

Classifying Complexity with the ZX Calculus

Alex Townsend-Teague^{0,1} and Konstantinos Meichanetzidis^{1,2}

⁰ Mathematical Institute, University of Oxford

¹ Department of Computer Science, University of Oxford

² Cambridge Quantum Computing Ltd.

The ZX calculus is a graphical language which allows for reasoning about suitably represented tensor networks, aka ZX diagrams, in terms of rewrite rules. Here, we focus on problems which amount to exactly computing a scalar encoded as a closed tensor network. In general, such problems are #P-hard. However, there are families of such problems which are known to be in P when the dimension is below a certain value. By expressing problem-instances from these families as ZX diagrams, we see that the easy instances belong to the stabiliser fragment of the ZX calculus. Building on previous work on efficient simplification of qubit stabiliser diagrams, we present simplifying rewrites for the case of qutrits. Finally, we look at the specific examples of evaluating the Jones polynomial and of counting graph-colourings. Our exposition further constitutes the ZX calculus as a suitable and unifying language for studying the complexity of a broad range of classical or quantum problems.

1 Introduction

A plethora of hard problems of interest to physics and computer science regard interacting many-body multi-state systems, classical or quantum, and reduce to *exactly* computing a single scalar. These range from probability amplitudes in quantum mechanics, partition functions in classical statistical mechanics, counting problems, probabilistic inference, and many more. Problem instances can be encoded as a *closed* tensor-networks over an appropriate (semi)ring. The scalar can be evaluated by *full tensor contraction*, which in general is #P-hard [7]. In fact, finding the optimal contraction path for a general tensor network is NP-hard.

The ZX calculus is a graphical language whose origin lies in the field of quantum foundations [6]. It allows for reasoning about ZX diagrams, i.e. tensor networks which are expressed in terms of primitive generators [21]. The generators have a concrete representation as tensors over a (semi)ring and they obey a set of rewrite rules which respect semantics [20]. Importantly, the rewrite rules *depend* on the (semi)ring over which the diagram is interpreted as a tensor network. If a scalar of interest is expressed as a ZX diagram, one can rewrite the diagram by applying rewrite rules. One's goal is then to apply a sequence of rewrites and perform *full diagram simplification* in order to compute the desired scalar. Note that given an arbitrary tensor network, one can attempt to invent rewrite rules by inspecting the contents of the tensors and performing linear-algebra operations [10].

In the context of quantum computing, the Gottesman-Knill theorem states that stabiliser (or Clifford) circuits can be simulated *efficiently* on classical computers [1]. By simulation here we mean the *exact* computation of *amplitudes*. These circuits can be expressed by the *stabiliser* fragment of the ZX calculus. An alternative, and in fact more general, proof of the Gottesman-Knill theorem for qubits has been obtained graphically in terms of the qubit ZX calculus. Interestingly, even if the motivation for studying stabiliser diagrams stems from quantum computation, the result that stabiliser ZX diagrams can be simplified efficiently has broader applicability. This is because ZX diagrams can express arbitrary

linear maps; not every ZX diagram is a quantum circuit, but every quantum circuit can be cast as a ZX diagram.

Beyond quantum computing, ZX diagrams have been leveraged to study the complexity of boolean satisfiability (SAT) and model counting (#SAT) [5]. Model counting entails computing the number of satisfying assignments to a boolean formula (which is given in conjunctive normal form) and this number can be expressed a closed diagram. Specifically, one finds that in the case of XORSAT and #XORSAT, which are known to be tractable, the corresponding ZX diagrams representing the problems exactly correspond to qubit stabilizer diagrams, which are efficient to simplify. Going further, the ZH calculus [3], an extension of the ZX calculus, can be imported to make graphically obvious why 2SAT is in P but #2SAT is #P-complete while 3SAT is NP-complete and #3SAT is #P-hard. That is, the counting version of 2SAT is hard while deciding is easy, but this is not the case for 3SAT where both deciding and counting are hard. The *key realisation* is in that the decision problem is a counting problem but where the diagram is interpreted over the Boolean semiring. Then the *rewrite rules change accordingly* and obviously imply the above result by allowing efficient simplification strategies.

Such observations make apparent the *unifying power of diagrammatic reasoning for studying the complexity* of problem families of universal interest, where the degrees of freedom are two-dimensional. Building on the spirit of [5], we treat ZX as a library comprising out-of-the-box and readily-available diagrammatic rewrite-rules from which we import the appropriate tools for the job depending on the occasion. The key rewrite rules that enable the efficient diagram simplification of qubit stabiliser ZX diagrams are called *local complementation* and *pivot*, the latter being composition of three of the former [8]. In this work, we recall the qutrit version of local complementation and derive the corresponding pivot rule which implies that qutrit stabiliser diagrams can be simplified efficiently. We then present two case studies: evaluating the Jones polynomial at lattice roots of unity, and graph colouring. Both of these problem families reduce to evaluating closed tensor networks and show a transition in complexity at a particular dimension, below which the tensor network corresponds to a stabiliser ZX diagram.

We underline that throughout this work, all rewrites are valid *up to scalar* since we are only concerned about making statements about complexity.

2 Simplifying Qubit ZX-Diagrams

In this section we briefly review the ZX calculus and recalling the definitions of a graph-like diagrams and stabiliser diagrams. Crucially, we also recall the simplifying rewrites that enable the efficient simplification of stabiliser diagrams.

2.1 The Qubit ZX-Calculus

Here we give a quick introduction to the qubit ZX-calculus. The qubit ZX-calculus is a diagrammatic language for quantum mechanics generated by *spiders*. The spider legs, or *wires*, represent vector spaces of dimension $d = 2$. Diagrams are read bottom-to-top; bottom open wires (not connected to anything) are *input wires* and top ones are *output*. Diagrams with only outputs are called *states* and with only inputs are called *effects*. A *closed* diagram is one with no input nor output wires and it represents a scalar.

Spiders can be *composed*; output wires can be connected to input wires to form spider networks which are again valid ZX diagrams. Placing diagrams side by side represents parallel composition (\otimes), with concrete interpretation as the tensor product. Vertically stacking diagrams corresponds to sequential composition (\circ) and concretely it means matrix multiplication. Specifically, it means tensor contraction

of two spider tensors along the wire connecting them, i.e. the common tensor index represented by this wire is summed over. The concrete representation of these operations of course depends on the (semi)ring over which the spiders are interpreted as tensors. For spiders S and S' , we denote these as

$$\llbracket S \otimes S' \rrbracket = \llbracket S \rrbracket \otimes \llbracket S' \rrbracket \quad , \quad \llbracket S \circ S' \rrbracket = \llbracket S \rrbracket \cdot \llbracket S' \rrbracket \quad (1)$$

Spiders come in two species: green Z-spiders and red X-spiders, decorated by a *phase* $\alpha \in [0, 2\pi)$. When $\alpha = 0$, we will omit it. The concrete representation of the spider generators as tensors over \mathbb{C} is:

$$\left[\begin{array}{c} \overbrace{\quad}^m \\ \vdots \\ \text{green spider } \alpha \\ \vdots \\ \underbrace{\quad}_n \end{array} \right] = |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n} , \quad \left[\begin{array}{c} \overbrace{\quad}^m \\ \vdots \\ \text{red spider } \alpha \\ \vdots \\ \underbrace{\quad}_n \end{array} \right] = |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |-\rangle^{\otimes m} \langle -|^{\otimes n} ,$$

where $\{|0\rangle, |1\rangle\}$ is the *Z-basis* and $|\pm\rangle = |0\rangle \pm |1\rangle$ the *X-basis* in \mathbb{C}^2 , in Dirac notation. The Hadamard gate H , whose function is to switch between the Z and X bases, is denoted as a yellow box. Often we will instead draw a dashed blue line to represent a *Hadamard edge*:

$$\begin{array}{c} | \\ \text{yellow box} \\ | \end{array} = \begin{array}{c} | \\ \text{dashed blue line} \\ | \end{array} = \begin{array}{c} \text{green circle } \frac{\pi}{2} \\ \text{red circle } \frac{\pi}{2} \\ \text{green circle } \frac{\pi}{2} \end{array} , \quad \left[\begin{array}{c} | \\ \text{yellow box} \\ | \end{array} \right] \simeq |0\rangle \langle 0| + |0\rangle \langle 1| + |1\rangle \langle 0| - |1\rangle \langle 1| \quad (2)$$

The ZX calculus is universal for multilinear maps; any tensor over a (semi)ring has a corresponding ZX diagram. The rewrite rules of the ZX calculus (see Fig.1 in E) allow manipulation of the diagrams by *local tensor-rewrites*. Importantly, the rewrites *preserve the semantics*, i.e. the concrete tensor representation over \mathbb{C} , up to a scalar. The ZX calculus is complete in the sense that any true equation between tensors can be proven *only* in terms of rewrites; the diagram on the left hand side of the equation can be transformed to that on the right hand side but applying rewrite rules. This gives ZX the status of a ‘calculus’. What is important about qubit ZX diagrams is that *only topology matters*; the concrete tensor semantics of the diagram is invariant under deformations of the network as long as the inter-spider connectivity is respected.

2.2 Stabilizer Qubit ZX Diagrams

The *stabiliser fragment* of the calculus consists of all diagrams in which all phases are $a = \kappa \frac{\pi}{2}$, $\kappa \in \mathbb{Z}$. In (Theorem 5.4 [8]) the authors give an efficient algorithm for simplifying any *qubit* ZX-diagram to an equivalent diagram with fewer spiders. The algorithm consists of consecutive applications of spider-eliminating rewrites.

First it is shown that every diagram is equivalent to a *graph-like* diagram: every spider is green, every edge is a Hadamard edge, there are no parallel edges or self-loops, every input and output wire is connected to a spider and every spider has at most one input or output wire. The following derivable

rules are key to this:

$$\begin{array}{c} \dots \\ \curvearrowright \\ \alpha \\ \curvearrowleft \\ \beta \\ \curvearrowright \\ \dots \end{array} = \begin{array}{c} \dots \\ \curvearrowright \\ \alpha \\ \dots \end{array}, \quad \begin{array}{c} \dots \\ \curvearrowright \\ \alpha \\ \curvearrowleft \\ \alpha \\ \dots \end{array} = \begin{array}{c} \dots \\ \curvearrowright \\ \alpha \\ \dots \end{array}, \quad \begin{array}{c} \dots \\ \curvearrowright \\ \alpha \\ \curvearrowleft \\ \pi \\ \dots \end{array} = \begin{array}{c} \dots \\ \curvearrowright \\ \alpha + \pi \\ \dots \end{array} \quad (3)$$

Then the following two rewrite rules, derived via *local complementation* and *pivoting*, can be used to eliminate spiders:

$$\begin{array}{c} \alpha_1 \quad \dots \quad \alpha_n \\ \vdots \quad \vdots \quad \vdots \\ \dots \end{array} \xrightarrow{\pm \frac{\pi}{2}} \begin{array}{c} \alpha_1 \mp \frac{\pi}{2} \quad \dots \quad \alpha_n \mp \frac{\pi}{2} \\ \vdots \quad \vdots \quad \vdots \\ \dots \end{array} \quad (4)$$

$$\begin{array}{c} \alpha_1 \quad \dots \quad \alpha_n \\ \vdots \quad \vdots \quad \vdots \\ \dots \end{array} \xrightarrow{j\pi, k\pi} \begin{array}{c} \alpha_1 + k\pi \quad \dots \quad \alpha_n + j\pi \\ \vdots \quad \vdots \quad \vdots \\ \dots \end{array} \quad (5)$$

For more details, see Section 4 [8]. Eq.4 says that we can remove any spider with phase $\pm \frac{\pi}{2}$ at the cost of performing a local complementation at said spider. Furthermore, Eq.5 says we can remove any pair of spiders with phases in $\{0, \pi\}$ connected by a Hadamard edge at the cost of performing a local pivot along said edge. After each application of (4) or (5), we can use (3) to remove any parallel Hadamard edges and ensure the diagram remains graph-like.

In particular, and importantly to this work, given a closed stabilizer diagram the algorithm *efficiently simplifies* it until it contains at most one spider. If it exists, this spider is green, legless and has phase in $\{0, \pi\}$. The point relevant to this work is that spiders can be *eliminated efficiently*, i.e. with a polynomial number of rewrites in the initial number of spiders. Note that every spider elimination introduces a polynomial number of edges in the diagram, which prevents an overwhelming memory cost of the simplification procedure.

3 Simplifying Qutrit ZX-Diagrams

We now turn to the qutrit ZX-calculus and examine the analogous story to that of the previous subsection but now for the case where the dimension of the vector space carried by the wires is $d = 3$.

3.1 The Qutrit ZX-Calculus

As in the qubit case, the qutrit ZX calculus is about spiders connected by wires, but there are key differences, some subtler than others. Again, spiders come in two species, Z (green) and X (red), with the three-dimensional Z basis denoted as $\{|0\rangle, |1\rangle, |2\rangle\}$. Let $\omega = e^{i\frac{2\pi}{3}}$ denote the third root of unity with $\bar{\omega} = \omega^2$ its complex conjugate. The qutrit X-basis consists of the three vectors:

$$|+\rangle = \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle), \quad |\omega\rangle = \frac{1}{\sqrt{3}}(|0\rangle + \omega|1\rangle + \bar{\omega}|2\rangle), \quad |\bar{\omega}\rangle = \frac{1}{\sqrt{3}}(|0\rangle + \bar{\omega}|1\rangle + \omega|2\rangle) \quad (6)$$

In qutrit ZX, spiders carry *two* phases α and β , and have the following concrete representation as linear maps:

$$\left[\begin{array}{c} m \\ \vdots \\ \text{spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \vdots \\ n \end{array} \right] = |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n} + e^{i\beta} |2\rangle^{\otimes m} \langle 2|^{\otimes n} \quad (7)$$

$$\left[\begin{array}{c} m \\ \vdots \\ \text{spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \vdots \\ n \end{array} \right] = |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |\omega\rangle^{\otimes m} \langle \omega|^{\otimes n} + e^{i\beta} |\bar{\omega}\rangle^{\otimes m} \langle \bar{\omega}|^{\otimes n} \quad (8)$$

When $\alpha = \beta = 0$ we will again omit the angles entirely, and just draw a small green or red dot. Throughout our work, whenever we use an integer n as a spider decoration, this is a shorthand for $\frac{2\pi}{3}n$. Since spider phases hold mod 2π , these integer decorations hold mod 3. Unless otherwise stated, we will use Greek letters to denote general angles, and Roman letters for these integer shorthands.

Hadamard gates are no longer self-adjoint, so we change our notation: we let a yellow box decorated with a 1 (mod 3) denote a Hadamard gate, while decorating with a 2 (mod 3) denotes its adjoint. We will shortly explain this choice. We also use a dashed blue line for the *Hadamard edge* (H -edge) and a purple dashed line for its adjoint (H^\dagger -edge):

$$\begin{array}{c} \text{yellow box } 1 \\ \text{dashed blue line} \end{array} = \begin{array}{c} \text{green circle } \begin{smallmatrix} 2 \\ 2 \end{smallmatrix} \\ \text{red circle } \begin{smallmatrix} 2 \\ 2 \end{smallmatrix} \\ \text{green circle } \begin{smallmatrix} 2 \\ 2 \end{smallmatrix} \end{array}, \quad \begin{array}{c} \text{yellow box } 2 \\ \text{dashed purple line} \end{array} = \begin{array}{c} \text{green circle } \begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \\ \text{red circle } \begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \\ \text{green circle } \begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \end{array}, \quad \begin{array}{c} \text{yellow box } 0 \\ \text{dashed blue line} \end{array} = \begin{array}{c} \text{empty diagram} \end{array} \quad (9)$$

The last equation above just says that a ‘0-Hadamard edge’ is in fact the empty diagram, and not an edge at all. A very important difference from the qubit case is that in qutrit ZX-calculus there is no ‘plain’ cap or cup:

$$\begin{array}{c} \text{green cap} \end{array} \neq \begin{array}{c} \text{red cap} \end{array}, \quad \begin{array}{c} \text{green cup} \end{array} \neq \begin{array}{c} \text{red cup} \end{array} \quad (10)$$

This has several consequences. Firstly, the maxim that ‘only topology matters’ no longer applies. That is, it is now important to make clear the distinction between a spider’s inputs and outputs, unlike in the qubit case where we could freely interchange the two. Diagram components can still be isotoped around the plane but only so long as this input/output distinction is respected. This gives the qutrit calculus a slightly more rigid flavour than its qubit counterpart. That said, this rigidity is loosened in certain cases; in particular, this distinction is irrelevant for H - and H^\dagger -edges [9]:

$$\begin{array}{c} \text{green spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \text{dashed blue line} \end{array} = \begin{array}{c} \text{green spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \text{dashed purple line} \end{array}, \quad \begin{array}{c} \text{green spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \text{dashed purple line} \end{array} = \begin{array}{c} \text{green spider } \begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \\ \text{dashed blue line} \end{array} \quad (11)$$

The full set of rules governing the qutrit ZX calculus is shown in Figure 2 in the Appendix.

3.2 Graph-Like Qutrit ZX Diagrams

A *graph-like qutrit ZX diagram* is one where every spider is green, spiders are only connected Hadamard edges (blue) or their adjoints (purple), every pair of spiders is connected by at most one H -edge or H^\dagger -edge, every input and output wire is connected to a spider, and every spider is connected to at most one input or output wire. A graph-like qutrit ZX diagram is a *graph state* when every spider has zero phase (top and bottom) and is connected to an output.

Note the difference compared to the qubit case: we need not worry about self-loops because the qutrit ZX calculus doesn't define a 'plain' cap or cup. But this comes at a cost: spiders in the qutrit case fuse more fussily. Specifically, when two spiders of the same colour are connected by at least one plain edge and at least one H - or H^\dagger -edge, fusion is not possible. Instead, we can ensure we have a graph-like diagram by replacing the plain wire:

$$\begin{array}{c} \text{(id)} \\ \text{(H)} \end{array} \quad \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array} \quad (12)$$

Indeed, we will shortly show that every qutrit ZX-diagram is equivalent to a graph-like one, making use of the following lemma:

Lemma 3.1. The following equations hold in the qutrit ZX-calculus:

$$\begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array} = \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array} = \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array}, \quad \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array} = \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array}, \quad \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array} = \begin{array}{c} \dots \\ \alpha \\ \beta \\ \gamma \\ \delta \\ \dots \end{array} \quad (13)$$

Proof. See Appendix B.3 □

This justifies our notation for Hadamard gates: we can think of Hadamard edges as 1-weighted edges and their adjoints as 2-weighted edges, then work modulo 3, since every triple of parallel edges disappears. Where the previous lemma relates single H - and H^\dagger -boxes across multiple edges, the next relates multiple H - and H^\dagger -boxes on single edges.

Lemma 3.2. The following three equations hold in the qutrit ZX calculus, for $h \in \{1, 2\}$:

$$\begin{array}{c} h \\ h \\ h \\ h \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{array}, \quad \begin{array}{c} h \\ h \\ h \\ h \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{array}, \quad \begin{array}{c} h \\ h \\ h \end{array} = \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \end{array} \quad (14)$$

Proof. For the first equation, negate the top two boxes via (s), then cancel twice via (H). Similarly, for the second equation, negate the top two boxes via (s), then cancel once via (H). The third equation is just a simple application of (id). □

Corollary 3.3. Every qutrit ZX diagram is equivalent to one that is graph-like.

Proof. First use the colour change rule to turn all X-spiders into Z-spiders. Then use Lemma 3.2 to remove excess H - and H^\dagger -boxes, inserting a spider between any remaining consecutive pair of such

boxes, so that all spiders are connected only by plain edges, H -edges or H^\dagger -edges. Fuse together as many as possible, and apply (12) where fusion is not possible, so that no plain edge connects two spiders. Apply Lemma 3.1 to all connected pairs of spiders until at most one H - or H^\dagger -edge remains between them. Finally, to ensure every input and output is connected to a spider and every spider is connected to at most one input or output, we can use **(H)** and **(id)** to add a few spiders, H - and H^\dagger -edges as needed:

$$\begin{array}{c} | \end{array} = \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array}, \quad \begin{array}{c} \text{---} \text{H} \text{---} \\ | \end{array} = \begin{array}{c} \text{---} \text{H} \text{---} \\ | \end{array} \circ \begin{array}{c} | \end{array}, \quad \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} = \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \quad (15)$$

□

A graph state is described fully by its underlying multigraph, or equivalently by an adjacency matrix, where edges take weights in \mathbb{Z}_3 [Lemma 4.2 [19]]. Nodes correspond to phaseless green spiders, edges of weight 1 correspond to Hadamard edges, and edges of weight 2 correspond to H^\dagger edges. As in the qubit case, graph states admit a local complementation operation Definition 2.6 [19], though the effect is now slightly more complicated. We'll give the intuition after the formal definition:

Definition 3.4. Given $a \in \mathbb{Z}_3$ and a graph state G with adjacency matrix $W = (w_{i,j})$, the a -local complementation at node x is the new graph state $G *_a x$, whose adjacency matrix $W' = (w'_{i,j})$ given by:

$$w'_{i,j} = w_{i,j} + aw_{i,x}w_{j,x} \quad (16)$$

So only those edges between neighbours of node x are affected, but rather than just having their weight increased by 1 (modulo 2) as in the qubit case, the increase in weight also depends on the weights of the edges from i and j to x . As always, this is best seen graphically. For two nodes i and j both connected to x by the same colour edge, a -local complementation at x increases weight $w_{i,j}$ by a . If instead i and j are connected to x by edges of different colour, a -local complementation at x decreases $w_{i,j}$ by a . We show a fragment of a ZX-diagram below under the effect of this operation:

$$\begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \xrightarrow{*a} \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \quad \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \xrightarrow{*a} \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \quad (17)$$

Both mappings above hold with the roles of blue and purple swapped. As in the qubit case, local complementation gives an equality up to introducing some single qubit phase gates on the outputs.

Theorem 3.5. Given $a \in \mathbb{Z}_3$ and a graph state (G, W) containing a node x , let $N(x)$ denote the neighbours of x - that is, nodes i with weight $w_{i,x} \in \{1, 2\}$. Then the following equality holds:

$$\begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \xrightarrow{*a} \begin{array}{c} \text{---} \circ \text{---} \\ | \end{array} \quad (18)$$

Proof. This is Theorem 4.4 and Corollary 4.5 in [19].

□

Composing local complementations gives a local pivot operation.

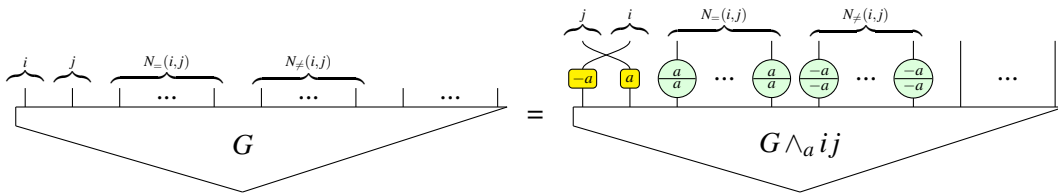
Definition 3.6. Given $a, b, c \in \mathbb{Z}_3$ and a graph state G containing nodes i and j , the (a, b, c) -local pivot along ij is the new graph state $G \wedge_{(a,b,c)} ij := ((G *_{a,b} i) *_{c,b} j) *_{c,b} i$.

This again results in an equality, up to introducing some extra gates on outputs. Here we shall only consider an $(a, -a, a)$ -local pivot along an edge ij of non-zero weight, for $a \in \{1, 2\}$. We will call this a *proper a -local pivot* along ij , and denote it $G \wedge_a ij$.

Theorem 3.7. Given $a \in \mathbb{Z}_3$ and a graph state (G, W) containing connected nodes i and j , define the following:

- $N_=(i, j) := \{x \in N(i) \cap N(j) \mid w_{x,i} = w_{x,j}\}$
- $N_\neq(i, j) := \{x \in N(i) \cap N(j) \mid w_{x,i} \neq w_{x,j}\}$

Then the following equation relates G and its proper a -local pivot along ij :



Proof. See Appendix C.1. □

3.3 Qutrit Elimination Theorems

We classify spiders into three families exactly as in Theorem 3.1 [19]:

$$\mathcal{M} = \left\{ \begin{pmatrix} \cdots \\ 0 \\ 0 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 1 \\ 2 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 2 \\ 1 \\ \cdots \end{pmatrix} \right\}, \quad \mathcal{N} = \left\{ \begin{pmatrix} \cdots \\ 0 \\ 1 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 1 \\ 0 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 0 \\ 2 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 2 \\ 0 \\ \cdots \end{pmatrix} \right\}, \quad \mathcal{P} = \left\{ \begin{pmatrix} \cdots \\ 1 \\ 1 \\ \cdots \end{pmatrix}, \begin{pmatrix} \cdots \\ 2 \\ 2 \\ \cdots \end{pmatrix} \right\}.$$

We call a spider in a graph-like ZX-diagram *interior* if it isn't connected to an input or output (so for our use case all spiders are interior). Given any graph-like ZX-diagram, we will show that we can eliminate standalone interior \mathcal{P} - and \mathcal{N} -spiders by local complementation, and pairs of connected interior \mathcal{M} -spiders by local pivoting.

First, we define (a very small extension of) the $!$ -notation (pronounced ‘bang notation’), as introduced in [ToDo] for general string diagrams. A $!$ -box in a ZX-diagram allows its contents to ‘unfold’ out arbitrarily; that is, the contents of the $!$ -box, along with any wires into or out of the box, can be copied n times for any non-negative integer n , and placed side-by-side. Following the style of [ToDo], we also allow a parameter over which a $!$ -box unfolds. We extend this notation as follows: in the corner of the $!$ -box we allow a small further diagram, which denotes that the unfolded diagrams should all be pairwise connected by the diagram in the corner of the box. The simplest choice of this ‘corner diagram’ is just the plain wire. In our context (qutrit ZX-calculus) this notation will only be well-defined for certain scenarios: in particular, it is well-defined when the corner diagram is a H - or H^\dagger -edge (since then the distinction between a spider’s input and output wires disappears) and the main contents of the $!$ -box is equivalent to a single spider (since then there is no ambiguity about which spiders are connected by the corner diagram). For example, letting $[K] = \{1, \dots, K\}$, we have:

$$\left\{ \left(\begin{array}{c} k \in [K] \\ \boxed{\alpha_k} \\ \beta_k \end{array} \right) \mid K \in \{0, 1, 2, 3\} \right\} = \left\{ \begin{array}{c} \bullet \\ \bullet \end{array}, \begin{array}{c} \alpha_1 \\ \beta_1 \end{array}, \begin{array}{c} \alpha_1 \quad \alpha_2 \\ \beta_1 \quad \beta_2 \end{array}, \begin{array}{c} \alpha_1 \quad \alpha_2 \quad \alpha_3 \\ \beta_1 \quad \beta_2 \quad \beta_3 \end{array} \right\}$$

Theorem 3.8. Given any graph-like ZX-diagram containing an interior \mathcal{P} -spider x with phase $\frac{p}{p}$ for $p \in \{1, 2\}$, suppose we perform a p -local complementation at x . Then the new ZX-diagram is related to the old one by the following equality:

$$\begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \begin{array}{c} p \\ p \end{array} = \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \begin{array}{c} p \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \begin{array}{c} p \end{array}$$

Proof. See Appendix D.4. \square

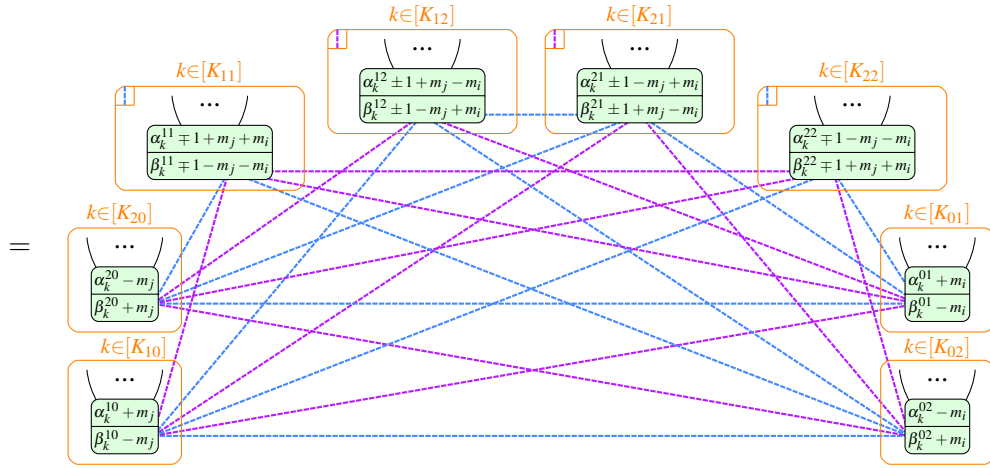
Theorem 3.9. Given any graph-like ZX-diagram containing an interior \mathcal{N} -spider x with phase $\frac{0}{n}$ or $\frac{n}{0}$ for $n \in \{1, 2\}$, suppose we perform a $(-n)$ -local complementation at x . Then, treating the two cases separately, the new ZX-diagrams are related to the old ones by the equalities:

$$\begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k \\ \beta_k \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k \\ \delta_k \end{array} \begin{array}{c} 0 \\ n \end{array} = \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k + n + 2 \\ \beta_k + n + 1 \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k + n + 1 \\ \delta_k + n + 2 \end{array} \begin{array}{c} -n \end{array}, \quad \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k \\ \beta_k \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k \\ \delta_k \end{array} \begin{array}{c} n \\ 0 \end{array} = \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k + n + 1 \\ \beta_k + n + 2 \end{array} \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k + n + 2 \\ \delta_k + n + 1 \end{array} \begin{array}{c} -n \end{array}$$

Proof. See Appendix D.7. \square

Theorem 3.10. Given any graph-like ZX-diagram containing two interior \mathcal{M} -spiders i and j connected by edge ij of weight $w_{i,j} =: w \in \{1, 2\}$, suppose we perform a proper $\pm w$ -local pivot along ij (both choices give the same result). For the case $w = 1$, the new ZX-diagram is related to the old one by the following equality:

$$\begin{array}{c} k \in [K_{11}] \\ \dots \\ \alpha_k^{11} \\ \beta_k^{11} \end{array} \begin{array}{c} k \in [K_{12}] \\ \dots \\ \alpha_k^{12} \\ \beta_k^{12} \end{array} \begin{array}{c} k \in [K_{21}] \\ \dots \\ \alpha_k^{21} \\ \beta_k^{21} \end{array} \begin{array}{c} k \in [K_{22}] \\ \dots \\ \alpha_k^{22} \\ \beta_k^{22} \end{array} \begin{array}{c} k \in [K_{20}] \\ \dots \\ \alpha_k^{20} \\ \beta_k^{20} \end{array} \begin{array}{c} k \in [K_{01}] \\ \dots \\ \alpha_k^{01} \\ \beta_k^{01} \end{array} \begin{array}{c} k \in [K_{10}] \\ \dots \\ \alpha_k^{10} \\ \beta_k^{10} \end{array} \begin{array}{c} k \in [K_{02}] \\ \dots \\ \alpha_k^{02} \\ \beta_k^{02} \end{array} \begin{array}{c} m_i \\ -m_i \end{array} \begin{array}{c} m_j \\ -m_j \end{array}$$



The case $w = 2$ differs only as follows: in the first diagram (the left hand side of the equation), the edge ij is purple (by definition), while in the lower diagram (the right hand side of the equation), a $\pm 2 = \mp 1$ will replace all occurrences of ± 1 , and the roles of purple and blue will be swapped throughout.

Proof. See Appendix D.8. \square

We can now combine these three elimination theorems into an algorithm for efficiently simplifying a closed graph-like ZX-diagram. First note that after applying any one of the three elimination theorems to such a diagram, the only way we might end up with a non-graph-like diagram is if we introduce parallel H - or H^\dagger -edges. Fortunately we have shown via Lemma 3.1 that these can always be reduced to just one edge, so we do indeed again have a graph-like diagram.

Theorem 3.11. Given any closed graph-like ZX-diagram, the following algorithm will always terminate after a finite number of steps, returning an equivalent graph-like ZX-diagram with no \mathcal{N} -spiders, \mathcal{P} -spiders, or adjacent pairs of \mathcal{M} -spiders. Repeat the steps below until no rule matches. After each step, apply Lemma 3.1 as needed until the resulting diagram is graph-like:

1. Eliminate a \mathcal{P} -spider via Theorem 3.8.
2. Eliminate an \mathcal{N} -spider via Theorem 3.9.
3. Eliminate two adjacent \mathcal{M} -spiders via Theorem 3.10.

Proof. At every step the total number of spiders decreases by at least one, so since we start with a finite diagram the algorithm terminates after a finite number of steps. By construction, when it does so we are left with an equivalent graph-like ZX-diagram with no \mathcal{N} -spiders, \mathcal{P} -spiders, or adjacent pairs of \mathcal{M} -spiders. \square

Corollary 3.12. In particular, if we start with a stabilizer diagram, we can eliminate all but perhaps one \mathcal{M} -spider, depending on whether the initial number of \mathcal{M} -spiders was odd or even.

Proof. No step introduces any non-stabilizer phases. \square

The algorithm above could be extended to graph-like diagrams with inputs or outputs as in Theorem 5.4 [8], but since for our purposes we don't need to do so, we have not gone to the trouble.

4 Case studies

In this section we present two problems which can naturally be cast in tensor-network form. These problem families are interesting in that they show a transition in complexity when the dimension d carried by the wires of the network is greater than a specific value. First we present the problem of evaluating the Jones polynomial at specific points, which is a prominent link invariant in knot theory. Finally, we briefly look at graph vertex-colouring.

4.1 Jones Polynomial at Lattice Roots of Unity

A knot K is a circle embedded in \mathbb{R}^3 . A set of knots tangled together make a *link* L . A link L can be represented by a *link diagram* as its projection on the plane, but retaining the information of *over or under crossings*. We say that $L \simeq L'$ iff the diagram of link L can be deformed to that of link L' without cutting or gluing the strands. The Jones polynomial $V_L(t)$ is a Laurent polynomial in a variable $t \in \mathbb{C}$ and is a *link invariant*. This means that $V_L(t) \neq V_{L'}(t) \Rightarrow L \not\simeq L'$.

In general, computing $V_L(t)$ is exponentially costly in the number of crossings c , something made explicit when one uses the Kauffman bracket method [12]. Exactly evaluating the Jones polynomial at points $t \in \mathbb{C}$ is #P-hard, *except* at the *lattice roots of unity* $t \in \Lambda = \{\pm 1, \pm i, \pm e^{i2\pi/3}, \pm e^{i4\pi/3}\}$, where it can be evaluated efficiently (in time $O(\text{poly}(c))$) [11].

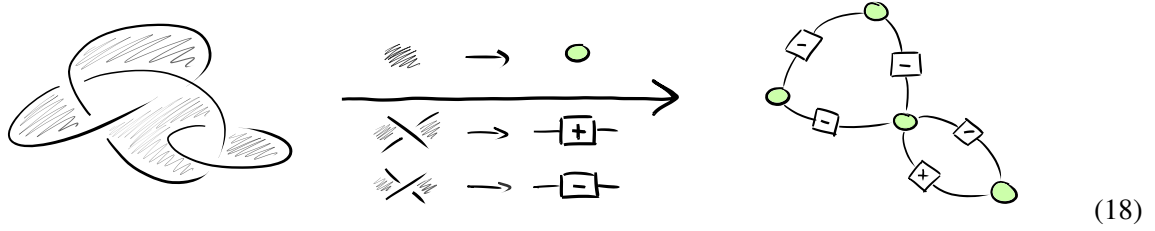
In the context of quantum computation, additively approximating the Jones polynomial at *non-lattice roots of unity* is the paradigmatic BQP-complete problem [2, 14]. From the more natural for knot theory point of view of topological quantum computation [17], the Jones polynomial at roots of unity corresponds to a quantum amplitude in the fusion space of non-abelian anyons, emergent quasiparticles with exotic exchange statistics. Computation is performed by initialising the anyons by creating them from the vacuum, then braiding them, and finally fusing them back to the vacuum. Specifically, in the case of $\text{SU}(2)_k$ anyons, the Jones polynomial is evaluated at $t(k) = e^{i\frac{2\pi}{2+k}}$ [18].

Remarkably, $V_L(t \in \Lambda)$ can be expressed as an unphysical partition function $Z_{G_L}(d)$ of a d -state Potts model with suitable spin-spin interactions [22]. The Potts model is defined on a signed graph G_L , obtained as follows. The link diagram is bicoloured checkerboard-style and every coloured area is mapped to a vertex and every crossing is mapped to a signed edge according its orientation relative to the surrounding colours. The Jones polynomial then is equal to this partition function up to an efficiently computable scalar which depends on the link diagram $V_L(t(d)) \simeq Z_{G_L}(d)$.

The relation between the Jones' evaluation point and the dimension of the spins is $d = t + t^{-1} + 2$, which can be solved for $t(d)$ [16]. Note the correspondence between the dimension d in the Potts approach and the level k in the anyon-braiding approach to the Jones polynomial, $\{t(d) \mid d \in \{1, 2, 3, 4\}\} = \{t(k) \mid k \in \{1, 2, 4\}\}$. This is consistent with the fact that braiding-only topological quantum computation with $\text{SU}(2)_2$ anyons (Ising) or $\text{SU}(2)_4$ anyons is not universal (unless the $\text{SU}(2)_4$ anyons are augmented by fusion and measurements [15]). A key observation is that if $d \in \{2, 3, 4\}$ then $t(d) \in \Lambda$ (the case $d = 1$ is trivial [11]).

The partition function $Z_{G_L}(d \in \mathbb{N})$ can be expressed as a closed tensor network in terms of phaseless

(green) d -dimensional Z-spiders connected via wires that go through \pm -boxes [16]:



The \pm -boxes have concrete interpretation as matrices as

$$\left[\begin{array}{c} | \\ \boxed{\pm} \\ | \end{array} \right] = \sum_{i,j=0}^{d-1} (1 - (1 + t(d)^{\mp}) \delta_{ij}) |i\rangle \langle j| . \quad (19)$$

We can now express the \pm -boxes in the ZX calculus.

Proposition 4.1. The \pm -matrices (19) for $d \in \{2, 4\}$ are equal up to a scalar to concrete interpretations of *qubit stabiliser ZX*-diagrams, and the \pm -matrices for $d = 3$ of *qutrit stabiliser ZX*-diagrams.

$$\left[\begin{array}{c} | \\ \boxed{\pm} \\ | \end{array} \right]_{d=2} \simeq \left[\begin{array}{c} | \\ \text{red circle with } \pm \frac{\pi}{2} \\ | \end{array} \right] , \quad \left[\begin{array}{c} | \\ \boxed{\pm} \\ | \end{array} \right]_{d=3} \simeq \left[\begin{array}{c} | \\ \text{red circle with } \pm 1 \\ | \end{array} \right] , \quad \left[\begin{array}{c} | \\ \boxed{\pm} \\ | \end{array} \right]_{d=4} \simeq \left[\begin{array}{c} | \\ \text{red circle with } \pi \text{ --- yellow square --- } \pi \\ | \end{array} \right] , \quad (20)$$

Proof. See Appendix E.1. □

Since these generators decompose as stabiliser diagrams, then $Z_{G_L}(d \in \{2, 3, 4\})$ can be evaluated *efficiently* via stabiliser ZX diagram simplification. Thus, we recover the known result that evaluating the Jones polynomial at $t \in \Lambda$ is in P, where we used the ZX calculus to express all the whole problem family on equal footing.

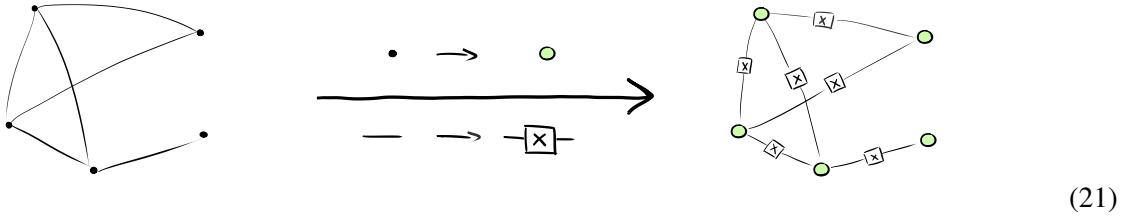
what does the \pm matrix look like for $d = 5$? Here it should not be a stabiliser, showing that ZX cannot efficiently get the Jones, as expected.

4.2 Graph Colouring

Finally, let us briefly look at the problem graph vertex-colouring. That is, given d colours and a graph G , count the number of colour assignments to the vertices of G so that no neighbouring vertices have the same colour. Again, this problem can be interpreted as the zero-temperature partition function of an antiferromagnetic Potts model (there is an energy cost for adjacent spins to be in the same state). To keep with the spirit of our graphical exposition, we recall that computing partition functions and counting problems are essentially the same problem, as they can be straightforwardly encoded as closed tensor networks [13, 5].

Given a graph G , the problem of counting vertex d -colourings can be encoded as a ZX diagram as follows. Every vertex of the graph is mapped to a d -dimensional (green) phaseless Z-spider. Every edge

is mapped to a wire connecting the spiders which goes through an X-box.



The X-boxes have the concrete interpretation (they are the special case of the \pm boxes for $t = 0$):

$$\left[\begin{array}{c} | \\ \boxed{X} \\ | \end{array} \right] = \sum_{i,j=0}^{d-1} (1 - \delta_{ij}) |i\rangle \langle j| . \quad (22)$$

for $d = 2$ it's just the Pauli X so it's stabiliser. what does the X matrix look like for $d = 3$? Here it should not be a stabiliser, showing that ZX cannot efficiently do 3-colouring which is #P-complete. [ref]

5 Outlook

This also motivates further work in circuit extraction for qutrit circuits so that one can attempt graph-theoretic simplification with the qutrit ZX calculus, analogous to the case of qubits [8, 4]. **harny's crazy generalisations: what can they potentially do for general CSP?**

Future work, scalar exact version of the qutrit rewrite rules, for concrete applications to problems where 3-state systems are involved. Further, generalisation of local complementation and pivot rewrites to arbitrary prime dimension. Software implementations a la pyzx could be extended accordingly.

6 Acknowledgments

Alex would like to thank Aleks Kissinger and Quanlong (Harny) Wang for their help throughout this research. KM wishes to thank Niel de Beaudrap, Aleks Kissinger, Stefanos Kourtis, and Quanlong (Harny) Wang for inspiring and helpful discussions. KM acknowledges financial support from the Royal Commission for the Exhibition of 1851 through a research fellowship.

References

- [1] Scott Aaronson & Daniel Gottesman (2004): *Improved simulation of stabilizer circuits*. *Physical Review A* 70(5), doi:10.1103/physreva.70.052328. Available at <http://dx.doi.org/10.1103/PhysRevA.70.052328>.
- [2] Dorit Aharonov, Vaughan Jones & Zeph Landau (2008): *A Polynomial Quantum Algorithm for Approximating the Jones Polynomial*. *Algorithmica* 55(3), p. 395–421, doi:10.1007/s00453-008-9168-0. Available at <http://dx.doi.org/10.1007/s00453-008-9168-0>.
- [3] Miriam Backens & Aleks Kissinger (2018): *ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity*.

- [4] Miriam Backens, Hector Miller-Bakewell, Giovanni de Felice, Leo Lobski & John van de Wetering (2020): *There and back again: A circuit extraction tale*.
- [5] Niel de Beaudrap, Aleks Kissinger & Konstantinos Meichanetzidis (2020): *Tensor Network Rewriting Strategies for Satisfiability and Counting*.
- [6] Bob Coecke & Ross Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New Journal of Physics* 13(4), p. 043016, doi:10.1088/1367-2630/13/4/043016. Available at <http://dx.doi.org/10.1088/1367-2630/13/4/043016>.
- [7] Carsten Damm, Markus Holzer & Pierre McKenzie (2002): *The complexity of tensor calculus*. *computational complexity* 11(1), pp. 54–89, doi:10.1007/s00037-000-0170-4. Available at <https://doi.org/10.1007/s00037-000-0170-4>.
- [8] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*.
- [9] Xiaoyan Gong & Quanlong Wang (2017): *Equivalence of Local Complementation and Euler Decomposition in the Qutrit ZX-calculus*.
- [10] Johnnie Gray & Stefanos Kourtis (2020): *Hyper-optimized tensor network contraction*.
- [11] F. Jaeger, D. L. Vertigan & D. J. A. Welsh (1990): *On the computational complexity of the Jones and Tutte polynomials*. *Mathematical Proceedings of the Cambridge Philosophical Society* 108(1), p. 35–53, doi:10.1017/S0305004100068936.
- [12] Louis H Kauffman (2001): *Knots and Physics*. WORLD SCIENTIFIC, doi:10.1142/4256. Available at <https://doi.org/10.1142/4256>.
- [13] Stefanos Kourtis, Claudio Chamon, Eduardo R. Mucciolo & Andrei E. Ruckenstein (2019): *Fast counting with tensor networks*. *SciPost Phys.* 7, p. 60, doi:10.21468/SciPostPhys.7.5.060. Available at <https://scipost.org/10.21468/SciPostPhys.7.5.060>.
- [14] Greg Kuperberg (2014): *How hard is it to approximate the Jones polynomial?*
- [15] Claire Levaillant, Bela Bauer, Michael Freedman, Zhenghan Wang & Parsa Bonderson (2015): *Universal gates via fusion and measurement operations on $SU(2)_4$ anyons*. *Physical Review A* 92(1), doi:10.1103/physreva.92.012301. Available at <http://dx.doi.org/10.1103/PhysRevA.92.012301>.
- [16] Konstantinos Meichanetzidis & Stefanos Kourtis (2019): *Evaluating the Jones polynomial with tensor networks*. *Phys. Rev. E* 100, p. 033303, doi:10.1103/PhysRevE.100.033303. Available at <https://link.aps.org/doi/10.1103/PhysRevE.100.033303>.
- [17] Jiannis K. Pachos (2012): *Introduction to Topological Quantum Computation*. Cambridge University Press, doi:10.1017/CBO9780511792908.
- [18] Eric C. Rowell & Zhenghan Wang (2018): *Mathematics of topological quantum computing*. *Bulletin of the American Mathematical Society* 55(2), p. 183–238, doi:10.1090/bull/1605. Available at <http://dx.doi.org/10.1090/BULL/1605>.
- [19] Quanlong Wang (2018): *Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics*.
- [20] Quanlong Wang (2020): *Completeness of algebraic ZX-calculus over arbitrary commutative rings and semirings*.
- [21] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*.
- [22] F. Y. Wu (1992): *Knot theory and statistical mechanics*. *Rev. Mod. Phys.* 64, pp. 1099–1131, doi:10.1103/RevModPhys.64.1099. Available at <https://link.aps.org/doi/10.1103/RevModPhys.64.1099>.

A Qubit ZX Calculus

The well-known rewrite rules of the qubit ZX calculus are shown in Fig.1.

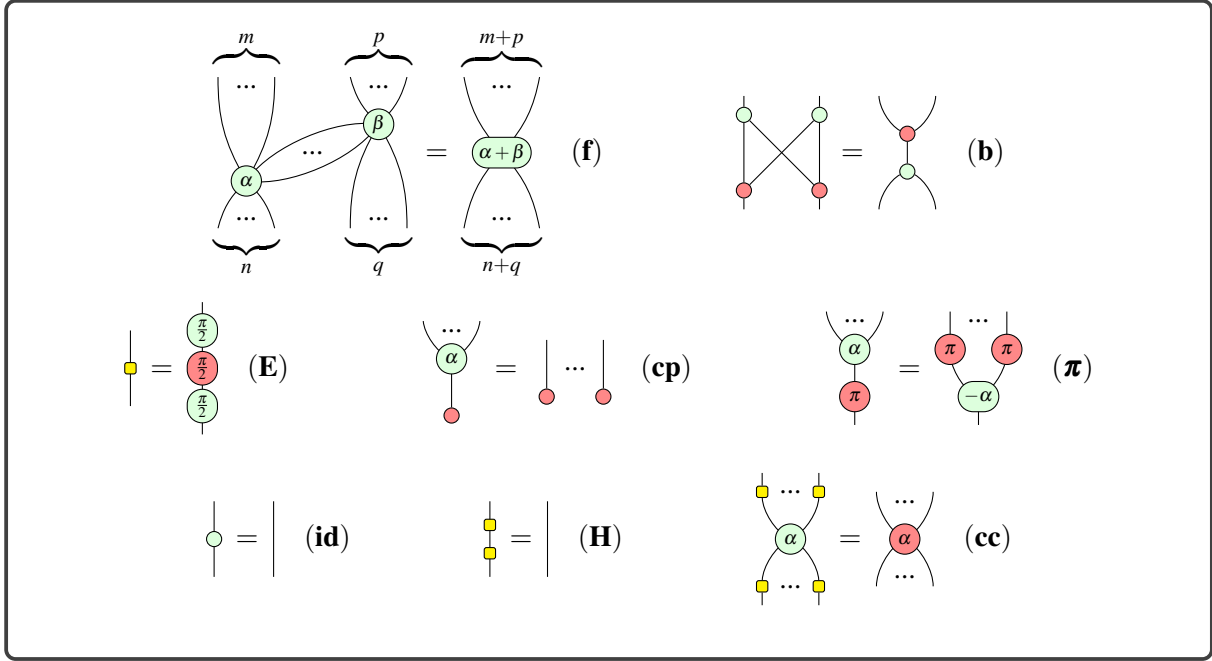


Figure 1: Rewrite rules for the qubit ZX-calculus where spiders are interpreted as tensors over \mathbb{C} .

B Qutrit ZX calculus

We now give a full set of rules defining the qutrit ZX-calculus. Our presentation aims for clarity and accessibility; for a more rigorous description, see [19].

Definition B.1. The *qutrit ZX-calculus* is a graphical calculus generated by the following diagrams, where $\alpha, \beta \in [0, 2\pi]$:

$$\begin{array}{c} \dots \\ \alpha \\ \beta \\ \dots \end{array} , \quad \begin{array}{c} \dots \\ \alpha \\ \beta \\ \dots \end{array} , \quad \begin{array}{c} \text{1} \\ | \end{array} , \quad \begin{array}{c} \diagup \quad \diagdown \\ \diagdown \quad \diagup \end{array} , \quad | \quad (23)$$

and their adjoints $(-)^{\dagger}$. Adjoints are found by swapping inputs and outputs and negating any decorations - recall negation is mod 2π for general spider phases, and mod 3 for integer spider phases and Hadamard boxes. Thus the two rightmost generators are self-adjoint, whereas the first three satisfy:

$$\left(\begin{array}{c} m \\ \dots \\ \alpha \\ \beta \\ \dots \\ n \end{array} \right)^{\dagger} = \begin{array}{c} n \\ \dots \\ -\alpha \\ -\beta \\ \dots \\ m \end{array} , \quad \left(\begin{array}{c} m \\ \dots \\ \alpha \\ \beta \\ \dots \\ n \end{array} \right)^{\dagger} = \begin{array}{c} n \\ \dots \\ -\alpha \\ -\beta \\ \dots \\ m \end{array} , \quad \left(\begin{array}{c} \text{1} \\ | \end{array} \right)^{\dagger} = \begin{array}{c} \text{2} \\ | \end{array} \quad (24)$$

These generators can be composed in parallel (\otimes) and sequentially (\circ), and the resulting diagrams are governed by the rewrite rules in Figure 2, wherein addition is modulo 2π . The fusion rule (f) applies

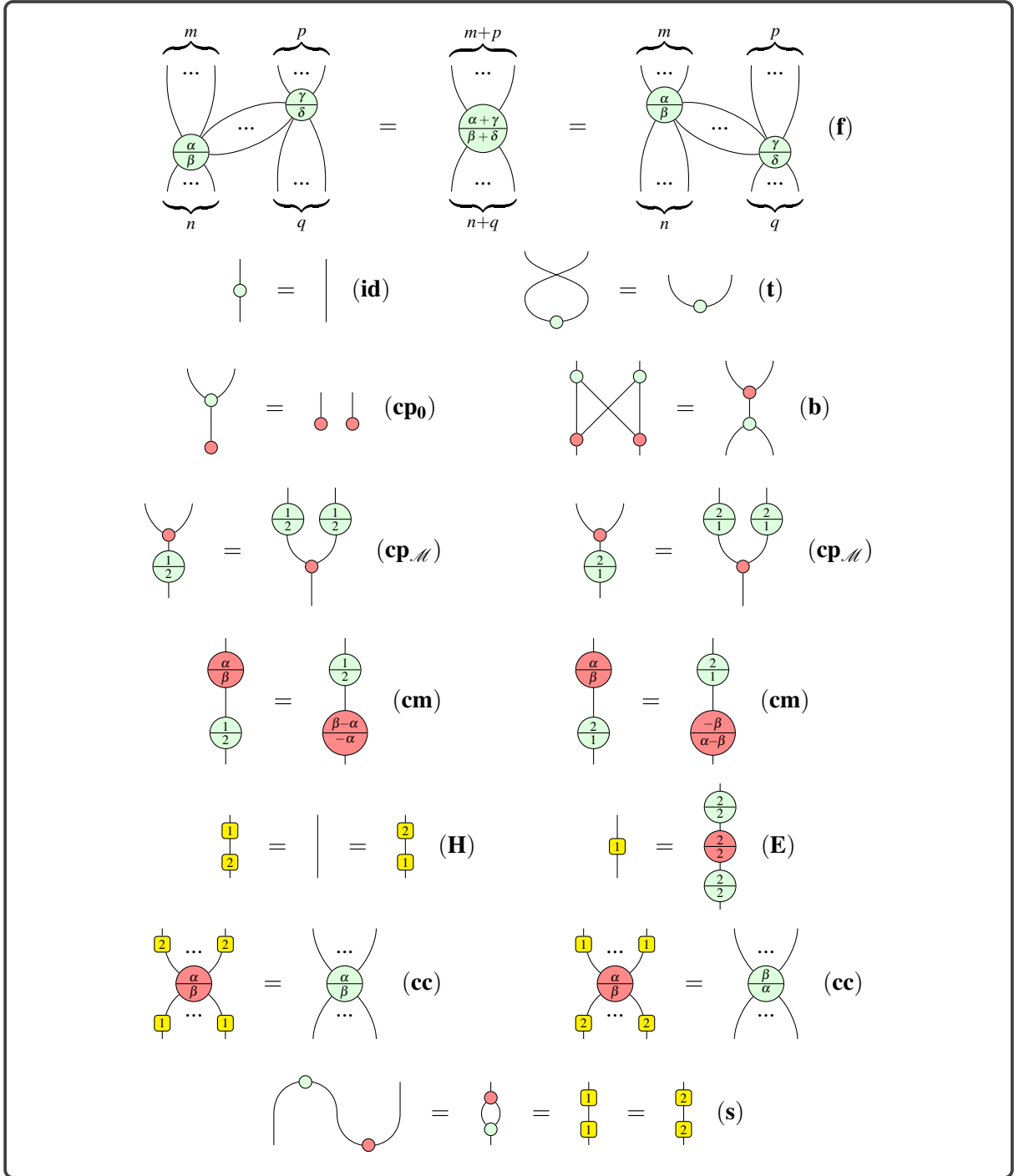


Figure 2: Rewrite rules for the qutrit ZX-calculus where spiders are interpreted as tensors over \mathbb{C} .

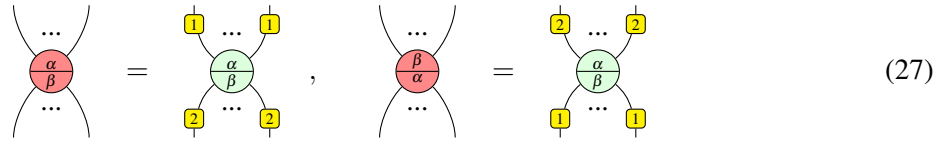
to spiders of the same colour connected by at least one wire. Importantly, all the rules hold under taking adjoints, where for diagrams D and E we have:

$$(D \otimes E)^\dagger = D^\dagger \otimes E^\dagger \quad (25)$$

$$(D \circ E)^\dagger = D^\dagger \circ E^\dagger \quad (26)$$

Furthermore, it can be derived that all but the commutation equations (**cm**) and the colour change equations (**cc**) continue to hold when the roles of green and red (i.e. Z and X) are interchanged. For these four exceptions, however, analogous equations can be derived from the existing ones; for example, the corresponding colour change equations will be relevant for us later.

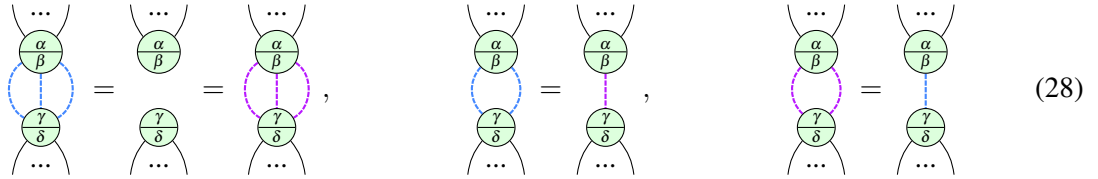
Proposition B.2. The following equations are derivable in the qutrit ZX-calculus:



$$(27)$$

Proof. Add H - and H^\dagger -boxes to both sides of the original colour change equations in such a way that we can then cancel Hadamards on the legs of the red spiders via (**H**). \square

Lemma B.3. / Lemma 3.1. The following equations hold in the qutrit ZX-calculus:



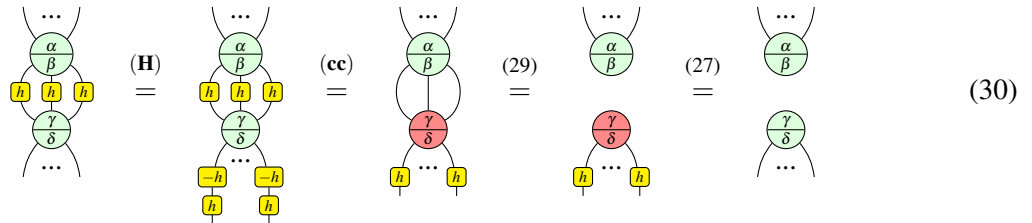
$$(28)$$

Proof. It is shown in Lemma 2.8 [9] that the qutrit ZX-calculus satisfies the following ‘Hopf law’:



$$(29)$$

Therefore we can argue as follows, for $h \in \{1, 2\}$:



$$(30)$$

\square

C Local Pivot in Qutrit ZX

Next we prove the local pivot equality from Theorem 3.7. We require the following: since (E) holds under taking adjoints, and swapping the roles of red and green, we have:

$$\begin{array}{c} \text{1} \\ \text{---} \end{array} = \begin{array}{c} \text{2} \\ \text{---} \end{array} = \begin{array}{c} \text{2} \\ \text{---} \end{array}, \quad \begin{array}{c} \text{2} \\ \text{---} \end{array} = \begin{array}{c} \text{1} \\ \text{---} \end{array} = \begin{array}{c} \text{1} \\ \text{---} \end{array} \quad (31)$$

Theorem C.1. / Theorem 3.7. Given $a \in \mathbb{Z}_3$ and a graph state (G, W) containing connected nodes i and j , define the following:

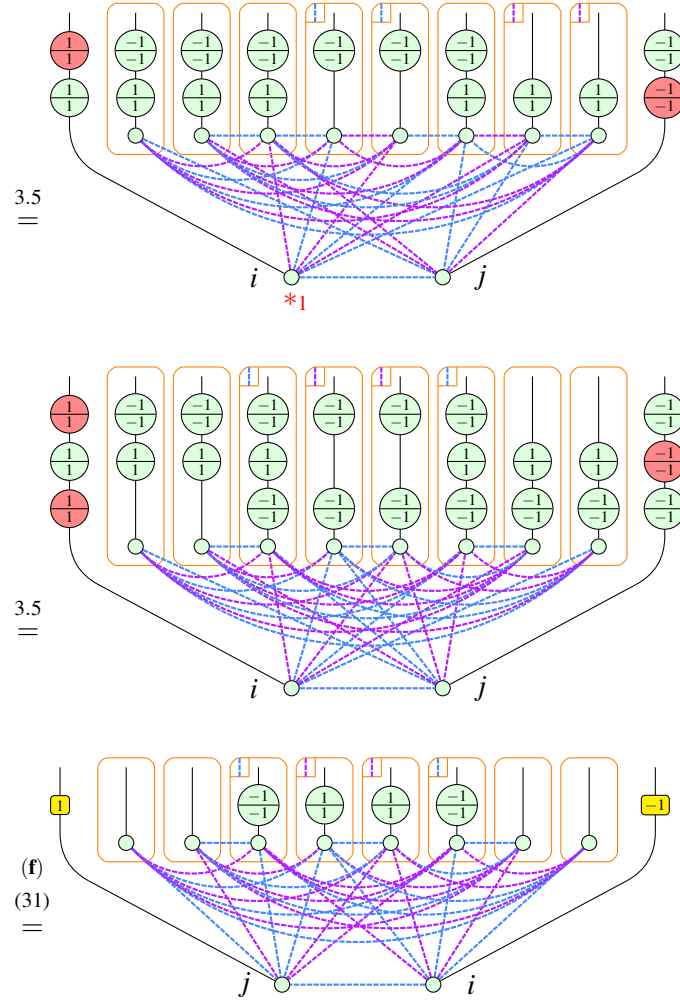
- $N_{=}(i, j) := \{k \in N(i) \cap N(j) \mid w_{k,i} = w_{k,j}\}$
- $N_{\neq}(i, j) := \{k \in N(i) \cap N(j) \mid w_{k,i} \neq w_{k,j}\}$

Then the following equation relates G and its proper a -local pivot along ij :

$$\begin{array}{c} \text{---} \end{array} \begin{array}{c} i \\ \text{---} \end{array} \begin{array}{c} j \\ \text{---} \end{array} \begin{array}{c} N_{=}(i,j) \\ \text{---} \end{array} \begin{array}{c} N_{\neq}(i,j) \\ \text{---} \end{array} \dots \begin{array}{c} \text{---} \end{array} = \begin{array}{c} \text{---} \end{array} \begin{array}{c} j \\ \text{---} \end{array} \begin{array}{c} i \\ \text{---} \end{array} \begin{array}{c} N_{=}(i,j) \\ \text{---} \end{array} \begin{array}{c} N_{\neq}(i,j) \\ \text{---} \end{array} \begin{array}{c} \text{---} \end{array} \begin{array}{c} G \wedge_a ij \end{array}$$

Proof. There are four cases ($a, w_{i,j} \in \{1, 2\}$), which split into two pairs of symmetric cases: $a = w_{i,j}$ and $a \neq w_{i,j}$. We show just one case - the 1-local pivot along ij of weight 1 - the remaining cases being analogous. We again employ the !-notation. An asterisk $*_a$ next to a spider means that at the next step an a -local complementation will be performed at this spider.

$$\begin{array}{c} \text{---} \end{array} \begin{array}{c} i \\ \text{---} \end{array} \begin{array}{c} j \\ \text{---} \end{array} \begin{array}{c} *1 \end{array} = \begin{array}{c} \text{---} \end{array} \begin{array}{c} i \\ \text{---} \end{array} \begin{array}{c} j \\ \text{---} \end{array} \begin{array}{c} *-1 \end{array}$$

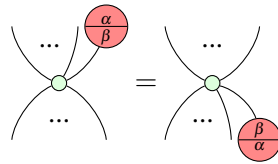


□

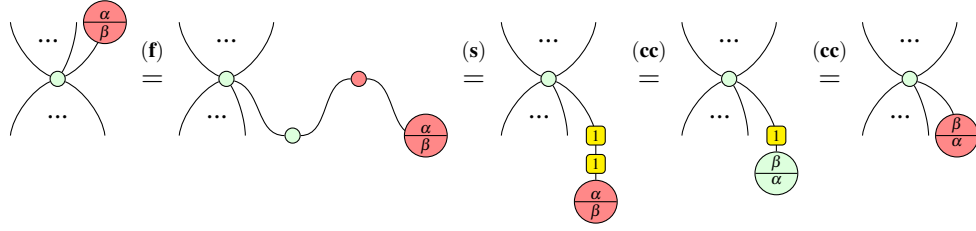
D Spider Elimination Rules for Qutrit ZX

Now we prove the three elimination theorems for \mathcal{P} -, \mathcal{N} - and \mathcal{M} -spiders. All three require the following two lemmas.

Lemma D.1. The following ‘leg flip’ equation holds in the qutrit ZX-calculus. Moreover, it holds with the roles of green and red swapped.

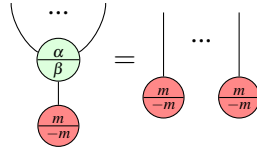


Proof.

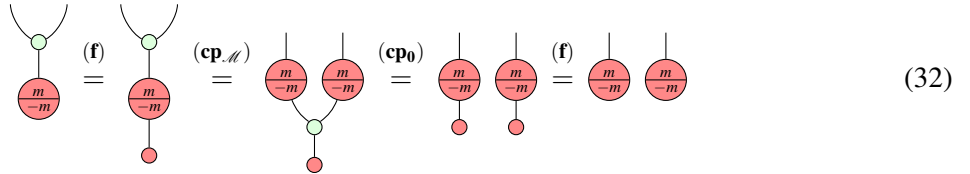


□

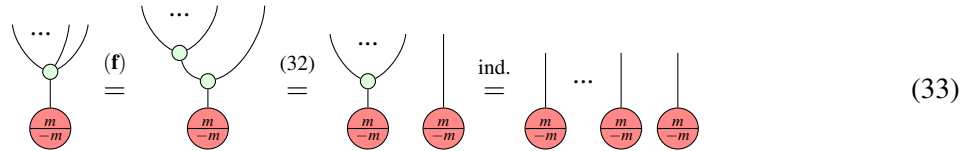
Lemma D.2. The following more substantial ‘ \mathcal{M} -copy’ rule holds in the qutrit ZX-calculus, for any \mathcal{M} -spider state (i.e. $m \in \{0, 1, 2\}$ below). Moreover, it holds with the roles of green and red swapped.



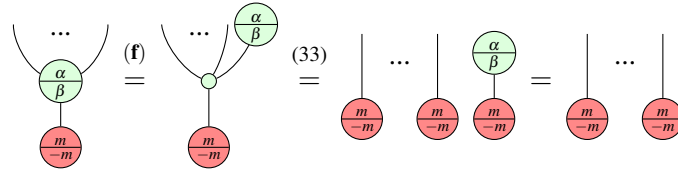
Proof. First we prove that an \mathcal{M} -spider with non-trivial phase (i.e. $m \in \{1, 2\}$) satisfies a copy rule exactly like the rule **(cp₀)**:



Then we prove the case $\alpha = \beta = 0$ by induction:



Which finally allows us to prove the full statement, where the last equality is just dropping the scalar term:



□

D.1 \mathcal{P} -spider Elimination

The \mathcal{P} -spider elimination theorem additionally requires a lemma allowing us to turn a \mathcal{P} -spider state of one colour into a \mathcal{P} -spider state of the other. We will use this lemma in its adjoint form in the proof of the main theorem.

Lemma D.3. The following rule holds in the qutrit ZX-calculus:

$$\begin{array}{c} | \\ \textcircled{\frac{p}{p}} \end{array} = \begin{array}{c} | \\ \textcircled{\frac{-p}{-p}} \end{array}$$

Proof.

$$\begin{array}{c} | \\ \textcircled{\frac{p}{p}} \end{array} \stackrel{(\text{cc})}{=} \begin{array}{c} | \\ \textcircled{\frac{p}{p}} \end{array} \stackrel{(31)}{=} \begin{array}{c} \textcircled{\frac{-p}{-p}} \\ | \\ \textcircled{\frac{p}{p}} \end{array} \stackrel{(\text{f})}{=} \begin{array}{c} \textcircled{\frac{-p}{-p}} \\ | \\ \textcircled{\frac{-p}{-p}} \end{array} \stackrel{D.2}{=} \begin{array}{c} | \\ \textcircled{\frac{-p}{-p}} \end{array} \stackrel{(\text{f})}{=} \begin{array}{c} | \\ \textcircled{\frac{-p}{-p}} \end{array}$$

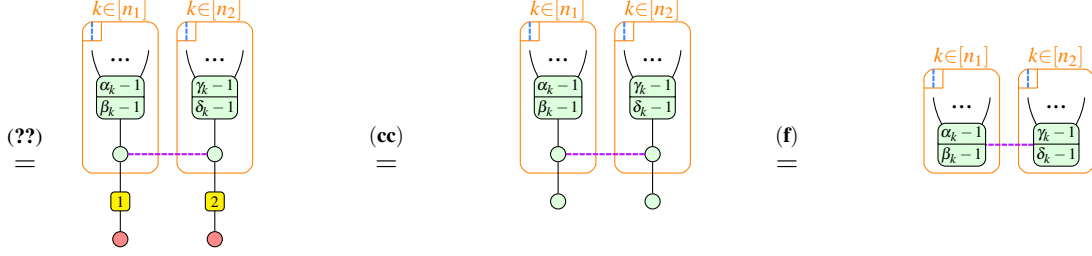
□

Theorem D.4. / Theorem 3.8. Given any graph-like ZX-diagram containing an interior \mathcal{P} -spider x with phase $\textcircled{\frac{p}{p}}$ for $p \in \{1, 2\}$, suppose we perform a p -local complementation at x . Then the new ZX-diagram is related to the old one by the following equality:

$$\begin{array}{c} \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \quad \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{p}{p}} \end{array} = \begin{array}{c} \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \quad \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{-p}{-p}} \end{array}$$

Proof. For clarity of presentation we only show the case $\textcircled{\frac{p}{p}} = \textcircled{\frac{1}{1}}$, the other case being near-identical.

$$\begin{array}{c} \begin{array}{c} k \in [K_1] \\ \dots \\ \alpha_k \\ \beta_k \end{array} \quad \begin{array}{c} k \in [K_2] \\ \dots \\ \gamma_k \\ \delta_k \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array} \stackrel{(\text{f})}{=} \begin{array}{c} \begin{array}{c} k \in [n_1] \\ \dots \\ \alpha_k \\ \beta_k \end{array} \quad \begin{array}{c} k \in [n_2] \\ \dots \\ \gamma_k \\ \delta_k \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array} \stackrel{3.5}{=} \begin{array}{c} \begin{array}{c} k \in [n_1] \\ \dots \\ \alpha_k \\ \beta_k \end{array} \quad \begin{array}{c} k \in [n_2] \\ \dots \\ \gamma_k \\ \delta_k \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array} \stackrel{D.3}{=} \begin{array}{c} \begin{array}{c} k \in [n_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \quad \begin{array}{c} k \in [n_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array} \stackrel{D.1}{=} \begin{array}{c} \begin{array}{c} k \in [n_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \quad \begin{array}{c} k \in [n_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array} \stackrel{D.2}{=} \begin{array}{c} \begin{array}{c} k \in [n_1] \\ \dots \\ \alpha_k - 1 \\ \beta_k - 1 \end{array} \quad \begin{array}{c} k \in [n_2] \\ \dots \\ \gamma_k - 1 \\ \delta_k - 1 \end{array} \\ \diagdown \quad \diagup \\ \textcircled{\frac{1}{1}} \end{array}$$



□

D.2 \mathcal{N} -spider Elimination

Proving the corresponding \mathcal{N} -spider elimination theorem again requires a lemma allowing us to turn an \mathcal{N} -spider state of one colour into an \mathcal{N} -spider state of the other.

Lemma D.5. The following rules hold in the qutrit ZX-calculus:

$$\begin{array}{c} | \\ \textcircled{\frac{0}{1}} \end{array} = \begin{array}{c} | \\ \textcircled{\frac{2}{0}} \end{array}, \quad \begin{array}{c} | \\ \textcircled{\frac{0}{2}} \end{array} = \begin{array}{c} | \\ \textcircled{\frac{0}{1}} \end{array}, \quad \begin{array}{c} | \\ \textcircled{\frac{1}{0}} \end{array} = \begin{array}{c} | \\ \textcircled{\frac{0}{2}} \end{array}, \quad \begin{array}{c} | \\ \textcircled{\frac{2}{0}} \end{array} = \begin{array}{c} | \\ \textcircled{\frac{1}{0}} \end{array}$$

Proof. The key observation is that for any green \mathcal{N} -spider state with phase $\frac{n}{n'}$, we have a choice of two colour change rules which we could use to turn it into a red \mathcal{N} -spider state with a H - or H^\dagger -box on top:

$$\begin{array}{c} \textcircled{\frac{n}{n'}} \\ \textcircled{\frac{n}{n'}} \end{array} \stackrel{(cc)}{=} \begin{array}{c} \textcircled{\frac{n}{n'}} \\ \textcircled{\frac{n}{n'}} \end{array} \stackrel{(cc)}{=} \begin{array}{c} \textcircled{\frac{n}{n'}} \\ \textcircled{\frac{n}{n'}} \end{array}$$

Of these two choices, exactly one has a decomposition of the H -/ H^\dagger -box as in (31) that allows the bottom two red spiders to fuse into an \mathcal{M} -spider, which we can then move past the green spider above it via D.2. For brevity we only show the case $\frac{n}{n'} = \frac{0}{1}$:

$$\begin{array}{c} | \\ \textcircled{\frac{0}{1}} \end{array} \stackrel{(cc)}{=} \begin{array}{c} \textcircled{\frac{0}{1}} \\ \textcircled{\frac{0}{1}} \end{array} \stackrel{(31)}{=} \begin{array}{c} \textcircled{\frac{1}{1}} \\ \textcircled{\frac{1}{1}} \\ \textcircled{\frac{1}{1}} \\ \textcircled{\frac{0}{1}} \end{array} \stackrel{(f)}{=} \begin{array}{c} \textcircled{\frac{1}{1}} \\ \textcircled{\frac{1}{1}} \\ \textcircled{\frac{1}{2}} \end{array} \stackrel{D.2}{=} \begin{array}{c} \textcircled{\frac{1}{1}} \\ \textcircled{\frac{1}{2}} \end{array} \stackrel{(f)}{=} \begin{array}{c} | \\ \textcircled{\frac{2}{0}} \end{array}$$

□

Corollary D.6. The following equations hold in the qutrit ZX-calculus:

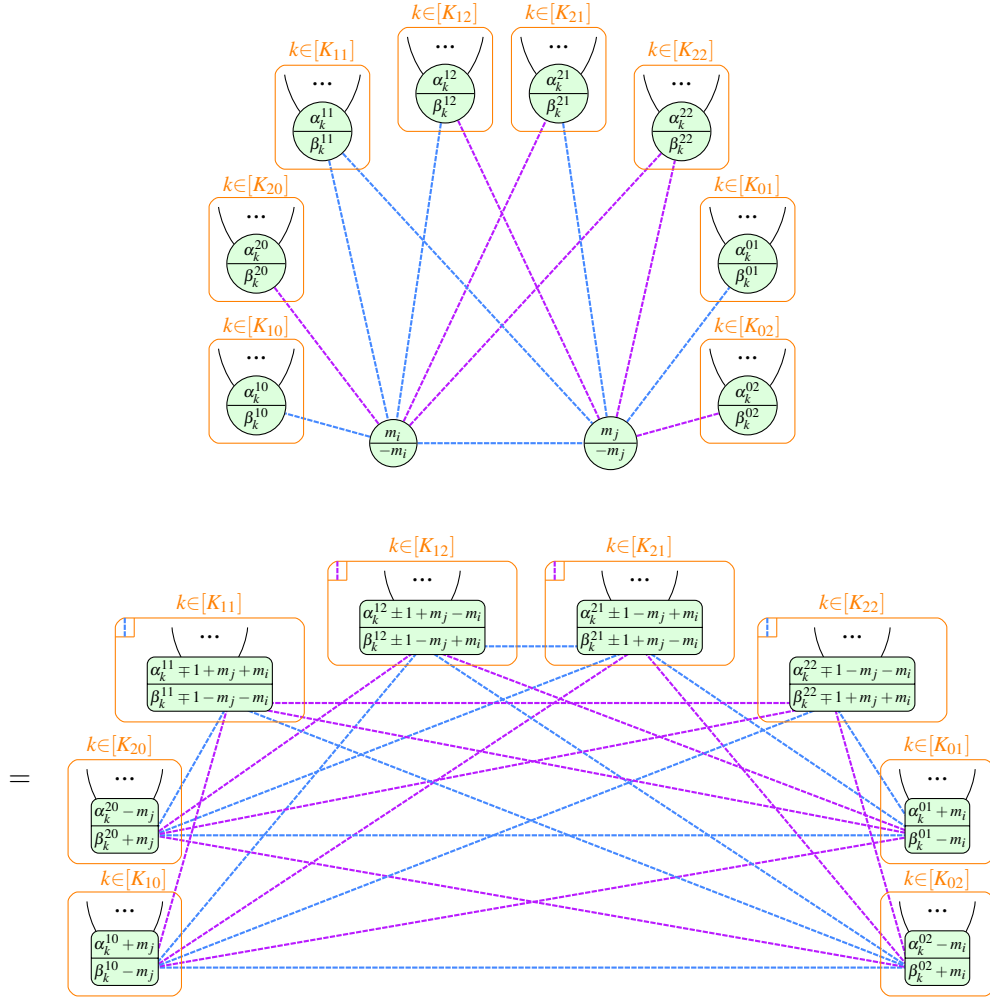
$$\begin{array}{c} \textcircled{\frac{0}{n}} \\ \textcircled{-n} \end{array} = \begin{array}{c} \textcircled{\frac{2}{1}} \\ | \end{array}, \quad \begin{array}{c} \textcircled{\frac{n}{0}} \\ \textcircled{-n} \end{array} = \begin{array}{c} \textcircled{\frac{1}{2}} \\ | \end{array}$$

Figure 1 shows a sequence of five diagrams illustrating the reduction of a 2x2 matrix to a scalar. The first diagram shows a 2x2 matrix with top row [0, 1] and bottom row [-1, -1]. The second diagram shows the result of adding the bottom row to the top row, resulting in [-1, 0]. The third diagram shows the result of multiplying the top row by -1, resulting in [1, 0]. The fourth diagram shows the result of adding the top row to the bottom row, resulting in [1, 0] and [0, -1]. The fifth diagram shows the final result, a scalar 2.

□

D.3 \mathcal{M} -spider Elimination

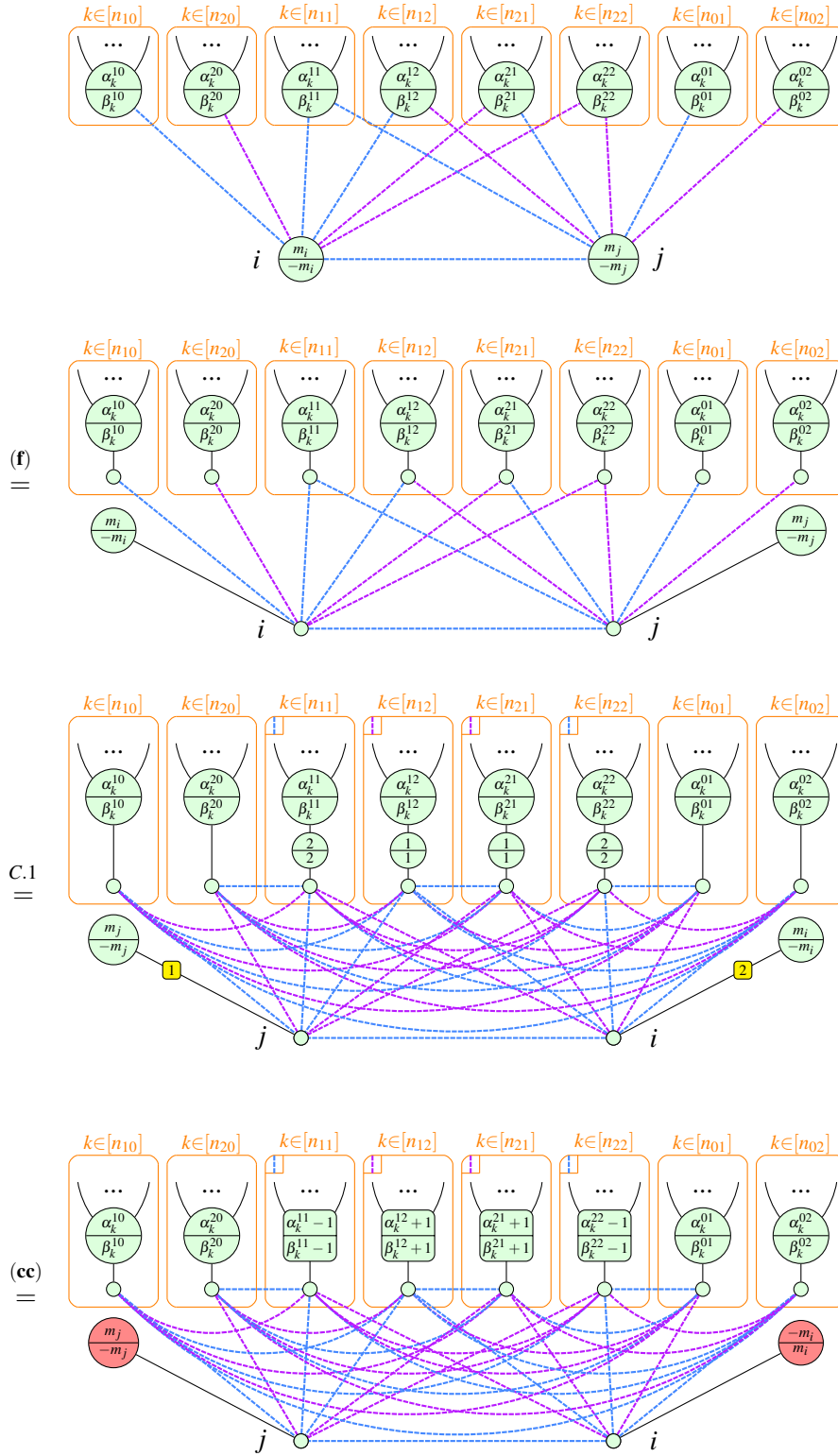
Theorem D.8. / Theorem 3.10. Given any graph-like ZX-diagram containing two interior \mathcal{M} -spiders i and j connected by edge ij of weight $w_{i,j} =: w \in \{1, 2\}$, suppose we perform a proper $\pm w$ -local pivot along ij (both choices give the same result). For the case $w = 1$, the new ZX-diagram is related to the old one by the following equality:

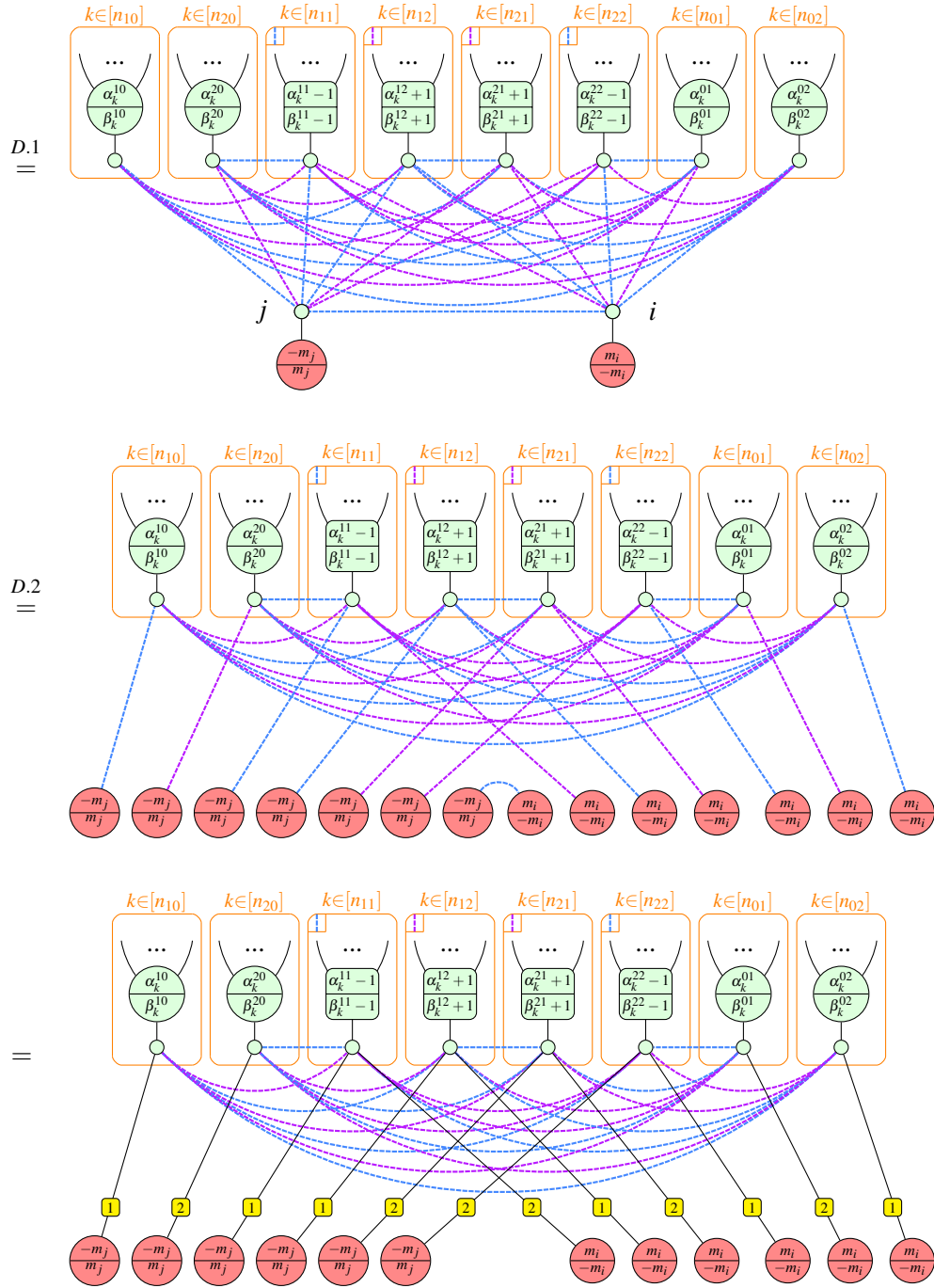


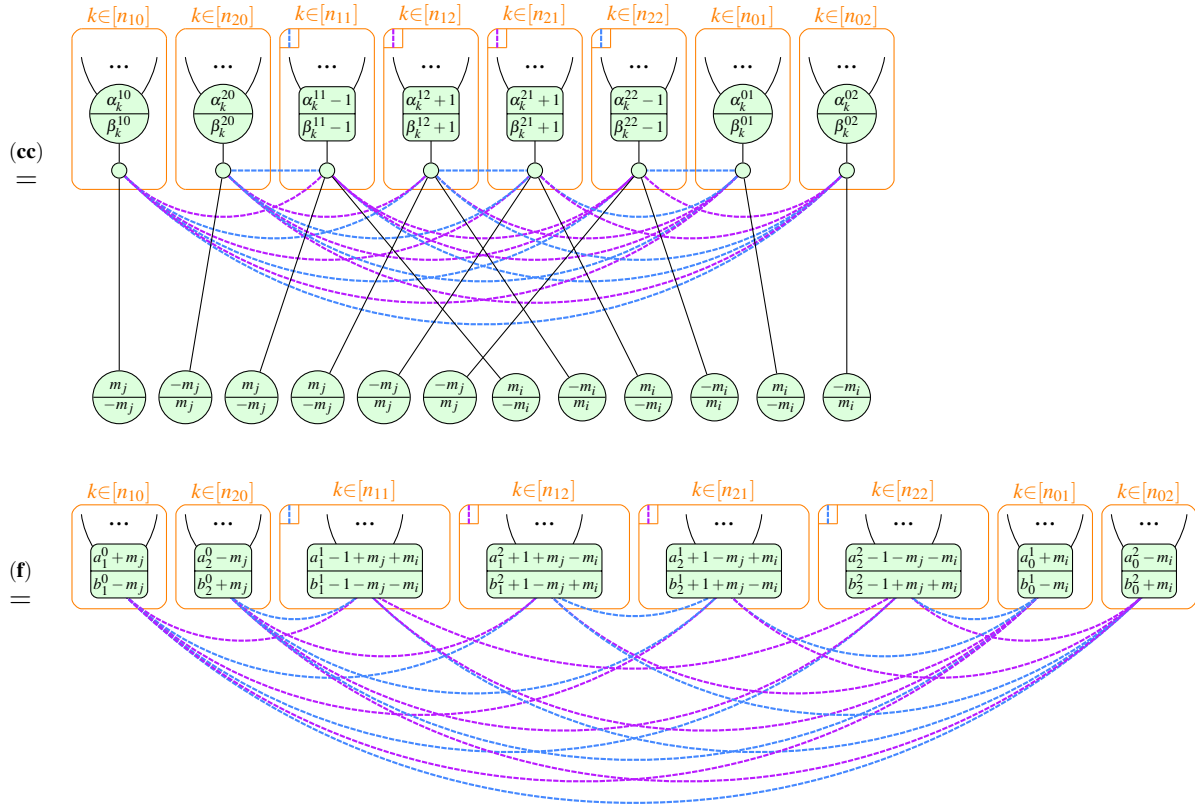
The case $w = 2$ differs only as follows: in the first diagram (the left hand side of the equation), the edge ij is purple (by definition), while in the lower diagram (the right hand side of the equation), a $\pm 2 = \mp 1$ will replace all occurrences of ± 1 , and the roles of purple and blue will be swapped throughout.

Proof. We show the case where $w_{i,j} =: w = 1$, with the case $w = 2$ being completely analogous. We can choose either a proper 1-local pivot or a proper 2-local pivot; both give the same result. Here we only

show the former:







□

E \pm Boxes in the ZX Calculus

We begin with proofs of the translations of the matrix $T_{\pm}^{(q)}$ into the ZX-calculus.

Proposition E.1. / **Propositions 4.1, ??.** The following equalities hold up to a scalar under the standard interpretation:

$$\left[\begin{array}{c} \pm \frac{\pi}{2} \\ | \end{array} \right] \simeq \left[\begin{array}{c} \pm \\ | \end{array} \right]_{q=2}, \quad \left[\begin{array}{c} \pm 1 \\ | \end{array} \right] \simeq \left[\begin{array}{c} \pm \\ | \end{array} \right]_{q=3}, \quad \left[\begin{array}{c} \pi \quad \text{---} \quad \pi \\ | \quad \text{---} \quad | \end{array} \right] \simeq \left[\begin{array}{c} \pm \\ | \end{array} \right]_{q=4}$$

Proof. Recalling $\omega = e^{i\frac{2\pi}{3}}$, the standard interpretations of phase gates in matrix form are:

$$\left[\begin{array}{c} \alpha \\ | \end{array} \right] = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}, \quad \left[\begin{array}{c} \alpha \\ | \end{array} \right] = \frac{1}{2} \begin{pmatrix} 1 + e^{i\alpha} & 1 - e^{i\alpha} \\ 1 - e^{i\alpha} & 1 + e^{i\alpha} \end{pmatrix}, \quad \left[\begin{array}{c} \alpha \\ \beta \\ | \end{array} \right] = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{i\alpha} & 0 \\ 0 & 0 & e^{i\beta} \end{pmatrix}$$

$$\left[\left[\begin{array}{c} \alpha \\ \beta \end{array} \right] \right] = \frac{1}{3} \begin{pmatrix} 1 + e^{i\alpha} + e^{i\beta} & 1 + \bar{\omega}e^{i\alpha} + \omega e^{i\beta} & 1 + \omega e^{i\alpha} + \bar{\omega}e^{i\beta} \\ 1 + \omega e^{i\alpha} + \bar{\omega}e^{i\beta} & 1 + e^{i\alpha} + e^{i\beta} & 1 + \bar{\omega}e^{i\alpha} + \omega e^{i\beta} \\ 1 + \bar{\omega}e^{i\alpha} + \omega e^{i\beta} & 1 + \omega e^{i\alpha} + \bar{\omega}e^{i\beta} & 1 + e^{i\alpha} + e^{i\beta} \end{pmatrix}$$

So in the simplest case $q = 2$ it is fairly straightforward to see that:

$$\left[\left[\begin{array}{c} \pm \frac{\pi}{2} \end{array} \right] \right] = \frac{1}{2} \begin{pmatrix} 1 \pm i & 1 \mp i \\ 1 \mp i & 1 \pm i \end{pmatrix} = \frac{\sqrt{2}}{2} e^{\mp i \frac{\pi}{4}} \begin{pmatrix} \pm i & 1 \\ 1 & \pm i \end{pmatrix} = \frac{\sqrt{2}}{2} e^{\mp i \frac{\pi}{4}} \left[\left[\begin{array}{c} \pm \end{array} \right] \right]_{q=2} \quad (34)$$

The next case $q = 3$ is proved similarly:

$$\begin{aligned} \left[\left[\begin{array}{c} \pm 1 \\ \pm 1 \\ \pm 1 \end{array} \right] \right] &= \frac{1}{3} \begin{pmatrix} 1 + e^{\pm i \frac{2\pi}{3}} + e^{\pm i \frac{2\pi}{3}} & 1 + \bar{\omega}e^{\pm i \frac{2\pi}{3}} + \omega e^{\pm i \frac{2\pi}{3}} & 1 + \omega e^{\pm i \frac{2\pi}{3}} + \bar{\omega}e^{\pm i \frac{2\pi}{3}} \\ 1 + \omega e^{\pm i \frac{2\pi}{3}} + \bar{\omega}e^{\pm i \frac{2\pi}{3}} & 1 + e^{\pm i \frac{2\pi}{3}} + e^{\pm i \frac{2\pi}{3}} & 1 + \bar{\omega}e^{\pm i \frac{2\pi}{3}} + \omega e^{\pm i \frac{2\pi}{3}} \\ 1 + \bar{\omega}e^{\pm i \frac{2\pi}{3}} + \omega e^{\pm i \frac{2\pi}{3}} & 1 + \omega e^{\pm i \frac{2\pi}{3}} + \bar{\omega}e^{\pm i \frac{2\pi}{3}} & 1 + e^{\pm i \frac{2\pi}{3}} + e^{\pm i \frac{2\pi}{3}} \end{pmatrix} \\ &= \frac{1}{3} \begin{pmatrix} \sqrt{3}e^{\pm i \frac{\pi}{2}} & \sqrt{3}e^{\mp i \frac{\pi}{6}} & \sqrt{3}e^{\mp i \frac{\pi}{6}} \\ \sqrt{3}e^{\mp i \frac{\pi}{6}} & \sqrt{3}e^{\pm i \frac{\pi}{2}} & \sqrt{3}e^{\mp i \frac{\pi}{6}} \\ \sqrt{3}e^{\mp i \frac{\pi}{6}} & \sqrt{3}e^{\mp i \frac{\pi}{6}} & \sqrt{3}e^{\pm i \frac{\pi}{2}} \end{pmatrix} \\ &= \frac{\sqrt{3}}{3} e^{\mp i \frac{\pi}{6}} \begin{pmatrix} e^{\pm i \frac{2\pi}{3}} & 1 & 1 \\ 1 & e^{\pm i \frac{2\pi}{3}} & 1 \\ 1 & 1 & e^{\pm i \frac{2\pi}{3}} \end{pmatrix} \\ &= \frac{\sqrt{3}}{3} e^{\mp i \frac{\pi}{6}} \left[\left[\begin{array}{c} \pm \end{array} \right] \right]_{q=3} \end{aligned} \quad (35)$$

For the other qubit case $q = 4$ we first note:

$$\left[\left[\begin{array}{c} \frac{\pi}{2} \\ \frac{\pi}{2} \end{array} \right] \right] = \left[\left[\begin{array}{c} \frac{\pi}{2} \\ \frac{\pi}{2} \end{array} \right] \right] = \left[\left[\begin{array}{c} \frac{\pi}{2} \end{array} \right] \right] \left[\left[\begin{array}{c} \frac{\pi}{2} \end{array} \right] \right] = \frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (36)$$

Then using the standard interpretation for spiders (Definition ??) we decompose the diagram in such a way that we can apply the standard interpretation:

$$\begin{aligned} \left[\left[\begin{array}{c} \pi \\ \pi \end{array} \right] \right] &= \left[\left[\begin{array}{c} \pi \\ \pi \end{array} \right] \right] = \left(\left[\left[\begin{array}{c} \pi \end{array} \right] \right] \otimes \left[\left[\begin{array}{c} \pi \end{array} \right] \right] \right) \left(\left[\left[\begin{array}{c} \pi \end{array} \right] \right] \otimes \left[\left[\begin{array}{c} \pi \end{array} \right] \right] \right) \left(\left[\left[\begin{array}{c} \pi \end{array} \right] \right] \otimes \left[\left[\begin{array}{c} \pi \end{array} \right] \right] \right) \\ &= \frac{\sqrt{2}}{8} \begin{pmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{pmatrix} \end{aligned}$$

$$= \frac{\sqrt{2}}{8} \left[\left[\begin{array}{c} | \\ \boxed{\pm} \\ | \end{array} \right] \right]_{q=4}$$

□