End-to-End Quantum Circuit Optimization using ZX-Calculus

PhD. Defense

Tobias Fischbach

P. Bouvry, A. Redinger, P. Talbot, M. Rančić, M. Backens, J. van de Wetering

28.04.2026

UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

The Case for Quantum Computing

$$L = [8, 76, 50, 10, 95, 99, 17, 12, 72, 43]$$

The Case for Quantum Computing

$$L = [8, 76, 50, 10, 95, 99, 17, 12, 72, 43]$$

Is 12 an element of unsorted list L?

The Case for Quantum Computing

$$L = [8, 76, 50, 10, 95, 99, 17, 12, 72, 43]$$

Is 12 an element of unsorted list L?

Classical

- Linear search: $\mathcal{O}(N)$

The Case for Quantum Computing

$$L = [8, 76, 50, 10, 95, 99, 17, 12, 72, 43]$$

Is 12 an element of unsorted list L?

Classical

- Linear search: $\mathcal{O}(N)$

Quantum

- Grover [3]: $\mathcal{O}(\sqrt{N})$

[3] Grover (1996), "A Fast Quantum Mechanical Algorithm for Database Search"

The Case for Quantum Computing

Classical

- Linear search: $\mathcal{O}(N)$
- Factorization[1]: $\mathcal{O}(\exp\sqrt{\frac{64}{9}}\log N^{\frac{1}{3}}\log\log N^{\frac{2}{3}})$
- General Minimal Residual Alg.[2]: $\mathcal{O}(N\kappa)$
- ...

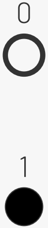
Quantum

- Grover[3]: $\mathcal{O}(\sqrt{N})$
- Shor[4]: $\mathcal{O}(\log N^3)$
- Harrow-Hassidim-Lloyd[5]: $\mathcal{O}(\log N\kappa^2)$
- ...

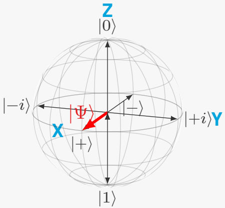
[1] Pomerance (2009), "A Tale of Two Sieves"
[2] Saad and Schultz (1986), "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems"
[3] Grover (1996), "A Fast Quantum Mechanical Algorithm for Database Search"
[4] Shor (1994), "Algorithms for Quantum Computation"
[5] Harrow, Hassidim, and Lloyd (2009), "Quantum Algorithm for Linear Systems of Equations"

Fundamental Building Blocks of Quantum Computing: Qubits

Classical Bit



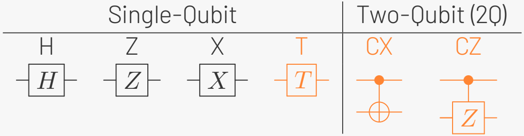
Qubit



- Basic unit of information
- *Physical Qubit*: encodes quantum information on real hardware; prone to noise and errors
- *Logical Qubit*: virtual qubit composed by multiple physical qubits; stable and reliable due to error-correction

Fundamental Building Blocks of Quantum Computing: Quantum Gates

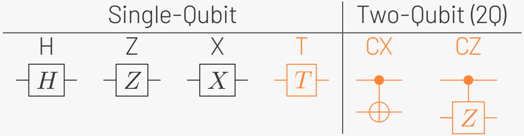
Quantum Gates

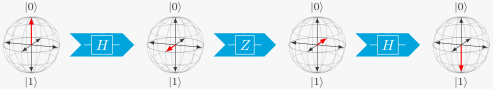


6 Itoko et al. (2020), “Optimization of quantum circuit mapping using gate transformation and commutation”

Fundamental Building Blocks of Quantum Computing: Quantum Gates

Quantum Gates



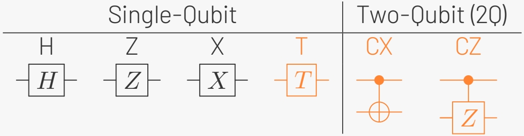


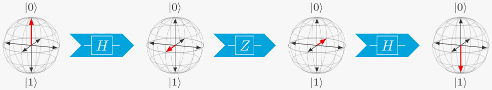
- Manipulate Qubit state

[6] Itoko et al. (2020), “Optimization of quantum circuit mapping using gate transformation and commutation”

Fundamental Building Blocks of Quantum Computing: Quantum Gates

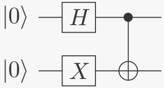
Quantum Gates





- Manipulate Qubit state

Quantum Circuit (QC)

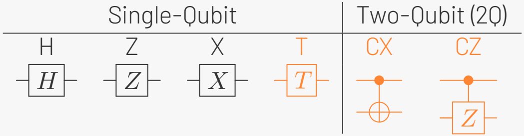


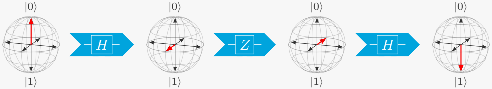
- Implements program flow

[6] Itoko et al. (2020), “Optimization of quantum circuit mapping using gate transformation and commutation”

Fundamental Building Blocks of Quantum Computing: Quantum Gates

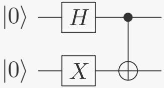
Quantum Gates





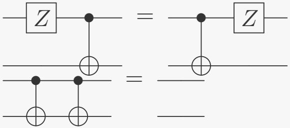
- Manipulate Qubit state

Quantum Circuit (QC)



- Implements program flow

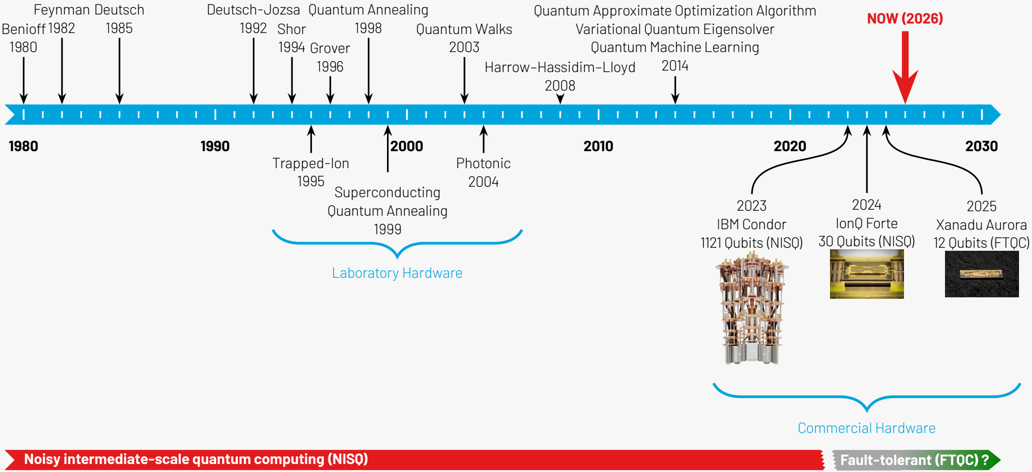
Gate Commutation & Cancellation [6]



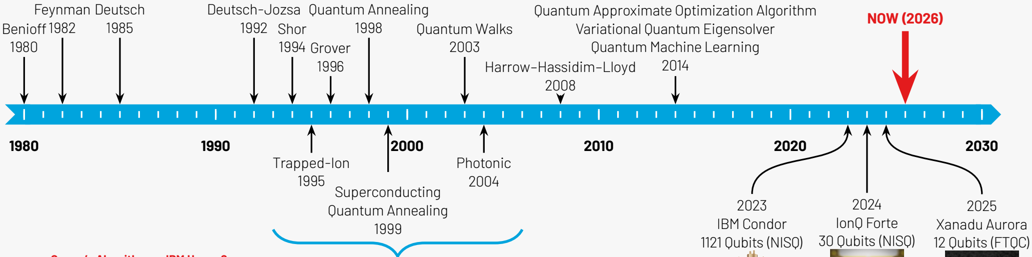
- Semantic-preserving QC transformations

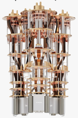
[6] Itoko et al. (2020), "Optimization of quantum circuit mapping using gate transformation and commutation"

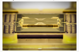
An incomplete History of Quantum Computing



An incomplete History of Quantum Computing

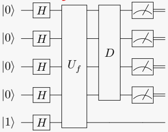






Commercial Hardware

Grover's Algorithm on IBM Heron 2



156 Physical Qubits
4 Logical Qubits + Ancilla Qubit
$2^4 = 16$ List Elements

Why are we not using quantum computer at a large-scale?
$\Rightarrow$ Resource limitations.

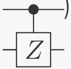
Noisy intermediate-scale quantum computing (NISQ) | **Fault-tolerant (FTQC) ?**

Noisy intermediate-scale Quantum Computing (NISQ) [7]

- *Limited by noise*
- Noise == Error
- Noise sources depend on the architecture
  - Circuit-depth

[7] Preskill (2018), "Quantum Computing in the NISQ era and beyond"

Noisy intermediate-scale Quantum Computing (NISQ) [7]

- *Limited by noise*
- Noise == Error
- Noise sources depend on the architecture
  - Circuit-depth
  - 2Q gates (e.g.,  and 

)
  - Connectivity
  - Physical qubits

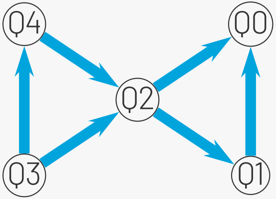
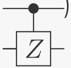


Figure: IBM QX4 coupling map [8].

[7] Preskill (2018), "Quantum Computing in the NISQ era and beyond"

[8] Zulehner, Paler, and Wille (2018), *An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures*

Noisy intermediate-scale Quantum Computing (NISQ) [7]

- *Limited by noise*
- Noise == Error
- Noise sources depend on the architecture
  - Circuit-depth
  - 2Q gates (e.g.,  and )
  - Connectivity
  - Physical qubits

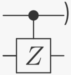
Fault tolerant Quantum Computing (FTQC) [9]

- *Limited by resources*
- Main limitations:
  - T-gate count 
  - T-gate depth 

[7] Preskill (2018), “Quantum Computing in the NISQ era and beyond”

[9] Shor (1996), “Fault-tolerant quantum computation”

Noisy intermediate-scale Quantum Computing (NISQ) [7]

- *Limited by noise*
- Noise == Error
- Noise sources depend on the architecture
  - Circuit-depth
  - 2Q gates (e.g.,  and 

)
  - Connectivity
  - Physical qubits

Fault tolerant Quantum Computing (FTQC) [9]

- *Limited by resources*
- Main limitations:
  - T-gate count $-\boxed{T}-$
  - T-gate depth $-\boxed{T}-\boxed{T}-\cdots-\boxed{T}-\boxed{T}-$
  - Error correction overhead (e.g., surface codes [10])
  - Logical qubits

[7] Preskill (2018), "Quantum Computing in the NISQ era and beyond"

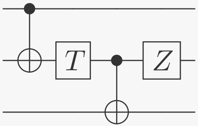
[9] Shor (1996), "Fault-tolerant quantum computation"

[10] Fowler et al. (2012), "Surface Codes"

ZX-Calculus

Quantum Circuits

- Architectures admit differing universal gate sets
- Gate commutation & cancellation depend on gate set
- Equivalence verification is computational expensive



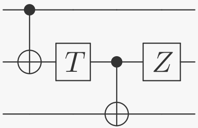
[11] Coecke and Duncan (2008), "Interacting Quantum Observables"

[12] van de Wetering (2020), *ZX-calculus for the Working Quantum Computer Scientist*

ZX-Calculus

Quantum Circuits

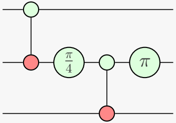
- Architectures admit differing universal gate sets
- Gate commutation & cancellation depend on gate set
- Equivalence verification is computational expensive



$\Rightarrow$

ZX-Calculus [11,12]

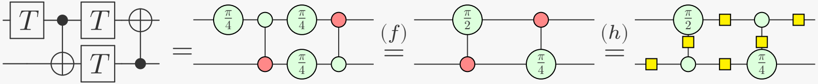
- Diagrammatic reasoning framework for QCs
- Semantic-preserving rewriting rules
- Intermediate language suitable for all gate sets



[11] Coecke and Duncan (2008), "Interacting Quantum Observables"
[12] van de Wetering (2020), *ZX-calculus for the Working Quantum Computer Scientist*

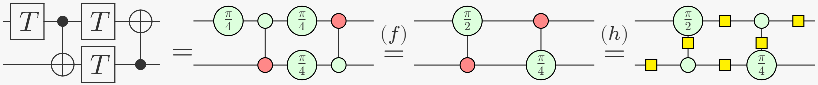
ZX-Calculus

Generators and Rewriting Rules



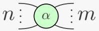
ZX-Calculus

Generators and Rewriting Rules

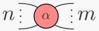


Generators

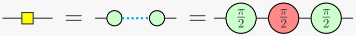
- Spiders operate in the Z- (green) or X- (red) basis



$$= |0,\ldots,0\rangle\langle 0,\ldots,0| + e^{i\alpha}|1,\ldots,1\rangle\langle 1,\ldots,1|$$



$$= |+,\ldots,+\rangle\langle +,\ldots,+| + e^{i\alpha}|-,\ldots,-\rangle\langle -,\ldots,-|$$

- Wires — $= |0\rangle\langle 0| + |1\rangle\langle 1|$
- Hadamard 

Rewriting Rules

- Semantic-preserving
- Non-terminating

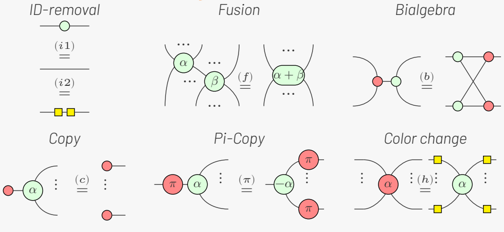
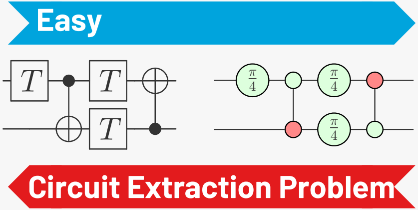


Figure: Scalar-free $\frac{\pi}{2}$ ZX-calculus.

ZX-Calculus

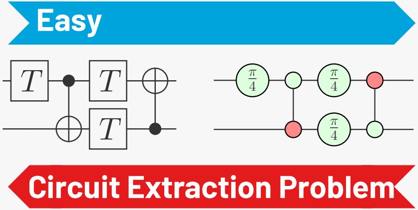
- Every QC can be expressed as a ZX-diagram
- Rewrite ZX-diagrams to optimize QCs
- Circuit extraction is #P-hard [13]
- SOTA circuit extractors increase 2Q gates and circuit-depth [14,15]



[13] Beaudrap, Kissinger, and Wetering (2022), "Circuit Extraction for ZX-Diagrams Can Be #P-Hard"
[14] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
[15] Backens et al. (2021), "There and back again: A circuit extraction tale"

ZX-Calculus

- Every QC can be expressed as a ZX-diagram
- Rewrite ZX-diagrams to optimize QCs
- Circuit extraction is #P-hard [13]
- SOTA circuit extractors increase 2Q gates and circuit-depth [14,15]



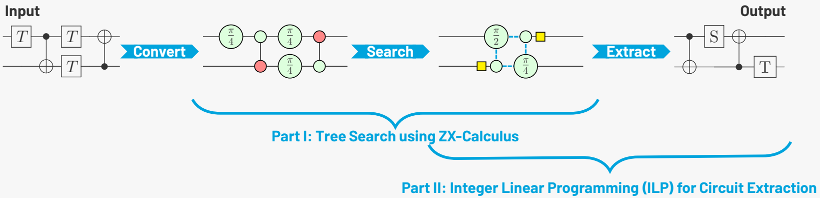
ZX-Calculus = Graph + Rewriting System

[13] Beaudrap, Kissinger, and Wetering (2022), "Circuit Extraction for ZX-Diagrams Can Be #P-Hard"
[14] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
[15] Backens et al. (2021), "There and back again: A circuit extraction tale"

My Thesis: End-to-End Quantum Circuit Optimization using ZX-Calculus

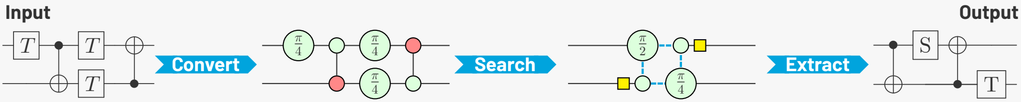
We address these issues by introducing QC optimization strategies that are:

- *general*
  - independent of a given quantum architecture
  - can be used for NISQ and FTQC devices (e.g., 2Q gates, circuit-depth, and T-gates)
- *extendable*
  - framework current and future metrics
- *end-to-end*
  - Part I: Tree search using ZX-Calculus
  - Part II: Integer Linear Programming for Circuit Extraction



ZX-Benchmark

Contribution I: Benchmarking Framework



Pruning Condition: Cflow

Algorithm	DFS		FR		IDDFS		LDS		LE	
Metric Δ %	2Q	T	2Q	T	2Q	T	2Q	T	2Q	T
Small	0.00	0.00	45.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Medium	0.00	-28.57	10.71	-33.33	0.00	-28.57	0.00	-28.57	0.00	-28.57
Large	70.26	-42.86	50.26	-42.86	64.10	-42.86	67.69	-42.86	70.26	-42.86
full	39.19	-25.60	38.60	-27.38	50.00	-27.38	48.66	-27.38	39.19	-25.60
Median	19.59	-27.08	41.80	-30.36	25.00	-27.98	24.33	-27.98	19.59	-27.08

Legend

Algorithm
LDS := Limited-discrepancy search
LE := Local-elimination
IDDFS := Iterative-deepening depth-first search
DFS := Depth-first search
FR := Full reduce

Metric
T := T-gate
2Q := Two-qubit

- Declarative end-to-end optimization pipeline
- Reproducible benchmarking framework
- Supports our tree search optimizers [16], ILP-based extractors, and 5 SOTA optimizers

GitLab

Example

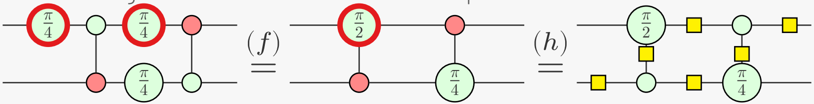
[16] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

Part I: Tree Search using ZX-Calculus

Contribution II: Connecting Combinatorial and ZX-based Optimization [17]

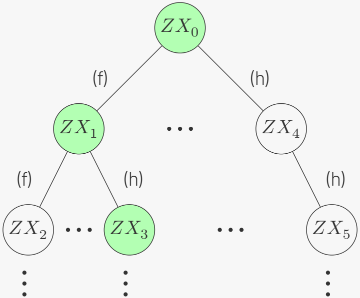


⇒ **Find finite rewriting rule sequence that minimizes a metric**

- Rewriting rules span search tree
- $\approx 72.6M$ nodes in search tree spanned by 6 rules with depth 10

Challenges

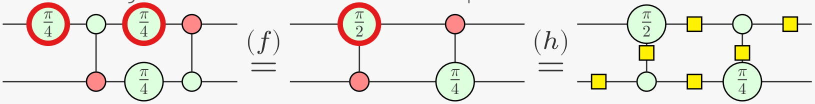
- Non-terminating → pruning conditions & search strategy
- High-memory requirements → rule bundling
- Failed circuit extraction → ensure extractability



[17] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

Contribution II: Connecting Combinatorial and ZX-based Optimization [17]

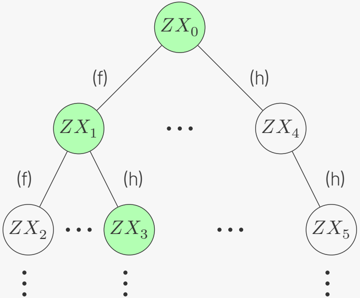


⇒ Find finite rewriting rule sequence that minimizes a metric

- Rewriting rules span search tree
- $\approx 72.6M$ nodes in search tree spanned by 6 rules with depth 10

Challenges

- Non-terminating → pruning conditions & search strategy
- High-memory requirements → rule bundling
- Failed circuit extraction → ensure extractability

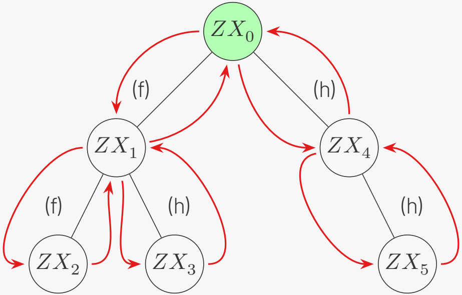


[17] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

Challenges in Tree Search [19]

- Pruning conditions:
  - No colour cycle
  - No spider unfusion
  - Preserve causal flow
- Exploration order determined by existing search algorithms [18] e.g.:
  - Depth-first search (DFS)



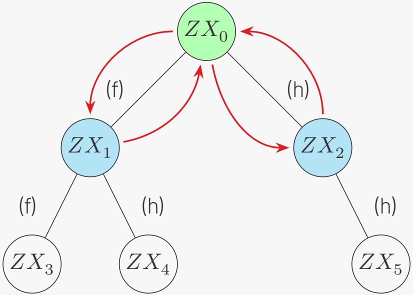
[18] Russell and Norvig (2021), "Search Algorithms"

[19] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

Challenges in Tree Search [19]

- Pruning conditions:
  - No colour cycle
  - No spider unfusion
  - Preserve causal flow
- Exploration order determined by existing search algorithms [18] e.g.:
  - Depth-first search (DFS)
  - Iterative-deepening depth-first search (IDDFS)



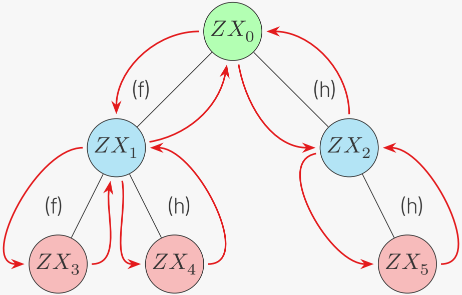
[18] Russell and Norvig (2021), "Search Algorithms"

[19] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

Challenges in Tree Search [19]

- Pruning conditions:
  - No colour cycle
  - No spider unfusion
  - Preserve causal flow
- Exploration order determined by existing search algorithms [18] e.g.:
  - Depth-first search (DFS)
  - Iterative-deepening depth-first search (IDDFS)



[18] Russell and Norvig (2021), "Search Algorithms"

[19] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus

Background [23]

- Most optimizers focus on ad-hoc or heuristic strategies
- Kissinger et al. contribute the ad-hoc T-gate optimizer "Full Reduce" (FR) [20]
- Holker introduces an edge heuristic to optimize 2Q count [21]
- Mattick at al. combine tree search and reinforcement learning (RL) to reduce edge and 2Q count [22]

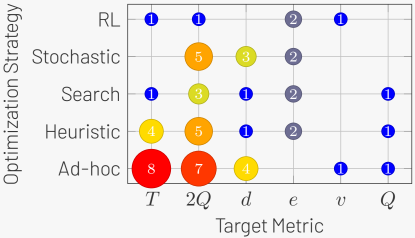


Figure: Target metric vs. optimization strategy.

---

[20] Kissinger and van de Wetering (2020), "Reducing T-count with the ZX-calculus"

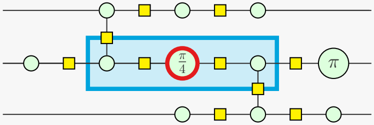
[21] Holker (2024), *Causal Flow Preserving Optimisation of Quantum Circuits in the ZX-calculus*

[22] Mattick et al. (2025), *Optimizing Quantum Circuits via ZX Diagrams using Reinforcement Learning and Graph Neural Networks*

[23] **Tobias M. Fischbach**, Talbot, and Bouvry (2025), "A Review on Quantum Circuit Optimization using ZX-Calculus" (preprint)

Contribution III: Local Elimination [24]

- Search tree grows with ZX-diagram size
- Rewriting rules are local transformations
- Select a small subdiagram around a target spider (e.g., T-spider)
- Apply tree search on smaller subdiagram
- Grow subdiagram if no improvement is possible



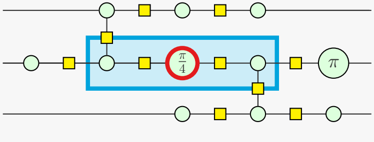
(a) $n = 1$

[24] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

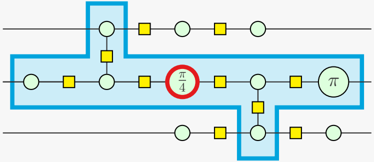
Part I: Tree Search using ZX-Calculus

Contribution III: Local Elimination [24]

- Search tree grows with ZX-diagram size
- Rewriting rules are local transformations
- Select a small subdiagram around a target spider (e.g., T-spider)
- Apply tree search on smaller subdiagram
- Grow subdiagram if no improvement is possible



(a) $n = 1$



(b) $n = 2$

[24] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus [25]

T-gate Optimization (FTQC)

- Global timeout of 1.5 hours
- Connectivity change takes precedent over spider count
- Baseline is Full Reduce (FR)
- LDS and IDDFS are within $\approx 2.4\%$ of FR
- LE and DFS outperform FR by $\approx 22\%$ w.r.t. 2Q and stay within $\approx 3.2\%$ w.r.t. T-gate count
- Worse performance is price for generality

Algorithm	FR		LDS		LE		IDDFS		DFS	
Metric $\Delta\%$	T	2Q	T	2Q	T	2Q	T	2Q	T	2Q
Small	**0.0**	45.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Medium	**-33.33**	10.71	-28.57	**0.0**	-28.57	**0.0**	-28.57	**0.0**	-28.57	**0.0**
Large	**-42.86**	**50.26**	**-42.86**	67.69	**-42.86**	70.26	**-42.86**	64.1	**-42.86**	70.26
Tpar	**-27.38**	**38.6**	**-27.38**	48.66	-25.6	39.19	**-27.38**	50.0	-25.6	39.19
Median	**-30.36**	41.8	-27.98	24.33	-27.08	**19.59**	-27.98	25.0	-27.08	**19.59**

[25] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part I: Tree Search using ZX-Calculus [26]

Edge Optimization (NISQ)

- Global timeout of 1.5 hours
- Connectivity change takes precedent over spider count
- Baseline Holker
- DFS ($\approx -14\%$), IDDFS ($\approx -17\%$), and LDS ($\approx -18\%$) outperform Holker w.r.t. edge count
- BUT increase the 2Q gate count DFS ($\approx 25\%$), IDDFS ($\approx 31\%$), and LDS ($\approx 31\%$)

Algorithm	Holker		LDS		IDDFS		DFS	
Metric $\Delta\%$	e	2Q	e	2Q	e	2Q	e	2Q
Small	-18.82	**0.0**	**-43.06**	**0.0**	**-43.06**	**0.0**	**-43.06**	**0.0**
Medium	-27.52	**-30.95**	**-42.64**	8.33	**-42.64**	8.33	-33.33	13.1
Large	-12.11	**-12.31**	**-26.55**	56.41	-25.93	53.33	-20.96	75.38
Tpar	-8.79	**-3.3**	**-16.96**	30.6	**-16.96**	31.67	-13.22	25.96
Clifford	**-18.8**	**0.0**	-16.49	104.76	-16.49	104.76	-14.16	87.5
Clifford+T	**-18.33**	**3.45**	**-18.33**	89.47	-15.83	89.47	-14.41	65.22
CX	**-18.02**	**0.0**	-16.54	103.7	-16.54	103.7	-14.29	89.47
Median	**-18.33**	**0.0**	**-18.33**	56.41	-16.96	53.33	-14.41	65.22

[26] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

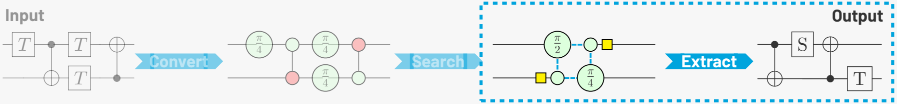
Part I: Tree Search using ZX-Calculus [27]

Take-Away Message

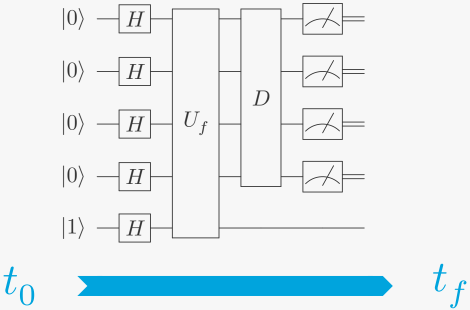
- Tree search is a *general* method that:
  - is suitable for current and future metrics
  - competes with optimizers for dedicated metrics
    - within $\approx 2.4\%$ w.r.t. T-gate count of the dedicated T-gate optimizer FR
    - improves the edge count up to $\approx 18\%$ compared to Holker optimizer
  - eventually explores the same rewriting sequence as dedicated optimizers

[27] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), "Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus" (published)

Part II: Integer Linear Programming for Circuit Extraction

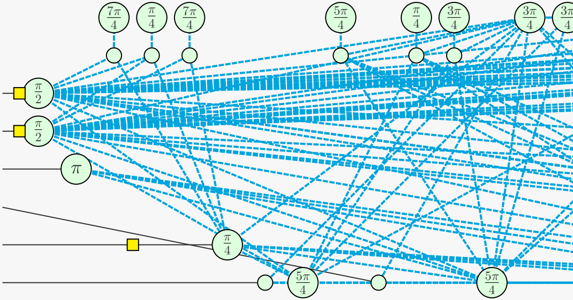


Quantum Circuit



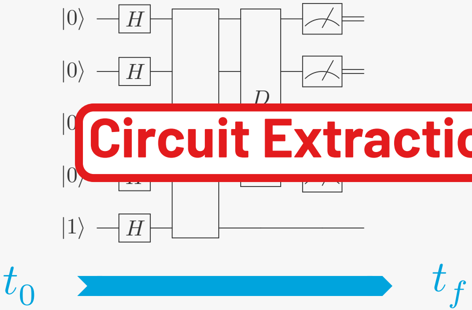
- Implicit time notation
- Clear gate execution order

ZX-Diagram

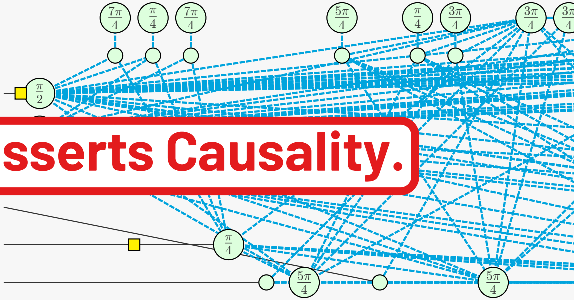


- Undirected edges
- Spider input & output are interchangeable

Quantum Circuit



ZX-Diagram



Circuit Extraction asserts Causality.

- Implicit time notation
- Clear gate execution order

- Undirected edges
- Spider input & output are interchangeable

Part II: Integer Linear Programming for Circuit Extraction

Background

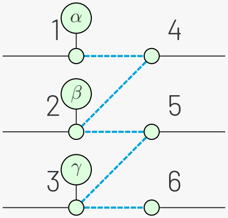
- Circuit extraction is #P-hard [28]
- Upper bound of optimized circuit extraction is $\mathrm{NP}^{\mathrm{NP}^{\#\mathrm{P}}}$ [29]
- Polynomial-time algorithms exists for ZX-diagrams that exhibit certain graph-theoretic properties [30,31]
- Modified algorithms for architecture-aware extraction exist [32,33]

[28] Beaudrap, Kissinger, and Wetering (2022), "Circuit Extraction for ZX-Diagrams Can Be #P-Hard"
[29] Mitosek (2024), "Constructing $\mathrm{NP}^{\#\mathrm{P}}$-complete problems and $\#\mathrm{P}$-hardness of circuit extraction in phase-free ZH"
[30] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
[31] Backens et al. (2021), "There and back again: A circuit extraction tale"
[32] Kissinger and Griend (2019), *CNOT circuit extraction for topologically-constrained quantum memories*
[33] Villoria, Basold, and Laarman (2026), "Optimisation and synthesis of quantum circuits with global gates"

UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

Part II: Integer Linear Programming for Circuit Extraction

- ZX-diagram is converted in MBQC-form



[34] Danos and Kashefi (2006), "Determinism in the one-way model"

[35] Browne et al. (2007), "Generalized flow and determinism in measurement-based quantum computation"

[36] Simmons (2021), "Relating Measurement Patterns to Circuits via Pauli Flow"

[37] Kissinger and Wetering (2026), *ZX-Flow: A Flexible Criterion for Deterministic Computation with ZX-Diagrams*

[38] Wei (2021), "Measurement-Based Quantum Computation"

- ZX-diagram is converted in MBQC-form

Definition

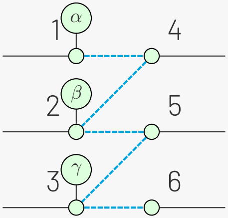
MBQC / Graph State Form [39]

- All spiders are Z-spiders
- Only Hadamard edges between spiders
- Inputs and Outputs are connected to a Z-spider
- Every Z-spiders connect maximally to one Input or Output
- No self-loops or parallel edges
- Each spider has a single incident output wire

[34] Danos and Kashefi (2006), “Determinism in the one-way model”
[35] Browne et al. (2007), “Generalized flow and determinism in measurement-based quantum computation”
[36] Simmons (2021), “Relating Measurement Patterns to Circuits via Pauli Flow”
[37] Kissinger and Wetering (2026), *ZX-Flow: A Flexible Criterion for Deterministic Computation with ZX-Diagrams*
[38] Wei (2021), “Measurement-Based Quantum Computation”
[39] Backens et al. (2021), “There and back again: A circuit extraction tale”

Part II: Integer Linear Programming for Circuit Extraction

- ZX-diagram is converted in MBQC-form



[34] Danos and Kashefi (2006), "Determinism in the one-way model"
[35] Browne et al. (2007), "Generalized flow and determinism in measurement-based quantum computation"
[36] Simmons (2021), "Relating Measurement Patterns to Circuits via Pauli Flow"
[37] Kissinger and Wetering (2026), *ZX-Flow: A Flexible Criterion for Deterministic Computation with ZX-Diagrams*
[38] Wei (2021), "Measurement-Based Quantum Computation"

Part II: Integer Linear Programming for Circuit Extraction

- ZX-diagram is converted in MBQC-form
- It exhibits either causal, general, Pauli or ZX flow [34], [35], [36], [37]
- Strict partial order that corresponds to a deterministic measurement-pattern [38]

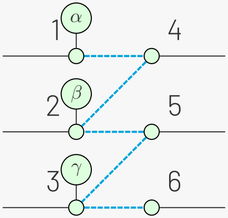
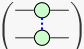


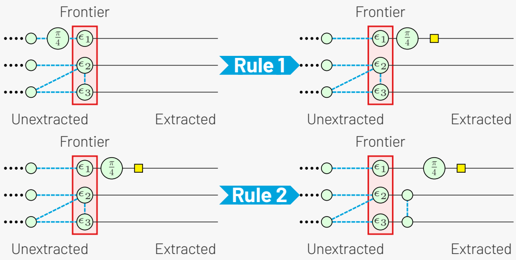
Figure: Causal flow with measurement order $1 \prec 2 \prec 3$

1. Measure $v = 1$, correct $g(1) = 4$: $1 \prec 4$
2. Measure $v = 2$, correct $g(2) = 5$: $2 \prec 5$
3. Measure $v = 3$, correct $g(1) = 4$: $3 \prec 6$

[34] Danos and Kashefi (2006), "Determinism in the one-way model"
[35] Browne et al. (2007), "Generalized flow and determinism in measurement-based quantum computation"
[36] Simmons (2021), "Relating Measurement Patterns to Circuits via Pauli Flow"
[37] Kissinger and Wetering (2026), *ZX-Flow: A Flexible Criterion for Deterministic Computation with ZX-Diagrams*
[38] Wei (2021), "Measurement-Based Quantum Computation"

Default Circuit Extraction Algorithm [40,41]

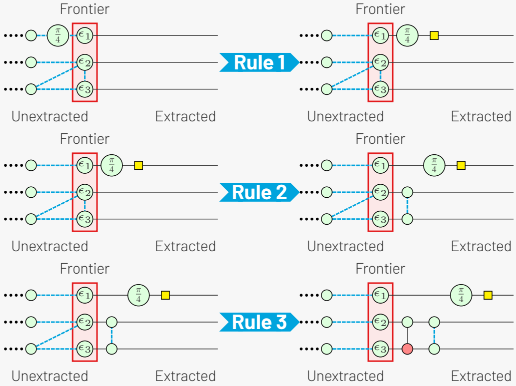
- ZX-diagram is in MBQC-form and has flow
- ZX-diagram is separated in an extracted and unextracted part
- Separated by frontier spiders
- Iteratively applies one out of three rules:
  1. Single qubit gate (e.g., $-\!\!\left(\frac{\pi}{4}\right)\!\!-$)
  2. CZ gate ()



40 Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
41 Backens et al. (2021), "There and back again: A circuit extraction tale"

Default Circuit Extraction Algorithm [40,41]

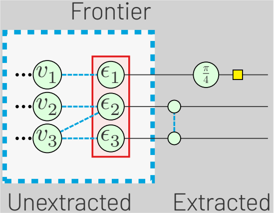
- ZX-diagram is in MBQC-form and has flow
- ZX-diagram is separated in an extracted and unextracted part
- Separated by frontier spiders
- Iteratively applies one out of three rules:
  1. Single qubit gate (e.g., $-\left(\frac{\pi}{4}\right)-$)
  2. CZ gate
  3. CNOT gate



[40] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
[41] Backens et al. (2021), "There and back again: A circuit extraction tale"

CNOT Extraction

- Frontier spiders with multiple wires are extracted by CNOTs [42]
- Adjacency matrix between Frontier and nearest neighbor spiders
- Row addition correspond to the addition of a CNOT gate



$$\mathbf{A} = \begin{array}{c} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

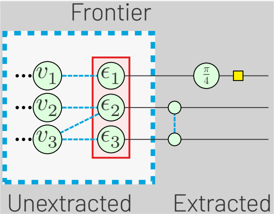
[42] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"

[43] Strassen (1969), "Gaussian elimination is not optimal"

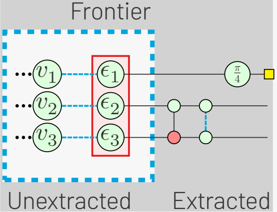
UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

CNOT Extraction

- Frontier spiders with multiple wires are extracted by CNOTs [42]
- Adjacency matrix between Frontier and nearest neighbor spiders
- Row addition correspond to the addition of a CNOT gate
- $\epsilon_2 := \epsilon_2 \oplus \epsilon_3$
- Adjacency Matrix solved by Gaussian Elimination



$$\mathbf{A} = \begin{array}{c} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$



$$\mathbf{A} = \begin{array}{c} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

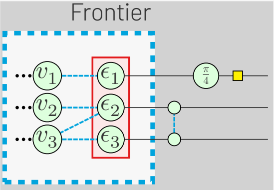
[42] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"

[43] Strassen (1969), "Gaussian elimination is not optimal"

UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

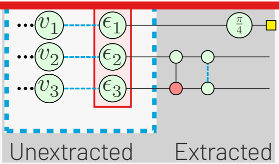
CNOT Extraction

- Frontier spiders with multiple wires are extracted by CNOTs [42]
- Adjacency matrix between Frontier and nearest neighbor spiders
- Rc ac
- $\epsilon_2 := \epsilon_2 \oplus \epsilon_3$
- Adjacency Matrix solved by Gaussian Elimination



$$\mathbf{A} = \begin{array}{c} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

Gaussian Elimination is not optimal! [43]



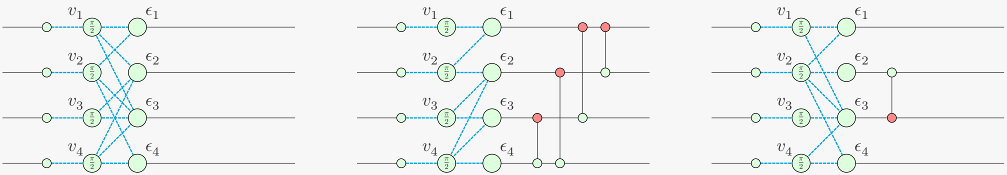
$$\mathbf{A} = \begin{array}{c} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

[42] Duncan et al. (2020), "Graph-Theoretic Simplification of Quantum Circuits with the ZX-calculus"
[43] Strassen (1969), "Gaussian elimination is not optimal"

UNIVERSITY OF LUXEMBOURG
Institute for Advanced Studies

Part II: Integer Linear Programming for Circuit Extraction

Non-Optimality of Gaussian Elimination - Example



(a) Unextracted **(b)** Gaussian Elimination **(c)** ILP

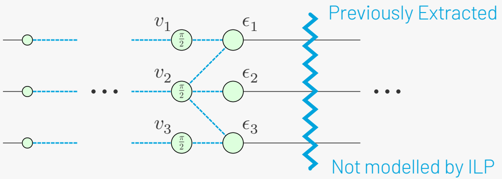
- Gaussian Elimination adds 4 CNOTs
- ILP adds 1 CNOT
- Minimizing row additions is NP-hard [44]

[44] Karp (2009), "Reducibility among combinatorial problems"

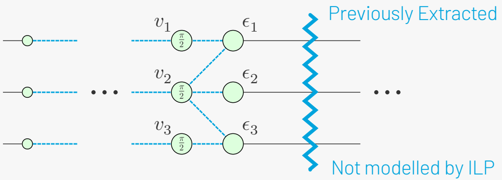
Part II: Integer Linear Programming for Circuit Extraction

Contribution IV: Extract a single Frontier using ILP [45]

- ILP that minimizes row additions to extract a single Frontier
- Extraction of a single Frontier is a local solution
- Local solutions are not unique



(a) Solution 1



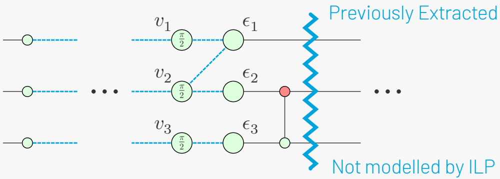
(b) Solution 2

[45] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

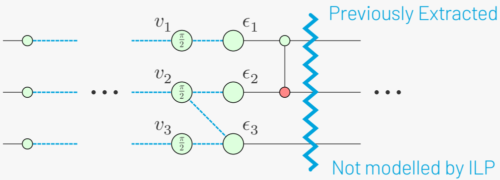
Part II: Integer Linear Programming for Circuit Extraction

Contribution IV: Extract a single Frontier using ILP [45]

- ILP that minimizes row additions to extract a single Frontier
- Extraction of a single Frontier is a local solution
- Local solutions are not unique



(a) Solution 1



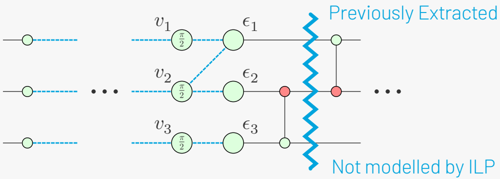
(b) Solution 2

[45] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

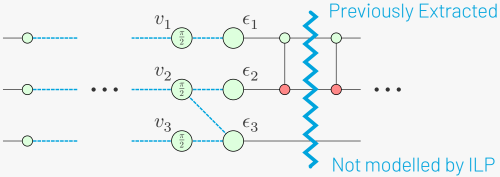
Part II: Integer Linear Programming for Circuit Extraction

Contribution IV: Extract a single Frontier using ILP [45]

- ILP that minimizes row additions to extract a single Frontier
- Extraction of a single Frontier is a local solution
- Local solutions are not unique



(a) Solution 1



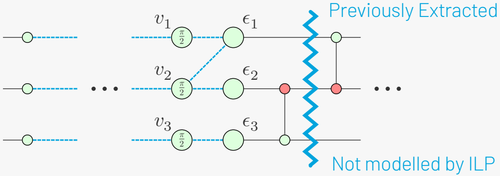
(b) Solution 2

[45] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

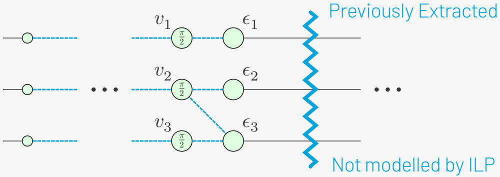
Part II: Integer Linear Programming for Circuit Extraction

Contribution IV: Extract a single Frontier using ILP [45]

- ILP that minimizes row additions to extract a single Frontier
- Extraction of a single Frontier is a local solution
- Local solutions are not unique
- CNOTs cancel out in Solution 2
- Evaluate local solutions with backtracking & gate commutation
- Nodes are partially extracted circuits
- Leafs are fully extracted circuits



(a) Solution 1



(b) Solution 2

[45] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

Experimental Setup [47]

Benchmark

- 512 Quantum Circuits
- 1 hour optimizer runtime (Holker)
- 1 hour extractor runtime
- Best-first search (BeFS) and DFS for backtracking
- GLPK solver for ILP [46]
- External procedure for gate commutation and cancellation

[46] Makhorin (2020), *GLPK (GNU Linear Programming Kit)*
[47] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

Part II: Integer Linear Programming for Circuit Extraction [48]

Row Addition Optimization

- Baseline Gaussian Elimination
- 86% of the instances finish for the backtracking extractors
- On instances where all extractors finish:
  - ILP-based extractors outperform the Gaussian Elimination Extractor by:
    - 20% w.r.t. circuit-depth
    - 27% w.r.t. total gate count
- 2Q count remains unchanged
- Identical performance among backtracking and non-backtracking ILPs

Algorithm	Gaussian			Single ILP			ILP DFS			ILP BeFS		
Metric $\Delta\%$	d	2Q	G	d	2Q	G	d	2Q	G	d	2Q	G
Small	0.0	0.0	12.3	-19.59	0.0	-20.53	-19.59	0.0	-20.53	-19.59	0.0	-20.53
Medium	23.94	-30.95	35.16	-18.31	-30.95	-13.74	-18.31	-30.95	-13.74	-18.31	-30.95	-13.74
T-Par+NAM	63.01	-0.76	66.67	1.94	-0.87	-7.51	1.94	-0.87	-7.51	1.94	-0.87	-7.51
CX	-8.89	0.0	-19.08	-27.35	0.0	-39.98	-27.35	0.0	-39.98	-27.35	0.0	-39.98
Clifford	-7.69	0.0	-18.66	-27.78	0.0	-39.79	-27.78	0.0	-39.79	-27.78	0.0	-39.79
Clifford+T	-10.0	0.0	-17.93	-29.22	0.0	-40.21	-29.22	0.0	-40.21	-29.22	0.0	-40.21
Median	-3.85	**0.0**	-2.81	**-23.47**	**0.0**	**-30.16**	**-23.47**	**0.0**	**-30.16**	**-23.47**	**0.0**	**-30.16**

[48] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)

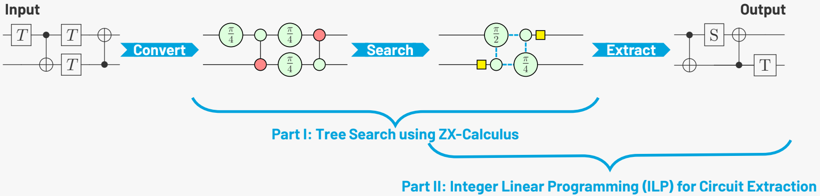
Conclusion

Part I: Tree search using ZX-Calculus

- General method that can compete with specialized optimizers
- Slightly worse performance in the price of generality
- Framework that can be extended for current and future metrics

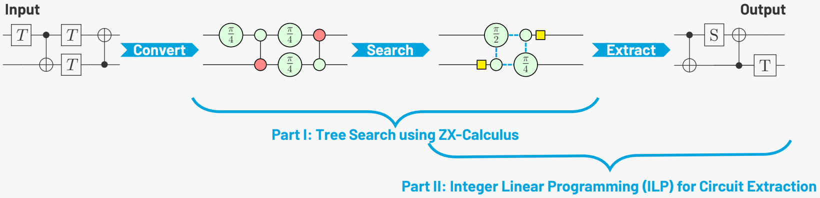
Part II: Integer Linear Programming for Circuit Extraction

- ILP-based extractors outperform the SOTA Gaussian Elimination Extractor w.r.t. circuit-depth and total gate count
- 2Q gate count remains unchanged after Holker
- ZX-diagram size impact backtracking performance



Contributions

1. Formalization of ZX-diagram optimization problem as state-space exploration spanned by rewriting rule sequences
2. Implementation of tree search algorithms (DFS, IDDFS, LDS, Local Elimination) to reduce the T-gate and edge count of ZX-diagrams
3. Enhancing the 2Q gate count circuit depth during extraction with an ILP model paired with backtracking
4. Introduction of the declarative *ZX-Benchmark* framework for ZX-based QC optimization



Perspectives

Tree Search using ZX-Calculus

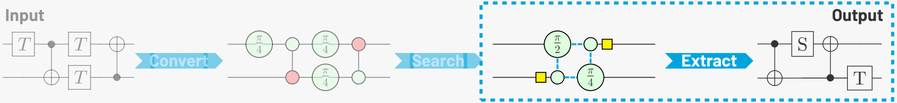
Part I: Tree Search using ZX-Calculus

- Scalability
- Implementation in higher-performant programming languages
- Surrogate models
- Multi-objective optimization
- Parallel and distributed search algorithms

Perspectives

Integer Linear Programming for Circuit Extraction

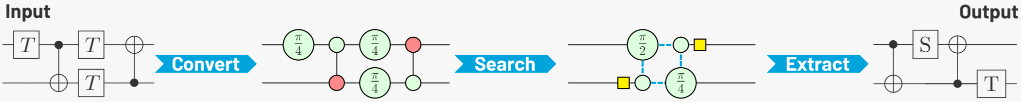
Part II: Integer Linear Programming for Circuit Extraction



- Extend the constraints to:
  - consider architectural constraints (e.g., connectivity)
  - apply gate cancellation and commutation
- Progressive refinement of fast non-optimal solution
- Development of a heuristic that only backtracks on promising adjacency matrices

Perspectives

Pipeline Improvements



- Expand list of supported optimizers
- Identify representative benchmark data sets
- Model architecture-aware cost functions based on third-party transpilation pipelines
- Generate website with public benchmarks

Submitted

- **Tobias M. Fischbach**, P. Talbot, and P. Bouvry. ***Integer Linear Programming for Quantum Circuit Extraction.*** 2026. (Submitted)

Publications

- **Tobias M. Fischbach**, P. Talbot, and P. Bouvry. **"Exhaustive Search for Quantum Circuit Optimization Using ZX Calculus".** In: *Optimization and Learning*. Ed. by B. Dorronsoro et al. Cham: Springer Nature Switzerland, 2026, pp. 239–253. (published)
- **Tobias M. Fischbach**, P. Talbot, and P. Bouvry. **"A Review on Quantum Circuit Optimization using ZX-Calculus".** In: *arXiv* (Sept. 2025). DOI: `https://doi.org/10.48550/arXiv.2509.20663`. (preprint)
- **Tobias M. Fischbach**, E. Kieffer, and P. Bouvry. ***Challenges in Automatic Software Optimization: the Energy Efficiency Case.*** 2023. (published)
- J.-B. Linse, **Tobias M. Fischbach**, and J. S. Hub. **"How protein hydration depends on amino acid composition, peptide conformation, and force fields".** In: *Biophysical Journal* (2025). DOI: `https://doi.org/10.1016/j.bpj.2025.11.2683`

Thank you for your attention!

Supporting Slides

Definition of ZX-Diagram Optimization

Quantum Circuit Optimization

- Input & output quantum circuit
  - Let **QC** be the set of quantum circuits
- opt is a function that maps a quantum circuit to a metric
  - $opt : \mathbf{QC} \to \mathbb{Z}$

Define quantum circuits

- Linear map of qubits
  - Let **LM** be the set of the linear map of qubits
- γ is a function that maps a quantum circuit to its linear map
  - $\gamma : \mathbf{QC} \to \mathbf{LM}$

Quantum circuits implement a linear map of qubits

- f is a function to transform quantum circuits
  - $f : \mathbf{QC} \to \mathbf{QC}$
  - f is semantic-preserving $\forall q \in \mathbf{QC},\ \gamma(q) = \gamma(f(q))$
  - f minimizes the objective function $\min opt$

Functions that transform quantum circuits need to **keep the same** linear map of qubits

$\Rightarrow$ f is a quantum circuit optimization algorithm

Definition of ZX-Diagram Optimization

ZX-based Quantum Circuit Optimization

- ZX diagrams
  - Let **ZX** be the infinite set of all finite ZX diagrams
- α is a function that converts quantum circuits to ZX diagrams
  - α : **QC** $\rightarrow$ **ZX**
- $extract$ is a function that converts ZX diagrams to quantum circuits
  - $extract$: **ZX** $\rightarrow$ **QC** $\cup \{\perp\}$
  - A ZX diagram is extractable if $zx \in$ **ZX**, $extract(zx) \neq \perp$
- Rewriting rules
  - Let **R** $= \{i1, i2, f, h, b, c, \pi, hd, lc\}$ be the set of rewriting rules
- $r \in R$ is a function that transforms a ZX diagram
  - $r \in R$: **ZX** $\rightarrow$ **ZX**

Every quantum circuit can be converted to a ZX diagram

Not every ZX diagram can be extracted

Rules are functions that transform ZX diagrams

$\Rightarrow$ A ZX based QC algorithm searches for extractable ZX diagrams that improves QC metric(s)

Definition of ZX-Diagram Optimization

ZX-based Quantum Circuit Optimization

State-space formed by finite sequence of rules on converted quantum circuit

- W is the ZX state-space
  - $q \in QC, W \subseteq$ **ZX** s.t. $w \in W$ exists for a finite sequence of rewriting rules
  - $w = (r_n \circ \ldots \circ r_1 \circ \alpha)(q)$
- w is a function that converts a QC to a ZX diagram and applies a finite sequence of rewriting rules
  - $w = (r_n \circ \ldots \circ r_1 \circ \alpha)(q)$
- Solutions are all extractable ZX diagrams in the state-space W:
  - $S \subseteq W$
  - $\forall w \in W,$ *extract*$(w) \neq \perp \Leftrightarrow w \in S$

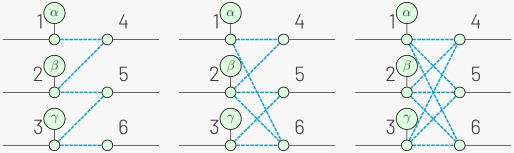
Extractable ZX diagrams of the state-space are solutions

- **Optimal solutions** is the largest subset of solutions such that a metric is minimized
  - $O \subseteq S$
  - $\forall o \in O, s \in S, opt(extract(o)) \leq opt(extract(s))$

All solutions ZX diagrams that minimize a metric are optimal

Flows and Determinism

- Deterministic measurement pattern if diagram has flow
- Measurement pattern is deterministic if only upcoming qubits are affected



(a) Causal flow **(b)** General flow **(c)** No flow

Extract a single Frontier using ILP

Minimize row additions $\rightarrow$ Minimize CNOTs

- Adjacency Matrix **M** $\in \mathbb{GF}(2)$
- Rows $i \in [1, m]$, Columns $j \in [1, n]$
- Only row additions are allowed
  $y_j = \left(\sum_i x_i M_{ij}\right) \mod 2, \forall j \in [1, n]$
- Only Frontier spiders with a single neighbor are extractable

Decision Variables	
$x_i \in \{0, 1\}$	select row i
$y_j \in \{0, 1\}$	bit at position j XOR vector
$s_j \in \mathbb{N}$	sum of selected row bits
$z_j \in \mathbb{N}$	auxiliary variable
Constraints	
$\sum_{j=1}^{n} y_j = 1$	extractability
$s_j = \sum_{i=1}^{m} x_i \cdot M_{ij}$	unmodded sum
$y_j = s_j - 2z_j$	modulo 2 linearization
Objective Function	
$\min \sum_{i=1}^{m} x_i$	minimize row additions

Backtracking ILP

Enumerate optimal Solutions

- Modify previous ILP with additional parameters from previous solutions
- Exclude previously encountered solutions (no-good cuts)[49]

Decision Variables	
$x_i \in \{0,1\}$	select row i
$y_j \in \{0,1\}$	bit at position j XOR vector
$s_j \in \mathbb{N}$	sum of selected row bits
$z_j \in \mathbb{N}$	auxiliary variable
Parameters	
$x_i^k \in \{0,1\}$	previous solution selecting row i
$n_{\text{opt}} \in \mathbb{N}^+$	optimal solution
Constraints	
$\sum_{j=1}^{n} y_j = 1$	extractability
$s_j = \sum_{i=1}^{m} x_i \cdot M_{ij}$	unmodded sum
$y_j = s_j - 2z_j$	modulo 2 linearization
$\sum_{i:x_i^k=1}(1-x_i) + \sum_{i:x_i^k=0} x_i \geq 1, \forall k \in [1,P]$	exclude enumerated solutions
$\sum_{i=1}^{m} x_i = n_{\text{opt}}$	enumerate optimal solutions

[49] Hooker (2011), *Logic-based methods for optimization: combining optimization and constraint satisfaction*

Backtracking ILP

Enumerate optimal Solutions

- Modify previous ILP with additional parameters from previous solutions
- Exclude previously encountered solutions (no-good cuts) [49]
- Backtrack on enumerated solutions
- Leafs are fully extracted circuits
- Optimize partial circuits

function $\textsc{AdjSolver}(M)$
    $n_{\text{opt}}, x \leftarrow \textsc{MinimizeILP}(M)$ (find minimal number of row additions)
    $\mathcal{S} \leftarrow \{x\}$
    while true do (enumerate solutions)
        $c \leftarrow \{\sum_{i=1}^{m} x_i \cdot M_{ij}, s_j - 2z_j, \sum_{j=1}^{n} y_j = 1, \sum_{i=1}^{m} x_i = n_{\text{opt}}\}$ (constraints)
        for $\forall x^{\text{prev}} \in \mathcal{S}$ **do** (exclude already enumerated solutions)
            $c \leftarrow c \cup \{\sum_{i: x_i^{\text{prev}}=1} (1 - x_i) + \sum_{i: x_i^{\text{prev}}=0} x_i \geq 1\} \cup c$
        end for
        $x \leftarrow \textsc{Solve}(M, c)$ (solve ILP)
        if x **is** $\emptyset$ **then**
            break (no feasible solution)
        end if
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{x\}$ (add solution)
    end while
    return $\mathcal{S}$
end function

[49] Hooker (2011), *Logic-based methods for optimization: combining optimization and constraint satisfaction*

Part II: Integer Linear Programming for Circuit Extraction

Row Addition Optimization [50]

- Full reduce + Extractor
- Baseline Gaussian Elimination

Algorithm	Gaussian			ILP BeFS			ILP DFS			Single ILP		
Metric Δ%	d	2Q	G	d	2Q	G	d	2Q	G	d	2Q	G
Small	30.43	**100.0**	33.33	**8.7**	**100.0**	**17.65**	**8.7**	**100.0**	**17.65**	**8.7**	**100.0**	**17.65**
Medium	97.18	**21.43**	19.23	**35.21**	**21.43**	**-1.65**	**35.21**	**21.43**	**-1.65**	59.15	30.95	0.55
full	65.0	113.64	25.0	34.62	**106.25**	3.03	**32.0**	**106.25**	**2.08**	37.5	122.22	8.08
CX	101.77	189.47	0.72	**82.21**	**169.22**	**-4.62**	**82.21**	**169.22**	**-4.62**	85.24	188.51	-3.3
Clifford	141.93	276.1	12.76	**106.59**	**241.13**	**1.47**	**106.59**	**241.13**	**1.47**	107.2	271.55	9.93
Clifford+T	59.43	175.87	-4.61	60.2	**172.03**	**-12.14**	61.51	**172.03**	**-12.14**	**58.88**	183.57	-6.91
Median	81.09	144.76	16.0	**47.7**	**137.74**	**-0.09**	48.36	**137.74**	**-0.09**	59.02	152.89	4.32

[50] **Tobias M. Fischbach**, Talbot, and Bouvry (2026), *Integer Linear Programming for Quantum Circuit Extraction* (submitted)