State of ZX 2024

Quantum Compilation

John van de Wetering
University of Amsterdam

September 24, 2024

Quantum compilation

Transforming high-level quantum computation into lower-level instructions.

Quantum compilation

Transforming high-level quantum computation into lower-level instructions.

- Convert gates in the circuit into *native* gates.

Quantum compilation

Transforming high-level quantum computation into lower-level instructions.

- Convert gates in the circuit into *native* gates.
- Routing: ensuring the connectivity of the multi-qubit gates fits in the machine topology.

Quantum compilation

Transforming high-level quantum computation into lower-level instructions.

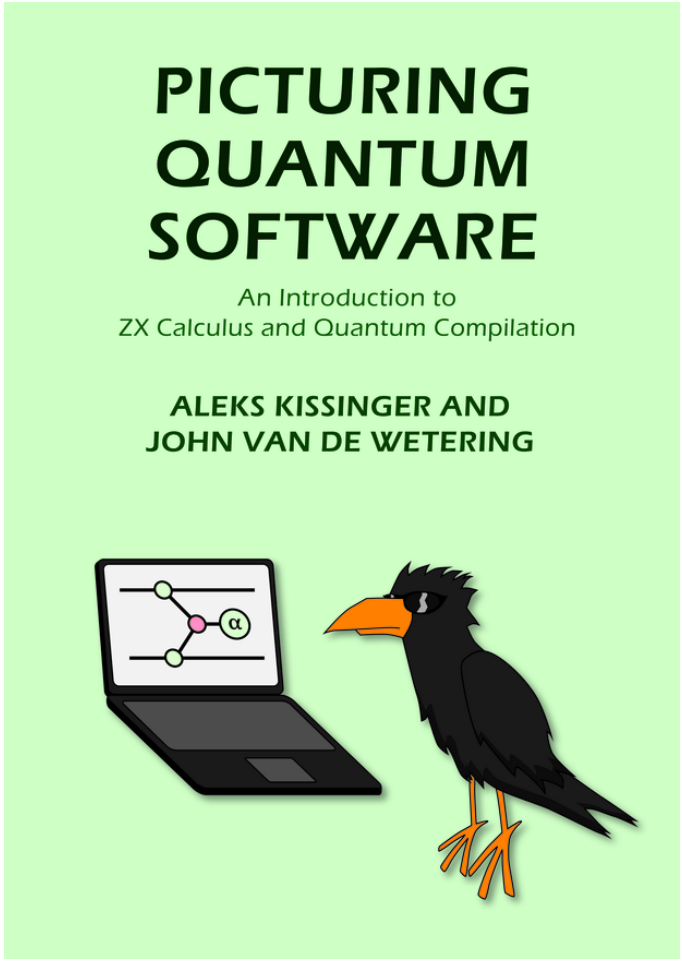
- Convert gates in the circuit into *native* gates.
- Routing: ensuring the connectivity of the multi-qubit gates fits in the machine topology.
- Optimisation: Transforming circuit into something more efficient.

Quantum compilation

Transforming high-level quantum computation into lower-level instructions.

- Convert gates in the circuit into *native* gates.
- Routing: ensuring the connectivity of the multi-qubit gates fits in the machine topology.
- Optimisation: Transforming circuit into something more efficient.
- Verification: Make sure all these steps are indeed correct.

New book on quantum compilation!



- Over 500 pages and 100 exercises!
- Synthesis of CNOT, Clifford, and Clifford+T circuits!
- Classical oracles and ZH!
- Measurement-based QC!
- Clifford+T synthesis, optimisation and catalysis!
- A new approach to understanding quantum error correction!
- And all this using ZX-diagrams!

`https://github.com/zxcalc/book`

Why ZX for compiling?

The theoretical case for compiling with ZX-diagrams as opposed to using circuits:

- Can represent more things (states, MBQC, non-unitary constructions).
- Less rigidity, gives more freedom for rewrites.
- ZX-rewrites are potentially 'more canonical': can use smaller number of different types of rewrites to accomplish same results.

Why ZX for compiling?

The theoretical case for compiling with ZX-diagrams as opposed to using circuits:

- Can represent more things (states, MBQC, non-unitary constructions).
- Less rigidity, gives more freedom for rewrites.
- ZX-rewrites are potentially 'more canonical': can use smaller number of different types of rewrites to accomplish same results.

Potential problems:

- ZX-diagrams *represent too much*.
  Need to convert them back into circuits.
- Careless rewriting loses original structure,
  might be hard to get back into an efficient regime.

History of compiling things in ZX

First compiling applications appeared in 2017:

- de Beaudrap, Horsman: *The ZX calculus is a language for surface code lattice surgery*.

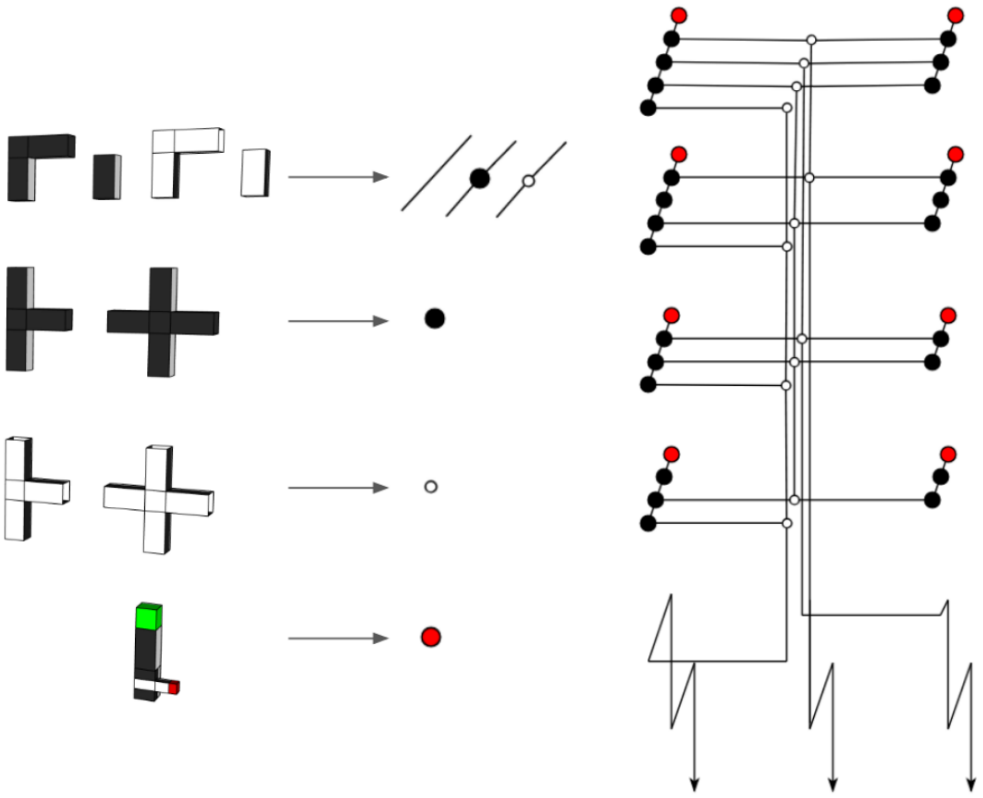
History of compiling things in ZX

First compiling applications appeared in 2017:

- de Beaudrap, Horsman: *The ZX calculus is a language for surface code lattice surgery*.
- Garvie, Duncan: *Verifying the Smallest Interesting Colour Code with Quantomatic*.

2018: Gidney's version of ZX-diagrams for surface codes

Gidney, Fowler 2018: *Efficient magic state factories with a catalyzed* $|CCZ\rangle$ *to 2*$|T\rangle$ *transformation.*



2019: The year it really kicked off

- Duncan, Kissinger, Perdrix, vdW: *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*.
- Kissinger, vdW: *Reducing T-count with the ZX-calculus*.

2019: The year it really kicked off

- Duncan, Kissinger, Perdrix, vdW: *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*.
- Kissinger, vdW: *Reducing T-count with the ZX-calculus*.
- Cowtan *et al.*: *Phase Gadget Synthesis for Shallow Circuits*.
- de Beaudrap, Xiaoning, Wang: *Techniques to Reduce $\pi/4$-Parity-Phase Circuits, Motivated by the ZX Calculus*.

2019: The year it really kicked off

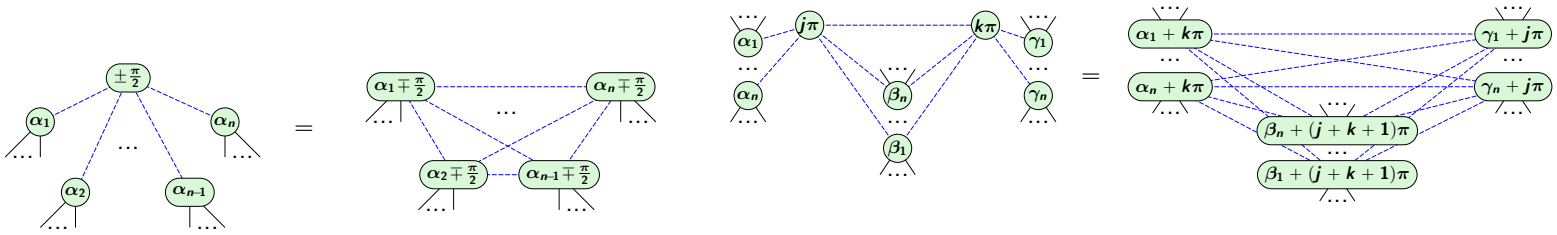
- Duncan, Kissinger, Perdrix, vdW: *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*.
- Kissinger, vdW: *Reducing T-count with the ZX-calculus*.
- Cowtan *et al.*: *Phase Gadget Synthesis for Shallow Circuits*.
- de Beaudrap, Xiaoning, Wang: *Techniques to Reduce $\pi/4$-Parity-Phase Circuits, Motivated by the ZX Calculus*.
- Hanks *et al.*: *Effective Compression of Quantum Braided Circuits Aided by ZX-Calculus*

Graph-theoretic simplification

1. Rewrite circuit to graph-like diagram.

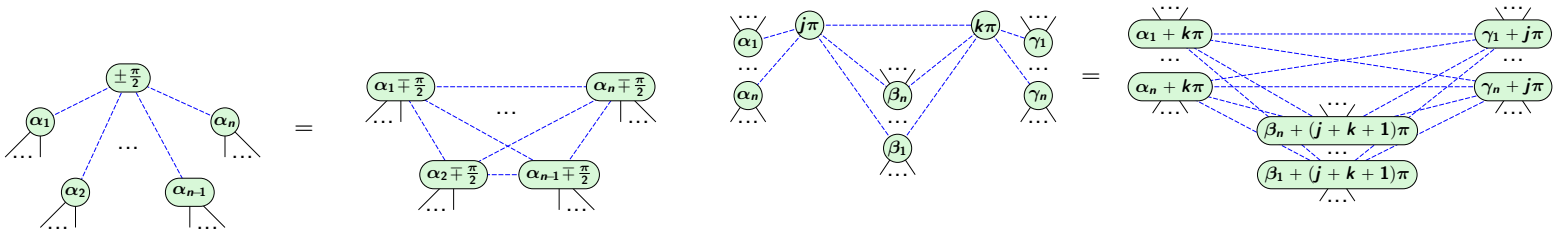
Graph-theoretic simplification

1. Rewrite circuit to graph-like diagram.
2. Remove internal Clifford spiders using local complementation and pivots:



Graph-theoretic simplification

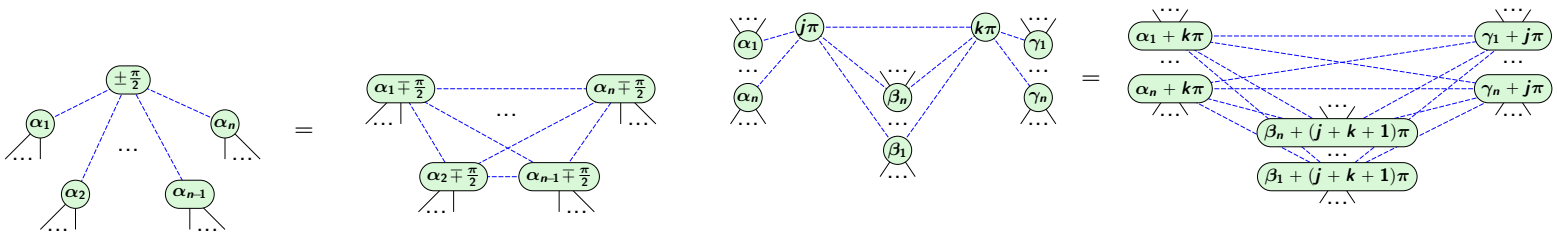
1. Rewrite circuit to graph-like diagram.
2. Remove internal Clifford spiders using local complementation and pivots:



3. *Extract* a circuit from resulting diagram.

Graph-theoretic simplification

1. Rewrite circuit to graph-like diagram.
2. Remove internal Clifford spiders using local complementation and pivots:



3. *Extract* a circuit from resulting diagram.

Variations in last 5 years:

- Pick a good ordering of rewrites in step 2.
- Try to extract more efficient circuit in step 3.

Optimising order of simplifications

- Using simulated annealing & genetic algorithms: Krueger 2021.
- Heuristic based on reducing #wires: Staudacher 2021.
- Training graph-neural-networks using reinforcement learning: Nägele & Marquardt 2023, Riu *et al.* 2023

Optimising order of simplifications

- Using simulated annealing & genetic algorithms: Krueger 2021.
- Heuristic based on reducing #wires: Staudacher 2021.
- Training graph-neural-networks using reinforcement learning: Nägele & Marquardt 2023, Riu *et al.* 2023

What is missing:

- Fundamentally new rewrites
- *Good* and *efficient* heuristics

Improving circuit extraction

- Improvements on Gaussian elimination in PyZX.
- Pushing out Pauli gadgets: Simmons 2021.
- Directly extracting multi-qubit phase gates: Staudacher 2024.
- Extracting fan-out gates for Global Gate Architectures: Villoria 2024.

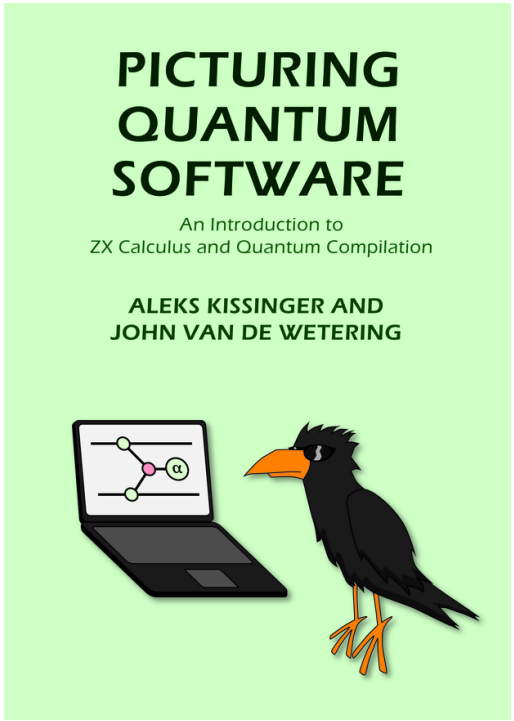
What's next?

- New approaches to circuit extraction? Involving ancillae?
- Scaling up the machine learning approaches to larger circuits?
- Benchmarking against actual state-of-the-art results
- Extracting directly to other models: MBQC, lattice surgery
- Topology-aware routing

What's next?

- New approaches to circuit extraction? Involving ancillae?
- Scaling up the machine learning approaches to larger circuits?
- Benchmarking against actual state-of-the-art results
- Extracting directly to other models: MBQC, lattice surgery
- Topology-aware routing

Thank you for your attention



https://github.com/zxcalc/book