*Vy*ZX

Formal Verification of a Graphical Language

Adrian Lehmann Benjamin Caldwell Bhakti Shah Robert Rand

Department of Computer Science
University of Chicago
Presented at MWPLS'23

The ZX Calculus...

... is a graphical language for reasoning about quantum systems

ZX diagrams are open graphs consisting of green "Z" or red "X" "spiders" and "connections" between them

The Z and X spider

The ZX Calculus...

... is a graphical language for reasoning about quantum systems

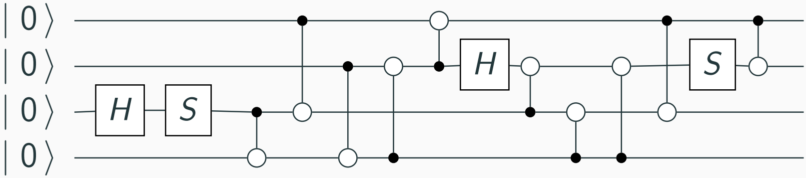
ZX diagrams are open graphs consisting of green “Z” or red “X” “spiders” and “connections” between them

The Z and X spider

Used for compilation, simulation, error correction & more

Key benefit: Diagrammatic rewrites complete & more comprehensible than circuits or matrices

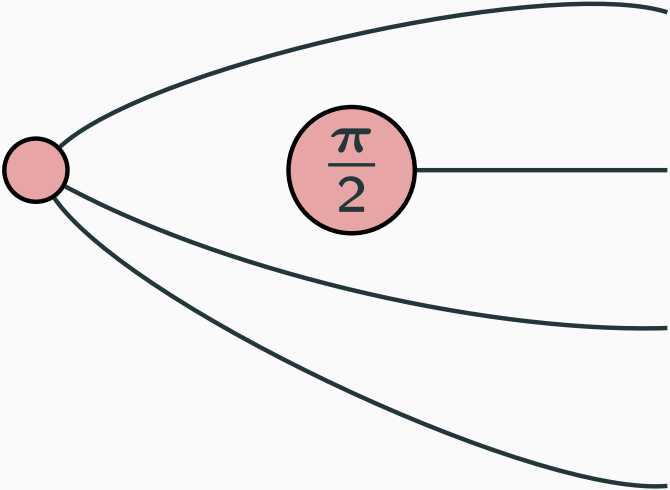
Example: Entanglement (Van de Wetering)



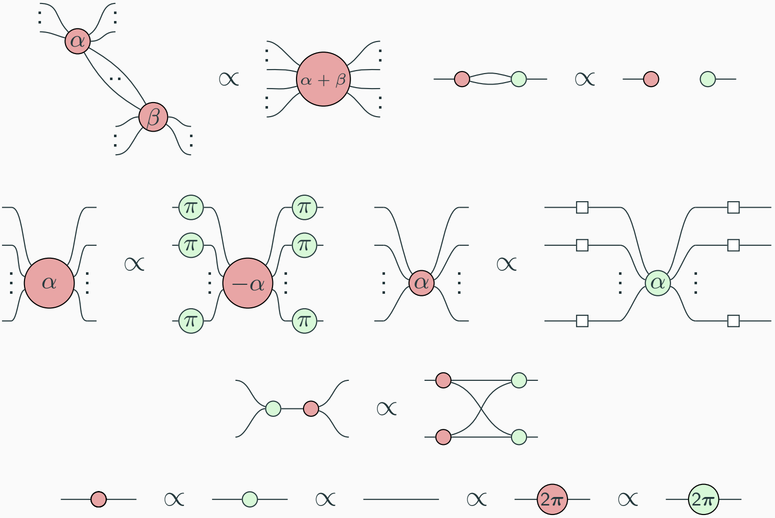
Example: Entanglement (Van de Wetering)

$$[1\ \ 0\ \ 0\ \ 0\ \ i\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ i]^\top$$

Example: Entanglement (Van de Wetering)



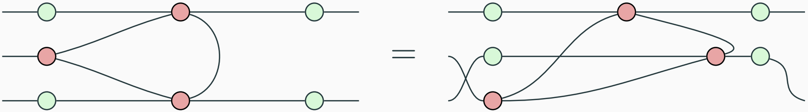
Rewriting Diagrams



Spider Fusion, The Hopf Rule, the Bi-pi Rule (Pi-Copy), Bi-hadamard Rule, the Bialgebra rule, and the identity rule

Only Connectivity Matters

We can freely move spiders around, as long as their connections and in/outputs remain the same



Reasoning Tools

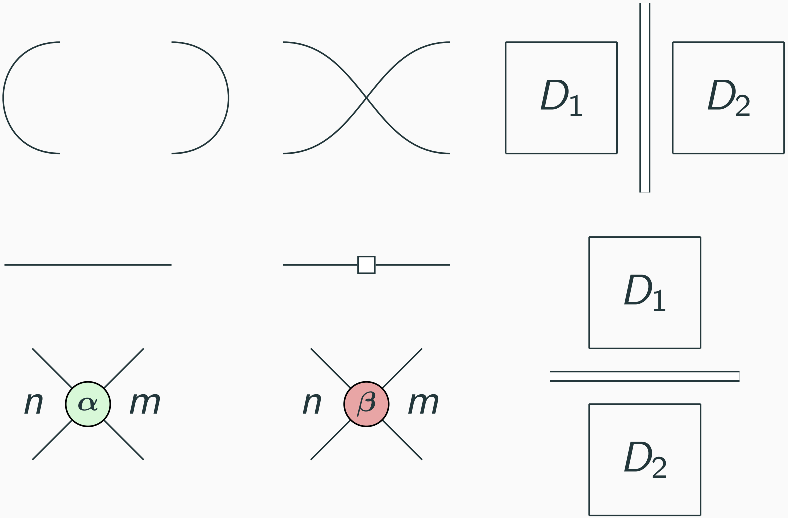
Unverified / Fully axiomatic:

- Quantomatic (https://quantomatic.github.io/)
- ZX Calculator (zx.cduck.me)
- Chyp

"Verify" by property testing:

- PyZX (https://github.com/Quantomatic/pyzx)

ZX Diagrams as string diagrams



Inductive ZX Diagrams

To define our ZX diagrams, we take these string diagram constructions and add Z and X spiders.

$$\frac{\text{in out} : \mathbb{N} \qquad \alpha : \mathbb{R}}{\texttt{Z\_Spider}\ \text{in out}\ \alpha : \texttt{ZX}\ \text{in out}} \qquad \frac{\text{in out} : \mathbb{N} \qquad \alpha : \mathbb{R}}{\texttt{X\_Spider}\ \text{in out}\ \alpha : \texttt{ZX}\ \text{in out}}$$

$$\frac{}{\texttt{Cap} : \texttt{ZX}\ 0\ 2} \qquad \frac{}{\texttt{Cup} : \texttt{ZX}\ 2\ 0} \qquad \frac{}{\texttt{Swap} : \texttt{ZX}\ 2\ 2} \qquad \frac{}{\texttt{Empty} : \texttt{ZX}\ 0\ 0}$$

$$\frac{\text{zx1} : \texttt{ZX}\ \text{in mid} \qquad \text{zx2} : \texttt{ZX}\ \text{mid out}}{\texttt{Compose}\ \text{zx1 zx2} : \texttt{ZX}\ \text{in out}} \qquad \frac{}{\texttt{Wire} : \texttt{ZX}\ 1\ 1}$$

$$\frac{\text{zx1} : \texttt{ZX}\ \text{in1 out1} \qquad \text{zx2} : \texttt{ZX}\ \text{in2 out2}}{\texttt{Stack}\ \text{zx1 zx2} : \texttt{ZX}\ (\text{in1} + \text{in2})\ (\text{out1} + \text{out2})} \qquad \frac{}{\texttt{Box} : \texttt{ZX}\ 1\ 1}$$

Semantics

To verify transformations on diagrams, we introduce a system of semantics. Our semantics system will rely on QuantumLib.

$$\texttt{Z\_Spider n m } \alpha \quad \mapsto \quad \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & 0 \\ 0 & 0 & e^{i\alpha} \end{bmatrix}$$

$$\texttt{X\_Spider n m } \alpha \quad \mapsto \quad H^{\otimes m} \times \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & 0 \\ 0 & 0 & e^{i\alpha} \end{bmatrix} \times H^{\otimes n}$$

More Semantics

$$
\begin{aligned}
\texttt{Cap} &\mapsto \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix} \\
\texttt{Cup} &\mapsto \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}^\top \\
\texttt{Swap} &\mapsto \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
\texttt{Empty} &\mapsto \begin{bmatrix} 1 \end{bmatrix} \\
\texttt{Wire} &\mapsto I_{2\times 2} \\
\texttt{Box} &\mapsto H \\
\texttt{Compose zx1 zx2} &\mapsto \texttt{semantics(zx2)} \times \texttt{semantics(zx1)} \\
\texttt{Stack zx1 zx2} &\mapsto \texttt{semantics(zx1)} \otimes \texttt{semantics(zx2)}
\end{aligned}
$$

Proportionality

Equivalence in ZX is up to constant factor

We define `proportionality` and use symbol $\propto$:

$$\exists c \neq 0 : \texttt{semantics}(zx1) = c * \texttt{semantics}(zx2) \implies zx1 \propto zx2$$

Allows Coq's rewriting capabilities in our proofs about diagrams

Why semantics?

- Smaller TCB
- Interoperability

Why semantics?

- Smaller TCB
- Interoperability

  We can ingest quantum circuits using sqir

  Circuit structure is very different

  Convert circuit components

  Prove equivalence through ground truth

Why semantics?

- Smaller TCB
- Interoperability
  - We can ingest quantum circuits using sqir
  - Circuit structure is very different
  - Convert circuit components
  - Prove equivalence through ground truth
- *Vy*ZX can calculate results

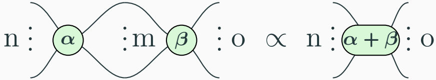
Three Proof Strategies

1. Proof through semantics

Three Proof Strategies

1. Proof through semantics

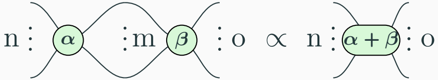
2. Inductive proof



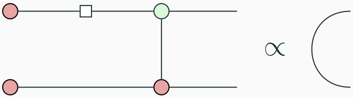
Three Proof Strategies

1. Proof through semantics

2. Inductive proof

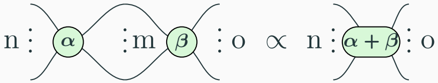


3. Diagrammatic proof

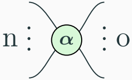


Three Proof Strategies

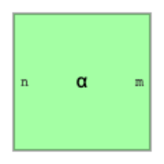
2. Inductive proof

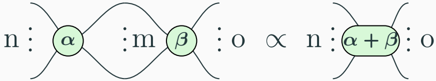


Other representation

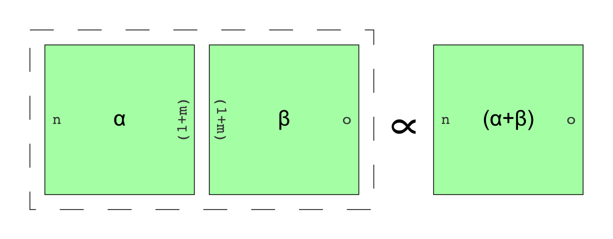


$$\widehat{=}$$



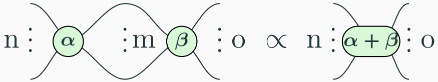


The lemma:

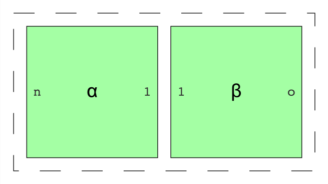


Let's induct over m

2. Inductive proof: Absolute Fusion

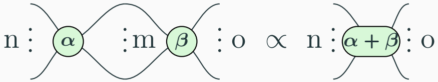


Base case:

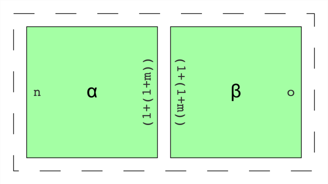


Solve with matrix semantics!

2. Inductive proof: Absolute Fusion

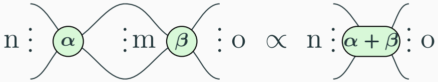


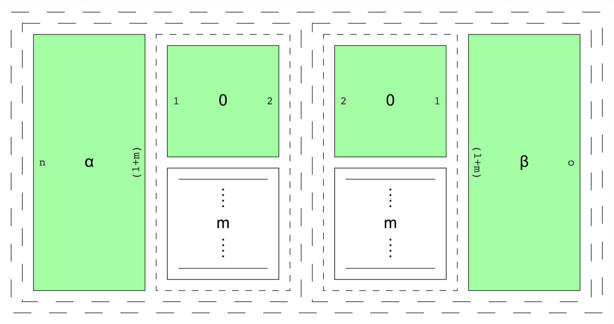
Inductive step:



Let's split out nodes to reduce size

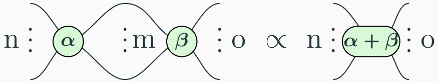
2. Inductive proof: Absolute Fusion



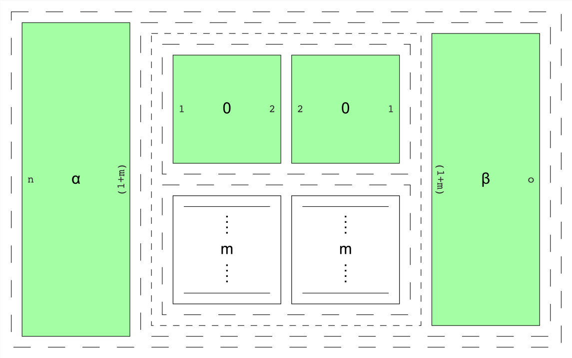


Idea: Fuse the small spiders. Problem: Association

2. Inductive proof: Absolute Fusion

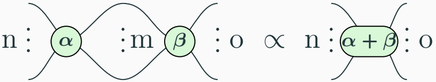


Reassociated diagram

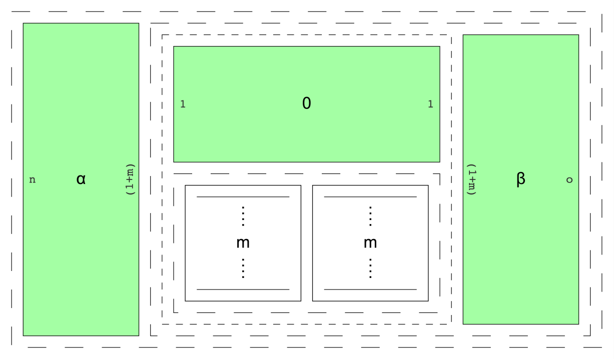


Let's fuse the small spiders

2. Inductive proof: Absolute Fusion



Upper spiders are fused



Apply identity rule, remove wires

2. Inductive proof: Absolute Fusion

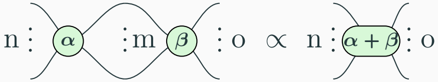
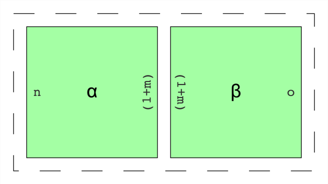
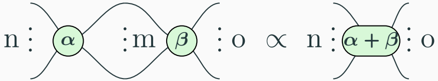


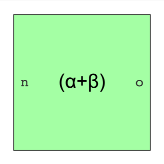
Diagram now corresponds to the IH



2. Inductive proof: Absolute Fusion



Qed.



Discussion

- Any ZX diagram can be expressed
- Multiple ways to encode
- Deal with associativity information
- Dimensionality issues

How can I verify my graphical language?

- Find underlying categorical structure
- Formally extend structure
- Translate into proof assistant
- Deal with resulting associtativity issues

Future work

- Restore connection information
- Verify ZX-based compiler
- Prove ZX results

Summary

- Defined ZX diagrams inductively
- Inspired by string diagrams
- Multiple proof strategies
- ZX calculus is interesting!

Find *Vy*ZX on GitHub

https://github.com/inQWIRE/VyZX

arXiv

Coming soon...

References

- Bob Coecke and Aleks Kissinger, *Picturing quantum processes: A first course in quantum theory and diagrammatic reasoning*, Cambridge University Press, 2017.
- Jonathan Castello, Patrick Redmond, and Lindsey Kuper, *Inductive diagrams for causal reasoning*, 2023.
- Kesha Hietala, Robert Rand, Shih-Han Hung, Xiaodi Wu, and Michael Hicks, *A verified optimizer for quantum circuits*, Proc. ACM Program. Lang. **5** (2021), no. POPL.
- John van de Wetering, *Zx-calculus for the working quantum computer scientist*, 2020.