Verifying Quantum Computations with the ZX-calculus

Ross Duncan
0. Motivations
“Verifying”

- **Not** verifying that a given computation is *quantum*

- Verifying that your program is *correct*:
  - is the program I *wrote* equivalent to its specification?
  - is the program I *ran* equivalent to the one I *wrote*?
The Stack

Algorithms

High-level languages

Classical compiler

Classical architecture (control operations)

Hardware building blocks (gates, bits)

VLSI circuits

Semiconductor transistors

Quantum compiler

Quantum architecture (QC gates, qubits, communication)

Error-correction and control pulses

Underlying technology (semiconductors, trapped ions)

Verifying Quantum Computations

1. No one has any real idea what the architecture of a working QC is going to be — not quantum circuits!

2. Large error-correction overhead will be required — need to deal with this extra layer of complexity

So: the programs we write for QC will not look anything like their implementation — powerful and flexible verification formalisms will be needed!
Motivations
Motivations

\[ \lambda x.\lambda y.\lambda z.xz(yz) \]
Motivations

\[ \lambda x.\lambda y.\lambda z.xz(yz) \]

Hilbert space, unitary transforms, self-adjoint operators....
Quantum Circuits

**Good Points:**
- Universal
- Clear operational interpretation

**Bad points:**
- Inflexible
- Mathematically ad hoc
- Few nice algebraic properties

**Meh Point:**
- Not complete
Motivations

Hilbert space, unitary transforms, self-adjoint operators....

\( \lambda x. \lambda y. \lambda z. xz(yz) \)

fun fact 0 = 1
1 fact x = x * fact (x-1)
ZX-calculus

**Good Points:**
- Universal
- Derived from the basic algebra of complementarity
- Powerful algebraic theory
- Can represent almost anything

**Bad point:**
- Need to impose operational meaning post-hoc

**Meh Point:**
- Not complete
ZX-calculus

**Good Points:**

+ Universal
+ Derived from the basic algebra of complementarity
+ Powerful algebraic theory
+ Can represent almost anything
+ **Now complete for the Clifford + T fragment!**

**Bad point:**

- Need to impose operational meaning post-hoc

**Meh Point:**

--- Not complete

--- Jeandel, Perdrix, Vilmart

*A Complete Axiomatisation of the ZX-Calculus for Clifford+T Quantum Mechanics*
ZX-calculus

**Good Points:**
+ Universal
+ Derived from the basic algebra of complementarity
+ Powerful algebraic theory
+ Can represent almost anything
+ Now complete for the Clifford + T fragment!

**Now universally complete!**

**Bad point:**
- Need to impose operational meaning post-hoc

---

**Ng and Wang**
A universal completion of the ZX-calculus for Clifford+T Quantum Mechanics
1706.09877
**Good Points:**

+ Universal
+ Derived from the basic algebra of complementarity
+ Powerful algebraic theory
+ Can represent almost anything
+ Now complete for the Clifford + T fragment!

+ **Now universally complete!**

**Bad point:**

- Need to impose operational meaning post-hoc
“Observables”

Non-degenerate, projective observables on finite dimensional spaces

i.e.

Orthonormal bases

\[ A = |a_1\rangle, |a_2\rangle, \ldots, |a_d\rangle \]
Unbiasedness and Phases

A state $|\psi\rangle$ is **unbiased** for a basis $A$ if

$$|\langle a_i | \psi \rangle| = \frac{1}{\sqrt{d}}$$

Every unbiased state determines a unitary map via

$$U_{\psi} : |a_i\rangle \mapsto \sqrt{d}\langle a_i | \psi \rangle |a_i\rangle$$

This is called a **phase map** for $A$. 
Unbiasedness and Phases

A state $|\psi\rangle$ is **unbiased** for a basis $A$ if

Proposition 1:
The phases of any observable form an abelian group

$$U_\psi : |a_i\rangle \mapsto \sqrt{d} \langle a_i | \psi \rangle |a_i\rangle$$

This is called a **phase map** for $A$. 
Phases & Unbiased Points

\[ Z_\alpha = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} \]
Complementarity

Two observables $A$ and $B$ are complementary if

$$|\langle a_i | b_j \rangle| = \frac{1}{\sqrt{d}} \quad \forall i, j$$

(aka mutually unbiased)

They are strongly complementary if $\{U_{b_i}\}_i$ is a subgroup of the phase group $\Phi_A$. 
Observables are Frobenius Algebras

**Theorem 2:** Observables are in bijection with $\dagger$-special commutative Frobenius algebras.

$$
\begin{align*}
\delta : A &\rightarrow A \otimes A \\
\epsilon : A &\rightarrow I \\
\mu : A \otimes A &\rightarrow A \\
\eta : I &\rightarrow A \\
\end{align*}
$$

Via:

$$
\begin{align*}
\delta :: |a_i\rangle &\rightarrow |a_i\rangle \otimes |a_i\rangle \\
\epsilon :: |a_i\rangle &\rightarrow 1 \\
\mu = \delta^\dagger \\
\eta = \epsilon^\dagger \\
\end{align*}
$$

Strongly Complementary Observables are Hopf algebras

**Theorem 3:** Two observables are strongly complementary iff they form a Hopf algebra

\[
\delta \circ \epsilon \circ \mu \circ \eta \\
\mu \circ \eta \circ \delta \circ \epsilon
\]
Strongly Complementary Observables are Hopf algebras

**Theorem 3:** Two observables are strongly complementary iff they form a Hopf algebra

Strongly Complementary Observables are Hopf algebras

**Theorem 3:** Two observables are strongly complementary iff they form a Hopf algebra

Strong Complementarity

\[ \iff \]

GHZ/Mermin Non-locality

**Theorem 4**: Two observables are strongly complementary iff they admit a Mermin-style probability-free proof of non-locality.


More?

For more on the theoretical perspective, come see my talk on thursday at LIG.

See also:
DOI: 10.1145/2933575.2934550
2. The ZX-Calculus
Z and X Observables

The Pauli $Z$ and $X$ observables are strongly complementary, with some additional features:

- The phase group is $[0, 2\pi)$
- Dimension 2 implies two classical points
- The action of the classical group is $\alpha \mapsto -\alpha$
Z and X Observables

- Both observables generate the same compact structure
  \[ \begin{array}{c}
  \includegraphics[width=0.1\textwidth]{diagram1.png}
  \end{array} \]
  \[=\]
  \[ \begin{array}{c}
  \includegraphics[width=0.1\textwidth]{diagram2.png}
  \end{array} \]
  \[=\]
  \[ \begin{array}{c}
  \includegraphics[width=0.2\textwidth]{diagram3.png}
  \end{array} \]
  - can just treat the diagram as an undirected graph

- the Z and X are related by a definable unitary
  - gives rise to colour change rule
**ZX-calculus syntax**

\[ \alpha \in [0, 2\pi) \]

**Defn:** A *diagram* is an undirected open graph generated by the above vertices.
The encoder maps a single input qubit to the 7-qubit code-space via the circuit shown in Figure 3 (a). Note that qubit 3 is the input. Using the translation described in Example 2.5 we obtain the \( z^x \)-calculus term shown in Figure 3 (b).

The encoding circuit is simple enough that Quantomatic can reduce it to its minimal form without human intervention. The resulting simplification is shown below; the final equation is merely a rearrangement of the vertices to show the structure more clearly.

The decoding circuit is simply the encoder in reverse.

The error corrector

An encoder is not very useful if we cannot detect and correct errors on the encoded data. The circuit for carrying out this function is shown in Figure 4. Notice that the circuit naturally splits into two parts. The error-detecting part introduces a 6-qubit ancilla, entangles it with the input, and then measures the ancillae. The resulting measurements are called the error syndrome. The error-correcting part comprises the Pauli operations at the end, which are applied conditionally, depending on the value of the error syndrome. One crucial detail has been omitted from this picture: we do not show the conditions which control the error-correcting part. We will derive these conditions in the next section.

We translate the circuit into the \( z^x \)-calculus as shown in Figure 5. The variables \( a, b, \ldots, f \) indicate the outcomes of the six syndrome measurements, while the variables \( x_i \) and \( z_i \) indicate whether a Pauli \( X \) or \( Z \) correction need be applied on qubit \( i \) of the codeword.
ZX-calculus semantics

\[ |0\rangle \otimes n \rightarrow |0\rangle \otimes m \]
\[ |1\rangle \otimes n \rightarrow e^{i\alpha} |1\rangle \otimes m \]
\[ |+\rangle \otimes n \rightarrow |+\rangle \otimes m \]
\[ |-\rangle \otimes n \rightarrow e^{i\alpha} |-\rangle \otimes m \]
Representing Qubits

\[
\begin{bmatrix}
\text{Red Circle} \\
\text{Black Dot}
\end{bmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle
\]

\[
\begin{bmatrix}
\text{Red Circle} \\
\pi
\end{bmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle
\]

\[
\begin{bmatrix}
\text{Green Circle} \\
\text{Black Dot}
\end{bmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = |+\rangle
\]

\[
\begin{bmatrix}
\text{Green Circle} \\
\pi
\end{bmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = |-\rangle
\]
Representing Paulis

\[
\begin{bmatrix}
\pi
\end{bmatrix} = \begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix}
\]

\[
\begin{bmatrix}
\pi
\end{bmatrix} = \begin{pmatrix}
1 & 0 \\
0 & -1
\end{pmatrix}
\]
Representing Phase shifts

\[
\begin{bmatrix}
\alpha
\end{bmatrix}
= \begin{pmatrix}
1 & 0 \\
0 & e^{i\alpha}
\end{pmatrix}
\]

\[
\begin{bmatrix}
\beta
\end{bmatrix}
= \begin{pmatrix}
\cos \frac{\beta}{2} & -i \sin \frac{\beta}{2} \\
-i \sin \frac{\beta}{2} & \cos \frac{\beta}{2}
\end{pmatrix}
\]
Representing CNot

\[ \wedge X = \begin{bmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix} = \begin{bmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \]
The ZX-calculus is universal

**Theorem**: Let $U$ be a unitary map on $n$ qubits; then there exists a ZX-calculus term $D$ such that:

$$[D] = U$$
The ZX-calculus is universal

**Theorem:** Let $U$ be a unitary map on $n$ qubits; then there exists a ZX-calculus term $D$ such that:

$$[D] = U$$
Equations

(spider)

(anti-loop)

(identity)
Equations

Generalised Spider

\[
\begin{align*}
\alpha & = \alpha + \beta \\
\beta & = 0
\end{align*}
\]

(spiders) (anti-loop) (identity)
Equations

(bialgebra)

(copying)

(hopf)

(π-commute)
Equations

“Strong Complementarity”

(bialgebra)

(copying)

(hopf)

(π-commute)
Equations

\[
\alpha + \frac{n\pi}{2} = \alpha
\]

(colour change)
Equations

A weird one specific to ZX

\[
\alpha + \frac{n\pi}{2} \quad = \quad \alpha
\]

(colour change)
Representing Hadamard

\[ H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{bmatrix} \frac{\pi}{2} \\ \frac{\pi}{2} \end{bmatrix} \]
More on the Hadamard

\[ H := \frac{\pi}{2} \]
More on the Hadamard
More on the Hadamard
More on the Hadamard
More on the Hadamard

Corollary: total symmetry between red and green
Example: Controlled-Z

\[
\begin{align*}
\text{Example 3.12 (The } \cdot \text{Z-gate)} & \\
\text{The } \cdot \text{Z-gate can be obtained by using a } H \text{ gate to transform the second qubit of a } \cdot \text{X gate. We obtain a simpler representation using the colour-change rule:}
\end{align*}
\]

\[
\begin{align*}
H H & = H \\
H & = H
\end{align*}
\]

From the presentation of \( \cdot \text{Z} \) in the \( \text{zx} \)-calculus, we can immediately read off that it is symmetric in its inputs. Furthermore, we can prove one of the basic properties of the \( \cdot \text{Z} \) gate, namely that it is self-inverse.

\[
\begin{align*}
H H & = H \\
H & = H
\end{align*}
\]

Example 3.13 (Bell state).

The following is a \( \text{zx} \)-calculus term representing a quantum circuit which produces a Bell state, \(|00\rangle + |11\rangle\). We can verify this fact by the equations of the calculus.

The corresponding \( \text{zx} \)-calculus derivation is a proof of the correctness of this circuit.

The \( \text{zx} \)-calculus can represent many things which do not correspond to quantum circuits. We now present a criterion to recognize which diagrams do correspond to quantum circuits.

3.2 Circuit-like diagrams

While the preceding has demonstrated the ease with which quantum circuits can be translated into diagrams, there are many diagrams which do not correspond to quantum circuits.
Example: Controlled-Z

Proof. It suffices to show that there are $zx$-calculus terms for the matrices $Z$, $H$, and $X$. We have $JH = H$, $J - K = Z - X$ and $JK = X$ which can be verified by direct calculation. Note that $JK = JK$ so the presentation of $X$ is unambiguous.

Example 3.12 (The $X$-gate). The $X$-gate can be obtained by using a Hadamard ($H$) gate to transform the second qubit of a $X$ gate. We obtain a simpler representation using the colour-change rule $H = H$.

From the presentation of $X$ in the $zx$-calculus, we can immediately read off that it is symmetric in its inputs. Furthermore, we can prove one of the basic properties of the $X$ gate, namely that it is self-inverse. $H = H = H = H = H = H$.

Example 3.13 (Bell state). The following is a $zx$-calculus term representing a quantum circuit which produces a Bell state, $|00⟩ + |11⟩$. We can verify this fact by the equations of the calculus.

The corresponding $zx$-calculus derivation is a proof of the correctness of this circuit.

The $zx$-calculus can represent many things which do not correspond to quantum circuits. We now present a criterion to recognize which diagrams do correspond to quantum circuits.
Example: CNOTS
Example: CNOTS
Example: 2-Qubit Quantum Fourier Transform

\[ j_1 = |1\rangle \]
\[ j_0 = |0\rangle \]

input qubits

controlled gate \[ Z_{\pi/2} \]

\[ -\pi/4, \pi/4 \]
Example: 2-Qubit Quantum Fourier Transform

\[ H/4 \]

\[ H/4 \]
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform

\[ \pi \]

\[ \pi \quad \pi \quad -\pi/4 \quad \pi/4 \]
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform
Example: 2-Qubit Quantum Fourier Transform

\[ j_0 = |0\rangle + e^{i\frac{\pi}{2}} |1\rangle \]
\[ j_1 = |0\rangle + e^{i\pi} |1\rangle \]
The ZX-calculus is complete

Question:
If $[D_1] = [D_2]$ then is it provable that $D_1 \overset{ZX}{=} D_2$?
The ZX-calculus is complete

Question:
If \[ [D_1] = [D_2] \] then is it provable that \( D_1 \xrightarrow{ZX} D_2 \)?

**Theorem** (Backens, 2012):
If \( D_1 \) and \( D_2 \) are from the **stabilizer** fragment then YES.
The ZX-calculus is complete

Question:
If \( [D_1] = [D_2] \) then is it provable that \( D_1 \overset{ZX}{=} D_2 \) ?

Theorem (Backens, 2012):
If \( D_1 \) and \( D_2 \) are from the stabilizer fragment then YES.

Stabilizer fragment:
Clifford group +
Stabilizer states
\[
\alpha \in \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\}
\]
The ZX-calculus is complete

Question:

If $\llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket$ then is it provable that $D_1 \overset{ZX}{=} D_2$?
The ZX-calculus is complete

**Question:**

If \( \llbracket D_1 \rrbracket = \llbracket D_2 \rrbracket \) then is it provable that \( D_1 \models ZX D_2 \)?

**Theorem** (Jeandel, Perdrix & Vilmart vs Ng & Wang, 2017):

If we add some new axioms (and maybe generators) then YES.
The ZX-calculus is complete

Question:
If $[[D_1]] = [[D_2]]$ then is it provable that $D_1 \overset{ZX}= D_2$?

Theorem (Jeandel, Perdrix & Vilmart vs Ng & Wang, 2017):
If we add some new axioms (and maybe generators) then YES.

But the new axioms kinda suck:
3. Application: the colour code
Liam’s project:

* Formalise the “smallest interesting colour code” in the ZX-calculus

* Formally verify its basic properties in quantomatic
Liam’s project:

* Formalise the “smallest interesting colour code” in the ZX-calculus

* Formally verify its basic properties in quantomatic
Liam’s project:

* Formalise the “smallest interesting colour code” in the ZX-calculus

* Formally verify its basic properties in quantomatic

Liam doesn’t know anything about quantum theory!
THE SMALLEST INTERESTING COLOUR CODE

September 26, 2016

What is the smallest interesting colour code? The first possible answer is the 7 qubit Steane code. Except, it isn’t especially interesting. In particular, the transversal gates of the Steane code are the Clifford group. Really interesting codes have transversal gates outside the Clifford group, and may be used in concert with gauge-fixing or magic state distillation.

Those familiar with the field will immediately retort that the 15 qubit Reed-Muller code is a colour code as was observed in 2006 by Bombin and Martin-Delgado. And it is interesting. The non-Clifford gate

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

is transversal for the 15 qubit code, in the sense that

$$T_L^\dagger = T^{\otimes 15}.$$
THE SMALLEST INTERESTING COLOUR CODE

What is the smallest interesting colour code? The first possible answer is the 7 qubit Steane code. Except, it isn’t especially interesting. In particular, the transversal gates of the Steane code are the Clifford group. Really interesting codes have transversal gates outside the Clifford group, and may be used in concert with gauge-fixing or magic state distillation.

Those familiar with the field will immediately retort that the 15 qubit Reed-Muller code is a colour code as was observed in 2006 by Bombin and Martin-Delgado. And it is interesting. The non-Clifford gate

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

is transversal for the 15 qubit code, in the sense that

$$T_L^{\dagger} = T^{\otimes 15}.$$
The smallest interesting colour code

- $[[8,3,2]]$ code:
  - 8 physical qubits
  - 3 logical qubits
  - distance 2
The smallest interesting colour code

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>000\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>001\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>010\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>011\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>100\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>101\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>110\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>111\rangle$</td>
</tr>
</tbody>
</table>
The smallest interesting colour code

The “smallest interesting colour code” takes its name from a blog post by Earl Campbell [7] which provided the inspiration for this project, and also appears in the papers [9, 8, 26]. It is a [8, 3, 2] code, meaning it encodes 3 logical qubits into 8 physical qubits, and has a distance of 2, meaning it can detect any single qubit error but not correct it. The code can be presented geometrically as a cube where each vertex corresponds to a qubit.

The logical Pauli operators are transversal, with the three logical $X$ operators obtained by applying $X$ to the three faces of the cube, while the logical $Z$ is obtained as an edge operator.

From the description of the $X$ operators, it's straightforward to find the translation of the computational basis states to codewords, which is shown in Table 1.
The smallest interesting colour code

<table>
<thead>
<tr>
<th>Logical State</th>
<th>Codeword State</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>00000000+11111111</td>
</tr>
<tr>
<td>001</td>
<td>10101010+01010101</td>
</tr>
<tr>
<td>010</td>
<td>11001100+00110011</td>
</tr>
<tr>
<td>011</td>
<td>01100110+10011001</td>
</tr>
<tr>
<td>100</td>
<td>11110000+00001111</td>
</tr>
<tr>
<td>101</td>
<td>01011010+10100101</td>
</tr>
<tr>
<td>110</td>
<td>00111100+11000011</td>
</tr>
<tr>
<td>111</td>
<td>10010110+01101001</td>
</tr>
</tbody>
</table>

The logical Pauli operators are transversal, with the three logical $X$ operators obtained by applying $X$ to the three faces of the cube, while the logical $Z$ is obtained as an edge operator.

From the description of the $X$ operators, it's straightforward to find the translation of the computational basis states to codewords, which is shown in Table 1.
The smallest interesting colour code

$$CCZ_{L_1,L_2,L_3} = T \otimes T^\dagger \otimes T^\dagger \otimes T \otimes T^\dagger \otimes T \otimes T \otimes T^\dagger$$
The smallest interesting colour code

$$CCZ_{L_1, L_2, L_3} = T \otimes T^\dagger \otimes T^\dagger \otimes T \otimes T^\dagger \otimes T \otimes T \otimes T^\dagger$$
The smallest interesting colour code

$$CCZ_{L_1,L_2,L_3} = T \otimes T^\dagger \otimes T^\dagger \otimes T \otimes T^\dagger \otimes T \otimes T \otimes T^\dagger$$

\[\frac{\pi}{2} \quad -\frac{\pi}{4} \quad -\frac{3\pi}{4} \quad \frac{\pi}{4} \quad \frac{3\pi}{4} \quad -\frac{\pi}{4} \quad -\frac{\pi}{4}\]
### Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>000\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>001\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>010\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>011\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>100\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>101\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>110\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>111\rangle)</td>
</tr>
</tbody>
</table>
The "smallest interesting colour code" takes its name from a blog post by Earl Campbell, meaning it can detect any single qubit error but not correct it. The code can be presented geometrically as a cube where each vertex corresponds to a qubit.

The code is a \([8, 3, 2]\) code, meaning it encodes 3 logical qubits into 8 physical qubits, and has a distance of 2. The code are publicly available as a downloadable Quantomatic project at [Quantomatic project files](https://example.com).

In this picture, the logical Pauli operators are transversal, with the three logical $X$ operators obtained by applying to the three faces of the cube, while the logical $Z$ operator is obtained as an edge operator.

The final encoder is obtained by composing these four diagrams along the 8 physical qubits; see [7] for nice pictures.

### Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>000\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>001\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>010\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>011\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>100\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>101\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>110\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>111\rangle)</td>
</tr>
</tbody>
</table>
Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>000\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>001\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>010\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>011\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>100\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>101\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>110\rangle)</td>
</tr>
<tr>
<td>(</td>
<td>111\rangle)</td>
</tr>
</tbody>
</table>
The "smallest interesting colour code" takes its name from a blog post by Earl Campbell \[8, 3, 2\] code, meaning it encodes 3 logical qubits into 8 physical qubits, and has a distance of 2.

### The Code

The logical Pauli operators are transversal, with the three logical $X$ operators obtained by applying to the three faces of the cube, while the logical $Z$ operator corresponds to the cell; see [7] for nice pictures.

### Table 1: The computational basis as codewords

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>000\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>001\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>010\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>011\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>100\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>101\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>110\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>111\rangle$</td>
</tr>
</tbody>
</table>
which provided the inspiration for this project, and also appears in the papers.

The "smallest interesting colour code" takes its name from a blog post by Earl Campbell at:

https://gitlab.cis.strath.ac.uk/kwb13215/Colour-Code-QPL

are publicly available as a downloadable Quantomatic project at

The Quantomatic project files

The Smallest Interesting Colour Code in Quantomatic

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>000\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>001\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>010\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>011\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>100\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>101\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>110\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>111\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
</tbody>
</table>

From the description of the \texttt{cube}, and the single \texttt{X} operation an 8-qubit GHZ state. In the \texttt{X} operators, it's straightforward to find the \texttt{Z} operators obtained by

\texttt{zx}-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli operators an 8-qubit GHZ state. In the \texttt{X} operations an 8-qubit GHZ state. In the \texttt{X} operators, it's straightforward to find the translation of the \texttt{Z} operators obtained by

\texttt{zx}-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>000\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>001\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>010\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>011\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>100\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>101\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>110\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>111\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
</tbody>
</table>

In the

\texttt{X} operations an 8-qubit GHZ state. In the

\texttt{Z} operators obtained by

\texttt{zx}-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli

Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>000\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>001\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>010\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>011\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>100\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>101\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>110\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
<tr>
<td></td>
<td>111\rangle \quad \frac{1}{\sqrt{2}} (</td>
</tr>
</tbody>
</table>

In the

\texttt{X} operations an 8-qubit GHZ state. In the

\texttt{Z} operators obtained by

\texttt{zx}-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli

Encoding
The "smallest interesting colour code" takes its name from a blog post by Earl Campbell, meaning it can detect any single qubit error but not correct it. The code can be presented as follows:

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>000⟩</td>
</tr>
<tr>
<td></td>
<td>001⟩</td>
</tr>
<tr>
<td></td>
<td>010⟩</td>
</tr>
<tr>
<td></td>
<td>011⟩</td>
</tr>
<tr>
<td></td>
<td>100⟩</td>
</tr>
<tr>
<td></td>
<td>101⟩</td>
</tr>
<tr>
<td></td>
<td>110⟩</td>
</tr>
<tr>
<td></td>
<td>111⟩</td>
</tr>
</tbody>
</table>
The Smallest Interesting Colour Code in Quantomatic

The "smallest interesting colour code" takes its name from a blog post by Earl Campbell which provided the inspiration for this project, and also appears in the papers 

The code can be presented geometrically as a cube where each vertex corresponds to a qubit. In this picture, the cube, and the single stabilizers of the code correspond to the face and cell operators of the cube, and the single logical state is obtained as an edge operator.

The logical Pauli operators are transversal, with the three logical X | 01100110 00000000 01010101 00110011 11000011 10100101 00001111 10100101 11111111 01011011 10011001 00001111 01101001 11000011 01101001 01101001

The Code

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>000⟩</td>
</tr>
<tr>
<td></td>
<td>001⟩</td>
</tr>
<tr>
<td></td>
<td>010⟩</td>
</tr>
<tr>
<td></td>
<td>011⟩</td>
</tr>
<tr>
<td></td>
<td>100⟩</td>
</tr>
<tr>
<td></td>
<td>101⟩</td>
</tr>
<tr>
<td></td>
<td>110⟩</td>
</tr>
<tr>
<td></td>
<td>111⟩</td>
</tr>
</tbody>
</table>
The “smallest interesting colour code” takes its name from a blog post by Earl Campbell geometrically as a cube where each vertex corresponds to a qubit.

The Quantomatic project files are publicly available as a downloadable Quantomatic project at the logical state which provided the inspiration for this project, and also appears in the papers that encoded 3 logical qubits into 8 physical qubits, and has a distance of 3 so our first task is to define the encoding operation in the computational basis states to codewords, which is shown in Table 1.

### Table 1: The computational basis as codewords

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>000\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>001\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>010\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>011\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>100\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>101\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>110\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>111\rangle )</td>
</tr>
</tbody>
</table>
The "smallest interesting colour code" takes its name from a blog post by Earl Campbell geometrically as a cube where each vertex corresponds to a qubit.

## The Code

The Quantomatic project files are publicly available as a downloadable Quantomatic project at [link](http://www.geometrically). From the description of the cube, and the single computational basis states to codewords, which is shown in Table 1.

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>000\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>001\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>010\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>011\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>100\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>101\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>110\rangle )</td>
</tr>
<tr>
<td>(</td>
<td>111\rangle )</td>
</tr>
</tbody>
</table>

The logical Pauli operators are transversal, with the three logical operators obtained by applying the logical state to the three faces of the cube, while the logical operators are obtained as an edge operator. The code can be presented at Table 1, we can see that the basis states are prepared by conditionally applying Pauli operations an 8-qubit GHZ state. In the bit flips are applied by "copying" a Pauli operators an 8-qubit GHZ state. In the

The Smallest Interesting Colour Code in Quantomatic

The logical state which provided the inspiration for this project, and also appears in the papers 2, meaning it can detect any single qubit error but not correct it. The code can be presented four physical qubits, as shown below.

All the proofs which appear in this paper and its appendix are publicly available as a downloadable Quantomatic project at [link](http://www.geometrically). It is an

[Figure 3: An example simproc:](https://example.com)
The smallest interesting colour code takes its name from a blog post by Earl Campbell, meaning it can detect any single qubit error but not correct it. The code can be presented geometrically as a cube where each vertex corresponds to a qubit.

## Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>000\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>001\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>010\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>011\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>100\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>101\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>110\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>111\rangle$</td>
</tr>
</tbody>
</table>
### Encoding

<table>
<thead>
<tr>
<th>logical state</th>
<th>codeword state</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>000\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>001\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>010\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>011\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>100\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>101\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>110\rangle$</td>
</tr>
<tr>
<td>$</td>
<td>111\rangle$</td>
</tr>
</tbody>
</table>
5 The Encoder and Decoder

No circuit for encoding the logical qubits into physical qubits is given in any of the references \cite{7,9,8,26} so our first task is to define the encoding operation in the \textit{zx}-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli $X$ operations an 8-qubit GHZ state. In the \textit{zx}-calculus GHZ states have a very simple form:

\[
|00000000\rangle + |11111111\rangle
\]

The bit flips are applied by “copying” a Pauli $X$ from a (logical) input qubit to the appropriate four physical qubits, as shown below.

The final encoder is obtained by composing these four diagrams along the 8 physical qubits; logical qubits come in on the left, codewords come out on the right. The decoder is simply the
The Encoder and Decoder

No circuit for encoding the logical qubits into physical qubits is given in any of the references \[7, 9, 8, 26\] so our first task is to define the encoding operation in the zx-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli X operations an 8-qubit GHZ state. In the zx-calculus GHZ states have a very simple form:

\[
|00000000\rangle + |11111111\rangle
\]

The bit flips are applied by “copying” a Pauli X from a (logical) input qubit to the appropriate four physical qubits, as shown below.

The final encoder is obtained by composing these four diagrams along the 8 physical qubits; logical qubits come in on the left, codewords come out on the right. The decoder is simply the

![Encoding Diagram](image)
The Encoder and Decoder

No circuit for encoding the logical qubits into physical qubits is given in any of the references [7, 9, 8, 26] so our first task is to define the encoding operation in the \( \text{zx} \)-calculus. Looking at Table 1, we can see that the basis states are prepared by conditionally applying Pauli \( X \) operations on an 8-qubit GHZ state. In the \( \text{zx} \)-calculus GHZ states have a very simple form:

\[
\left| 00000000 \right> + \left| 11111111 \right>
\]

The bit flips are applied by “copying” a Pauli \( X \) from a (logical) input qubit to the appropriate four physical qubits, as shown below.

The final encoder is obtained by composing these four diagrams along the 8 physical qubits; logical qubits come in on the left, codewords come out on the right. The decoder is simply the
The Smallest Interesting Colour Code in Quantomatic

Encoding

Enc :=

We note that the graph for $\text{Enc}$ contains no $\text{H}$ vertices and no angles. For such terms we have the following result:

**Proposition 5.1.**

$\mathbb{Z}_2$-calculus term without angles or $\text{H}$ vertices is equal to a term in subset-spanning form; further, the subset-spanning form is quasi-unique and can be computed by a terminating algorithm.

Space does not permit a description of the quasi-normal forms; the important point here is that the normalisation procedure is implemented in a Quantomatic simproc called $\text{rotate\_simp}$.

We will make extensive use of this in the sequel.

**Proposition 5.2.**

Encoding followed by decoding is the identity, i.e.

$$\text{Enc} \xrightarrow{\text{Dec}} \text{id}$$

This result can be proven by $\text{rotate\_simp}$ without human intervention; it requires 66 rewrite steps and takes approximately 5 seconds of real time. See Appendix C.1.1. Note that Proposition 5.2 doesn't establish that the 8 codewords of Table 1 are prepared as required; this is a corollary of the tranversality of the Pauli group, which we show in the next section.

The astute reader will have noticed that the encoder is not actually a quantum circuit: the top "rails" end in a projection onto the state $|\text{+}\rangle$. However, no post-selection is required: we can implement $\text{Enc}$ with a circuit using five ancilla qubits.

**Proposition 5.3.**

$\text{Enc}$ is a unitary embedding.

**Proof.**

We do the proof in two stages, shown below.
The Smallest Interesting Colour Code in Quantomatic

\[ \text{Enc} := \text{Dec} = \text{Enc} \]

We note that the graph for \( \text{Enc} \) contains no \( H \) vertices and no angles. For such terms we have the following result:

Proposition 5.1. Any \( zx \)-calculus term without angles or \( H \) vertices is equal to a term in subset-spanning form; further, the subset-spanning form is quasi-unique and can be computed by a terminating algorithm.

Space does not permit a description of the quasi-normal forms; the important point here is that the normalisation procedure is implemented in a Quantomatic simproc called \( \text{rotate}_\text{simp} \).

We will make extensive use of this in the sequel.

Proposition 5.2. Encoding followed by decoding is the identity, i.e. \( \text{Enc} \; \text{Dec} = \text{id} \).

Pictures:

This result can be proven by \( \text{rotate}_\text{simp} \) without human intervention; it requires 66 rewrite steps and takes approximately 5 seconds of real time. See Appendix C.1.1. Note that Proposition 5.2 doesn't establish that the 8 codewords of Table 1 are prepared as required; this is a corollary of the tranversality of the Pauli group, which we show in the next section.

The astute reader will have noticed that the encoder is not actually a quantum circuit: the top "rails" end in a projection onto the state \( |+\rangle \). However, no post-selection is required: we can implement \( \text{Enc} \) with a circuit using five ancilla qubits.

Proposition 5.3. \( \text{Enc} \) is a unitary embedding.

Proof. We do the proof in two stages, shown below.
The Smallest Interesting Colour Code in Quantomatic

encoder in reverse.

Enc :=

Dec := Enc †

We note that the graph for Enc contains no H vertices and no angles. For such terms we have the following result:

Proposition 5.1.

[23]

A n y zx-calculus term without angles or H vertices is equal to a term in subset-spanning form; further, the subset-spanning form is quasi-unique and can be computed by a terminating algorithm.

Space does not permit a description of the quasi-normal forms; the important point here is that the normalisation procedure is implemented in a Quantomatic simproc called rotate_simp.

We will make extensive use of this in the sequel.

Proposition 5.2.

Encoding followed by decoding is the identity, i.e.

Enc† ‡ Enc = id 3, or in pictures:

This result can be proven by rotate_simp without human intervention; it requires 66 rewrite steps and takes approximately 5 seconds of real time. See Appendix C.1.1. Note that Proposition 5.2 doesn't establish that the 8 codewords of Table 1 are prepared as required; this is a corollary of the transversality of the Pauli group, which we show in the next section.

The astute reader will have noticed that the encoder is not actually a quantum circuit: the top “rails” end in a projection onto the state | + . However, no post-selection is required: we can implement Enc with a circuit using five ancilla qubits.

Proposition 5.3.

Enc is a unitary embedding.

Proof.

We do the proof in two stages, shown below.
Summary of results
Summary of results

1. $\text{Enc}^\dagger \circ \text{Enc} = \text{id}_3$
Summary of results

1. $\text{Enc}^\dagger \circ \text{Enc} = \text{id}_3$

2. $\text{Enc}$ is a unitary embedding.
Summary of results

1. \( \text{Enc}^\dagger \circ \text{Enc} = \text{id}_3 \)

2. \( \text{Enc} \) is a unitary embedding.

3. Find \( P \) satisfying:
   \[
   L = \text{Enc}^\dagger \circ P \circ \text{Enc}
   \]
   where
   \[
   L \in \{ X, Z, \text{CNOT}, \text{CCZ} \}.
   \]
The Smallest Interesting Colour Code in Quantomatic encoder in reverse.

\[ \text{Enc} \quad \text{Dec} \quad \text{Enc}^* = \]

We note that the graph for \( \text{Enc} \) contains no \( H \) vertices and no angles. For such terms we have the following result:

**Proposition 5.1.**

\[ [x, y] - \text{calculus term without angles or } H \text{ vertices is equal to a term in subset-spanning form}; \]

further, the subset-spanning form is quasi-unique and can be computed by a terminating algorithm.

Space does not permit a description of the quasi-normal forms; the important point here is that the normalisation procedure is implemented in a Quantomatic simproc called \( \text{rotate}_s\).

We will make extensive use of this in the sequel.

**Proposition 5.2.**

\[ \text{Encoding followed by decoding is the identity, i.e. } \text{Enc}^* \] \[ \text{Dec} = \text{id}, \]

pictures:

\[ = \]

This result can be proven by \( \text{rotate}_s\) without human intervention; it requires 66 rewrite steps and takes approximately 5 seconds of real time. See Appendix C.1.1. Note that Proposition 5.2 doesn't establish that the 8 codewords of Table 1 are prepared as required; this is a corollary of the transversality of the Pauli group, which we show in the next section.

The astute reader will have noticed that the encoder is not actually a quantum circuit: the top “rails” end in a projection onto the state \( |+\rangle \). However, no post-selection is required: we can implement \( \text{Enc} \) with a circuit using five ancilla qubits.

**Proposition 5.3.**

\[ \text{Enc} \text{ is a unitary embedding.} \]

**Proof.**

We do the proof in two stages, shown below.
Encode-Decode

- Proof:
Encode-Decode

- Proof:

![Diagram](image-url)
Encode-Decide

• Proof:

The Smallest Interesting Colour Code in Quantomatic
• Proof:

The Smallest Interesting Colour Code in Quantomatic
Encode-Decide

• Proof (cont.):

= 

= 

= 

= 

= 

= 

= 

= 

□
STOP!

Quanto time!
arXiv Help

To: Ross Duncan <ross.duncan@strath.ac.uk>
Reply-To: help@arxiv.org

[arXiv #197065] Request oversize exception for submit/191538

Dear Ross,

We can usually get around the timeout issue, but not this time. We encourage you to proceed with uploading the PDF.

It will likely go on hold after it is submitted. Send us an email the next time you try.

--
Jim
arXiv admin
Try it at home:

1. Download quantomatic
   
   http://quantomatic.github.io

2. Download the proof development
   
   https://gitlab.cis.strath.ac.uk/kwb13215/Colour-Code-QPL
4.

Application: the one-way model
Application 1: MBQC

1WQC is a quantum computer design based on single qubit projective measurements on a graph state.

- We can use the ZX-calculus to translate from the 1WQC to circuit model
- Relies on the Hopf algebra normal form
- Produces circuits with minimal space complexity
Graph States

Let $G = (V,E)$ be a simple, undirected graph. Then define:

$|G\rangle = \bigotimes_{(v,u) \in E} CZ_{vu} \bigotimes_{v \in V} |+\rangle$

Viewed as circuit we get this:

![Circuit diagram](image-url)
Graph States

Let $G = (V, E)$ be a simple, undirected graph. Then define:

$$|G\rangle = \bigotimes_{(v, u) \in E} CZ_{vu} \bigotimes_{v \in V} |+\rangle$$

Example 3.9 (Bell state). A Bell state, $|00\rangle + |11\rangle$, is prepared by the circuit below, on the left.

The corresponding $\text{zx}$-calculus derivation is a proof of the correctness of this circuit.

To simplify the next example, we introduce some shorthand for some useful circuit elements.

$Z = H \otimes H |+\rangle$

$|0\rangle = H |\alpha\rangle$

Example 3.10 (1D-cluster state).

A 1-dimensional cluster state consists of a collection of qubits $q_1, \ldots, q_n$ initialised in the state $|+\rangle = |0\rangle + |1\rangle$, which are then entangled by applying a $\cdot Z$ operation to qubits $i$ and $i \neq 1$, and to $i$ and $i + 1$.

The corresponding circuit is shown below:

The $\text{zx}$-calculus translation can be radically simplified by repeated use of the spider rule:

The 1D-cluster is a special case of a more general class of states: graph states. These states are the basis of measurement-based quantum computation, and will play an important role later in the paper.

3.2 Circuit-like diagrams

While the preceding has demonstrated the ease with quantum circuits can be translated into diagrams, there are many diagrams which do not correspond to
Graph States

Let $G = (V,E)$ be a simple, undirected graph. Then define:

$$|G\rangle = \bigotimes_{(v,u) \in E} CZ_{vu} \bigotimes_{v \in V} |+\rangle$$
Graph States

Let $G = (V,E)$ be a simple, undirected graph. Then define:

$$|G\rangle = \bigotimes_{(v,u) \in E} CZ_{vu} \bigotimes_{v \in V} |+\rangle$$

Or in 2D:
The One-Way Model

A graph state, coupled to some input qubits:

The only operation is to measure single qubits in the basis:

\[ |+\alpha\rangle := \frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle) \]

\[ |-\alpha\rangle := \frac{1}{\sqrt{2}}(|0\rangle - e^{i\alpha}|1\rangle) \]
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.

Raussendorf and Briegel. PRL (86) 2001
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.

Finally, any output qubits can be corrected by a local Pauli
The One-Way Model

A graph state, coupled to some input qubits:

* Measured qubits are removed from the cluster;
* The outcome of measurement alters the remaining state.

Finally, any output qubits can be corrected by a local Pauli
Non-determinism

Non-determinism of measurements leads to probabilistic branching

Attempt to control branching by using *adaptive measurements*:

(choice of later measurements depends on the outcome of earlier ones)
Non-determinism

Non-determinism of measurements leads to probabilistic branching

Can we carry out measurement-based computation deterministically?

Attempt to control branching by using *adaptive measurements*:

(choice of later measurements depends on the outcome of earlier ones)
Example: 1-Qubit Unitary

Prepared $|+\rangle$ qubits

$\wedge Z$-gates

Projective measurements
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary
Example: 1-Qubit Unitary

\[= Z_\gamma X_\beta Z_\alpha\]
How to Measure

Suppose we have a measuring device for the standard 1-qubit basis:

$$|0\rangle$$  
Yay!  
$$|1\rangle$$  
Boo!

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ i \end{bmatrix}$$
How to Measure

Suppose we have a measuring device for the standard 1-qubit basis:

\[ |0\rangle \quad \text{Yay!} \quad |\frac{1}{\sqrt{2}} \rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \]

\[ |1\rangle \quad \text{X} \quad |\frac{1}{\sqrt{2}} \rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \quad \text{Yay!} \]
Should be this:

Yay!

$|\pm\rangle = \frac{1}{\sqrt{2}} |\uparrow\rangle \pm \frac{1}{\sqrt{2}} |\downarrow\rangle$

Yay!
Should be this:

\[ |+\rangle \quad \text{Yay!} \quad |\pi\rangle \quad \text{Yay!} \quad |-\rangle \]
Example: Hadamard

Prepared $|+\rangle$ qubit

$\wedge Z$-gate

Projective measurement
Example: Hadamard

Yay!
Example: Hadamard

Yay!
Example: Hadamard

Yay!
Example: Hadamard
Example: Hadamard

Boo!
Example: Hadamard

Boo!
Example: Hadamard

Boo!
Example: Hadamard

π

Boo!
Example: Hadamard

Add correction here:
Example: Hadamard

Add correction here:

 Boo!
Example: Hadamard

Add correction here:

Boo!
Conditional ZX

\[ \rho \mapsto \sum_{v \in 2^S} \left[ \hat{v}(D) \right] \rho \left[ \hat{v}(D) \right]. \]
Example

A measurement in the $|+\rangle, |-\rangle$ basis:

\[ \rho \mapsto \langle +| \rho |+\rangle + \langle +| Z \rho Z |+\rangle = \langle +| \rho |+\rangle + \langle -| \rho |-\rangle \]
Translation from MC to ZX-calculus

We can translate any measurement pattern to a diagram using the table below:

<table>
<thead>
<tr>
<th>$N_i$</th>
<th>$E_{ij}$</th>
<th>$M_i^\alpha$</th>
<th>$X_i^s$</th>
<th>$Z_i^s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Diagram for $N_i$]</td>
<td>![Diagram for $E_{ij}$]</td>
<td>![Diagram for $M_i^\alpha$]</td>
<td>![Diagram for $X_i^s$]</td>
<td>![Diagram for $Z_i^s$]</td>
</tr>
</tbody>
</table>
Example

**CLAIM**: this deterministically computes the CNOT of its input
We removed all the conditional operations, therefore this pattern is deterministic.

Can we do this in general?
Determinism in MBQC

Measurement
Pattern
“low-level program”
Determinism in MBQC

Measurement
Pattern
“low-level program”

implicitly defines

Geometry
“entangled resource”
“graph state”
Determinism in MBQC

- Measurement Pattern
  - "low-level program"
  - implicitly defines
- Geometry
  - "entangled resource"
  - "graph state"
  - can possess
- Flow and GFlow
  - "correction strategy"
Determinism in MBQC

Measurement Pattern
"low-level program"

Geometry
"entangled resource"
"graph state"

Flow and GFlow
"correction strategy"

implicitly defines

can possess

Uniformly Deterministic Pattern

jointly determine
Determinism in MBQC

- Measurement Pattern
  - "low-level program"

- Geometry
  - "entangled resource"
  - "graph state"

- Flow and GFlow
  - "correction strategy"

- Uniformly Deterministic Pattern

implicitly defines

not necessarily the same!
Determinism in MBQC

- Measurement Pattern
  - "low-level program"
- Geometry
  - "entangled resource"
  - "graph state"
- Flow and GFlow
  - "correction strategy"
- Uniformly Deterministic Pattern
- Quantum Circuit

- Implicitly defines
- Can possess
- Jointly determine

Note: not necessarily the same!
Determinism in MBQC

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Uniformly Deterministic Pattern

Quantum Circuit
translates to
with ancilla qubits

not necessarily the same!
The plan:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”
The plan:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”
The plan:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Translation

Graphical form
“direct translation”

Included in

Minimal Graphical Form
“annotated geometry”

Rewrites to
The plan:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Quantum Circuit
if original pattern is deterministic

Graphical form
“direct translation”

Minimal Graphical Form
“annotated geometry”

“flow strategy”

translation
included in
rewrites to
The plan:

Measurement Pattern
  "low-level program"

Geometry
  "entangled resource"  
  "graph state"

Flow and GFlow
  "correction strategy"

Quantum Circuit

Graphical form
  "direct translation"

Minimal Graphical Form
  "annotated geometry"

Circuit-like form
  has flow

if original pattern
is deterministic

"flow strategy"

"gflow strategy"

rewrites to

included in

translation
Hopf Algebra Equivalence

**Thm:** any Hopf algebra expression can be put into normal form:
GFlow Strategy
Flow Strategy
Application 1: MBQC

Result: complicated MBQC implementation to simpler circuit specification.

WIP: the converse, optimally.
Circuit Perspective

\[ X_\alpha - Z_\alpha - X_\alpha \]
Circuit Perspective

Inputs

Outputs
Circuit Perspective

Inputs

Outputs
Circuit Perspective
Circuit Perspective
??? Perspective

\[ \delta \quad \gamma \]

\[ \alpha \quad \beta \]
Hopf algebra expression
??? Perspective

Hopf algebra
normal form

$\delta$ $\gamma$

$\alpha$ $\beta$
MBQC Perspective
MBQC Perspective

Physical qubits
Prepared qubits
MBQC Perspective

Prepared qubits

Measured qubits
Any ZX-calculus term can be interpreted as an MBQC in this way.
NQIT Perspective
NQIT Perspective
Few qubit ion traps
NQIT Perspective

Optical interconnect

Few qubit ion traps
NQIT Perspective

Optical interconnect

Few qubit ion traps
5. Prospectus
I did not talk about:

• Mixed states, CP-maps, probabilities
• Causality
• Contextuality, Non-locality
• Infinite dimensional spaces
I did not talk about:

- Mixed states, CP-maps, probabilities
- Causality
- Contextuality, Non-locality
- Infinite dimensional spaces

**ArXiv papers**

- arXiv:1107.6019
- arXiv:1203.4988
- arXiv:1011.6123
- arXiv:1605.04305
I did not talk about:

• Algorithms: Deutsch-Josza, Grover, Hidden Subgroup (Vicary, Zeng, Gogiosso, Kissinger)

• Quantum Key Distribution (Hillebrand, Kissinger, Tull, Westerbaan)

• Error Correction (RD, Horsman et al)

• Secret sharing (Gogiosso, Perdrix)

• Topological Quantum Computing (Horsman, de Beaudrap)
Now let's take a ribbon-graphdual pair of graphs underlying the Kitaev model on a surface:

The plaquette operators are the same, but with the colours reversed. Since the vertex and plaquette operators commute, we can compose them in any order by matching up the external terminals. For example like this:

Tobias Fritz — recent talk available on PIRSA
What next?

• “ZX-like” theories exist for any quantum system (including codeword spaces!)

• ZX-calculus has no commitment to architecture or causal structure

• ZX-calculus encodes quantum theory “below the gate level”
Extensional / Denotational

\[ \lambda x. \lambda y. \lambda z. xz(yz) \]
Extensional / Denotational

\[ \lambda x. \lambda y. \lambda z. xz(yz) \]

Intensional / Operational
\[ \lambda x. \lambda y. \lambda z. xz(yz) \]

Compilation

Extensional / Denotational

Intensional / Operational
The Next Step

Summary:
Our goal is to develop a flexible intermediate representation for quantum software which enables formally verified program transformations, and based on this, to construct the back-end for a re-targetable optimising compiler for a variety of realistic quantum computer architectures.

Context:
High-level programming languages (HLLs) increase programmer productivity and software reliability, provided that the HLL compiler can generate machine code which runs well on the intended hardware platform. Quantum algorithm designers have the choice of several powerful quantum programming languages. However, proposed implementations of quantum computers vary greatly due to differing underlying technologies (ion traps, superconducting circuits, optics) and architectural concepts (networked vs. hybrid, measurement based, ancilla driven), and no language takes account of the specific characteristics of any given platform. Worse, the technology is evolving quickly, none of these characteristics are stable, and no consensus has yet emerged on the best choice. For classical programs, modern compiler toolchains such as LLVM decouple the HLL from the machine by using an intermediate representation (IR), which is independent of both. By translating to and from the IR, any HLL may be used on any platform. A quantum IR accommodating dissimilar architectures is needed to support software on rapidly shifting hardware.

Targeted breakthrough:
We will define a universal intermediate representation language, called azx, which is platform-agnostic but which facilitates program transformations appropriate to specific hardware. Specifically, azx will provide the following (see §3 for more detail):

- An easy-to-generate universal language.
- Automated IR transformations for error correction, resource optimisation, and execution layout.
- A complete compilation pipeline from HLL to hardware for (i) superconducting transmon qubits (QuTech) and (ii) optically-coupled ion traps (NQIT).

The theory, algorithms, and software architecture behind this will be flexible and extensible, thus permitting programming languages and architectures not explicitly included in the project scope to be supported. This will greatly improve the software ecosystem for quantum computers: by supporting azx, future quantum devices may easily run existing programs, and future programming languages automatically gain support on a wide range of hardware.