Verifying one-way programs with the ZX-calculus
Verification

Does a classical program do what it is supposed to do?

- is it equivalent to its specification?
  - often undecidable
- does it have some desirable property?
  - e.g. never hangs
Verification

Does a MBQC program do what it is supposed to do?

- is it equivalent to its specification?
  - some quantum circuit
- does it have some desirable property?
  - is it deterministic?
The need for abstraction:

This is an 8-bit adder

\[ D \sim 2^{1764} \]
ZX-calculus and Quantum Circuits

Quantum processes, diagrammatically
**ZX-calculus syntax**

**Defn:** A *diagram* is an undirected open graph generated by the above vertices.

**Defn:** Let $\mathcal{D}$ be the dagger compact category of diagrams s.t.

$$ (\cdot)^\dagger : \alpha \mapsto -\alpha $$
**ZX-calculus semantics**

**Defn:** Let $[\cdot] : \mathbb{D} \rightarrow \mathbb{FDHilb}$ be the traced monoidal functor such that $[1] = \mathbb{C}^2$ and define its action on generators by:

$$
\begin{align*}
\left[
\begin{array}{c}
\alpha \\
\end{array}
\right] &= \left\{
\begin{array}{l}
|0\rangle \otimes m \mapsto |0\rangle \otimes n \\
|1\rangle \otimes m \mapsto e^{i\alpha} |1\rangle \otimes n
\end{array}
\right.
\end{align*}
$$

$$
\begin{align*}
\left[
\begin{array}{c}
\alpha \\
\end{array}
\right] &= \left\{
\begin{array}{l}
|+\rangle \otimes m \mapsto |+\rangle \otimes n \\
|\rangle \otimes m \mapsto e^{i\alpha} |\rangle \otimes n
\end{array}
\right.
\end{align*}
$$
Representing Qubits

\[
\begin{bmatrix}
\text{red dot} \\
\text{black dot}
\end{bmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle
\]

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

\[
\begin{bmatrix}
\text{green dot} \\
\text{black dot}
\end{bmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = |+\rangle
\]

\[
\begin{bmatrix}
\text{green dot} \\
\pi
\end{bmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = |\rightarrow\rangle
\]
Representing Phase shifts

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

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

\[ \wedge X = \begin{bmatrix} \text{Green} & \text{Red} \\ \text{Red} & \text{Green} \end{bmatrix} = \begin{bmatrix} \text{Green} & \text{Red} \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \]
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:

$$[D] = U$$
Equations

(bialgebra) = (copying) = (hopf)

\[ \alpha = \ldots \]

\[ \pi \]

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

\[ \pi \]

\[ -\frac{\pi}{2} \]

\[ \alpha = \ldots \]

\[ \alpha \]

\[ \beta \]

\[ \beta \]

(colour change)
The ZX-calculus is sound

Theorem: Let $D_1$ and $D_2$ be ZX-calculus terms such that $D_1 = D_2$ by the equations shown previously; then:

$$[[D_1]] = [[D_2]]$$
Representing Logic Gates

\[ \land X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \]
Reasoning via rewrites:
Reasoning via rewrites:
Reasoning via rewrites:
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Example: 3 CNOTs
Representing Hadamard

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

\[
\begin{align*}
\text{H} &:= \\
\end{align*}
\]
Hadamard Rule

Corollary

\[ \text{Corollary} \]

\[ H \quad \alpha \quad H \]

\[ \text{Friday 30 March 2012} \]
Hadamard Rule

Corollary

\[ \cdots \]

\[ \alpha \]

\[ \cdots \]

\[ \frac{\pi}{2} \]

\[ \frac{\pi}{2} \]

\[ \frac{n\pi}{2} \]

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

Friday 30 March 2012
Hadamard Rule

Corollary

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

\[ = \]

\[ \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \quad \alpha \quad \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \]

\[ = \]

\[ \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \quad \alpha \quad \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \quad \ldots \]
Hadamard Rule

Corollary: total symmetry between red and green
Representing Logic Gates

\[ \land X = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{pmatrix} \]
Representing Logic Gates

\[ \wedge Z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \]
Representing Logic Gates

\[ \wedge Z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \]
Example: 2 CZs
Example: 2 CZs

(spider)

Friday 30 March 2012
Example: 2 CZs
Example: 2 CZs

(H-commute)
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs
Example: 2 CZs

\[H\text{-cancel}\]
Example: 2 CZs
Which diagrams are circuits?

**Defn:** A diagram is called **circuit-like** if:

- All of its \( \alpha \), \( \alpha \), and boundary vertices can be covered by a set of disjoint paths \( P \), each of which ends in an output.

- Every cycle in the diagram which overlaps with 2 paths in \( P \) traverses an edge in the opposite direction to \( P \).

- It is 3-coloured.
Which diagrams are circuits?
Which diagrams are circuits?

YES!
Which diagrams are circuits?
Which diagrams are circuits?
Which diagrams are circuits?

NO :(
Which diagrams are circuits?

**Thm:** if a diagram $D$ is circuit-like then $[D]$ is a unitary embedding.

Being circuit-like is a little stronger than being a unitary embedding: we require a particular gate set, and a particular encoding of the gate set as diagrams, and the minimality w.r.t. to spider.
The 1-Way Quantum Computer

Using measurements to compute
The One-Way Model

Some qubits, each initialised in the state $|+\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$
The One-Way Model

Some qubits, each initialised in the state $|+\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

Entangling operation on pairs of qubits to make a graph state.

$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$
The One-Way Model

A graph state, coupled to some input qubits:

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

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

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.
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, they can be corrected by a 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, they can be corrected by a Pauli
Non-determinism

Non-determinism of measurements leads to probabilistic

Attempt to control branching by using *adaptive measurements*:

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

Non-determinism of measurements leads to probabilistic

Can we carry out measurement-based computation deterministically?

Attempt to control branching by using *adaptive measurements*: (choice of measurements depend on the outcome of earlier ones)
The measurement calculus is a formal syntax for programming the one-way model.

- \( N_i \): initialise qubit \( i \) in the \(|+\rangle\) state
- \( E_{ij} \): entangle qubits \( i \) and \( j \) using a CZ operation
- \( s^{M_i^\alpha} \): measure qubit \( i \) in the basis \(|0\rangle \pm e^{(-1)^s \alpha + t\pi} |1\rangle\)
- \( X_i^s, Z_i^s \): apply a Pauli X or Z operator to qubit \( i \)

The boolean variables \( s, t \) are signals; their values are determined by the results of measuring the corresponding qubit.

- Standard form: operations occur in the order above
  - Every pattern can be standardised
Example: 1-Qubit Unitary

Prepared $|\pm\rangle$ qubits

$\wedge Z$-gates

Projective measurements

$H \gamma H \beta H \alpha$
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

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

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

$|0\rangle$

Yay!

$|1\rangle$

Boo!
How to Measure

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

| 0 \rangle
| 1 \rangle
---
Yay!   | 1
Yay!

Friday 30 March 2012
Should be this:

\[ |+\rangle \quad |\pi\rangle \quad |\rangle \quad |\rangle \]

Yay!  

Yay!
Should be this:

\[ |+\rangle \quad \text{Yay!} \]

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

Prepared $|+\rangle$ qubit

$\wedge Z$-gate

Projective measurement

Friday 30 March 2012
Example: Hadamard
Example: Hadamard

H

Yay!
Example: Hadamard

Yay!
Example: Hadamard
Example: Hadamard
Example: Hadamard

Boo!
Example: Hadamard

Boo!
Example: Hadamard

\[ \pi \]

Boo!
Example: Hadamard

Add correction here:

Boo!
Example: Hadamard

Add correction here:

Boo!
Example: Hadamard

Add correction here:

Boo!
Conditional Operations

Defn: Let $S$ be a set of variables ("signals"); define $\mathcal{D}(S)$ to be the category of diagrams generated by:

where $S' \subseteq S$
Semantics

**Defn:** a *valuation* of $S$ is a function $v : S \rightarrow \{0, 1\}$. For each valuation we can define a map $\hat{v} : \mathbb{D}(S) \rightarrow \mathbb{D}$ by replacing the angle alpha with 0 if

$$\prod_{s \in S'} v(s) = 0$$

and leaving if unchanged otherwise.

**Defn:** The *denotation* of a diagram $D$ in $\mathbb{D}(S)$ is a superoperator:

$$\rho \mapsto \sum_{v \in 2^S} [\hat{v}(D)]\rho[\hat{v}(D)]^\dagger$$
Example

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

\[
\rho \mapsto \langle + | \rho | + \rangle + \langle + | Z \rho Z | + \rangle \\
= \langle + | \rho | + \rangle + \langle - | \rho | - \rangle
\]
Equations for conditional diagrams

(spider)

(anti-loop)

(identity)

(α-commute)

(π-commute)
Equations for conditional diagrams
Equations for conditional diagrams

.... plus the same again with the colours exchanged
Translation from MC to ZX-calculus

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

The CNOT is computed by the pattern:

$$\varphi = \overline{X_4^3 Z_4^2 Z_1^2 M_3^0 M_2^0 E_{13} E_{23} E_{34} N_3 N_4}$$

Which yields the diagram:
Example Rewrite

We removed all the conditional operations, therefore this pattern is deterministic.

Can we do this in general?
Determinism and Flow

... if you like that sort of thing.
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: Flow and GFlow

Jointly determines: Deterministic Pattern

Uniformly Deterministic Pattern
Determinism in MBQC

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

  jointly determine Uniformly Deterministic Pattern

  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

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

not necessarily the same!

translates to
with ancilla qubits
Summary of this talk:

- Measurement Pattern
  - “low-level program”

- Geometry
  - “entangled resource”
  - “graph state”

- Flow and GFlow
  - “correction strategy”
Summary of this talk:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”
Summary of this talk:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Graphical form
“direct translation”

Minimal Graphical Form
“annotated geometry”

Translation

Included in

Rewrites to

Friday 30 March 2012
Summary of this talk:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Graphical form
“direct translation”

Minimal Graphical Form
“annotated geometry”

“flow strategy”

Quantum Circuit
if original pattern is deterministic

Friday 30 March 2012
Summary of this talk:

Measurement Pattern
“low-level program”

Geometry
“entangled resource”
“graph state”

Flow and GFlow
“correction strategy”

Quantum Circuit
“low-level program”

Graphical form
“direct translation”

Minimal Graphical Form
“annotated geometry”

“flow strategy”

Circuit-like form
has flow

if original pattern
is deterministic

“gflow strategy”

rewrites to
Geometry of a pattern

The following pattern computes the CNOT gate:

$$\mathcal{P} = \bar{X}_4^3 Z_4^2 Z_1^2 M_3^0 M_2^0 E_{13} E_{23} E_{34} N_3 N_4$$

Each pattern implicitly defines a geometry \((G, I, O)\)
Geometry of a pattern

The following pattern computes the CNOT gate:

\[ \Psi = \bar{X}_4^3 Z_4^2 Z_1^2 M_3^0 M_2^0 E_{13} E_{23} E_{34} N_3 N_4 \]

Each pattern implicitly defines a \textit{geometry} \((G, I, O)\)

Entanglement graph
Geometry of a pattern

The following pattern computes the CNOT gate:

$$\psi = X_4^3 Z_4^2 Z_1^2 M_3^0 M_2^0 E_{13} E_{23} E_{34} N_3 N_4$$

Each pattern implicitly defines a geometry $(G, I, O)$
Geometry of a pattern

The following pattern computes the CNOT gate:

$$\mathcal{P} = X_4^3 Z_4^2 Z_1^2 M_3^0 M_2^0 E_{13} E_{23} E_{34} N_3 N_4$$

Each pattern implicitly defines a geometry \((G, I, O)\)

Output vertices

Input vertices

Entanglement graph
Flow and determinism

**Defn:** let \((G, I, O)\) be a geometry; a *flow* on \(G\) is a pair \((f, <)\) where \(f : \mathcal{O}^C \to \mathcal{I}^C\) is a function and \(<\) is a partial order s.t.

- \(f(u) \sim u\)
- \(u < f(u)\)
- If \(f(u) \sim v, v \neq u\) then \(u < v\)

Intuitively \(f(u)\) is the qubit which must correct the error produced by measuring \(u\). The partial order guarantees that this is consistent with causality.

**Thm:** if a geometry has a flow there exists a uniformly deterministic pattern on it.
Flow

Input vertices

Output vertices

Friday 30 March 2012
\textbf{Flow}

Input vertices

Output vertices

\textbf{Boo!}

\( v \)

\( f(v) \)
Flow

Input vertices

Output vertices

Boo!

\[ f(v) \]
Flow

Input vertices

Output vertices

$v$  
$f(v)$
The sequence $u, f(u), f^2(u), \ldots$ determines a path from inputs to outputs.

Effectively the flow path” is a “logical qubit” in a circuit equivalent to the pattern.
Geometry of a pattern

We can define a diagram directly from the geometry \((G, I, O)\)
Geometry of a pattern

We can define a diagram directly from the geometry \((G, I, O)\)

\[
\Gamma \varphi_{\wedge x} = \begin{array}{c}
\text{inputs} \\
1
\end{array} \quad \begin{array}{c}
\text{outputs} \\
4
\end{array} \quad 2 \quad 3
\]

\[
D(\Gamma \varphi_{\wedge x}) = \begin{array}{c}
1
\end{array} \quad \begin{array}{c}
\text{H} \\
\text{H}
\end{array} \quad \begin{array}{c}
2
\end{array} \quad \begin{array}{c}
3
\end{array} \quad \begin{array}{c}
\text{H} \\
\text{H}
\end{array} \quad 4
\]

**Theorem:** If geometry \(G\) has a flow then \(D(G)\) is (equivalent to) a circuit-like diagram.
Getting back to the pattern
Getting back to the pattern

We have $D(G(P))$ and $D(P)$. What is the relation?
Getting back to the pattern

We have $D(G(P))$ and $D(P)$. What is the relation?

Define $D^*(P)$ by adding pieces to $D(G(P))$: 
Getting back to the pattern

We have $D(G(P))$ and $D(P)$. What is the relation?

Define $D^*(P)$ by adding pieces to $D(G(P))$:

Prop: $D(P)$ rewrites to $D^*(P)$
Flow Strategy

If the geometry has a flow we can define a rewrite strategy to propagate the “errors” forward:
Flow Strategy
Flow Strategy

\[ \alpha \xrightarrow{} H \quad \pi, s_1 \xrightarrow{} H \quad \pi, s_1 \\]

\[ \beta \quad \pi, s_2 \quad \pi, s_2 \]
Flow Strategy
Flow Strategy

\[ \alpha \rightarrow H \rightarrow \pi, s_2 \rightarrow H \rightarrow \pi, s_1 \rightarrow \beta \rightarrow \pi, s_2 \rightarrow \pi, s_1 \]
Flow Strategy

\[
\begin{align*}
\alpha & \rightarrow H & \rightarrow \pi, s_2 & \rightarrow \beta & \rightarrow \pi, s_2 \\
& & & & \rightarrow \pi, s_1 \\
& & & & \rightarrow \pi, s_1
\end{align*}
\]
Flow Strategy

\[ \alpha \rightarrow H \rightarrow \pi, s_2 \rightarrow \pi, s_2 \rightarrow \beta \]
Flow Strategy
Flow Strategy
Flow Strategy
Flow Strategy
Flow Strategy
Flow Strategy

\[ D_1 = \begin{cases} \pi, \{s\} \\ \alpha \end{cases} \quad D_2 = \begin{cases} \alpha \\ \pi, \{s\} \end{cases} \]

\[ [D_1] \neq [D_2] \text{ unless } \alpha = 0 \text{ or } \alpha = \pi \]

From here we can reconstruct the original pattern given by DK as the uniformly deterministic pattern based on \( G \).
Flow Strategy

**Thm:** This strategy terminates in one of two states:

- All conditional operations removed $\Rightarrow$ deterministic
- Not all removed, but their location reveals where extra corrections are needed.
Gflow rewrite strategy

aka: proving special cases of general facts the hard way
Generalised Flow is the generalisation of flow where each qubit has a set of correcting qubits.

**Theorem** [BKMP 2007]: A geometry has generalised flow iff it supports a step-wise strongly uniformly deterministic pattern.

**Theorem** [DP 2010] If geometry has flow then the corresponding diagram can rewrite to a circuit-like diagram.
GFlow and determinism

**Defn:** let \((G, I, O)\) be an open graph; a gflow on \(G\) is a pair \((g, \prec)\) where \(g : O^C \to \mathcal{P}(I^C)\) is a function and \(\prec\) is a partial order s.t.

- If \(v \in g(u)\) then \(u \prec v\)
- \(u \in \text{Odd}(g(u))\)
- If \(v \in \text{Odd}(g(u)) \setminus \{u\}\) then \(u \prec v\)

\[ v \prec u \quad \text{and} \quad u \prec v \]
GFlow and determinism

**Defn:** let \((G, I, O)\) be an open graph; a gflow on \(G\) is a pair \((g, <)\) where \(g : O^C \rightarrow \mathcal{P}(I^C)\) is a function and \(<\) is a partial order s.t.

- If \(v \in g(u)\) then \(u < v\)
- \(u \in \text{Odd}(g(u))\)
- If \(v \in \text{Odd}(g(u)) \setminus \{u\}\) then \(u < v\)

\[
\begin{align*}
&v < u \quad | \quad u < v
\end{align*}
\]

**Thm:** a geometry has a gflow *if and only if* there exists a uniformly deterministic pattern on it.
If a graph state $G$ can be 2-coloured, we can eliminate the $H$:

This will be useful later!
Hopf Algebra Equivalence

Recall that the red and green operators form a Hopf Algebra with a trivial antipode:
**Thm:** any Hopf algebra expression can be put into normal form:
Hopf Algebra Equivalence

Cor: this rewrite is admissable:
Lemma 2.23 (Main Lemma). Given a diagram $D$, let $\mathcal{X} = \{x_0, \ldots, x_k\}$ and $\mathcal{Z} = \{z_0, \ldots, z_\ell\}$ be sets of its $X$ and $Z$ vertices respectively, such that the subdiagram $G$, induced by $\mathcal{Z} \cup \mathcal{X}$, is bipartite—that is, for all $i, j$, we have $x_i \not\sim x_j$ and $z_i \not\sim z_j$ in $G$.

Define a new graph $G'$ with vertices $V_{G'} = V_G \cup \{u_1, \ldots, u_\ell\} \cup \{v_1, \ldots, v_\ell\}$, and such that for any $0 \leq i \leq k$ and $1 \leq j \leq \ell$,

- there are edges $(u_j, v_j), (u_j, x_{j-1})$ and $(u_j, x_j) \in G'$;
- there is an edge $(x_i, z_0) \in G'$ iff $x_i \in \text{Odd}_G(\mathcal{Z})$;
- there is an edge $(x_i, v_j) \in G'$ iff $x_i \in \text{Odd}_G(\{z_j, \ldots, z_\ell\})$.

Then $G \leftrightarrow^* G'$. 
GFlow Strategy

What about GFlow?

- If a pattern has a GFlow, there exists a strategy that will transform the diagram into a diagram with flow, and no ancilla qubits or acausal loops.

- This strategy is based on the Hopf algebra normal form equivalence for graph states. (aka Main Lemma)
GFlow Strategy
GFlow strategy

Thm: if a diagram has a flow, it is circuit-like
Non-uniform determinism

Note the ZX-calculus works with the pattern, not the geometry, so it can also treat non-uniform cases. E.g.:
Papers

MBQC - classic


MBQC - categorical