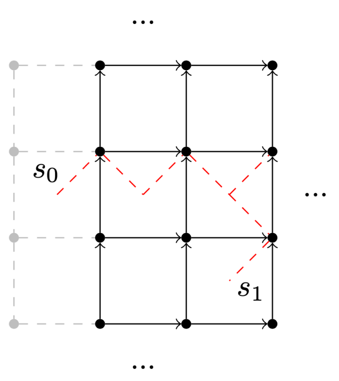
Definition 6.2.18. Given an irrep of $D(G)$ provided by $(\mathcal{C},\pi)$, we define the *boundary projection* $P_{i^*(\mathcal{C},\pi)} \in \Xi(R,K)$ by

$$P_{i^*(\mathcal{C},\pi)} = \sum_{(\mathcal{O},\rho)\ \mid\ n^{(\mathcal{O},\rho)}_{(\mathcal{C},\pi)} \neq 0} P_{(\mathcal{O},\rho)}$$

i.e. we sum over the projectors of all the types of irreps of $\Xi(R,K)$ contained in the restriction of the given $D(G)$ irrep.

It is clear that $P_{i^*(\mathcal{C},\pi)}$ is a projection as a sum of orthogonal projections.

Proposition 6.2.19. Let ξ be an open ribbon extending from an external site s_0 on a smooth boundary with associated algebra $\Xi(R,K)$ to a site s_1 in the bulk, for example:



Then

$$P_{(\mathcal{O},\rho)} \triangleright^b_{s_0} W^{\mathcal{C},\pi}_\xi|\mathrm{vac}\rangle = 0 \quad \text{iff} \quad n^{(\mathcal{O},\rho)}_{(\mathcal{C},\pi)} = 0.$$

In addition, we have

$$P_{i^*(\mathcal{C},\pi)} \triangleright^b_{s_0} W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle = W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle = W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle \triangleleft_{s_1} P_{(\mathcal{C},\pi)},$$

where we see the left action at s_1 of $P_{(\mathcal{C}^*,\pi^*)}$ as a right action using the antipode.

Proof. Under the isomorphism in Lemma 6.2.17 we have that $W^{\mathcal{C},\pi}_\xi|\mathrm{vac}\rangle \mapsto P_{(\mathcal{C},\pi)} \in D(G)$. For the first part we therefore have

$$P_{(\mathcal{O},\rho)} \triangleright^b_{s_0} W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle \mapsto i(P_{(\mathcal{O},\rho)})P_{(\mathcal{C},\pi)}$$

so the result follows from the last part of Lemma 6.2.7. Since the sum of projectors over the irreps of Ξ is 1, this then implies the second part:

$$W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle = \sum_{\mathcal{O},\rho} P_{(\mathcal{O},\rho)} \triangleright^b_{s_0} W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle = P_{i^*(\mathcal{C},\pi)} \triangleright^b_{s_0} W^{\mathcal{C},\pi}_\xi |\mathrm{vac}\rangle.$$

The action at s_1 is the same as for bulk ribbon operators. ■

The physical interpretation is that application of a ribbon trace operator $W^{\mathcal{C},\pi}_\xi$ to a vacuum state creates a quasiparticle at s_0 of all the types contained in $i^*(\mathcal{C},\pi)$, while still creating one of type $(\mathcal{C}^*,\pi^*)$ at s_1; this is called the *condensation* of $(\mathcal{C},\pi)$ at the boundary. While we used a smooth boundary in this example, the proposition applies equally to rough boundaries with the specified orientation in Remark 6.2.16 by similar arguments.

Note that by Proposition 6.2.19, it does not make sense to have irreps with multiplicities greater than 1 'living' at a site. Thus, if one were to take the purely categorical model and map irreps from ${}_{D(G)}\mathcal{M}$ to ${}_{\Xi(R,K)}\mathcal{M}$ this would yield unphysical representations at the boundary; one must then truncate all multiplicities to 1. In [CCW17] this feature of the model is called having *multiple condensation channels.*

Example 6.2.20. In the bulk, we take the $D(S_3)$ model. Then by Example 6.1.7, we have exactly 8 irreps in the bulk. At the boundary, we take $K = \{e, u\} = \mathbb{Z}_2$ with $R = \{e, uv, vu\}$. As per the table in Example 6.2.9 and Proposition 6.2.19 above, we then have for example that

$$(P_{\mathcal{O}_{0,-1}} + P_{\mathcal{O}_{1,1}}) \triangleright^b_{s_0} W^{\mathcal{C}_{1,-1}}_\xi |\mathrm{vac}\rangle = W^{\mathcal{C}_{1,-1}}_\xi |\mathrm{vac}\rangle = W^{\mathcal{C}_{1,-1}}_\xi |\mathrm{vac}\rangle \triangleleft_{s_1} P_{\mathcal{C}_{1,-1}}.$$

We can see this explicitly. Recall that

$$\Lambda_{\mathbb{C}(R)} \triangleright^b_{s_0} |\mathrm{vac}\rangle = \Lambda_{\mathbb{C}K} \triangleright^b_{s_0} |\mathrm{vac}\rangle = |\mathrm{vac}\rangle.$$

All other vertex and face actions give 0 by orthogonality. Then,

$$P_{\mathcal{O}_{0,-1}} = \frac{1}{2}\delta_e \otimes (e - u); \quad P_{\mathcal{O}_{1,1}} = (\delta_{uv} + \delta_{vu}) \otimes e$$

and

$$W^{\mathcal{C}_{1,-1}}_\xi = \sum_{c \in \{u,v,w\}} F^{c,e}_\xi - F^{c,c}_\xi$$

by Lemmas 6.2.4 and 6.1.6 respectively. For convenience, we break the calculation up into two parts, one for each projector. Throughout, we will make use of Lemma 6.2.14 to commute projectors through ribbon operators. First, we have that

$$\begin{aligned}
P_{\mathcal{O}_{0,-1}}\triangleright^b_{s_0} W^{\mathcal{C}_{1,-1}}_\xi|\mathrm{vac}\rangle &= \frac{1}{2}(\delta_e \otimes (e-u))\triangleright^b_{s_0} \sum_{c\in\{u,v,w\}} (F^{c,e}_\xi - F^{c,c}_\xi)|\mathrm{vac}\rangle \\
&= \frac{1}{2}\delta_e\triangleright^b_{s_0}[\sum_{c\in\{u,v,w\}} (F^{c,e}_\xi - F^{c,c}_\xi) - (F^{u,u}_\xi - F^{e,u}_\xi + F^{v,u}_\xi - F^{v,uv}_\xi + F^{w,u}_\xi - F^{w,vu}_\xi)]|\mathrm{vac}\rangle \\
&= \frac{1}{2}[(F^{u,e}_\xi - F^{u,u}_\xi)\delta_e\triangleright^b_{s_0} + (F^{v,e}_\xi - F^{v,v}_\xi)\delta_{vu}\triangleright^b_{s_0} + (F^{w,e}_\xi - F^{w,w}_\xi)\delta_{uv}\triangleright^b_{s_0} \\
&+ (F^{u,e}_\xi - F^{u,u}_\xi)\delta_e\triangleright^b_{s_0} + (F^{v,uv}_\xi - F^{v,u}_\xi)\delta_{vu}\triangleright^b_{s_0} + (F^{w,vu}_\xi - F^{w,u}_\xi)\delta_{uv}\triangleright^b_{s_0}]|\mathrm{vac}\rangle \\
&= (F^{u,e}_\xi - F^{u,u}_\xi)|\mathrm{vac}\rangle
\end{aligned}$$

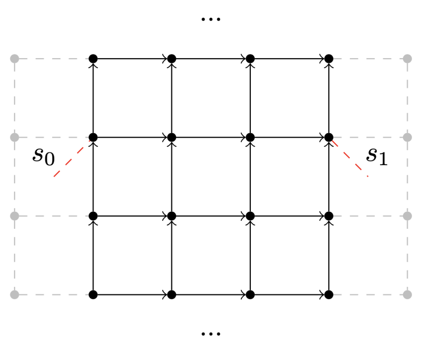
where we used the fact that $u = eu, v = vuu, w = uvu$ to factorise these elements in terms of R, K. Second,

$$\begin{aligned}
P_{\mathcal{O}_{1,1}}\triangleright^b_{s_0} W^{\mathcal{C}_{1,-1}}_\xi|\mathrm{vac}\rangle &= ((\delta_{uv} + \delta_{vu}) \otimes e)\triangleright^b_{s_0} \sum_{c\in\{u,v,w\}} (F^{c,e}_\xi - F^{c,c}_\xi)|\mathrm{vac}\rangle \\
&= (F^{v,e}_\xi - F^{v,v}_\xi + F^{w,e}_\xi - F^{w,w}_\xi)(\delta_e \otimes e)\triangleright^b_{s_0}|\mathrm{vac}\rangle \\
&= (F^{v,e}_\xi - F^{v,v}_\xi + F^{w,e}_\xi - F^{w,w}_\xi)|\mathrm{vac}\rangle.
\end{aligned}$$

The result follows immediately. All other boundary projections of $D(S_3)$ ribbon trace operators can be worked out in a similar way.

Remark 6.2.21. Proposition 6.2.19 does not tell us exactly how *all* ribbon operators in the quasiparticle basis are detected at the boundary, only the ribbon trace operators. A similar general formula for all ribbon operators is given in [CCW17, Thm 2.12] without proof.

Now, consider a lattice in the plane with two boundaries, namely to the left and right,



Recall that a lattice on an infinite plane admits a single ground state $|\mathrm{vac}\rangle$ as explained in[CM22]. However, in the present case, we may be able to also use ribbon operators in the quasiparticle basis extending from one boundary, at s_0 say, to the other, at s_1 say, such that no quasiparticles are detected at either end. These ribbon operators do not form a closed, contractible loop, as all undetectable ones do in the bulk; the corresponding states $|\psi\rangle$ are ground states and the vacuum has increased degeneracy. We can similarly induce additional degeneracy of excited states. This justifies the term *gapped boundaries*, as the boundaries give rise to additional states with energies that are 'gapped'; that is, they have a finite energy difference Δ (which may be zero) independently of the width of the lattice.

6.3 Lattice surgery with patches

For any nontrivial group, G there are always at least two distinct choices of boundary conditions, namely with $K = \{e\}$ and $K = G$ respectively. In these cases, we necessarily have $R = G$ and $R = \{e\}$ respectively.

Considering $K = \{e\}$ on a smooth boundary, we can calculate that $A_1^b(v) = \mathrm{id}$ and $B_1^b(s)g = \delta_{e,g}g$, for g an element corresponding to the single edge associated with the boundary site s. This means that after performing the measurements at a boundary, these edges are totally constrained and not part of the large entangled state incorporating the rest of the lattice, and hence do not contribute to the model whatsoever. If we remove these edges then we are left with a rough boundary, in which all edges participate, and therefore we may consider the $K = \{e\}$ case to imply a rough boundary. A similar argument applies for $K = G$ when considered on a rough boundary, which has $A_2^b(v)g = A(v)g = \frac{1}{|G|}\sum_k kg = \frac{1}{|G|}\sum_k k$ for an edge with state g and $B_2^b(s) = \mathrm{id}$. $K = G$ therefore naturally corresponds instead to a smooth boundary, as otherwise the outer edges are totally constrained by the projectors. From now on, we will accordingly use smooth to refer always to the $K = G$ condition, and rough for $K = \{e\}$.

These boundary conditions are convenient because the condensation of bulk excitations to the vacuum at a boundary can be partially worked out in the group basis. For $K = \{e\}$, it is easy to see that the ribbon operators which are undetected at the boundary (and therefore leave the system in a vaccum state) are exactly those of the form $F_\xi^{e,g}$, for all $g \in G$, as any nontrivial h in $F_\xi^{h,g}$ will be detected by the boundary face projectors. This can also be worked out representation-theoretically using Proposition 6.2.8.

Lemma 6.3.1. Let $K = \{e\}$. Then the multiplicity of an irrep $(\mathcal{C}, \pi)$ of $D(G)$ with respect to the trivial representation of $\Xi(G, \{e\})$ is

$$n_{(\mathcal{C},\pi)}^{(\{e\},1)} = \delta_{\mathcal{C},\{e\}}\mathrm{dim}(W_\pi)$$

Proof. Applying Proposition 6.2.8 in the case where V_i is trivial, we start with

$$n_{(\mathcal{C},\pi)}^{(\{e\},1)} = \frac{|G|}{|\mathcal{C}||G^{c_0}|}\sum_{c\in\mathcal{C}\cap\{e\}} |\{e\}^c| n_{1,\tilde{\pi}}$$

where $\mathcal{C}\cap\{e\} = \{e\}$ iff $\mathcal{C} = \{e\}$, or otherwise $\emptyset$. Also, $\tilde{\pi} = \oplus_{\mathrm{dim}(W_\pi)}(\{e\}, 1)$, and if $\mathcal{C} = \{e\}$ then $|G^{c_0}| = |G|$. ■

The factor of $\mathrm{dim}(W_\pi)$ in the r.h.s. implies that there are no other terms in the decomposition of $i^*(\{e\}, \pi)$. In physical terms, this means that the trace ribbon operators $W_\xi^{e,\pi}$ are the only undetectable trace ribbon operators, and any ribbon operators which do not lie in the block associated to (e, π) are detectable. In fact, in this case we have a further property which is that all ribbon operators in the chargeon sector are undetectable, as by equation (6.5) chargeon sector ribbon operators are Fourier isomorphic to those of the form $F_\xi^{e,g}$ in the group basis. In the more general case of a rough boundary for an arbitrary choice of $\Xi(R, K)$ the orientation of the ribbon is important for using the representation-theoretic argument. When $K = \{e\}$, for $F_\xi^{e,g}$ one can check that regardless of orientation the rough boundary version of Proposition 6.2.17 applies.

The $K = G$ case is slightly more complicated:

Lemma 6.3.2. Let $K = G$. Then the multiplicity of an irrep $(\mathcal{C}, \pi)$ of $D(G)$ with respect to the trivial representation of $\Xi(\{e\}, G)$ is

$$n^{(\{e\},1)}_{(\mathcal{C},\pi)} = \delta_{\pi,1}$$

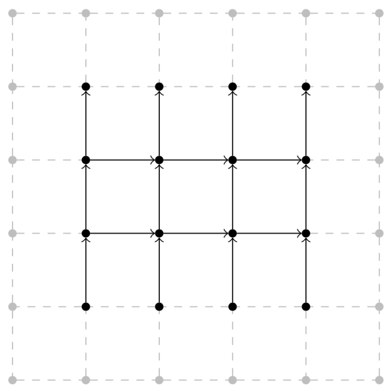
Proof. We start with

$$n^{(\{e\},1)}_{(\mathcal{C},\pi)} = \frac{1}{|\mathcal{C}||G^{c_0}|}\sum_{c\in\mathcal{C}} |G^c| n_{1,\tilde{\pi}}.$$

Now, $K^{r,c} = G^c$ and so $\tilde{\pi} = \pi$, giving $n_{1,\tilde{\pi}} = \delta_{1,\pi}$. Then $\sum_{c\in\mathcal{C}} |G^c| = |\mathcal{C}||G^{c_0}|$. ■

This means that the only undetectable ribbon operators between smooth boundaries are those in the fluxion sector, i.e. those with assocated irrep $(\mathcal{C}, 1)$. However, there is no factor of $|\mathcal{C}|$ on the r.h.s. and so the decomposition of $i^*(\mathcal{C}, 1)$ will generally have additional terms other than just $(\{e\}, 1)$ in ${}_{\Xi(\{e\},G)}\mathcal{M}$. As a consequence, a fluxion trace ribbon operator $W^{\mathcal{C},1}_{\zeta}$ between smooth boundaries is undetectable iff its associated conjugacy class is a singlet, say $\mathcal{C} = \{c_0\}$, and thus $c_0 \in Z(G)$, the centre of G.

Definition 6.3.3. A *patch* is a finite rectangular lattice segment with two opposite smooth sides, each equipped with boundary conditions $K = G$, and two opposite rough sides, each equipped with boundary conditions $K = \{e\}$, for example:



One can alternatively define patches with additional sides, such as in [Lit19], or with other boundary conditions which depend on another subgroup K and transversal R, but we find this definition convenient. Note that our definition does not put conditions on the size of the lattice; the above diagram is just a conveniently small and yet nontrivial example.

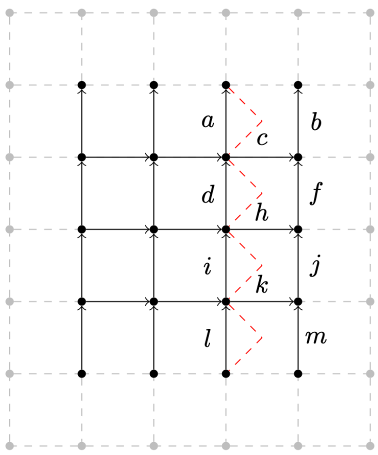
We would like to characterise the vacuum space $\mathcal{H}_{\mathrm{vac}}$ of the patch. To do this, let us begin with $|\mathrm{vac}_1\rangle$ from equation (6.10), and denote $|e\rangle_L := |\mathrm{vac}_1\rangle$. This is the *logical zero state* of the patch. We will use this as a reference state to calculate other states in $\mathcal{H}_{\mathrm{vac}}$.

Now, for any other state $|\psi\rangle$ in $\mathcal{H}_{\mathrm{vac}}$, there must exist some linear map $D \in \mathrm{End}(\mathcal{H}_{\mathrm{vac}})$ such that $D|e\rangle_L = |\psi\rangle$, and thus if we can characterise the algebra of linear maps $\mathrm{End}(\mathcal{H}_{\mathrm{vac}})$, we automatically characterise $\mathcal{H}_{\mathrm{vac}}$. To help with this, we have the following useful property:

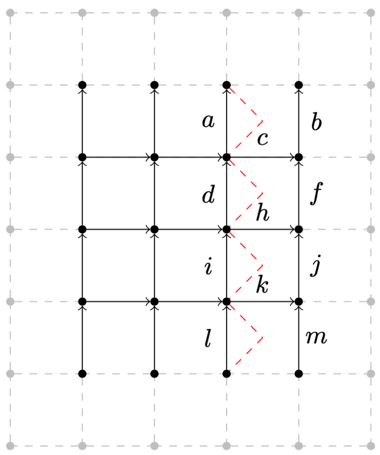
Lemma 6.3.4. Let $F^{e,g}_{\xi}$ be a ribbon operator for some $g \in G$, with ξ extending from the top rough boundary to the bottom rough boundary. Then the endpoints of ξ may be moved along the rough boundaries with $G = \{e\}$ boundary conditions while leaving the action invariant on any vacuum state.

Proof. We explain this on an example patch with initial state $|\psi\rangle \in \mathcal{H}_{\text{vac}}$ and a ribbon ξ. $|\psi\rangle$ is a linear sum of terms of the following form, and so while this proof uses only one term it applies to any $|\psi\rangle \in \mathcal{H}_{\text{vac}}$.

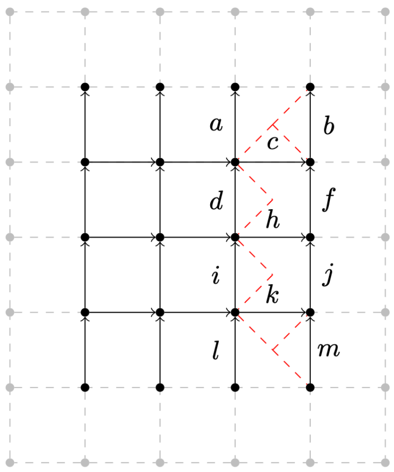
$F_{\xi}^{e,g}$



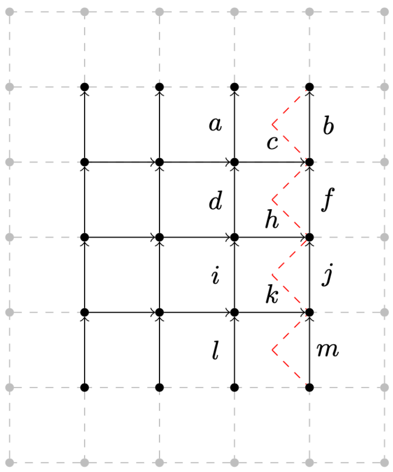
$= \delta_g(a^{-1}d^{-1}i^{-1}l^{-1})$



$= F_{\xi'}^{e,g}$



$= F_{\xi''}^{e,g}$



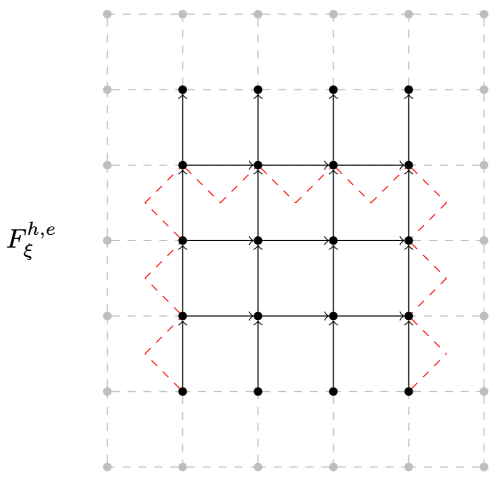
using the fact that $a = cb$ and $m = lk$ by the definition of $\mathcal{H}_{\text{vac}}$ for the second equality. Thus, we see that the ribbon through the bulk may be deformed as usual. As the only new component of the proof concerned the endpoints, we see that this property holds regardless of the size of the patch. ■

One can calculate in particular that $F_{\xi}^{e,g}|e\rangle_L = \delta_{e,g}|e\rangle_L$, which we will prove more generally later. The undetectable ribbon operators between the smooth boundaries are of the form

$$W_{\xi}^{\mathcal{C},1} = \sum_{n \in G} F_{\zeta}^{c_0,n}$$

when $\mathcal{C} = \{c_0\}$ by Lemma 6.3.2, hence $G^{c_0} = G$. Technically, this lemma only tells us the ribbon trace operators which are undetectable, but in the present case none of the individual component operators are undetectable, only the trace operators. There are thus exactly $|Z(G)|$ orthogonal undetectable ribbon operators between smooth boundaries. These do not play an important role, but we describe them to characterise the operator algebra on $\mathcal{H}_{\text{vac}}$. They obey a similar rule as Lemma 6.3.4, which one can check in the same way.

In addition to the ribbon operators between sides, we also have undetectable ribbon operators between corners on the lattice. These corners connect smooth and rough boundaries, and thus careful application of specific ribbon operators can avoid detection from either face or vertex measurements,



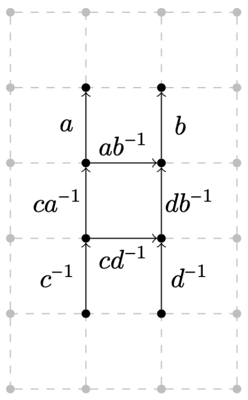
where one can check that these do indeed leave the system in a vacuum using familiar arguments about $B(p)$ and $A(v)$. We could equally define such operators extending from either left corner to either right corner, and they obey the discrete isotopy laws as in the bulk. If we apply $F_\xi^{h,g}$ for any $g \neq e$ then we have $F_\xi^{h,g}|\psi\rangle = 0$ for any $|\psi\rangle \in \mathcal{H}_{\mathrm{vac}}$, and so these are the only ribbon operators of this form.

Remark 6.3.5. Corners of boundaries are algebraically interesting themselves, and can be used for quantum computation, but for brevity we skim over them. See e.g. [Bom10, BKLW17] for details.

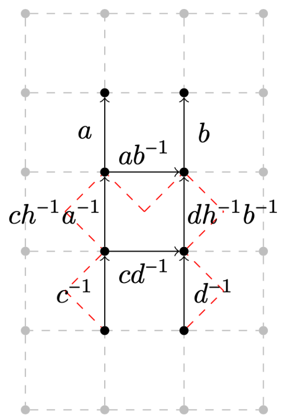
These corner to corner, left to right and top to bottom ribbon operators span $\mathrm{End}(\mathcal{H}_{\mathrm{vac}})$, the linear maps which leave the system in vacuum. Due to Lemma 6.1.5, all other linear maps must decompose into ribbon operators, and these are the only ribbon operators in $\mathrm{End}(\mathcal{H}_{\mathrm{vac}})$ up to linearity.

As a consequence, we have well-defined patch states $|h\rangle_L := \sum_g F_\xi^{h,g}|e\rangle_L$ for each $h \in G$, where ξ is any ribbon extending from the bottom left corner to right. Now, working explicitly on the small patch below, we have

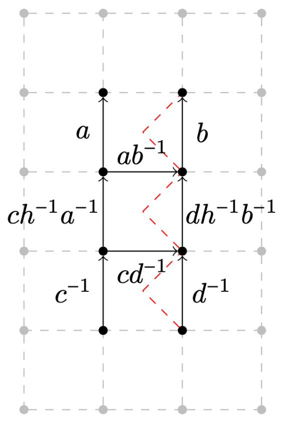
$$|e\rangle_L = \frac{1}{|G|^4}\sum_{a,b,c,d}$$



to start with, then:

$|h\rangle_L = \frac{1}{|G|^4}\sum_{a,b,c,d}$ 

It is easy to see that we may always write $|h\rangle_L$ in this manner, for an arbitrary size of patch. Now, ribbon operators which are undetectable when ξ extends from bottom to top are those of the form $F_\xi^{e,g}$, for example

$F_\xi^{e,g}|h\rangle_L = \frac{1}{|G|^4}\sum_{a,b,c,d}\delta_g(h)$ 

and so $F_\xi^{e,g}|h\rangle_L = \delta_{g,h}|h\rangle_L$, where again if we take a larger patch all additional terms will clearly cancel. Lastly, undetectable ribbon operators for a ribbon ζ extending from left to right are exactly those of the form $\sum_{n\in G}F_\zeta^{c_0,n}$ for any $c_0 \in Z(G)$. One can check that $|c_0h\rangle_L = \sum_{n\in G}F_\zeta^{c_0,n}|h\rangle_L$, thus these give us no new states in $\mathcal{H}_{\text{vac}}$.

Lemma 6.3.6. For a patch with the $D(G)$ model in the bulk, $\dim(\mathcal{H}_{\text{vac}}) = |G|$.

Proof. By the above characterisation of undetectable ribbon operators, the states $\{|h\rangle_L\}_{h\in G}$ span $\dim(\mathcal{H}_{\text{vac}})$. The result then follows from the adjointness of ribbon operators, which means that the states $\{|h\rangle_L\}_{h\in G}$ are orthogonal. ∎

We can also work out that for $|\text{vac}_2\rangle$ from equation (6.11), we have $|\text{vac}_2\rangle = \sum_h |h\rangle_L$. More generally:

Corollary 6.3.7. $\mathcal{H}_{\text{vac}}$ has an alternative basis with states $|\pi; i, j\rangle_L$, where π is an irreducible representation of G and i, j are indices such that $0 \leq i, j < \dim(V_\pi)$. We call this the quasiparticle basis of the patch.

Proof. First, use the nonabelian Fourier transform on the ribbon operators $F_\xi^{e,g}$, so we have $F_\xi'^{e,\pi;i,j} = \sum_{n\in G}\pi(n^{-1})_{ji}F_\xi^{e,n}$. If we start from the reference state $|1; 0, 0\rangle_L := \sum_h |h\rangle_L =$

$|\mathrm{vac}_2\rangle$ and apply these operators with ξ from bottom to top of the patch then we get

$$|\pi; i,j\rangle_L = F_\xi^{\prime e,\pi;i,j}|1;0,0\rangle_L = \sum_{n\in G}\pi(n^{-1})_{ji}|n\rangle_L$$

which are orthogonal. Now, as $\sum_{\pi\in\hat{G}}\sum_{i,j=0}^{\dim(V_\pi)} = |G|$ and we know $\dim(\mathcal{H}_{\mathrm{vac}}) = |G|$ by the previous Lemma 6.3.6, $\{|\pi;i,j\rangle_L\}_{\pi,i,j}$ forms a basis of $\dim(\mathcal{H}_{\mathrm{vac}})$. ■

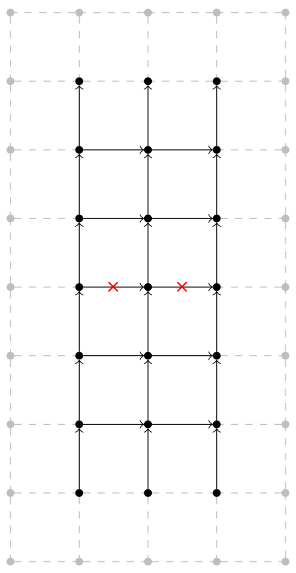
Remark 6.3.8. Kitaev models are designed in general to detect and correct for errors. The minimum number of component Hilbert spaces, that is copies of $\mathbb{C}G$ on edges, for which simultaneous errors will undetectably change the logical state and cause errors in the computation is called the 'code distance' d in the language of quantum codes. For the standard method of computation using nonabelian anyons [Kit03], data is encoded using excited states, which are states with nontrivial quasiparticles at certain sites. The code distance can then be extremely small, and constant in the size of the lattice, as the smallest errors need only take the form of ribbon operators winding round a single quasiparticle at a site. This is no longer the case when encoding data in vacuum states on patches, as the only logical operators are specific ribbon operators extending from top to bottom, left to right or corner to corner. The code distance, and hence error resilience, of any vacuum state of the patch therefore increases linearly with the width of the patch as it is scaled, and so the square root of the number n of component Hilbert spaces in the patch, that is $d \sim \sqrt{(}n)$.

6.3.1 Nonabelian lattice surgery

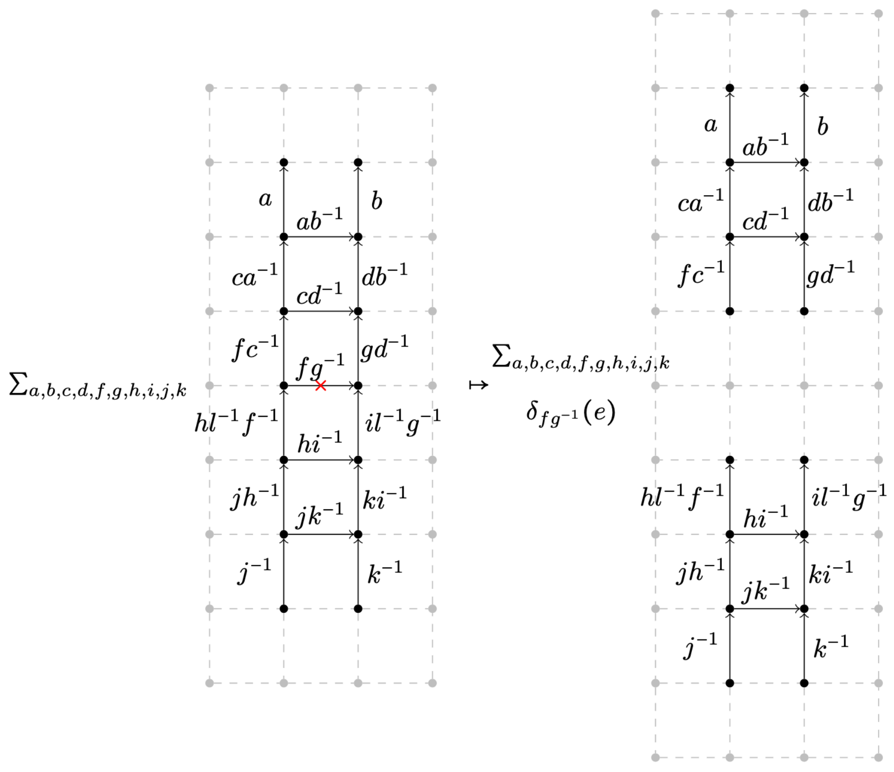
Lattice surgery was invented as a method of fault-tolerant computation with the qubit, i.e. $\mathbb{C}\mathbb{Z}_2$, surface code [HFDM12]. The first author generalised it to qudit models using $\mathbb{C}\mathbb{Z}_d$ in [Cow22], and gave a fresh perspective on lattice surgery as 'simulating' the Hopf algebras $\mathbb{C}\mathbb{Z}_d$ and $\mathbb{C}(\mathbb{Z}_d)$ on the logical space $\mathcal{H}_{\mathrm{vac}}$ of a patch. In this section, we prove that lattice surgery generalises to arbitrary finite group models, and 'simulates' $\mathbb{C}G$ and $\mathbb{C}(G)$ in a similar way. Throughout, we assume that the projectors $A(v)$ and $B(p)$ may be performed deterministically for simplicity. In Appendix 23 we discuss the added complication that in practice we may only perform measurements which yield projections nondeterministically.

Remark 6.3.9. When proving the linear maps that nonabelian lattice surgeries yield, we will use specific examples, but the arguments clearly hold generally. For convenience, we will also tend to omit normalising scalar factors, which do not impact the calculations as the maps are $\mathbb{C}$-linear.

Let us begin with a large rectangular patch. We now remove a line of edges from left to right by projecting each one onto e:

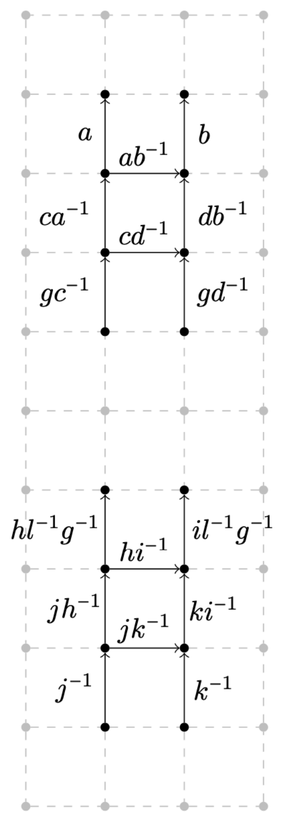


We call this a *rough split*, as we create two new rough boundaries. We no longer apply $A(v)$ to the vertices which have had attached edges removed. If we start with a small patch in the state $|l\rangle_L$ for some $l \in G$ then we can explicitly calculate the linear map.



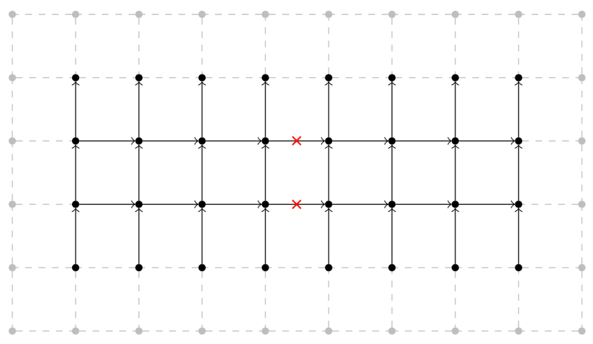
where we have separated the two patches afterwards for clarity, showing that they have two separate vacuum spaces. We then have that the last expression is

$$\cdots = \quad \sum_{a,b,c,d,g,h,i,j,k}$$



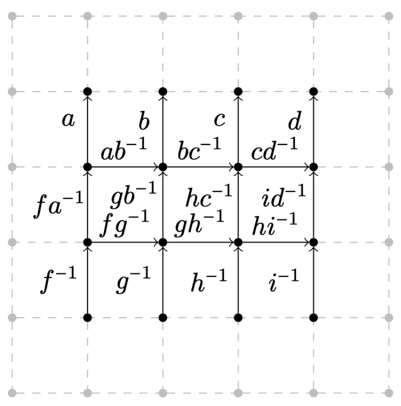
Observe the factors of g in particular. The state is therefore now $\sum_g |g^{-1}\rangle_L \otimes |gl\rangle_L$, where the l.h.s. of the tensor product is the Hilbert space corresponding to the top patch, and the r.h.s. to the bottom. A change of variables gives $\sum_g |g\rangle_L \otimes |g^{-1}l\rangle_L$, the outcome of comultiplication of $\mathbb{C}(G)$ on the logical state $|l\rangle_L$ of the original patch.

Similarly, we can measure out a line of edges from bottom to top, for example

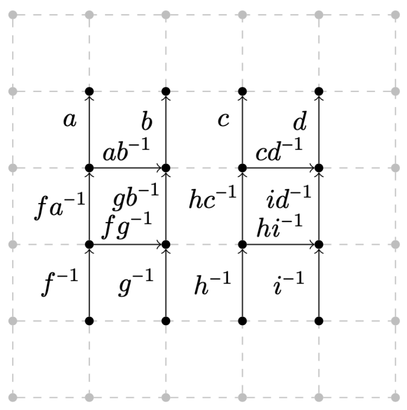


We call this a *smooth split*, as we create two new smooth boundaries. Each deleted edge is projected into the state $\frac{1}{|G|}\sum_g g$. We also cease measurement of the faces which have had edges removed, and so we end up with two adjacent but disjoint patches. Working on a small example, we start with $|e\rangle_L$:

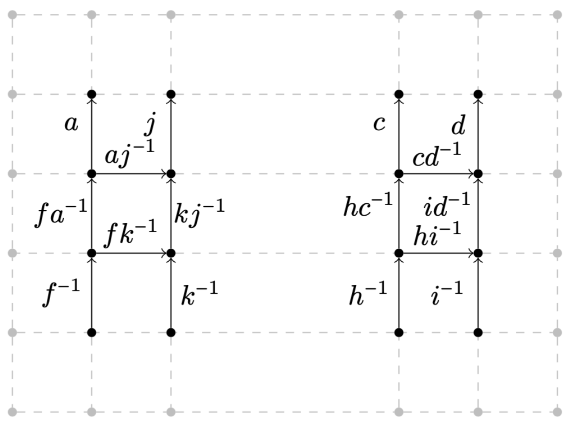
$$\sum_{a,b,c,d,f,g,h,i}$$



$$\mapsto \sum_{a,b,c,d,f,g,h,i,j,k} \delta_{bc^{-1}}(j)\delta_{gh^{-1}}(k)$$



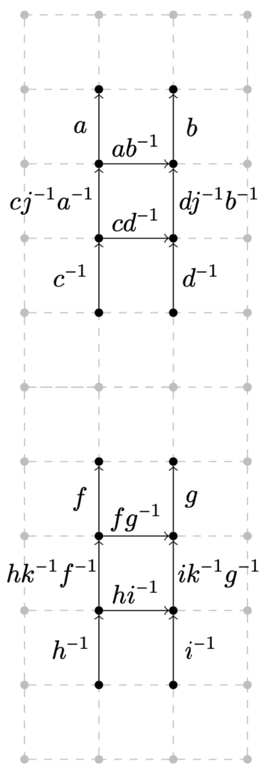
$$= \sum_{a,c,d,f,h,i,j,k}$$



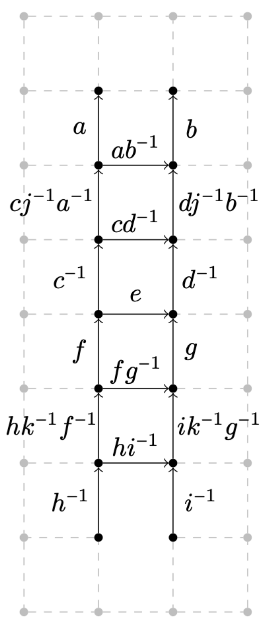
where in the last step we have taken $b \mapsto jc$, $g \mapsto kh$ from the δ-functions and then a change of variables $j \mapsto jc^{-1}$, $k \mapsto kh^{-1}$ in the summation. Thus, we have ended with two disjoint patches, each in state $|e\rangle_L$. One can see that this works for any $|h\rangle_L$ in exactly the same way, and so the smooth split linear map is $|h\rangle_L \mapsto |h\rangle_L \otimes |h\rangle_L$, the comultiplication of $\mathbb{C}G$.

The opposite of splits are merges, whereby we take two disjoint patches and introduce edges to bring them together to a single patch. For the rough merge below, say we start with the basis states $|k\rangle_L$ and $|j\rangle_L$ on the bottom and top. First, we introduce an additional joining edge in the state e.

$\sum_{a,b,c,d,f,g,h,i}$

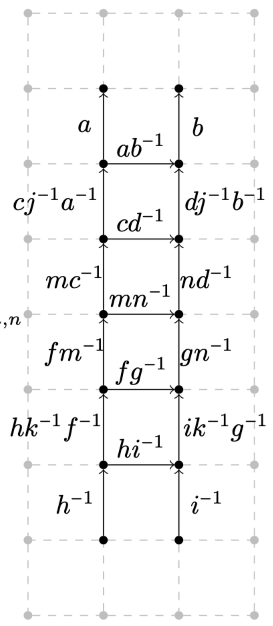


$\mapsto \sum_{a,b,c,d,f,g,h,i}$

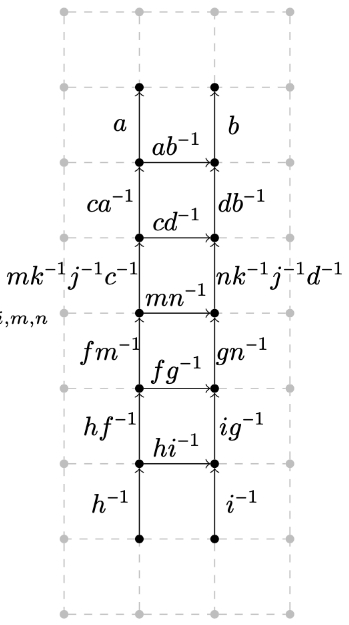


This state $|\psi\rangle$ automatically satisfies $B(p)|\psi\rangle = |\psi\rangle$ everywhere. But it does not satisfy the conditions on vertices, so we apply $A(v)$ to the two vertices adjacent to the newest edge. Then we have the last expression

$= \sum_{a,b,c,d,f,g,h,i,m,n}$

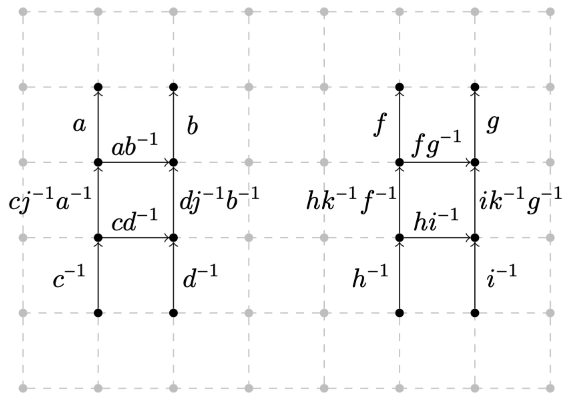


which by performing repeated changes of variables yields

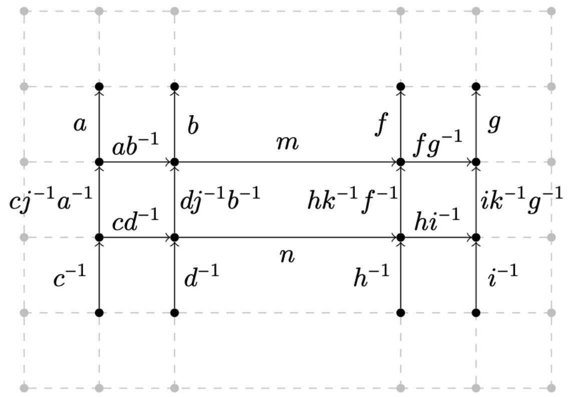
$= \sum_{a,b,c,d,f,g,h,i,m,n}$


Thus the rough merge yields the map $|j\rangle_L \otimes |k\rangle_L \mapsto |jk\rangle_L$, the multiplication of $\mathbb{C}G$, where again the tensor factors are in order from top to bottom.

Similarly, we perform a smooth merge with the states $|j\rangle_L, |k\rangle_L$ as

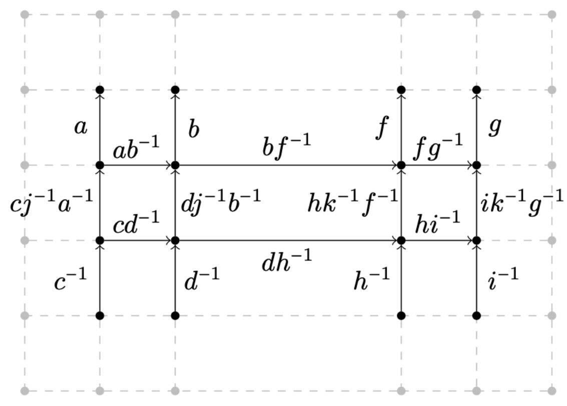
$\sum_{a,b,c,d,f,g,h,i}$


We introduce a pair of edges connecting the two patches, each in the state $\sum_m m$,

$\sum_{a,b,c,d,f,g,h,i,m,n}$


The resultant patch automatically satisfies the conditions relating to $A(v)$, but we must apply $B(p)$ to the freshly created faces to acquire a state in $\mathcal{H}_{\text{vac}}$, giving

$\sum_{a,b,c,d,f,g,h,i} \delta_{j,k}$



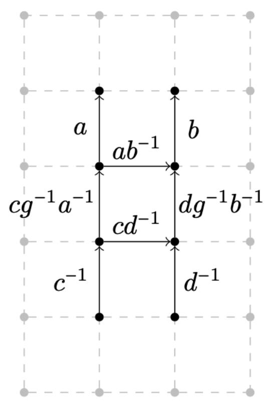
where the $B(p)$ applications introduced the δ-functions

$$\delta_e(bf^{-1}m^{-1}), \quad \delta_e(dh^{-1}n^{-1}), \quad \delta_e(dj^{-1}b^{-1}bf^{-1}fkh^{-1}hd^{-1}) = \delta_e(j^{-1}k).$$

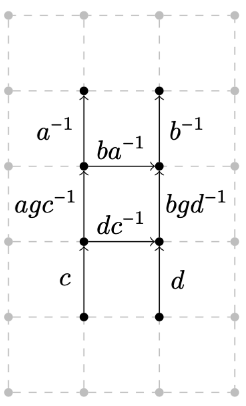
In summary, the linear map on logical states is evidently $|j\rangle_L \otimes |k\rangle_L \mapsto \delta_{j,k}|j\rangle_L$, the multiplication of $\mathbb{C}(G)$.

The units of $\mathbb{C}G$ and $\mathbb{C}(G)$ are given by the states $|e\rangle_L$ and $|1;0,0\rangle_L$ respectively. The counits are given by the maps $|g\rangle_L \mapsto 1$ and $|g\rangle_L \mapsto \delta_{g,e}$ respectively. The logical antipode S_L is given by applying the antipode to each edge individually, i.e. inverting all group elements. For example:

$S_L|g\rangle_L = S_L\frac{1}{|G|^4}\sum_{a,b,c,d}$

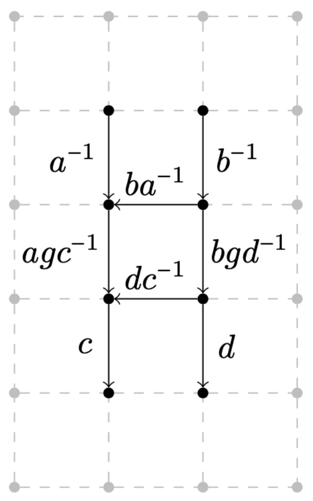


$= \frac{1}{|G|^4}\sum_{a,b,c,d}$



This state is now no longer in the original $\mathcal{H}_{\text{vac}}$, so to compensate we must modify the lattice. We flip all arrows in the lattice:

$$\dots = \frac{1}{|G|^4}\sum_{a,b,c,d}$$



This amounts to exchanging left and right regular representations, and redefining the Hamiltonian accordingly. In the resultant new vacuum space, the state is now $|g^{-1}\rangle_L = F_\xi^{e,g^{-1}}|e\rangle_L$, with ξ running from the bottom left corner to bottom right as previously.

Remark 6.3.10. This trick of redefining the vacuum space is employed in [HFDM12] to perform logical Hadamards, although in their case the lattice is rotated by $\pi/2$, and the edges are directionless as the model is restricted to $\mathbb{C}\mathbb{Z}_2$.

Thus, we have all the ingredients of the Hopf algebras $\mathbb{C}G$ and $\mathbb{C}(G)$ on the same vector space $\mathcal{H}_{\mathrm{vac}}$. For applications, one should like to know which quantum computations can be performed using these algebras (ignoring the subtlety with nondeterministic projectors). Recall that a quantum computer is called approximately universal if for any target unitary U and desired accuracy $\epsilon \in \mathbb{R}$, the computer can perform a unitary V such that $||V - U|| \leq \epsilon$, i.e. the operator norm error of V from U is no greater than ϵ.

We believe that when the computer is equipped with just the states $\{|h\rangle_L\}_{h\in G}$ and the maps from lattice surgery above then one cannot achieve approximately universal computation [Gog22], but leave the proof to a further paper. If we also have access to all matrix algebra states $|\pi; i, j\rangle_L$ as defined in Corollary 6.3.7, we do not know whether the model of computation is then universal for some choice of G, and we do not know whether these states can be prepared efficiently. In fact, how these states are defined depends on a choice of basis for each irrep, so whether it is universal may depend not only on the choice of G but also choices of basis. The computational power of nonabelian lattice surgery is an interesting question for future work.

6.4 Quasi-Hopf algebra structure of $\Xi(R, K)$

We now return to our boundary algebra Ξ. It is known that Ξ has a great deal more structure. This structure generalises a well-known bicrossproduct Hopf algebra construction for when a finite group G factorises as $G = RK$ into two subgroups R, K. Then each acts on the set of the other to form a *matched pair of actions* $\triangleright, \triangleleft$ and we use $\triangleright$ to make a cross product algebra $\mathbb{C}K{\triangleright\!\!\!<}\mathbb{C}(R)$ (which has the same form as our algebra Ξ except that we have chosen to flip the tensor factors) and $\triangleleft$ to make a cross product coalgebra $\mathbb{C}K{>\!\!\!\blacktriangleleft}\mathbb{C}(R)$. These fit together to form a bicrossproduct Hopf algebra $\mathbb{C}K{\triangleright\!\!\!\blacktriangleleft}\mathbb{C}(R)$. This construction has been used in the Lie group version to construct quantum Poincaré groups for quantum spacetimes[Maj95]. In this section we describe the more general $\Xi(R,K)$ and in more detail than we have seen elsewhere. In physical terms the 'coproduct' $\Delta:\Xi\to\Xi\otimes\Xi$ explicitly controls the tensor product of representations, not only up to isomorphism, and the antialgebra 'antipode' $S:\Xi\to\Xi$ explicitly controls left-right conversion and hence adjunction of representations.

In [Beg03] was considered the more general case where rather than R and K both being subgroups we are just given a single subgroup $K\subseteq G$ and a choice of transversal R with the group identity $e\in R$. As we noted, we still have unique factorisation $G=RK$ but in general R need not be a group. We can still follow the same steps. First of all, unique factorisation entails that $R\cap K=\{e\}$. It also implies maps

$$\triangleright:K\times R\to R,\quad \triangleleft:K\times R\to K,\quad \cdot:R\times R\to R,\quad \tau:R\times R\to K$$

defined by

$$xr=(x{\triangleright}r)(x{\triangleleft}r),\quad rs=r\cdot s\tau(r,s)$$

for all $x\in K, r,s\in R$, but this time these inherit the properties

$$\begin{gathered}(xy){\triangleright}r=x{\triangleright}(y{\triangleright}r),\quad e{\triangleright}r=r,\\ x{\triangleright}(r\cdot s)=(x{\triangleright}r)\cdot((x{\triangleleft}r){\triangleright}s),\quad x{\triangleright}e=e,\end{gathered}\tag{6.12}$$

$$\begin{gathered}(x{\triangleleft}r){\triangleleft}s=\tau\left(x{\triangleright}r,(x{\triangleleft}r){\triangleright}s\right)^{-1}(x{\triangleleft}(r\cdot s))\,\tau(r,s),\quad x{\triangleleft}e=x,\\ (xy){\triangleleft}r=(x{\triangleleft}(y{\triangleright}r))(y{\triangleleft}r),\quad e{\triangleleft}r=e,\end{gathered}\tag{6.13}$$

$$\begin{gathered}\tau(r,s\cdot t)\tau(s,t)=\tau\left(r\cdot s,\tau(r,s){\triangleright}t\right)\left(\tau(r,s){\triangleleft}t\right),\quad \tau(e,r)=\tau(r,e)=e,\\ r\cdot(s\cdot t)=(r\cdot s)\cdot(\tau(r,s){\triangleright}t),\quad r\cdot e=e\cdot r=r\end{gathered}\tag{6.14}$$

for all $x,y\in K$ and $r,s,t\in R$. We see from (6.12) that $\triangleright$ is indeed an action (we have been using it in preceding sections) but $\triangleleft$ in (6.13) is only only up to τ (termed in [KM10A] a 'quasiaction'). Both $\triangleright,\triangleleft$ 'act' almost by automorphisms but with a back-reaction by the other just as for a matched pair of groups. Meanwhile, we see from (6.14) that $\cdot$ is associative only up to τ and τ itself obeys a kind of cocycle condition.

Clearly, R is a subgroup via $\cdot$ if and only if $\tau(r,s)=e$ for all r,s, and in this case we already see that $\Xi(R,K)$ is a bicrossproduct Hopf algebra, with the only difference being that we prefer to build it on the flipped tensor factors. More generally, [Beg03] showed that there is still a natural monoidal category associated to this data but with nontrivial associators. This corresponds by Tannaka-Krein reconstruction to a Ξ as quasi-bialgebra which in some cases is a quasi-Hopf algebra[Nat05]. Here we will give these latter structures explicitly and in maximum generality compared to the literature (but still needing a restriction on R for the antipode to be in a regular form). We will also show that the obvious $*$-algebra structure makes a $*$-quasi-Hopf algebra in an appropriate sense under restrictions on R. These aspects are new, but more importantly, we give direct proofs at an algebraic level rather than categorical arguments, which we believe are essential for detailed calculations. Related works on similar algebras and coset decompositions include [Sch02A, KM10B] in addition to [Beg03, Nat05, KM10A].

Lemma 6.4.1. [Beg03, Nat05, KM10A] *$(R,\cdot)$ has the same unique identity e as G and has the left division property, i.e. for all $t,s\in R$, there is a unique solution $r\in R$ to the equation $s\cdot r=t$ (one writes $r=s\backslash t$). In particular, we let r^R denote the unique solution to $r\cdot r^R=e$, which we call a right inverse.*

This means that $(R,\cdot)$ is a left loop (a left quasigroup with identity). The multiplication table for $(R,\cdot)$ has one of each element of R in each row, which is the left division property. In particular, there is one instance of e in each row. One can recover G knowing $(R,\cdot)$, K and the data $\triangleright,\triangleleft,\tau$[KM10A, Prop.3.4]. Note that a parallel property of left inverse $(\ )^L$ need not be present.

Definition 6.4.2. We say that R is *regular* if $(\ )^R$ is bijective.

R is regular iff it has both left and right inverses, and this is iff it satisfies $RK = KR$ by[KM10A, Prop. 3.5]. If there is also right division then we have a loop (a quasigroup with identity) and under further conditions[KM10A, Prop. 3.6] we have $r^L = r^R$ and a 2-sided inverse property quasigroup. The case of regular R is studied in [Nat05] but this excludes some interesting choices of R and we do not always assume it. Throughout, we will specify when R is required to be regular for results to hold. Finally, if R obeys a further condition $x\triangleright(s\cdot t) = (x\triangleright s)\triangleright t$ in [KM10A] then Ξ is a Hopf quasigroup in the sense introduced in [KM10B]. This is even more restrictive but will apply to our octonions-related example. Here we just give the choices for our go-to cases for S_3.

Example 6.4.3. $G = S_3$ with $K = \{e,u\}$ has four choices of transversal R meeting our requirement that $e \in R$. Namely

1. $R = \{e, uv, vu\}$ (our standard choice) *is a subgroup* $R = \mathbb{Z}_3$, so it is associative and there is 2-sided division and a 2-sided inverse. We also have $u\triangleright(uv) = vu$, $u\triangleright(vu) = uv$ but $\triangleleft,\tau$ trivial.

2. $R = \{e, w, v\}$ which is *not a subgroup* and indeed $\tau(v,w) = \tau(w,v) = u$ (and all others are necessarily e). There is an action $u\triangleright v = w$, $u\triangleright w = v$ but $\triangleleft$ is still trivial. For examples

$$vw = wu \Rightarrow v\cdot w = w,\ \tau(v,w) = u; \quad wv = vu \Rightarrow w\cdot v = v,\ \tau(w,v) = u$$
$$uv = wu \Rightarrow u\triangleright v = w,\ u\triangleleft v = u; \quad uw = vu \Rightarrow u\triangleright w = v,\ u\triangleleft w = u.$$

   This has left division/right inverses as it must but *not right division* as $e\cdot w = v\cdot w = w$ and $e\cdot v = w\cdot v = v$. We also have $v\cdot v = w\cdot w = e$ and $(\ )^R$ is bijective so this *is regular*.

3. $R = \{e, uv, v\}$ which is *not a subgroup* and $\tau,\triangleright,\triangleleft$ are all nontrivial with

$$\tau(uv,uv) = \tau(v,uv) = \tau(uv,v) = u, \quad \tau(v,v) = e,$$
$$v\cdot v = e, \quad v\cdot uv = uv, \quad uv\cdot v = e, \quad uv\cdot uv = v,$$
$$u\triangleright v = uv, \quad u\triangleright(uv) = v, \quad u\triangleleft v = e, \quad u\triangleleft uv = e$$

   and all other cases determined from the properties of e. Here $v^R = v$ and $(uv)^R = v$ so this is *not regular.*

4. $R = \{e, w, vu\}$ which is analogous to the preceding case, so *not a subgroup*, $\tau,\triangleright,\triangleleft$ all nontrivial and *not regular.*

We will also need the following useful lemma in some of our proofs.

Lemma 6.4.4. [KM10A] For any transversal R with $e \in R$, we have

1. $(x\triangleleft r)^{-1} = x^{-1}\triangleleft(x\triangleright r)$;

2. $(x \triangleright r)^R = (x \triangleleft r) \triangleright r^R$;
3. $\tau(r, r^R)^{-1} \triangleleft r = \tau(r^R, r^{RR})^{-1}$;
4. $\tau(r, r^R)^{-1} \triangleleft r = r^{RR}$;

for all $x \in K, r \in R$.

Proof. The first two items are elementary from the matched pair axioms. For (1), we use $e = (x^{-1}x) \triangleleft r = (x^{-1} \triangleleft (x \triangleright r))(x \triangleleft r)$ and for (2) $e = x \triangleright (r \cdot r^R) = (x \triangleright r) \cdot ((x \triangleleft r) \triangleright r^R)$. The other two items are a left-right reversal of [KM10A, Lem. 3.2] but given here for completeness. For (3),

$$\begin{aligned} e &= (\tau(r, r^R)\tau(r, r^R)^{-1}) \triangleleft r = (\tau(r, r^R) \triangleleft (\tau(r, r^R) \triangleright r))(\tau(r, r^R)^{-1} \triangleleft r) \\ &= (\tau(r, r^R) \triangleleft r^{RR})(\tau(r, r^R)^{-1} \triangleleft r) \end{aligned}$$

which we combine with

$$\tau(r^R, r^{RR}) = \tau(r \cdot r^R, r^{RR})\tau(r^R, r^{RR}) = \tau(r \cdot r^R, \tau(r, r^R) \triangleright r^{RR})(\tau(r, r^R) \triangleleft r^{RR}) = \tau(r, r^R) \triangleleft r^{RR}$$

by the cocycle property. For (4), $\tau(r, r^R) \triangleleft r^{RR} = (r \cdot r^R)\tau(r, r^R) \triangleleft r^{RR} = r \cdot (r^R \cdot r^{RR}) = r$ by one of the matched pair conditions. ■

Using this lemma, it is not hard to prove cf[KM10A, Prop.3.3] that

$$s \backslash t = s^R \cdot \tau^{-1}(s, s^R) \triangleright t; \quad s \cdot (s \backslash t) = s \backslash (s \cdot t) = t, \tag{6.15}$$

which can also be useful in calculations.

6.4.1 $\Xi(R, K)$ as a quasi-bialgebra

We recall that a quasi-bialgebra is a unital algebra H, a coproduct $\Delta : H \to H \otimes H$ which is an algebra map but is no longer required to be coassociative, and $\epsilon : H \to \mathbb{C}$ a counit for Δ in the usual sense $(\mathrm{id} \otimes \epsilon)\Delta = (\epsilon \otimes \mathrm{id})\Delta = \mathrm{id}$. Instead, we have a weaker form of coassociativity[Dri87, Maj95]

$$(\mathrm{id} \otimes \Delta)\Delta = \phi((\Delta \otimes \mathrm{id})\Delta(\ ))\phi^{-1}$$

for an invertible element $\phi \in H^{\otimes 3}$ obeying the 3-cocycle identity

$$(1 \otimes \phi)((\mathrm{id} \otimes \Delta \otimes \mathrm{id})\phi)(\phi \otimes 1) = ((\mathrm{id} \otimes \mathrm{id} \otimes \Delta)\phi)(\Delta \otimes \mathrm{id} \otimes \mathrm{id})\phi, \quad (\mathrm{id} \otimes \epsilon \otimes \mathrm{id})\phi = 1 \otimes 1$$

(it follows that ϵ in the other positions also gives $1 \otimes 1$). If V, W are representations then $V \otimes W$ is also, by $h.(v \otimes w) = (\Delta h).(v \otimes w)$ where one copy of H acts on $v \in V$ and the other on $w \in W$. In our case, we already know that $\Xi(R, K)$ is a unital algebra.

Lemma 6.4.5. $\Xi(R, K)$ is a quasi-bialgebra with

$$\Delta x = \sum_{s \in R} x\delta_s \otimes x \triangleleft s, \quad \Delta \delta_r = \sum_{s,t \in R} \delta_{s \cdot t, r} \delta_s \otimes \delta_t, \quad \epsilon x = 1, \quad \epsilon \delta_r = \delta_{r,e}$$

for all $x \in K, r \in R$, and

$$\phi = \sum_{r,s \in R} \delta_r \otimes \delta_s \otimes \tau(r, s)^{-1}, \quad \phi^{-1} = \sum_{r,s \in R} \delta_r \otimes \delta_s \otimes \tau(r, s).$$

Proof. This follows by reconstruction arguments, but it is useful to check directly,

$$\begin{aligned}(\Delta x)(\Delta y) &= \sum_{s,r}(x\delta_s\otimes x{\triangleleft}s)(y\delta_r\otimes y{\triangleleft}r) = \sum_{s,r}(x\delta_s y\delta_r\otimes x{\triangleleft}s)(y{\triangleleft}r)\\ &= \sum_{r,s} xy\delta_{y^{-1}{\triangleright}s}\delta_r\otimes(x{\triangleleft}s)(y{\triangleleft}r) = \sum_r xy\delta_r\otimes(x{\triangleleft}(y{\triangleright}r))(y{\triangleleft}r) = \Delta(xy)\end{aligned}$$

as $s = y{\triangleright}r$ and using the formula for $(xy){\triangleleft}r$ at the end. Also,

$$\begin{aligned}\Delta(\delta_{x\triangleright s}x) &= (\Delta\delta_{x\triangleright s})(\Delta x) = \sum_{r,p.t=x\triangleright s}\delta_p x\delta_r\otimes\delta_t x{\triangleleft}r\\ &= \sum_{r,p.t=x\triangleright s} x\delta_{x^{-1}\triangleright p}\delta_r\otimes x{\triangleleft}r\delta_{(x\triangleleft r)^{-1}\triangleright t} = \sum_{(x\triangleright r).t=x\triangleright s} x\delta_r\otimes x{\triangleleft}r\delta_{(x\triangleleft r)^{-1}\triangleright t}\\ &= \sum_{(x\triangleright r).((x\triangleleft r)\triangleright t')=x\triangleright s} x\delta_r\otimes x{\triangleleft}r\delta_{t'} = \sum_{r\cdot t'=s} x\delta_r\otimes(x{\triangleleft}r)\delta_{t'} = (\Delta x)(\Delta\delta_s) = \Delta(x\delta_s)\end{aligned}$$

using the formula for $x{\triangleright}(r\cdot t')$. This says that the coproducts stated are compatible with the algebra cross relations. Similarly, one can check that

$$\begin{aligned}(\sum_{p,r}\delta_p\otimes\delta_r\otimes\tau(p,r))((\mathrm{id}\otimes\Delta)\Delta x) &= \sum_{p,r,s,t}(\delta_p\otimes\delta_r\otimes\tau(p,r))(x\delta_s\otimes(x{\triangleleft}s)\delta_t\otimes(x{\triangleleft}s){\triangleleft}t)\\ &= \sum_{p,r,s,t}\delta_p x\delta_s\otimes\delta_r(x{\triangleleft}s)\delta_t\otimes\tau(p,r)((x{\triangleleft}s){\triangleleft}t)\\ &= \sum_{s,t}x\delta_s\otimes(x{\triangleleft}s)\delta_t\otimes\tau(x{\triangleright}s,(x{\triangleleft}s){\triangleright}t)(x{\triangleleft}s){\triangleleft}t)\\ &= \sum_{s,t}x\delta_s\otimes(x{\triangleleft}s)\delta_t\otimes(x{\triangleleft}(s.t))\tau(s,t)\\ &= \sum_{p,r,s,t}(x\delta_s\otimes(x{\triangleleft}s)\delta_t\otimes(x{\triangleleft}(s.t))(\delta_p\otimes\delta_r\otimes\tau(p,r)\\ &= ((\Delta\otimes\mathrm{id})\Delta x)(\sum_{p,r}\delta_p\otimes\delta_r\otimes\tau(p,r))\end{aligned}$$

as $p = x{\triangleright}s$ and $r = (x{\triangleleft}s){\triangleright}t$ and using the formula for $(x{\triangleleft}s){\triangleleft}t$. For the remaining cocycle relations, we have

$$(\mathrm{id}\otimes\epsilon\otimes\mathrm{id})\phi = \sum_{r,s}\delta_{s,e}\delta_r\otimes\tau(r,s)^{-1} = \sum_r\delta_r\otimes 1 = 1\otimes 1$$

and

$$(1\otimes\phi)((\mathrm{id}\otimes\Delta\otimes\mathrm{id})\phi)(\phi\otimes 1) = \sum_{r,s,t}\delta_r\otimes\delta_s\otimes\delta_t\tau(r,s)^{-1}\otimes\tau(s,t)^{-1}\tau(r,s\cdot t)$$

after multiplying out δ-functions and renaming variables. Using the value of $\Delta\tau(r,s)^{-1}$ and similarly multiplying out, we obtain on the other side

$$\begin{aligned}((\mathrm{id}\otimes\mathrm{id}\otimes\Delta)\phi)(\Delta\otimes\mathrm{id}\otimes\mathrm{id})\phi &= \sum_{r,s,t}\delta_r\otimes\delta_s\otimes\tau(r,s)^{-1}\delta_t\otimes(\tau(r,s)^{-1}{\triangleleft}t)\tau(r\cdot s,t)^{-1}\\ &= \sum_{r,s,t'}\delta_r\otimes\delta_s\otimes\delta_{t'}\tau(r,s)^{-1}\otimes(\tau(r,s)^{-1}{\triangleleft}(\tau(r,s){\triangleright}t'))\tau(r\cdot s,\tau(r,s){\triangleright}t')^{-1}\\ &= \sum_{r,s,t'}\delta_r\otimes\delta_s\otimes\delta_{t'}\tau(r,s)^{-1}\otimes(\tau(r,s){\triangleleft}t')^{-1}\tau(r\cdot s,\tau(r,s){\triangleright}t')^{-1},\end{aligned}$$

where we change summation to $t' = \tau(r,s)\triangleright t$ then use Lemma 6.4.4. Renaming t' to t, the two sides are equal in view of the cocycle identity for τ. Thus, we have a quasi-bialgebra with ϕ as stated. ■

This is of the same semidirect product (but dual) form as the coquasi-Hopf algebra noted in [KM10A, Rem 4.3] and also known in [Nat05], however. A coproduct is also stated in [CCW16] but in very different notations and without proof. Physically, recall that we consider the representations of $\Xi(R,K)$ to be quasiparticles located at the boundary of the Kitaev model, with the restriction that we cannot have multiplicities of irreps greater than 1 as per Proposition 6.2.19. As $\Xi(R,K)$ is a quasi-bialgebra, ${}_\Xi\mathcal{M}$ is monoidal, albeit with a nontrivial associator, and hence we can have quasiparticles existing in separate locations; considering time as well, this means parallel worldlines of boundary quasiparticles are allowed by the theory, as one would expect. The nontrivial associator is an interesting feature, and one which the bulk ${}_{D(G)}\mathcal{M}$ model does not exhibit. We discuss this in further detail in Section 6.5.

Remark 6.4.6. If we want to write the coproduct on Ξ explicitly as a vector space, the above becomes

$$\Delta(\delta_r\otimes x) = \sum_{s\cdot t=r}\delta_s\otimes x\otimes\delta_t\otimes(x^{-1}\triangleleft s)^{-1},\quad \epsilon(\delta_r\otimes x)=\delta_{r,e}$$

which is ugly due to our decision to build it on $\mathbb{C}(R)\otimes\mathbb{C}K$. (2) If we built it on the other order then we could have $\Xi = \mathbb{C}K\triangleright\!\!\!<\mathbb{C}(R)$ as an algebra, where we have a right action

$$(f\triangleleft x)(r) = f(x\triangleright r);\quad \delta_r\triangleleft x = \delta_{x^{-1}\triangleright r}$$

on $f\in\mathbb{C}(R)$. Now make a right handed cross product

$$(x\otimes\delta_r)(y\otimes\delta_s) = xy\otimes(\delta_r\triangleleft y)\delta_s = xy\otimes\delta_s\delta_{r,y\triangleright s}$$

which has cross relations $\delta_r y = y\delta_{y^{-1}\triangleright r}$. These are the same relations as before. So this is the same algebra, just we prioritise a basis $\{x\delta_r\}$ instead of the other way around. This time, we have

$$\Delta(x\otimes\delta_r) = \sum_{s\cdot t=r} x\otimes\delta_s\otimes x\triangleleft s\otimes\delta_t.$$

We do not do this in order to be compatible with the most common form of $D(G)$ as $\mathbb{C}(G)\rtimes\mathbb{C}G$ as in [CM22].

Lemma 6.4.7. The $*$-algebra structure on $\Xi(R,K)$ commutes with the coproduct and counit, $\Delta\circ * = (*\otimes *)\circ\Delta$ and $\epsilon\circ * = \overline{\epsilon(\ )}$. Moreover $\phi^{*\otimes *\otimes *} = \phi^{-1}$.

Proof. Since $*$ is an involution it is enough to check the assertion on $x\in K$ and $\delta_r\in\mathbb{C}(R)$ separately. The latter is immediate from the form of $\Delta\delta_r$ in Lemma 6.4.5 and $\delta_r^* = \delta_r$. For the other case, we have

$$(*\otimes *)\Delta x = \sum_s \delta_s x^{-1}\otimes(x\triangleleft s)^{-1} = \sum_s x^{-1}\delta_{x\triangleright s}\otimes x^{-1}\triangleleft(x\triangleright s) = \sum_{s'} x^{-1}\delta_{s'}\otimes x^{-1}\triangleleft s' = \Delta x^{-1}$$

which is Δx^*. The remaining properties are more immediate. ■

Note that this is not a usual property of $*$-quasi-balgebras as such a condition in general would not be compatible with the quasi-coassociativity controlled by ϕ. But it is true for a usual Hopf $*$-algebra where it is important for preserving unitarity (in a Hopf sense defined by $*$) of tensor products of representations, hence it is convenient that it also applies to $\Xi(R,K)$. The latter is also a $*$-quasibialgebra but this is deferred to Appendix 24.

6.4.2 $\Xi(R,K)$ as a quasi-Hopf algebra

A quasi-bialgebra is a quasi-Hopf algebra if there are elements $\alpha,\beta\in H$ and an antialgebra map $S:H\to H$ such that[Dri87, Maj95]

$$(S\xi_1)\alpha\xi_2=\epsilon(\xi)\alpha,\quad \xi_1\beta S\xi_2=\epsilon(\xi)\beta,\quad \phi^1\beta(S\phi^2)\alpha\phi^3=1,\quad (S\phi^{-1})\alpha\phi^{-2}\beta S\phi^{-3}=1$$

where $\Delta\xi=\xi_1\otimes\xi_2$, $\phi=\phi^1\otimes\phi^2\otimes\phi^3$ with inverse $\phi^{-1}\otimes\phi^{-2}\otimes\phi^{-3}$ is a compact notation (sums of such terms to be understood). It is usual to assume S is bijective but we do not require this. The α,β,S are not unique and can be changed to $S'=U(S\ )U^{-1},\alpha'=U\alpha,\beta'=\beta U^{-1}$ for any invertible U. In particular, if α is invertible then we can transform to a standard form replacing it by 1. For the purposes of this Chapter, we therefore call the case of α invertible a (left) *regular antipode*. The antipode provides a kind of linearised analogue of group inversion and is needed for example in the quantum adjoint action and in the dualisation of representations. If H acts on V then it acts on V^* by $(h.f)(v)=f(Sh.v)$ for $v\in V$ and $f\in V^*$.

Theorem 6.4.8. If $(\ )^R$ is bijective, $\Xi(R,K)$ is a quasi-Hopf algebra with regular antipode

$$S(\delta_r\otimes x)=\delta_{(x^{-1}\triangleright r)^R}\otimes x^{-1}\triangleleft r,\quad \alpha=\sum_{r\in R}\delta_r\otimes 1,\quad \beta=\sum_r\delta_r\otimes\tau(r,r^R).$$

Equivalently in subalgebra terms,

$$S\delta_r=\delta_{r^R},\quad Sx=\sum_{s\in R}(x^{-1}\triangleleft s)\delta_{s^R},\quad \alpha=1,\quad \beta=\sum_{r\in R}\delta_r\tau(r,r^R).$$

Proof. For the axioms involving ϕ, we have

$$\begin{aligned}\phi^1\beta(S\phi^2)\alpha\phi^3&=\sum_{s,t,r}(\delta_s\otimes 1)(\delta_r\otimes\tau(r,r^R))(\delta_{t^R}\otimes\tau(s,t)^{-1})\\&=\sum_{s,t}(\delta_s\otimes\tau(s,s^R))(\delta_{t^R}\otimes\tau(s,t)^{-1})=\sum_{s,t}\delta_s\delta_{s,\tau(s,s^R)\triangleright t^R}\otimes\tau(s,s^R)\tau(s,t)^{-1}\\&=\sum_{s^R.t^R=e}\delta_s\otimes\tau(s,s^R)\tau(s,t)^{-1}=1,\end{aligned}$$

where we used $s\cdot(s^R\cdot t^R)=(s\cdot s^R)\cdot\tau(s,s^R)\triangleright t^R=\tau(s,s^R)\triangleright t^R$. So $s=\tau(s,s^R)\triangleright t^R$ holds iff $s^R\cdot t^R=e$ by left cancellation. In the sum, we can take $t=s^R$ which contributes $\delta_s\otimes e$. Here $s^R\cdot t^R=s^R\cdot(s^R)^R=e$; there is a unique element t^R which does this and hence a unique t provided $(\ )^R$ is injective, and hence a bijection.

$$\begin{aligned}S(\phi^{-1})\alpha\phi^{-2}\beta S(\phi^{-3})&=\sum_{s,t,u,v}(\delta_{s^R}\otimes 1)(\delta_t\otimes 1)(\delta_u\otimes\tau(u,u^R))(\delta_{(\tau(s,t)^{-1}\triangleright v)^R}\otimes(\tau(s,t)^{-1}\triangleleft v))\\&=\sum_{s,v}(\delta_{s^R}\otimes\tau(s^R,s^{RR}))(\delta_{(\tau(s,s^R)^{-1}\triangleright v)^R}\otimes\tau(s,s^R)^{-1}\triangleleft v).\end{aligned}$$

Upon multiplication, we will have a δ-function dictating that

$$s^R=\tau(s^R,s^{RR})\triangleright(\tau(s,s^R)^{-1}\triangleright v)^R,$$

so we can use the fact that

$$\begin{aligned} s\cdot s^R = e &= s\cdot(\tau(s^R,s^{RR}){\triangleright}(\tau(s,s^R)^{-1}{\triangleright}v)^R)\\ &= s\cdot(s^R\cdot(s^{RR}\cdot(\tau(s,s^R)^{-1}{\triangleright}v)^R))\\ &= \tau(s,s^R){\triangleright}(s^{RR}\cdot(\tau(s,s^R){\triangleright}v)^R), \end{aligned}$$

where we use similar identities to before. Therefore $s^{RR}{\cdot}(\tau(s,s^R)^{-1}{\triangleright}v)^R = e$, so $(\tau(s,s^R)^{-1}{\triangleright}v)^R = s^{RRR}$. When $(\ )^R$ is injective, this gives us $v=\tau(s,s^R){\triangleright}s^{RR}$. Returning to our original calculation we have that our previous expression is

$$\begin{aligned} \cdots &= \sum_s \delta_{s^R}\otimes\tau(s^R,s^{RR})(\tau(s,s^R)^{-1}{\triangleleft}(\tau(s,s^R){\triangleright}s^{RR}))\\ &= \sum_s \delta_{s^R}\otimes\tau(s^R,s^{RR})(\tau(s,s^R){\triangleleft}s^{RR})^{-1} = \sum_s \delta_{s^R}\otimes 1 = 1. \end{aligned}$$

We now prove the antipode axiom involving α,

$$\begin{aligned} (S(\delta_s{\otimes}x)_1)(\delta_s\otimes x)_2 &= \sum_{r\cdot t=s}(\delta_{(x^{-1}{\triangleright}r)^R}\otimes(x^{-1}{\triangleleft}r))(\delta_t\otimes(x^{-1}{\triangleleft}r)^{-1})\\ = \sum_{r\cdot t=s}\delta_{(x^{-1}{\triangleright}r)^R,(x^{-1}{\triangleleft}r){\triangleright}t}\delta_{(x^{-1}{\triangleright}r)^R}\otimes 1 &= \delta_{e,s}\sum_r \delta_{(x^{-1}{\triangleright}r)^R}\otimes 1 = \epsilon(\delta_s\otimes x)1. \end{aligned}$$

The condition from the δ-functions is

$$(x^{-1}{\triangleright}r)^R = (x^{-1}{\triangleleft}r){\triangleright}t$$

which by uniqueness of right inverses holds iff

$$e = (x^{-1}{\triangleright}r)\cdot(x^{-1}{\triangleleft}r){\triangleright}t = x^{-1}{\triangleright}(r\cdot t)$$

which is iff $r\cdot t=e$, so $t=r^R$. As we also need $r\cdot t=s$, this becomes $\delta_{s,e}$ as required.

We now prove the axiom involving β, starting with

$$\begin{aligned} (\delta_s{\otimes}x)_1\beta S((\delta_s\otimes x)_2) &= \sum_{r\cdot t=s,p}(\delta_r\otimes x)(\delta_p\otimes\tau(p,p^R))S(\delta_t\otimes(x^{-1}{\triangleleft}r)^{-1})\\ = \sum_{r\cdot t=s,p}&(\delta_r\delta_{r,x{\triangleright}p}\otimes x\tau(p,p^R))(\delta_{((x^{-1}{\triangleleft}r){\triangleright}t)^R}\otimes(x^{-1}{\triangleleft}r){\triangleleft}t)\\ = \sum_{r\cdot t=s}&(\delta_r\otimes x\tau(x^{-1}{\triangleright}r,(x^{-1}{\triangleright}r)^R))(\delta_{((x^{-1}{\triangleleft}r){\triangleright}t)^R}\otimes(x^{-1}{\triangleleft}r){\triangleleft}t). \end{aligned}$$

When we multiply this out, we will need from the product of δ-functions that

$$\tau(x^{-1}{\triangleright}r,(x^{-1}{\triangleright}r)^R)^{-1}{\triangleright}(x^{-1}{\triangleright}r) = ((x^{-1}{\triangleleft}r){\triangleright}t)^R,$$

but note that $\tau(q,q^R)^{-1}{\triangleright}q = q^{RR}$ from Lemma 6.4.4. So the condition from the δ-functions is

$$(x^{-1}{\triangleright}r)^{RR} = ((x^{-1}{\triangleleft}r){\triangleright}t)^R,$$

so

$$(x^{-1}{\triangleright}r)^R = (x^{-1}{\triangleleft}r){\triangleright}t$$

when $(\ )^R$ is injective. By uniqueness of right inverses, this holds iff

$$e=(x^{-1}{\triangleright}r)\cdot((x^{-1}{\triangleleft}r){\triangleright}t)=x^{-1}{\triangleright}(r\cdot t),$$

where the last equality is from the matched pair conditions. This holds iff $r\cdot t=e$, that is, $t=r^R$. This also means in the sum that we need $s=e$. Hence, when we multiply out our expression so far, we have

$$\cdots=\delta_{s,e}\sum_r\delta_r\otimes x\tau(x^{-1}{\triangleright}r,(x^{-1}{\triangleright}r)^R)(x^{-1}{\triangleleft}r){\triangleleft}r^R=\delta_{s,e}\sum_r\delta_r\otimes\tau(r,r^R)=\delta_{s,e}\beta,$$

as required, where we used

$$x\tau(x^{-1}{\triangleright}r,(x^{-1}{\triangleright}r)^R)(x^{-1}{\triangleleft}r){\triangleleft}r^R=\tau(r,r^R)$$

by the matched pair conditions. The subalgebra form of Sx is the same using the commutation relations and Lemma 6.4.4 to reorder.

It remains to check that

$$\begin{aligned}S(\delta_s\otimes y)S(\delta_r\otimes x)&=(\delta_{(y^{-1}{\triangleright}s)^R}\otimes y^{-1}{\triangleleft}s)(\delta_{(x^{-1}{\triangleright}x)^R}\otimes x^{-1}{\triangleleft}r)\\&=\delta_{r,x{\triangleright}s}\delta_{(y^{-1}{\triangleright}s)^R}\otimes(y^{-1}{\triangleleft}s)(x^{-1}{\triangleleft}r)=\delta_{r,x{\triangleright}s}\delta_{(y^{-1}x^{-1}{\triangleright}r)^R}\otimes(y^{-1}{\triangleleft}(x^{-1}{\triangleright}r))(x^{-1}{\triangleleft}r)\\&=S(\delta_r\delta_{r,x{\triangleright}s}\otimes xy)=S((\delta_r\otimes x)(\delta_s\otimes y)),\end{aligned}$$

where the product of δ-functions requires $(y^{-1}{\triangleright}s)^R=(y^{-1}{\triangleleft}s){\triangleright}(x^{-1}{\triangleright}r)^R$, which is equivalent to $s^R=(x^{-1}{\triangleright}r)^R$ using Lemma 6.4.4. This imposes $\delta_{r,x{\triangleright}s}$. We then replace $s=x^{-1}{\triangleright}r$ and recognise the answer using the matched pair identities. ■

The antipode here has the identical form (after allowing for changes in conventions) to the related Hopf quasi-algebra antipode in [KM10A]. An antipode is also stated in [CCW16] but in very different notations and without proof. Moreover, we have the following novel results about this standard S on $\Xi(R,K)$.

Proposition 6.4.9. There exist invertible $\gamma\in\Xi(R,K)$ and $\mathcal{G}\in\Xi(R,K)^{\otimes 2}$ such that the standard S in Theorem 6.4.8 obeys

$$\Delta\circ S=\mathcal{G}^{-1}((S\otimes S)\circ\Delta^{op}(\ ))\mathcal{G},\quad \epsilon\circ S=\epsilon,\quad (*\circ S)^2=\gamma(\ )\gamma^{-1}$$

along with

$$S\gamma=\gamma^{-1},\quad ((S\otimes S)\mathcal{G}_{21}^{-1})\mathcal{G}=(\gamma\otimes\gamma)\Delta\gamma^{-1}$$

and the 'quasi-cocycle' condition

$$(S^{\otimes 3}\phi_{321}^{-1})(1\otimes\mathcal{G})((\mathrm{id}\otimes\Delta)\mathcal{G})\phi=(\mathcal{G}\otimes 1)(\Delta\otimes\mathrm{id})\mathcal{G}.$$

Moreover, $\gamma^*=\gamma^{-1},\quad \mathcal{G}^{*\otimes *}=\mathcal{G}^{-1}$. Here $\mathcal{G}_{21}$ and ϕ_{321} are $\mathcal{G},\phi$ with the tensor factors taken in reverse order.

Proof. The proof is given in Appendix 24 where it follows from the $*$-quasi-Hopf structure proven there given Lemma 6.4.7 proven above. Without ϕ, the 'quasi-cocycle' condition would be the standard notion of a Drinfeld-twist 2-cocycle as in [Maj95, Chapter 2]. ■

A usual Hopf algebra and Hopf $*$-algebra would obey these conditions with $\mathcal{G}=1\otimes 1$ and $\gamma=1$ but we see how these familiar and key properties of the antipode still hold for the standard antipode of $\Xi(R,K)$, up to a certain conjugation.

Example 6.4.10. (i) $\Xi(R,K)$ for $S_2 \subset S_3$ with its standard transversal. As an algebra, this is generated by $\mathbb{Z}_2$, which means by an element u with $u^2 = e$, and by $\delta_0, \delta_1, \delta_2$ for δ-functions as the points of $R = \{e, uv, vu\}$. The relations are δ_i orthogonal and add to 1, and cross relations

$$\delta_0 u = u\delta_0, \quad \delta_1 u = u\delta_2, \quad \delta_2 u = u\delta_1.$$

The dot product is the additive group $\mathbb{Z}_3$, i.e. addition mod 3. The coproducts etc are

$$\Delta\delta_i = \sum_{j+k=i} \delta_j \otimes \delta_k, \quad \Delta u = u \otimes u, \quad \phi = 1 \otimes 1 \otimes 1$$

with addition mod 3. The cocycle and right action are trivial and the dot product is that of $\mathbb{Z}_3$ as a subgroup generated by uv. This gives an ordinary cross product Hopf algebra $\Xi = \mathbb{C}(\mathbb{Z}_3)\rtimes\!\!\!\lhd \mathbb{C}\mathbb{Z}_2$. Here $S\delta_i = \delta_{-i}$ and $Su = u$. The cocycle is trivial so $\gamma = 1$ and $\mathcal{G} = 1 \otimes 1$ in Proposition 6.4.9 and we have an ordinary Hopf $*$-algebra.

(ii) $\Xi(R,K)$ for $S_2 \subset S_3$ with its second transversal. For this R, the dot product is specified by e the identity and $v \cdot w = w$, $w \cdot v = v$. The algebra has relations

$$\delta_e u = u\delta_e, \quad \delta_v u = u\delta_w, \quad \delta_w u = u\delta_v$$

and the quasi-Hopf algebra coproducts etc. are

$$\Delta\delta_e = \delta_e \otimes \delta_e + \delta_v \otimes \delta_v + \delta_w \otimes \delta_w, \quad \Delta\delta_v = \delta_e \otimes \delta_v + \delta_v \otimes \delta_e + \delta_w \otimes \delta_v,$$

$$\Delta\delta_w = \delta_e \otimes \delta_w + \delta_w \otimes \delta_e + \delta_v \otimes \delta_w, \quad \Delta u = u \otimes u,$$
$$\phi = 1 \otimes 1 \otimes 1 + (\delta_v \otimes \delta_w + \delta_w \otimes \delta_v) \otimes (u - 1) = \phi^{-1}.$$

The antipode is

$$S\delta_s = \delta_{s^R} = \delta_s, \quad Su = \sum_s \delta_{(u \triangleright s)^R} u = u, \quad \alpha = 1, \quad \beta = \sum_s \delta_s \otimes \tau(s,s) = 1$$

from the antipode lemma, since the map $(\ )^R$ happens to be injective and indeed acts as the identity. In this case, we see that $\Xi(R,K)$ is nontrivially a quasi-Hopf algebra. Only $\tau(v,w) = \tau(w,v) = u$ are nontrivial, hence we have

$$\gamma = 1, \quad \mathcal{G} = 1 \otimes 1 + (\delta_v \otimes \delta_w + \delta_w \otimes \delta_v)(u \otimes u - 1 \otimes 1).$$

in Proposition 6.4.9. Moreover, $*S$ acts as the identity on our basis (but is antilinear).

We also note that the algebras $\Xi(R,K)$ here are manifestly isomorphic for the two R, but the coproducts are different, so the tensor products of representations is different, although they turn out isomorphic. The set of irreps does not change either, but how we construct them can look different. We will see in the next that this is part of a monoidal equivalence of categories.

Example 6.4.11. $S_2 \subset S_3$ with its 2nd transversal. Here R has two orbits: (a) $\mathcal{C} = \{e\}$ with $r_0 = e, K^{r_0} = K$ with two 1-diml irreps V_ρ as ρ=trivial and ρ = sign, and hence two irreps of $\Xi(R,K)$; (b) $\mathcal{C} = \{w, v\}$ with $r_0 = v$ or $r_0 = w$, both with $K^{r_0} = \{e\}$ and hence only ρ trivial, leading to one 2-dimensional irrep of $\Xi(R,K)$. So, altogether, there are again three irreps of $\Xi(R,K)$:

$$V_{(\{e\},\rho)}: \quad \delta_r.1 = \delta_{r,e}, \quad u.1 = \pm 1,$$
$$V_{(\{w,v\}),1)}: \quad \delta_r.v = \delta_{r,v} v, \quad \delta_r.w = \delta_{r,w} w, \quad u.v = w, \quad u.w = v$$

acting on $\mathbb{C}$ and on the span of v, w respectively. These irreps are equivalent to what we had in Example 6.2.9 when computing irreps from the standard R.

6.5 Categorical justification and twisting theorem

We have shown that the boundaries can be defined using the action of the algebra $\Xi(R,K)$ and that one can perform novel methods of fault-tolerant quantum computation using these boundaries. The full story, however, involves the quasi-Hopf algebra structure verified in the preceding section and now we would like to connect back up to the category theory behind this.

6.5.1 G-graded K-bimodules.

We start by proving the equivalence ${}_{\Xi(R,K)}\mathcal{M} \simeq {}_K\mathcal{M}_K^G$ explicitly and use it to derive the coproduct studied in Section 6.4. Although this equivalence is known[Sch02A], we believe this to be a new and more direct derivation.

Lemma 6.5.1. If V_ρ is a K^{r_0}-module and $V_{\mathcal{O},\rho}$ the associated $\Xi(R,K)$ irrep, then

$$\tilde{V}_{\mathcal{O},\rho} = V_{\mathcal{O},\rho} \otimes \mathbb{C}K, \quad x.(r\otimes v\otimes z).y = x\triangleright r\otimes \zeta_r(x).v\otimes (x\triangleleft r)zy, \quad |r\otimes v\otimes z| = rz$$

is a G-graded K-bimodule. Here $r\in\mathcal{O}$ and $v\in V_\rho$ in the construction of $V_{\mathcal{O},\rho}$.

Proof. That this is a G-graded right K-module commuting with the left action of K is trivial. That the left action works and is G-graded is

$$\begin{aligned} x.(y.(r\otimes v\otimes z)) &= x.(y\triangleright r\otimes \zeta_r(y).v\otimes (y\triangleleft r)z) = xy\triangleright r\otimes \zeta_r(xy).v\otimes (x\triangleleft(y\triangleright r))(y\triangleleft r)z \\ &= xy\triangleright r\otimes \zeta_r(xy).v\otimes ((xy)\triangleleft r)z \end{aligned}$$

and

$$|x.(r\otimes v\otimes z).y| = (x\triangleright r)(x\triangleleft r)zy = xrzy = x|r\otimes v\otimes z|y.$$

■

Remark 6.5.2. Recall that we can also think more abstractly of $\Xi = \mathbb{C}(G/K)\rtimes\mathbb{C}K$ rather than using a transversal. In these terms, a representation of $\Xi(R,K)$, which is an R-graded K-module V such that $|x.v| = x\triangleleft|v|$, now becomes a G/K-graded K-module. This has that $|x.v| = x|v|$, where $|v|\in G/K$, and we multiply from the left by $x\in K$. Moreover, the role of an orbit $\mathcal{O}$ above is played by a double coset $T = \mathcal{O}K \in {}_KG_K$. In these terms, the role of the isometry group K^{r_0} is played by

$$K^{r_T} := K\cap r_TKr_T^{-1},$$

where r_T is any representative of the same double coset. One can take $r_T = r_0$ but we can also chose it more freely. Then an irrep is given by a double coset T and an irreducible representation ρ_T of K^{r_T}. If we denote by V_{ρ_T} the carrier space for this then the associated irrep of $\mathbb{C}(G/K)\rtimes\mathbb{C}K$ is $V_{T,\rho_T} = \mathbb{C}K\otimes_{K^{r_T}} V_{\rho_T}$ which is manifestly a K-module and we give it the G/K-grading by $|x\otimes_{K^{r_T}} v| = xK$. The construction in the last lemma is then equivalent to

$$\tilde{V}_{T,\rho_T} = \mathbb{C}K \underset{K^{r_T}}{\otimes} V_{\rho_T}\otimes\mathbb{C}K, \quad |x\underset{K^{r_T}}{\otimes} v\otimes z| = xz$$

as manifestly a G-graded K-bimodule. This is an equivalent point of view, but we prefer our more explicit one based on R, hence details are omitted.

Also note that the category ${}_K\mathcal{M}_K^G$ of G-graded K-bimodules has an obvious monoidal structure inherited from that of K-bimodules, where we tensor product over $\mathbb{C}K$. Here $|w\otimes_{\mathbb{C}K}w'|=|w||w'|$ in G is well-defined and $x.(w\otimes_{\mathbb{C}K}w').y=x.w\otimes_{\mathbb{C}K}w'.y$ has degree $x|w||w'|y=x|w\otimes_{\mathbb{C}K}w'|y$ as required.

Proposition 6.5.3. We let R be a transversal and $W=V\otimes\mathbb{C}K$ made into a G-graded K-bimodule by

$$x.(v\otimes z).y=x.v\otimes(x{\triangleleft}|v|)zy,\quad |v\otimes z|=|v|z\in G,$$

where now we view $|v|\in R$ as the chosen representative of $|v|\in G/K$. This gives a functor $F:{}_{\Xi}\mathcal{M}\to{}_K\mathcal{M}_K^G$ which is a monoidal equivalence for a suitable quasibialgebra structure on $\Xi(R,K)$. The latter depends on R since F depends on R.

Proof. We define $F(V)$ as stated, which is clearly a right module that commutes with the left action, and the latter is a module structure as

$$x.(y.(v\otimes z))=x.(y.v\otimes(y{\triangleleft}|v|)z)=xy.v\otimes(x{\triangleleft}(y{\triangleright}|v|))(y{\triangleleft}|v|)z=(xy).(v\otimes z)$$

using the matched pair axiom for $(xy){\triangleleft}|v|$. We also check that $|x.(v\otimes z).y|=|x.v|zy=(x{\triangleright}|v|)(x{\triangleleft}|v|)zy=x|v|zy=x|v\otimes z|y$. Hence, we have a G-graded K-bimodule. Conversely, if W is a G-graded K-bimodule, we let

$$V=\{w\in W\mid |w|\in R\},\quad x.v=xv(x{\triangleleft}|v|)^{-1},\quad \delta_r.v=\delta_{r,|v|}v,$$

where v on the right is viewed in W and we use the K-bimodule structure. This is arranged so that $x.v$ on the left lives in V. Indeed, $|x.v|=x|v|(x{\triangleleft}|v|)^{-1}=x{\triangleright}|v|$ and $x.(y.v)=xyv(y{\triangleleft}|v|)^{-1}(x{\triangleleft}(y{\triangleright}|v|))^{-1}=xyv((xy){\triangleleft}|v|)^{-1}$ by the matched pair condition, as required for a representation of $\Xi(R,K)$. One can check that this is inverse to the other direction. Thus, given $W=\oplus_{rx\in G}W_{rx}=\oplus_{x\in K}W_{Rx}$, where we let $W_{Rx}=\oplus_{r\in R}W_{rx}$, the right action by $x\in K$ gives an isomorphism $W_{Rx}{\cong}V\otimes x$ as vector spaces and hence recovers $W=V\otimes\mathbb{C}K$. This clearly has the correct right K-action and from the left $x.(v\otimes z)=xv(x{\triangleleft}|v|)^{-1}\otimes(x{\triangleleft}|v|)z$, which under the identification maps to $xv(x{\triangleleft}|v|)^{-1}(x{\triangleleft}|v|)z=xvz\in W$ as required given that $v\otimes z$ maps to vz in W.

Now, if V,V' are $\Xi(R,K)$ modules then as vector spaces,

$$F(V)\underset{\mathbb{C}K}{\otimes}F(V')=(V\otimes\mathbb{C}K)\underset{\mathbb{C}K}{\otimes}(V'\otimes\mathbb{C}K)=V\otimes V'\otimes\mathbb{C}K\overset{f_{V,V'}}{\cong}F(V\otimes V')$$

by the obvious identifications except that in the last step we allow ourselves the possibility of a nontrivial isomorphism as vector spaces. For the actions on the two sides,

$$x.(v\otimes v'\otimes z).y=x.(v\otimes v')\otimes(x{\triangleleft}|v\otimes v'|)zy=x.v\otimes(x{\triangleleft}|v|).v'\otimes((x{\triangleleft}|v|){\triangleleft}|v'|)zy,$$

where on the right, we have $x.(v\otimes 1)=x.v\otimes x{\triangleleft}|v|$ and then take $x{\triangleleft}|v|$ via the $\otimes_{\mathbb{C}K}$ to act on $v'\otimes z$ as per our identification. Comparing the x action on the $V\otimes V'$ factor, we need

$$\Delta x=\sum_{r\in R}x\delta_r\otimes x{\triangleleft}r=\sum_{r\in R}\delta_{x{\triangleright}r}\otimes x\otimes 1\otimes x{\triangleleft}r$$

as a modified coproduct without requiring a nontrivial $f_{V,V'}$ for this to work. The first expression is viewed in $\Xi(R,K)^{\otimes 2}$ and the second is on the underlying vector space. Likewise, looking at the grading of $F(V\otimes V')$ and comparing with the grading of $F(V)\otimes_{\mathbb{C}K}F(V')$, we need to define $|v\otimes v'|=|v|\cdot|v'|\in R$ and use $|v|\cdot|v'|\tau(|v|,|v'|)=|v||v'|$ to match the degree on the left hand side. This amounts to the coproduct of δ_r in $\Xi(R,K)$,

$$\Delta\delta_r=\sum_{s\cdot t=r}\delta_s\otimes\delta_t=\sum_{s\cdot t=r}\delta_s\otimes 1\otimes\delta_t\otimes 1$$

and a further isomorphism

$$f_{V,V'}(v\otimes v'\otimes z)=v\otimes v'\otimes\tau(|v|,|v'|)z$$

on the underlying vector space. After applying this, the degree of this element is $|v\otimes v'|\tau(|v|,|v'|)z=|v||v'|z=|v\otimes 1||v'\otimes z|$, which is the degree on the original $F(V)\otimes_{\mathbb{C}K}F(V')$ side. Now we show that $f_{V,V'}$ respects associators on each side of F. Taking the associator on the $\Xi(R,K)$-module side as

$$\phi_{V,V',V''}:(V\otimes V')\otimes V''\to V\otimes(V'\otimes V''),\quad \phi_{V,V',V''}((v\otimes v')\otimes v'')=\phi^1.v\otimes(\phi^2.v'\otimes\phi^3.v'')$$

and ϕ trivial on the G-graded K-bimodule side, for F to be monoidal with the stated $f_{V,V'}$ etc, we need equality of

$$\begin{aligned}
&F(\phi_{V,V',V''})f_{V\otimes V',V''}f_{V,V'}(v\otimes v'\otimes z)\\
&\qquad=F(\phi_{V,V',V''})f_{V\otimes V',V''}(v\otimes v'\otimes\tau(|v|,|v'|).v''\otimes(\tau(|v|,|v'|)\triangleleft|v''|)z)\\
&\qquad=F(\phi_{V,V',V''})(v\otimes v'\otimes\tau(|v|,|v'|).v''\otimes\tau(|v|\cdot|v'|,\tau(|v|,|v'|)\triangleright|v''|)(\tau(|v|,|v'|)\triangleleft|v''|)z)\\
&\qquad=F(\phi_{V,V',V''})(v\otimes v'\otimes\tau(|v|,|v'|).v''\otimes\tau(|v|,|v'|\cdot|v''|)\tau(|v'|,|v''|)z,\\
&\quad f_{V,V'\otimes V''}f_{V',V''}(v\otimes v'\otimes v''\otimes z)=f_{V,V'\otimes V''}(v\otimes v'\otimes v''\otimes\tau(|v'|,|v''|)z)\\
&\qquad=v\otimes v'\otimes v''\otimes\tau(|v|,|v'\otimes v''|)\tau(|v'|,|v''|)z=v\otimes v'\otimes v''\otimes\tau(|v|,|v'|\cdot|v''|)\tau(|v'|,|v''|)z,
\end{aligned}$$

where for the first equality we moved $\tau(|v|,|v'|)$ in the output of $f_{V,V'}$ via $\otimes_{\mathbb{C}K}$ to act on the v''. We used the cocycle property of τ for the 3rd equality. Comparing results, we need

$$\phi_{V,V',V''}((v\otimes v')\otimes v'')=v\otimes(v'\otimes\tau(|v|,|v'|)^{-1}.v''),\quad \phi=\sum_{s,t\in R}(\delta_s\otimes 1)\otimes(\delta_s\otimes 1)\otimes(1\otimes\tau(s,t)^{-1}).$$

Note that we can write

$$f_{V,V'}(v\otimes v'\otimes z)=(\sum_{s,t\in R}(\delta_s\otimes 1)\otimes(\delta_t\otimes 1)\otimes\tau(s,t)).(v\otimes v'\otimes z)$$

but we are not saying that ϕ is a coboundary since this is not given by the action of an element of $\Xi(R,K)^{\otimes 2}$. ■

This derives the quasibialgebra structure on $\Xi(R,K)$ used in Section 6.4 but now so as to obtain an equivalence of categories.

6.5.2 Drinfeld twists induced by change of transversal

We recall that if H is a quasiHopf algebra and $\chi\in H\otimes H$ is a *cochain* in the sense of being invertible and $(\mathrm{id}\otimes\epsilon)\chi=(\epsilon\otimes\mathrm{id})\chi=1$, then its *Drinfeld twist* $\bar H$ is another quasi-Hopf algebra

$$\bar\Delta=\chi^{-1}\Delta(\ )\chi,\quad \bar\phi=\chi_{23}^{-1}((\mathrm{id}\otimes\Delta)\chi^{-1})\phi((\Delta\otimes\mathrm{id})\chi)\chi_{12},\quad \bar\epsilon=\epsilon$$
$$S=S,\quad \bar\alpha=(S\chi^1)\alpha\chi^2,\quad \bar\beta=(\chi^{-1})^1\beta S(\chi^{-1})^2$$

where $\chi = \chi^1 \otimes \chi^2$ is with a sum of such terms understood and we use same notation for χ^{-1}, see [Maj95, Thm. 2.4.2] but note that our χ is denoted F^{-1} there. In categorical terms, this twist corresponds to a monoidal equivalence $G : {}_H\mathcal{M} \to {}_{\bar{H}}\mathcal{M}$ which is the identity on objects and morphisms but has a nontrivial natural transformation

$$g_{V,V'} : G(V)\bar{\otimes}G(V')\cong G(V\otimes V'), \quad g_{V,V'}(v\otimes v') = \chi^1.v\otimes\chi^2.v'.$$

The next theorem follows by the above reconstruction arguments, but here we check it directly. The logic is that for different $R, \bar{R}$ the category of modules are both monoidally equivalent to ${}_K\mathcal{M}_K^G$ and hence monoidally equivalent but not in a manner that is compatible with the forgetful functor to Vect. Hence these should be related by a cochain twist.

Theorem 6.5.4. Let $R, \bar{R}$ be two transversals with $\bar{r} \in \bar{R}$ representing the same coset as $r \in R$. Then $\Xi(\bar{R}, K)$ is a cochain twist of $\Xi(R, K)$ at least as quasi-bialgebras (and as quasi-Hopf algebras if one of them is). The Drinfeld cochain is $\chi = \sum_{r\in R}(\delta_r\otimes 1)\otimes(1\otimes r^{-1}\bar{r})$.

Proof. Let $R, \bar{R}$ be two transversals. Then for each $r \in R$, the class rK has a unique representative $\bar{r}K$ with $\bar{r} \in \bar{R}$. Hence $\bar{r} = rc_r$ for some function $c : R \to K$ determined by the two transversals as $c_r = r^{-1}\bar{r}$ in G. One can show that the cocycle matched pairs are related by

$$x\bar{\triangleright}\bar{r} = (x\triangleright r)c_{x\triangleright r}, \quad x\bar{\triangleleft}\bar{r} = c^{-1}_{x\triangleright r}(x\triangleleft r)c_r$$

among other identities. On using

$$\begin{aligned}\bar{s}\bar{t} = sc_stc_t = s(c_s\triangleright t)(c_s\triangleleft t)c_t &= (s\cdot c_s\triangleright t)\tau(s, c_s\triangleright t)(c_s\triangleleft t)c_t\\ &= \overline{s\cdot(c_s\triangleright t)}c^{-1}_{s\cdot c_s\triangleright t}\tau(s, c_s\triangleright t)(c_s\triangleleft t)c_t\end{aligned}$$

and factorising using $\bar{R}$, we see that

$$\bar{s}\,\bar{\cdot}\,\bar{t} = \overline{s\cdot c_s\triangleright t}, \quad \bar{\tau}(\bar{s},\bar{t}) = c^{-1}_{s\cdot c_s\triangleright t}\tau(s, c_s\triangleright t)(c_s\triangleleft t)c_t. \tag{6.16}$$

We will construct a monoidal functor $G : {}_{\Xi(R,K)}\mathcal{M} \to {}_{\Xi(\bar{R},K)}\mathcal{M}$ with $g_{V,V'}(v\otimes v') = \chi^1.v\otimes\chi^2.v'$ for a suitable $\chi \in \Xi(R,K)^{\otimes 2}$. First, let $F : {}_{\Xi(R,K)}\mathcal{M} \to {}_K\mathcal{M}_K^G$ be the monoidal functor above with natural isomorphism $f_{V,V'}$ and $\bar{F} : {}_{\Xi(\bar{R},K)}\mathcal{M} \to {}_K\mathcal{M}_K^G$ the parallel for $\Xi(\bar{R}, K)$ with isomorphism $\bar{f}_{V,V'}$. Then

$$C : F \to \bar{F}\circ G, \quad C_V : F(V) = V\otimes\mathbb{C}K \to V\otimes\mathbb{C}K = \bar{F}G(V), \quad C_V(v\otimes z) = v\otimes c^{-1}_{|v|}z$$

is a natural isomorphism. Check on the right we have, denoting the $\bar{R}$ grading by $||\ ||$, the G-grading and K-bimodule structure

$$\begin{aligned}|C_V(v\otimes z)| &= |v\otimes c^{-1}_{|v|}z| = ||v||c^{-1}_{|v|}z = |v|z = |v\otimes z|,\\ x.C_V(v\otimes z).y &= x.(v\otimes c^{-1}_{|v|}z).y = x.v\otimes(x\bar{\triangleleft}||v||)c^{-1}_{|v|}zy = x.v\otimes c^{-1}_{x\triangleright|v|}(x\triangleleft|v|)zy\\ &= C_V(x.(v\otimes z).y).\end{aligned}$$

We want these two functors to not only be naturally isomorphic but for this to respect that they are both monoidal functors. Here $\bar{F}\circ G$ has the natural isomorphism

$$\bar{f}^g_{V,V'} = \bar{F}(g_{V,V'})\circ\bar{f}_{G(V),G(V')}$$

by which it is a monoidal functor.

The natural condition on a natural isomorphism C between monoidal functors is that C behaves in the obvious way on tensor product objects via the natural isomorphisms associated to each monoidal functor. In our case, this means

$$\bar{f}^g_{V,V'}\circ(C_V\otimes C_{V'})=C_{V\otimes V'}\circ f_{V,V'}:F(V)\otimes F(V')\to\bar{F}G(V\otimes V').$$

Putting in the specific form of these maps, the right hand side is

$$C_{V\otimes V'}\circ f_{V,V'}(v\otimes 1\underset{K}{\otimes}v'\otimes z)=C_{V\otimes V'}(v\otimes v'\otimes\tau(|v|,|v'|)z)=v\otimes v'\otimes c^{-1}_{|v\otimes v'|}\tau(|v|,|v'|)z,$$

while the left hand side is

$$\begin{aligned}\bar{f}^g_{V,V'}\circ(C_V\otimes C_{V'})(v\otimes 1\underset{K}{\otimes}v'\otimes z)&=\bar{f}^g_{V,V'}(v\otimes c^{-1}_{|v|}\underset{K}{\otimes}v'\otimes c^{-1}_{|v'|}z)\\&=\bar{f}^g_{V,V'}(v\otimes 1\underset{K}{\otimes}c^{-1}_{|v|}.v'\otimes(c^{-1}_{|v|}{\triangleright}||v'||)c^{-1}_{|v'|}z)\\&=\bar{F}(g_{V,V'})(v\otimes c^{-1}_{|v|}.v'\otimes\bar{\tau}(||v||,||c^{-1}_{|v|}.v'||)(c^{-1}_{|v|}{\triangleright}||v'||)c^{-1}_{|v'|}z)\\&=\bar{F}(g_{V,V'})(v\otimes c^{-1}_{|v|}.v'\otimes c^{-1}_{|v\otimes v'|}\tau(|v|,|v'|)z,\end{aligned}$$

using the second of (6.16) and $|v\otimes v'|=|v|\cdot|v'|$. We also used $\bar{f}^g_{V,V'}=\bar{F}(g_{V,V'})\bar{f}_{G(V),G(V')}:\bar{F}G(V)\otimes\bar{F}G(V')\to\bar{F}G(V\otimes V')$. Comparing, we need $\bar{F}(g_{V,V'})$ to be the action of the element

$$\chi=\sum_{r\in R}\delta_r\otimes c_r\in\Xi(R,K)^{\otimes 2}.$$

It follows from the arguments, but one can also check directly, that ϕ indeed twists as stated to $\bar{\phi}$ when these are given by Lemma 6.4.5, again using (6.16). ■

The twisting of a quasi-Hopf algebra is again one. Hence, we have:

Corollary 6.5.5. If R has $(\ )^R$ bijective giving a quasi-Hopf algebra with regular antipode $S,\alpha=1,\beta$ as in Theorem 6.4.8 and $\bar{R}$ is another transversal then $\Xi(\bar{R},K)$ in the twisting form of Theorem 6.5.4 has an antipode

$$\bar{S}=S,\quad\bar{\alpha}=\sum_r\delta_{r^R}c_r,\quad\bar{\beta}=\sum_r\delta_r\tau(r,r^R)(c_r^{-1}{\triangleleft}r^R)^{-1}.$$

This is a regular antipode if $(\ )^R$ for $\bar{R}$ is also bijective (i.e. $\bar{\alpha}$ is then invertible and can be transformed back to standard form to make it 1).

Proof. We work with the initial quasi-Hopf algebra $\Xi(R,K)$ and $\triangleright,\triangleleft,\tau$ refer to this but note that $\Xi(\bar{R},K)$ is the same algebra when δ_r is identified with the corresponding $\delta_{\bar{r}}$. Then

$$\bar{\alpha}=(S\chi^1)\chi^2=\sum_r S\delta_r\otimes c_r=\delta_{r^R}c_r$$

using the formula for $S\delta_r=\delta_{r^R}$ in Theorem 6.4.8. Similarly, $\chi^{-1}=\sum_r\delta_r\otimes c_r^{-1}$ and we use S,β from the above lemma, where

$$S(1\otimes x)=\sum_s\delta_{(x^{-1}\triangleright s)^R}\otimes x^{-1}{\triangleleft}s=\sum_t\delta_{t^R}\otimes x^{-1}{\triangleleft}(x{\triangleright}t)=\sum_t\delta_{t^R}\otimes(x{\triangleleft}t)^{-1}.$$

Then

$$\begin{aligned}\bar{\beta} = \chi^{-1}\beta S\chi^{-2} &= \sum_{r,s,t}\delta_r\delta_s\tau(s,s^R)\delta_{t^R}(c_r^{-1}\triangleleft t)^{-1}\\ &= \sum_{r,t}\delta_r\tau(r,r^R)\delta_{t^R}(c_r^{-1}\triangleleft t)^{-1} = \sum_{r,t}\delta_r\delta_{\tau(r,r^R)\triangleright t^R}\tau(r,r^R)(c_r^{-1}\triangleleft t)^{-1}.\end{aligned}$$

Commuting the δ-functions to the left requires $r=\tau(r,r^R)\triangleright t^R$ or $r^{RR}=\tau(r,r^R)^{-1}\triangleright r=t^R$ so $t=r^R$ under our assumptions, giving the answer stated.

If $(\ )^R$ is bijective then $\bar{\alpha}^{-1}=\sum_r c_r^{-1}\delta_{r^R}=\sum_r\delta_{c_r^{-1}\triangleright r^R}c_r^{-1}$ provides the left inverse. On the other side, we need $c_r^{-1}\triangleright r^R=c_s^{-1}\triangleright s^R$ iff $r=s$. This is true if $(\ )^R$ for $\bar{R}$ is also bijective. That is because, if we write $(\ )^{\bar{R}}$ for the right inverse with respect to $\bar{R}$, one can show by comparing the factorisations that

$$\bar{s}^{\bar{R}}=\overline{c_s^{-1}\triangleright s^R},\quad \overline{s^R}=c_s\bar{\triangleright}\bar{s}^{\bar{R}}$$

and we use the first of these. ■

Example 6.5.6. With reference to the list of transversals for $S_2\subset S_3$, we have four quasi-Hopf algebras of which two were already computed in Example 6.4.10.

(i) 2nd transversal as twist of the first. Here $\bar{\Xi}$ is generated by $\mathbb{Z}_2$ as u again and $\delta_{\bar{r}}$ with $\bar{R}=\{e,w,v\}$. We have the same cosets represented by these with $\bar{e}=e$, $\overline{uv}=w$ and $\overline{vu}=v$, which means $c_e=e, c_{vu}=u, c_{uv}=u$. To compare the algebras in the two cases, we identify $\delta_0=\delta_e,\delta_1=\delta_w,\delta_2=\delta_v$ as delta-functions on G/K (rather than on G) in order to identify the algebras of $\bar{\Xi}$ and Ξ. The cochain from Theorem 6.5.4 is

$$\chi=\delta_e\otimes e+(\delta_{vu}+\delta_{uv})\otimes u=\delta_0\otimes 1+(\delta_1+\delta_2)\otimes u=\delta_0\otimes 1+(1-\delta_0)\otimes u$$

as an element of $\Xi\otimes\Xi$. One can check that this conjugates the two coproducts as claimed. We also have

$$\chi^2=1\otimes 1,\quad (\epsilon\otimes\mathrm{id})\chi=(\mathrm{id}\otimes\epsilon)\chi=1.$$

We spot check (6.16), for example $v\bar{\cdot}w=\overline{vu}\,\bar{\cdot}\,\overline{uv}=\overline{uv}=\overline{vuvu}=\overline{vu(u\triangleright(uv))}$, as it had to be. We should therefore find that

$$((\Delta\otimes\mathrm{id})\chi)\chi_{12}=((\mathrm{id}\otimes\Delta)\chi)\chi_{23}\bar{\phi}.$$

We have checked directly that this indeed holds. Next, the antipode of the first transversal should twist to

$$\bar{S}=S,\quad \bar{\alpha}=\delta_e c_e+\delta_{uv}c_{vu}+\delta_{vu}c_{uv}=\delta_e(e-u)+u=\delta_e c_e+\delta_{vu}c_{vu}+\delta_{uv}c_{uv}=\bar{\beta}$$

by Corollary 6.5.5 for twisting the antipode. Here, $U=\bar{\alpha}^{-1}=\bar{\beta}=U^{-1}$ and $\bar{S}'=U(S\ )U^{-1}$ with $\bar{\alpha}'=\bar{\beta}'=1$ should also be an antipode. We can check this:

$$Uu=(\delta_0(e-u)+u)u=\delta_0(u-e)+e=u(\delta_{u^{-1}\triangleright 0}(e-u)+u)=uU$$

so $\bar{S}'u=UuU^{-1}=u$, and

$$\bar{S}'\delta_1=U(S\delta_1)U=U\delta_2 U=(\delta_0(e-u)+u)\delta_2(\delta_0(e-u)+u)=\delta_1.$$

(ii) 3rd transversal as a twist of the first. A mixed up choice is $\bar R=\{e,uv,v\}$ which is not a subgroup so τ is nontrivial. One has

$$\tau(uv,uv)=\tau(v,uv)=\tau(uv,v)=u,\quad \tau(v,v)=e,\quad v{\cdot}v=e,\quad v{\cdot}uv=uv,\quad uv{\cdot}v=e,\quad uv.{\cdot}uv=v,$$

$$u{\triangleright}v=uv,\quad u{\triangleright}(uv)=v,\quad u{\triangleleft}v=e,\quad u{\triangleleft}uv=e$$

and all other cases implied from the properties of e. Here $v^R=v$ and $(uv)^R=v$. These are with respect to $\bar R$, but note that twisting calculations will take place with respect to R.

Writing $\delta_0=\delta_e,\delta_1=\delta_{uv},\delta_2=\delta_v$ we have the same algebra as before (as we had to) and now the coproduct etc.,

$$\bar\Delta u=u\otimes 1+\delta_0u\otimes(u-1),\quad \bar\Delta\delta_0=\delta_0\otimes\delta_0+\delta_2\otimes\delta_2+\delta_1\otimes\delta_2$$

$$\bar\Delta\delta_1=\delta_0\otimes\delta_1+\delta_1\otimes\delta_0+\delta_2\otimes\delta_1,\quad \bar\Delta\delta_2=\delta_0\otimes\delta_2+\delta_2\otimes\delta_0+\delta_1\otimes\delta_1,$$
$$\bar\phi=1\otimes1\otimes1+(\delta_1\otimes\delta_2+\delta_2\otimes\delta_1+\delta_1\otimes\delta_1)(u-1)=\bar\phi^{-1}$$

for the quasibialgebra. We used the $\tau,\triangleright,\triangleleft,\cdot$ for $\bar R$ for these direct calculations.

Now we consider twisting with

$$c_0=e,\quad c_1=(uv)^{-1}uv=1,\quad c_2=v^{-1}vu=u,\quad \chi=1\otimes1+\delta_2\otimes(u-1)=\chi^{-1}$$

and check twisting the coproducts

$$(1\otimes1+\delta_2\otimes(u-1))(u\otimes u)(1\otimes1+\delta_2u\otimes(u-1))=u\otimes1+\delta_0\otimes(u-1)=\bar\Delta u,$$

$$(1\otimes1+\delta_2\otimes(u-1))(\delta_0\otimes\delta_0+\delta_1\otimes\delta_2+\delta_2\otimes\delta_1)(1\otimes1+\delta_2\otimes(u-1))=\bar\Delta\delta_0,$$
$$(1\otimes1+\delta_2\otimes(u-1))(\delta_0\otimes\delta_1+\delta_1\otimes\delta_0+\delta_2\otimes\delta_2)(1\otimes1+\delta_2\otimes(u-1))=\bar\Delta\delta_1,$$
$$(1\otimes1+\delta_2\otimes(u-1))(\delta_0\otimes\delta_2+\delta_2\otimes\delta_0+\delta_1\otimes\delta_1)(1\otimes1+\delta_2\otimes(u-1))=\bar\Delta\delta_2.$$

One can also check that (6.16) hold, e.g. for the first half,

$$\bar2=\bar1{\cdot}\bar1=\overline{1+c_1{\triangleright}1}=\overline{1+1},\quad \bar0=\bar1{\cdot}\bar2=\overline{1+c_1{\triangleright}2}=\overline{1+2},$$

$$\bar1=\bar2{\cdot}\bar1=\overline{2+c_2{\triangleright}1}=\overline{2+2},\quad \bar0=\bar2{\cdot}\bar2=\overline{2+c_2{\triangleright}2}=\overline{2+1}$$

as it must.

Now we apply the twisting of antipodes in Corollary 6.5.5, remembering to do calculations now with R where $\tau,\triangleleft$ are trivial, to get

$$\bar S=S,\quad \bar\alpha=\delta_0+\delta_1c_2+\delta_2c_1=1+\delta_1(u-1),\quad \bar\beta=\delta_0+\delta_2c_2+\delta_1c_1=1+\delta_2(u-1),$$

which obey $\bar\alpha^2=\bar\alpha$ and $\bar\beta^2=\bar\beta$ and are therefore not (left or right) invertible. Hence, we cannot set either equal to 1 by U and there is an antipode, but it is not regular. One can check the antipode indeed works:

$$\begin{aligned}(Su)\alpha+(Su)(S\delta_0)\alpha(u-1)&=u(1+\delta_1(u-1))+\delta_0u(1+\delta_1(u-1))(u-1)\\&=u+\delta_2(1-u)+\delta_0(1-u)=u+(1-\delta_1)(1-u)=\alpha\\ u\beta+\delta_0u\beta S(u-1)&=u(1+\delta_2(u-1))+\delta_0u(1+\delta_2(u-1))(u-1)\\&=u+\delta_1(1-u)+\delta_0(1-u)=u+(1-\delta_2)(1-u)=\beta\end{aligned}$$

$$\begin{aligned}(S\delta_0)\alpha\delta_0+(S\delta_2)\alpha\delta_2+(S\delta_1)\alpha\delta_2&=\delta_0(1+\delta_1(u-1))\delta_0+(1-\delta_0)(1+\delta_1(u-1))\delta_2\\&=\delta_0+(1-\delta_0)\delta_2+\delta_1(\delta_1u-\delta_2)=\delta_0+\delta_2+\delta_1u=\alpha\\ \delta_0\beta S\delta_0+\delta_2\beta S\delta_2+\delta_1\beta S\delta_2&=\delta_0(1+\delta_2(u-1))\delta_0+(1-\delta_0)(1+\delta_2(u-1))\delta_1\\&=\delta_0+(1-\delta_0)\delta_1+(1-\delta_0)\delta_2(u-1)\delta_1=\delta_0+\delta_1+\delta_2(\delta_2u-\delta_1)=\beta\end{aligned}$$

and more simply on δ_1,δ_2.

The fourth transversal has a similar pattern to the 3rd, so we do not list its coproduct etc. explicitly.

In general, there will be many different choices of transversal. For $S_{n-1}\subset S_n$, the first two transversals for $S_2\subset S_3$ generalise as follows, giving a Hopf algebra and a strictly quasi-Hopf algebra respectively.

Example 6.5.7. *(*i) First transversal. Here $R=\mathbb{Z}_n$ is a subgroup with $i=0,1,\cdots,n-1$ mod n corresponding to the elements $(12\cdots n)^i$. Neither subgroup is normal for $n\geq 4$, so both actions are nontrivial but τ is trivial. This expresses S_n as a double cross product $\mathbb{Z}_n{\bowtie}S_{n-1}$ (with trivial τ) and the matched pair of actions

$$\sigma{\triangleright}i=\sigma(i),\quad(\sigma{\triangleleft}i)(j)=\sigma(i+j)-\sigma(i)$$

for $i,j=1,\cdots,n-1$, where we add and subtract mod n but view the results in the range $1,\cdots,n$. This was actually found by twisting from the 2nd transversal below, but we can check it directly as follows. First.

$$\sigma(1\cdots n)^i=(\sigma{\triangleright}i)(\sigma{\triangleleft}i)=(12\cdots n)^{\sigma(i)}\left((1\cdots n)^{-\sigma(i)}\sigma(12\cdots n)^i\right)$$

and we check that the second factor sends $n\to i\to\sigma(i)\to n$, hence lies in S_n. It follows by the known fact of unique factorisation into these subgroups that this factor is $\sigma{\triangleleft}i$. Its action on $j=1,\cdots,n-1$ is

$$(\sigma{\triangleright}i)(j)=(12\cdots n)^{-\sigma(i)}\sigma(12\cdots n)^i(j)=\begin{cases}n-\sigma(i)&i+j=n\\ \sigma(i+j)-\sigma(i)&i+j\neq n\end{cases}=\sigma(i+j)-\sigma(i),$$

where $\sigma(i+j)\neq\sigma(i)$ as $i+j\neq i$ and $\sigma(n)=n$ as $\sigma\in S_{n-1}$. It also follows since the two factors are subgroups that these are indeed a matched pair of actions. We can also check the matched pair axioms directly. Clearly, $\triangleright$ is an action and

$$\sigma(i)+(\sigma{\triangleleft}i)(j)=\sigma(i)+\sigma(i+j)-\sigma(i)=\sigma{\triangleright}(i+j)$$

for $i,j\in\mathbb{Z}_n$. On the other side,

$$\begin{aligned}((\sigma{\triangleleft}i){\triangleleft}j)(k)&=(\sigma{\triangleleft}i)(j+k)-(\sigma{\triangleleft}i)(j)=\sigma(i+(j+k))-\sigma(i)-\sigma(i+j)+\sigma(i)\\&=\sigma((i+j)+k)-\sigma(i+j)=(\sigma{\triangleleft}(i+j))(k),\\((\sigma{\triangleleft}(\tau{\triangleright}i))(\tau{\triangleleft}i))(j)&=(\sigma{\triangleleft}\tau(i))(\tau(i+j))-\tau(i))=\sigma(\tau(i)+\tau(i+j)-\tau(i))-\sigma(\tau(i))\\&=\sigma(\tau(i+j))-\sigma(\tau(i))=((\sigma\tau){\triangleleft}i)(j)\end{aligned}$$

for $i,j\in\mathbb{Z}_n$ and $k\in 1,\cdots,n-1$.

This gives $\mathbb{C}S_{n-1}{\triangleright\!\!\blacktriangleleft}\mathbb{C}(\mathbb{Z}_n)$ as a natural bicrossproduct Hopf algebra which we identify with Ξ (which we prefer to build on the other tensor product order). From Lemma 6.4.5 and Theorem 6.4.8, this is spanned by products of δ_i for $i=0,\cdots n-1$ as our labelling of $R=\mathbb{Z}_n$ and $\sigma\in S_{n-1}=K$, with cross relations $\sigma\delta_i=\delta_{\sigma(i)}\sigma$, $\sigma\delta_0=\delta_0\sigma$, and coproduct etc.,

$$\Delta\delta_i=\sum_{j\in\mathbb{Z}_n}\delta_j\otimes\delta_{i-j},\quad \Delta\sigma=\sigma\delta_0+\sum_{i=1}^{n-1}(\sigma{\triangleleft}i),\quad \epsilon\delta_i=\delta_{i,0},\quad \epsilon\sigma=1,$$

$$S\delta_i=\delta_{-i},\quad S\sigma=\sigma^{-1}\delta_0+(\sigma^{-1}{\triangleleft}i)\delta_{-i},$$

where $\sigma{\triangleleft}i$ is as above for $i=1,\cdots,n-1$. This is a usual Hopf $*$-algebra with $\delta_i^*=\delta_i$ and $\sigma^*=\sigma^{-1}$.

(ii) 2nd transversal. Here $R=\{e,(1\,n),(2\,n),\cdots,(n-1\,n)\}$, which has nontrivial $\triangleright$ in which S_{n-1} permutes the 2-cycles according to the i label, but again trivial $\triangleleft$ since

$$\sigma(i\,n)=(\sigma(i)\,n)\sigma,\quad \sigma{\triangleright}(i\ n)=(\sigma(i)\,n)$$

for all $i=1,\cdots,n-1$ and $\sigma\in S_{n-1}$. It has nontrivial τ as

$$(i\,n)(j\,n)=(j\,n)(i\,j)\Rightarrow(i\,n)\cdot(j\,n)=(j\,n),\quad \tau((i\,n),(j\,n))=(ij)$$

for $i\neq j$ and we see that $\cdot$ has right but not left division or left but not right cancellation. We also have $(in)\cdot(in)=e$ and $\tau((in),(in))=e$ so that $(\ )^R$ is the identity map, hence R is regular.

This transversal gives a cross-product quasiHopf algebra $\Xi=\mathbb{C}S_{n-1}{\bowtie}_\tau\mathbb{C}(R)$ where R is a left quasigroup (i.e. unital and with left cancellation) except that we prefer to write it with the tensor factors in the other order. From Lemma 6.4.5 and Theorem 6.4.8, this is spanned by products of δ_i and $\sigma\in S_{n-1}$, where δ_0 is the delta function at $e\in R$ and δ_i at (i,n) for $i=1,\cdots,n-1$. The cross relations have the same algebra $\sigma\delta_i=\delta_{\sigma(i)}\sigma$ for $i=1,\cdots,n-1$ as before but now the tensor coproduct etc., and nontrivial associator

$$\Delta\delta_0=\sum_{i=0}^{n-1}\delta_i\otimes\delta_i,\quad \Delta\delta_i=1\otimes\delta_i+\delta_i\otimes\delta_0,\quad \Delta\sigma=\sigma\otimes\sigma,\quad \epsilon\delta_i=\delta_{i,0},\quad \epsilon\sigma=1,$$

$$S\delta_i=\delta_i,\quad S\sigma=\sigma^{-1},\quad \alpha=\beta=1,$$

$$\phi=(1\otimes\delta_0+\delta_0\otimes(1-\delta_0)+\sum_{i=1}^{n-1}\delta_i\otimes\delta_i)\otimes 1+\sum_{\substack{i,j=1\\ i\neq j}}^{n-1}\delta_i\otimes\delta_j\otimes(ij).$$

This time we have nontrivial

$$\gamma=1,\quad \mathcal{G}=1\otimes\delta_0+\delta_0\otimes(1-\delta_0)+\sum_{i=1}^{n-1}\delta_i\otimes\delta_i+\sum_{\substack{i,j=1\\ i\neq j}}^{n-1}\delta_i(ij)\otimes\delta_j(ij)$$

in Proposition 6.4.9 from the $*$-quasi-Hopf structure in the Appendix 24.

(iii) Twisting between the above two transversals. We denote the first transversal $R=\mathbb{Z}_n$, where i is identified with $(12\cdots n)^i$, and we denote the 2nd transversal by $\bar R$ with corresponding elements $\bar i=(i\ n)$. Then

$$c_i=(12\cdots n)^{-i}(i\ n)\in S_{n-1},\quad c_i(j)=\begin{cases}n-i & j=i\\ j-i & else\end{cases}$$

for $i, j = 1, \cdots, n-1$. If we use the stated $\triangleright$ for the first transversal then one can check that the first half of (6.16) holds,

$$\overline{i + c_i \triangleright i} = \overline{i + n - i} = e = \bar{\bar{i}} \cdot \bar{i}, \quad \overline{i + c_i \triangleright j} = \overline{i + j - i} = \bar{j} = \bar{\bar{i}} \cdot \bar{j}$$

as it must. We can also check that the actions are indeed related by twisting. Thus,

$$\sigma \triangleleft \bar{i} = c^{-1}_{\sigma \triangleright i}(\sigma \triangleleft i) c_i = (\sigma(i), n)(12 \cdots n)^{\sigma(i)}(\sigma \triangleleft i)(12 \cdots n)^{-i}(i, n) = (\sigma(i), n)\sigma(i, n) = \sigma$$

$$\sigma \bar{\triangleright} \bar{i} = (\sigma \triangleright i) c_{\sigma \triangleright i} = (12 \cdots n)^{\sigma(i)}(12 \cdots n)^{-\sigma(i)}(\sigma(i), n) = (\sigma(i), n),$$

where we did the computation with $\mathbb{Z}_n$ viewed in S_n.

It follows that the Hopf algebra from case (i) cochain twists to a simpler quasihopf algebra in case (ii). The required cochain from Theorem 6.5.4 is

$$\chi = \delta_0 \otimes 1 + \sum_{i=1}^{n-1} \delta_i \otimes (12 \cdots n)^{-i}(in).$$

The above example is a little similar to the Drinfeld $U_q(g)$ as Hopf algebras which are cochain twists of $U(g)$ viewed as a quasi-Hopf algebra. We conclude with the promised example related to the octonions. This is a version of [KM10A, Example 4.6], but with left and right swapped and some cleaned up conventions.

Example 6.5.8. We let $G = Cl_3 \rtimes \mathbb{Z}_2^3$, where Cl_3 is generated by $1, -1$ and e_i, $i = 1, 2, 3$, with relations

$$(-1)^2 = 1, \quad (-1)e_i = e_i(-1), \quad e_i^2 = -1, \quad e_i e_j = -e_j e_i$$

for $i \neq j$ and the usual combination rules for the product of signs. Its elements can be enumerated as $\pm e_{\vec{a}}$ where $\vec{a} \in \mathbb{Z}_2^3$ is viewed in the additive group of 3-vectors with entries in the field $\mathbb{F}_2 = \{0, 1\}$ of order 2 and

$$e_{\vec{a}} = e_1^{a_1} e_2^{a_2} e_3^{a_3}, \quad e_{\vec{a}} e_{\vec{b}} = e_{\vec{a}+\vec{b}}(-1)^{\sum_{i \geq j} a_i b_j}.$$

This is the twisted group ring description of the 3-dimensional Clifford algebra over $\mathbb{R}$ in [AM99], but now restricted to coefficients $0, \pm 1$ to give a group of order 16. For an example,

$$e_{110} e_{101} = e_2 e_3 e_1 e_3 = e_1 e_2 e_3^2 = -e_1 e_2 = -e_{011} = -e_{110+101}$$

with the sign given by the formula.

We similarly write the elements of $K = \mathbb{Z}_2^3$ multiplicatively as $g^{\vec{a}} = g_1^{a_1} g_1^{a_2} g_3^{a_3}$ labelled by 3-vectors with values in $\mathbb{F}_2$. The generators g_i commute and obey $g_i^2 = e$. The general group product becomes the vector addition, and the cross relations are

$$(-1)g_i = g_i(-1), \quad e_i g_i = -g_i e_i, \quad e_i g_j = g_j e_i$$

for $i \neq j$. This implies that G has order 128.

(i) If we take $R = Cl_3$ itself then this will be a subgroup and we will have for $\Xi(R, K)$ an ordinary Hopf $*$-algebra as a semidirect product $\mathbb{C}\mathbb{Z}_2^3 \bowtie \mathbb{C}(Cl_3)$ except that we build it on the opposite tensor product.

(ii) Instead, we take as representatives the eight elements again labelled by 3-vectors over $\mathbb{F}_2$,

$$r_{000} = 1, \quad r_{001} = e_3, \quad r_{010} = e_2, \quad r_{011} = e_2 e_3 g_1$$

$$r_{100}=e_1,\quad r_{101}=e_1e_3g_2,\quad r_{110}=e_1e_2g_3,\quad r_{111}=e_1e_2e_3g_1g_2g_3$$

and their negations, as a version of [KM10A, Example 4.6]. This can be written compactly as

$$r_{\vec a}=e_{\vec a}g_1^{a_2a_3}g_2^{a_1a_3}g_3^{a_1a_2}$$

Proposition 6.5.9. [KM10A] This choice of transversal makes $(R,\cdot)$ the octonion two sided inverse property quasigroup $G_{\mathbb{O}}$ in the Albuquerque-Majid description of the octonions[AM99],

$$r_{\vec a}\cdot r_{\vec b}=(-1)^{f(\vec a,\vec b)}r_{\vec a+\vec b},\quad f(\vec a,\vec b)=\sum_{i\geq j}a_ib_j+a_1a_2b_3+a_1b_2a_3+b_1a_2a_3$$

with the product on signed elements behaving as if bilinear. The action $\triangleleft$ is trivial, and the left action and cocycle τ are

$$g^{\vec a}{\triangleright}r_{\vec b}=(-1)^{\vec a\cdot\vec b}r_{\vec b},\quad \tau(r_{\vec a},r_{\vec b})=g^{\vec a\times\vec b}=g_1^{a_2b_3+a_3b_2}g_2^{a_3b_1+a_1b_3}g_3^{a_1b_2+a_2b_1}$$

with the action extended with signs as if linearly and τ independent of signs in either argument.

Proof. We check in the group

$$\begin{aligned}r_{\vec a}r_{\vec b}&=e_{\vec a}g_1^{a_2a_3}g_2^{a_1a_3}g_3^{a_1a_2}e_{\vec b}g_1^{b_2b_3}g_2^{b_1b_3}g_3^{b_1b_2}\\&=e_{\vec a}e_{\vec b}(-1)^{b_1a_2a_3+b_2a_1a_3+b_3a_1a_2}g_1^{a_2a_3+b_2b_3}g_2^{a_1a_3+b_1b_3}g_3^{a_1a_2+b_1b_2}\\&=(-1)^{f(a,b)}r_{\vec a+\vec b}g_1^{a_2a_3+b_2b_3-(a_2+b_2)(a_3+b_3)}g_2^{a_1a_3+b_1b_3-(a_1+b_1)(a_3+b_3)}g_3^{a_1a_2+b_1b_2-(a_1+b_1)(a_2+b_2)}\\&=(-1)^{f(a,b)}r_{\vec a+\vec b}g_1^{a_2b_3+b_2a_3}g_2^{a_1b_3+b_1a_3}g_3^{a_1b_2+b_1a_2},\end{aligned}$$

from which we read off $\cdot$ and τ. For the second equality, we moved the g_i to the right using the commutation rules in G. For the third equality we used the product in Cl_3 in our description above and then converted $e_{\vec a+\vec b}$ to $r_{\vec a+\vec b}$. ■

The product of the quasigroup $G_{\mathbb{O}}$ here is the same as the octonions product as an algebra over $\mathbb{R}$ in the description of [AM99], restricted to elements of the form $\pm r_{\vec a}$. The cocycle-associativity property of $(R,\cdot)$ says

$$r_{\vec a}\cdot(r_{\vec b}\cdot r_{\vec c})=(r_{\vec a}\cdot r_{\vec b})\cdot\tau(\vec a,\vec b){\triangleright}r_{\vec c}=(r_{\vec a}\cdot r_{\vec b})\cdot r_{\vec c}(-1)^{(\vec a\times\vec b)\cdot\vec c}$$

giving -1 exactly when the 3 vectors are linearly independent as 3-vectors over $\mathbb{F}_2$. One also has $r_{\vec a}\cdot r_{\vec b}=\pm r_{\vec b}\cdot r_{\vec a}$ with -1 exactly when the two vectors are linearly independent, which means both nonzero and not equal, and $r_{\vec a}\cdot r_{\vec a}=\pm1$ with -1 exactly when the one vector is linearly independent, i.e. not zero. (These are exactly the quasiassociativity, quasicommutativity and norm properties of the octonions algebra in the description of [AM99].) The 2-sided inverse is

$$r_{\vec a}^{-1}=(-1)^{n(\vec a)}r_{\vec a},\quad n(0)=0,\quad n(\vec a)=1,\quad\forall\vec a\neq0$$

with the inversion operation extended as usual with respect to signs.

The quasi-Hopf algebra $\Xi(R,K)$ is spanned by $\delta_{(\pm,\vec a)}$ labelled by the points of R and products of the g_i with the relations $g^{\vec a}\delta_{(\pm,\vec b)}=\delta_{(\pm(-1)^{\vec a\cdot\vec b},\vec b)}g^{\vec a}$ and tensor coproduct etc.,

$$\Delta\delta_{(\pm,\vec a)}=\sum_{(\pm',\vec b)}\delta_{(\pm',\vec b)}\otimes\delta_{(\pm\pm'(-1)^{n(\vec b)},\vec a+\vec b)},\quad\Delta g^{\vec a}=g^{\vec a}\otimes g^{\vec a},\quad\epsilon\delta_{(\pm,\vec a)}=\delta_{\vec a,0}\delta_{\pm,+},\quad\epsilon g^{\vec a}=1,$$

$$S\delta_{(\pm,\vec{a})}=\delta_{(\pm(-1)^{n(\vec{a})},\vec{a}},\quad Sg^{\vec{a}}=g^{\vec{a}},\quad \alpha=\beta=1,\quad \phi=\sum_{(\pm,\vec{a}),(\pm',\vec{b})}\delta_{(\pm,\vec{a})}\otimes\delta_{(\pm',\vec{b})}\otimes g^{\vec{a}\times\vec{b}}$$

We also have $*$ the identity on $\delta_{(\pm,\vec{a})}, g^{\vec{a}}$ and nontrivial

$$\gamma=1,\quad \mathcal{G}=\sum_{(\pm,\vec{a}),(\pm',\vec{b})}\delta_{(\pm,\vec{a})}g^{\vec{a}\times\vec{b}}\otimes\delta_{(\pm',\vec{b})}g^{\vec{a}\times\vec{b}}$$

in Proposition 6.4.9. The general form here is not unlike our S_n example.

6.5.3 Module categories context

This section does not contain anything new beyond [Ost03A, EGNO10], but completes the categorical picture that connects our algebra $\Xi(R,K)$ to the more general context of module categories, adapted to our notations.

Our first observation is that if $\otimes:\mathcal{C}\times\mathcal{V}\to\mathcal{V}$ is a left action of a monoidal category $\mathcal{C}$ on a category $\mathcal{V}$ (one says that $\mathcal{V}$ is a left $\mathcal{C}$-module) then one can check that this is the same thing as a monoidal functor $F:\mathcal{C}\to\mathrm{End}(\mathcal{V})$ where the set $\mathrm{End}(\mathcal{V})$ of endofunctors can be viewed as a strict monoidal category with monoidal product the endofunctor composition $\circ$. Here $\mathrm{End}(\mathcal{V})$ has monoidal unit $\mathrm{id}_{\mathcal{V}}$ and its morphisms are natural transformations between endofunctors. F just sends an object $X\in\mathcal{C}$ to $X\otimes(\ )$ as a monoidal functor from $\mathcal{V}$ to $\mathcal{V}$. A monoidal functor comes with natural isomorphisms $\{f_{X,Y}\}$ and these are given tautologically by

$$f_{X,Y}(V):F(X)\circ F(Y)(V)=X\otimes(Y\otimes V)\cong(X\otimes Y)\otimes V=F(X\otimes Y)(V)$$

as part of the monoidal action. Conversely, if given a functor F, we define $X\otimes V=F(X)V$ and extend the monoidal associativity of $\mathcal{C}$ to mixed objects using $f_{X,Y}$ to define $X\otimes(Y\otimes V)=F(X)\circ F(Y)V\cong F(X\otimes Y)V=(X\otimes Y)\otimes V$. The notion of a left module category is a categorification of the bijection between an algebra action $\cdot:A\otimes V\to V$ and a representation as an algebra map $A\to\mathrm{End}(V)$. There is an equally good notion of a right $\mathcal{C}$-module category extending $\otimes$ to $\mathcal{V}\times\mathcal{C}\to\mathcal{V}$. In the same way as one uses $\cdot$ for both the algebra product and the module action, it is convenient to use $\otimes$ for both in the categorified version. Similarly for the right module version.

Another general observation is that if $\mathcal{V}$ is a $\mathcal{C}$-module category for a monoidal category $\mathcal{C}$ then $\mathrm{Fun}_{\mathcal{C}}(\mathcal{V},\mathcal{V})$, the (left exact) functors from $\mathcal{V}$ to itself that are compatible with the action of $\mathcal{C}$, is another monoidal category. This is denoted $\mathcal{C}^*_{\mathcal{V}}$ in [EGNO10], but should not be confused with the dual of a monoidal functor which was one of the origins[Maj91] of the centre $\mathcal{Z}(\mathcal{C})$ construction as a special case. Also note that if $A\in\mathcal{C}$ is an algebra in the category then $\mathcal{V}={}_A\mathcal{C}$, the left modules of A in the category, is a *right* $\mathcal{C}$-module category. If V is an A-module then we define $V\otimes X$ as the tensor product in $\mathcal{C}$ equipped with an A-action from the left on the first factor. Moreover, for certain 'nice' right module categories $\mathcal{V}$, there exists a suitable algebra $A\in\mathcal{C}$ such that $\mathcal{V}\simeq{}_A\mathcal{C}$, see [Ost03A][EGNO10, Thm 7.10.1] in other conventions. For such module categories, $\mathrm{Fun}_{\mathcal{C}}(\mathcal{V},\mathcal{V})\simeq{}_A\mathcal{C}_A$ the category of A-A-bimodules in $\mathcal{C}$. Here, if given an A-A-bimodule E in $\mathcal{C}$, the corresponding endofunctor is given by $E\otimes_A(\ )$, where we require $\mathcal{C}$ to be Abelian so that we can define $\otimes_A$. This turns $V\in{}_A\mathcal{C}$ into another A-module in $\mathcal{C}$ and $E\otimes_A(V\otimes X)\cong(E\otimes_A V)\otimes X$, so the construction commutes with the right $\mathcal{C}$-action.

Before we explain how these abstract ideas lead to ${}_K\mathcal{M}^G_K$, a more 'obvious' case is the study of left module categories for $\mathcal{C}={}_G\mathcal{M}$. If $K\subseteq G$ is a subgroup, we set $\mathcal{V}={}_K\mathcal{M}$ for i :

$K \subseteq G$. The functor $\mathcal{C} \to \mathrm{End}(\mathcal{V})$ just sends $X \in \mathcal{C}$ to $i^*(X) \otimes (\ )$ as a functor on $\mathcal{V}$, or more simply $\mathcal{V}$ is a left $\mathcal{C}$-module by $X \otimes V = i^*(X) \otimes V$. More generally[Ost03A][EGNO10, Example 7..4.9], one can include a cocycle $\alpha \in H^2(K, \mathbb{C}^\times)$ since we are only interested in monoidal equivalence, and this data (K, α) parametrises all indecomposable left ${}_G\mathcal{M}$-module categories. Moreover, here $\mathrm{End}(\mathcal{V}) \simeq {}_K\mathcal{M}_K$, the category of K-bimodules, where a bimodule E acts by $E \otimes_{\mathbb{C}K} (\ )$. So the data we need for a ${}_G\mathcal{M}$-module category is a monoidal functor ${}_G\mathcal{M} \to {}_K\mathcal{M}_K$. This is of potential interest but is not the construction we were looking for.

Rather, we are interested in right module categories of $\mathcal{C} = \mathcal{M}^G$, the category of G-graded vector spaces. It turns out that these are classified by the exact same data (K, α) (this is related to the fact that the $\mathcal{M}^G, {}_G\mathcal{M}$ have the same centre) but the construction is different. Thus, if $K \subseteq G$ is a subgroup, we consider $A = \mathbb{C}K$ regarded as an algebra in $\mathcal{C} = \mathcal{M}^G$ by $|x| = x$ viewed in G. One can also twist this by a cocycle α, but here we stick to the trivial case. Then $\mathcal{V} = {}_A\mathcal{C} = {}_K\mathcal{M}^G$, the category of G-graded left K-modules, is a right $\mathcal{C}$-module category. Explicitly, if $X \in \mathcal{C}$ is a G-graded vector space and $V \in \mathcal{V}$ a G-graded left K-module then

$$V \otimes X, \quad x.(v \otimes w) = v.x \otimes w, \quad |v \otimes w| = |v||w|, \quad \forall\ v \in V,\ w \in X$$

is another G-graded left K-module. Finally, by the general theory, there is an associated monoidal category

$$\mathcal{C}^*_{\mathcal{V}} := \mathrm{Fun}_{\mathcal{C}}(\mathcal{V}, \mathcal{V}) \simeq {}_K\mathcal{M}^G_K \simeq {}_{\Xi(R,K)}\mathcal{M}.$$

which is the desired category to describe quasiparticles on boundaries in [KK12]. Conversely, if $\mathcal{V}$ is an indecomposable right $\mathcal{C}$-module category for $\mathcal{C} = \mathcal{M}^G$, it is explained in [Ost03A][EGNO10, Example 7.4.10] (in other conventions) that the set of indecomposable objects has a transitive action of G and hence can be identified with G/K for some subgroup $K \subseteq G$. This can be used to put the module category up to equivalence in the above form (with some cocycle α).

6.6 Concluding remarks

We have given a detailed account of the algebra behind the treatment of boundaries in the Kitaev model based on subgroups K of a finite group G, as well as how it sits between the abstract categorical picture on the one hand and concrete applications on the other. New results include the quasi-bialgebra $\Xi(R, K)$ in full generality, a more direct derivation from the category ${}_K\mathcal{M}^G_K$ that connects to the module category point of view, a theorem that $\Xi(R, K)$ changes by a Drinfeld twist as R changes, and a $*$-quasi-Hopf algebra structure that ensures a nice properties for the category of representations (these form a strong bar category) and for the standard antipode S. On the computer science side, we edged towards how one might use these ideas in quantum computations and detect quasiparticles across ribbons where one end is on a boundary. We also gave new decomposition formulae relating representations of $D(G)$ in the bulk to those of $\Xi(R, K)$ in the boundary.

Both the algebraic and the computer science aspects can be taken much further. The case treated here of trivial cocycle α is already complicated enough but the ideas do extend to include these and should similarly be worked out. Whereas most of the abstract literature on such matters is at the conceptual level only up to categorical equivalence, we set out to give constructions more explicitly, which we believe is essential for concrete calculations and should also be relevant to the physics. For example, much of the literature on anyons is devoted to so-called F-moves which express the associativity isomorphisms even though, by Mac Lane's theorem, monoidal categories are equivalent to strict ones. On the physics side, the covariance properties of ribbon operators also involve the coproduct and hence how they are realised depends on the choice of R. The same applies to how $*$ interacts with tensor products, which would be relevant to the unitarity properties of composite systems. Of interest, for example, should be the case of a lattice divided into two parts A, B with a boundary between them and how the entropy of states in the total space relate to those in the subsystem. This is an idea of considerable interest in quantum gravity, but the latter has certain parallels with quantum computing and could be explored concretely using the results of the Chapter. We also would like to expand further the concrete use of patches and lattice surgery, as we considered only the cases of boundaries with $K = \{e\}$ and $K = G$, and only a square geometry. Additionally, it would be useful to know under what conditions the model gives universal quantum computation. While there are broadly similar such ideas in the physics literature, e.g., [CCW16], we believe our fully explicit treatment will help to take these forward.

Further on the algebra side, the Kitaev model generalises easily to replace G by a finite-dimensional semisimple Hopf algebra, with some aspects also in the nonsemisimple case[CM22]. The same applies easily enough to at least a quasi-bialgebra associated to an inclusion $L \subseteq H$ of finite-dimensional Hopf algebras[Sch02B] and to the corresponding module category picture. Ultimately here, it is the nonsemisimple case that is of interest as such Hopf algebras (e.g. of the form of reduced quantum groups $u_q(g)$) generate the categories where anyons as well as TQFT topological invariants live. It is also known that by promoting the finite group input of the Kitaev model to a more general semisimple weak Hopf algebra, one can obtain a unitary fusion category in the role of $\mathcal{C}$[Cha14]. There remains a lot of work, therefore, to properly connect these theories to computer science and in particular to established methods for quantum circuits. A step here could be braided ZX-calculus[Maj21], although precisely how remains to be developed. These are some directions for further work.

Bibliography

[Aar05] S. Aaronson, Quantum computing, postselection, and probabilistic polynomial-time, Proc. R. Soc. A. 461 (2005) 3473-–3482

[AG04] S. Aaronson and D. Gottesman, Improved Simulation of Stabilizer Circuits, Phys. Rev. A 70, 052328 (2004)

[AAA24] R. Acharya, L. Aghababaie-Beni, I. Aleiner et al., Quantum error correction below the surface code threshold, arXiv:2408.13687 [quant-ph]

[AM99] H. Albuquerque and S. Majid, Quasialgebra structure of the octonions, J. Algebra 220 (1999) 188–224

[AGS96] A. Alekseev, H. Grosse and V. Schomerus, Combinatorial quantization of the Hamiltonian Chern–Simons theory II. Commun. Math. Phys. 174 (1996) 561–604

[ADK15] V. Arvind, B. Das, J. Köbler et al. Colored Hypergraph Isomorphism is Fixed Parameter Tractable. Algorithmica 71, 120–138 (2015), https://doi.org/10.1007/s00453-013-9787-y

[AC19] B. Audoux and A. Couvreur, On tensor products of CSS Codes, Ann. Inst. Henri Poincaré Comb. Phys. Interact. 6 (2019), no. 2, pp. 239–287, https://doi.org/10.4171/aihpd/71

[Bac16] M. Backens, Completeness and the ZX-calculus, arXiv:1602.08954 [quant-ph]

[BK12] B. Balsam and A. Kirillov, Jr., Kitaev's lattice model and Turaev-Viro TQFTs, arXiv:1206.2308

[Beg03] E. Beggs, Making non-trivially associated tensor categories from left coset representatives, J. Pure. App. Alg. 177 (2003) 5–41

[BGM96] E. Beggs, J. Gould and S. Majid, Finite group factorisations and braiding J. Algebra 181 (1996) 112–151

[BM09] E. Beggs and S. Majid, Bar categories and star operations, Alg. and Repn. Theory 12 (2009) 103–152

[BM20] E. J. Beggs and S. Majid, *Quantum Riemannian Geometry*, Springer International Publishing, 1 Feb 2020, https://doi.org/10.1007/978-3-030-30294-8

[BSW11] S. Beigi and P. Shor and D. Whalen, The quantum double model with boundary: condensations and symmetries, Comm. Math. Phys. 306 (2011) 663–694

[BKS21] M. E. Beverland, A. Kubica, and K. M. Svore, Cost of Universality: A Comparative Study of the Overhead of State Distillation and Code Switching with Color Codes, PRX Quantum 2, 020341 (2021), doi:10.1103/PRXQuantum.2.020341

[Bom15] H. Bombin, Gauge Color Codes: Optimal Transversal Gates and Gauge Fixing in Topological Stabilizer Codes, New J. Phys. 17 (2015) 083002, doi:10.1088/1367-2630/17/8/083002

[Bom10] H. Bombin, Topological Order with a Twist: Ising Anyons from an Abelian Model, Phys. Rev. Lett. 105 (2010)

[BDMNPR21] H. Bombin, C. Dawson, R. V. Mishmash, N. Nickerson, F. Pastawski, S. Roberts, Logical blocks for fault-tolerant topological quantum computation, arXiv:2112.12160 [quant-ph]

[BM-D08] H. Bombin and M. A. Martin-Delgado, Family of non-Abelian Kitaev models on a lattice: Topological condensation and confinement, Phys. Rev. B 78 (2008) 115421

[BM-D07A] H. Bombin and M. A. Martin-Delgado, Homological error correction: Classical and quantum codes, Journal of Mathematical Physics, vol. 48, no. 5, p. 052105 (2007), https://doi.org/10.1063/1.2731356

[BM-D07B] H. Bombin and M. A. Martin-Delgado, Optimal resources for topological two-dimensional stabilizer codes: Comparative study, Physical Review A 76, (2007), doi:10.1103/PhysRevA.76.012305

[BM-D06] H. Bombin and M. A. Martin-Delgado, Topological Quantum Distillation, Phys. Rev. Lett. 97, 180501 (2006), https://doi.org/10.1103/PhysRevLett.97.180501

[BCGMRY24] S. Bravyi, A. W. Cross, J. M. Gambetta, D. Maslov, P. Rall, T. J. Yoder, High-threshold and low-overhead fault-tolerant quantum memory, Nature 627, 778-782 (2024), https://doi.org/10.1038/s41586-024-07107-7

[BH12] S. Bravyi and J. Haah, Magic-state distillation with low overhead, Physical Review A 86, (2012), https://doi.org/10.1103/PhysRevA.86.052329

[BKKK22] S. Bravyi, I. Kim, A. Kliesch and R. Koenig, Adaptive constant-depth circuits for manipulating non-abelian anyons, arXiv:2205.01933 [quant-ph]

[BK98] S. Bravyi and A. Kitaev, Quantum codes on a lattice with boundary, arXiv:quant-ph/9811052

[BK05] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, Phys. Rev. A 71 (2005), 022316, https://doi.org/10.1103/PhysRevA.71.022316

[BT09] S. Bravyi and B. Terhal, A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes, New Journal of Physics 11, 043029 (2009), doi:10.1088/1367-2630/11/4/043029

[BAC09] G. K. Brennen1, M. Aguado and J.I. Cirac, Simulations of quantum double models, New J. Phys. 11 053009 (2009)

[BB24] N. P. Breuckmann and S. Burton, Fold-Transversal Clifford Gates for Quantum Codes, Quantum 8, 1372 (2024), https://doi.org/10.22331/q-2024-06-13-1372

[BE21A] N. P. Breuckmann and J. N. Eberhardt, Balanced Product Quantum Codes, IEEE Transactions on Information Theory 2021, https://doi.org/10.1109/TIT.2021.3097347

[BE21B] N. P. Breuckmann and J. N. Eberhardt, Quantum Low-Density Parity-Check Codes, PRX Quantum 2 (4), 040101, 2021, https://doi.org/10.1103/PRXQuantum.2.040101

[BVCKT17] N. P. Breuckmann, C. Vuillot, E. Campbell, A. Krishna and B. M. Terhal, Hyperbolic and Semi-Hyperbolic Surface Codes for Quantum Storage, Quantum Science and Technology, Volume 2, Number 3, 2017, https://doi.org/10.1088/2058-9565/aa7d3b

[BKLW17] B. J. Brown, K. Laubscher, M. S. Kesselring and J. R. Wootton, Poking Holes and Cutting Corners to Achieve Clifford Gates with the Surface Code, Phys. Rev. X 7, 021029 (2017), https://doi.org/10.1103/PhysRevX.7.021029

[BMCA13] O. Buerschaper, J.M. Mombelli, M. Christandl and M. Aguado, A hierarchy of topological tensor network states, J. Math. Phys. 54 (2013) 012201

[BD19] A. Bullivant and C. Delcamp, Tube algebras, excitations statistics and compactification in gauge models of topological phases, Journal of High Energy Physics 10 (2019) 1–77

[CS96] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A 54, 1098 (1996), https://doi.org/10.1103/PhysRevA.54.1098

[Cam19] E. T. Campbell, A theory of single-shot error correction for adversarial noise, Quantum Science and Technology 4, 025006 (2019), https://doi.org/10.1088/2058-9565/aafc8f

[Cha14] L. Chang, Kitaev models based on unitary quantum groupoids, J. Math. Phys. 55 (2014) 041703

[CD11] B. Coecke and R. Duncan, Interacting quantum observables: categorical algebra and diagrammatics, New J. Phys. 13 (2011)

[CK17] B. Coecke and A. Kissinger, *Picturing Quantum Processes*, Cambridge University Press (2017)

[CPV12] B. Coecke, D. Pavlovic and J. Vicary, A new description of orthogonal bases, Mathematical Structures in Computer Science (2012)

[CKBB22] L. Z. Cohen, I. H. Kim, S. D. Bartlett and B. J. Brown, Low-overhead fault-tolerant quantum computing using long-range connectivity, Sci. Adv. 8, eabn1717 (2022), https://doi.org/10.1126/sciadv.abn1717

[CD19] J Collins and R. Duncan, Hopf-Frobenius algebras and a simpler Drinfeld double, in eds. B. Coecke and M. Leifer, *Quantum Physics and Logic* 2019, EPTCS 318 (2020)150–180

[CCW17] I. Cong, M. Cheng and Z. Wang, Hamiltonian and algebraic theories of gapped boundaries in topological phases of matter, Comm. Math. Phys. 355 (2), 645-689 (2017)

[CCW16] I. Cong, M. Cheng and Z. Wang, Topological Quantum Computation with Gapped Boundaries, arXiv:1609.02037 [quant-ph]

[CFSV04] L. P. Cordella, P. Foggia, C. Sansone and M. Vento, A (sub)graph isomorphism algorithm for matching large graphs, in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 10, pp. 1367-1372, Oct. 2004, doi:10.1109/TPAMI.2004.75.

[Cow22] A. Cowtan, Qudit lattice surgery, Accepted to QPL 2022, arXiv:2204.13228 [quant-ph], https://doi.org/10.48550/arXiv.2204.13228

[Cow24] A. Cowtan, SSIP: automated surgery with quantum LDPC codes, arXiv:2407.09423 [quant-ph]

[CB24] A. Cowtan and S. Burton, CSS code surgery as a universal construction, Quantum 8, 1344 (2024), https://doi.org/10.22331/q-2024-05-14-1344

[CM23] A. Cowtan and S. Majid, Algebraic aspects of boundaries in the Kitaev quantum double model, J. Math. Phys. 64, 102203 (2023), https://doi.org/10.1063/5.0127285

[CM22] A. Cowtan and S. Majid, Quantum double aspects of surface code models, J. Math. Phys. 63, 042202 (2022)

[CHRY24] A. Cross, Z. He, P. Rall and T. Yoder, Linear-Size Ancilla Systems for Logical Measurements in QLDPC Codes, arXiv:2407.18393 [quant-ph]

[CDH20] S. Cui, D. Ding, X. Han, G. Penington, D. Ranard, B Rayhaun and Z. Shangnan, Kitaev's quantum double model as an error correcting code, Quantum 4 (2020) 331–356

[CHW15] S. Cui, S. Hong and Z. Wang, Universal quantum computation with weakly integral anyons, Quantum Inf Process 14 (2015) 2687-–2727

[dBH20] N. de Beaudrap and D. Horsman, The ZX calculus is a language for surface code lattice surgery, Quantum 4, 218 (2020), https://doi.org/10.22331/q-2020-01-09-218 [dMB08] L. de Moura and N. Bjorner, Z3: an efficient SMT solver. In Proceedings of the Theory and practice of software, 14th international conference on Tools and algorithms for the construction and analysis of systems (TACAS'08/ETAPS'08), Springer-Verlag, Berlin, Heidelberg, 337–340 (2008), https://doi.org/10.1007/978-3-540-78800-3-24

[Del14] N. Delfosse, Decoding color codes by projection onto surface codes, Phys. Rev. A 89, 012317 (2014), https://doi.org/10.1103/PhysRevA.89.012317

[Dri87] V.G. Drinfeld, Quantum groups, in Proc. ICM Berkeley. AMS, 1987

[D-CP10] G. Duclos-Cianci and D. Poulin, A renormalization group decoding algorithm for topological quantum codes, Information Theory Workshop (ITW), 2010 IEEE, pp.1-5, Aug. 30 2010-Sept. 3 2010, https://doi.org/10.1109/CIG.2010.5592866

[DKP17] I. Dumer, A. A. Kovalev, and L. P. Pryadko, Distance verification for classical and quantum LDPC codes, IEEE Transactions on Information Theory, vol. 63, no. 7, pp. 4675-4686 (2017), doi: 10.1109/TIT.2017.2690381

[EK09] B. Eastin and E. Knill, Restrictions on Transversal Encoded Quantum Gate Sets, Phys. Rev. Lett. 102, 110502 (2009), doi:10.1103/PhysRevLett.102.110502

[ES24] J. N. Eberhardt and V. Steffan, Logical Operators and Fold-Transversal Gates of Bivariate Bicycle Codes, arXiv:2407.03973 [quant-ph], https://doi.org/10.48550/arXiv.2407.03973

[EEK82] A. L. Edmonds and J. H. Ewing and R. S. Kulkarni, Regular Tessellations of Surfaces and (p, q, 2)-Triangle Groups, Annals of Mathematics 116 (1982) 113–132

[EGNO10] P. Etingof, S. Gelaki, D. Nikshych and V. Ostrik, Tensor Categories, Mathematical Surveys and Monographs 205 (2010)

[Far03] D. S. Farley, Finiteness and CAT(0) properties of diagram groups, Topology, Vol. 42, Issue 5 (2003) pp. 1065-1082, https://doi.org/10.1016/S0040-9383(02)00029-0

[FMMC12] A. G. Fowler, M. Mariantoni, J. M. Martinis and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A 86 (2012), https://doi.org/10.1103/PhysRevA.86.032324

[FSG09] A. G. Fowler, A. M. Stephens and Peter Groszkowski, High-threshold universal quantum computation on the surface code, Phys. Rev. A 80 (2009) 052312

[FM01] M. H. Freedman and D. A. Meyer, Projective plane and planar quantum codes, Foundations of Computational Mathematics 1, 325 (2001), https://doi.org/10.1007/s102080010013

[FS03] J. Fuchs and C. Schweigert, Category theory for conformal boundary conditions, Fields Institute Communications 39 (2003) 25–71

[GF09] C. Gidney and A. G. Fowler, Efficient magic state factories with a catalyzed $|CCZ\rangle$ to $2|T\rangle$ transformation, Quantum 3, 135 (2019)

[GSJ24] C. Gidney, N. Shutty and C. Jones, Magic state cultivation: growing T states as cheap as CNOT gates, https://arxiv.org/abs/2409.17595

[Gog22] S. Gogioso, private communication (2022)

[Got24] D. Gottesman, Surviving as a Quantum Computer in a Classical World, https://www.cs.umd.edu/class/spring2024/cmsc858G/QECCbook-2024-ch1-15.pdf (2024), accessed 27/09/2025

[Haa16] J. Haah, Algebraic Methods for Quantum Codes on Lattices, Revista Colombiana de Matemáticas, 50(2), 299-349 (2016), https://doi.org/10.15446/recolma.v50n2.62214

[HSS08] A. Hagberg, P. J. Swart and D. A. Schult, Exploring network structure, dynamics, and function using NetworkX, United States: N. p., 2008

[HS97] G. Hahn and G. Sabidussi, Graph symmetry: algebraic methods and applications, NATO Advanced Science Institutes Series, vol. 497, Springer, p. 116 (1997) ISBN 978-0-7923-4668-5, https://doi.org/10.1007/978-94-015-8937-6

[HHTBP92] F. D. M. Haldane, Z. N. C. Ha, J. C. Talstra, D. Bernard and V. Pasquier, Yangian symmetry of integrable quantum chains with long-range interactions and a new description of states in conformal field theory, Physical Review Letters, 69 (1992)

[HV19] C. Heunen and J. Vicary, Categories for Quantum Theory: An Introduction, Oxford University Press (2019) DOI:10.1093/oso/9780198739623.001.0001

[Hig22] O. Higgott, Pymatching: A python package for decoding quantum codes with minimum-weight perfect matching, ACM Transactions on Quantum Computing, Vol. 3, Issue 3 (2022), https://doi.org/10.1145/3505637

[HWH21] O. Higgott, M. Wilson, J. Hefford, J. Dborin, F. Hanif, S. Burton and D. E. Browne, Optimal local unitary encoding circuits for the surface code, Quantum 5, 517 (2021), https://doi.org/10.22331/q-2021-08-05-517

[HFDM12] D. Horsman, A. G. Fowler, S. Devitt and R. Van Meter, Surface code quantum computing by lattice surgery, New J. Phys. 14 (2012) 123011, https://doi.org/10.1088/1367-2630/14/12/123011

[HJY23] S. Huang, T. Jochym-O'Connor and T. J. Yoder, Homomorphic Logical Measurements, PRX Quantum 4, 030301 (2023), https://doi.org/10.1103/PRXQuantum.4.030301

[IGND24] B. Ide, M. G. Gowda, P. J. Nadkarni and G. Dauphinais, Fault-tolerant logical measurements via homological measurement, arXiv:2410.02753 [quant-ph]

[JKT22] Z. Jia, D. Kaszlikowski and S. Tan, Boundary and domain wall theories of 2d generalized quantum double model, arXiv:2207.03970 [quant-ph]

[JS91] A. Joyal and R. Street, The geometry of tensor calculus, I, Advances in Mathematics, Volume 88, Issue 1, July 1991, https://doi.org/10.1016/0001-8708(91)90003-P

[KS11] A. Kapustin and N. Saulina, Topological boundary conditions in abelian Chern-Simons theory, Nucl.Phys.B845:393-435 (2011)

[Kis22] A. Kissinger, Phase-free ZX diagrams are CSS codes (...or how to graphically grok the surface code), In Proceedings QPL 2022, arXiv:2204.14038 [quant-ph], https://doi.org/10.48550/arXiv.2204.14038

[KM-v20] A. Kissinger, A. Meijer-van de Griend, CNOT circuit extraction for topologically-constrained quantum memories, Quantum Information and Computation, 20, 7& 8, (2020), https://doi.org/10.26421/QIC20.7-8

[Kit03] A. Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. 303 (2003) 3–20, https://doi.org/10.1016/S0003-4916

[KK12] A. Kitaev and L. Kong, Models for Gapped Boundaries and Domain Walls, Commun. Math. Phys. 313 (2012) 351-–373

[KM10A] J. Klim and S. Majid, Bicrossproduct Hopf quasigroups, Comment. Math. Univ. Carolin. 51 (2010) 287—304

[KM10B] J. Klim and S. Majid, Hopf quasigroups and the algebraic 7-sphere, Journal of Algebra 323 (2010), doi:10.1016/j.jalgebra.2010.03.011

[KL00] E. Knill and R. Laflamme, A theory of quantum error correcting codes, Phys. Rev. Lett. 84 (2000) 2525–2528

[KLZ96] E. Knill, R. Laflamme and W. Zurek, Threshold accuracy for quantum computation, arXiv preprint quant-ph/9610011, doi:10.48550/arXiv.quant-ph/9610011

[Koc03] J. Kock, Frobenius Algebras and 2-D Topological Quantum Field Theories (London Mathematical Society Student Texts), Cambridge: Cambridge University Press (2003) doi:10.1017/CBO9780511615443

[Kon14] L. Kong, Anyon condensation and tensor categories, Nuclear Physics B 886 (2014) 436–482

[KZ22] L. Kong, Z. Zhang, An invitation to topological orders and category theory, arXiv:2205.05565 [cond-mat.str-el]

[KP13] A. A. Kovalev and L. P. Pryadko, Quantum Kronecker sum-product low-density parity-check codes with finite rate, Phys. Rev. A 88, 012311 (2013), https://doi.org/10.1103/PhysRevA.88.012311

[KLP05] D. Kribs, R. Laflamme, and D. Poulin, Unified and Generalized Approach to Quantum Error Correction, Physical Review Letters 94, (2005), https://doi.org/10.1103/PhysRevLett.94.180501

[KP21] A. Krishna and D. Poulin, Fault-tolerant gates on hypergraph product codes, Phys. Rev. X 11, 011023 (2021), https://doi.org/10.1103/PhysRevX.11.011023

[Kup91] G. Kuperberg, Involutory hopf algebras and 3-manifold invariants, Int. J. Math. 2 (1991), 41–66

[LAR11] A. J. Landahl, J. T. Anderson, and P. R. Rice, Fault-tolerant quantum computing with color codes, arXiv:1108.5738 [quant-ph], https://doi.org/10.48550/arXiv.1108.5738

[LR-A14] A. J. Landahl and C. Ryan-Anderson, Quantum computing by color-code lattice surgery, arXiv:1407.5103 [quant-ph], https://doi.org/10.48550/arXiv.1407.5103

[Lau05] A. D. Lauda, Frobenius algebras and planar open string topological field theories, arXiv:math/0508349 [math.QA]

[Lei14] T. Leinster, Basic Category Theory, Cambridge Studies in Advanced Mathematics, Vol. 143, Cambridge University Press, 2014, https://doi.org/10.1017/CBO9781107360068

[LTZ15] A. Leverrier, J. P. Tillich and G. Zémor, Quantum Expander Codes, 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, Berkeley, CA, USA, 2015, pp. 810-824, doi: 10.1109/FOCS.2015.55.

[LB13] D. Lidar and T. A. Brun, *Quantum Error Correction*, Cambridge University Press (2013)

[LP24] H. Lin and L. P. Pryadko, Quantum two-block group algebra codes, Phys. Rev. A 109, 022407 (2024), https://doi.org/10.1103/PhysRevA.109.022407

[Lit19] D. Litinski, A Game of Surface Codes: Large-Scale Quantum Computing with Lattice Surgery, Quantum 3, 128 (2019)

[LHG11] Xi-W. Luo, Y-J. Han, G-C. Guo, X. Zhou and Z-W. Zhou, Simulation of non-Abelian anyons using ribbon operators connected to a common base site, Phys. Rev. A 84 (2011) 052314

[Mac78] S. Mac Lane, *Categories for the working mathematician*, Graduate Texts in Mathematics vol. 5 (1978)

[MS83] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North Holland Mathematical Library: Volume 16 (1983)

[Maj02] S. Majid, *A Quantum Groups Primer*, L.M.S. Lect. Notes 292 (2002)

[Maj04] S. Majid, Classification of differentials on quantum doubles and finite noncommutative geometry, Lect. Notes Pure Appl. Maths 239 (2004) 167-188, Marcel Dekker

[Maj95] S. Majid, *Foundations of Quantum Group Theory*, Cambridge University Press, (1995); paperback ed. (2000)

[Maj90] S. Majid, Physics for algebraists: non-commutative and non-cocommutative Hopf algebras by a bicrossproduct construction, J. Algebra 130 (1990) 17–64

[Maj21] S. Majid, Quantum and braided ZX-calculus, arXiv: 2103.07264 (math.qa)

[Maj98] S. Majid, Quantum double of quasi-Hopf algebras, Lett. Math. Phys. 45 (1998) 1-9

[Maj91] S. Majid, Representations, duals and quantum doubles of monoidal categories, Proceedings of the Winter School ”Geometry and Physics” (1991) 197–206

[Maj92] S. Majid, Tannaka-Krein Theorem for quasiHopf algebras and other results, Contemp. Math. 134 (1992) 219–232

[MR21] S. Majid and K. Rietsch, Planar spider theorem and asymmetric Frobenius algebras, arXiv:2109.12106 [math.QA]

[Mat] Math stackexchange, https://math.stackexchange.com/questions/1046209/pullbacks-and-pushouts-in-the-category-of-graphs, accessed 25/10/2022

[Meu17] C. Meusburger, Kitaev lattice models as a Hopf algebra gauge theory, Commun. Math. Phys. 353 (2017) 413–468, https://doi.org/10.1007/s00220-017-2860-7

[Mic14] K. P. Michnicki, 3D Topological Quantum Memory with a Power-Law Energy Barrier, Phys. Rev. Lett. 113, 130501 (2014), https://doi.org/10.1103/PhysRevLett.113.130501

[Moc03] C. Mochon, Anyon computers with smaller groups, Phys. Rev. A 69 (2004) 032306

[Moc04] C. Mochon, Anyons from nonsolvable finite groups are sufficient for universal quantum computation, Phys. Rev. A 67(2003) 022315

[Mon16] A. Montanaro, Quantum algorithms: an overview, npj Quantum Inf 2, 15023 (2016), https://doi.org/10.1038/npjqi.2015.23

[NFB17] H. P. Nautrup, N. Friis and H. J. Briegel, Fault-tolerant interface between quantum memories and quantum processors, Nat. Commun. 8, 1321 (2017), https://doi.org/10.1038/s41467-017-01418-2

[Nat05] S. Natale, Frobenius–Schur indicators for a class of fusion categories, Pacific J. Math. 221 (2005) 353–377,doi:10.2140/pjm.2005.221.353

[NC10] M. Nielsen and I. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition (2010), doi:10.1017/CBO9780511976667

[Nov24] S. Novák, Homological Quantum Error Correction with Torsion, arXiv:2405.03559 [quant-ph]

[ORM24] J. Old, M. Rispler and M. Müller, Lift-Connected Surface Codes, arXiv:2401.02911 [quant-ph], https://doi.org/10.48550/arXiv.2401.02911

[Ost03A] V. Ostrik, Module categories over the Drinfeld double of a finite group, International Mathematics Research Notices 27 (2003) 1507—1520, doi:10.1155/S1073792803205079

[Ost03B] V. Ostrik, Module categories, weak Hopf algebras and modular invariants, Transform. Groups 8 (2003) 177—206

[PK22A] P. Panteleev and G. Kalachev, Asymptotically Good Quantum and Locally Testable Classical LDPC Codes, STOC 2022: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, https://doi.org/10.1145/3519935.3520017

[PK21] P. Panteleev and G. Kalachev, Degenerate Quantum LDPC Codes With Good Finite Length Performance, Quantum 5, 585 (2021), https://doi.org/10.22331/q-2021-11-22-585

[PK22B] P. Panteleev and G. Kalachev, Quantum LDPC Codes With Almost Linear Minimum Distance, in IEEE Transactions on Information Theory, vol. 68, no. 1, pp. 213-229, Jan. 2022, https://doi.org/10.1109/TIT.2021.3119384

[Par71] B. Pareigis, When Hopf algebras are Frobenius algebras, J. Alg., Volume 18, Issue 4 (1971) 588-596

[Pen63] R. Penrose, Asymptotic Properties of Fields and Space-Times, Phys. Rev. Lett. 10, 66 (1963), https://doi.org/10.1103/PhysRevLett.10.66

[Per23] M. A. Perlin, qLDPC, https://github.com/Infleqtion/qLDPC, 2023, accessed 06/06/2024

[PW27] F. Peter and H. Weyl, Die Vollstandigkeit der primitiven Darstellungen einer geschlossenen kontinuierlichen Gruppe, Math. Ann., 97: 737–755 (1927), doi:10.1007/BF01447892

[PST06] J. Plefka, F. Spill, A. Torrielli, Hopf algebra structure of the AdS/CFT S-matrix, Phys. Rev. D74 (2006) 066008, https://doi.org/10.1103/PhysRevD.74.066008

[PSW24] B. Poor, R. A. Shaikh and Q. Wang, ZX-calculus is Complete for Finite-Dimensional Hilbert Spaces, arXiv:2405.10896 [quant-ph]

[PSK22] L P. Pryadko, V. A. Shabashov, V. K. Kozin, QDistRnd: A GAP package for computing the distance of quantum error-correcting codes, Journal of Open Source Software, 7(71) (2022) 4120, https://doi.org/10.21105/joss.04120

[QWV23] A. O. Quintavalle, P. Webster and M. Vasmer, Partitioning qubits in hypergraph product codes to implement logical gates, Quantum 7, 1153 (2023), https://doi.org/10.22331/q-2023-10-24-1153

[Reu19] D. Reutter, Higher linear algebra in topology and quantum information theory, Oxford DPhil thesis (2019)

[Rof22] J. Roffe, LDPC: Python tools for low density parity check codes, https://pypi.org/project/ldpc/, 2022, accessed 06/06/2024

[RWBC20] J. Roffe, D. R. White, S. Burton, and E. Campbell, Decoding across the quantum low-density parity-check code landscape, Phys. Rev. Research 2, 043423 (2020), https://doi.org/10.1103/PhysRevResearch.2.043423

[R-ABB24] C. Ryan-Anderson, N. C. Brown, C. H. Baldwin et al., High-fidelity and Fault-tolerant Teleportation of a Logical Qubit using Transversal Gates and Lattice Surgery on a Trapped-ion Quantum Computer, arXiv:2404.16728 [quant-ph]

[Sab] E. Sabo, A basic coding theory library for Julia, https://github.com/esabo/CodingTheory, accessed 06/06/2024

[Sch02A] P. Schauenburg, Hopf Algebra Extensions and Monoidal Categories, New Dir. Hopf Alg. 43 (2002)

[Sch16] P. Schauenburg, Computing Higher Frobenius-Schur Indicators in Fusion Categories Constructed from Inclusions of Finite Groups, Pacific J. Math. 280 (2016) 177–201

[Sch02B] P. Schauenburg, Hopf Bimodules, Coquasibialgebras, and an Exact Sequence of Kac, Advances in Mathematics 165 (2002) 194—263

[Sch01] P. Schauenburg, Turning Monoidal Categories into Strict Ones, New York Journal of Mathematics 7 (2001) 257-265

[Sch95] H.-J. Schneider, Lectures on Hopf algebras; Notes by Sonia Natale, Trabajos de Matematica 31/95, FaMAF, 1995.

[SHR24] T. R. Scruby, T. Hillmann and J. Roffe, High-threshold, low-overhead and single-shot decodable fault-tolerant quantum memory, arXiv:2406.14445 [quant-ph], https://doi.org/10.48550/arXiv.2406.14445

[Sel11] P. Selinger, A survey of graphical languages for monoidal categories, Springer Lecture Notes in Physics 813, pp. 289-355, 2011

[Sho95] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, Physical Review A. 52 (4): R2493–R2496, 1995, https://doi.org/10.1103/PhysRevA.52.R2493

[Ste96] A. Steane, Multiple-particle interference and quantum error correction, Proceedings of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences 452(1954), pp. 2551–2577 (1996), doi:10.1098/rspa.1996.0136.

[Tak81] M. Takeuchi, Matched pairs of groups and bismash products of Hopf algebras, Comm. Algebra 9 (1981) 841

[TNG24] D. B. Tan, M. Y. Niu and C. Gidney, A SAT Scalpel for Lattice Surgery: Representation and Synthesis of Subroutines for Surface-Code Fault-Tolerant Quantum Computing, arXiv:2404.18369 [quant-ph], https://doi.org/10.48550/arXiv.2404.18369

[TZ14] J.P. Tillich and G. Zémor, Quantum LDPC Codes With Positive Rate and Minimum Distance Proportional to the Square Root of the Blocklength, in IEEE Transactions on Information Theory, vol. 60, no. 2, pp. 1193-1202, Feb. 2014, doi: 10.1109/TIT.2013.2292061.

[TTWL08] S. Trebst, M. Troyer, Z. Wang and A. Ludwig, A short introduction to Fibonacci anyon models, Prog. Theor. Phys. Suppl. 176 (2008) 384–407

[Wet12] J. van de Wetering, ZX-calculus for the working quantum computer scientist, arXiv:2012.13966 [quant-ph], https://doi.org/10.48550/arXiv.2012.13966

[VLC19] C. Vuillot, L. Lao, B. Criger, C. G. Almudéver, K. Bertels and B. M. Terhal, Code deformation and lattice surgery are gauge fixing, New J. Phys. 21 033028 (2019), https://doi.org/10.1088/1367-2630/ab0199

[Wan21] Q. Wang, Qufinite ZX-calculus: a unified framework of qudit ZX-calculi, arXiv:2104.06429 [quant-ph]

[Wat24] G. Watkins et al., A High Performance Compiler for Very Large Scale Surface Code Computations, Quantum 8, 1354 (2024), https://doi.org/10.22331/q-2024-05-22-1354

[Wei84] C. A. Weibel, An Introduction to Homological Algebra (Cambridge Studies in Advanced Mathematics), Cambridge University Press (1994), https://doi.org/10.1017/CBO9781139644136

[WY24] D. J. Williamson and T. J. Yoder, Low-overhead fault-tolerant quantum computation by gauging logical operators, arXiv:2410.02213 [quant-ph]

[WB24] S. Wolanski and B. Barber, Ambiguity Clustering: an accurate and efficient decoder for qLDPC codes, arXiv:2406.14527 [quant-ph], https://doi.org/10.48550/arXiv.2406.14527

[WLP09] J. R. Wootton, V. Lahtinen and J. K. Pachos, Universal quantum computation with a non-Abelian topological memory, in *Theory of Quantum Computation, Communication, and Cryptography* (2009) 56–65

[WD98] Chuan-Kun Wu and Ed Dawson, Existence of generalized inverse of linear transformations over finite fields, Finite Fields and Their Applications 4 (1998) 307–315, https://doi.org/10.1006/ffta.1998.0215

[YCC22] B. Yan, P. Chen, S. X. Cui, Ribbon operators in the generalized Kitaev quantum double model based on Hopf algebras, Journal of Physics A: Mathematical and Theoretical 55 (2022) 185201

[ZL24] G. Zhang and Y. Li, Time-efficient logical operations on quantum LDPC codes, arXiv:2408.01339 [quant-ph]

[ZSP23] G. Zhu, S. Sikander, E. Portnoy, A. W. Cross and B. J. Brown, Non-Clifford and parallelizable fault-tolerant logical gates on constant and almost-constant rate homological quantum LDPC codes via higher symmetries, arXiv:2310.16982 [quant-ph], https://doi.org/10.48550/arXiv.2310.16982

Appendix

1 Graphs and cell complexes

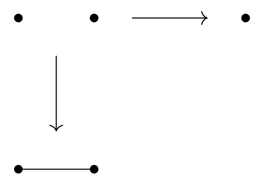
In this appendix we give some categorical background on abstract cell complexes. This is not necessary to define CSS code surgery, but codes obtained from cell complexes are an important motivating example, as they include surface codes, toric codes [Kit03], hyperbolic codes [BVCKT17] and the expander lifted product codes from [PK22A]. In general, if a CSS code comes from tessellating a manifold, it is likely to use cell complexes. Cell complexes are important in the study of topological spaces, and many of the constructions of CSS codes, such as balanced/lifted products, can also be phrased in the language of topology, but we stick to cell complexes for brevity. As a warm-up, we describe certain categories of graphs, and then move on to a specific kind of cell complex.

Let Γ be a finite simple undirected graph. Recall that as a simple graph, Γ has at most one edge between any two vertices and no self-loops on vertices. Γ can be defined as a pair of sets, $V(\Gamma)$ and $E(\Gamma)$, with $E(\Gamma) \subseteq 2^{V(\Gamma)}$, the powerset of vertices, where each $e \in E(\Gamma)$ has 2 elements i.e. it can be expressed as $e = \{v_1, v_2\}$. An example of a graph is $\mathcal{C}_n$, the cycle graph with n vertices and edges. We will also use $\mathcal{P}_n$, the path graph with n edges and $n + 1$ vertices.

Definition 1.1. Let `Grph` be the category of finite simple undirected graphs. A morphism $\Gamma \to \Delta$ in `Grph` is a function $f : V(\Gamma) \to V(\Delta)$ such that $\{v_1, v_2\} \in E(\Gamma) \implies \{f(v_1), f(v_2)\} \in E(\Delta)$, i.e. the function respects the incidence of edges.

`Grph` has several different products and other categorical features. We are particularly interested in colimits. `Grph` has a coproduct $\Gamma+\Delta$ being the disjoint union, with $V(\Gamma+\Delta) = V(\Gamma) \sqcup V(\Delta)$ and $E(\Gamma + \Delta) = E(\Gamma) \sqcup E(\Delta)$. It also has an initial object I given by the empty graph. However, `Grph` is not cocomplete, as it does not have all pushouts.

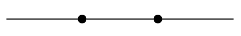
Example 1.2. As a counterexample [Mat], given the diagram



no cocone exists, as the graphs are not allowed self-loops. Therefore, no pushout exists.

One can easily see that there are diagrams for which pushouts do exist, though.

More than just graphs, we would like to allow for *open* graphs, i.e. graphs which may have edges which connect to only one vertex, but are not self-loops. For example,



We call $\mathcal{G}_3$ the 3rd *open path graph*, where the nth open path graph $\mathcal{G}_n$ has n edges in a line with $n - 1$ vertices between them. We now give a particular formalisation of open graphs.

Definition 1.3. Let Γ be a finite simple undirected graph with two disjoint vertex sets $V(\Gamma)$ and $B(\Gamma)$, where $E(\Gamma) \subseteq 2^{V(\Gamma) \cup B(\Gamma)}$. We then say that Γ is an open graph. We call $V(\Gamma)$ the internal vertices and $B(\Gamma)$ the boundary vertices.

So in the picture of $\mathcal{G}_3$ above there are vertices at either end of the open wires, but they are considered 'invisible', i.e. they belong to $B(\Gamma)$.

Definition 1.4. Let `OGrph` be the category of open graphs. A morphism $\Gamma \to \Delta$ in `OGrph` is a function $f : V(\Gamma) \cup B(\Gamma) \to V(\Delta) \cup B(\Delta)$ such that $\{v_1, v_2\} \in E(\Gamma) \implies \{f(v_1), f(v_2)\} \in E(\Delta)$ and $f(x) \in V(\Delta) \iff x \in V(\Gamma)$.

This restriction disallows internal vertices from being 'created' or 'deleted' by a graph morphism by converting them to boundary vertices. `OGrph` has very similar properties to `Grph`. Its initial object is the empty open graph. `OGrph` has a coproduct, where $V(\Gamma+\Delta) = V(\Gamma) \sqcup V(\Delta)$ and $B(\Gamma + \Delta) = B(\Gamma) \sqcup B(\Delta)$. Like `Grph`, `OGrph` is not cocomplete, as Example 1.2 also works in the setting of open graphs. It is obvious that `Grph` is a subcategory of `OGrph`.

We now move on to cell complexes, in particular abstract cubical complexes. These are abstract cell complexes which are 'square', unlike their 'triangular' relatives simplicial complexes.

Definition 1.5. The abstract d-cube is the set $\{0,1\}^d$, with the 0-cube $\{0,1\}^0 := \{0\}$. A face of the abstract d-cube is a product $A_1 \times \cdots \times A_d$, where each A_i is a nonempty subset of $\{0,1\}$.

Definition 1.6. [Far03] Let S be a finite set and let Ω be a collection of nonempty subsets of S such that:

- Ω covers S.
- For $X, Y \in \Omega$, $X \cap Y \in \Omega$ or $X \cap Y = \emptyset$.
- For each $X \in \Omega$, there is a bijection from X to the abstract d-cube for some choice of d, such that any $Y \subset X$ is in Ω iff it is mapped to a face of the d-cube.

Then Ω is an abstract cubical complex.

Abstract cubical complexes are combinatorial versions of cubical complexes, meaning they are stripped of their associated geometry. The elements in Ω are still called *faces*. We can consider Ω to be a graded poset, with subset inclusion as the partial order, and the grading $\dim(X) = \log_2 |X|$. We also call this grading the dimension d of X, and we call X a d-face. The set of d-faces in Ω is called Ω_d. There is a relation $\Omega_d \to \Omega_{d-1}$ taking a d-face to its $(d-1)$-face subsets.

We call the vertex set $V(\Omega) = S = \Omega_0$, and also define the dimension of a cubical complex

$$\dim(\Omega) = \max_{X \in \Omega} \dim(X)$$

The d-skeleton of Ω is the maximal subcomplex $\Upsilon \subseteq \Omega$ such that $\dim(\Upsilon) = d$. The 1-skeleton of an abstract cubical complex is a finite simple undirected graph. The 2-skeleton of an abstract cubical complex is 'like' a square lattice, in that it has 2-faces which each have 4 0-faces as subsets and 4 1-faces.

Definition 1.7. Let ACC be the category of abstract cubical complexes. A morphism $f : \Omega \to \Upsilon$ in ACC is a function $f : V(\Omega) \to V(\Upsilon)$, such that $\{x, \cdots, y\} \in \Omega_d \implies \{f(x), \cdots, f(y)\} \in \Upsilon_d$, i.e. incidence is preserved at each dimension.

Similar to Grph, ACC has coproduct given by $(\Omega + \Upsilon)_i = \Omega_i \sqcup \Upsilon_i$ and an initial object $I = \emptyset$, and does not generally have pushouts, where we can reuse the same counterexample as Grph. Another categorical property we highlight here is that ACC has a monoidal product called the *box product*.

Definition 1.8. Let $\Upsilon \,\square\, \Omega$ be the box product of abstract cubical complexes. Then

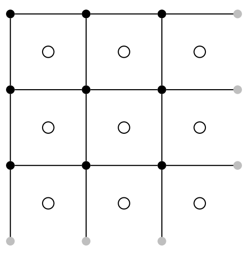
$$(\Upsilon \,\square\, \Omega)_n = \sum_{i+j=n} \Upsilon_i \times \Omega_j.$$

We now check that $\Upsilon \,\square\, \Omega$ is indeed an abstract cubical complex.

Proof. First, it has a vertex set $V(\Upsilon \,\square\, \Omega) = V(\Upsilon) \times V(\Omega)$, and thus trivially covers $\Upsilon_0 \times \Omega_0$. Second, let $X \times Y \in \Upsilon_i \times \Omega_j$ and $T \times U \in \Upsilon_k \times \Omega_l$. This has $(X \times Y) \cap (T \times U) = (X \cap T) \times (Y \cap U)$ which is either in $\Upsilon_m \times \Omega_n$ for some $m \leq i, m \leq k$ and $n \leq j, n \leq l$, and thus $(X \cap T) \times (Y \cap U) \in \Upsilon \,\square\, \Omega$, or $(X \cap T) \times (Y \cap U) = \emptyset$. Third, if X and Y each have a bijection to an i-cube and j-cube respectively, then $X \times Y$ has a bijection to an $(i + j)$-cube. Any $W \subset X \times Y$ can be expressed as $T \times U$, for $T \subset X$ and $U \subset Y$. Then W is in $\Omega \,\square\, \Upsilon$ iff T is mapped to a face of the i-cube and U to a face of the j-cube, thus W to a face of the $(i + j)$-cube. ■

Let us compile this into a more digestible form for the case when Υ and Ω are both graphs. Given vertices (u, u') and (v, v') in $V(\Upsilon \,\square\, \Omega)$, the 1-face $\{(u, u'), (v, v')\} \in (\Upsilon \,\square\, \Omega)_1$ iff $(u = v \ \&\ (u', v') \in \Omega)$ or $((u, v) \in \Upsilon \ \&\ u' = v')$. Then $(\Upsilon \,\square\, \Omega)_2 \cong E(\Upsilon) \times E(\Omega)$. The 1-skeleton of $\Upsilon \,\square\, \Omega$ is just the normal box product of graphs [HS97].

Example 1.9. Let $\mathcal{C}_m$ and $\mathcal{C}_n$ be cycle graphs with m and n vertices respectively, considered as abstract cubical complexes. Then $\mathcal{T} = \mathcal{C}_m \,\square\, \mathcal{C}_n$ admits an embedding as a square lattice on the torus, and has $\dim(\mathcal{C}_m \,\square\, \mathcal{C}_n) = 2$. Setting $m = n = 3$ we have



where the grey dots indicate periodic boundary conditions and the white circles specify 2-faces. This example comes up in the form of the toric code in Section 2.2.

Obviously, Grph is a subcategory of ACC.

We are also interested in *open* abstract cubical complexes.

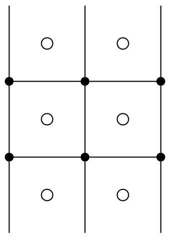
Definition 1.10. Let Υ be an open abstract cubical complex. Υ is an abstract cubical complex where Υ_0 is divided into two disjoint vertex sets $V(\Upsilon)$ and $B(\Upsilon)$.

The 1-skeleton of an open abstract cubical complex is an open graph.

Definition 1.11. Let OACC be the category of open abstract cubical complexes. A morphism $f : \Omega \to \Upsilon$ in OACC is a function $f : V(\Omega) \cup B(\Omega) \to V(\Upsilon) \cup B(\Upsilon)$ such that $f(x) \in V(\Upsilon) \iff x \in V(\Omega)$ and $\{x, \cdots, y\} \in \Omega_d \implies \{f(x), \cdots, f(y)\} \in \Upsilon_d$.

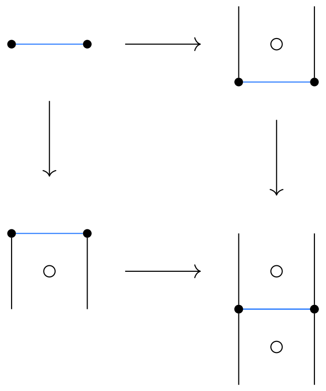
As in our previous examples, `OACC` has the obvious coproduct and initial object, and does not have pushouts in general.

Example 1.12. Let Υ be a 'patch', a square lattice with two rough and two smooth boundaries:



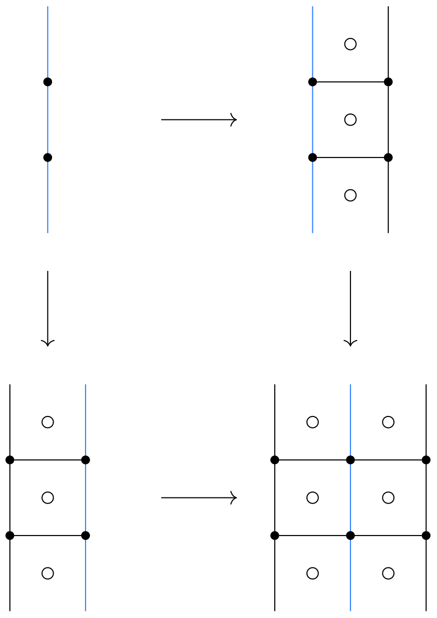
This patch has 6 2-faces, 13 1-faces and 6 0-faces.

Example 1.13. We can perform the pushout of two smaller open abstract cubical complexes to acquire a patch:



where the apex is $\mathcal{P}_1$, the blue edge indicates where the apex is mapped to, and the bottom right open abstract cubical complex is the object of the pushout.

Example 1.14. Let $\mathcal{G}_3$ be the open path graph, and let Ω be a patch. Then we have a pushout



This example comes up in the context of lattice surgery on surface codes. Evidently, both `OGrph` and `ACC` are subcategories of `OACC`, and one can define a box product for `OACC` in the same way as we did for `ACC` in Definition 1.8.

One can define quantum codes using abstract cell complexes more generally, but abstract cubical complexes are the specific type which we make use of in examples in Section 2.2 and onwards. We now relate the above cell complexes to chain complexes by way of functors.

Definition 1.15. Given an abstract cubical complex Ω we can define the incidence chain complex $C_\bullet$ in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, where each nonzero component has a basis $\tilde{C}_n = \Omega_n$, and each nonzero differential $\partial^{C_\bullet}_{n+1}$ takes an $n+1$-face to its n-dimensional subsets. The differential is thus a matrix with a 1 where an n-face is contained within an $(n+1)$-face, and 0 elsewhere. It is an elementary fact that every $(d-2)$-face in a d-face is the intersection of exactly 2 $(d-1)$-faces, thus $\partial^{C_\bullet}_{n-1} \circ \partial^{C_\bullet}_n = 0 \mod 2$. Clearly, the incidence chain complex of a dimension 1 abstract cubical complex is just the incidence matrix of a simple undirected graph.

We can do essentially the same thing given an open abstract cubical complex Υ. In this case, each nonzero component has a basis $\tilde{C}_n = \{X \in \Omega_n \mid X \not\subseteq B(\Omega)\}$, that is we ignore all faces which are made up only of boundary vertices, and differentials are the same matrices as above, with a 1 where an n-face which is not a subset of $B(\Omega)$ (and therefore would be 'invisible') is contained in an $(n+1)$-face. It is easy to see that we still have $\partial^{C_\bullet}_n \circ \partial^{C_\bullet}_{n+1} = 0 \mod 2$, as making vertices 'invisible' corresponds to deleting rows in $\partial^{C_\bullet}_1$, edges rows in $\partial^{C_\bullet}_2$ etc. The incidence chain complex of a dimension 1 open abstract cubical complex is the incidence matrix of an open graph.

Definition 1.16. Let $C_\bullet$ and $D_\bullet$ be the incidence chain complexes of two abstract cubical complexes Ω and Υ with a morphism $f : \Omega \to \Upsilon$, and set $\tilde{C}_0, \tilde{D}_0$ as $V(\Omega), V(\Upsilon)$ respectively. This induces a chain map $g_\bullet : C_\bullet \to D_\bullet$, with the matrix g_1 given by f, and all matrices on higher components generated inductively. Degrees $i < 1$ are assumed to be zero.

As a consequence, we can define a functor $\varphi : \mathtt{ACC} \to \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, sending each abstract cell complex to its free chain complex as described in Definition 1.15. One can check that $\varphi(f) \in \mathrm{Hom}(\varphi(\Omega), \varphi(\Upsilon))$ for any morphism $f : \Omega \to \Upsilon$ between abstract cubical complexes. φ is faithful but not full, as there exist morphisms, such as the zero morphism, which are not in the image of φ.

Definition 1.17. There is also a functor $\vartheta : \mathtt{OACC} \to \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$. On objects, this again follows Definition 1.15. On morphisms this is the same as φ except it must obviously ignore maps between boundary vertices everywhere. Thus ϑ is not faithful.

Example 1.18. Let Ω and Υ be two abstract cubical complexes. Then $\varphi(\Omega + \Upsilon) = \varphi(\Omega) \oplus \varphi(\Upsilon)$, which is easy to check. Similarly, $\varphi(\emptyset) = \mathbf{0}_\bullet$. The same is true of ϑ, except that $\vartheta(\Xi) = \mathbf{0}_\bullet$ for any Ξ with $V(\Xi) = \emptyset$.

Lemma 1.19. The functors φ and ϑ are cocontinuous i.e. they preserve colimits.

Proof. We give a proof sketch here. We know already that φ preserves coproducts so it is sufficient to check that it preserves pushouts. Let

$$\begin{tikzcd}
\Xi \arrow[r, "g"] \arrow[d, "f"'] & \Upsilon \arrow[d, "l"] \arrow[dl, phantom, "\ulcorner", very near end] \\
\Omega \arrow[r, "k"'] & \chi
\end{tikzcd}$$

be a pushout in $\mathtt{ACC}$. Then $\chi_0 = \Omega_0 \sqcup \Upsilon_0 / f \sim g$, and we have elements in χ_n of the form $([x], \cdots, [y])$, which can be seen as pushouts at each dimension. Also, $(x, \cdots, y) \in \Omega_n \implies$

$([x], \cdots, [y]) \in \chi_n$, and the same for Υ_n. Then $\tilde{\varphi(\chi)}_0 = \chi_0$. We then have basis elements of the form $([x], \cdots, [y]) \in \tilde{\varphi(\chi)}_n$, and differentials have their obvious form. If we take the diagram in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$:

$$\begin{tikzcd}
\varphi(\Xi) \arrow[r, "\varphi(g)"] \arrow[d, "\varphi(f)"'] & \varphi(\Upsilon) \\
\varphi(\Omega) &
\end{tikzcd}$$

Then we have $Q_\bullet$ as the pushout. Basis elements in Q_n are then also of the form $([x], \cdots, [y])$ for $[x], [y] \in \chi_n$. The differentials also match up correctly, and so $Q_\bullet = \varphi(\chi)$.

The same checks apply if we take $\vartheta : \mathtt{OACC} \to \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ instead. Observe that in this case f and g may have images only in $B(\Omega)$ and $B(\Upsilon)$, in which case Ξ must have empty $V(\Xi)$. Then the pushout in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ will just be a direct sum, i.e. the pushout with $\vartheta(\Xi) = \mathbf{0}_\bullet$ as the apex.

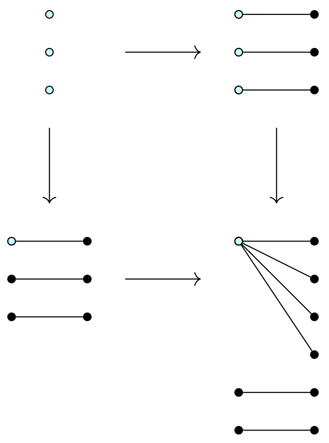
Recall that $\mathtt{ACC}$ and $\mathtt{OACC}$ do not themselves have all pushouts, and therefore all colimits, but φ and ϑ preserve those which they do have. ■

Definition 1.20. For any chain complex $C_\bullet$ we have also the pth translation $C[p]_\bullet$, where all indices are shifted down by p, i.e. $C[p]_n = C_{n+p}$ and $\partial_n^{C[p]_\bullet} = \partial_{n+p}^{C_\bullet}$. This extends to an invertible endofunctor $p : \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2}) \to \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ in the obvious way.

Lemma 1.21. Let Υ and Ω be two open abstract cubical complexes. Recalling the functor $\vartheta : \mathtt{OACC} \to \mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$ from Definition 1.16, we have $\vartheta(\Upsilon \,\square\, \Omega) = \vartheta(\Upsilon) \otimes \vartheta(\Omega)$, so ϑ is a monoidal functor.

2 Pushouts and properties of codes

Here we describe a few problems with using general pushouts to construct new quantum codes, even when the spans are basis-preserving. First, in a certain sense the pushout of LDPC codes is not necessarily LDPC. To illustrate this, consider the following pushout of graphs:



where the light dots indicate the graph morphisms. As ϑ is cocontinuous this pushout exists also in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$. There, it represents a merge of two binary classical codes, although we can consider a binary linear code to just be a CSS code without any Z measurements. As a consequence, we have two initial codes with P_X having maximal weights 1 each, and the merged code has maximal weight 4. Evidently, one can scale this with the size of the input graphs: here, the input graphs each have 3 edges, but if there are graphs with m

edges each (and weight 1) and the apex with m vertices (and weight 0) then the pushout graph will have maximal weight $m+1$. As a consequence the family of pushout graphs as m scales is not bounded above by a constant, and so the corresponding family of codes is not LDPC.

Conjecture 2.1. Let

$$\begin{tikzcd} A_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & D_\bullet \\ C_\bullet & \end{tikzcd}$$

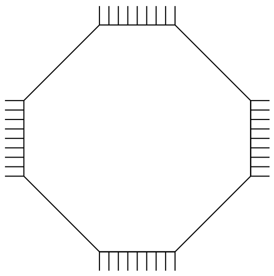
be a basis-preserving monic span in $\mathtt{Ch}(\mathtt{Mat}_{\mathbb{F}_2})$, and let $Q_\bullet$ be the pushout chain complex of this monic span. Further, let the monic span be a representative of a family of monic spans which are parameterised by some $n \in \mathbb{N}$, and let $A_\bullet$, $C_\bullet$ and $D_\bullet$ be the Z-type complexes of quantum LDPC codes. Then $(Q_\bullet, Q^\bullet)$ is also LDPC.

Formulating this conjecture properly requires specifying what it means for a monic span to be parameterised.

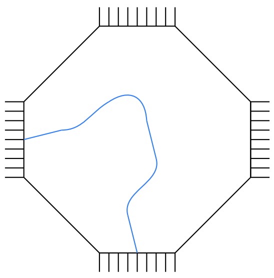
Lastly, taking pushouts evidently preserves neither homologies nor code distances, as easy examples with lattice surgery demonstrate. Moreover, we do not know of a way of giving bounds on these quantities for general pushouts, although again we suspect it should be easier for monic spans.

3 Octagonal surface code patch

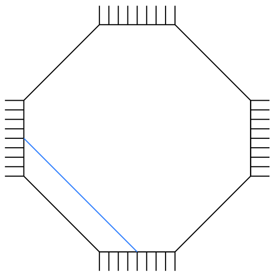
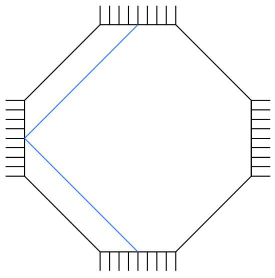
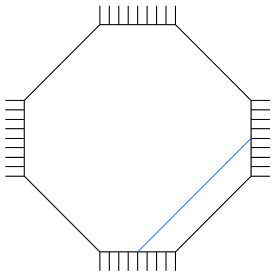
Consider the following patch of surface code:



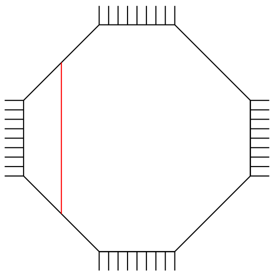
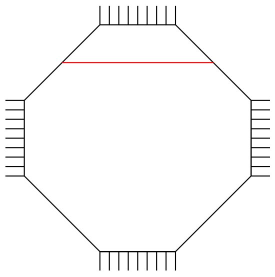
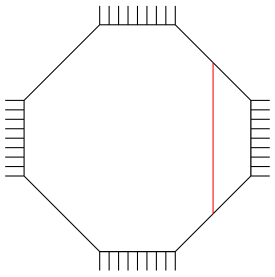
where the bristled edges are rough boundaries, and the diagonal edges are smooth boundaries. We have abstracted away from the actual cell complex as the tessellation is not important. Z-type logical operators take the form of strings extending from one rough boundary to another, e.g.



Two strings belong to the same equivalence class iff they are isotopic on the surface, allowing for the endpoints to slide up and down a rough boundary. There are exactly 3 nontrivial such classes out of which all other strings can be composed. As a consequence, this patch of surface code has logical space V with $\dim V = 2^3 = 8$. [1] We can choose a basis for this logical space, which has logical $\overline{Z}$ operators with representatives:

$\overline{Z}_1 =$  $\overline{Z}_2 =$  $\overline{Z}_3 =$ 

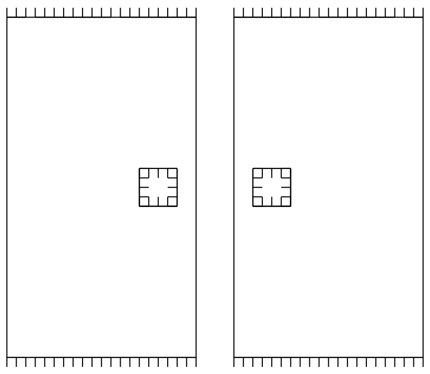
where the middle operator can be smoothly deformed to a vertical line from top to bottom if desired. Recall that on the surface code an $\overline{X}$ operator anticommutes with a $\overline{Z}$ operator iff the strings cross an odd number of times. Thus, given the basis above, the duality pairing of Lemma 2.2.4 forces a similar basis of $\overline{X}$ operators, with representatives:

$\overline{X}_1 =$  $\overline{X}_2 =$  $\overline{X}_3 =$ 

We see that $\overline{Z}_1$ is contained entirely within $\overline{Z}_2$ on physical qubits. Thus it is possible to construct a $\overline{Z}$ merge which is not irreducible, in the parlance of Definition 2.3.13. If we choose a different representative, by deforming $\overline{Z}_2$ to be a vertical line, then we can also perform a irreducible $\overline{Z}$ merge.

4 A $\overline{Z}$-merge map which is not distance preserving

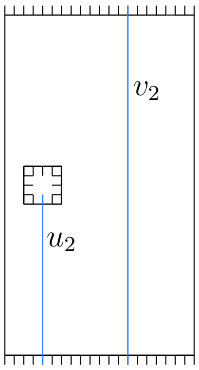
Here we provide an illustrative example to show that it is possible to create $\overline{Z}$ operators in a $\overline{Z}$-merged code which are of lower weight than any logical operator in the initial code. Consider the following surface code patches:



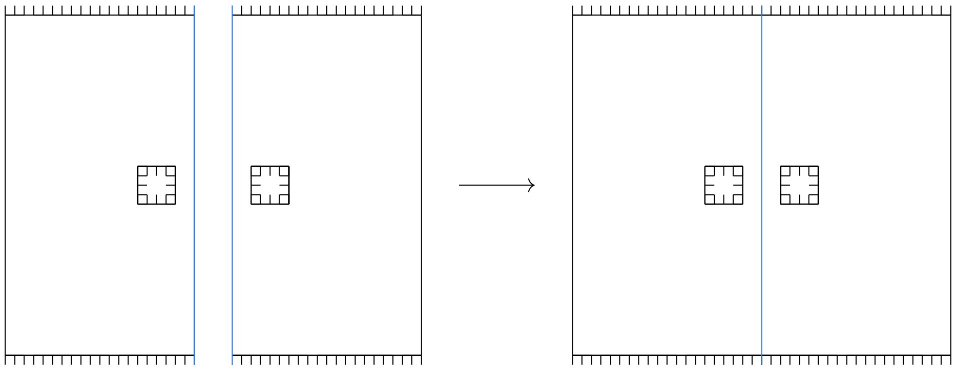
where, as in the previous appendix, bristled edges represent rough boundaries and non-bristled edges represent smooth boundaries. There is a hole in each patch, with bristled

[1]More generally, a patch with $2m$ edges, alternating rough and smooth, has $\dim V = 2^{m-1}$, i.e. the number of edges in a minimal spanning tree on the complete graph with m vertices.

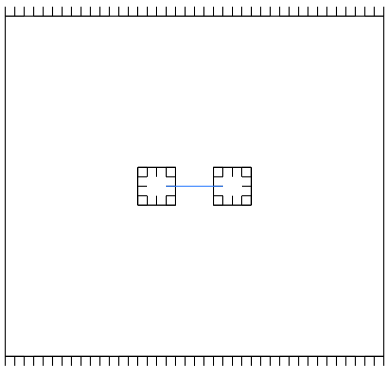
edges around it. As a consequence, each patch has 2 logical qubits. We can assign $\overline{Z}$ logical operators u_2 and v_2, representatives of each equivalence class $[u_2]$ and $[v_2]$ from which all other classes can be composed, like so:



and the same for $[u_1]$, $[v_1]$ on the other patch. We quotient out a $\overline{Z}$ operator in $[v_1]$ and $[v_2]$ going along the right and left boundaries, like so:



leaving a $\overline{Z}$-merged code. This has new $\overline{Z}$ operators, which belong to the equivalence class $[u_1 + u_2]$. These operators are of the form:



to see that these do belong to $[u_1 + u_2]$, label this operator t and see that $t + u_1 + u_2$ is in $[0]$, as it forms a contractible loop. Then $[t] = [-u_1 - u_2] = [u_1 + u_2]$, recalling that we are working over $\mathbb{F}_2$. This new operator t has a weight lower than any of those in the original codes, which one can see from the diagrams.

5 A merged code with larger logical space

Take the lift-connected surface (LCS) codes from [ORM24] with $\ell = 1$, $L = 3$. This is a $[\![15, 3, 3]\!]$ code $C_\bullet$, with the parity-check matrices:

$$P_Z = \begin{pmatrix} 1&0&0&0&0&0&1&1&0&0&0&0&1&0&0\\ 0&1&0&0&0&0&0&1&1&0&0&0&0&1&0\\ 0&0&1&0&0&0&1&0&1&0&0&0&0&0&1\\ 0&0&0&1&0&0&0&0&0&1&1&0&1&0&1\\ 0&0&0&0&1&0&0&0&0&0&1&1&1&1&0\\ 0&0&0&0&0&1&0&0&0&1&0&1&0&1&1 \end{pmatrix}$$

$$P_X = \begin{pmatrix} 1&0&0&1&1&0&0&0&0&0&0&0&1&0&0\\ 0&1&0&0&1&1&0&0&0&0&0&0&0&1&0\\ 0&0&1&1&0&1&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&1&0&0&1&1&0&1&0&1\\ 0&0&0&0&0&0&0&1&0&0&1&1&1&1&0\\ 0&0&0&0&0&0&0&0&1&1&0&1&0&1&1 \end{pmatrix}$$

This code has an irreducible $\overline{Z}$-logical with support on qubits $(2, 9, 14)$, starting from 0. If we merge two copies of $C_\bullet$ along this logical we obtain a $[\![27, 6, 2]\!]$ code, when naively one would expect a code with 5 logical qubits. The additional logical has appeared because, while quotienting the logicals together in the two codes, we have inadvertently increased the size of $\ker(P_X)$, increasing the size of the logical space.

One explanation for why this can occur is because of the complication of bases for our chain complexes. In Definition 2.3.10, we set $\tilde{V_0} = \bigcup_{u \in \mathrm{im}(\partial_1^{V_\bullet})} \mathrm{supp}\ u$, so we incorporated all basis elements with support in the image of the logical operator; if we were to instead set $V_0 = \mathrm{im}(\partial_1^{V_\bullet})$ we believe that this occurrence would be impossible, as the only quotient would be on precisely those vectors in the image of the logical operator, not those vectors' basis elements. However, we cannot do this in general while keeping all chain maps basis-preserving.

6 Irreducibility is gauge-fixability

We will prove in this section that for all CSS codes having irreducible and gauge-fixable operators are equivalent properties.

Lemma 6.1. A vector $u \in \ker(P_X) \backslash \mathrm{im}(P_Z^\intercal)$ is gauge-fixable iff for every pair (e_i, e_j) of basis vectors in supp (u) there is a vector a in $\mathrm{im}(P_X^\intercal)$ such that $u \odot a = e_i + e_j$.

Proof. If there are two vectors v, w paired with u such that $v \odot u = e_i$ and $w \odot u = e_j$ then $v + w \odot u = e_i + e_j$. As each basis vector in u must be safely correctable, u always has a vector $a = v + w \in \mathrm{im}(P_X^\intercal)$ such that $a \odot u = e_i + e_j$. Going the other way, there must be at least one paired vector v with u such that $u \cdot v = 1$ – we don't assume that $|u \odot v| = 1$, just that the dot product is 1, i.e. they have an odd number of intersecting basis vectors. This can be reduced to an intersection of 1 by applying a vector in $\mathrm{im}(P_X^\intercal)$ corresponding to a pair (e_i, e_j) to each of the pairs of intersecting basis vectors apart from the last one. This single basis vector can then be moved around u by further applications of vectors in $\mathrm{im}(P_X^\intercal)$. ■

We find this equivalent definition of gauge-fixing more helpful in practice, as it requires only data about the X stabilisers, rather than paired logical operators, the choice of which depends on the choice of basis of $H_1(C_\bullet)$.

Lemma 6.2. Let $\partial_A : A_1 \to A_0$ be a matrix over $\mathbb{F}_2$. Then for any $v \in \ker(\partial_A)$, either for any pair of basis vectors of A_1 $(e_i, e_j) \in \operatorname{supp}(v)$ there is a vector $b \in \operatorname{im}(\partial_A^\intercal)$ such that $v \odot b = e_i + e_j$ or there is another non-zero vector $u \in \ker(\partial_A)$ such that $\operatorname{supp}(u) \subset \operatorname{supp}(v)$.

Proof. Define the vector space $S = \{w \odot v : w \in \ker(\partial_A)^\perp\}$. Observe that any vector in S must have even Hamming weight, and that

$$S \cong \{w \odot v : w \in \ker(\partial_A \restriction_{\operatorname{supp}\ v})^\perp\} = \ker(\partial_A \restriction_{\operatorname{supp}\ v})^\perp,$$

as any vectors in $\ker(\partial_A)^\perp$ wholly outside of $\operatorname{supp}(v)$ will not contribute to the Hadamard product, and the isomorphism merely entails chopping off some entries which will always be 0. Now, $\dim(S) \leq |v| - 1$, as $\dim(\ker(\partial_A \restriction_{\operatorname{supp}\ v})) \geq 1$ by definition.

Suppose $\dim(S) = |v| - 1$. Then v has no other vectors in $\ker(\partial_A)$ contained in its support, as then $\dim(\ker(\partial_A \restriction_{\operatorname{supp}\ v})) = 1$, and for any pair of basis vectors of A_1 $(e_i, e_j) \in \operatorname{supp}(v)$ there is a vector $b \in \operatorname{im}(\partial_A^\intercal)$ such that $v \odot b = e_i + e_j$. To see this, view S as the row space of a matrix:

$$\begin{pmatrix} w_1 \odot v \\ w_2 \odot v \\ \vdots \\ w_m \odot v \end{pmatrix}$$

where $m = |v| - 1$, with some chosen basis of S. Then, put the matrix in row echelon form by performing Gaussian elimination:

$$\begin{pmatrix} 1 & * & * & * & \cdots & * \\ 0 & 1 & * & * & \cdots & * \\ & & & \vdots & & \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

where $*$ values are unknown. As $\dim(S) = |v| - 1$, without knowing anything else about the code, there will be exactly 1 row which is indented by 2 from the previous row. But each row must have an even number of 1s in it, including the last row, so the matrix must actually have the row echelon form:

$$\begin{pmatrix} 1 & * & * & \cdots & * & * \\ 0 & 1 & * & \cdots & * & * \\ 0 & 0 & 1 & \cdots & * & * \\ & & \vdots & & & \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

Then, add rows from the bottom to the top as necessary to give

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & * \\ 0 & 1 & 0 & \cdots & 0 & * \\ 0 & 0 & 1 & \cdots & 0 & * \\ & & \vdots & & & \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

i.e. an identity matrix with one column unknown to the right. But once again each row must have an even number of non-zero entries, as $S = \ker(\partial_A \restriction_{\operatorname{supp}\ v})^\perp$, so the column to the right must have all entries equal to 1. Therefore, we have $|v| - 1$ different pairs, and combinations of these suffice to give any pair in $\mathrm{supp}(v)$.

Now, suppose $\dim(S) < |v| - 1$. Then, as $\dim(\ker(\partial_A \restriction_{\mathrm{supp}\ v})^{\perp}) > 1$, there must be another vector in $\ker(\partial_A)$ contained in $\mathrm{supp}(v)$. ■

This means that a vector v in $\ker(\partial_A)$, with no other vectors in $\ker(\partial_A)$ contained in $\mathrm{supp}(v)$, will always have the property that for any pair of basis vectors of A_1 $(e_i, e_j) \in \mathrm{supp}(v)$ there is a vector $b \in \mathrm{im}(\partial_A^{\intercal})$ such that $v \odot b = e_i + e_j$.

This lemma implies that irreducibility and gauge-fixing coincide.

7 Error-corrected $\overline{Z}$-merge with the Shor code

In this appendix we work through an example explicitly, using the techniques of Section 2.4 to perform a distance 3 error-corrected $\overline{Z} \otimes \overline{Z}$ measurement between two copies of the Shor code, for which see Example 2.2.7.

Let us say the two copies are labelled $(C_\bullet, C^\bullet)$ and $(D_\bullet, D^\bullet)$, with

$$C_\bullet = D_\bullet = \ \mathbb{F}_2^6 \xrightarrow{\partial_2} \mathbb{F}_2^9 \xrightarrow{\partial_1} \mathbb{F}_2^2$$

and

$$\partial_2^{C_\bullet} = \partial_2^{D_\bullet} = \begin{pmatrix} 1&1&0&0&0&0\\ 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&1&1&0&0\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&1&1\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{pmatrix} \ ; \quad \partial_1^{C_\bullet} = \partial_1^{D_\bullet} = \begin{pmatrix} 1&1&1&1&1&1&0&0&0\\ 1&1&1&0&0&0&1&1&1 \end{pmatrix}.$$

We will use the $\overline{Z}$ operator $Z_1 \otimes Z_4 \otimes Z_7$, denoted $u = \begin{pmatrix} 1&0&0&1&0&0&1&0&0 \end{pmatrix}^{\intercal}$, with $u \in C_1$ and $u \in D_1$, to glue along.

The logical operator subcomplex $V_\bullet$ is then

$$V_\bullet = \ \mathbb{F}_2^3 \xrightarrow{\partial_1^{V_\bullet}} \mathbb{F}_2^2$$

with $\partial_1^{V_\bullet} = \begin{pmatrix} 1&1&0\\ 1&0&1 \end{pmatrix}$ and all other components of $V_\bullet$ being 0.

We now make the tensor product chain complex $W_\bullet = (P \otimes V)_\bullet$ from Definition 2.4.4, where $P_\bullet = \ P_1 \xrightarrow{\begin{pmatrix} 1\\ 1 \end{pmatrix}} P_0$. We have

$$W_\bullet = \ \mathbb{F}_2^3 \xrightarrow{\partial_2^{W_\bullet}} \mathbb{F}_2^8 \xrightarrow{\partial_1^{W_\bullet}} \mathbb{F}_2^4$$

with

$$\partial_2^{W_\bullet} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} ; \quad \partial_1^{W_\bullet} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

For $T_\bullet$ we take the two pushouts from Definition 2.4.6. First, we have

$$\begin{tikzcd} V_\bullet \arrow[r, hook, "g_\bullet"] \arrow[d, hook, "f_\bullet"'] & C_\bullet \arrow[d, "q_\bullet"] \\ W_\bullet \arrow[r, "p_\bullet"'] & R_\bullet \arrow[ul, phantom, "\ulcorner", very near start] \end{tikzcd}$$

Giving

$$R_\bullet = \; \mathbb{F}_2^9 \xrightarrow{\partial_2^{R_\bullet}} \mathbb{F}_2^{14} \xrightarrow{\partial_1^{R_\bullet}} \mathbb{F}_2^4$$

with $R_2 = W_2 \oplus C_2$, as $V_2 = 0$. The other components of $R_\bullet$ require taking quotients, identifying elements of W_1 and C_1, and the same for W_0 and C_0. One can then use Definition 2.3.1 to show that

$$\partial_2^{R_\bullet} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} ; \quad \partial_1^{R_\bullet} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} .$$

For the second pushout, that is

$$\begin{tikzcd} V_\bullet \arrow[r, hook] \arrow[d, hook] & W_\bullet \arrow[r] & R_\bullet \arrow[d] \\ D_\bullet \arrow[rr] & & T_\bullet \arrow[ul, phantom, "\ulcorner", very near start] \end{tikzcd}$$

we then have

$$T_\bullet = \; \mathbb{F}_2^{15} \xrightarrow{\partial_2^{T_\bullet}} \mathbb{F}_2^{20} \xrightarrow{\partial_1^{T_\bullet}} \mathbb{F}_2^4 \; .$$

The differentials are somewhat unwieldy, but we include them for completeness:

$$\partial_2^{T_\bullet} = \begin{pmatrix}
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1
\end{pmatrix}$$

$$\partial_1^{T_\bullet} = \begin{pmatrix}
1&0&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&1&1&0&0&0&1&1&1&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&1&1&1&0&0&0&1&1&1
\end{pmatrix}$$

One can check the various properties of this code. For example, $\mathrm{rank}\partial_1^{T_\bullet} = 4$ and $\mathrm{rank}\partial_2^{T_\bullet} = 15$. Thus $\dim H_1(T_\bullet) = \dim T_1 - 4 - 15 = 1$, and so the code $(T_\bullet, T^\bullet)$ encodes one logical qubit.

We can compare this with $(C \oplus D)_\bullet$ from before the merge:

$$(C \oplus D)_\bullet = \ \mathbb{F}_2^{12} \xrightarrow{\partial_2^{(C\oplus D)_\bullet}} \mathbb{F}_2^{18} \xrightarrow{\partial_1^{(C\oplus D)_\bullet}} \mathbb{F}_2^{4}$$

where the differentials are easy to see from those of $C_\bullet$ and $D_\bullet$, with $\partial_2^{(C\oplus D)_\bullet} = \partial_2^{C_\bullet} \oplus \partial_2^{D_\bullet}$ etc. Evidently, $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$ encodes 2 logical qubits. As expected, there are 2 new qubits, and 3 new Z-measurements in $(T_\bullet, T^\bullet)$. Each of the 2 new qubits participates in 2 of the new Z-measurements (and no other Z-measurements). We can check that $d_T^Z \geq 3$, i.e. the code has distance bounded below.

For the error-corrected $\overline{Z} \otimes \overline{Z}$ measurement, we therefore start with the code $((C \oplus D)_\bullet, (C \oplus D)^\bullet)$. Recall that this has $d = 3$. We then initialise the 2 new qubits in the $|+\rangle$ state and measure 3 rounds of the stabilisers specified by $\partial_2^{T_\bullet}$ and $\partial_1^{T_\bullet}$. As the 2 new qubits each participate in 2 of the new Z-measurements, the product of the outcomes is insensitive to initialisation errors. We apply the gauge-fixing operators from Example 2.4.2 to correct for the 3 new Z-measurements which may output the -1 measurement outcome. We end up with the code $(T_\bullet, T^\bullet)$.

8 Subsystem code distance calculation

Given a CSS code $C_\bullet$ defined by two parity-check matrices $P_X \in \mathbb{F}_2^{m_X \times n}$, $P_Z \in \mathbb{F}_2^{m_Z \times n}$ and a function f which calculates or estimates the distance of a CSS code, we show that the same function f can be called to calculate or estimate the distance of a subsystem CSS code. For simplicity we assume that f can yield d_Z, the Z-distance, or d_X, the X-distance, as desired, as this is what `DistRandCSS` can do [PSK22]. However, the method works even if f outputs $d = \min(d_Z, d_X)$.

First, find a spanning set of the gauge $\overline{Z}$ logicals $\mathcal{G} \subset H_1(C_\bullet)$. Typically this set will be a basis, but it does not have to be. Say the set chosen has cardinality l_Z. Then append these logicals to P_Z, making a new matrix $P'_Z \in \mathbb{F}_2^{(m_Z+l_Z)\times n}$. These gauge logicals must commute with the X stabilisers, so we have a new CSS code $C'_\bullet$ with parity-check matrices P_X and P'_Z. Apply f to $C'_\bullet$ to acquire d'_Z. This is the lowest weight $\overline{Z}$ logical which is not in the image of P'_Z, and so is the lowest weight dressed $\overline{Z}$ logical in our subsystem code.

Then do the same the other way round. Make a new matrix $P'_X \in \mathbb{F}_2^{(m_X+l_X)\times n}$, giving a new CSS code $C''_\bullet$ with parity-check matrices P'_X and P_Z. Apply f to acquire d'_X. Then the distance of our subsystem CSS code is $d' = \min(d'_Z, d'_X)$.

9 Computing colimits

Here we show explicitly how to calculate the colimits necessary for Algorithms 1 and 2. We only need to check it for coequalisers, as an external merge can always be viewed as an internal merge within a direct sum codeblock.

We start off with the following diagram

$$\begin{tikzcd}
 & V_2 \arrow[r] \arrow[ddl, "g_2"'] & V_1 \arrow[r] \arrow[ddl, "g_1"'] & V_0 \arrow[ddl, "g_0"'] \\
V_2 \arrow[r] \arrow[d, "f_2"'] & V_1 \arrow[r] \arrow[d, "f_1"'] & V_0 \arrow[d, "f_0"'] & \\
R_2 \arrow[r] \arrow[d, "\mathrm{coeq}_2"] & R_1 \arrow[r] \arrow[d, "\mathrm{coeq}_1"] & R_0 \arrow[d, "\mathrm{coeq}_0"] & \\
T_2 \arrow[r] & T_1 \arrow[r] & T_0 &
\end{tikzcd}$$

and our task is to find ∂_2^T and ∂_1^T. We know from the universal property that once the components T_i are fixed, the mediating maps are also unique, so we need only find differentials such that the diagram commutes.

Recall that $g_\bullet$ and $f_\bullet$ are basis-preserving chain maps, and $\mathrm{coeq}_i := \mathrm{coeq}(f_i, g_i)$ is the coequaliser of matrices at degree i, which is also basis-preserving. $V_\bullet$ is simultaneously the operator subcomplex of two irreducible $\overline{Z}$ logicals in $R_\bullet$. The first observation is that for all merges in the Z basis, $V_2 = 0$, hence $f_2 = g_2 = 0$ and $T_2 = R_2$. Therefore $\partial_2^T = \mathrm{coeq}_1 \circ \partial_2^R$. This can be computed simply be taking the XOR of pairs of rows in the mutual image of g_1 and f_1 and assigning the new rows entries in ∂_2^T – it does not matter which so long as we are consistent.

For ∂_1^T, we have that $\partial_1^T \circ \mathrm{coeq}_1 = \mathrm{coeq}_0 \circ \partial_1^R$. We can compute $\partial_1^T \circ \mathrm{coeq}_1$ in the same way, by taking XORs of pairs of rows. We must now find ∂_1^T. This time we take the OR (not XOR) of pairs of columns in $\partial_1^T \circ \mathrm{coeq}_1$ which correspond to basis elements in the mutual image of g_1 and f_1. The assignment of new column entries must be consistent with the assignment of row entries to ∂_2^T so that coeq_1 is identical to before.

One can check that the diagram now commutes everywhere, and so we have calculated the mediating maps, i.e. differentials of $T_\bullet$, successfully. When performing an external merge the matrices of $R_\bullet$ are block-diagonal and this procedure reduces to the computation described in Section 3.3.

10 Generalised bicycle codes

Generalised bicycle (GB) codes are codes with $P_X = \begin{pmatrix} A & B \end{pmatrix}$ and $P_Z = \begin{pmatrix} B^\intercal & A^\intercal \end{pmatrix}$ where A and B are elements of $\mathscr{C}_\ell$, the ring of circulant matrices [PK21, KP13]. GB codes are a special case of lifted product codes where the classical codes over $\mathscr{C}_\ell$ have the check matrices A and B. In this sense they are smallest possible lifted product codes.

$\mathscr{C}_\ell \cong \mathbb{F}_2^{\langle \ell \rangle}$, so A and B can thus be described uniquely by polynomials over $\mathbb{F}_2$ in a single variable x, where $x^\ell = 1$.

There is no exact formula for all the parameters of GB codes, but certain input polynomials have been found to yield exceptional parameters. For example, $\ell = 63$ with $a(x) = 1 + x + x^{14} + x^{16} + x^{22}$ and $b(x) = 1 + x^3 + x^{13} + x^{20} + x^{42}$ gives a $[\![126, 28, 8]\!]$ code. In all our benchmarks here we use the GB codes (A1) to (A5) from [PK21, App. B], ranging from $n = 46$ to $n = 254$. For codes (A1) and (A5) the distance is not known, with only upper and lower bounds given. We assume the upper bound is tight for the purposes of the paper. The distances are large enough that we do not check distances of merged codes using Z3 but instead rely solely on `QDistRnd` to estimate.

10.1 Individual merges

We perform the same benchmarking as in Section 3.3.2, but with GB codes instead of LCS codes. The codes reach high dimensions, with up to 28 logical qubits, so we only perform merges using the first 7 logical qubits to prevent exorbitant compute time. As we are only using `QDistRnd` to estimate the merged code distances we cannot prove that at the merge depths given the distance does not decrease, but this is somewhat justified as the distance is not even known for some of the initial codes.

Results are shown in Figure 1 for individual X and Z merges respectively. We also show a comparison with surface codes and [CKBB22] in Figure 2.

GB codes appear to be highly amenable to surgery, with extremely efficient individual merges compared to surface codes and [CKBB22]. We cannot discount the possibility that `QDistRnd` fails to find low weight logicals in the merged codes, however.

10.2 Parallel merges

We redo the benchmarking in Section 3.3.2 but with GB codes instead. We again restrict the benchmark to only parallelise up to 7 merges for efficiency of computation.

We see that even when parallelised, the merges are still quite cheap; in particular the total number of qubits required increases much slower than for surface codes or [CKBB22]. We do not have a good understanding of what about GB codes allows for this efficiency,

ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
23	1	0.23	9
24	1	0.19	9
63	1.29	0.35	11
90	1.14	0.24	9
127	1	0.24	11

ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
23	1	0.22	9
24	1.17	0.3	9
63	1	0.25	11
90	1	0.18	9
127	1	0.23	11

Figure 1: Figures of merit for individual X and Z merges between GB codes.

ℓ	n_{initial}	$\langle n_{\text{ancilla}}\rangle$	$\langle n_{\text{total}}\rangle$
23	92	20.24	112.24
	580	8	588
	92	560	652
24	96	28.8	124.8
	1356	7	1363
	96	541.33	637.33
63	252	63	315
	6328	7	6335
	252	1526.86	1778.86
90	360	64.8	424.8
	12260	17	12277
	360	3857.71	4217.71
127	508	116.84	624.84
	42616	19	42635
	508	7493.71	8001.71

Figure 2: Comparison of GB code individual Z-merges to surface codes and [CKBB22]. The first row in each box is our homological approach using Algorithm 1. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to GB codes.

ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	1	0.46	10
24	1	1.125	14
63	2	4.34	17
90	1	1.34	15
127	1	1.66	17

ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	1	0.42	10
24	1	1.35	14
63	1	1.71	17
90	1	1.3	15
127	1	1.64	17

Figure 3: Figures of merit for parallel X and Z merges between GB codes.

ℓ	n_{initial}	n_{ancilla}	n_{total}
23	92	39	131
	580	16	596
	92	1120	1212
24	96	130	226
	1356	42	1398
	96	3248	3344
63	252	430	682
	6328	49	6377
	252	10688	10940
90	360	468	828
	12260	119	12379
	360	27004	27364
127	508	831	1339
	42616	133	42749
	508	52456	52964

Figure 4: Comparison of GB code parallel Z-merges to surface codes and [CKBB22]. The first row in each box is our homological approach using Algorithm 1. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to GB codes.

ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
23	1.5	0.85	9
24	1.7	0.77	9
63	1.29	0.7	11
90	1.57	0.74	9
127	1	0.47	11

ℓ	$\langle r\rangle$	$\langle n_{\text{ancilla}}/n_{\text{initial}}\rangle$	$\langle\omega\rangle$
23	2	1.12	9
24	1.5	0.86	9
63	1	0.49	11
90	1.71	0.8	9
127	1	0.47	11

Figure 5: Figures of merit for individual single-qubit logical X and Z measurements with GB codes.

and it is possible that `QDistRnd` is giving an upper bound on distance which isn't tight, which would lead to merges which appear more efficient than they are. Lastly, the stabiliser weights increase substantially, up to a maximum of 17, to the point that fault-tolerance may be impossible with such merged codes.

10.3 Individual single-qubit measurements

This section is a re-run of Section 3.3.2 but for GB codes, and once again using only the first 7 logical qubits. See Figure 5 and Figure 6 for the results.

10.4 Parallel single-qubit measurements

This time we re-run Section 3.3.2, but with the same modifications for GB codes: we use `QDistRnd` to estimate code distances, and measure at most the first 7 logical qubits in parallel. See Figure 7 and Figure 8 for results. Unfortunately, the compute time of parallel single-qubit measurements for $\ell = 127$ became too high, so we omit this data.

11 Detailed SSIP results

In this appendix we present more fine-grained versions of the tables presented in Section 3.3. Throughout, i refers to the logical qubit used in the merge/measurement.

12 The vacuum space of $D(G)$ models

This appendix finds expressions for an orthogonal basis of $\mathcal{H}_{vac}$ in the $D(G)$ models, following [CDH20]. This is included for completeness in order to have a self-contained account of the theory. Let $g := \bigotimes_{l\in E} g^l$ be the state in $\mathcal{H}$ with a group element g^l viewed

ℓ	n_{initial}	$\langle n_{\text{ancilla}} \rangle$	$\langle n_{\text{total}} \rangle$
23	46	51.52	97.52
	290	0	290
	46	275.5	321.5
24	48	41.28	89.28
	678	0	678
	48	266.67	314.67
63	126	61.74	187.74
	3164	0	3164
	126	759.43	885.42
90	180	144	324
	6130	0	6130
	180	1919.86	2099.86
127	254	119.38	373.38
	21308	0	21308
	254	3736.86	3990.86

Figure 6: Comparison of GB code individual single-qubit logical Z-measurements to surface codes and a naive application of [CKBB22]. The first row uses the method described in Section 3.2.3. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to GB codes.

ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	2	1.26	9
24	2	3	11
63	2	8.68	17
90	2	5.46	13

ℓ	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	2	1.15	9
24	3	6.15	11
63	1	3.41	17
90	2	4.87	13

Figure 7: Figures of merit for parallel single-qubit logical X and Z measurements with GB codes.

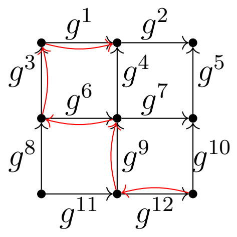
ℓ	n_{initial}	n_{ancilla}	n_{total}
23	46	53	99
	290	0	290
	46	551	597
24	48	295	343
	678	0	678
	48	1600	1648
63	126	430	556
	3164	0	3164
	126	5316	5442
90	180	877	1057
	6130	0	6130
	180	13439	13619

Figure 8: Comparison of GB code parallel single-qubit logical Z-measurements to surface codes and a naive application of [CKBB22]. The first row uses the method described in Section 3.2.3. The second is lattice surgery with surface code patches. The third is a naive application of [CKBB22] to GB codes.

in $\mathbb{C}G$ at each edge. Let γ be an oriented path in the lattice. Now, define

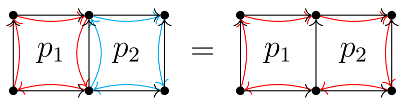
$$\gamma(g) := \prod_{l\in\gamma}(g^l)^\epsilon$$

where $\epsilon = 1$ if the path orientation agrees with the lattice orientation, and -1 otherwise. For example, given a segment of the lattice segment with an oriented path γ,



we would have $\gamma(g) := (g^{12})^{-1}g^9(g^6)^{-1}g^3g^1$. (We choose arrow composition to be this way round, rather than $\gamma(g) := g^1g^3(g^6)^{-1}g^9(g^{12})^{-1}$, as it is convenient for the proof of Theorem 12.2 below.)

Observe that for $B(p)$ at a given face p, the condition $B(p)g = g$ is equivalent to $\partial p(g) = e$, where ∂p is the boundary of p interpreted as a clockwise-oriented path, and e is the identity element of G. Note that the choice of basepoint of this path is immaterial, as the product is still e under cyclic rotations. Now, consider two adjacent boundaries ∂p_1 (red) and ∂p_2 (cyan) such that $\partial p_1(g) = \partial p_2(g) = e$. Then $\partial p_{1,2}(g) = \partial p_1(g)\partial p_2(g) = e$ for the boundary of the combined face,



It follows that the subspace $\{\psi \mid B(p)\psi = \psi \text{ for all } p\}$ is spanned by the following set:

$$S = \{g \mid \partial p(g) = e \text{ for all } p\} = \{g \mid \gamma(g) = e \text{ for all contractible closed } \gamma\}.$$

L	ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	0	1	0.17	6
1	3	1	1	0.17	6
1	3	2	1	0.1	6
1	4	0	1	0.125	6
1	4	1	1	0.175	6
1	4	2	1	0.125	6
1	4	3	1	0.075	6
1	5	0	1	0.1	6
1	5	1	1	0.16	6
1	5	2	1	0.14	6
1	5	3	1	0.1	6
1	5	4	1	0.06	6
2	4	0	2	0.25	7
2	4	1	2	0.36	7
2	4	2	2	0.25	7
2	4	3	1	0.11	7
2	5	0	3	0.33	7
2	5	1	1	0.1	7
2	5	2	3	0.45	7
2	5	3	3	0.33	7
2	5	4	3	0.49	7
2	6	0	3	0.28	7
2	6	1	3	0.44	7
2	6	2	3	0.34	7
2	6	3	3	0.37	7
2	6	4	3	0.28	7
2	6	5	3	0.49	7
3	5	0	2	0.15	7
3	5	1	3	0.23	7
3	5	2	1	0.072	7
3	5	3	3	0.45	7
3	5	4	3	0.456	7
3	6	0	4	0.25	7
3	6	1	1	0.06	7
3	6	2	1	0.057	7
3	6	3	4	0.27	7
3	6	4	4	0.53	7
3	6	5	4	0.7	7

Figure 9: Expanded figures of merit for individual Z-merges between LCS codes.

L	ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	0	1	0.17	6
1	3	1	1	0.2	6
1	3	2	1	0.1	6
1	4	0	1	0.15	6
1	4	1	1	0.125	6
1	4	2	1	0.2	6
1	4	3	1	0.075	6
1	5	0	1	0.14	6
1	5	1	1	0.12	6
1	5	2	1	0.1	6
1	5	3	1	0.2	6
1	5	4	1	0.06	6
2	4	0	1	0.096	7
2	4	1	1	0.14	7
2	4	2	1	0.15	7
2	4	3	1	0.048	7
2	5	0	2	0.28	7
2	5	1	2	0.2	7
2	5	2	2	0.37	7
2	5	3	3	0.91	7
2	5	4	1	0.038	7
2	6	0	2	0.26	7
2	6	1	2	0.24	7
2	6	2	2	0.19	7
2	6	3	2	0.35	7
2	6	4	3	0.74	7
2	6	5	1	0.032	7
3	5	0	2	0.21	7
3	5	1	2	0.3	7
3	5	2	3	0.5	7
3	5	3	3	0.71	7
3	5	4	3	0.15	7
3	6	0	2	0.19	7
3	6	1	4	0.40	7
3	6	2	2	0.30	7
3	6	3	3	0.46	7
3	6	4	2	0.35	7
3	6	5	4	0.18	7

Figure 10: Expanded figures of merit for individual X-merges between LCS codes.

L	ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	0	2	1.1	6
1	3	1	2	1.1	6
1	3	2	1	0.2	6
1	4	0	2	0.8	6
1	4	1	2	1.05	6
1	4	2	2	0.8	6
1	4	3	1	0.15	6
1	5	0	2	0.64	6
1	5	1	2	0.96	6
1	5	2	2	0.84	6
1	5	3	2	0.64	6
1	5	4	1	0.12	6
2	4	0	1	0.17	7
2	4	1	1	0.25	7
2	4	2	1	0.17	7
2	4	3	3	0.98	7
2	5	0	3	0.66	7
2	5	1	2	0.58	7
2	5	2	4	1.25	7
2	5	3	4	0.92	7
2	5	4	4	1.37	7
2	6	0	4	0.77	7
2	6	1	4	1.22	7
2	6	2	4	0.95	7
2	6	3	4	1.04	7
2	6	4	4	0.77	7
2	6	5	4	1.37	7
3	5	0	1	0.11	7
3	5	1	3	0.46	7
3	5	2	2	0.42	7
3	5	3	4	1.25	7
3	5	4	4	1.28	7
3	6	0	3	0.35	7
3	6	1	2	0.35	7
3	6	2	3	0.55	7
3	6	3	2	0.23	7
3	6	4	2	0.46	7
3	6	5	5	1.8	7

Figure 11: Expanded figures of merit for individual single qubit logical Z-measurements with LCS codes.

L	ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
1	3	0	2	1.0	6
1	3	1	2	1.2	6
1	3	2	1	0.2	6
1	4	0	2	0.9	6
1	4	1	2	0.75	6
1	4	2	2	1.2	6
1	4	3	1	0.15	6
1	5	0	2	0.84	6
1	5	1	2	0.72	6
1	5	2	2	0.6	6
1	5	3	2	1.2	6
1	5	4	1	0.12	6
2	4	0	1	0.19	7
2	4	1	2	0.81	7
2	4	2	3	1.46	7
2	4	3	1	0.1	7
2	5	0	3	0.94	7
2	5	1	2	0.4	7
2	5	2	3	1.21	7
2	5	3	4	2.54	7
2	5	4	1	0.08	7
2	6	0	2	0.85	7
2	6	1	3	0.78	7
2	6	2	3	0.62	7
2	6	3	3	1.14	7
2	6	4	3	1.49	7
2	6	5	1	0.06	7
3	5	0	4	0.97	7
3	5	1	4	1.39	7
3	5	2	4	1.4	7
3	5	3	3	1.42	7
3	5	4	4	0.42	7
3	6	0	2	0.37	7
3	6	1	5	1.03	7
3	6	2	1	0.21	7
3	6	3	4	1.29	7
3	6	4	4	1.6	7
3	6	5	4	0.35	7

Figure 12: Expanded figures of merit for individual single qubit logical X-measurements with LCS codes.

ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	0	1	0.22	9
23	1	1	0.21	9
24	0	2	0.69	9
24	1	1	0.22	9
24	2	1	0.25	9
24	3	1	0.23	9
24	4	1	0.21	9
24	5	1	0.2	9
63	0	1	0.25	11
63	1	1	0.24	11
63	2	1	0.25	11
63	3	1	0.24	11
63	4	1	0.25	11
63	5	1	0.25	11
63	6	1	0.24	11
90	0	1	0.18	9
90	1	1	0.19	9
90	2	1	0.17	9
90	3	1	0.21	9
90	4	1	0.18	9
90	5	1	0.19	9
90	6	1	0.18	9
127	0	1	0.22	11
127	1	1	0.25	11
127	2	1	0.23	11
127	3	1	0.23	11
127	4	1	0.23	11
127	5	1	0.23	11
127	6	1	0.24	11

Figure 13: Expanded figures of merit for individual Z-merges between GB codes.

ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	0	1	0.24	9
23	1	1	0.22	9
24	0	1	0.2	9
24	1	1	0.2	9
24	2	1	0.2	9
24	3	1	0.2	9
24	4	1	0.17	9
24	5	1	0.17	9
63	0	1	0.23	11
63	1	1	0.23	11
63	2	1	0.25	11
63	3	1	0.25	11
63	4	2	0.63	11
63	5	1	0.23	11
63	6	2	0.63	11
90	0	1	0.22	9
90	1	2	0.56	9
90	2	1	0.21	9
90	3	1	0.18	9
90	4	1	0.22	9
90	5	1	0.16	9
90	6	1	0.16	9
127	0	1	0.23	11
127	1	1	0.24	11
127	2	1	0.24	11
127	3	1	0.24	11
127	4	1	0.22	11
127	5	1	0.24	11
127	6	1	0.24	11

Figure 14: Expanded figures of merit for individual X-merges between GB codes.

ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	0	2	1.15	9
23	1	2	1.09	9
24	0	3	2.25	9
24	1	1	0.44	9
24	2	1	0.5	9
24	3	1	0.46	9
24	4	2	1.08	9
24	5	1	0.4	9
63	0	1	0.5	11
63	1	1	0.48	11
63	2	1	0.5	11
63	3	1	0.49	11
63	4	1	0.49	11
63	5	1	0.49	11
63	6	1	0.48	11
90	0	2	0.94	9
90	1	1	0.38	9
90	2	2	0.89	9
90	3	2	1.08	9
90	4	2	0.97	9
90	5	2	1.0	9
90	6	1	0.37	9
127	0	1	0.43	11
127	1	1	0.49	11
127	2	1	0.46	11
127	3	1	0.47	11
127	4	1	0.46	11
127	5	1	0.47	11
127	6	1	0.48	11

Figure 15: Expanded figures of merit for individual single qubit logical Z-measurements with GB codes.

ℓ	i	r	$n_{\text{ancilla}}/n_{\text{initial}}$	ω
23	0	2	1.26	9
23	1	1	0.43	9
24	0	2	1.0	9
24	1	2	1.0	9
24	2	2	1.0	9
24	3	1	0.4	9
24	4	1	0.33	9
24	5	2	0.88	9
63	0	1	0.47	11
63	1	1	0.47	11
63	2	1	0.5	11
63	3	1	0.49	11
63	4	2	1.25	11
63	5	1	0.47	11
63	6	2	1.25	11
90	0	1	0.43	9
90	1	2	1.12	9
90	2	1	0.42	9
90	3	2	0.94	9
90	4	2	1.17	9
90	5	2	0.82	9
90	6	1	0.31	9
127	0	1	0.46	11
127	1	1	0.48	11
127	2	1	0.49	11
127	3	1	0.48	11
127	4	1	0.45	11
127	5	1	0.48	11
127	6	1	0.47	11

Figure 16: Expanded figures of merit for individual single qubit logical X-measurements with GB codes.

Clearly, S is invariant under change of orientation of γ. Next, we define an equivalence relation on S. We say that $g \sim g'$ if $g' = \bigotimes_{v \in V} h_v \triangleright_v g$ for some collecton $\{h_v \in G\}$. In other words, there is some sequence of vertex operators that takes g to g'. The set of equivalence classes is $[S]$, and a given class is called $[g]$. Define

$$\kappa_{[g]} := \sum_{g' \in [g]} g' \in \mathcal{H}$$

For any two tensor products states $g = \bigotimes_{l \in E} g^l$ and $g' = \bigotimes_{l \in E} g'^l$ in $\mathcal{H}$, define the inner product $(g, g') = \prod_{l \in E} \delta_{g^l, g'^l}$.

Lemma 12.1. $\{\kappa_{[g]} \mid [g] \in [S]\}$ forms an orthogonal basis of $\mathcal{H}_{vac}$.

Proof. Clearly, $h \triangleright \kappa_{[g]} = \kappa_{[g]}$. Therefore, $\kappa_{[g]} = A(v)\kappa_{[g]}, \forall v \in V$ and we also know that $\kappa_{[g]} = B(p)\kappa_{[g]}, \forall p \in P$, so $\kappa_{[g]} \in \mathcal{H}_{vac}$. In addition, for any two $\kappa_{[g]}$ and $\kappa_{[g']}$, either $\kappa_{[g]} = \kappa_{[g']}$ or they have no overlapping terms, by definition of the equivalence relation. Therefore, $(\kappa_{[g]}, \kappa_{[g']}) = |[g]|\delta_{[g],[g']}$, where $|[g]|$ is the cardinality of $[g]$. Thus all $\kappa_{[g]}$ are orthogonal.

Next we prove that $\kappa_{[g]}$ span $\mathcal{H}_{vac}$. For any state $\psi \in \mathcal{H}_{vac}$, write $\psi = \sum_{g \in S} \alpha_g g$, where $g = \bigotimes_{l \in E} g^l$. Now, choose a vertex v. We know that $h \triangleright_v \psi = \psi$, $\forall h \in G$. Given some g, consider the set of states $\{g'\}$ such that g' agrees with g everywhere except at v, where $g' = h' \triangleright_v g$ for some $h' \in G$. For any such g', $h \triangleright_v g' \in \{g'\}$, so by definition $h \triangleright_v$ permutes through the set. Therefore, as all g are orthogonal and $h \triangleright_v \sum_{g \in S} \alpha_g g = \sum_{g \in S} \alpha_g g$, each element in $\{g'\}$ must appear with the same weight. Repeating for all vertices, it is clear that $\psi = \sum_{[g] \in [S]} \beta_{[g]} \kappa_{[g]}$, for some coefficients $\{\beta_{[g]}\}$, and hence that $\{\kappa_{[g]} \mid [g] \in [S]\}$ spans $\mathcal{H}_{vac}$. ■

Theorem 12.2. [CDH20] Let Σ be a closed, orientable surface. Then

$$\dim(\mathcal{H}_{vac}) = |\mathrm{Hom}(\pi_1(\Sigma), G)/G|.$$

where the G-action on any $\phi \in \mathrm{Hom}(\pi_1(\Sigma), G)$ is $\phi \mapsto \{h\phi h^{-1} \mid h \in G\}$.

Proof. We define an equivalence relation between closed, but not necessarily contractible, paths acting on the ground state, by $\gamma \sim \gamma'$ if $\gamma = \gamma' \prod_{p \in I} \partial p$, for some set of faces $I \subseteq P$. Denoting the set of all closed paths K, the equivalence relation defines a homotopy class of Σ. By taking the obvious group composition we identify $[K]$, the set of equivalence classes, with $\pi_1(\Sigma)$. We now define a map

$$\Theta : S \to \mathrm{Hom}(\pi_1(\Sigma), G), \quad \Theta(g)([\gamma]) := \gamma(g),$$

where γ is any closed path in $[\gamma]$. The choice of γ is immaterial, as $\partial p(g) = e, \forall p \in P$. For any $g \in S$, let $[\gamma]_0$ be the class of contractible, closed paths, i.e. the identity of $\pi_1(\Sigma)$. By definition, $\gamma_0(g) = e \in G$, for any $\gamma_0 \in [\gamma]_0$. Now, again for any $g \in S$, let $[\gamma]_a$ and $[\gamma]_b$ be two classes of closed paths. Let $\gamma_a(g) = g_a$ and $\gamma_b(g) = g_b$. Observe that $(\gamma_a \circ \gamma_b)(g) = g_a g_b$. Therefore the image of Θ is indeed in the set $\mathrm{Hom}(\pi_1(\Sigma), G)$ of group homomorphisms.

Next, we show that Θ is surjective. For any group homomorphism $\phi : \pi_1(\Sigma) \to G$, consider a maximum spanning tree T on Σ, with root r. By definition, T has $m := |V| - 1$ edges. For any edge ϵ outwith the tree, let u_ϵ and v_ϵ be the end vertices of ϵ. u_ϵ and v_ϵ are in T. There is now a unique path γ_u through T from r to u_ϵ and γ_v from r to v_ϵ. Therefore, we may define a closed path:

$$\gamma_\epsilon = \gamma_u \circ \epsilon \circ \gamma_v^{-1}$$

where γ_v^{-1} is the reverse path of γ_v. By construction of T, the group element $\epsilon(g)$ associated to ϵ is uniquely fixed by the group elements $\gamma_u(g)$ and $\gamma_v(g)$. Conversely, given edge ϵ each group element $\epsilon(g)$ may be acquired by $|G|^m$ choices of group elements for edges in T. Applying the same logic for any edge outwith T, Θ is therefore a $|G|^m$-to-1 map. This is invariant under choice of root r and maximum spanning tree T.

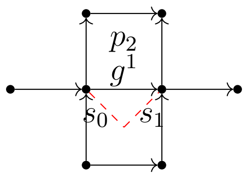
The proof is now completed by setting up a bijection between $[S]$ and orbits of $\mathrm{Hom}(\pi_1(\Sigma), G)$ under the G-action. By definition, any closed path through a given vertex $v \neq r$ will have exactly one incoming arrow and one outgoing arrow. The product along this path is invariant under $h\triangleright_v$, so $\Theta(g) = \Theta(h\triangleright_v g)$. However, $\Theta(h\triangleright_r g) = h\Theta(g)h^{-1}$, as r is the endpoint of the path. Therefore, the preimage of any $\phi \in \mathrm{Hom}(\pi_1(\Sigma), G)$ is exactly the set of elements of S which agree on edges adjacent to r, but are just related by some family $h_v\triangleright_v$ at other vertices. Additionally, if $\Theta(g) = \phi$, then $\Theta([g]) = G\triangleright\phi = \{h\phi h^{-1} \mid h \in G\}$. Therefore, $[S] \cong \mathrm{Hom}(\pi_1(\Sigma), G)/G$, where the G-action is conjugacy as above. Using Lemma 12.1, $\dim(\mathcal{H}_{vac}) = |\mathrm{Hom}(\pi_1(\Sigma), G)/G|$. ■

13 Proof of part (2) of Proposition 4.2.10

That $|\psi^{h,g}\rangle \in \mathcal{L}(s_0, s_1)$ follows from the commutation relations with operators at sites $t \neq s_0, s_1$ in Lemma 4.2.8. In this Appendix, we show these states span $\mathcal{L}(s_0, s_1)$. Note that there is a stronger claim in [BM-D08, Prop 7] for their 'ribbon algebra' $\mathcal{F}_\rho$ but we have not been able to reproduce the proof there at a number of points, specifically (B58), (B59), (B62) and (B63) appear to assume that certain projectors are right-cancellable, which in general is not possible.

Our proof by induction will involve 3 series of cases: (i) the base cases, when s_0 and s_1 are separated only be a single edge (direct or dual); (ii) the case when s_0, s_1 are distance 2 away, i.e. the smallest ribbon connecting them has exactly 2 triangle operatorions; (iii) distance 3 or greater.

(i) The two base cases occur when s_0 and s_1 are adjacent, so the minimal ribbon ξ required is of length 1, either a direct or dual triangle. We start with a direct triangle, for example

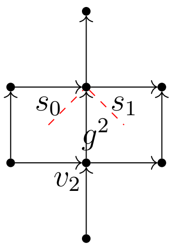


where $s_0 = (v_0, p_0)$ and $s_1 = (v_1, p_0)$. Consider a state $|\Psi\rangle \in \mathcal{L}(s_0, s_1)$ and all operators O on $\mathcal{L}(s_0, s_1)$ such that $O|\mathrm{vac}\rangle = |\Psi\rangle$. Since the conditions on $|\Psi\rangle$ away from the end sites are the same as for a vacuum, we can assume that O acts trivially on $|\mathrm{vac}\rangle$ around all vertices $v \notin \{v_0, v_1\}$ and $p \neq p_0$ and hence that O can be chosen to act only on the edge shared by (v_0, p_0) and (v_1, p_0), which has state g^1 as in the diagram. A fuller explanation requires arguments similar to those for the vacuum in Appendix 12.

Note next that $\mathrm{End}(\mathbb{C}G) \cong \mathbb{C}(G) \rtimes \mathbb{C}G \cong \mathbb{C}G \ltimes \mathbb{C}(G)$, which is to say any operator acting on $g^1 \in \mathbb{C}G$ is a sum of terms factorising as $\mathbb{C}G$ acting by multiplication (one can fix the side to be from the left or the right) and $\mathbb{C}(G)$ acting by evaluation against the coproduct, the second of these being the action of a direct triangle operator. But any contributions from a nontrivial part of $\mathbb{C}G$ in O will cease to satisfy $B(p_2)O|\mathrm{vac}\rangle = O|\mathrm{vac}\rangle$ from the conditions for $\mathcal{L}(s_0, s_1)$, so $O \in \mathbb{C}(G)$ acts like a direct triangle operator. Hence

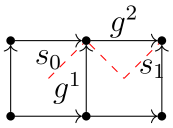
$\{T^g_\xi|\text{vac}\rangle \mid g \in G\}$ span $\mathcal{L}(s_0, s_1)$, and $T^g_\xi = F^{e,g}_\xi$, (or any h in place of e) so $\mathcal{L}(s_0, s_1)$ is spanned by $\{\psi^{e,g} \mid g \in G\}$, and therefore also by $\{\psi^{h,g} \mid h, g \in G\}$.

For the dual-triangle, we similarly consider



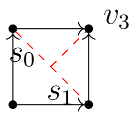
Now, $|\Psi\rangle$ can be characterised by an operator O which acts only on g^2 and similarly factorises as $\mathbb{C}G$ and $\mathbb{C}(G)$ acting as before, where the first is the action of a dual triangle operator. But any contributions from a nontrivial part of $\mathbb{C}(G)$ in O will cease to satisfy $A(v_2)O|\text{vac}\rangle = O|\text{vac}\rangle$, so $O \in \mathbb{C}G$ acts like a dual triangle operator. Hence $\{L^g h\xi|\text{vac}\rangle \mid h \in G\}$ span $\mathcal{L}(s_0, s_1)$, and $L^h_\xi = F^{h,e}_\xi$, so $\mathcal{L}(s_0, s_1)$ is spanned by $\{\psi^{h,e} \mid h \in G\}$, and therefore also by $\{\psi^{h,g} \mid h, g \in G\}$. This concludes the base cases.

(ii) There are four distance 2 cases, which can all be calculated. If s_0, s_1 occupy positions as in

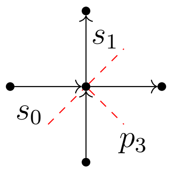


where the smallest ribbon has 1 direct and 1 dual triangle such that $\xi = \tau \circ \tau^*$, then by the same arguments as above $|\Psi\rangle \in \mathcal{L}(s_0, s_1)$ can be characterised by an operator O such that $|\Psi\rangle = O|\text{vac}\rangle$, where O acts only on the edges g^1, g^2, and we can see by considering $A(v)$ and $B(p)$ acting at v and p adjacent to g^1, g^2 that O must be a sum of terms $T^g_\tau \circ L^h_{\tau^*}$. We can then set $F^{h,g}_{\tau_2\circ\tau_1} = T^g_{\tau_2} \circ L^h_{\tau_1}$. That $\{F^{h,g}_{\tau\circ\tau^*}|\text{vac}\rangle \mid h, g \in G\}$ spans $\mathcal{L}(s_0, s1)$ is then immediate. The same argument applies if $\xi = \tau^* \circ \tau$ instead, but the other way round.

If the smallest ribbon is instead 2 direct triangles, for example



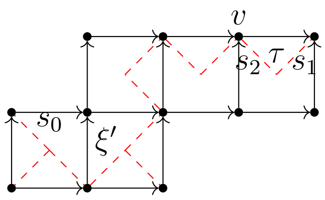
then consider $T^{g^2}_{\tau_2} \circ T^{g^1}_{\tau_1}|\text{vac}\rangle$, for any $g^1, g^2 \in G$, $T^{g^2}_{\tau_2} \circ T^{g^1}_{\tau_1}|\text{vac}\rangle \in \mathcal{L}(s_0, s_1)$ iff $g^1 = g^2$, by considering commutation with $A(v_3)$. The same applies for different orientations, and the same argument for dual triangles. For example



with $h^1 = h^2$ by considering $B(p_3)$. That $\{F^{h,g}_{\tau_2\circ\tau_1} \mid h, g \in G\}$ spans $\mathcal{L}(s_0, s_1)$ is then immediate, where either the first or second variable is surplus respectively.

(iii) For any s_0, s_1 which are distance 3 or further, we follow similar arguments but with an extended set of chosen edges which characterise the state $|\Psi\rangle$ along a chosen ribbon ξ between s_0 and s_1. Outside of this ribbon, the operator O used to characterise $|\Psi\rangle$ must act trivially. Unlike the previous cases, it must also act trivially at at least one site inside the ribbon too, and we use this to calculate the states.

The last triangle in the ribbon ξ must be either direct or dual, so we cover a similar splitting of cases into direct and dual as in (i). First we consider the direct non-adjacent case, $\xi = \tau \circ \xi'$, for example



Assume that $\mathcal{L}(s_0, s_2)$ is spanned by $\{F_{\xi'}^{h,g}|\text{vac}\rangle\}$ and observe that $\mathcal{L}(s_0, s_1)$ is a subspace of the space spanned by $\{T_\tau^{g'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle \mid g', g, h \in G\}$ or $\{F_\tau^{e,g'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle \mid g', g, h \in G\}$. Specifically, $\mathcal{L}(s_0, s_1)$ is the subspace where $A(v)$ acts as the identity for v at the site connecting ξ' and τ. Hence, for any $O|\text{vac}\rangle \in \mathcal{L}(s_0, s_1)$,

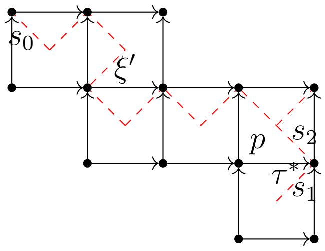
$$O|\text{vac}\rangle = \frac{1}{|G|}\sum_{h' \in G} h' \triangleright_v O|\text{vac}\rangle$$

which we apply to $O = F_\tau^{e,g'} \circ F_{\xi'}^{h,g}$,

$$\begin{aligned} F_\tau^{e,g'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle &= \frac{1}{|G|}\sum_{h' \in G} h' \triangleright_v F_\tau^{e,g'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle = \frac{1}{|G|}\sum_{h' \in G} F_\tau^{e,h'g'} \circ F_{\xi'}^{h,gh'^{-1}}|\text{vac}\rangle \\ &= \frac{1}{|G|}\sum_{f \in G} F_\tau^{e,f^{-1}gg'} \circ F_{\xi'}^{h,f}|\text{vac}\rangle = \sum_{f \in G} F_\tau^{f^{-1}hf,f^{-1}gg'} \circ F_{\xi'}^{h,f}|\text{vac}\rangle = F_\xi^{h,gg'}|\text{vac}\rangle \end{aligned}$$

after a change of variables to $f = gh'^{-1}$ and then using that $F_\tau^{a,b}$ is independent of a for a direct triangle operator. This allows us to recognise F_ξ using (4.15). Denoting gg' as g, it follows that $\mathcal{L}(s_0, s_1)$ is spanned by $\{F_\xi^{h,g}|\text{vac}\rangle \mid h, g \in G\}$ as required.

A similar argument applies for the dual distance 3 case, $\xi = \tau^* \circ \xi'$. Given for example



we have this time $\mathcal{L}(s_0, s_1)$ is a subspace of the space spanned by $\{L_{\tau^*}^{h'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle \mid g, h', h \in G\}$ and such that $B(p) = \delta_e \triangleright_{s_2}$ acts as the identity, where p is the face connecting ξ' and τ^* so that $p \in s_2$. Then

$$\begin{aligned} L_{\tau^*}^{h'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle &= \delta_e \triangleright_{s_2} L_{\tau^*}^{h'} \circ F_{\xi'}^{h,g}|\text{vac}\rangle = L_{\tau^*}^{h'} \delta_{h'^{-1}} \triangleright \circ F_{\xi'}^{h,g}|\text{vac}\rangle \\ &= L_{\tau^*}^{h'} \circ F_{\xi'}^{h,g} \delta_{h'^{-1}g^{-1}hg} \triangleright_{s_2}|\text{vac}\rangle \end{aligned}$$

which only holds if $h' = g^{-1}hg$, so for elements of $\mathcal{L}(s_0, s_1)$ we need only consider

$$\begin{aligned} L_{\tau^*}^{g^{-1}hg} \circ F_{\xi'}^{h,g}|\text{vac}\rangle &= F_{\tau^*}^{g^{-1}hg,e} \circ F_{\xi'}^{h,g}|\text{vac}\rangle = \sum_f \delta_{f,g} F_{\tau^*}^{f^{-1}hf,e} \circ F_{\xi'}^{h,g}|\text{vac}\rangle \\ &= \sum_f F_{\tau^*}^{f^{-1}hf,f^{-1}g} \circ F_{\xi'}^{h,g}|\text{vac}\rangle = F_\xi^{h,g}|\text{vac}\rangle \end{aligned}$$

on noting that $F_{\tau^*}^{f^{-1}hf,f^{-1}g} = \delta_{e,f^{-1}g} F_{\tau^*}^{f^{-1}hf,e}$ and using (4.15). Thus, $\mathcal{L}(s_0, s_1)$ is spanned by $\{F_\xi^{h,g}|\text{vac}\rangle \mid h, g \in G\}$ as required.

14 Universal Quantum Computation with $D(S_3)$

Here, we outline and comment on further aspects of the logical qubit within $D(S_3)$ in [WLP09]. First, we describe a Z-basis measurement on the logical qubit. It is claimed in [BAC09, LHG11] that there exist 'transport' operations M^τ_ξ which move τ quasiparticles along the lattice deterministically. In particular, these should exist such that

$$M^\tau_{-\xi'}W^\tau_\xi|\text{vac}\rangle = W^\tau_{\xi'\circ\xi}|\text{vac}\rangle$$

for all composeable open ribbons ξ, ξ'. It is beyond our scope to construct $M^\tau_{-\xi'}$ here, but assuming it exists, it is a linear combination of chargeon ribbons, and therefore satisfies (4.17). Taking $-\xi$ to be a ribbon that completes ξ to a closed contractible ribbon, we have

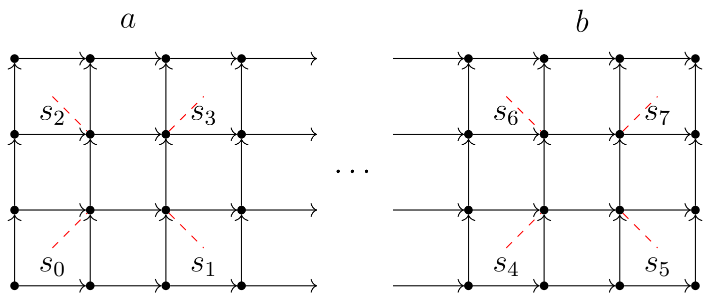
$$M^\tau_{-\xi}W^\tau_\xi|\text{vac}\rangle = W^\tau_{(-\xi)\circ\xi}|\text{vac}\rangle = |\text{vac}\rangle$$

Hence, referring to ξ, ξ', ξ'' in Section 4.2.5, we have that applying $M^\tau_{-\xi}M^\tau_{-\xi'}$ to $|0_L\rangle$ and measuring the projector $P_{e,1}$ at any s_i will always yield $|\text{vac}\rangle$. On the other hand, if we begin with $|1_L\rangle$, we have

$$\begin{aligned} M^\tau_{-\xi}M^\tau_{-\xi'}|1_L\rangle &= M^\tau_{-\xi}M^\tau_{-\xi'}W^\sigma_{\xi''}W^\tau_{\xi'}W^\tau_\xi|\text{vac}\rangle \\ &= W^\sigma_{\xi''}M^\tau_{-\xi}M^\tau_{-\xi'}W^\tau_{\xi'}W^\tau_\xi|\text{vac}\rangle \\ &= W^\sigma_{\xi''}|\text{vac}\rangle \end{aligned}$$

by (4.17), and so applying $P_{e,1}$ at s_0, s_1 will return 0. The operation $M^\tau_{-\xi}M^\tau_{-\xi'}$ followed by measuring $P_{e,1}\triangleright_{s_0}$, say, therefore constitutes a destructive Z-basis measurement on the logical qubit: it tells us whether the qubit was in state $|0_L\rangle$ or $|1_L\rangle$, but at the cost of taking us out of the degenerate subspace.

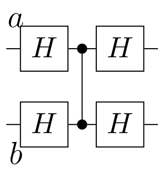
Now consider two distant groups of 4 τ quasiparticles labelled a, b:

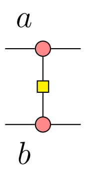


where group a is as before and group b is a parallel copy with parallel notation. Entanglement between a, b is achieved with the gate

$$K_{a,b} := \frac{1}{2}(\text{id}_a \otimes \text{id}_b + X_a \otimes \text{id}_b + \text{id}_a \otimes X_b - X_a \otimes X_b)$$

where X_a, X_b are the logical operators on the respective qubits. $K_{a,b}$ has the following representations as a quantum circuit and a ZX-diagram respectively:



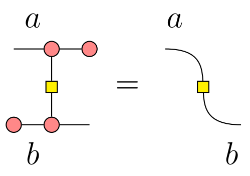


In terms of ribbon operators, this is:

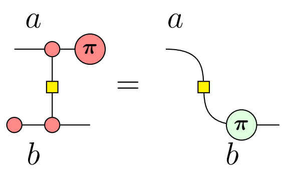
$$K_{a,b} = \frac{1}{2}(\mathrm{id}_a \otimes \mathrm{id}_b + W^{\sigma}_{\xi''_a} \otimes \mathrm{id}_b + \mathrm{id}_a \otimes W^{\sigma}_{\xi''_b} - W^{\sigma}_{\xi''_a} \otimes W^{\sigma}_{\xi''_b})$$

by straighforward substitution. Note that, while $K_{a,b}$ is an entangling operation between the two logical qubits, it only acts along ribbons ξ''_a, ξ''_b, and doesn't require ribbons between the two qubits, and must rely on the large entangled state on the lattice to transmit information. As $K_{a,b}$ requires only the ribbons ξ''_a, ξ''_b, it keeps the state within the combined degenerate subspace where there are τ quasiparticles at all sites $s_0, s_1, \cdots, s_7$.

A logical Hadamard can be performed non-deterministically on qubit a using an ancillary qubit. We initialise the ancilla with $|0_b\rangle$, apply $K_{a,b}$ and then perform a Z-basis measurement on qubit a. This teleports the state $|\psi\rangle$ on qubit a to $H_b|\psi\rangle$ on qubit b, with a possible additional Z_b factor depending on the measurement outcome. This is obvious from a short calculation with the ZX-calculus [CD11]. Consider branch 1, where the measurement results in $\langle 0_a|$:



and branch 2, where the measurement gives $\langle 1_a|$:

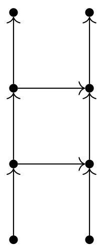


If we reach branch 2, the process is repeated until the Hadamard alone is implemented (this is quite inefficient).

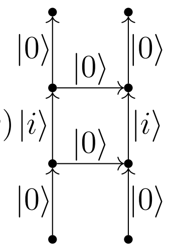
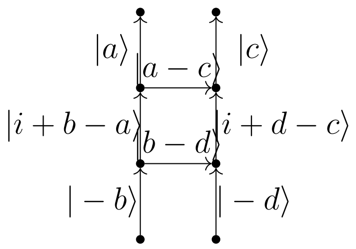
Equipped with the logical Hadamard and X rotations, we can reach anywhere on the Bloch sphere, and the addition of the entangling gate $K_{a,b}$ allows the implementation of any unitary [NC10, Sec 4.5.2]. We note that several other schemes for universal computation using representations of $D(S_3)$ have been described in [CHW15], although the formulation is categorical rather than in terms of the quantum double on a lattice. We do not know whether these categorical schemes can be implemented on the lattice.

15 Fourier basis for patches

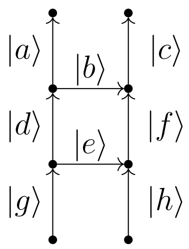
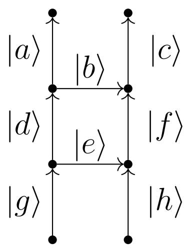
Consider the small patch

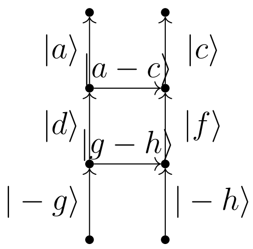


Now, $|i\rangle_L$ is the following state:

$\prod_v A(v)$ 
$= \sum_{a,b,c,d}$ 

where we have taken $|0\rangle_L$ and applied an X-type string from left to right. Now, consider $|\delta_0\rangle_L$:

$\prod_p B(p) \sum_{a,b,c,d,e,f,g,h}$ 
$= \sum_{a,b,c,d,e,f,g,h} \delta_0(a-c-b)\, \delta_0(d+b-f-e)\, \delta_0(g+e-h)$ 

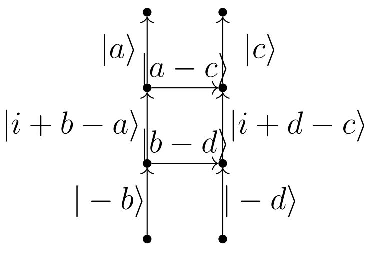
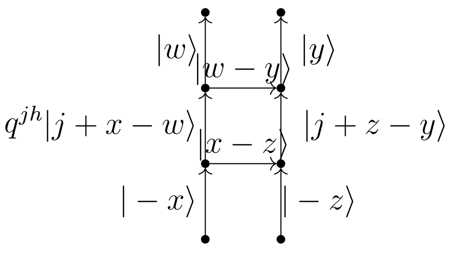
$= \sum_{a,c,d,f,g,h} \delta_0(d+a-c-f-g+h)$ 

where we performed a change of variables $g \mapsto -g$, $h \mapsto -h$. Now, $\delta_0(d+a-c-f-g+h)$ holds iff $d+a-g=i$ and $-f-c+h=-i$ for some $i \in \mathbb{Z}_d$. Thus we have $|\delta_0\rangle_L = \sum_i |i\rangle_L$. If we then apply a Z-type string operator from top to bottom in the quasiparticle basis we see that $|\delta_j\rangle_L = \sum_i q^{-ij}|i\rangle_L$.

One could then show that the bases are consistent under Fourier transform for all sizes of patch by induction, using the above as the base case.

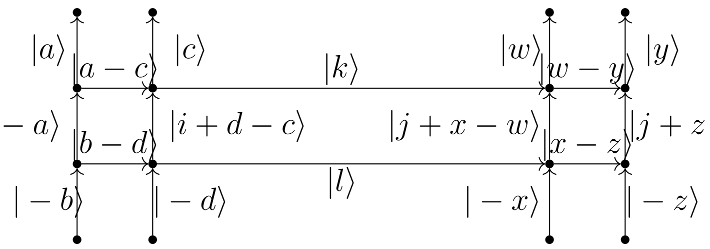
16 Proof of lattice merges

We demonstrate the smooth merge on a small patch but it is easy to see that the same method applies for arbitrary large patches. We begin with two patches, in the $|\delta_g\rangle_L$ and $|\delta_h\rangle_L$ states respectively.

$\sum_{a,b,c,d,i} q^{ig}$ 
$\sum_{w,x,y,z,j} q^{jh}$ 

Then initialise two new edges between, each in the $|\delta_0\rangle$ state.

$\sum_{a,b,c,d,i,j,k,l,w,x,y,z} q^{ig+jh}$



where we have exaggerated the length of the new edges for emphasis. Now if we apply stabiliser measurements at all points we see that the only relevant ones are the face measurements including the new edges (the vertex measurements will still yield $A(v)$ unless a physical error has appeared there). The relevant measurements give us

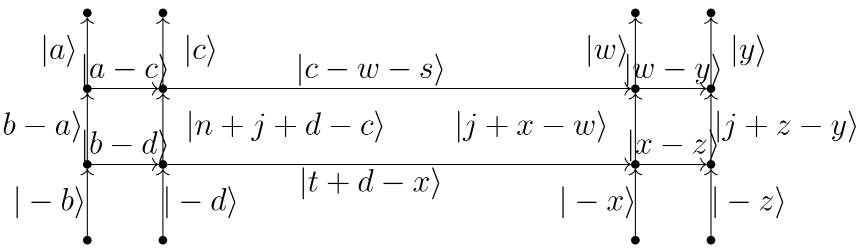
$$\delta_s(c-w-k); \quad \delta_r(k+i+d-c-j-x+w-l); \quad \delta_t(-d+l+x)$$

for each new face, where $r, s, t \in \mathbb{Z}_d$. By substitution this gives

$$\delta_r(k+i+d-c-j-x+w-l) = \delta_r(-t-s+i-j) = \delta_{r+t+s}(i-j) = \delta_n(i-j) = \delta_i(n+j)$$

where n is the group product of r, t, s in $\mathbb{Z}_d$. Computationally, n is the important *measurement outcome* of the merge. Plugging back in to the patches we have

$\sum_{a,b,c,d,j,w,x,y,z} q^{ng+j(g+h)}$



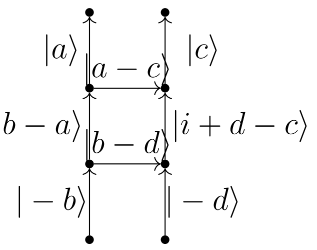
In the positive outcome case, i.e. when $s = r = t = 0$, it is immediate that we have $|\delta_{g+h}\rangle_L$ on the combined patch. Otherwise, we can 'fix' the internal additions of s, t, n to the edges with string operators or alternatively accommodate them into the Pauli frame in the same manner as described in e.g. [dBH20]. Then we are left with $q^{ng}|\delta_{g+h}\rangle_L$, as stated.

The Fourier transformed version of the above explains the rough merges as well, so we do not describe it explicitly.

17 Proof of lattice counits

We now show a 'smooth counit' on a patch with state $|\delta_j\rangle_L$:

$\sum_{a,b,c,d,i} q^{ig}$



Measure out all edges in the Z basis, giving

$$\sum_{a,b,c,d,i} q^{ij}\delta_r(a)\delta_s(a-c)\delta_t(c)\delta_u(i+b-a)\delta_v(b-d)\delta_w(i+d-c)\delta_x(-b)\delta_y(-d)$$

for some $r, \cdots, y \in \mathbb{Z}_d$. Then we observe that $\delta_u(i+b-a) = \delta_i(a-b-u) = \delta_i(n)$ for $n = a-b-u$, and by performing some other substitutions we arrive at

$$q^{nj}\delta_v(y-x)\delta_w(n-y-t)\delta_s(n-u-x-t)$$

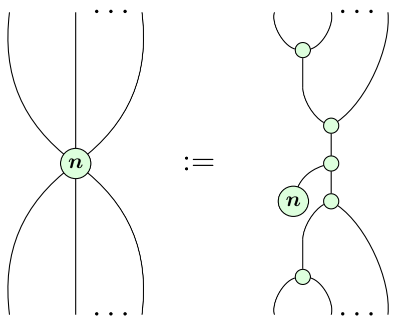
Importantly, the only factor here which depends on the input state is q^{nj}. All the δ-functions are merely conditions regarding which measurement outcomes are possible due to the lattice geometry. These will always be satisfied by our measurements, thus we have just

$$|\delta_j\rangle_L \mapsto q^{nj}$$

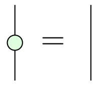
for $n \in \mathbb{Z}_d$, which in the other basis is $|i\rangle_L \mapsto \delta_{n,i}$ as stated. The rough counit follows similarly.

18 Qudit ZX-calculus axioms

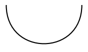
We show some relevant axioms for the fragment of qudit ZX-calculus which interests us. These simply coincide with the rules from Hopf and Frobenius structures, along with the Fourier transform. We ignore the more general phase group [Wan21], and also leave out non-zero scalars. First, we define a spider

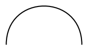


which is well-defined due to associativity and specialty of the underlying Frobenius structure. The spider is also invariant under exchange of input wires with each other and the same for outputs, as the Frobenius algebra is (co)-commutative. A phaseless spider with 1 input and 1 output is identity:

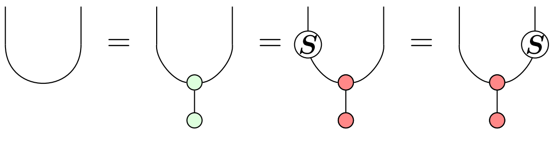


Now, we can define duality morphisms on the object $\mathbb{C}\mathbb{Z}_d$, which we call a ‘cup’ and similarly a ‘cap’:

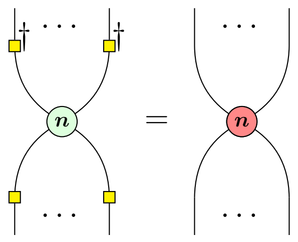
 $\rightsquigarrow \quad \sum_{i\in\mathbb{Z}_d} |i\rangle \otimes |i\rangle$

 $\rightsquigarrow \quad |i\rangle \otimes |j\rangle \mapsto \delta_{i,j}$

which correspond to:

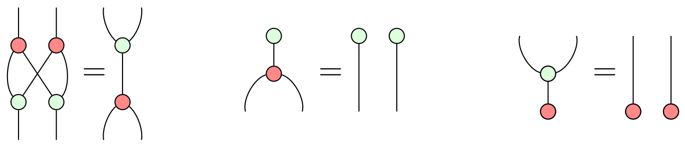


for the cup, and the vertically flipped version for the cap. The antipodes included here are responsible for the antipodes in the CX gate in Section 5.2.1. Then we have the Fourier exchange rule:

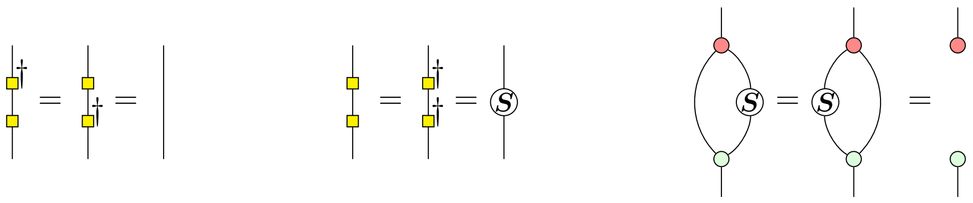


which encodes Lemma 5.0.3 graphically.

Then we have the bialgebra rules



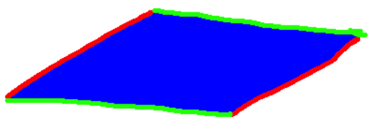
and rules pertaining to the antipode:



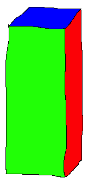
This is far from an exhaustive set of rules.

19 The logical block depiction

The lattice at a given time is drawn with a red line for a smooth boundary and green for a rough boundary:



where the surface is shaded blue for clarity. A block extending upwards represents the transformation over time. For example:



We call this the 'logical block' depiction, following similar work in [BDMNPR21].

Table 1 is an explicit dictionary between lattice surgery operations, qudit ZX-calculus and linear maps in the multiplicative fragment, i.e. the $n = 0$ measurement outcomes. We

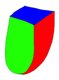
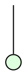
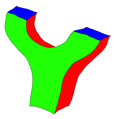
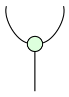
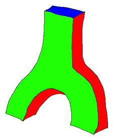
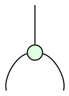
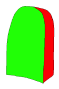
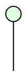
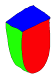
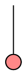
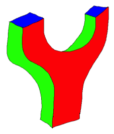
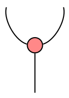
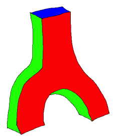
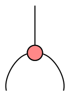
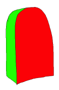
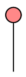
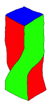
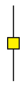
Lattice operation	Logical block	ZX-diagram	Linear map
smooth unit			$\sum_i \lvert i\rangle$
smooth split			$\lvert i\rangle \mapsto \lvert i\rangle \otimes \lvert i\rangle$
smooth merge			$\lvert i\rangle \otimes \lvert j\rangle \mapsto \delta_{i,j}\lvert i\rangle$
smooth counit			$\lvert i\rangle \mapsto 1$
rough unit			$\lvert 0\rangle$
rough split			$\lvert i\rangle \mapsto \sum_h \lvert h\rangle \otimes \lvert i-h\rangle$
rough merge			$\lvert i\rangle \otimes \lvert j\rangle \mapsto \lvert i+j\rangle$
rough counit			$\lvert i\rangle \mapsto \delta_{i,0}$
rotation			$\lvert i\rangle \mapsto \sum_j q^{-ij}\lvert j\rangle$

Table 1: Dictionary of lattice surgery operations in the multiplicative fragment.

choose to use the multiplicative fragment to highlight the visual connection between the columns. We see that red and green spiders correspond to rough and smooth operations respectively.

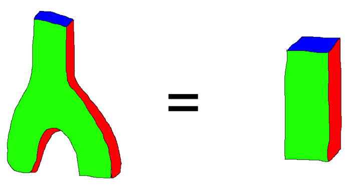
We have no new results or proofs in this section, but we would like to discuss the diagrams of logical blocks. These sorts of diagrams for lattice surgery have been used in an engineering setting to compile quantum circuits to lattice surgery [GF09, BDMNPR21]. To go from the cubes shown there to the tubes which we show here we merely relax the discretisation of space and time somewhat to expose the relationship with algebra. This relationship with algebra is relevant because such diagrams have appeared in a seemingly quite different context.

It is well known that the category of '2-dimensional thick tangles', **2Thick**, is monoidally equivalent to the category **Frob** freely generated by a noncommutative Frobenius algebra [Lau05]. This should be unsurprising to those familiar with the notion of a 'pair of pants' algebra. We say that **2Thick** is a *presentation* of **Frob**. Similarly, the symmetric monoidal category **2Cob** of (diffeomorphism classes of) 2-dimensional cobordisms between (disjoint unions of) circles is a presentation of **ComFrob**, the category freely generated by a commutative Frobenius algebra [Koc03].

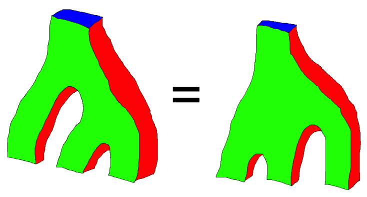
This fact is important for topological quantum field theories (TQFTs). One can define an n-dimensional TQFT as a symmetric monoidal functor from $\mathbf{nCob} \to \mathbf{Vect}$, the category of finite-dimensional vector spaces. The key point is that the functor takes (diffeomorphism classes of) manifolds as inputs and outputs linear maps between vector spaces, which are by definition manifold invariants. One can see that 2D TQFTs are in bijection with commutative Frobenius algebras in **Vect**.

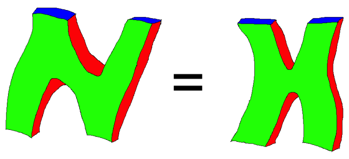
In [Reu19], Reutter gives a slightly different monoidal category, which we will call **2Block**. It has as objects disjoint unions of squares, with the same shading of sides as those in the logical block diagrams above. Then morphisms are classes of surfaces between the squares, such that the borders between the surfaces match up with the edges of the squares at the source and target objects and the surface colours are consistent with those of the squares' sides. While the morphisms are obviously quotiented by equivalence of surfaces up to border-preserving diffeomorphism, Reutter quotients by 'saddle-invertibility' as well, which is not a rule one can acquire through topological moves alone, as it involves the closing and opening of holes.

Reutter conjectures that $\mathbf{2Block} \simeq \mathbf{uHopf}$, where **uHopf** is the category freely generated by a unimodular Hopf algebra.[2] While we do not know enough about topology or geometry to prove (or disprove) this conjecture, we suspect one route is to consider Morse functions and classify the diffeomorphism classes near critical points. This is similar to one proof of $\mathbf{2Cob} \simeq \mathbf{ComFrob}$ [Koc03]. For the reader's convenience, we now reproduce a handful of the equivalences under topological deformation which motivate this conjecture. We have the axioms of a Frobenius algebra,

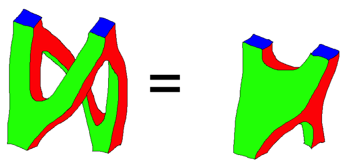


[2]In **Vect**, unimodularity is typically defined using integrals [Maj95]. In this more abstract setting it is defined by some axioms on dualities.





and the same for red faces. These are just widened versions of the diagrams in **2Thick**. Then one can see the interpretation of a unimodular Hopf algebra as two interacting Frobenius algebras. We start with two Frobenius algebras and glue them together in such a way that they give the bialgebra and antipode axioms. The main bialgebra rule is



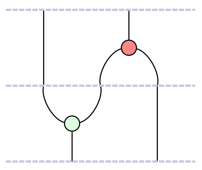
where we require saddle invertibility to close up a hole in the middle. This is also required for showing that comultiplication is a unit map and so on. Given all of these deformations and those involving the antipode, which is a twist by π, one can see that they define a functor **uHopf** $\to$ **2Block**; the hard part is proving that this is an equivalence.

Now, Reutter also draws a comparison with representation theory and tensor category theory. It is striking that, given the unimodular Hopf algebra $\mathbb{C}\mathbb{Z}_d$, we can create a logical space on a patch isomorphic to the vector space of $\mathbb{C}\mathbb{Z}_d$ itself, and the logical operations precisely coincide with the linear maps defined by the algebra. We conjecture that lattice surgery is the 'computational implementation' of this presentation of unimodular Hopf algebras, in the same way that the logical space of the Kitaev model on a closed orientable manifold $\mathcal{M}$ is isomorphic to the vector space $F(\mathcal{M})$ in the image of a Dijkgraaf-Witten theory F : **2Cob** $\to$ **Vect** when given the same manifold $\mathcal{M}$ [CM22, Thm 3.2].

20 Logical CX gate

Here we check the correctness of the CX gate implementations from Section 5.2.1.

First, observe that the diagram:

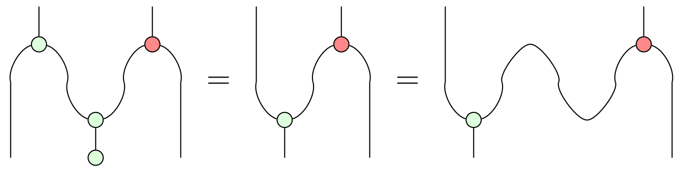


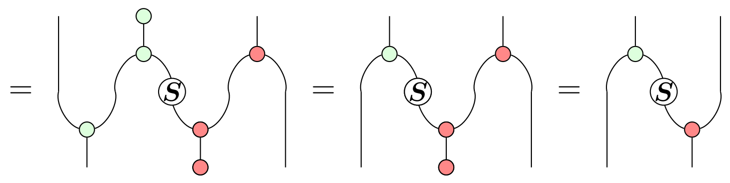
yields the linear map:

$$|i\rangle \otimes |j\rangle \mapsto |i\rangle \otimes |i\rangle \otimes |j\rangle \mapsto |i\rangle \otimes |i+j\rangle$$

where we have considered the diagram piecemeal from bottom to top, indicated by the dashed lines.

Then we can perform a sequence of rewrites between all four diagrams, labelled below:





where at each stage we have either used the spider rule, inserted duals, or swapped between duals and spiders; see Appendix 18.

21 Generalisations and Hopf algebras

While we have shown that lattice surgery works for arbitrary dimensional qudits, we emphasise that the algebraic structures involved are very simple so far. The lattice model in the bulk can be generalised significantly: first, one can replace $\mathbb{C}\mathbb{Z}_d$ with another finite abelian group algebra. As all finite abelian groups decompose into direct sums of cyclic groups this case follows immediately from the work herein and is uninteresting.

At the second level up, we can replace it with an arbitrary finite group algebra $\mathbb{C}G$. At this level several assumptions break down:

- $\mathbb{C}G$ still has a dual function algebra $\mathbb{C}(G)$, but the Fourier transform no longer coincides with Pontryagin duality, and the two algebras will no longer be isomorphic in general. One can still define a Fourier transform in the sense that it translates between convolution and multiplication, but in this case the Fourier transform is the Peter-Weyl isomorphism, i.e. a bimodule isomorphism between $\mathbb{C}G$ and a direct sum of matrix algebras labelled by the irreps of G.

- The $\mathbb{C}G$ lattice model can no longer be described using string operators, and these must be promoted to ribbon operators [Kit03]. This is because the lattice model is based on the Drinfeld double $D(G) = \mathbb{C}(G) \rtimes \mathbb{C}G$, where the associated action is conjugation. In the abelian case conjugation acts trivially and so we have $D(\mathbb{Z}_d) = \mathbb{C}(\mathbb{Z}_d) \otimes \mathbb{C}\mathbb{Z}_d$: the double splits into independent algebras, which give the X-type and Z-type string operators respectively.

- There are still canonical choices of rough and smooth boundary, labelled by subgroups $K = \{e\}$ and $K = G$ for rough and smooth boundaries respectively. Similarly, we still have well-defined measurements, using representations of CG and $C(G)$ for vertices and faces. However, the algebra of ribbon operators which are undetectable at the boundary, and hence the logical operations on a patch, becomes significantly more complicated, see [Sch16] for the underlying module theory.

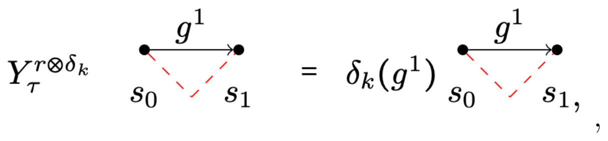
Of course, the Kitaev model can be generalised much further still. The third level would be arbitrary finite-dimensional Hopf $\mathbb{C}^*$-algebras. At this level even the calculations in the bulk are tricky, and many features were only recently resolved [CM22, Meu17, YCC22].

The fourth (and highest) level is the maximal generality, which are weak Hopf $\mathbb{C}^*$-algebras, in bijection (up to an equivalence) with so-called *unitary fusion categories* [EGNO10]. Even at this extreme generality, there are glimpses of hope. There are two canonical choices of boundaries given by the trivial (rough) and regular (smooth) module categories [Ost03B], and we speculate that calculating some basic features like $\dim(\mathcal{H}_{vac})$ of a patch could be done using techniques from topological quantum field theory (TQFT). At this level of generality, the connections with TQFT become more tantalising. The parallels between topological quantum computing in the bulk and TQFTs are well-known, see e.g. [BK12], but lattice surgery introduces discontinuous deformations in the manner of geometric surgery. While boundaries of TQFTs are well-studied [KS11, FS03], we do not know whether TQFT theorists study the relation between geometric surgery on manifolds and linear algebra in the same manner as they do for, say, diffeomorphism classes of cobordisms.

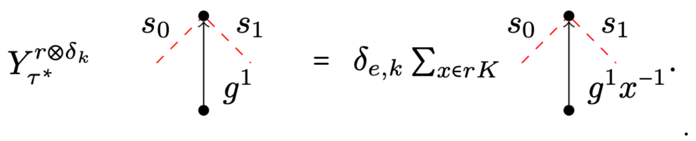
22 Boundary ribbon operators with $\Xi(R,K)^\star$

In [CCW16, Sec 2.3.3] it is claimed that one can use boundary ribbon operators built from Ξ to create quasiparticles on the boundary, in a similar manner to the bulk ribbon operators, and that it commutes with their $A^K(v)$ and $B^K(p)$ terms at intermediate sites. In this appendix, we show that this does not work due to issues with equivariance, at least when the boundary ribbon operators act in the same way as bulk ribbon operators.

Definition 22.1. Let ξ be a ribbon, $r \in R$ and $k \in K$. Then $Y^{r\otimes\delta_k}_\xi$ acts on a direct triangle τ as



and on a dual triangle τ^* as



Concatenation of ribbons is given by

$$Y^{r\otimes\delta_k}_{\xi'\circ\xi} = Y^{(r\otimes\delta_k)_2}_{\xi'} \circ Y^{(r\otimes\delta_k)_1}_{\xi} = \sum_{x\in K} Y^{(x^{-1}\triangleright r)\otimes\delta_{x^{-1}k}}_{\xi'} \circ Y^{r\otimes\delta_x}_{\xi},$$

where we see the comultiplication $\Delta(r\otimes\delta_k)$ of $\Xi(R,K)^*$. Here, $\Xi(R,K)^*$ is a coquasi-Hopf algebra, and so has coassociative comultiplication (it is the multiplication which is only quasi-associative). Therefore, we can concatenate the triangles making up the ribbon in any order, and the concatenation above uniquely defines $Y^{r\otimes\delta_k}_\xi$ for any ribbon ξ.

Let $s_0 = (v_0, p_0)$ and $s_1 = (v_1, p_1)$ be the sites at the start and end of a triangle. The direct triangle operators satisfy

$$k'\triangleright_{v_0} \circ Y^{r\otimes\delta_k}_\tau = Y^{r\otimes\delta_{k'k}}_\tau \circ k'\triangleright_{v_0}, \quad k'\triangleright_{v_1} \circ Y^{r\otimes\delta_k}_\tau = Y^{r\otimes\delta_{kk'^{-1}}}_\tau \circ k'\triangleright_{v_1}$$

and

$$[\delta_{r'}\triangleright_{s_i}, Y^{r\otimes\delta_k}_{\tau}] = 0$$

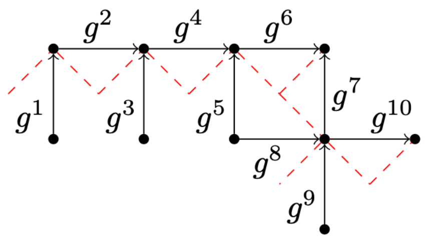
for $i \in \{1,2\}$. For the dual triangle operators, we have

$$k'\triangleright_{v_i} \circ \sum_k Y^{r\otimes\delta_k}_{\tau^*} = Y^{(k'\triangleright r)\otimes\delta_k}_{\tau^*} \circ k'\triangleright_{v_i}$$

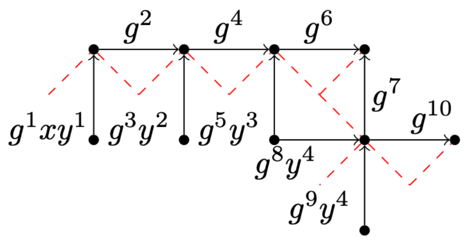
again for $i \in \{1,2\}$ and $k \in \mathbb{C}K$. However, there are not similar commutation relations for the actions of $\mathbb{C}(R)$ on faces of dual triangle operators. In addition, in the bulk, one can reconstruct the vertex and face actions using suitable ribbons [BM-D08, CM22] because of the duality between $\mathbb{C}(G)$ and $\mathbb{C}G$; this is not true in general for $\mathbb{C}(R)$ and $\mathbb{C}K$.

Example 22.2. Given the ribbon ξ on the lattice below, we see that $Y^{r\otimes\delta_k}_{\xi}$ acts only along the ribbon and trivially elsewhere. We have

$Y^{r\otimes\delta_k}_{\xi}$



$= \delta_k(g^2g^4g^6(g^7)^{-1}g^{10}) \sum_{x^1,x^2,x^3,x^4\in K}$



if $g^2, g^4, g^6(g^7)^{-1}, g^{10} \in K$, and 0 otherwise, and

$$\begin{aligned} y^1 &= (rx^1)^{-1} \\ y^2 &= ((g^2)^{-1}rx^2)^{-1} \\ y^3 &= ((g^2g^4)^{-1}rx^3)^{-1} \\ y^4 &= ((g^2g^4g^6(g^7)^{-1})^{-1}rx^4)^{-1} \end{aligned}$$

One can check this using Definition 22.1.

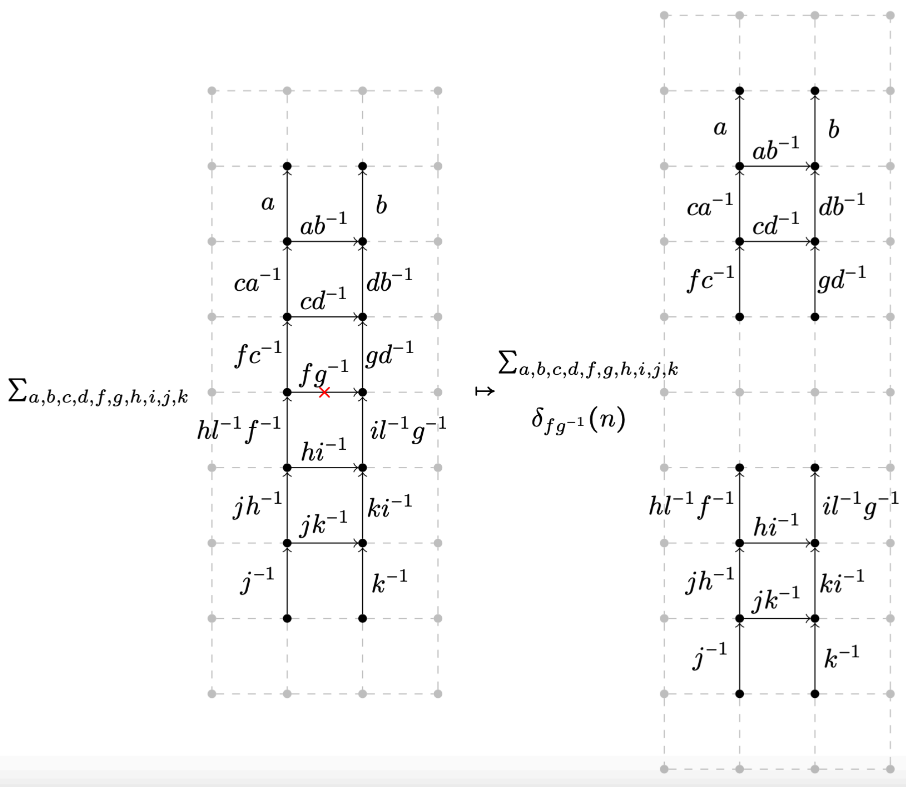
It is claimed in [CCW16, Sec 2.3.3] that these ribbon operators obey similar equivariance properties with the site actions of $\Xi(R,K)$ as the bulk ribbon operators, but such equivariance properties do not generally hold. Precisely, we find that when such ribbons are 'open' in the sense of [Kit03, BM-D08, CM22] then an intermediate site s_2 on a ribbon ξ between either endpoints s_0, s_1 does *not* satisfy

$$\Lambda_{\mathbb{C}K}\triangleright_{s_2} \circ Y^{r\otimes\delta_k}_{\xi} = Y^{r\otimes\delta_k}_{\xi} \circ \Lambda_{\mathbb{C}K}\triangleright_{s_2}.$$

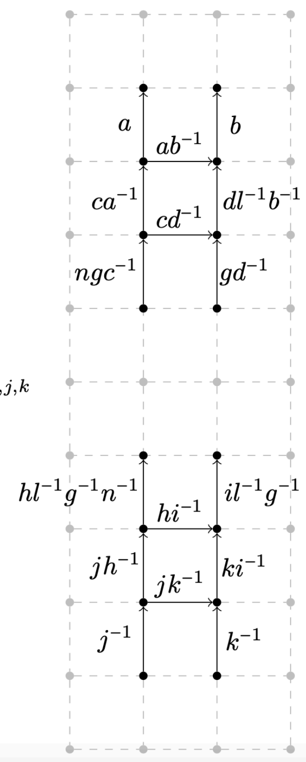
in general, nor the corresponding relation for $\Lambda_{\mathbb{C}(R)}\triangleright_{s_2}$. For example, consider the vertex between edges labelled g^2 and g^4 in Example 22.2 above - the equivariance property is not satisfied.

23 Measurements and nonabelian lattice surgery

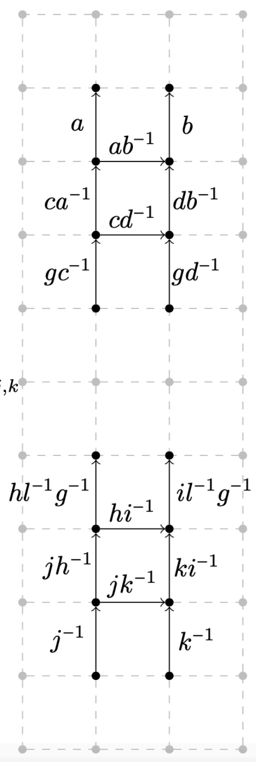
In Section 6.3.1, we described nonabelian lattice surgery for a general underlying group algebra $\mathbb{C}G$, but for simplicity of exposition we assumed that the projectors $A(v)$ and $B(p)$ could be applied deterministically. In practice, we can only make a measurement, which will only sometimes yield the desired projectors. As the splits are easier, we discuss how to handle these first, beginning with the rough split. We demonstrate on the same example as previously:



$\cdots = \sum_{a,b,c,d,g,h,i,j,k}$



$\rightsquigarrow \sum_{a,b,c,d,g,h,i,j,k}$



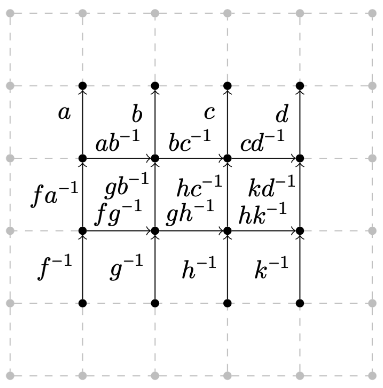
where we have measured the edge to be deleted in the $\mathbb{C}G$ basis. The measurement outcome n informs which corrections to make. The last arrow implies corrections made using ribbon operators. These corrections are all unitary, and if the measurement outcome is e then no corrections are required at all. The generalisation to larger patches is straightforward, but requires keeping track of multiple different outcomes.

Next, we discuss how to handle the smooth split. In this case, we measure the edges to be deleted in the Fourier basis, that is we measure the self-adjoint operator $\sum_\pi p_\pi P_\pi\triangleright$ at a particular edge, where

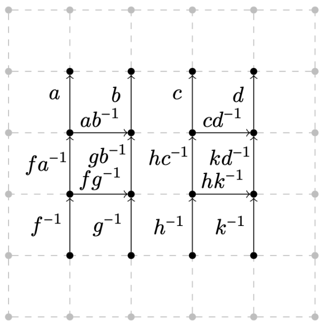
$$P_\pi := P_{e,\pi} = \frac{\dim(W_\pi)}{|G|}\sum_{g\in G}\mathrm{Tr}_\pi(g^{-1})g$$

from Section 6.1.2 acts by the left regular representation. Thus, for a smooth split, we have the initial state $|e\rangle_L$:

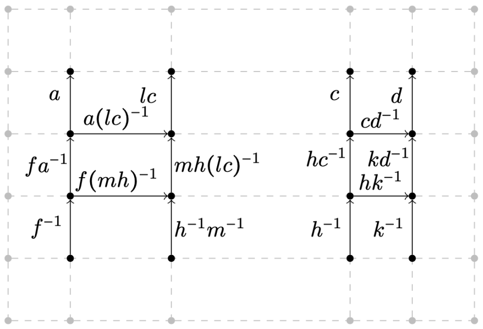
$\sum_{a,b,c,d,f,g,h,k}$



$\sum_{a,b,c,d,f,g,h,k,l,m} \mathrm{Tr}_\pi(l^{-1})\mathrm{Tr}_{\pi'}(m^{-1})\delta_{bc^{-1}}(l)\delta_{gh^{-1}}(m)$



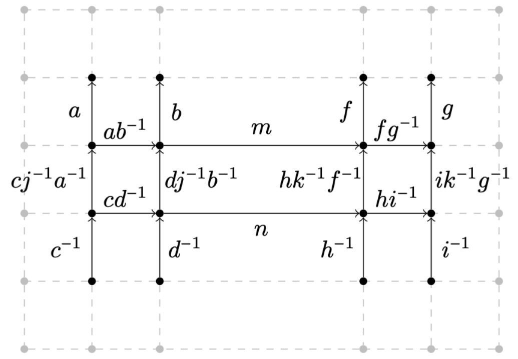
$= \sum_{a,c,d,f,h,k,l,m} \mathrm{Tr}_\pi(l^{-1})\mathrm{Tr}_{\pi'}(m^{-1})$



and afterwards we still have coefficients from the irreps of $\mathbb{C}G$. In the case when $\pi = 1$, we are done. Otherwise, we have detected quasiparticles of type (e, π) and (e, π') at two vertices. In this case, we appeal to e.g. [BKKK22, BAC09], which claim that one can modify these quasiparticles deterministically using ribbon operators and quantum circuitry. The procedure should be similar to initialising a fresh patch in the zero logical state, but we do not give any details ourselves. Then we have the desired result.

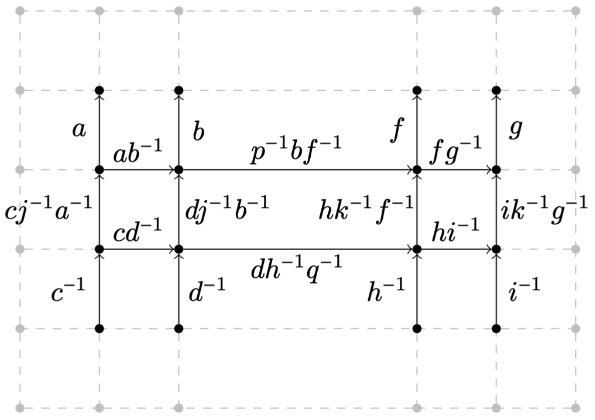
For merges, we start with a smooth merge, as again all outcomes are in the group basis. Recall that after generating fresh copies of $\mathbb{C}G$ in the states $\sum_{m\in G} m$, we have

$\sum_{a,b,c,d,f,g,h,i,m,n}$



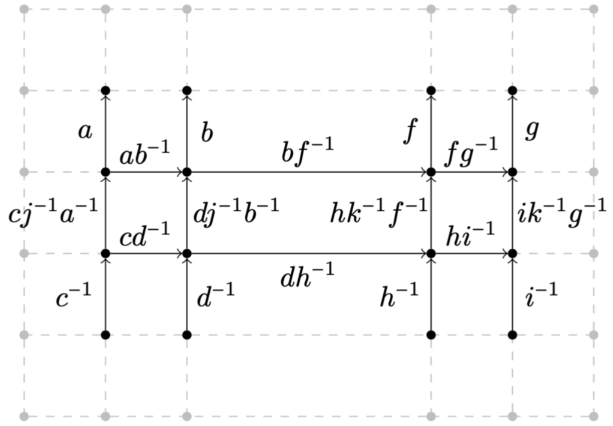
we then measure at sites which include the top and bottom faces, giving:

$\sum_{a,b,c,d,f,g,h,i} \sum_{p\in\mathcal{C}} \sum_{q\in\mathcal{C}'}$



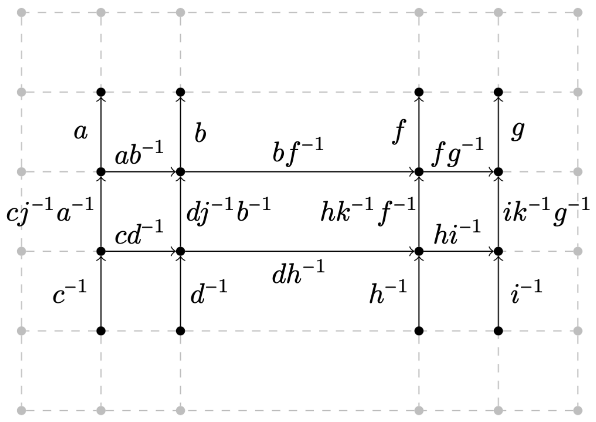
for some conjugacy classes $\mathcal{C}, \mathcal{C}'$. There are no factors of π as the edges around each vertex already satisfy $A(v)|\psi\rangle = |\psi\rangle$. When $\mathcal{C} = \mathcal{C}' = \{e\}$, we may proceed, but otherwise we require a way of deterministically eliminating the quasiparticles detected at the top and bottom faces. Appealing to e.g. [BKKK22, BAC09] as earlier, we assume that this may be done, but do not give details. Alternatively one could try to 'switch reference frames' in the manner of Pauli frames with qubit codes [HFDM12], and redefine the Hamiltonian. The former method gives

$\sum_{a,b,c,d,f,g,h,i}$



Lastly, we measure the inner face, yielding

$\sum_{a,b,c,d,f,g,h,i} \sum_{s \in \mathcal{C}''} \delta_s(kj^{-1})$



so $|j\rangle_L \otimes |k\rangle_L \mapsto \sum_{s \in \mathcal{C}''} \delta_{js,k} |js\rangle_L$, which is a direct generalisation of the result for when $G = \mathbb{Z}_n$ in [Cow22], where now we sum over the conjugacy class $\mathcal{C}''$ which in the $\mathbb{Z}_n$ case are all singletons.

The rough merge works similarly, where instead of having quasiparticles of type $(\mathcal{C}, 1)$ appearing at faces, we have quasiparticles of type (e, π) at vertices.

24 $\Xi(R, K)$ as a $*$-quasi-Hopf algebra

Although we have seen in Lemma 6.4.7 that $\Xi(R, K)$ has a $*$-algebra that commutes with the coalgebra, this is a very special feature and not something one can impose as a general axiom for a $*$-quasi-Hopf algebra. This is because when there is a nontrivial associator ϕ then coassociativity holds only up to conjugation and hence the properties of $*$ will normally also need to be modified up to a conjugation, i.e. hold in a weak sense. The correct notion of a $*$-quasi-Hopf algebra H, like the quasi-coassociativity axiom comes from the monoidal category structure, now equipped with a functorial complex conjugation as a bar category [BM09, Def. 3.16]. Note that the usual notion of a † or $\mathbb{C}^*$-category in computer science captures the notion of adjoints, rather than conjugation, and does not describe the behaviour under tensor products well in our case, as the tensor product of representations is only associative up to a non-trivial isomorphism. This is quite a subtle point - on the other hand, a practical consequence is Proposition 6.4.9, for which this appendix therefore provides a proof.

The natural axioms here at least for a $*$-quasi-bialgebra, fixing a typo in [BM09, Def. 3.16], involve an additional map θ obeying the first three of:

1. an antilinear algebra map $\theta : H \to H$;
2. an invertible element $\gamma \in H$ such that $\theta(\gamma) = \gamma$ and $\theta^2 = \gamma(\ )\gamma^{-1}$;

3. an invertible element $\mathcal{G}\in H\otimes H$ such that

$$\Delta\theta=\mathcal{G}^{-1}(\theta\otimes\theta)(\Delta^{op}(\ ))\mathcal{G},\quad (\epsilon\otimes\mathrm{id})(\mathcal{G})=(\mathrm{id}\otimes\epsilon)(\mathcal{G})=1,\tag{1}$$

$$(\theta\otimes\theta\otimes\theta)(\phi_{321})(1\otimes\mathcal{G})((\mathrm{id}\otimes\Delta)\mathcal{G})\phi=(\mathcal{G}\otimes 1)((\Delta\otimes\mathrm{id})\mathcal{G}).\tag{2}$$

4. We say the $*$-quasi bialgebra is strong if

$$(\gamma\otimes\gamma)\Delta\gamma^{-1}=((\theta\otimes\theta)(\mathcal{G}_{21}))\mathcal{G}.\tag{3}$$

Next, if we have a quasi-Hopf algebra then S is antimultiplicative and hence $\theta=*S$ defines an antimultiplicative antilinear map $*$. However, S is not unique for a quasi-Hopf algebra and specifying θ directly is more canonical.

Lemma 24.1. Let $(\ )^R$ be bijective. Then Ξ has an antilinear algebra automorphism θ such that

$$\theta(x)=\sum_s x{\triangleleft}s\,\delta_{s^R},\quad \theta(\delta_s)=\delta_{s^R},$$

$$\theta^2=\gamma(\ )\gamma^{-1};\quad \gamma=\sum_s\tau(s,s^R)^{-1}\delta_s,\quad \theta(\gamma)=\gamma.$$

Proof. We compute,

$$\theta(\delta_s\delta_t)=\delta_{s,t}\delta_{s^R}=\delta_{s^R,t^R}\delta_{s^R}=\theta(\delta_s)\theta(\delta_t)$$

$$\theta(x)\theta(y)=\sum_{s,t}x{\triangleleft}s\delta_{s^R}y{\triangleleft}t\delta_{t^R}=\sum_t(x{\triangleleft}(y{\triangleright}t))(y{\triangleleft}t)\delta_{t^R}=\sum_t(xy){\triangleleft}t\delta_{t^R}=\theta(xy),$$

where imagining commuting δ_{t^R} to the left fixes $s^R=(y{\triangleleft}t){\triangleright}t^R=(y{\triangleright}t)^R$ to obtain the 2nd equation. We also have

$$\theta(x\delta_s)=\sum_t x{\triangleleft}t\delta_{t^R}\delta_{s^R}=x{\triangleleft}s\delta_{s^R}=\delta_{(x\triangleleft s)\triangleright s^R}x{\triangleleft}s=\delta_{(x\triangleright s)^R}x{\triangleleft}s$$

$$\theta(\delta_{x\triangleright s}x)=\sum_t\delta_{(x\triangleright s)^R}x{\triangleleft}t\delta_{t^R}=\sum_t\delta_{(x\triangleright s)^R}\delta_{(x\triangleleft t)\triangleright t^R}=\sum_t\delta_{(x\triangleright s)^R}\delta_{(x\triangleright t)^R}x{\triangleleft}t,$$

which is the same as it needs $t=s$. Next

$$\gamma^{-1}=\sum_s\tau(s,s^R)\delta_{s^{RR}}=\sum_s\delta_s\tau(s,s^R),$$

where we recall from previous calculations that $\tau(s,s^R){\triangleright}s^{RR}=s$. Then

$$\begin{aligned}\theta^2(x)&=\sum_s\theta(x{\triangleleft}s\delta_{s^R})=\sum_{s,t}(x{\triangleleft}s){\triangleleft}t\delta_{t^R}\delta_{s^R}=\sum_s(x{\triangleleft}s){\triangleleft}s\delta_{s^R}=\sum_s(x{\triangleleft}s){\triangleleft}x^R\delta_{s^{RR}}\\&=\sum_s\tau(x{\triangleright}s,(x{\triangleright}s)^R)^{-1}x\tau(s,s^R)\delta_{s^{RR}}=\sum_{s,t}\tau(t,t^R)^{-1}\delta_t x\tau(s,s^R)\delta_{s^{RR}}\\&=\sum_{s,t}\delta_{t^{RR}}\tau(t,t^R)^{-1}x\tau(s,s^R)\delta_{s^{RR}}=\gamma x\gamma^{-1}\end{aligned}$$

where for the 6th equality if we were to commute $\delta_{s^{RR}}$ to the left, this would fix $t = x\tau(s,s^R){\triangleright}s^{RR} = x{\triangleright}s$. We then use $\tau(t,t^R)^{-1}{\triangleright}t = t^{RR}$ and recognise the answer. We also check that

$$\gamma\delta_s\gamma^{-1} = \tau(s,s^R)^{-1}\delta_s\tau(s,s^R) = \delta_{s^{RR}} = \theta^2(\delta_s),$$
$$\theta(\gamma) = \sum_{s,t}\tau(s,s^R)^{-1}{\triangleleft}t\delta_{t^R}\delta_{s^R} = \sum_s\tau(s,s^R)^{-1}{\triangleleft}s\delta_{s^R} = \sum_s\tau(s^R,s^{RR})^{-1}\delta_{s^R} = \gamma$$

using Lemma 6.4.4. ∎

Next, we find $\mathcal{G}$ obeying the conditions above.

Lemma 24.2. If $(\ )^R$ is bijective then equations (1)-(3) hold for $\Xi(R,K)$ with

$$\mathcal{G} = \sum_{s,t}\delta_{t^R}\tau(s,t)^{-1}\otimes\delta_{s^R}\tau(t,t^R)(\tau(s,t){\triangleleft}t^R)^{-1},$$

$$\mathcal{G}^{-1} = \sum_{s,t}\tau(s,t)\delta_{t^R}\otimes(\tau(s,t){\triangleleft}t^R)\tau(t,t^R)^{-1}\delta_{s^R}.$$

Proof. The proof that $\mathcal{G},\mathcal{G}^{-1}$ are indeed inverse is straightforward on matching the δ-functions to fix the summation variables in $\mathcal{G}^{-1}$ in terms of $\mathcal{G}$. This then comes down to proving that the map $(s,t)\to(p,q) := (\tau(s,t){\triangleright}t^R, \tau'(s,t){\triangleright}s^R)$ is injective. Indeed, the map $(p,q)\mapsto(p,p\cdot q)$ is injective by left division, so it's enough to prove that

$$(s,t)\mapsto(p,p\cdot q) = (\tau(s,t){\triangleright}t^R, \tau(s,t){\triangleright}(t^R\cdot\tau(t,t^R)^{-1}{\triangleright}s^R)) = ((s\cdot t)\backslash s,(s\cdot t)^R)$$

is injective. We used $(s\cdot t)\cdot\tau(s,t){\triangleright}t^R = s\cdot(t\cdot t^R) = s$ by quasi-associativity to recognise p, recognised $t^R\cdot\tau(t,t^R)^{-1}{\triangleright}s^R = t\backslash s^R$ from (6.15) and then

$$(s\cdot t)\cdot\tau(s,t){\triangleright}(t\backslash s^R) = s\cdot(t\cdot(t\backslash s^R)) = s\cdot s^R = e$$

to recognise $p\cdot q$. That the desired map is injective is then immediate by $(\ )^R$ injective and elementary properties of division.

We use similar methods in the other proofs. Thus, writing

$$\tau'(s,t) := (\tau(s,t){\triangleleft}t^R)\tau(t,t^R)^{-1} = \tau(s\cdot t,\tau(s,t){\triangleright}t^R)^{-1}$$

for brevity, we have

$$\mathcal{G}^{-1}(\theta\otimes\theta)(\Delta^{op}\delta_r) = \mathcal{G}^{-1}\sum_{p\cdot q=r}(\delta_{q^R}\otimes\delta_{p^R}) = \sum_{s\cdot t=r}\tau(s,t)\delta_{t^R}\otimes\tau'(s,t)\delta_{s^R},$$
$$(\Delta\theta(\delta_r))\mathcal{G}^{-1} = \sum_{p\cdot q=r^R}(\delta_p\otimes\delta_q)\mathcal{G}^{-1} = \sum_{p\cdot q=r^R}\tau(s,t)\delta_{t^R}\otimes\tau'(s,t)\delta_{s^R},$$

where in the second line, commuting the δ_{t^R} and δ_{s^R} to the left sets $p = \tau(s,t){\triangleright}t^R$, $q = \tau'(s,t){\triangleright}s^R$ as studied above. Hence $p\cdot q = r^R$ in the sum is the same as $s\cdot t = r$, so the two sides are equal and we have proven (1) on δ_r. Similarly,

$$\begin{aligned}&\mathcal{G}^{-1}(\theta\otimes\theta)(\Delta^{op}x)\\ &\quad= \sum_{p,q,s,t}\big(\tau(p,q)\delta_{q^R}\otimes(\tau(p,q){\triangleleft}q^R)\tau(q,q^R)^{-1}\delta_{p^R}\big)\big((x{\triangleleft}s){\triangleleft}t\,\delta_{t^R}\otimes\delta_{(x\triangleright s)^R}x{\triangleleft}s\big)\\ &\quad= \sum_{s,t}(x{\triangleleft}s\cdot t)\tau(s,t)\delta_{t^R}\otimes\tau(x{\triangleright}(s\cdot t),(x{\triangleleft}s\cdot t)\tau(s,t){\triangleright}t^R)^{-1}(x{\triangleleft}s)\delta_{s^R}\end{aligned}$$

where we first note that for the δ-functions to connect, we need

$$p = x{\triangleright}s, \quad ((x{\triangleleft}s){\triangleleft}t){\triangleright}t^R = q^R,$$

which is equivalent to $q = (x{\triangleleft}s){\triangleright}t$ since $e = (x{\triangleleft}s){\triangleright}(t \cdot t^R) = ((x{\triangleleft}s){\triangleright}t) \cdot (((x{\triangleleft}s){\triangleleft}t){\triangleright}t^R)$. In this case

$$\tau(p,q)((x{\triangleleft}s){\triangleleft}t) = \tau(x{\triangleright}s, (x{\triangleleft}s){\triangleright}t)((x{\triangleleft}s){\triangleleft}t) = (x{\triangleleft}s \cdot t)\tau(s,t)$$

by the cocycle axiom. Similarly, $(x{\triangleleft}s)^{-1}{\triangleright}(x{\triangleright}s)^R = s^R$ by Lemma 6.4.4 gives us δ_{s^R}. For its coefficient, note that $p \cdot q = (x{\triangleright}s) \cdot ((x{\triangleleft}s){\triangleright}t) = x{\triangleright}(s \cdot t)$ so that, using the other form of $\tau'(p.q)$, we obtain

$$\tau(p \cdot q, \tau(p,q){\triangleright}q^R)^{-1}(x{\triangleleft}s) = \tau(x{\triangleright}(s \cdot t), \tau(p,q)((x{\triangleleft}s){\triangleleft}t){\triangleright}t^R)^{-1}(x{\triangleleft}s)$$

and we use our previous calculation to put this in terms of s, t. On the other side, we have

$$\begin{aligned}(\Delta\theta(x))\mathcal{G}^{-1} &= \sum_t \Delta(x{\triangleleft}t\,\delta_{t^R})\mathcal{G}^{-1}\\ &= \sum_{p,q,s\cdot r=t^R} x{\triangleleft}t\,\delta_s\tau(p,q)\delta_{q^R} \otimes (x{\triangleleft}t){\triangleleft}r\,\delta_r\tau(p\cdot q, \tau(p,q){\triangleright}q^R)^{-1}\delta_{p^R}\\ &= \sum_{p,q} x{\triangleleft}(p\cdot q)\,\tau(p,q)\delta_{q^R} \otimes (x{\triangleleft}p\cdot q){\triangleleft}s\,\tau(p\cdot q, s)^{-1}\delta_{p^R},\end{aligned}$$

where, for the δ-functions to connect, we need

$$s = \tau(p,q){\triangleright}q^R, \quad r = \tau'(p,q){\triangleright}p^R.$$

The map $(p,q) \mapsto (s,r)$ has the same structure as the one we studied above but applied now to p, q in place of s, t. It follows that $s \cdot r = (p \cdot q)^R$ and hence this being equal t^R is equivalent to $p \cdot q = t$. Taking this for the value of t, we obtain the second expression for $(\Delta\theta(x))\mathcal{G}^{-1}$.

We now use the identity for $(x{\triangleleft}p \cdot q){\triangleleft}s$ and $(p \cdot q) \cdot \tau(p,q){\triangleright}q^R = p \cdot (q \cdot q^R) = p$ to obtain the same as we obtained for $\mathcal{G}^{-1}(\theta \otimes \theta)(\Delta^{op}x)$ on x, upon renaming s, t there to p, q. The proofs of (2), (3) are similarly quite involved, but omitted given that it is known that the category of modules is a strong bar category. ■

The key property of any quasi-bialgebra is that its category of modules is monoidal with associator $\phi_{V,W,U} : (V \otimes W) \otimes U \to V \otimes (W \otimes U)$ given by the action of ϕ. In the $*$-quasi case, this becomes a bar category as follows[BM09]. First, there is a functor bar from the category to itself which sends a module V to a 'conjugate', $\bar{V}$. In our case, this has the same set and abelian group structure as V but $\lambda.\bar{v} = \overline{\bar{\lambda}v}$ for all $\lambda \in \mathbb{C}$, i.e. a conjugate action of the field, where we write $v \in V$ as $\bar{v}$ when viewed in $\bar{V}$. Similarly,

$$\xi.\bar{v} = \overline{\theta(\xi).v}$$

for all $\xi \in \Xi(R,K)$. On morphisms $\psi : V \to W$, we define $\bar{\psi} : \bar{V} \to \bar{W}$ by $\bar{\psi}(\bar{v}) = \overline{\psi(v)}$. Next, there is a natural isomorphism $\Upsilon : \mathrm{bar} \circ \otimes \Rightarrow \otimes^{op} \circ (\mathrm{bar} \times \mathrm{bar})$, given in our case for all modules V, W by

$$\Upsilon_{V,W} : \overline{V \otimes W} \cong \bar{W} \otimes \bar{V}, \quad \Upsilon_{V,W}(\overline{v \otimes w}) = \overline{\mathcal{G}^2.w} \otimes \overline{\mathcal{G}^1.v}$$

and making a hexagon identity with the associator, namely

$$(\mathrm{id}\otimes\Upsilon_{V,W})\circ\Upsilon_{V\otimes W,U}=\phi_{\bar U,\bar W,\bar V}\circ(\Upsilon_{W,U}\otimes\mathrm{id})\circ\Upsilon_{V,W\otimes U}\circ\overline{\phi_{V,W,U}}.$$

We also have a natural isomorphism $\mathrm{bb}:\mathrm{id}\Rightarrow\mathrm{bar}\circ\mathrm{bar}$, given in our case for all modules V by

$$\mathrm{bb}_V:V\to\overline{\overline{V}},\quad \mathrm{bb}_V(v)=\overline{\overline{\gamma.v}}$$

and obeying $\overline{\mathrm{bb}_V}=\mathrm{bb}_{\bar V}$. In our case, we have a strong bar category, which means also

$$\Upsilon_{\bar W,\bar V}\circ\overline{\Upsilon_{V,W}}\circ\mathrm{bb}_{V\otimes W}=\mathrm{bb}_V\otimes\mathrm{bb}_W.$$

Finally, a bar category has some conditions on the unit object $\underline{1}$, which in our case is the trivial representation with these automatic. That $G=RK$ leads to a strong bar category is in [BM09, Prop. 3.21] but without the underlying $*$-quasi-Hopf algebra structure as found above.

Now take the standard antipode S in Theorem 6.4.8 and θ constructed above. It is easy to check that

$$*Sx=*(\sum_s\delta_{(x^{-1}\triangleright s)^R}x^{-1}\triangleleft s)=\sum_s(x^{-1}\triangleleft s)^{-1}\delta_{(x^{-1}\triangleright s)^R}=\sum_{s'}x\triangleleft s'\delta_{s'^R}=\theta(x),$$

where $s'=x^{-1}\triangleright s$ and we used Lemma 6.4.4. We also have $*S\delta_s=\delta_{s^R}=\theta(\delta_s)$, so the implicit the standard S recovers the standard $*$-structure used in the main body of the paper from θ. It is also immediate from the above formula for $\gamma,\mathcal{G}$ that $\gamma^*=\gamma^{-1}$ and $\mathcal{G}^{*\otimes *}=\mathcal{G}^{-1}$ as claimed in Proposition 6.4.9. Using these facts and Lemma 6.4.7, on applying $*$ to both sides, the properties of the $*$-quasi bialgebra proven above immediately become the remaining antipode stated in Proposition 6.4.9, completing its proof.