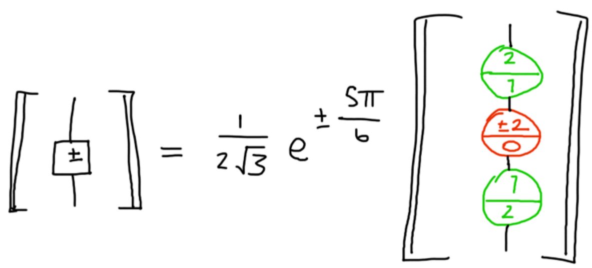
Efficiently Calculating Jones Polynomials at Lattice Roots of Unity with the ZX Calculus (Draft)

Alex Townsend-Teague Konstantinos Meichandzitis

1 Introduction

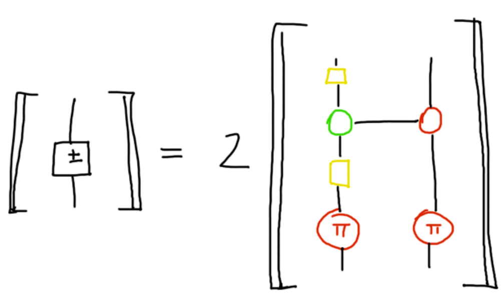
[Use Konstantinos' O.G. notes from the start of this project, and introduce the qubit and qutrit ZX calculi]

For the case $q = 3$ we use the qutrit ZX calculus:



(1)

For the case $q = 4$ we are back again in the usual qubit ZX calculus:



(2)

In each case, the resulting map is in the stabilizer fragment of the ZX calculus.

2 Simplifying Qubit ZX Diagrams

In [cite Aleks' paper] the authors give an efficient algorithm for reducing any qubit stabilizer ZX diagram to one of vastly smaller size [specific bound?]. In our case, since any knot will be transformed by the process above into a ZX diagram with zero inputs and zero outputs, this gives an efficient algorithm for reducing the diagram to a scalar.

[Summarise Aleks' algorithm]

This suffices to prove our main result for the cases $q \in \{2, 4\}$. However, it doesn't yet give us a useful algorithm for explicitly calculating Jones polynomials, because throughout the method above equality is considered only up to a complex scalar multiple. Therefore we now give a scalar-exact version.

[Give scalar-exact version]

3 Simplifying Qutrit ZX Diagrams

For the remaining case $q = 3$, we will show that we can generalise all the ideas of the previous section to the qutrit ZX calculus.

3.1 Graph-Like Qutrit ZX Diagrams

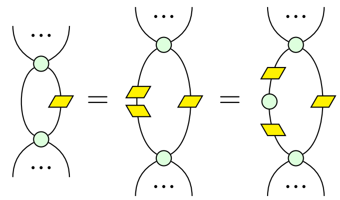
[Make all of the following scalar-exact]

We first define a graph-like diagram in the qutrit ZX calculus.

Definition 3.1. A qutrit ZX diagram is *graph-like* when:

1. Every spider is a Z-spider.
2. Spiders are only connected by Hadamard edges (H-edges) or their adjoints ($H^\dagger$-edges).
3. Every pair of spiders is connected by at most one H-edge or $H^\dagger$-edge.
4. Every input and output is connected to a spider.
5. Every spider is connected to at most one input or 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. The following equation, which holds with the roles of H and $H^\dagger$ reversed, helps us get around this:



(3)

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

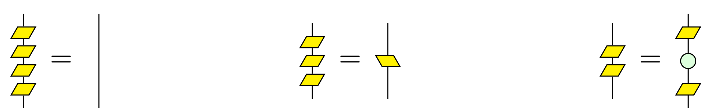
Lemma 3.2. The following two equations hold in the qutrit ZX calculus. Moreover, they hold with the roles of H and $H^\dagger$ interchanged:

(4)

Proof. This is Lemma 3.4 in [cite Harny's local complementation paper]. □

As we will formalise later, the lemma above says we can think of Hadamard edges as single edges and their adjoints as double edges, then work modulo 3, since every triple of parallel edges disappears.

Lemma 3.3. The following three equations hold in the qutrit ZX calculus. Moreover, they hold with the roles of H and $H^\dagger$ interchanged:



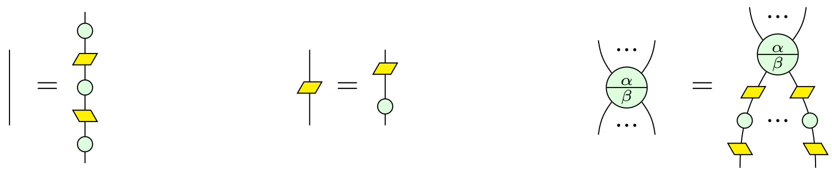
(5)

Proof. □

Again intuitively we can think of Hadamard boxes of having value 1 and their adjoints -1 and then work modulo 4.

Corollary 3.4. 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 3.3 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 Equation 3 where fusion is not possible, so that no plain edge connects two spiders. Apply Lemma 3.2 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 again add a few spiders, H- and $H^\dagger$-boxes as needed:



(6)

□

Definition 3.5. 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.

A graph state is described fully by its underlying multigraph, or equivalently by an adjacency matrix, where edges take weights in $\mathbb{Z}_3$ [reference Harny]. 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 [Harny's completeness paper, Definition 2.6], though the effect is now slightly more complicated. We'll give the intuition after the formal definition:

Definition 3.6. Given $a \in \mathbb{Z}_3$ and a graph state $|G\rangle$ with adjacency matrix $W = (w_{i,j})$, the *(a)-local complentation* at node k is the new graph state $|G *_a i\rangle$, whose adjacency matrix $W' = (w'_{i,j})$ given by:

$$w'_{i,j} = w_{i,j} + a w_{i,k} w_{j,k} \tag{7}$$

So only those edges between neighbours of node k 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 k. As always, this is best seen graphically. We reintroduce the blue dashed line notation for Hadamard edges, and now also use purple dashed lines for $H^\dagger$-edges:

(8)

So now for two nodes i and j both connected to k by the same colour edge, a-local complementation at k increases weight $w_{i,j}$ by a. If instead i and j are connected to k by edges of different colour, a-local complementation at k decreases $w_{i,j}$ by a:

(9)

Just as in the qubit case, composing local complementations gives a local pivot operation:

Definition 3.7. Given $a, b, c \in \mathbb{Z}_3$ and a graph state $|G\rangle$, the (a, b, c)-*local pivot* at nodes i and j is the new graph state $\left|G \wedge_{(a,b,c)} ij\right\rangle := |((G *_a i) *_b j) *_c i\rangle$