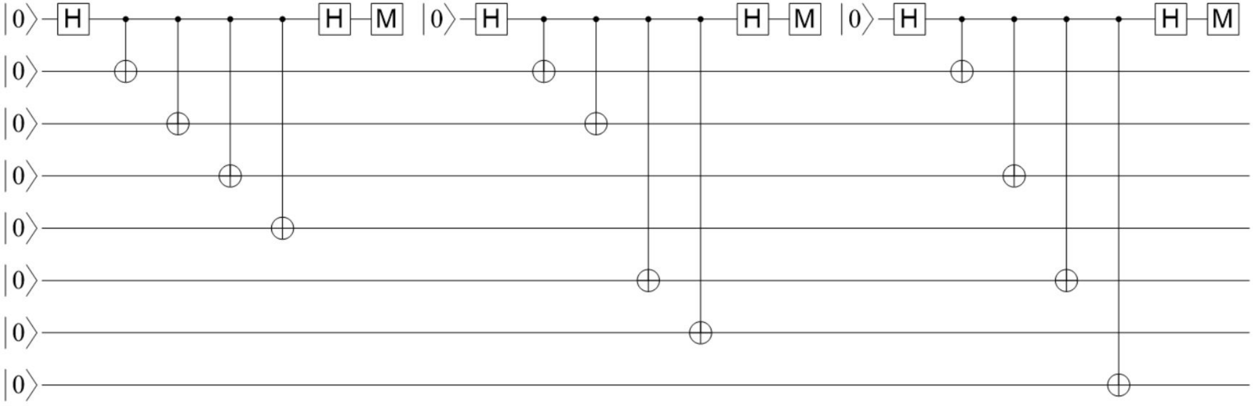
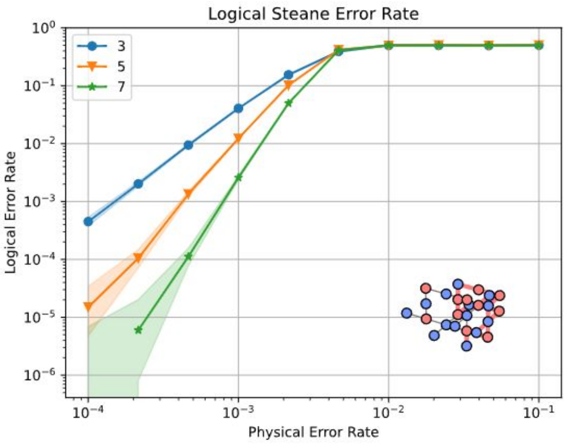
tqec: optimizing and simulating lattice surgery

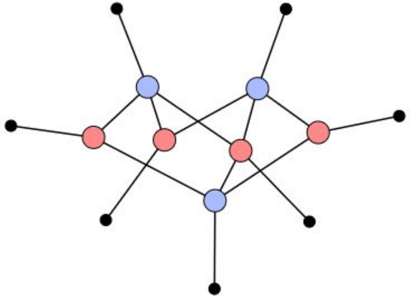
https://groups.google.com/g/tqec-design-automation

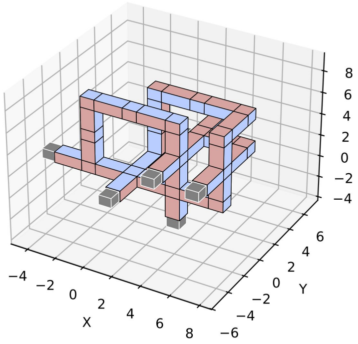
Adrien Suau, Yiming Zhang, Purva Thakre, Sam Burdick, Yilun Zhao, Ricky Young, Arabella Schelpe, Ángela Elisa Álvarez, Victory Omole, Gian Giacomo Guerreschi, Kabir Dubey, Jose A Bolanos, Tianyi Hao, Reinhard Stahn, Jerome Lenssen, Moritz Schmidt, Mohammed Imaduddin, Brendan Reid, Milo Moses, Sean Collins, Mark Agib, Kwok Ho Wan, Austin Fowler

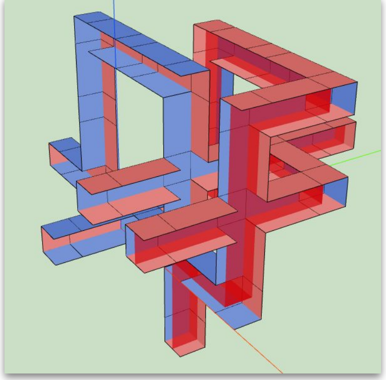


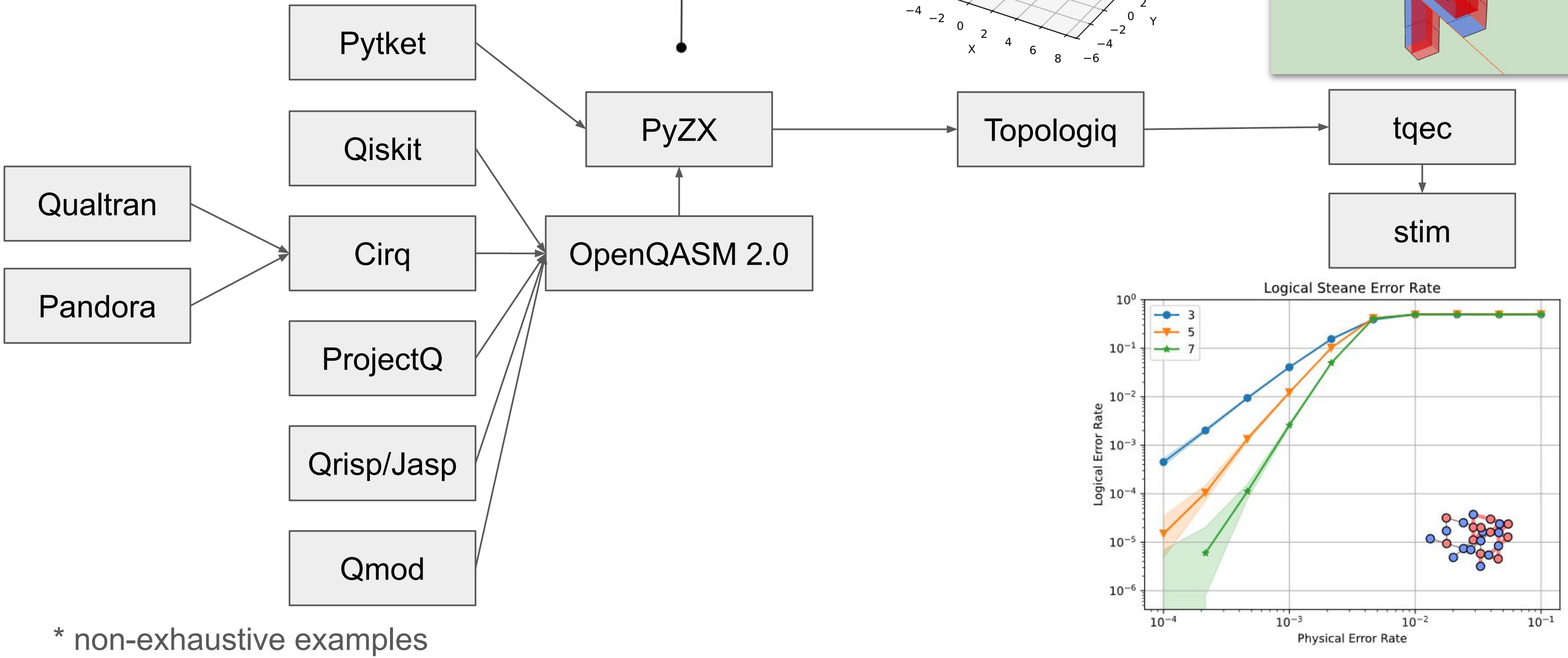


Program flow*



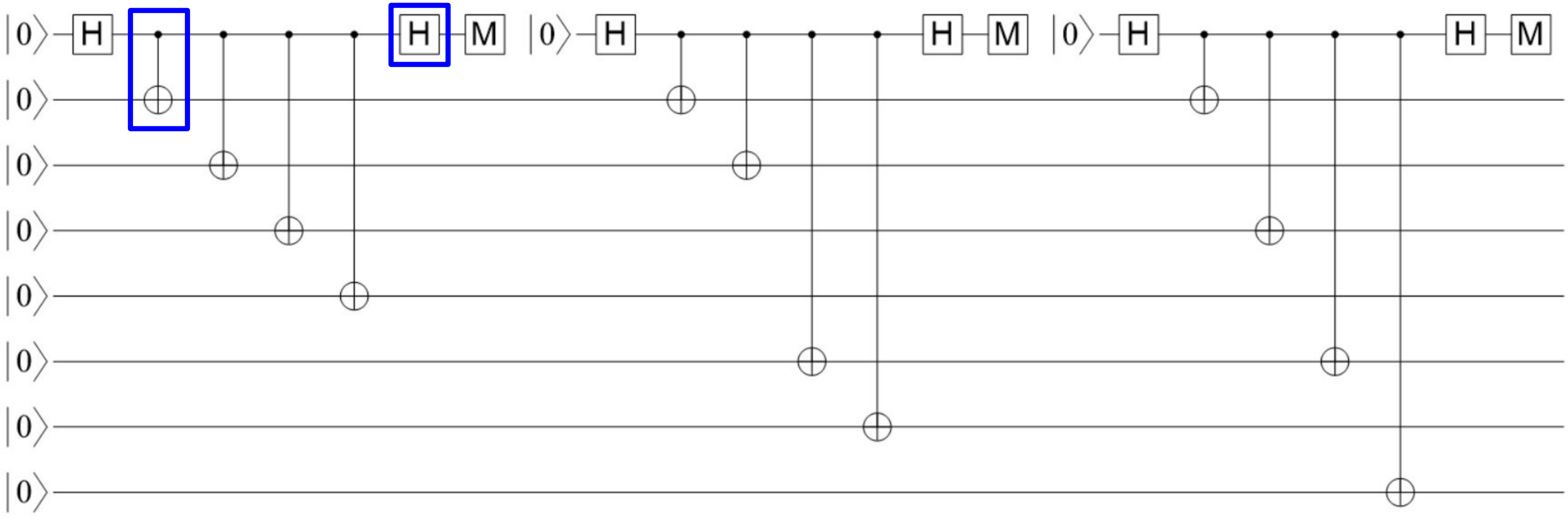






* non-exhaustive examples

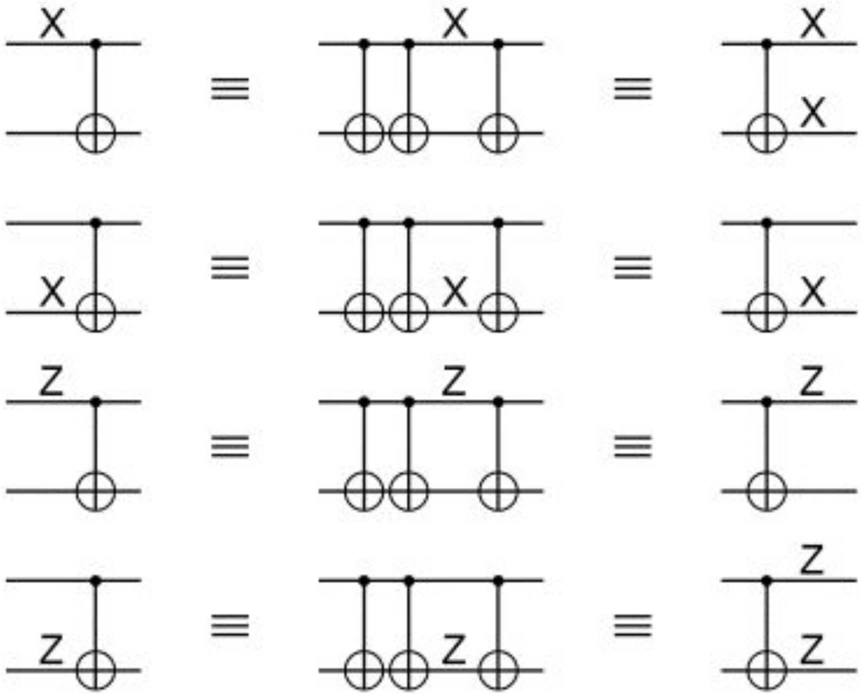
Let's work through an example step by step



This circuit prepares a 7 qubit Steane code, but all that really matters is that it's made of gates that we can realize in tqec.

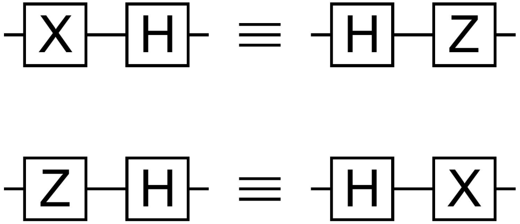
Focus initially on CNOT and Hadamard.

Gates as maps of matrices



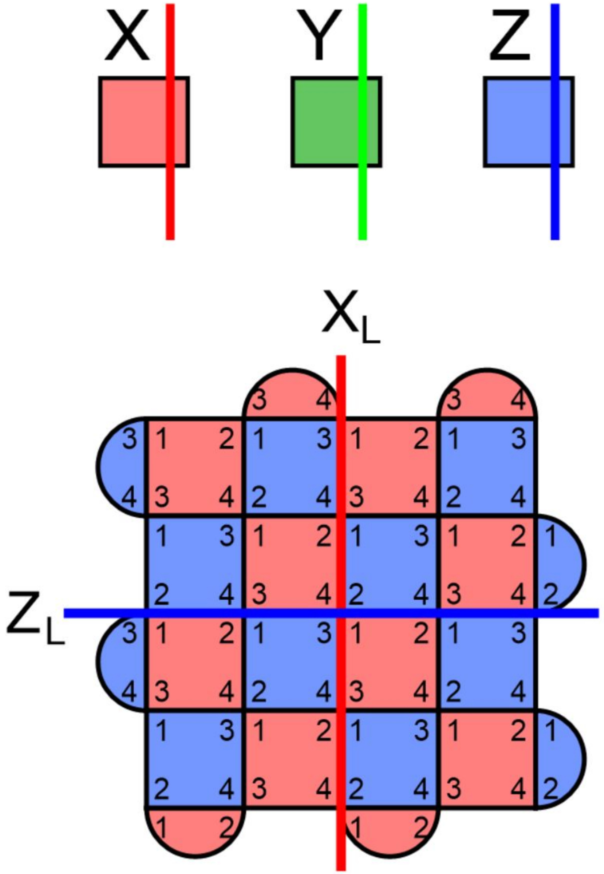
CNOT gates copy X on control, and Z on target to both outputs.

Hadamard converts X into Z and vice versa.



Any process that maps logical operators as shown also implements that gate.

RGB = XYZ mnemonic

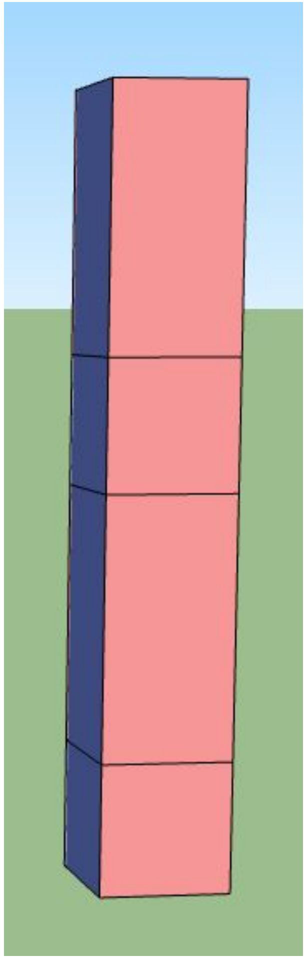


0 rounds

d rounds

0 rounds

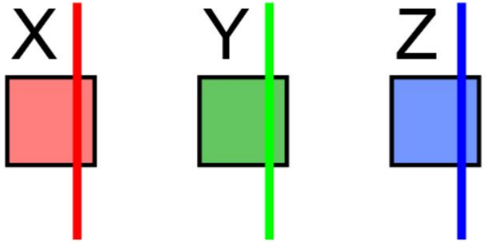
d rounds

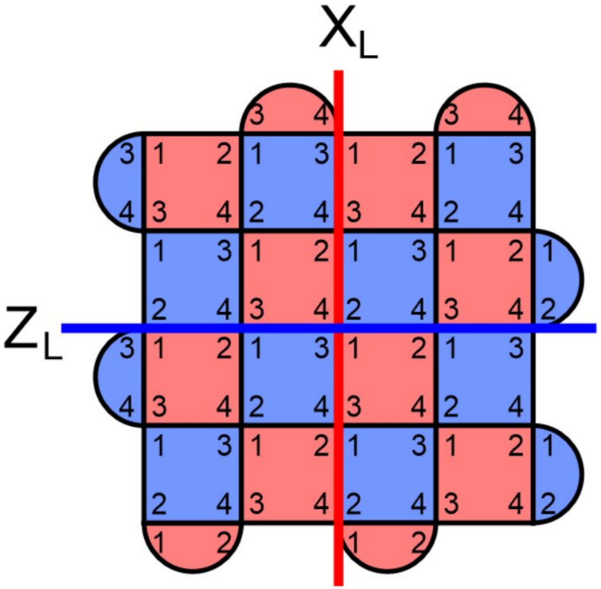


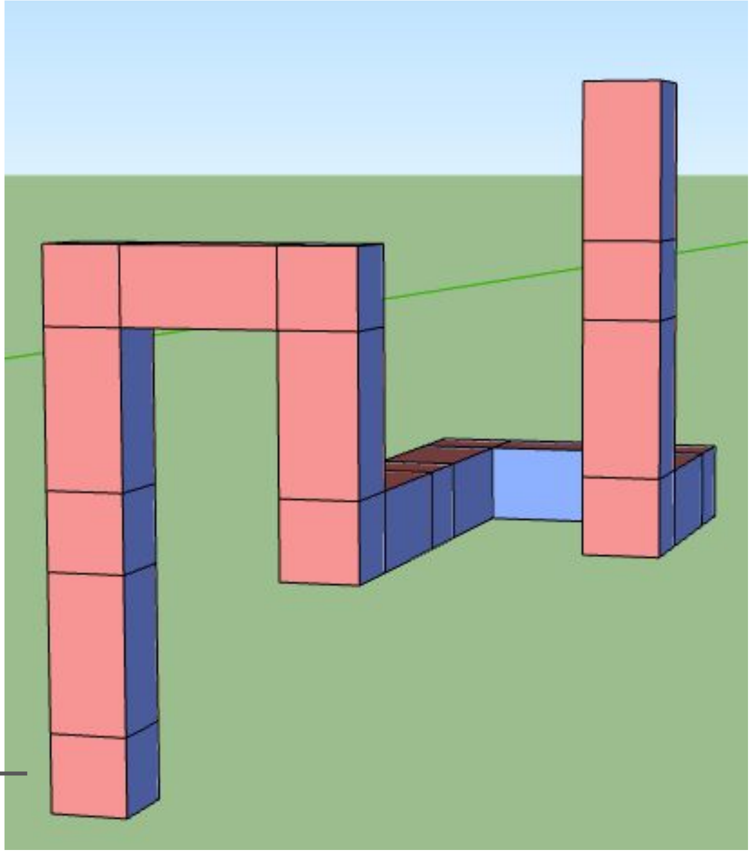
Logical operators stick to walls of the same color.

This is 2d rounds of memory.

RGB = XYZ mnemonic



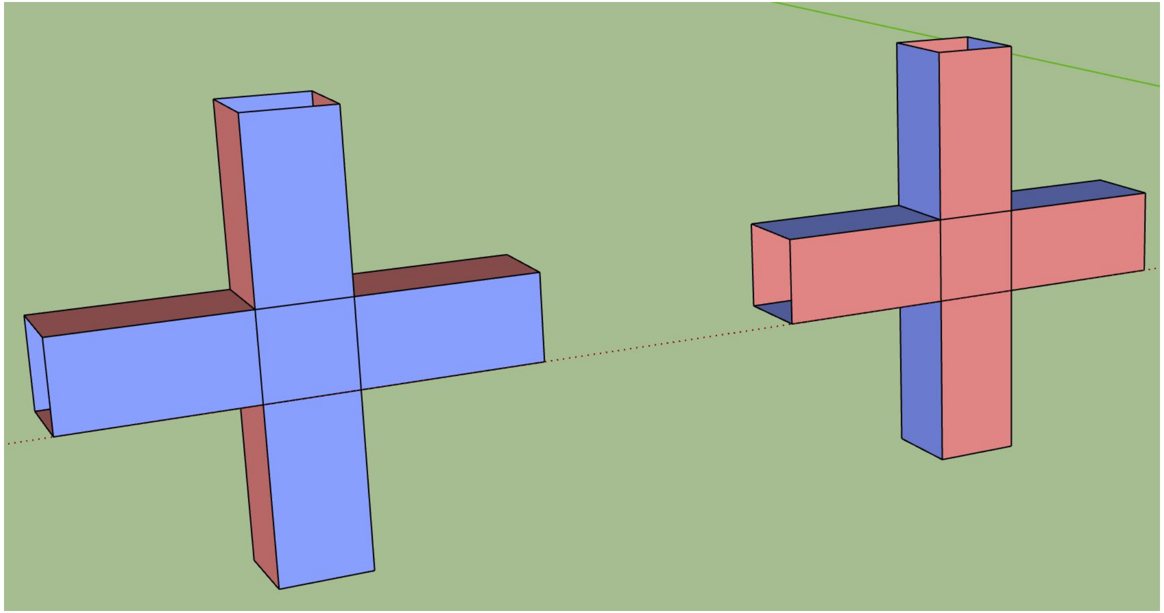




This is also just memory

Slices of the structure tell you what the computer should be doing at every instant of time

What about junctions?



blue node

red node

Logical operators with opposite color to the node must propagate to all edges, same color logical operators must touch an even number of edges.

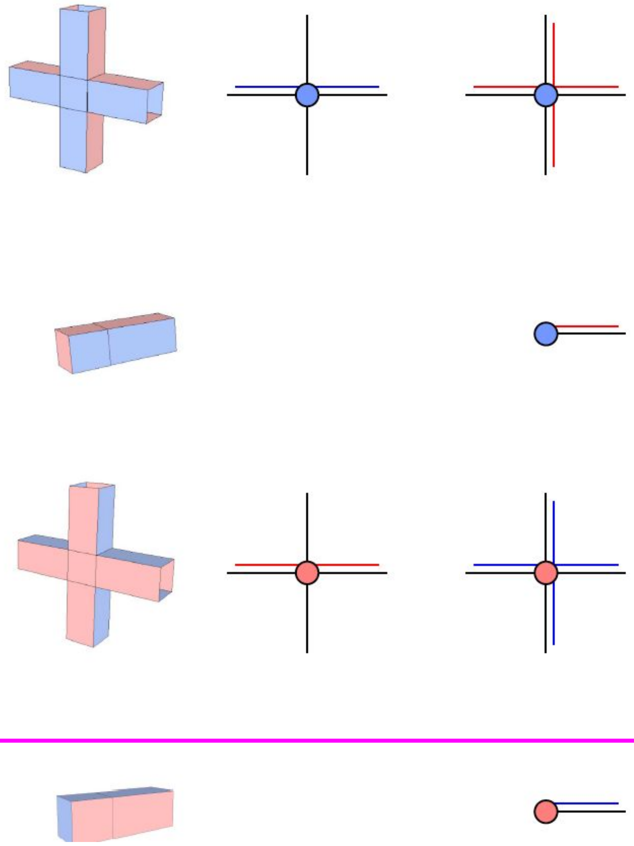
Let’s formalize these rules

a)

b)

c)

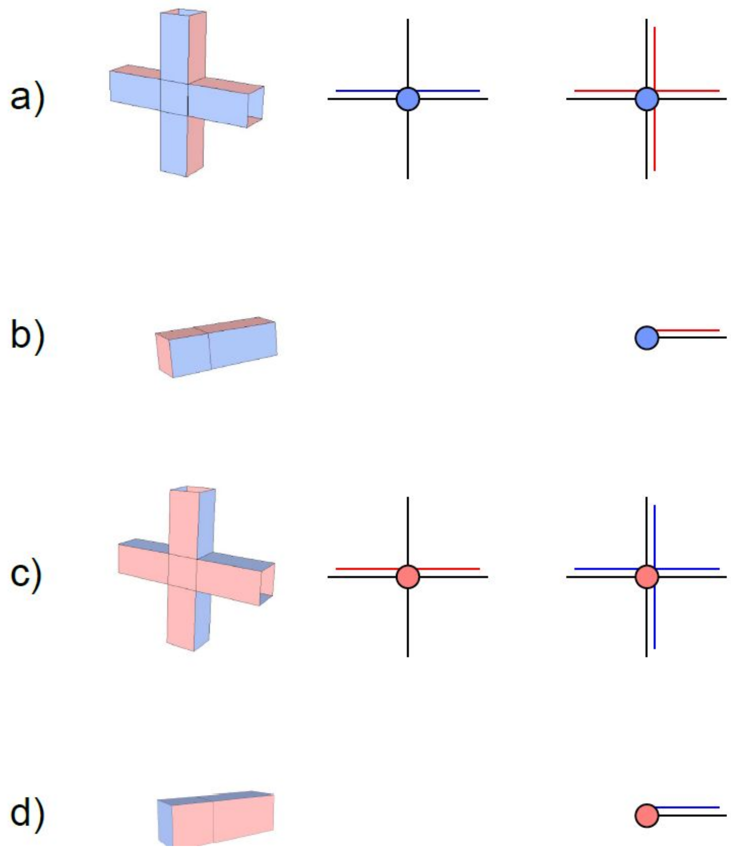
d)

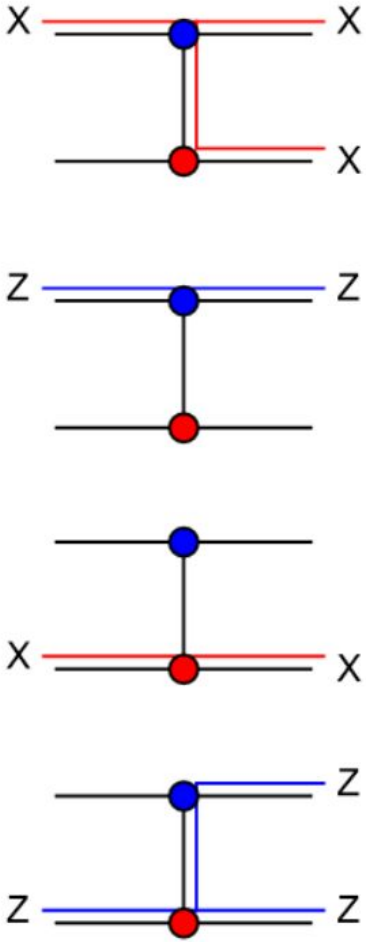


Logical operators with opposite color to the node must propagate to all edges, same color logical operators must touch an even number of edges.

Prep |0> or Z basis measurement

Let’s use these rules



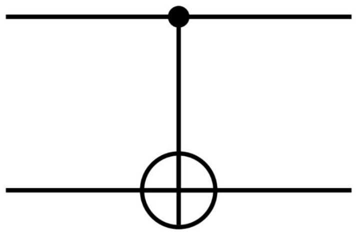


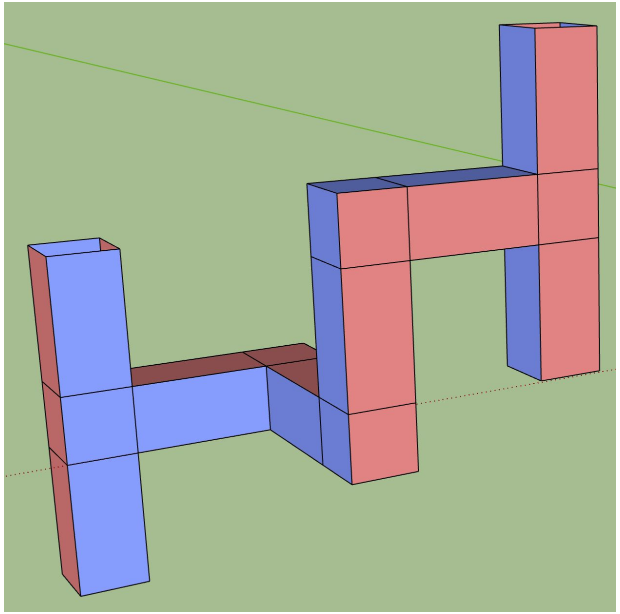
A lattice surgery CNOT is a blue node (control) connected to a red node (target)

These paths are called correlation surfaces, basically a set of measurements that gives you sign information.

CNOT

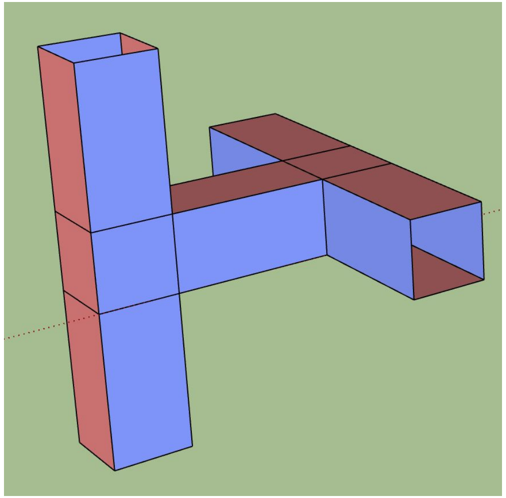
Control = blue node
Target = red node





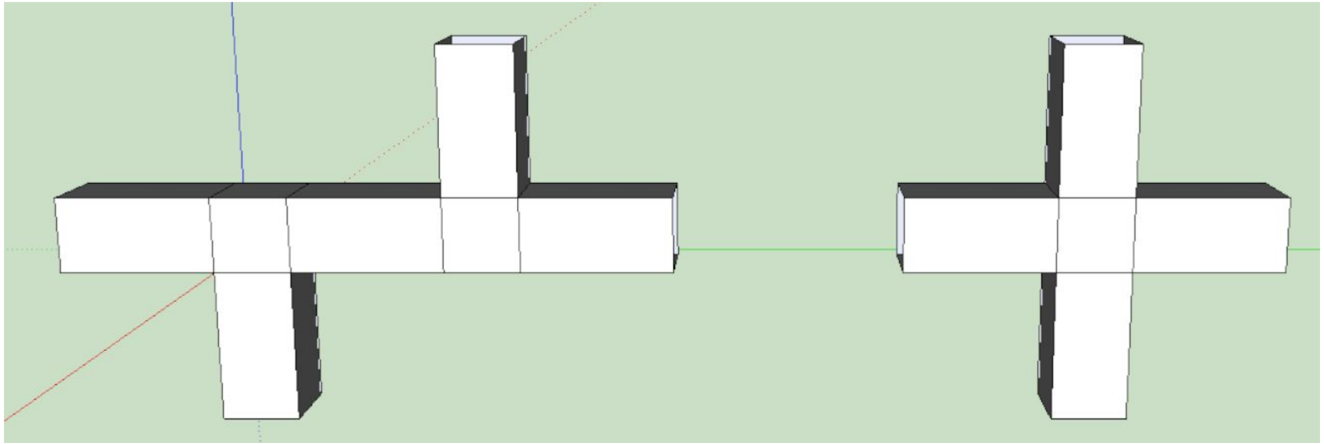
This is CNOT

This is also CNOT

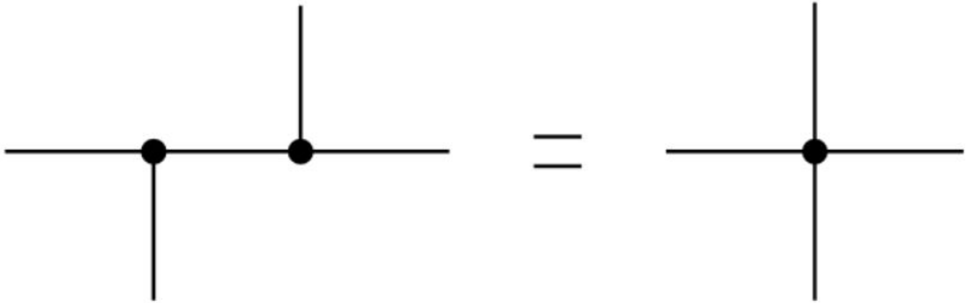


Provided you get the nodes right, the rest is arbitrary noodles

Fun with structures

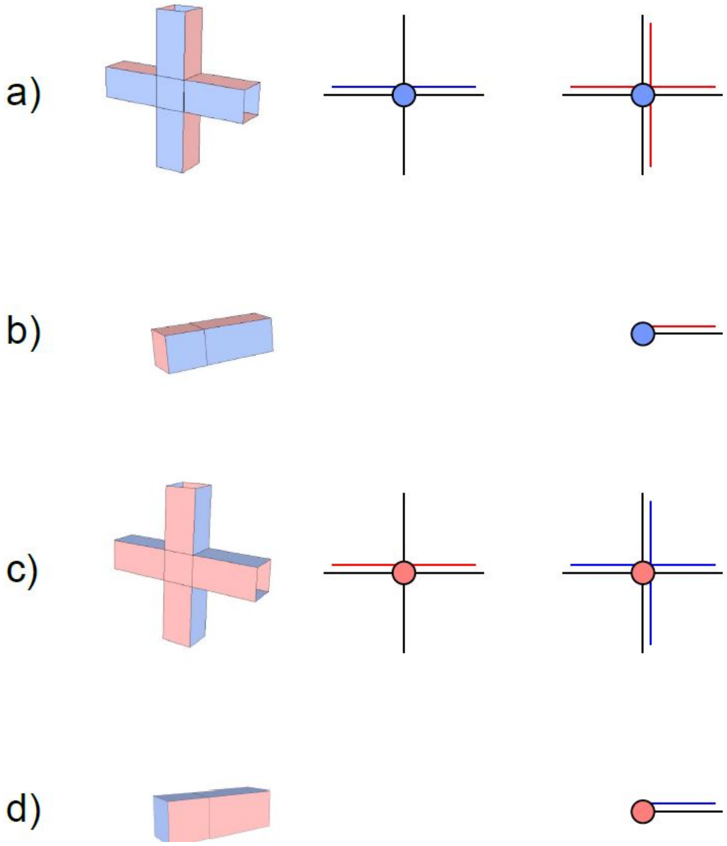


Nodes of the same color connected by an edge can be merged/unmerged.

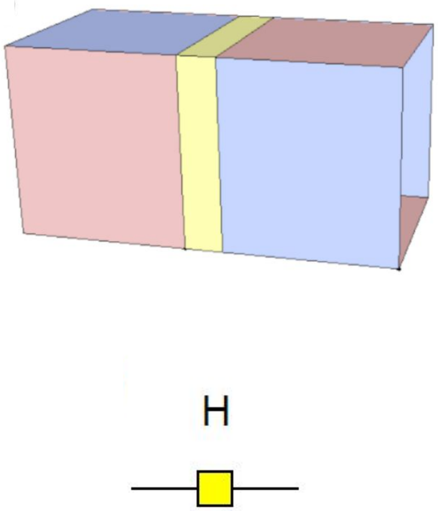


This is called the "spider rule"

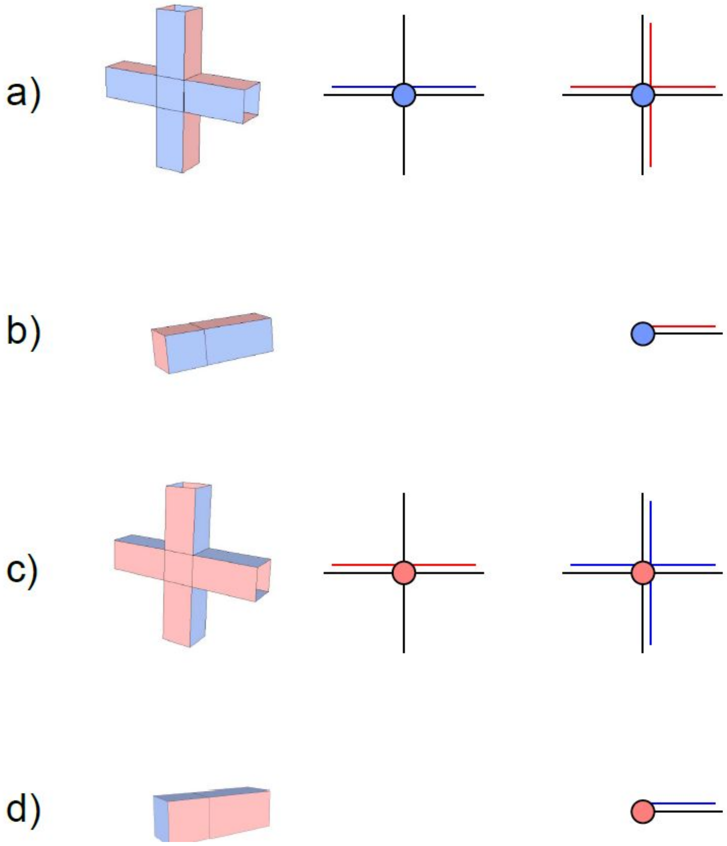
One more logical gate

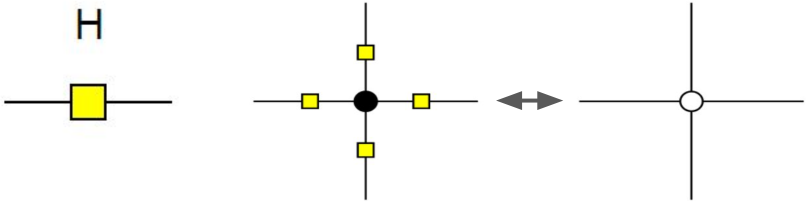


Logical operators that pass through a layer of Hadamard gates are interchanged X↔Z.

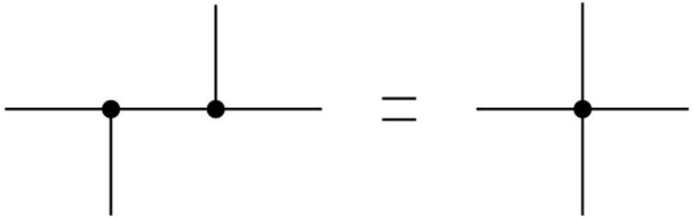


All the structures and rules we need…





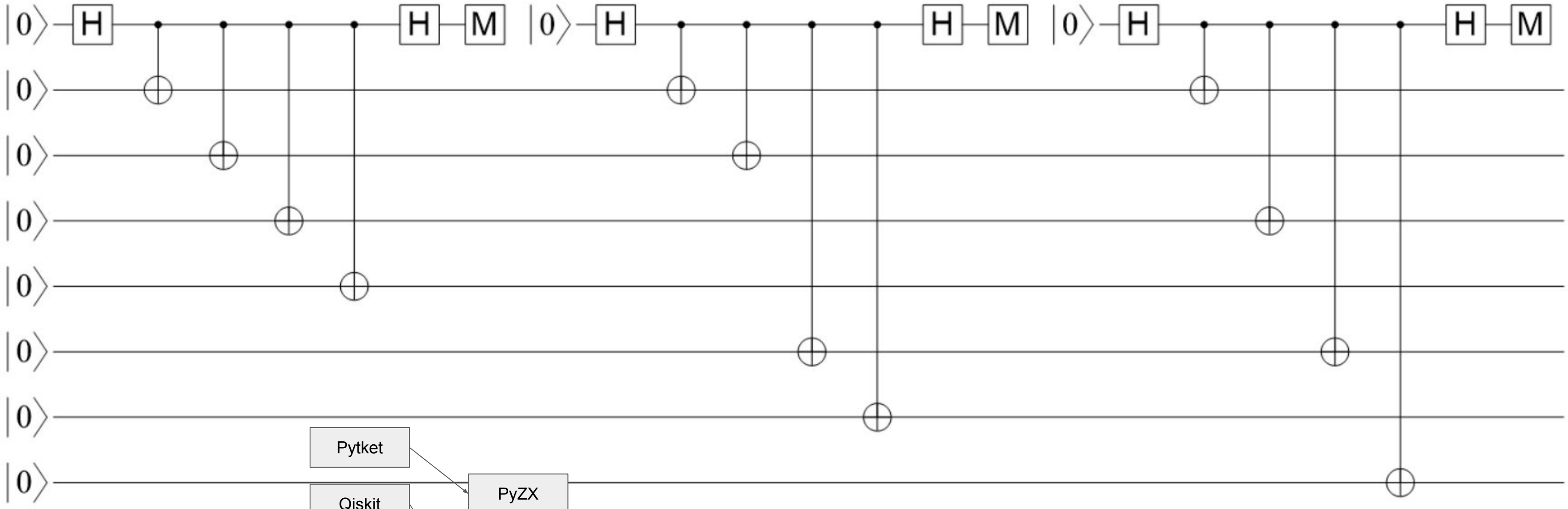
Hadamard rule: can change color of node with only H edges

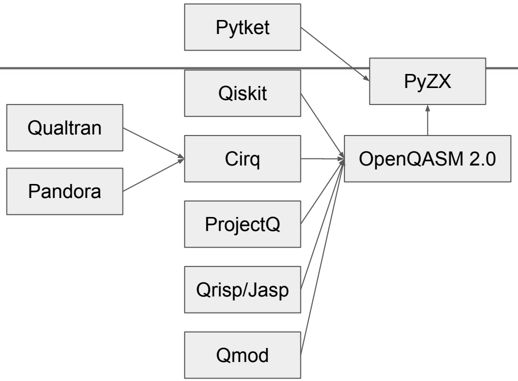


Spider rule: merge and split same color nodes

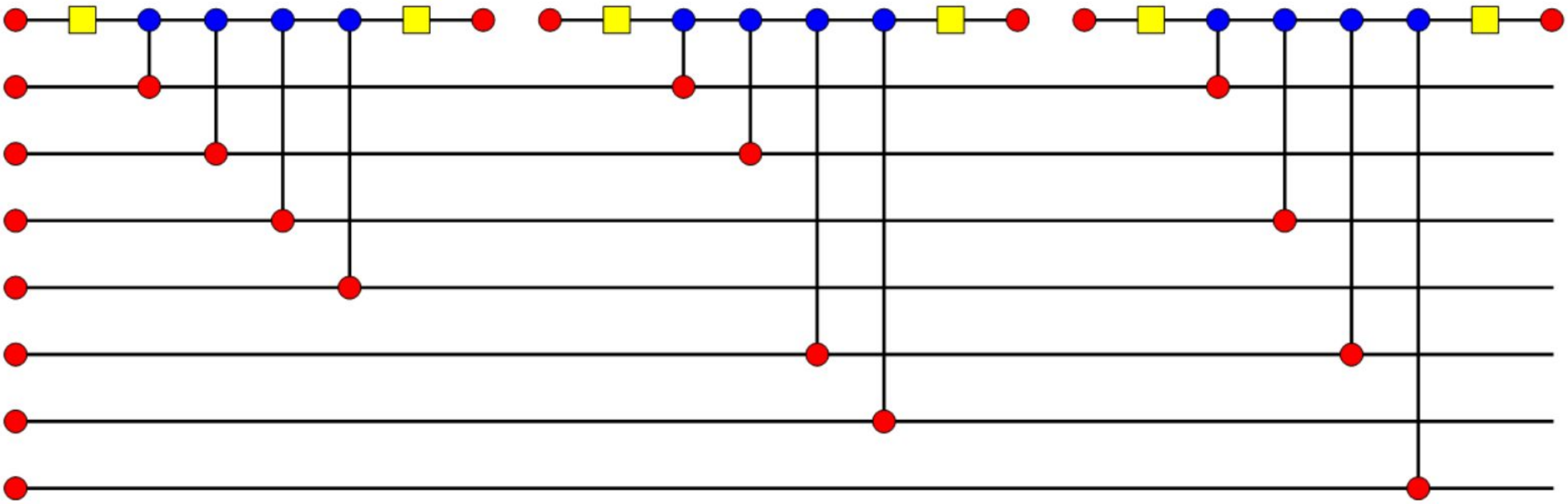
Two edge spider rule gives the identity rule.

…to convert this in your favorite tool…



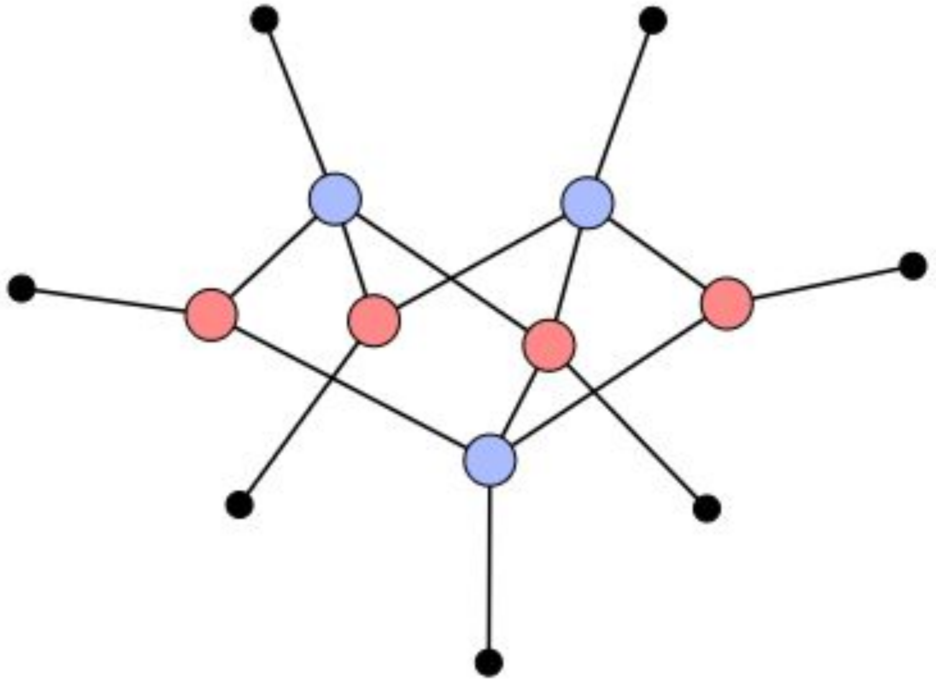


…into this in PyZX

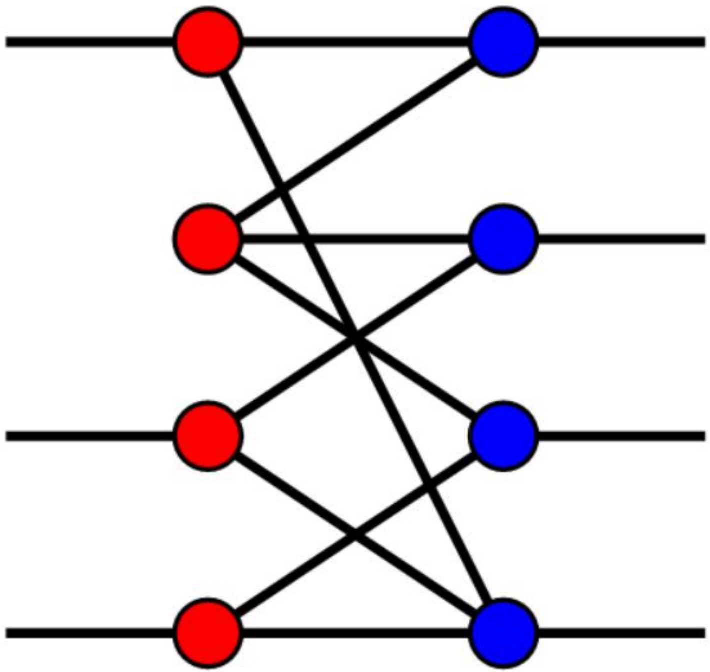


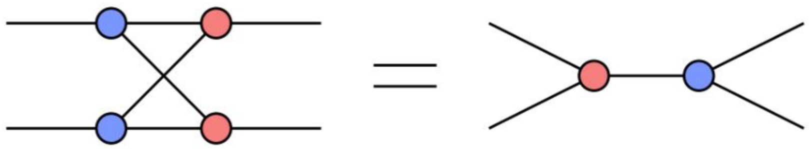
This is the canonical ZX graph version of the quantum circuit that prepares a 7 qubit Steane code.

Which PyZX can simplify to just this…



Which is equivalent* to this…

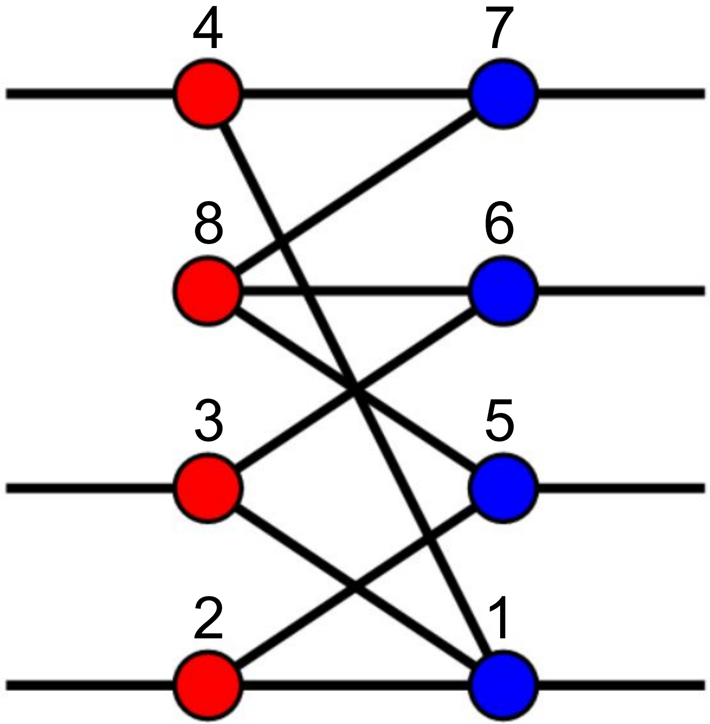




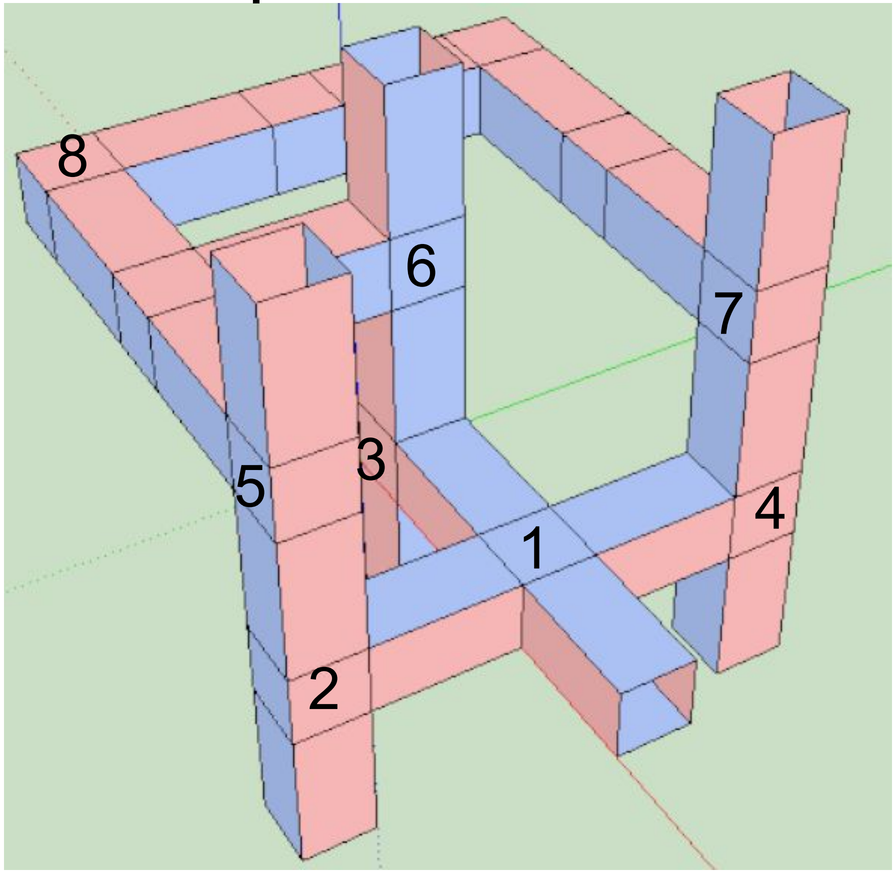
* Used the above, which is true because both structures support one red left + two reds right, and two blues left + one blue right. This is the “bialgebra-rule”.

Q: How can we turn this into lattice surgery?

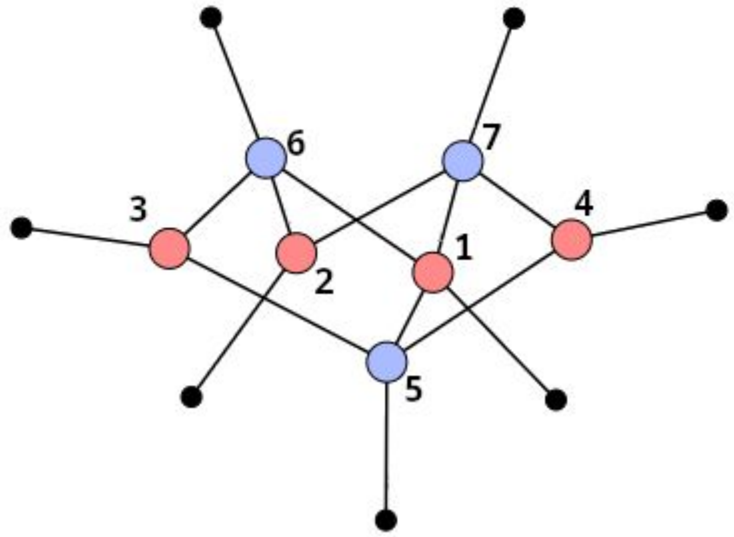
You can do it manually in SketchUp…

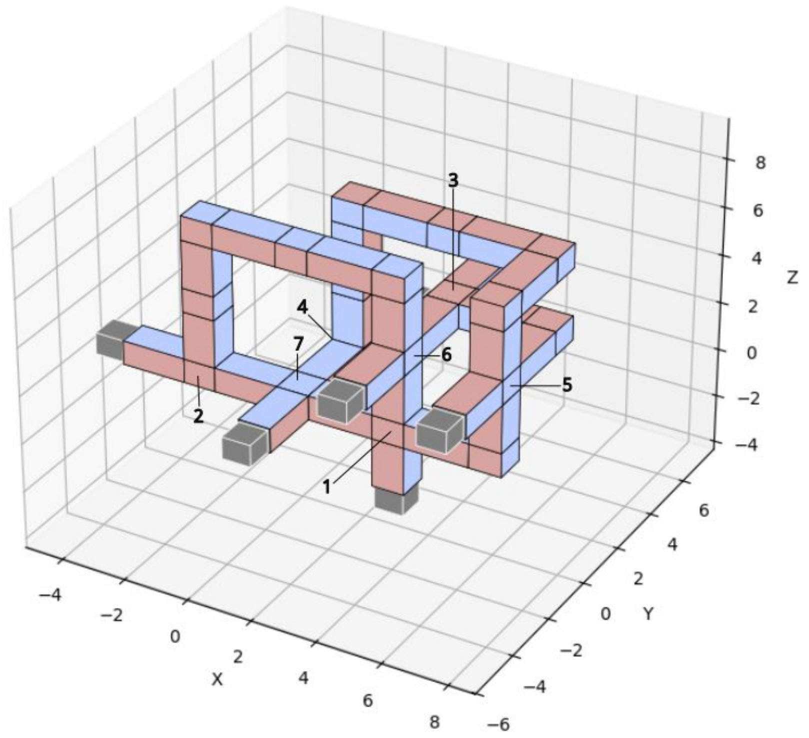


Volume = 12, best so far



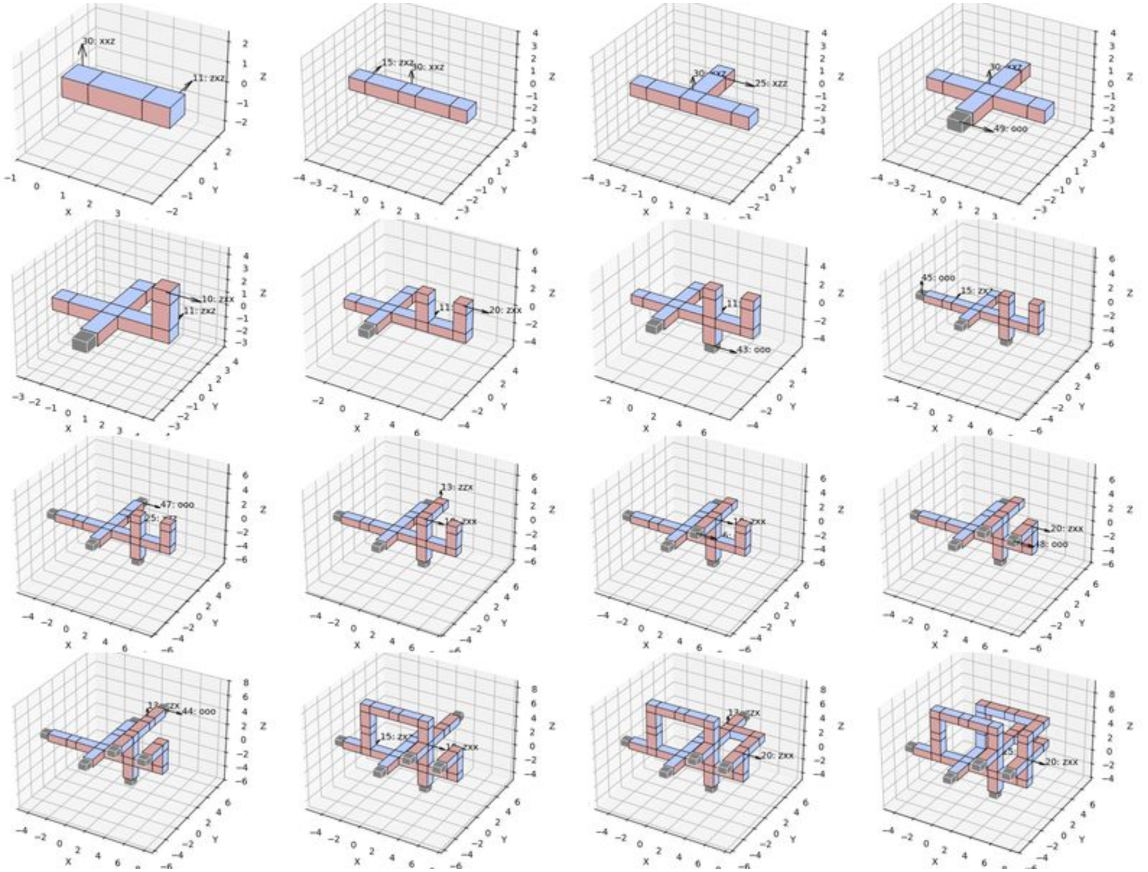
Or automatically using topologiq (Jose A Bolanos)…





Volume = 18, best so far

Topologiq step by step

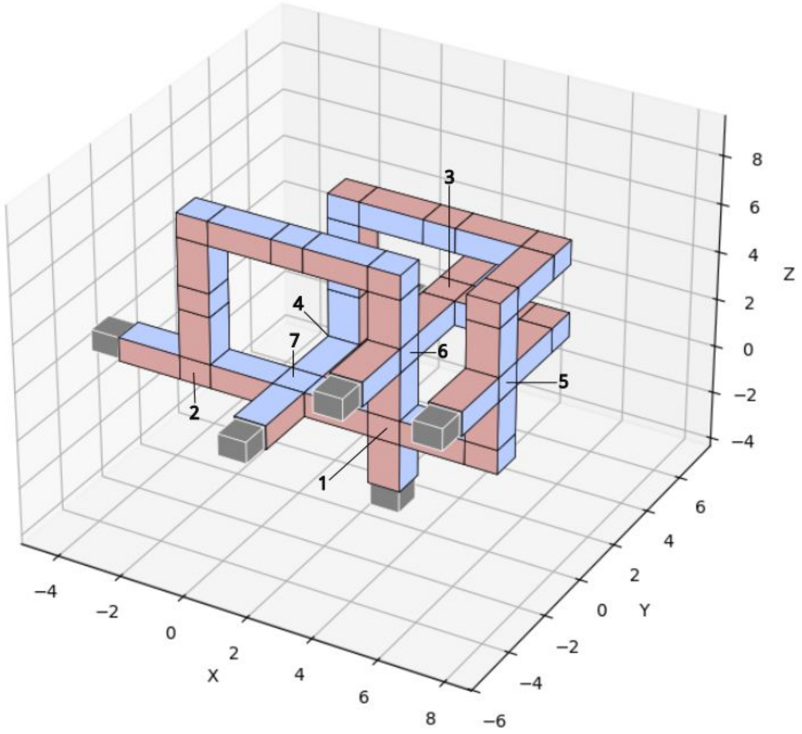


Topologiq uses an edge by edge algorithm with a randomly determined starting point.

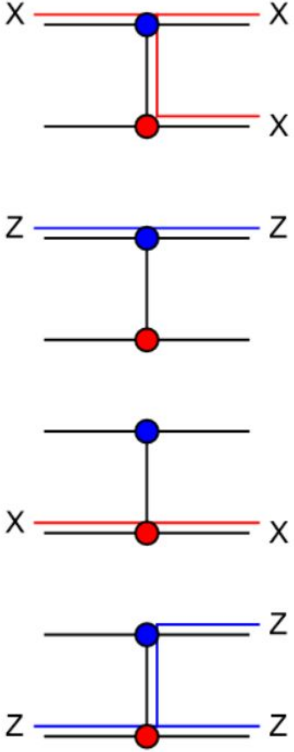
Vast scope for further research.

Now what?

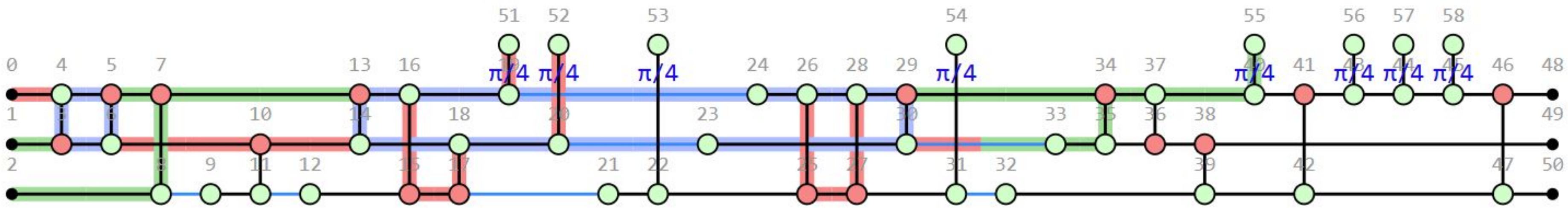
We have a structure…



But just as with CNOT we also need correlation surfaces…



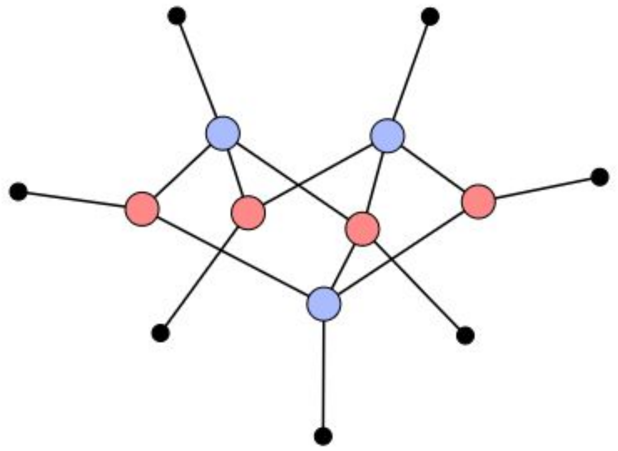
Revisiting PyZX



Back when our computation was in PyZX we actually had correlation surfaces, future work to use them.

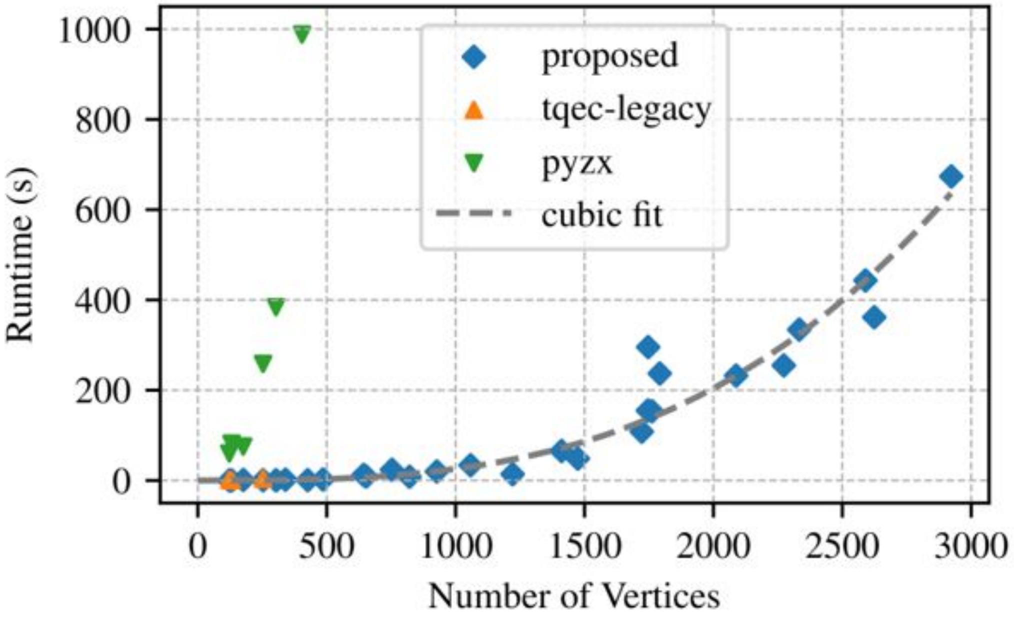
Algorithm: Tianyi Hao

CS dept University of Wisconsin-Madison



1. Start from any leaf
2. Explore the graph node-by-node
    2.1. Keep a generator set of correlation surfaces for the subgraph explored
3. Generate valid correlation surfaces given the newly explored node
4. Prune redundant ones to keep the generator set minimal
5. Repeat from 2, **overall complexity $O(n^4)$**

Benchmarks



Setup

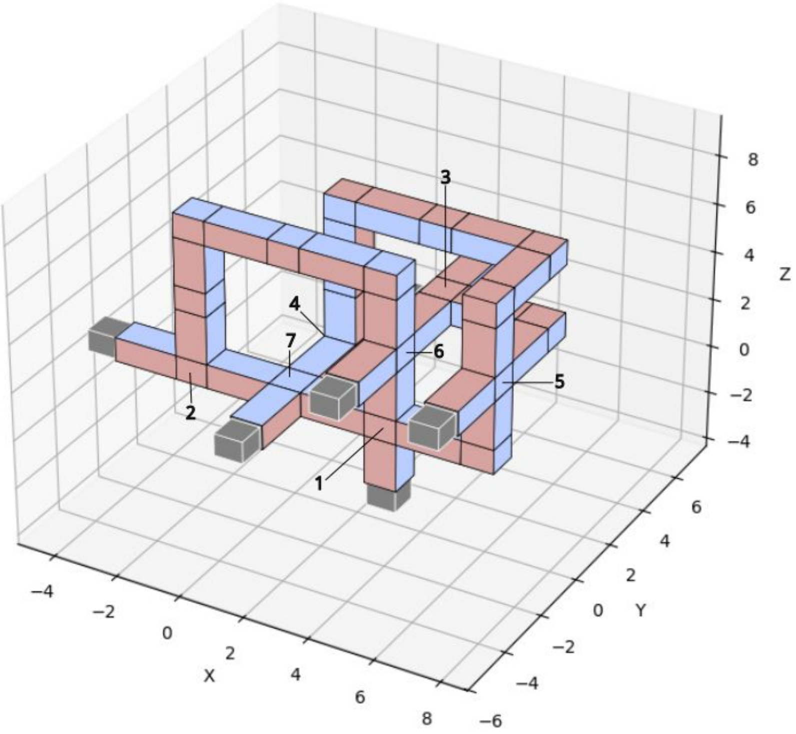
- Circuits from benchpress and Hamiltonian evolutions from hamlib, compiled to Clifford+T
- All three algorithms are written with pure Python objects, running on a single CPU

Caveats

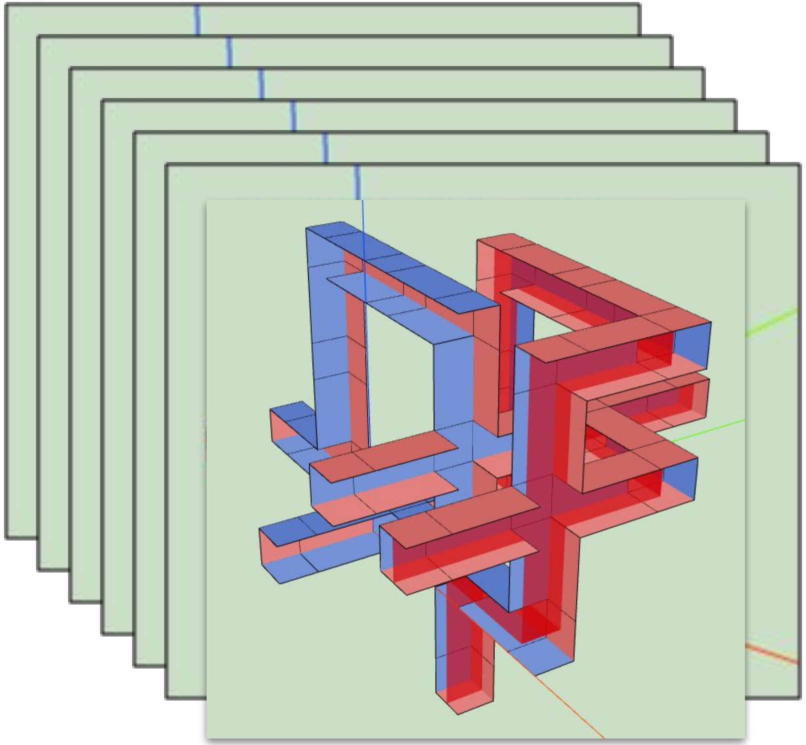
- ZX graphs are not optimized by PyZX
- Clifford proxy of non-Clifford
- T gates are not injected
- S gates are not placed at the boundary

Finding correlation surfaces

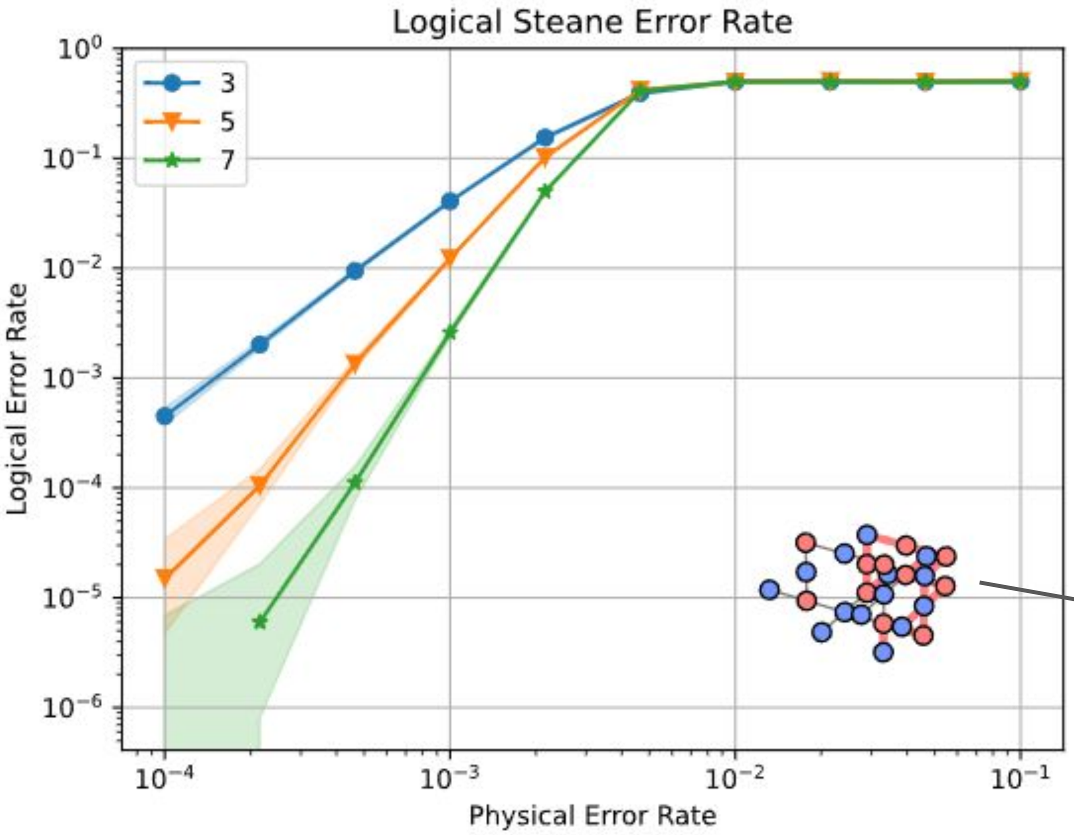
We have a structure…

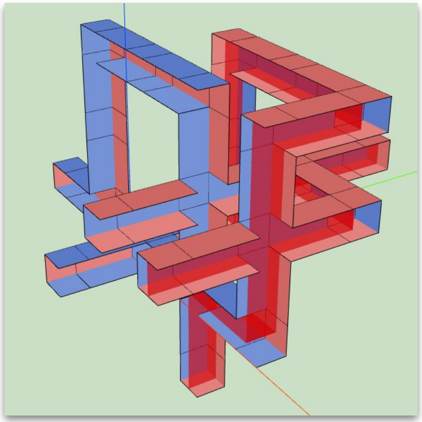


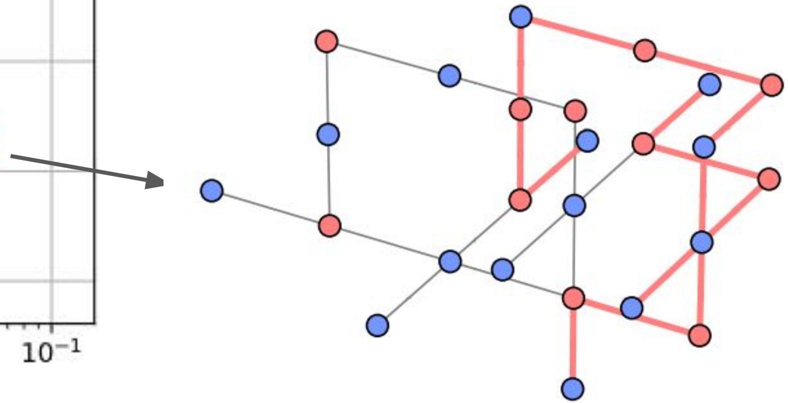
Get a set of correlation surfaces
(observables in stim)



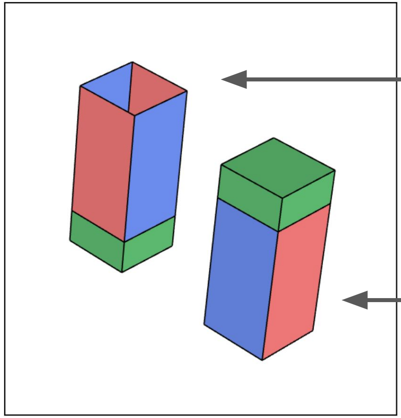
Use stim to simulate or drive quantum computer





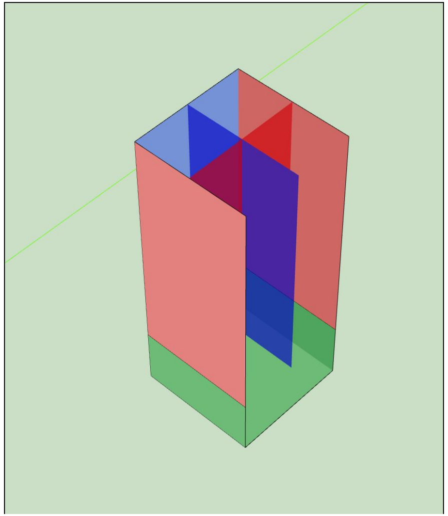


Oct 15: Y-basis Initialization/Measurement



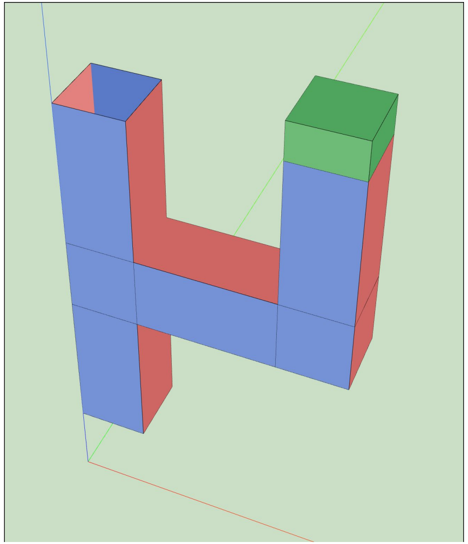
Prepare $\frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$

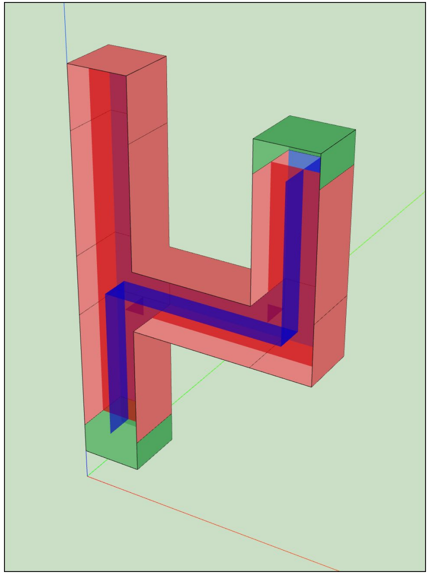
Measure in the Y basis

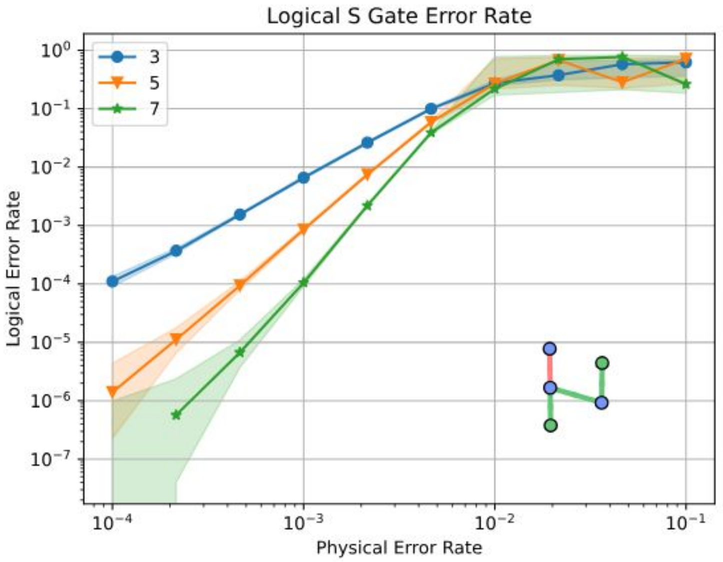


Yiming Zhang, Kabir Dubey: now have special YHalfCube that must be aligned with the temporal direction. YHalfCube supports a Y (implemented as X+Z) basis correlation surface.

Oct 15: simulation of logical S

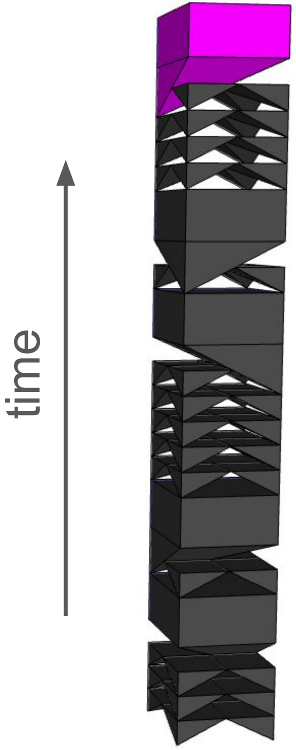






YHalfCube is a carefully designed circuit, modified from Gidney's circuit to enable it to connect seamlessly with other tqec blocks.

Future work: achieving universal quantum computation



$$|T\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\pi/4}|1\rangle\right)$$

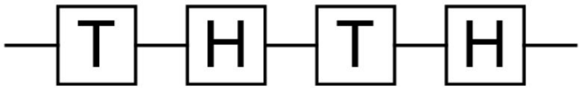
Magic state cultivation: growing T states as cheap as CNOT gates
Craig Gidney, Cody Jones, Noah Shutty
arXiv:2409.17595

Can get $O(p^5)$ suppression.

Likely good enough for practical purposes at Lambda ~10.

How does a T state give us a T gate?

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

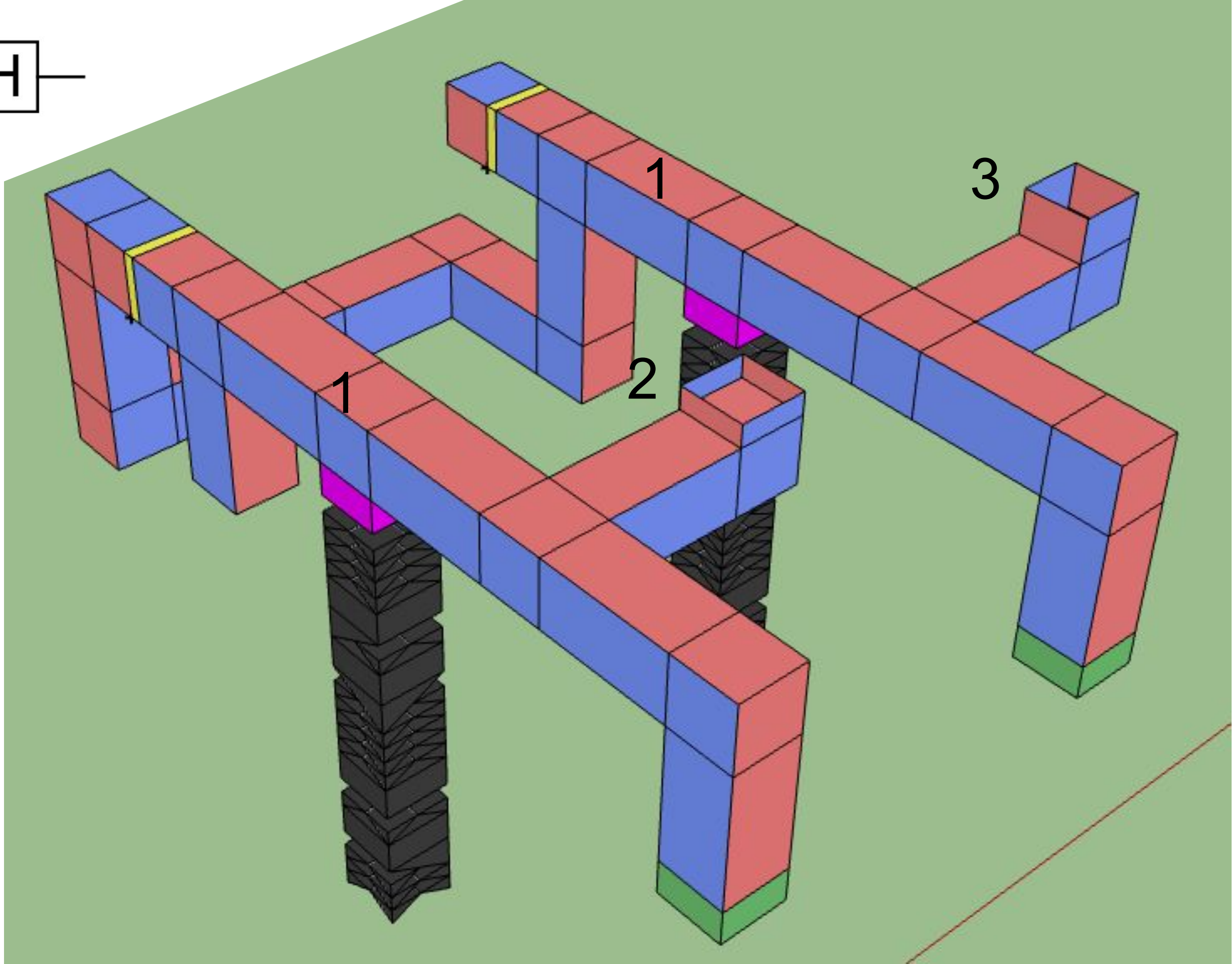


1) Execute most of the structure

2) After classical processing determine basis and measure

3) After classical processing determine basis and measure

Output format?



Summary

tqec is a group of enthusiasts working towards building an open source full stack fault-tolerant quantum compiler

There is much work to be done! Please join the community, all are welcome :-)

https://groups.google.com/g/tqec-design-automation