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 - namely *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 stabilizer fragment of the ZX-calculus. Building on previous work on efficient simplification of qubit stabilizer diagrams, we present simplifying rewrites for the case of qutrits, which are of independent interest in the field of quantum circuit optimisation. Finally, we look at the specific examples of evaluating the Jones polynomial and of counting graph-colourings. Our exposition further champions the ZX calculus as a suitable and unifying language for studying the complexity of a broad range of classical and quantum problems.

1 Introduction

A plethora of hard problems of interest to physics and computer science regard interacting many-body multi-state systems - both classical and 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 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 [22]. The generators have a concrete representation as tensors over a (semi)ring and they obey a set of rewrite rules which respect semantics [21]. 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 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 [11].

In the context of quantum computing, the Gottesman-Knill theorem states that stabilizer (or Clifford) circuits can be simulated *efficiently* on classical computers [1]. These circuits can be expressed by the *stabilizer* 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, though the motivation for studying stabilizer diagrams stems from quantum computation, the result that stabilizer ZX diagrams can be simplified efficiently has broader applicability. This is because ZX diagrams can express arbitrary linear maps; every quantum circuit is a ZX diagram but not vice versa.

Submitted to:
QPL 2021

Indeed, 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, and this number can be expressed as 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 the decision problem 2SAT is easy but the counting version #2SAT is hard, and why this is not the case for 3SAT, where both deciding and counting are hard.

Such observations make apparent the unifying power of diagrammatic reasoning for studying the complexity of problem families. Building on the spirit of [5], we treat the ZX-calculus as a library comprising out-of-the-box 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 simplification of qubit stabilizer ZX diagrams are called *local complementation* and *pivot*, the latter being composition of three of the former [9]. In this work, we recall the qutrit version of local complementation and derive the corresponding pivot rule, which implies that qutrit stabilizer 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 stabilizer 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. Importantly, note that keeping track of multiplicative scalar factors that arise under diagram rewriting can be done efficiently. We are thus justified in omitting them for the purposes of this work. We also emphasise that the qutrit *elimination* rewrite rules we derive in Section 3 have independent ramifications for circuit optimisation, though these are not explored here.

2 Simplifying Qubit ZX-Diagrams

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

2.1 The Qubit ZX-Calculus

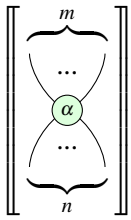
Here we give a quick introduction to the qubit ZX-calculus. The qubit ZX-calculus is a diagrammatic language for quantum mechanics generated by *spiders*. The spider legs, or *wires*, represent vector spaces of dimension $d = 2$. Diagrams are read bottom-to-top; bottom open wires (not connected to anything) 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

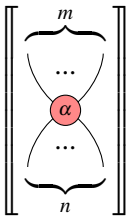
(semi)ring over which the spiders are interpreted as tensors. For spiders S and S', we denote these as

$$[\![S \otimes S']\!] = [\![S]\!] \otimes [\![S']\!] \quad , \quad [\![S \circ S']\!] = [\![S]\!] \cdot [\![S']\!] \tag{1}$$

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 *standard representation* of the spider generators as tensors over $\mathbb{C}$ is:

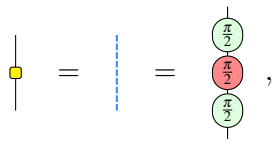


$$= |0\rangle^{\otimes m} \langle 0|^{\otimes n} + e^{i\alpha} |1\rangle^{\otimes m} \langle 1|^{\otimes n},$$



$$= |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |-\rangle^{\otimes m} \langle -|^{\otimes n} \tag{2}$$

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*:



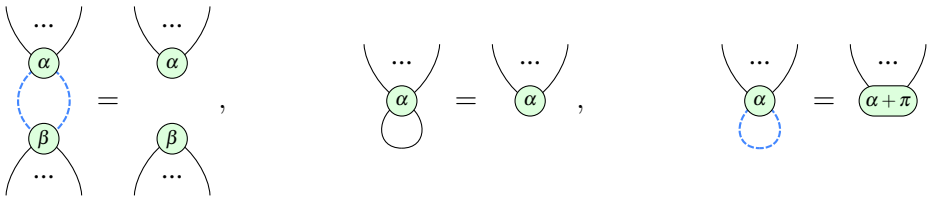
$$[\![\; \square \;]\!] \simeq |0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1| \tag{3}$$

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.**??** in Appendix **??**) 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 by 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 are invariant under deformations of the network as long as the inter-spider connectivity is respected.

2.2 Stabilizer Qubit ZX Diagrams

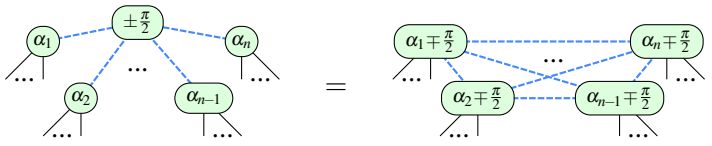
The *stabilizer fragment* of the calculus consists of all diagrams in which all phases are $\alpha = \frac{\pi n}{2}$, $n \in \mathbb{Z}$. In (Theorem 5.4, [9]) the authors give an efficient algorithm for simplifying any qubit ZX-diagram to an equivalent diagram with fewer spiders. The algorithm consists 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:

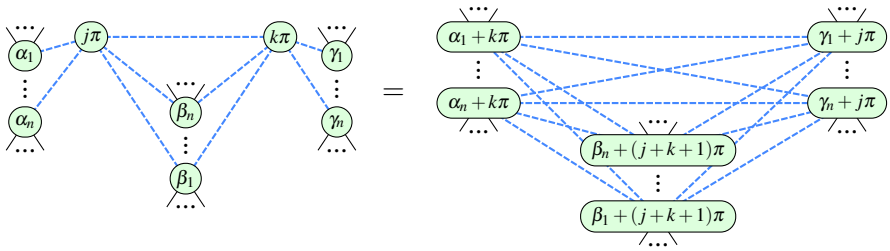


(4)

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



(5)



(6)

For more details, see (Section 4, [9]). Eq. (5) 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. (6) says we can remove any pair of spiders with phases in $\{0,\pi\}$ connected by a Hadamard edge at the cost of performing a pivot along said edge. After each application of (5) or (6), we can use (4) to remove any parallel Hadamard edges and ensure the diagram remains graph-like.

In particular, and importantly to this work, this algorithm will *efficiently* simplify any closed stabilizer diagram until it contains at most one spider, at which point the scalar it represents can be easily read off. By 'efficiently' we mean via a sequence of spider elimination rewrites whose length is polynomial in the initial number of spiders. Note each such rewrite updates only a polynomial number of edges in the diagram, which prevents an overwhelming memory cost of the simplification procedure.

3 Simplifying Qutrit ZX-Diagrams

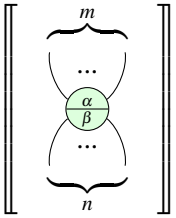
We now turn to the qutrit ZX-calculus and examine the analogous story to that of the previous subsection but now for the case where the dimension of the vector space carried by the wires is $d = 3$.

3.1 The Qutrit ZX-Calculus

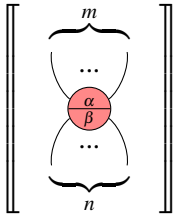
As in the qubit case, the qutrit ZX calculus concerns spiders connected by wires, but there are key differences, some subtler than others. Again, spiders come in two species, Z (green) and X (red), with the three-dimensional *Z-basis* denoted as $\{|0\rangle, |1\rangle, |2\rangle\}$. Let $\omega = e^{i\frac{2\pi}{3}}$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

$$|+\rangle = \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle) \ , \quad |\omega\rangle = \frac{1}{\sqrt{3}}(|0\rangle + \omega|1\rangle + \bar{\omega}|2\rangle) \ , \quad |\bar{\omega}\rangle = \frac{1}{\sqrt{3}}(|0\rangle + \bar{\omega}|1\rangle + \omega|2\rangle) \tag{7}$$

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



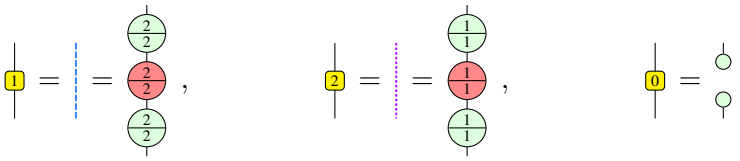
$$= |0\rangle^{\otimes m}\langle 0|^{\otimes n} + e^{i\alpha}|1\rangle^{\otimes m}\langle 1|^{\otimes n} + e^{i\beta}|2\rangle^{\otimes m}\langle 2|^{\otimes n} \tag{8}$$



$$= |+\rangle^{\otimes m} \langle +|^{\otimes n} + e^{i\alpha} |\omega\rangle^{\otimes m} \langle \omega|^{\otimes n} + e^{i\beta} |\bar{\omega}\rangle^{\otimes m} \langle \bar{\omega}|^{\otimes n} \tag{9}$$

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*). Those familar with the ZH-calculus should note that this notation is *not* analogous to the H-boxes therein.

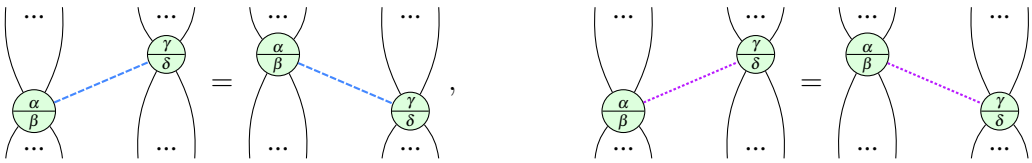


(10)

The last equation above says that in a graph-like diagram (which we will define shortly), a Hadamard edge decorated by a 0 is in fact not an edge at all, thanks to spider fusion. A very important difference from the qubit case is that in qutrit ZX-calculus there is no *plain* cap or cup:

(11)

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 [10]:



(12)

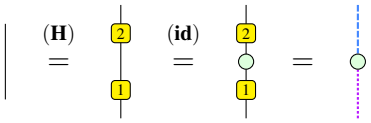
The full set of rules governing the qutrit ZX calculus is shown in Figure **??** in Appendix **??**.

3.2 Graph-Like Qutrit ZX Diagrams

A *graph-like qutrit ZX diagram* is one where every spider is green, spiders are only connected by blue Hadamard edges (*H-edges*) or their purple adjoints (*$H^\dagger$-edges*), 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

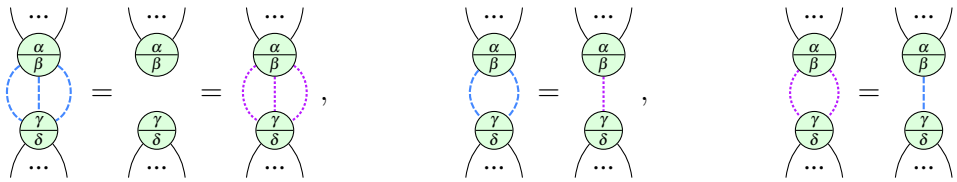
connected to at most one input or output wire. A graph-like qutrit ZX diagram is a *graph state* when every spider has zero phase (top and bottom) and is connected to an output.

Note the difference compared to the qubit case: we need not worry about self-loops beacuse 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:



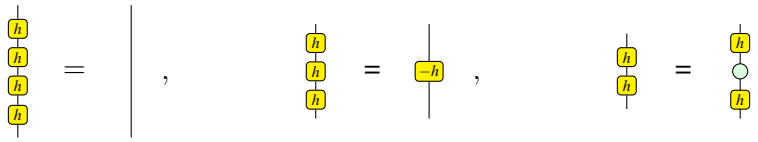
(13)

Indeed, we can show that every qutrit ZX-diagram is equivalent to a graph-like one. The following equations, derived in Appendix **??**, are vital to this:



(14)

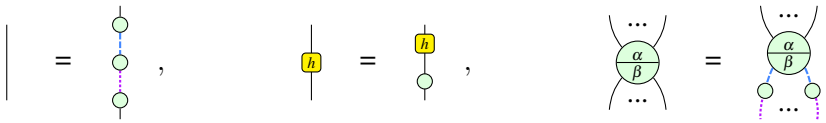
This justifies our notation for Hadamard gates: we can think of Hadamard edges as 1-weighted edges and their adjoints as 2-weighted edges, then work modulo 3, since every triple of parallel edges disappears. Where the previous equations relate single H- and $H^\dagger$-boxes across multiple edges, the next three relate multiple H- and $H^\dagger$- boxes on single edges. They hold for $h \in \{1,2\}$, and are proved via simple applications of rules (**id**), (**H**) and (**s**).



(15)

Proposition 3.1. 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 (15) 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 (13) where fusion is not possible, so that no plain edge connects two spiders. Apply (14) 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:



(16)

□

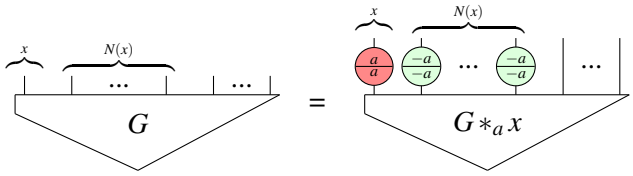
A graph state is described fully by its underlying multigraph, or equivalently by an adjacency matrix, where edges take weights in $\mathbb{Z}_3$ (Lemma 4.2, [20]). Nodes correspond to phaseless green spiders, edges of weight 1 correspond to Hadamard edges, and edges of weight 2 correspond to $H^\dagger$ edges. As in the qubit case, graph states admit a *local complementation* operation (Definition 2.6, [20]), though the effect is now slightly more complicated. We'll give the intuition after the formal definition:

Definition 3.2. Given $a \in \mathbb{Z}_3$ and a graph state G with adjacency matrix $W = (w_{i,j})$, the *a-local complentation* at node x is the new graph state $G *_a x$, whose adjacency matrix $W' = (w'_{i,j})$ is given by $w'_{i,j} = w_{i,j} + a w_{i,x} w_{j,x}$.

So only those edges between neighbours of node x are affected. Specifically, 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* colours, a-local complementation at x *decreases* $w_{i,j}$ by a. This is shown graphically below, and holds with the roles of blue and purple interchanged:

(17)

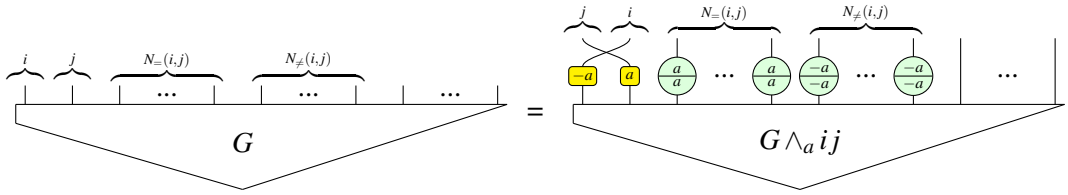
Theorem 3.3. (Theorem 4.4, Corollary 4.5, [20]) Given $a \in \mathbb{Z}_3$ and a graph state (G,W) containing a node x, let $N(x)$ denote the neighbours of x; that is, nodes i with weight $w_{i,x} \in \{1,2\}$. Then the following equality holds:



Definition 3.4. Given $a,b,c \in \mathbb{Z}_3$ and a graph state G containing nodes i and j, the (a,b,c)-*pivot* along ij is the new graph state $G \wedge_{(a,b,c)} ij := ((G *_a i) *_b j) *_c i$.

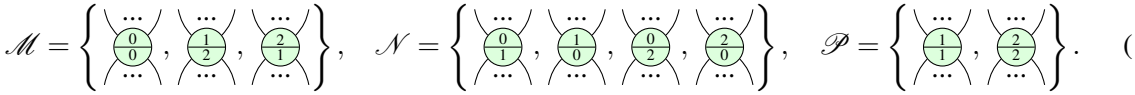
This pivot operation again leads to an equality, up to introducing some extra gates on outputs, whose proof is found in Appendix **??**. Here we shall only consider an $(a,-a,a)$-pivot along an edge ij of non-zero weight, for $a \in \{1,2\}$. We will call this a *proper a-pivot* along ij, and denote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

Theorem 3.5. Given $a \in \mathbb{Z}_3$ and a graph state (G,W) containing connected nodes i and j, define $N_=(i,j) := \{x \in N(i) \cap N(j) \mid w_{x,i} = w_{x,j}\}$ and $N_\neq(i,j) := \{x \in N(i) \cap N(j) \mid w_{x,i} \neq w_{x,j}\}$. Then the following equation relates G and its proper a-pivot along ij:



3.3 Qutrit Elimination Theorems

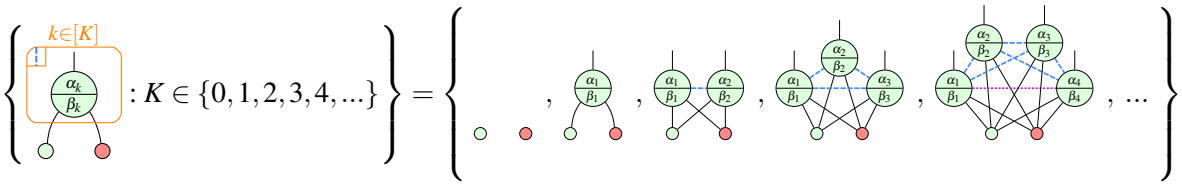
We classify spiders into three families exactly as in (Theorem 3.1, [20]):



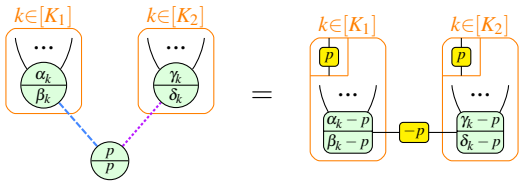
(18)

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

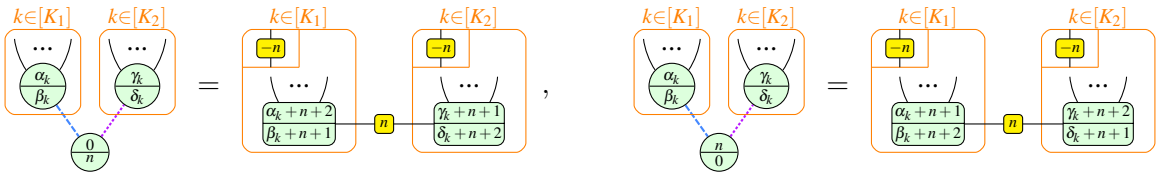
First, we define (a very small extension of) the *!-box* notation (pronounced *'bang-box'*), as introduced in [8] for general string diagrams. A !-box in a ZX-diagram is a compressed notation for a family of diagrams; the contents of the !-box, along with any wires into or out of it, are 'unfolded' (copied) $n \geq 0$ times and placed side-by-side. Following the style of [3], we also allow a parameter over which a !-box unfolds. We extend this notation as follows: in the top-left corner of the !-box we introduce a further diagram, which denotes that the spiders unfolded by the !-box have a complete graph imposed on them, whose edges take the form of this 'corner diagram'. In qutrit ZX-diagrams this notation will only be well-defined in 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 $[K] = \{1,...,K\}$, we have:



Theorem 3.6. Given any graph-like ZX-diagram containing an interior $\mathscr{P}$-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:

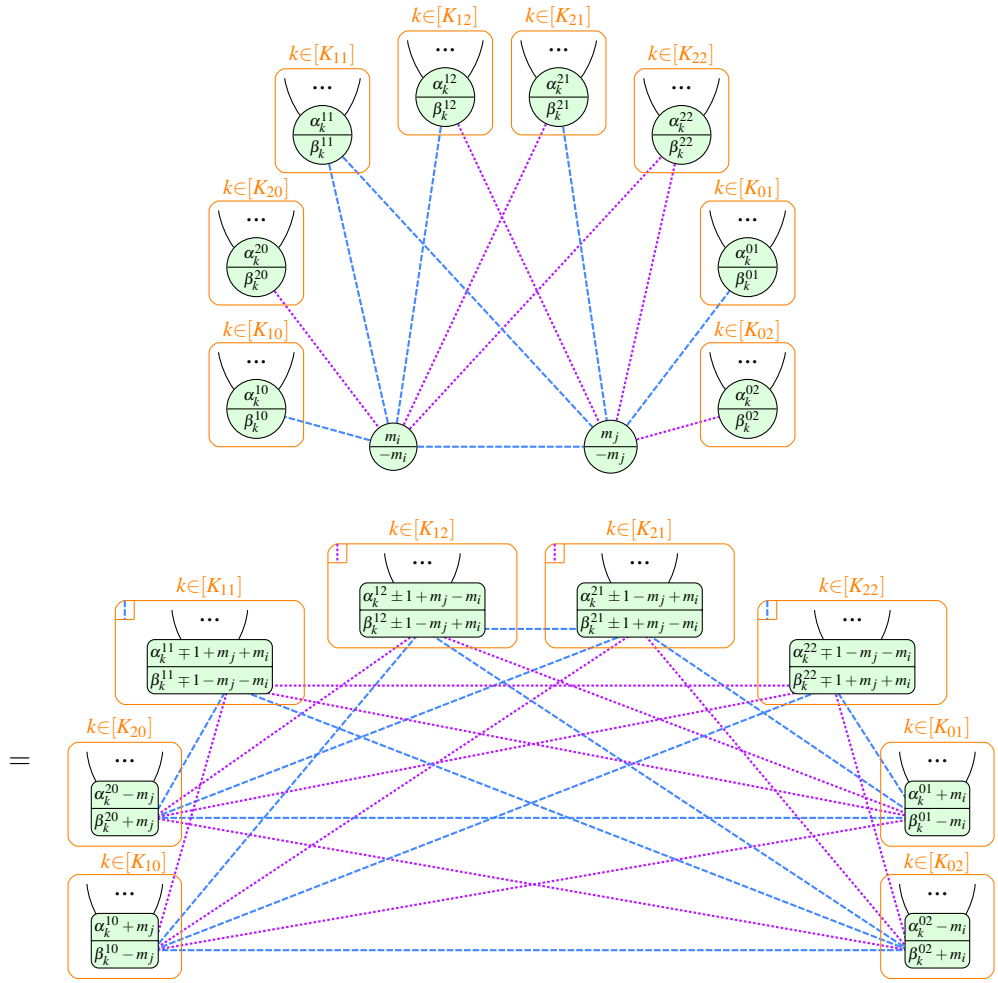


Theorem 3.7. Given any graph-like ZX-diagram containing an interior $\mathscr{N}$-spider x with phase $\frac{0}{n}$ or $\frac{n}{0}$ for $n \in \{1,2\}$, suppose we perform a $(-n)$-local complementation at x. Then, treating the two cases separately, the new ZX-diagrams are related to the old ones by the equalities:



Theorem 3.8. Given any graph-like ZX-diagram containing two interior $\mathscr{M}$-spiders i and j connected by edge ij of weight $w_{i,j} =: w \in \{1,2\}$, suppose we perform a proper $\pm w$-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 occurences of ± 1, and the roles of purple and blue will be swapped throughout.

Proofs of these theorems are found in Appendix **??**, **??** and **??** respectively. We can now combine them 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, and perhaps removing parallel H- or $H^\dagger$-edges via (14), we again have a graph-like diagram.

Theorem 3.9. 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 $\mathscr{N}$-spiders, $\mathscr{P}$-spiders, or adjacent pairs of $\mathscr{M}$-spiders. Repeat the steps below until no rule matches. After each step, apply (14) as needed until the resulting diagram is graph-like:

1. Eliminate a $\mathscr{P}$-spider via Theorem 3.6.
2. Eliminate an $\mathscr{N}$-spider via Theorem 3.7.
3. Eliminate two adjacent $\mathscr{M}$-spiders via Theorem 3.8.

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 $\mathscr{N}$-spiders, $\mathscr{P}$-spiders, or adjacent pairs of $\mathscr{M}$-spiders. □

In particular, if we start with a stabilizer diagram, we can eliminate all but perhaps one spider, depending on whether the initial number of $\mathscr{M}$-spiders was odd or even. This is because no step introduces any non-stabilizer phases. The algorithm above could be extended to a *non-closed* graph-like diagrams as in (Theorem 5.4, [9]) - for example, as part of a qutrit circuit optimisation algorithm. However, since this paper focuses on using the ZX-calculus to simplify *closed* tensor networks, we have not done so.

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. Subsequently, we briefly look at graph 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* by projecting it on to 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 $V_L(t) \neq V_{L'}(t) \Rightarrow L \not\simeq L'$.

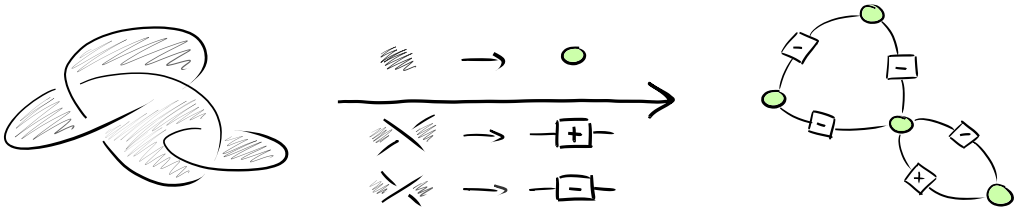
In general, computing $V_L(t)$ is exponentially costly in the number of crossings c, something made explicit when one uses the Kauffman bracket method [13]. Exactly evaluating the Jones polynomial at points $t \in \mathbb{C}$ is #P-hard, *except* at the *lattice roots of unity* $t \in \Lambda = \{\pm 1, \pm i, \pm e^{i2\pi/3}, \pm e^{i4\pi/3}\}$, where it can be evaluated efficiently (in time $O(poly(c))$) [12].

In the context of quantum computation, additively approximating the Jones polynomial at *non-lattice roots of unity* is the paradigmatic BQP-complete complete problem [2, 15]. From the point of view of topological quantum computation [18], which is more natural for knot theory, 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 creating anyons from the vacuum, then braiding them, and finally fusing them back to the vacuum. Specifically, in the case of $\mathrm{SU}(2)_k$ anyons, the Jones polynomial is evaluated at $t(k) = e^{i\frac{2\pi}{2+k}}$ [19].

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 [23]. The Potts model is defined on a signed graph G_L, obtained as follows. The link diagram is bicoloured checkerboard-style, then 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 is then equal to this partition function up to an efficiently computable scalar which depends on the link diagram. The relation between the point t at which the Jones is evaluated and the dimension d of the spins is $d = t + t^{-1} + 2$, which can be solved for $t(d)$ [17]. Note the correspondence between the dimension d in the Potts approach and the level k in the anyon-braiding approach to the Jones polynomial: $\{t(d) |\, d \in \{1,2,3,4\}\} = \{t(k) |\, k \in \{1,2,4\}\}$. This is consistent with the fact that braiding-only topological quantum computation with $\mathrm{SU}(2)_2$ anyons (Ising)

or $\mathrm{SU}(2)_4$ anyons is not universal (unless the $\mathrm{SU}(2)_4$ anyons are augmented by fusion and measurements [16]). A key observation is that if $d \in \{2,3,4\}$ then $t(d) \in \Lambda$ (the case $d = 1$ is trivial [12]).

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* [17]:

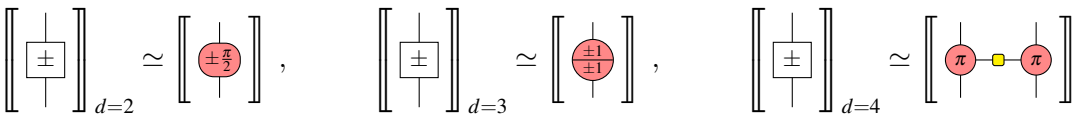


(19)

The $\pm$-boxes have the following concrete interpretation as matrices:

$$\left[\!\left[\; \boxed{\pm} \;\right]\!\right] = \sum_{i,j=0}^{d-1} \left(1-(1+t(d)^{\mp 1})\delta_{ij}\right)|i\rangle\langle j| = \begin{pmatrix} -t(d)^{\mp 1} & 1 & \dots & 1 \\ 1 & -t(d)^{\mp 1} & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & -t(d)^{\mp 1} \end{pmatrix} \tag{20}$$

We can now express the $\pm$-boxes in the ZX calculus. The $\pm$-matrices above for $d \in \{2,4\}$ are equal up to a scalar to concrete interpretations of *qubit stabilizer* ZX-diagrams, and the $\pm$-matrices for $d = 3$ of *qutrit stabilizer* ZX-diagrams (see Appendix **??**):



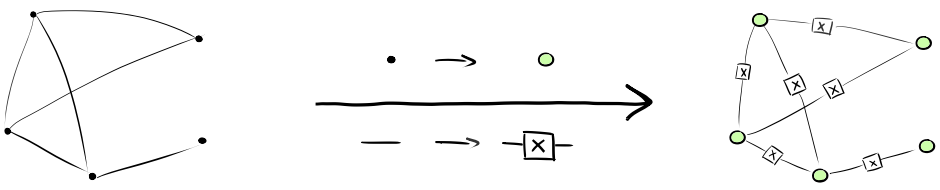
(21)

Since these generators decompose as stabilizer diagrams, $Z_{G_L}(d \in \{2,3,4\})$ can be evaluated *efficiently* via stabilizer ZX diagram simplification. Thus, we recover the known result that evaluating the Jones polynomial at $t \in \Lambda$ is in P.

4.2 Graph Colouring

Finally, let us briefly look at the *graph colouring problem*. A *d-colouring* of a graph G is an assignment of colours $\{1,...,d\}$ to the vertices of G so that no neighbouring vertices have the same colour. Given a graph G and an integer d, we wish to count the number of such d-colourings. 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). Through the lens of our graphical exposition we see that counting problems and computing partition functions are essentially the same problem, since both can be straightforwardly encoded as closed tensor networks [14, 5].

Given a graph G, the graph colouring problem can be encoded as a ZX diagram as follows. Every vertex of the graph is mapped to a d-dimensional phaseless (green) Z-spider. Every edge is mapped to a wire connecting the spiders which goes through an X-box.



(22)

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

$$\left[\!\left[\; \boxed{X} \;\right]\!\right] = \sum_{i,j=0}^{d-1} (1-\delta_{ij})\,|i\rangle\langle j| = \begin{pmatrix} 0 & 1 & \dots & 1 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 0 \end{pmatrix} \tag{23}$$

It is known that for arbitrary graphs 2-colouring is in P **[ref]**. For $d = 2$, the X-matrix is just the Pauli X, and so the problem reduces to simplifying a stabilizer diagram, which can be done efficiently. However, for $d = 3$ the X-matrix cannot be expressed as a stabilizer qutrit diagram, and so it is not expected that there exists an efficient simplification strategy for the corresponding qutrit ZX diagram. This is consistent with the fact that 3-colouring is #P-complete **[ref]**.

$$\left[\!\left[\; \boxed{X} \;\right]\!\right]_{d=2} \simeq \left[\!\left[\; \pm\pi \;\right]\!\right] \quad , \qquad \left[\!\left[\; \boxed{X} \;\right]\!\right]_{d=3} \not\simeq \left[\!\left[\; \begin{matrix} \pm a \\ \pm b \end{matrix} \;\right]\!\right] \quad , \quad \forall a,b \in \{0,1,2\}\,. \tag{24}$$

5 Outlook

In this work, we have advocated for the use of the ZX calculus as a convenient graphical framework for treating a broad range of many-body problems on equal footing, and for diagnosing the complexity of interesting families of problems. When the solution to a problem is encoded as a closed ZX diagram, this solution can be obtained via full diagram simplification by applying the rules of the calculus. The problem can be solved efficiently when the diagram encoding its answer is in the stabilizer fragment of the calculus. In this work specifically, we have fleshed out the simplifications strategies that make this apparent for the case of qutrits.

We also looked at two case studies. We reviewed complexity results on evaluating the Jones polynomial and counting graph colourings, two problems which can be cast in tensor network form. We then recovered known complexity results using only the language of the ZX-calculus.

A main path for future work entails the generalisation of our stabilizer simplification rules for all prime dimensions. This in turn also motivates work in circuit extraction [4] for qudit circuits; one could hope to find graph-theoretic simplification [9] strategies analogous to those for the case of qubits. Another direction regards defining scalar-exact versions of the rewrite rules, which would be necessary for concrete applications to problems where d-state systems are the native degrees of freedom.

6 Acknowledgments

ATT 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 postdoctoral 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] Lucas Dixon & Ross Duncan (2009): *Graphical reasoning in compact closed categories for quantum computation*. Annals of Mathematics and Artificial Intelligence 56(1), pp. 23–42.

[9] Ross Duncan, Aleks Kissinger, Simon Perdrix & John van de Wetering (2020): *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*.

[10] Xiaoyan Gong & Quanlong Wang (2017): *Equivalence of Local Complementation and Euler Decomposition in the Qutrit ZX-calculus*.

[11] Johnnie Gray & Stefanos Kourtis (2020): *Hyper-optimized tensor network contraction*.

[12] 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.

[13] Louis H Kauffman (2001): *Knots and Physics*. WORLD SCIENTIFIC, doi:10.1142/4256. Available at https://doi.org/10.1142/4256.

[14] 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.

[15] Greg Kuperberg (2014): *How hard is it to approximate the Jones polynomial?*

[16] Claire Levaillant, Bela Bauer, Michael Freedman, Zhenghan Wang & Parsa Bonderson (2015): *Universal gates via fusion and measurement operations onSU(2)4anyons*. Physical Review A 92(1), doi:10.1103/physreva.92.012301. Available at http://dx.doi.org/10.1103/PhysRevA.92.012301.

[17] 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.

[18] Jiannis K. Pachos (2012): *Introduction to Topological Quantum Computation*. Cambridge University Press, doi:10.1017/CBO9780511792908.

[19] 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.

[20] Quanlong Wang (2018): *Qutrit ZX-calculus is Complete for Stabilizer Quantum Mechanics*.

[21] Quanlong Wang (2020): *Completeness of algebraic ZX-calculus over arbitrary commutative rings and semirings*.

[22] John van de Wetering (2020): *ZX-calculus for the working quantum computer scientist*.

[23] 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`.