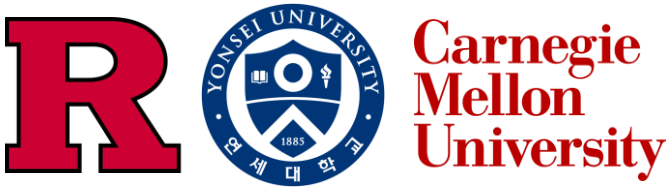
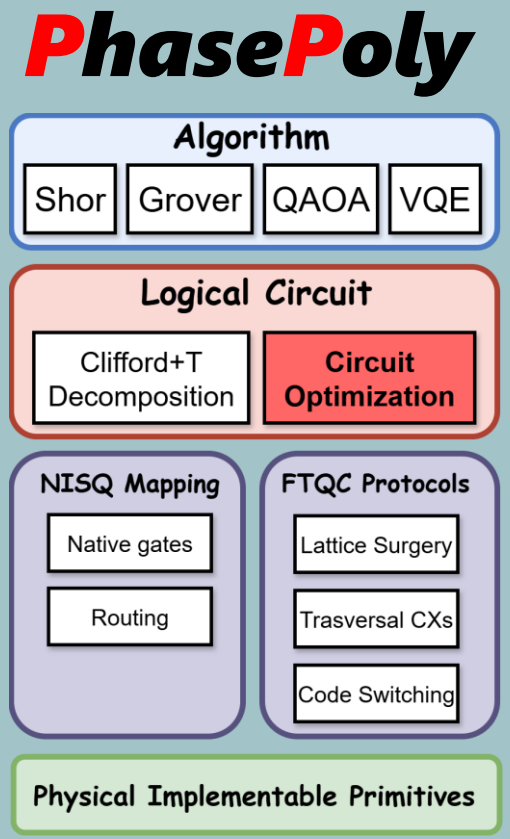


Leveraging Phase Polynomials for Quantum Circuit Optimization

Zihan Chen*, Henry Chen*, Yuwei Jin*, Enhyeok Jang†, Mingkuan Xu‡, Vannessa Chan*, Won Woo Ro†, Eddy Z. Zhang*

* Rutgers University, NJ, USA † Yonsei University, Seoul, Korea ‡ Carnegie Mellon University, PA, USA

Presenter: Zihan Chen (zihan.chen.cs@rutgers.edu)



Leveraging Phase Polynomials for Quantum Circuit Optimization

- Zihan Chen*, Henry Chen*, Yuwei Jin*, Enhyeok Jang†, Mingkuan Xu‡, Vannessa Chan*, Won Woo Ro†, Eddy Z. Zhang*

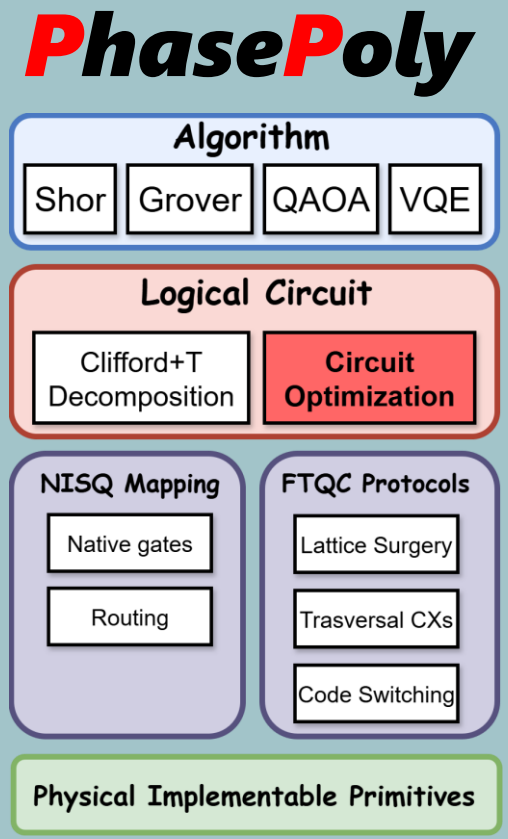
  * Rutgers University-New Brunswick, NJ, USA
  † Yonsei University, Seoul, Korea
  ‡ Carnegie Mellon University, PA, USA

- arXiv preprint: 2506.20624

- Software access: https://github.com/ruadapt/PhasePoly

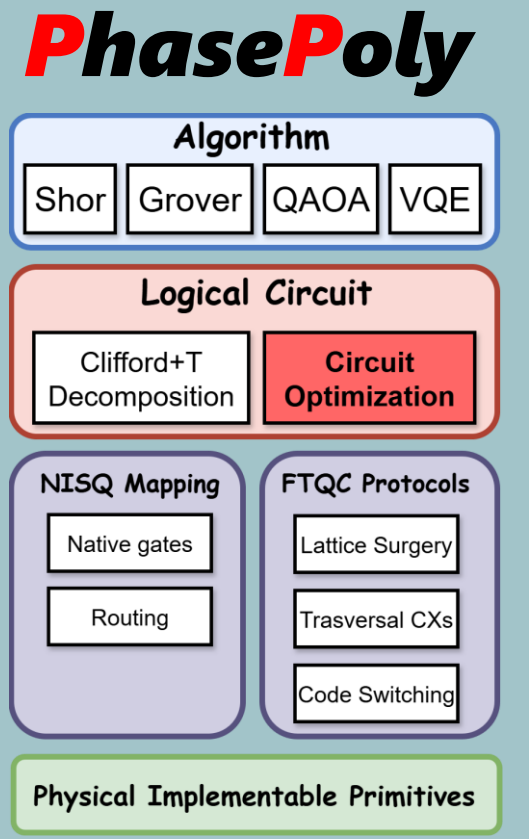
Preprint:

Code:



1. Background and Motivation
2. PhasePoly's Compilation Techniques
3. Evaluation and Insights
4. Future Work and Conclusion

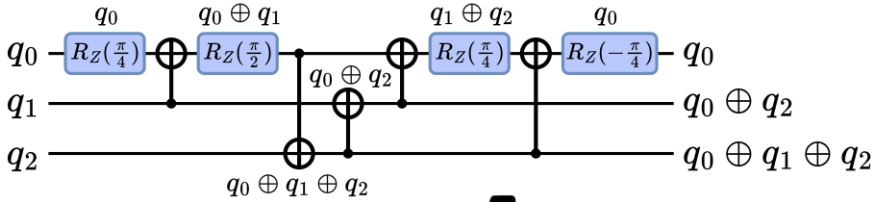
RUTGERS Carnegie Mellon University



RUTGERS | Carnegie Mellon University

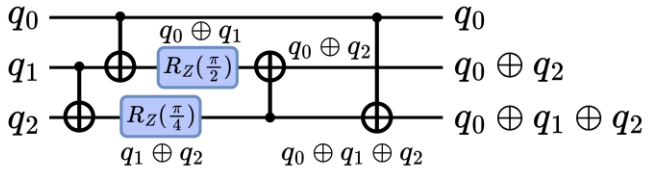
1. Background and Motivation
2. PhasePoly's Compilation Techniques
3. Evaluation and Insights
4. Future Work and Conclusion

1.1 Logical Circuit Optimization



(a) Original Circuit
(5 CNOT gates & 3 T gates)

Phase Polynomial Optimization



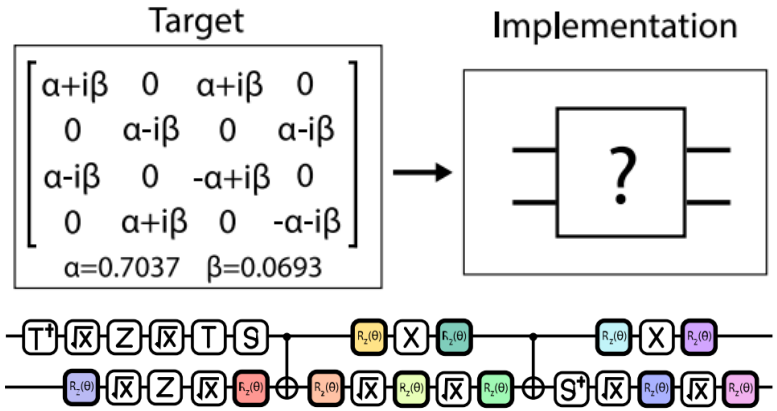
(b) Optimized Circuit
(4 CNOT gates & 1 T gates)

- Input a quantum circuit, output an **equivalent**, **better** quantum circuit
- Minimize:
  - Total gate count
  - Specific gate count
  - ...(other metrics)
- Why it matters:
  - Less error
  - Less runtime
  - ...(other metrics)
- Optimizing quantum circuit is generally HARD[1]

RUTGERS | Carnegie Mellon University

[1] van de Wetering, John, and Matt Amy. "Optimising quantum circuits is generally hard." arXiv preprint arXiv:2310.05958 (2023).

1.2 Existing Methods



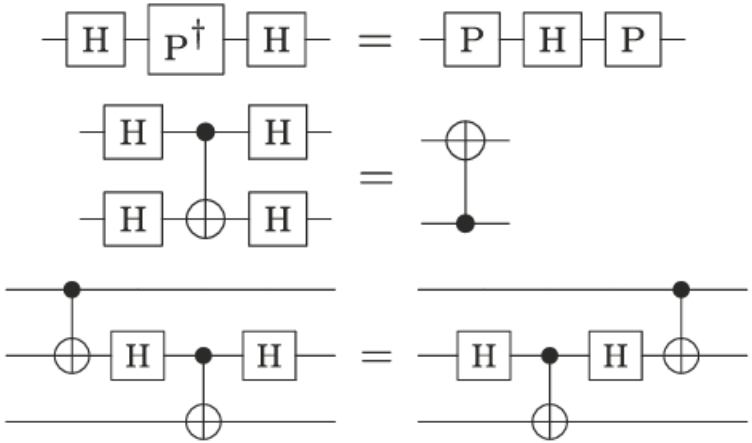
Unitary Synthesis

- Gridsynth
- Trasyn
- BQSKit — Berkeley Quantum Synthesis Toolkit
- Paradis, Anouk, et al. "Synthetiq: Fast and versatile quantum circuit synthesis." 2024

Source: Weiden, Mathias, et al. "High-Precision Multi-Qubit Clifford+ T Synthesis by Unitary Diagonalization." arXiv preprint arXiv:2409.00433 (2024).

1.2 Existing Methods

- Consecutive H gate
- Consecutive Rotation gate
- Consecutive CNOT gate

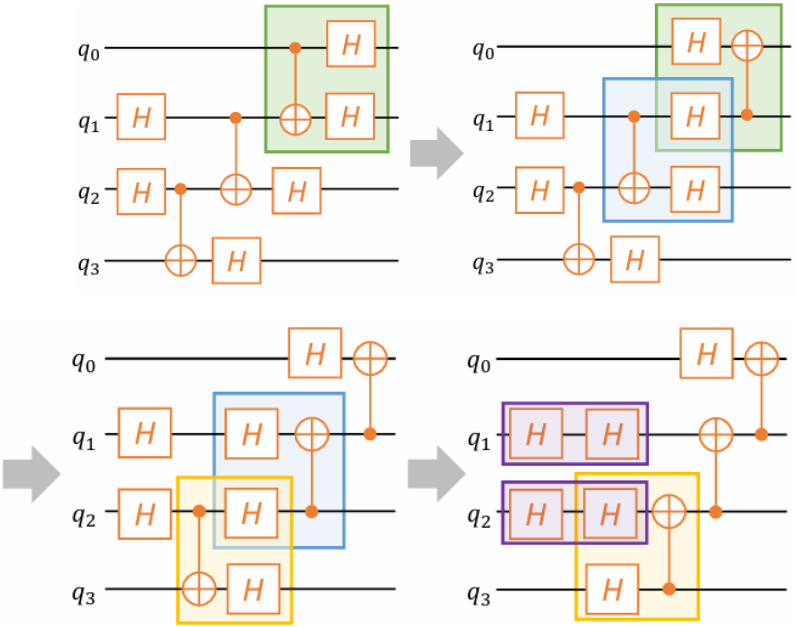


Manual Rule-based Optimization

- Qiskit optimized transpiler
- Tket CliffordSimp
- Nam, Yunseong, et al. "Automated optimization of large quantum circuits with continuous parameters." 2018
- Hietala, Kesha, et al. "A verified optimizer for quantum circuits." 2021

Source: Nam, Yunseong, et al. "Automated optimization of large quantum circuits with continuous parameters." npj Quantum Information 4.1 (2018): 23.

1.2 Existing Methods



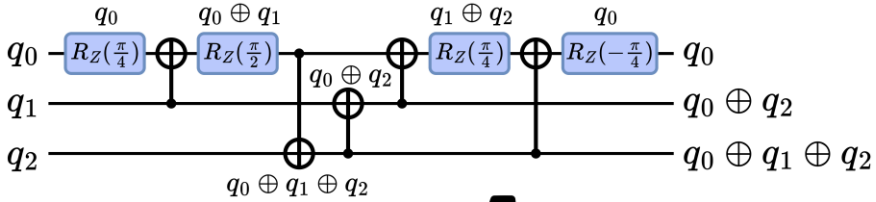
Subcircuit Rewriting Methods

- Quartz 2022: ECC(Equivalent Circuit Clas) sets generation
- QUESO 2023: PIF(Polynomial identity filter) enhanced ECC sets
- Quarl 2024: Reinforcement learning
- GUOQ 2025: Unitary synthesis
- Quasar 2026: E-graph
- ……

RUTGERS Carnegie Mellon University

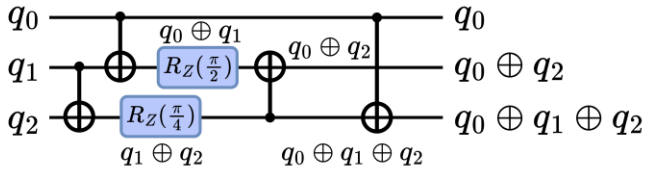
Source: Xu, Mingkuan, et al. "Quartz: superoptimization of quantum circuits." Proceedings of the 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation. 2022.

1.3 Phase Polynomial Representation



(a) Original Circuit
(5 CNOT gates & 3 T gates)

Phase Polynomial Optimization



(b) Optimized Circuit
(4 CNOT gates & 1 T gates)

- Phase-polynomial circuit: {CNOT, Rz}
- *sum-over-paths*[1]:

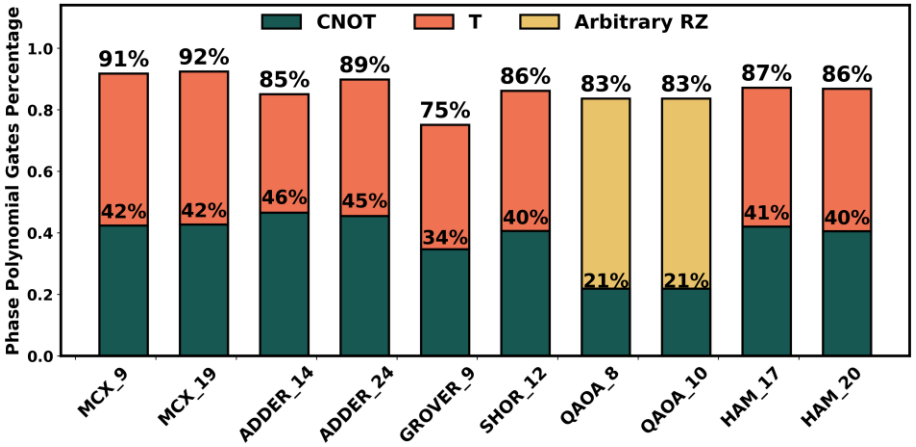
$$U\,|x_1,\ldots,x_n\rangle = e^{ip(x_1,\ldots,x_n)}\,|g(x_1,\ldots,x_n)\rangle$$

  - Phase parity function: $p(x)$
  - Output basis transformation function: $g(x)$
- Phase polynomial representation is useful in logical/hardware optimization and equivalence checking
- Non phase polynomial gate will break the representation: H, *measure*, *U3*
- Treated as auxiliary tools…

RUTGERS | Carnegie Mellon University

[1] Amy, Matthew, Dmitri Maslov, and Michele Mosca. "Polynomial-time T-depth optimization of Clifford+ T circuits via matroid partitioning." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33.10 (2014): 1476-1489.

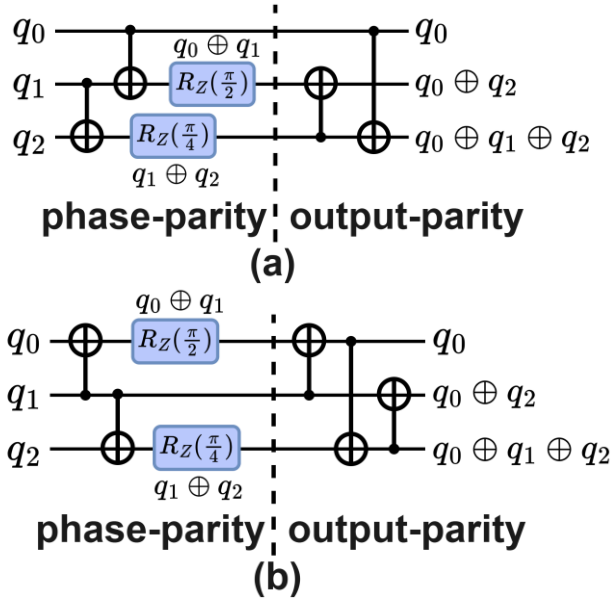
1.4 Why Phase Polynomial Matters: {CNOT, Rz} is the Important Cost Driver



- Why is it worth optimizing phase polynomials?
- Most operations in target workloads fall into {CNOT, Rz} regions.
- CNOT and T gates are important in both NISQ and FTQC era

RUTGERS | Carnegie Mellon University

1.5 Single Phase Polynomial Block Optimization: One Stone Two Birds Needed

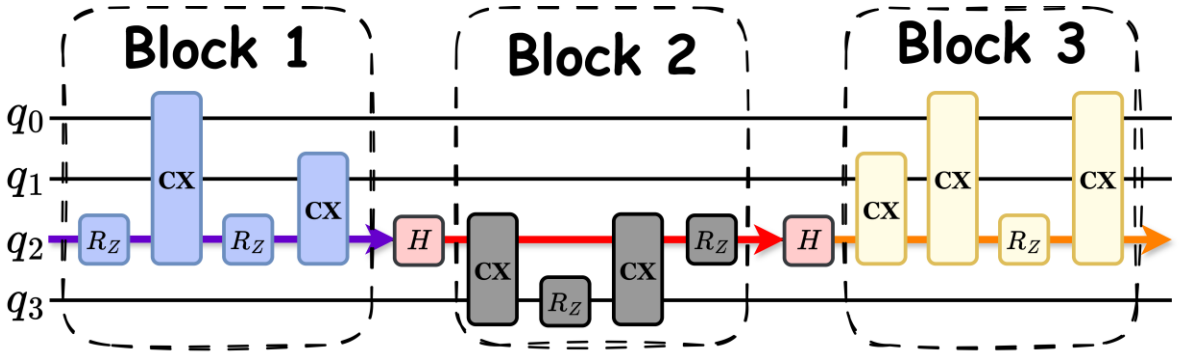


- Same phase function can have multiple implementations
- Different choices leave different output parity states
- Handle separately may miss co-optimization opportunities

RUTGERS | Carnegie Mellon University

1.6 Breaking the Block Barrier: Long-Range Generalized Optimizations

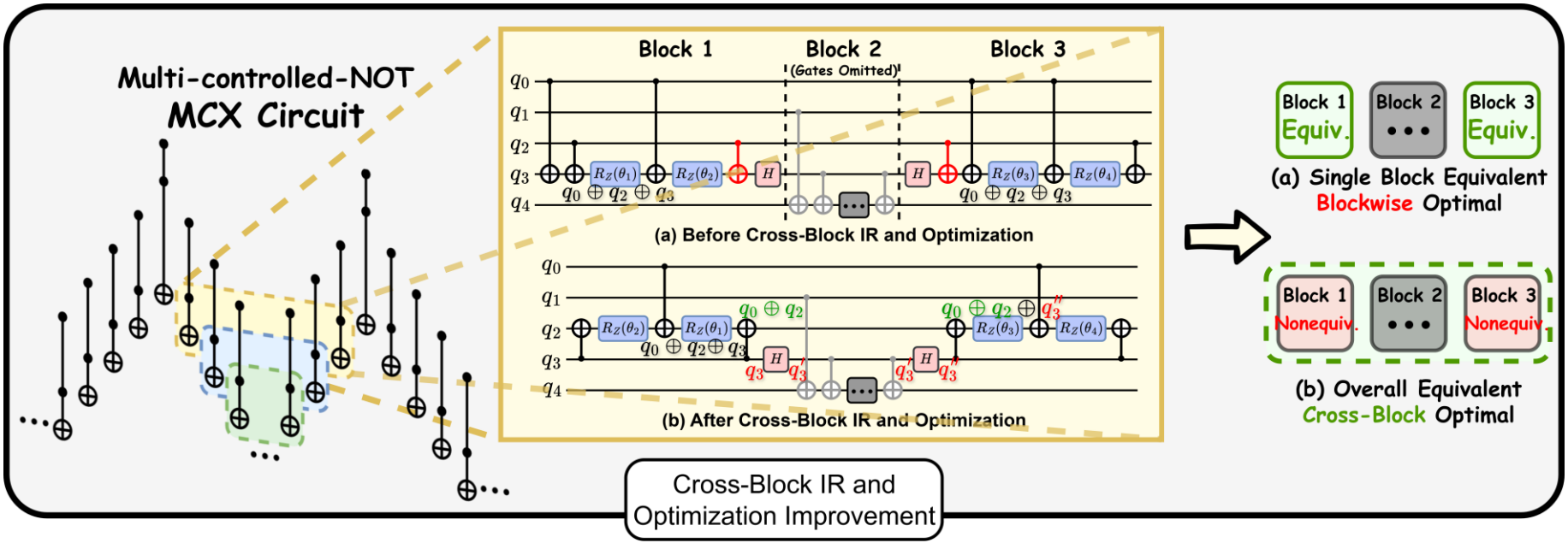
- H gate is necessary for universal quantum computation.
- General quantum circuits will be partitioned by H gates into multiple phase polynomial blocks.



RUTGERS Carnegie Mellon University

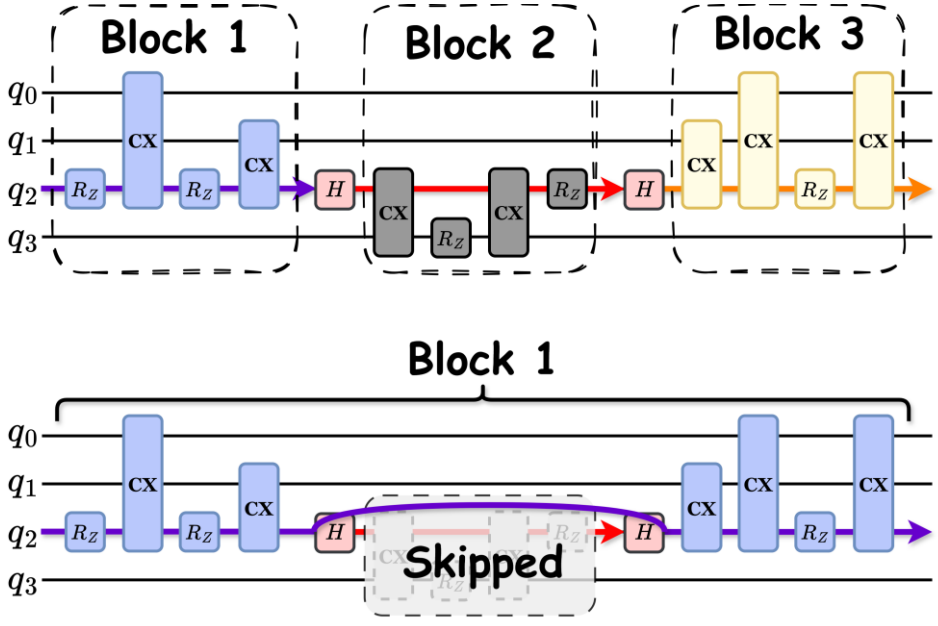
1.6 Breaking the Block Barrier: Long-Range Generalized Optimizations

- We need an intermediate representation can allow us to seize phase parity, output parity and cross-block optimization opportunities at the same time!



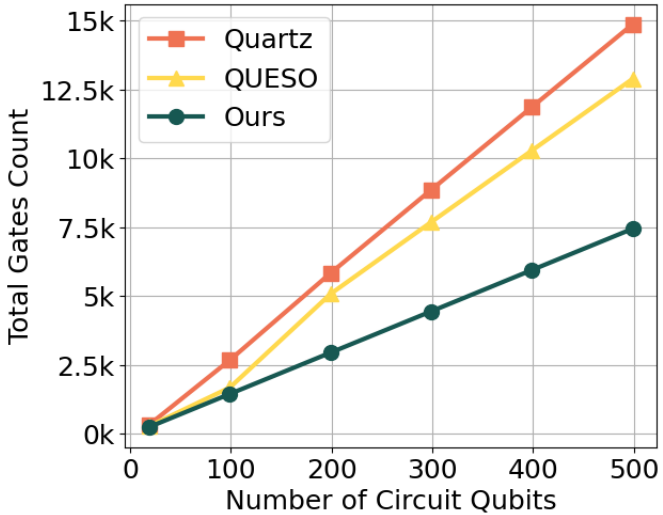
1.7 Overcoming the Scaling Challenges

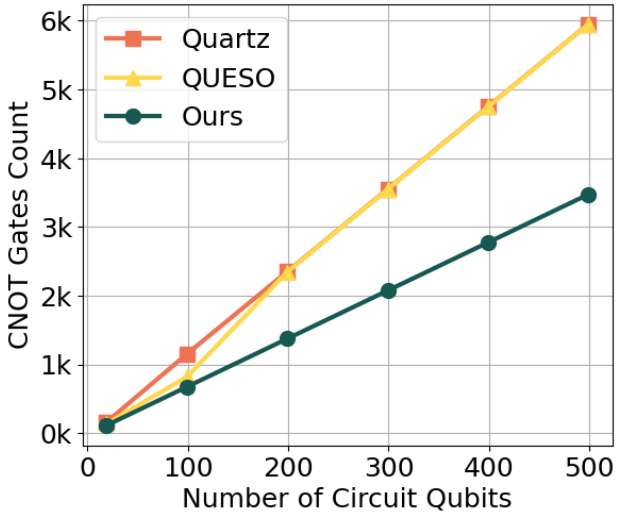
- Subcircuit rewriting is hard to scale when gate sequences are very huge



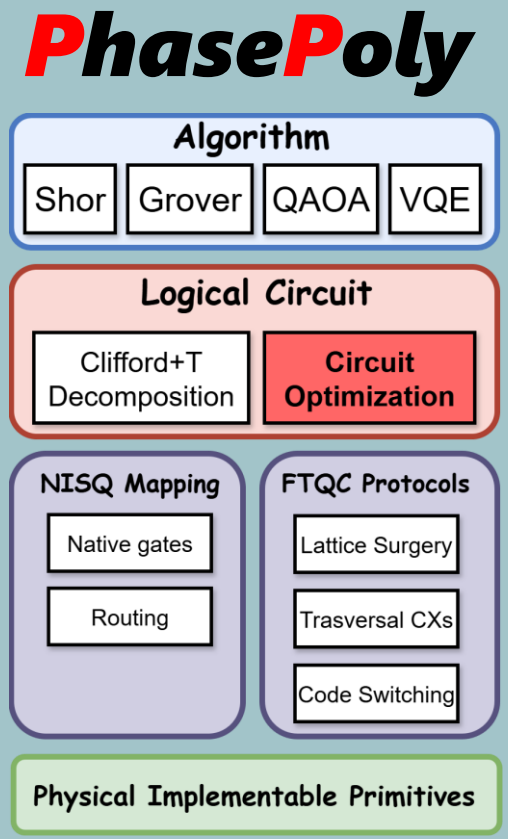
1.7 Overcoming the Scaling Challenges

- Subcircuit rewriting is hard to scale when gate sequences are very huge
- PhasePoly can maintain a good optimization level even in very large circuits





RUTGERS | Carnegie Mellon University



1. Background and Motivation
2. PhasePoly's Compilation Techniques
3. Evaluation and Insights
4. Future Work and Conclusion

RUTGERS Carnegie Mellon University

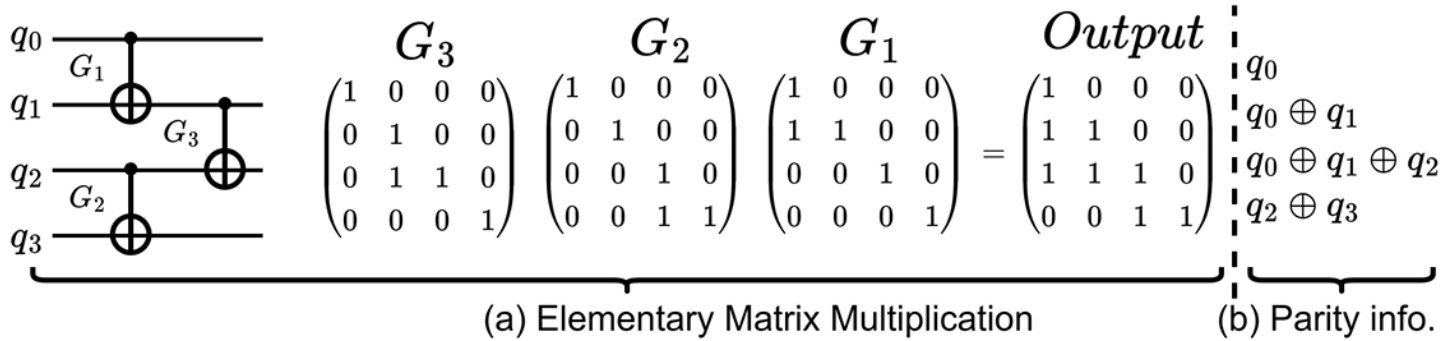
2.1 Coupled Parity Matrix

$$p(x_1, x_2, x_3, x_4) = \theta_1(x_1 \oplus x_2) + \theta_2(x_1 \oplus x_2 \oplus x_3) + \theta_3(x_1 \oplus x_3 \oplus x_4).$$

- Phase-parity network:
  Goal: Eliminate all **1**

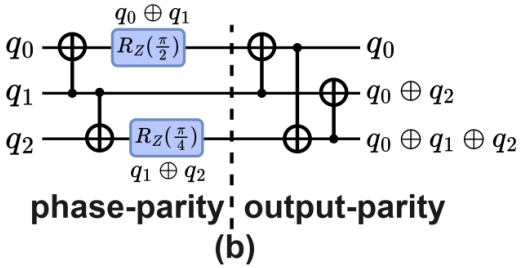
$$P = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

- Output-parity network:
  Goal: Transform it to **diagonal** matrix

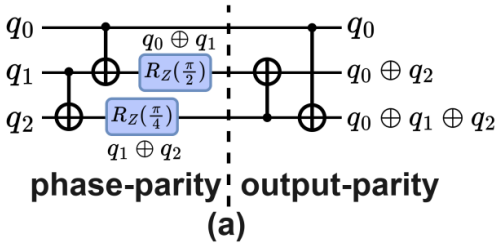


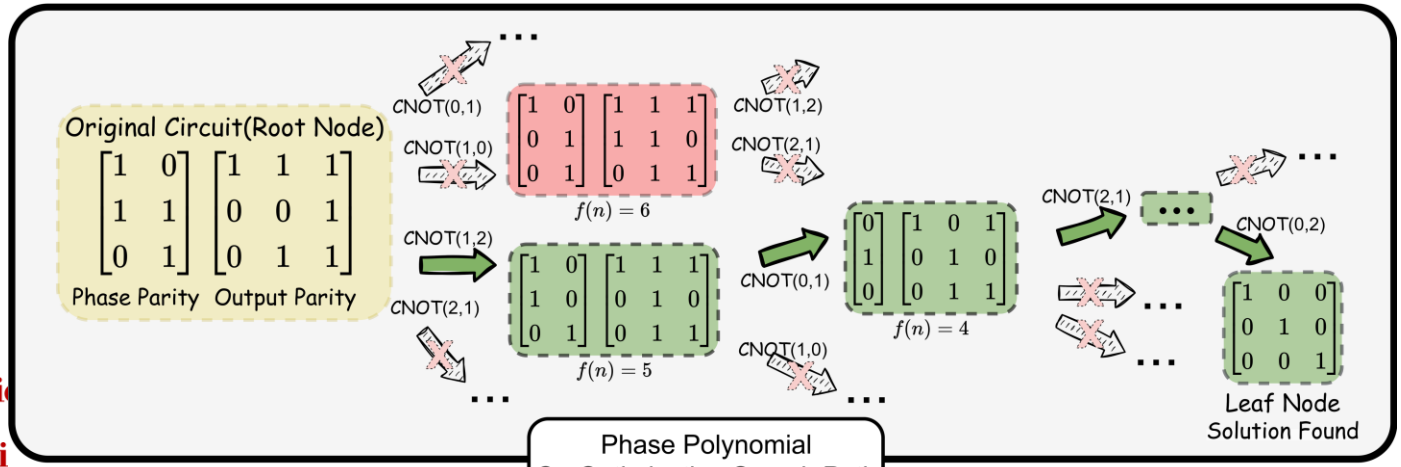
Source: Vandaele, Vivien, Simon Martiel, and Timothée Goubault de Brugière. "Phase polynomials synthesis algorithms for NISQ architectures and beyond." Quantum Science & Technology 7.4 (2022): 045027.

2.1 Coupled Parity Matrix



$$\left[\begin{array}{cc|ccc} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{array}\right] \xrightarrow{\text{CNOT}(q_1, q_0)} \left[\begin{array}{cc|ccc} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{array}\right] \xrightarrow{\text{Insert } R_z} \left[\begin{array}{c|ccc} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}\right]$$



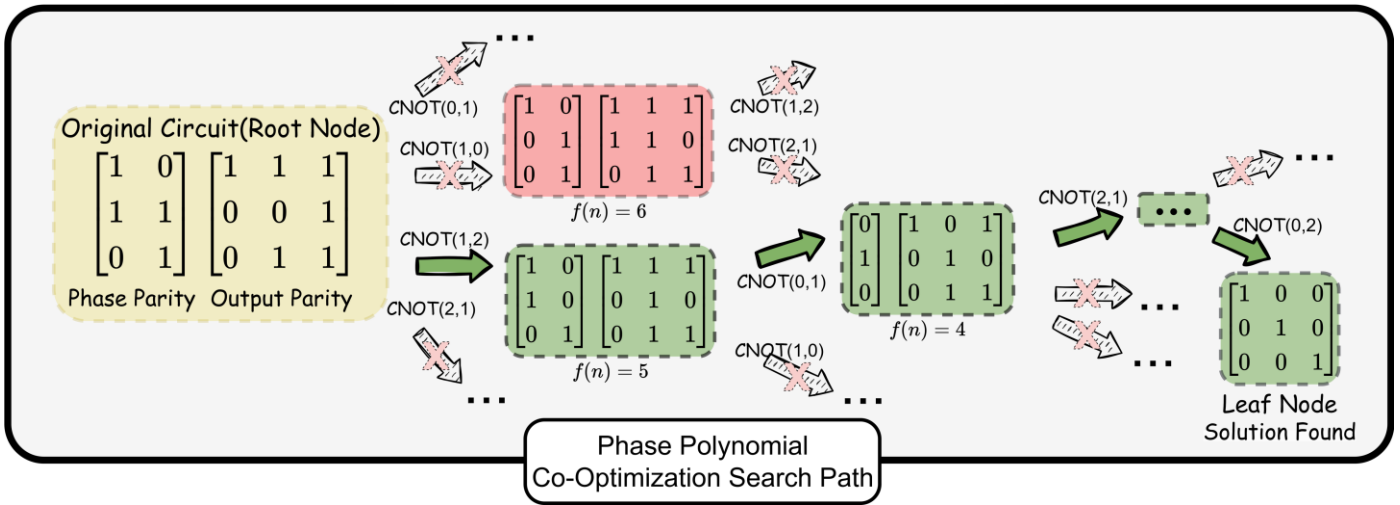


RUTGERS

2.2 Space-bounded A* Search

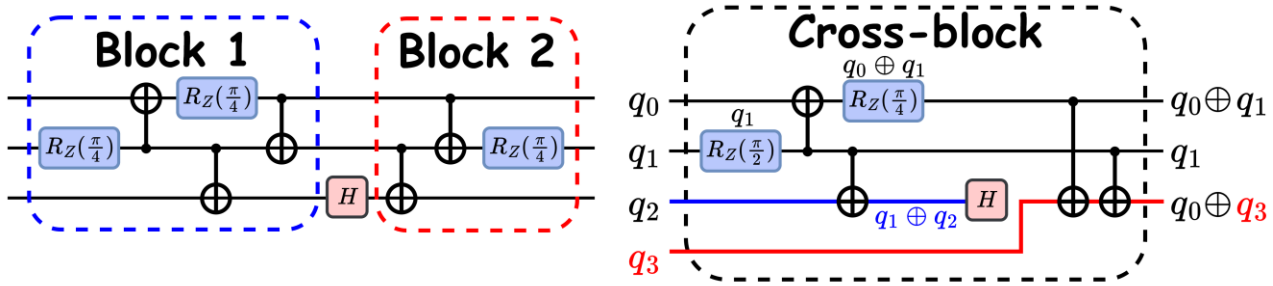
- Phase-parity cost $h_1(n)$: the total Hamming weight in the phase-parity matrix
- Output-parity cost $h_2(n)$: the estimated number of CNOTs required for output-parity matrix
- Accumulated cost $g(n)$: the number of CNOTs already applied
- Pruning to ensure efficiency, bounded priority queue to store intermediate states

$$f(n) = g(n) + h_1(n) + h_2(n)$$



2.3 Cross-Block IR & Optimization

- SSA(static single assignment)-style qubit-state renaming and rotation merging
- Cross-block parity matrix intermediate representation
  - Visual new wire initialized
  - Post-H qubit states remain inactive until H insert
- Merging block *iff* sharing qubits



(a) Before Cross-Block IR and Optimization
(5 CNOT gates & 3 T gates)

(b) After Cross-Block IR and Optimization
(4 CNOT gates & 1 T gates)

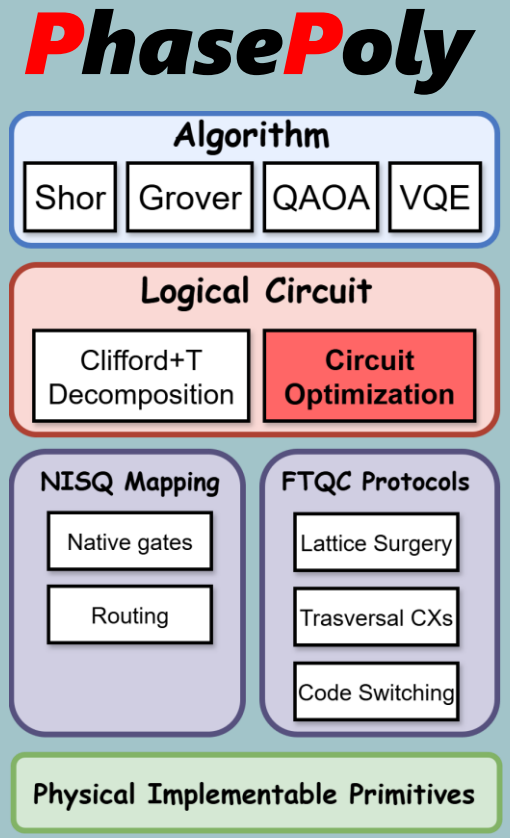
RUTGERS | Carnegie Mellon University

2.3 Cross-Block IR & Optimization

- Condition to active a new wire:
  - No pending phase terms on the row
  - Column/row isolation
  - A linear combination problem solving
  - Included in action sets
- Linear dependency check for pruning
- More blocks merging, more complexity for solving: Incremental Block Merging strategy

$$\left[\begin{array}{cc|cccc} 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \xrightarrow[\text{Insert } R_z(s)]{\text{CNOT}(q_1,q_0)} \left[\begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \xrightarrow{\text{CNOT}(q_1,q_2)} \left[\begin{array}{cccc} 1 & 0 & \mathbf{0} & 1 \\ 0 & 1 & \mathbf{0} & 1 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 0 & \mathbf{0} & 1 \end{array}\right] \xrightarrow[\text{Insert } H]{\text{Eliminate } q_2} \left[\begin{array}{ccc} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{array}\right]$$

RUTGERS Carnegie Mellon University

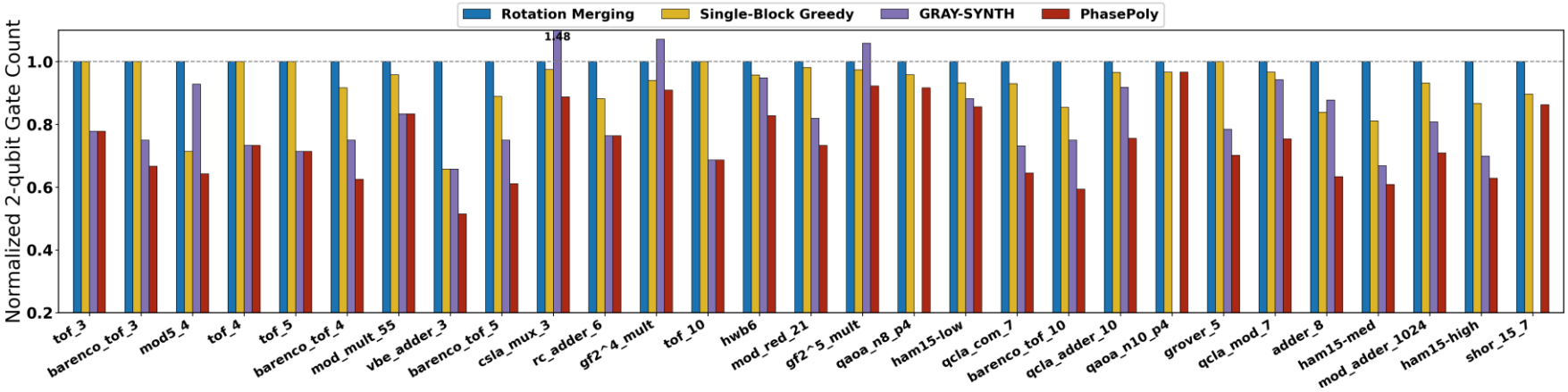


1. Background and Motivation
2. PhasePoly's Compilation Techniques
3. Evaluation and Insights
4. Future Work and Conclusion

RUTGERS Carnegie Mellon University

3.1 Existing Phase Polynomial Optimization

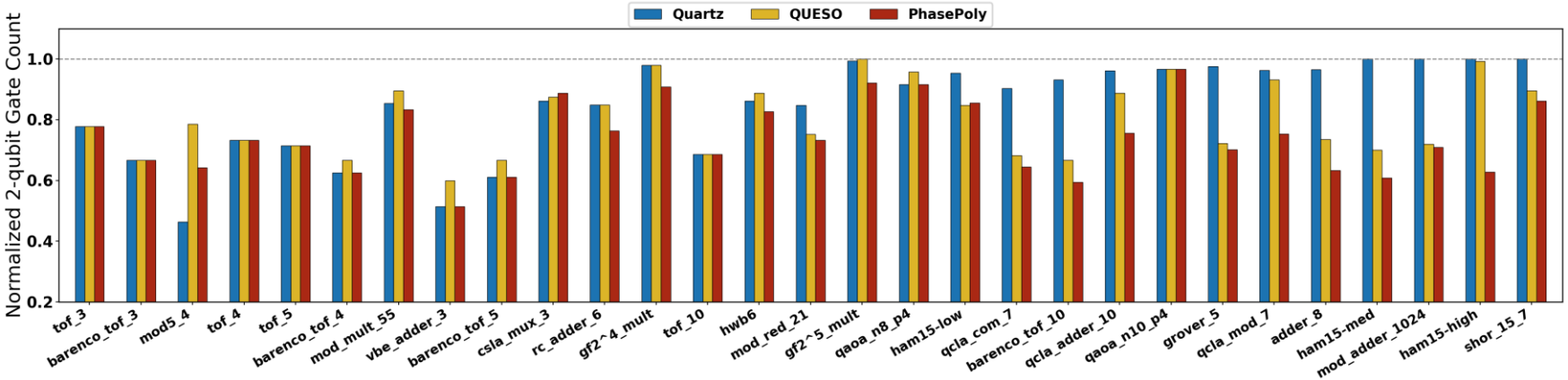
- PhasePoly achieves up to **50%** total-gate reduction and **48.57%** CNOT reduction—**34.70%** and **26.83%** on average
- surpassing all phase polynomial optimization baselines and improving upon Gray-Synth by 9.21% in CNOT reduction.



RUTGERS | Carnegie Mellon University

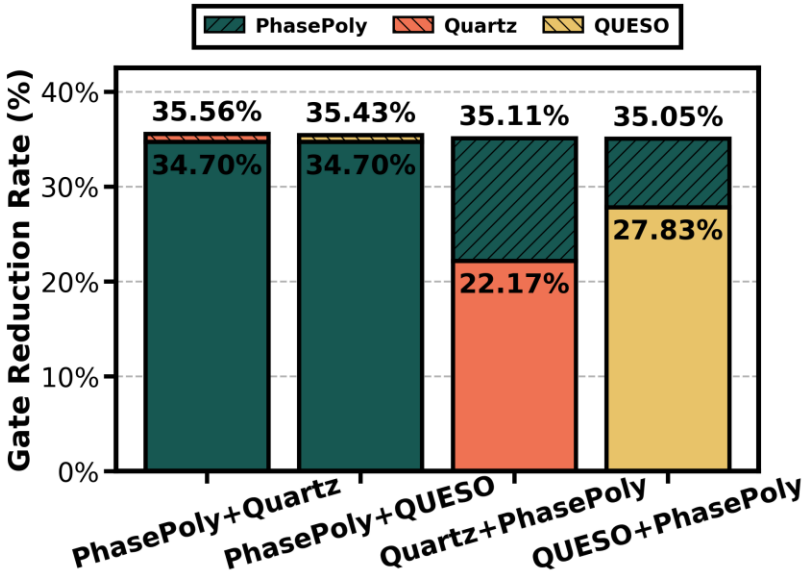
3.2 Existing Subcircuit Rewriting Methods

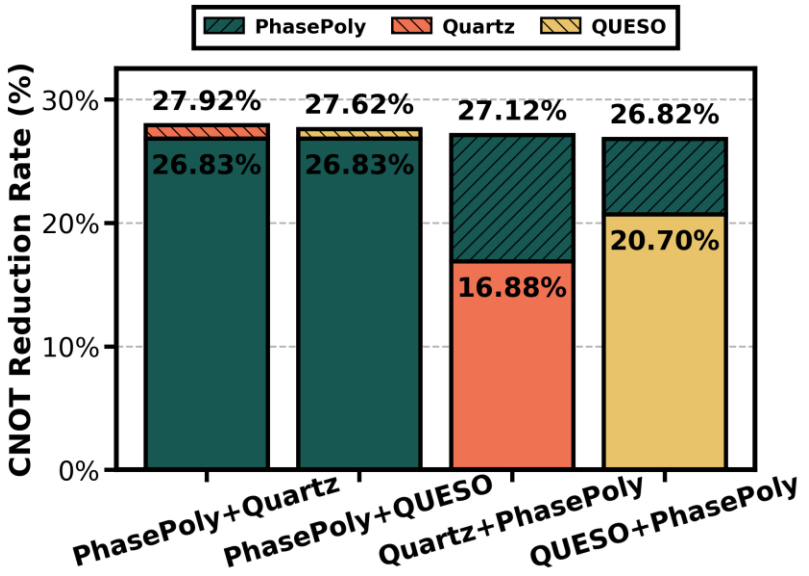
- Quartz and QUESO reduce total gate by 22.17% and 27.83% (CNOTs by 16.88% and 20.70%), while PhasePoly achieves total reduction by **34.70%** (CNOTs by **26.83%**) on average.
- Small (<200 gates): Subcircuit rewriting 2w/6t/2l vs. PhasePoly.
- Medium (200–500 gates): PhasePoly wins 7 out of 10 circuits vs. Subcircuit rewriting.
- Large (>500 gates): PhasePoly wins 8 out of 9 circuits vs. Subcircuit rewriting.



RUTGERS | Carnegie Mellon University

3.3 Orthogonal Integration with Rewriting



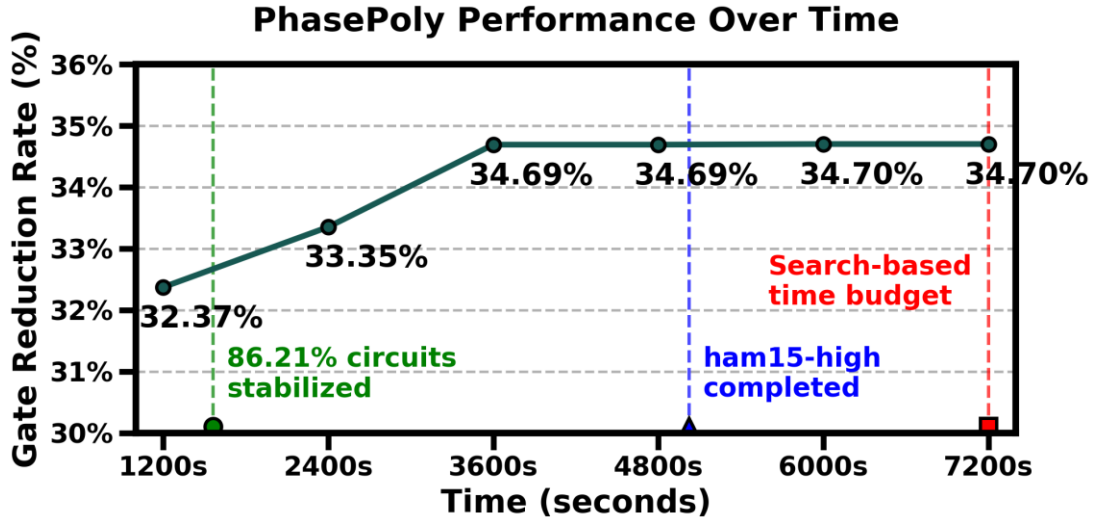


RUTGERS

Carnegie Mellon University

3.4 Scalability and Long-range Optimization

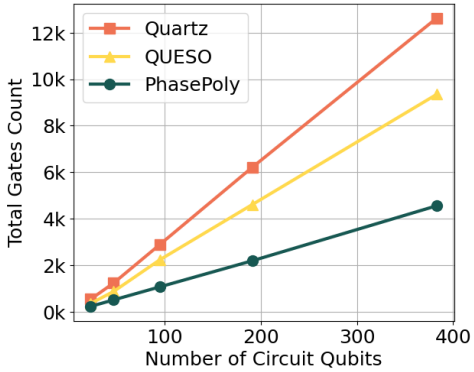
- Even with a deliberate over-optimization Incremental Block Merging strategy.
- PhasePoly reaches 32.37% average gate reduction within 1,200s, and 86.21% of benchmarks stabilize by 1,562s.

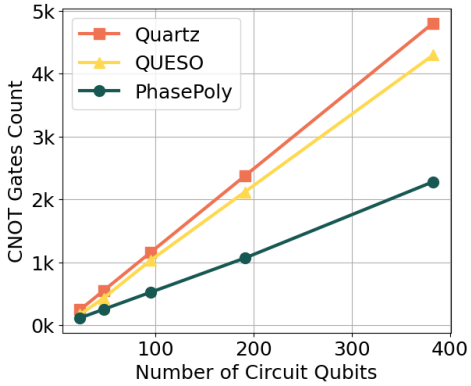


RUTGERS Carnegie Mellon University

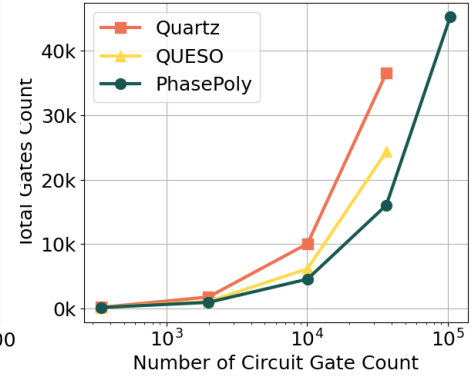
3.4 Scalability and Long-range Optimization

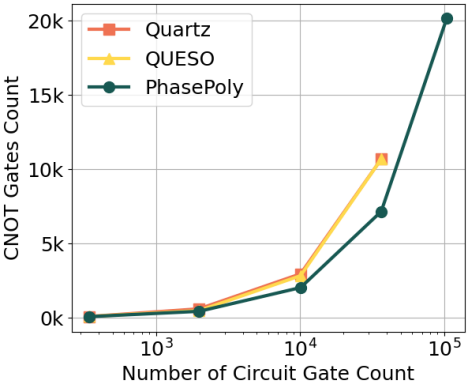
- Adder circuits (**23–383** qubits; **637–12,637** gates); HWB (Hamming coding functions) with a fixed count of **16** qubits but rapidly growing gates (**345–104,068**).
- Quartz and QUESO under a 2-hour time budget; PhasePoly never exceeded **5,500 seconds** even on the largest instance (hwb8_113).





Adder circuits



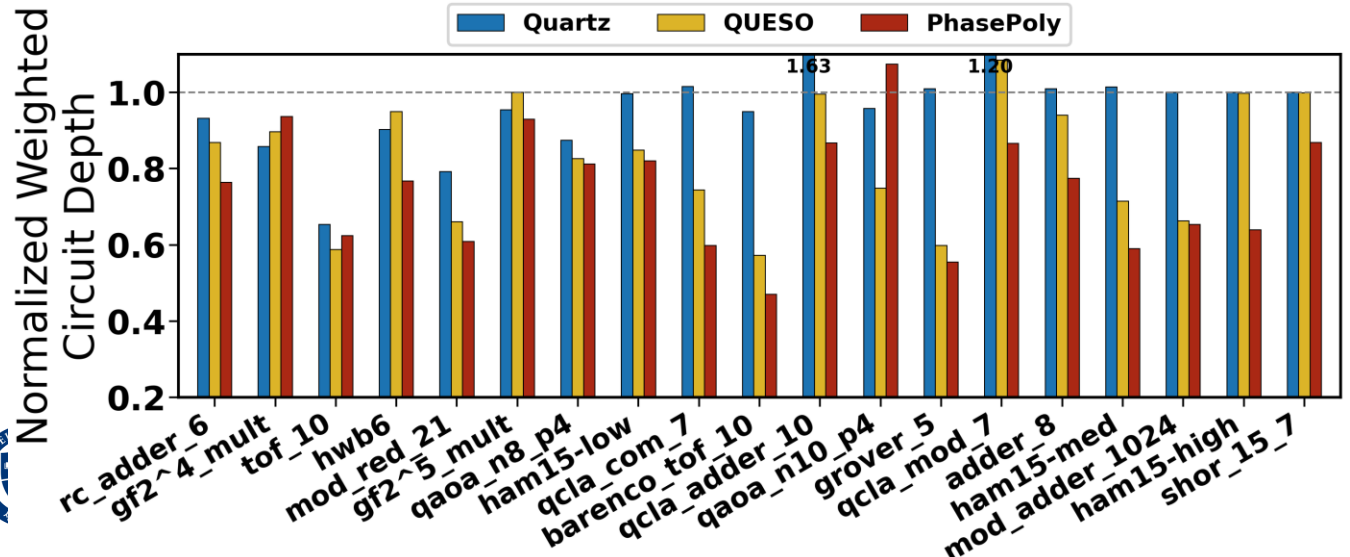


HWB circuits

RUTGERS | Carnegie Mellon University

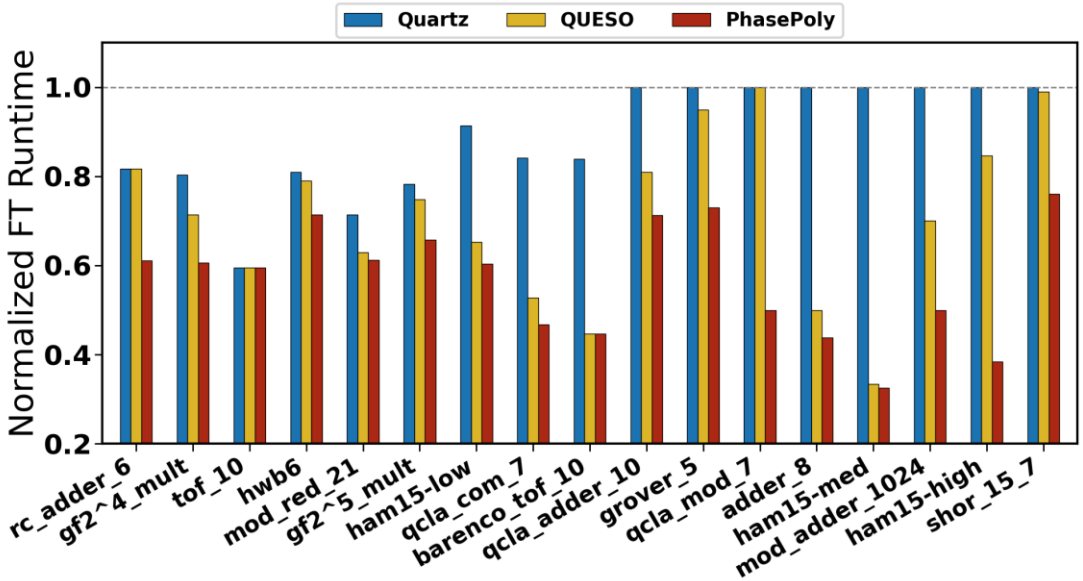
3.5 Hardware-Level Execution Improvements

- 2D planar coupling graphs (square grids), using Qiskit SABRE
- Most of time hardware mapping can amplify circuit optimization gains, depth reductions of 22.47% become 28.35% in physical circuits. More evident in large circuits benchmark.
- Sometimes it doesn't gf2^x_mult, qaoa



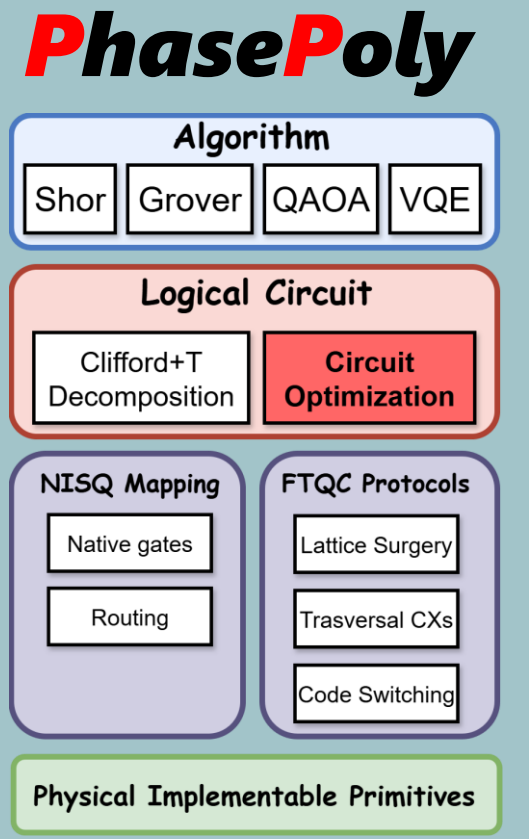
3.6 Fault-Tolerant Compilation Benefits

- Surface-code, nearest-neighbour architecture, using Microsoft Azure Resource Estimator
- Quartz, QUESO, and PhasePoly achieve average reductions of 11.99%, 31.80%, and 44.62%, respectively, with PhasePoly providing the largest improvement.
- CNOT cost is no-trivial because of magic state cultivation[1] and modern resource estimator [2]



- Integration with downstream FTQC compilation protocols will be our future work

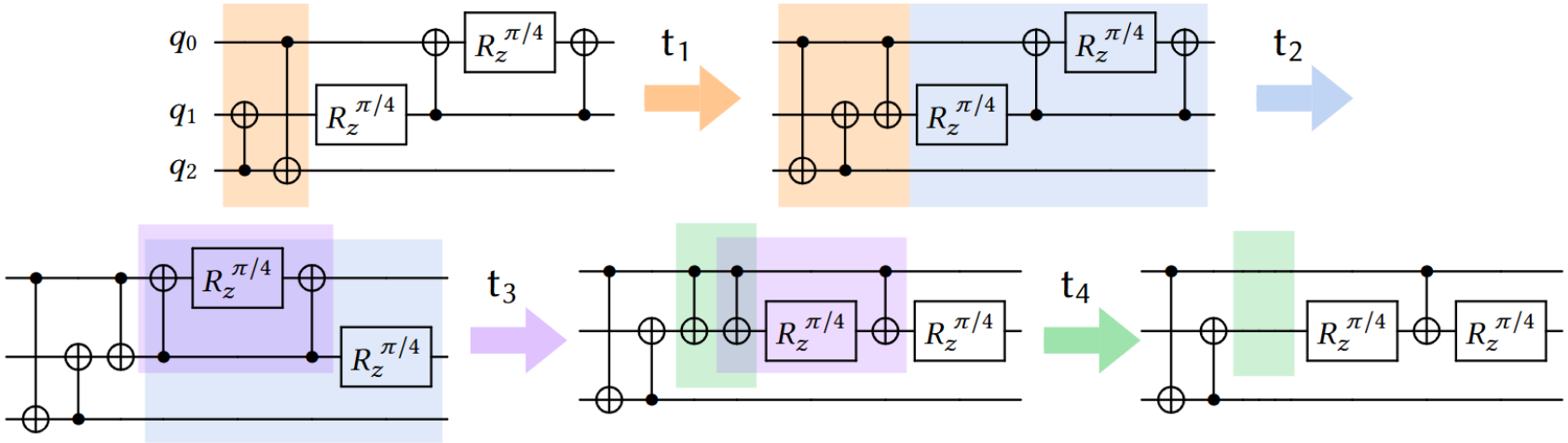
[1] Gidney, Craig, Noah Shutty, and Cody Jones. "Magic state cultivation: growing T states as cheap as CNOT gates." arXiv preprint arXiv:2409.17595 (2024).
[2] Huggins, William J., et al. "The FLuid Allocation of Surface code Qubits (FLASQ) cost model for early fault-tolerant quantum algorithms." arXiv preprint arXiv:2511.08508 (2025).



1. Background and Motivation
2. PhasePoly's Compilation Techniques
3. Evaluation and Insights
4. Future Work and Conclusion

4.1 Better Heuristic Design

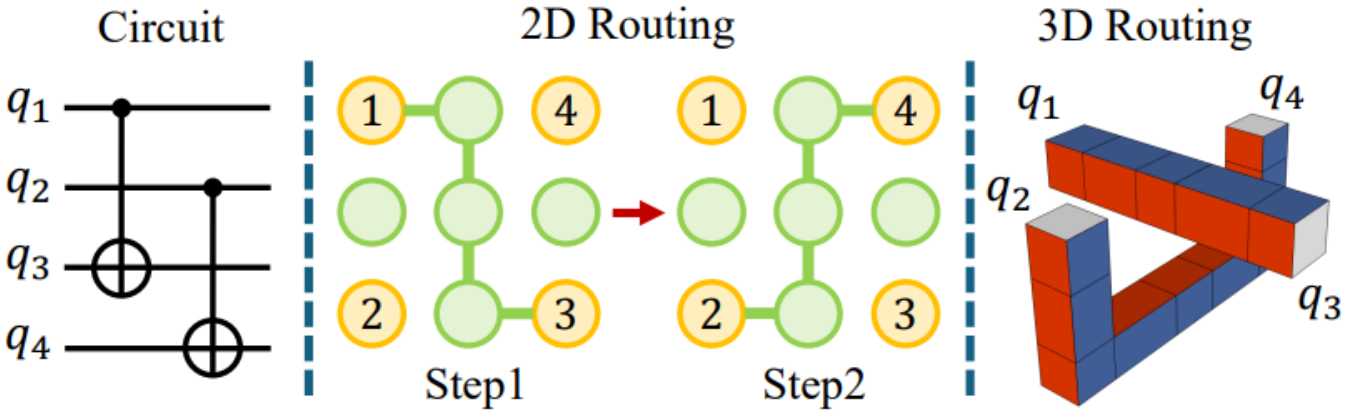
- The current heuristic design focuses on identifying local reduction opportunities at every step
- The extensive search required for cross-block optimization means that the algorithm doesn't always easily identify optimization opportunities.



Source: Yang, Ganxiang, et al. "Equality Saturation for Quantum Circuit Optimization." Proceedings of the ACM on Programming Languages 10.PLDI (2026): 174-199.

4.2 Cross-layer FTQC-friendly Optimizer

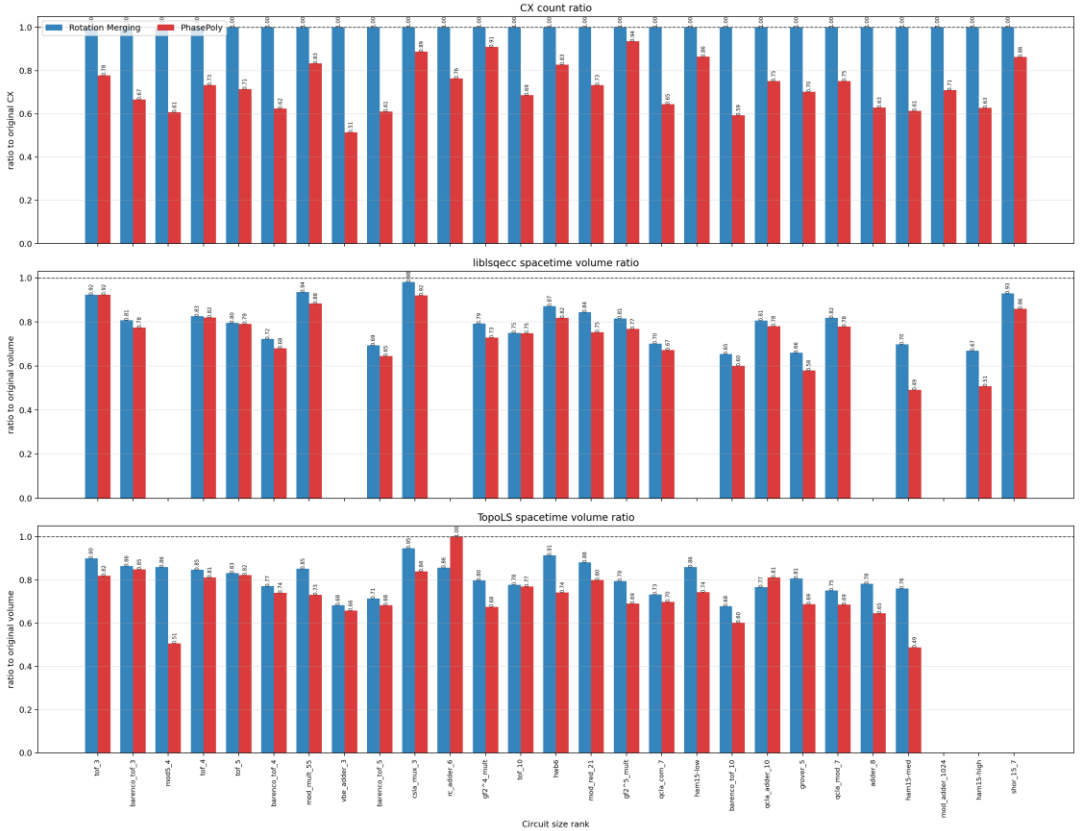
- The current end-to-end compilation framework for logical circuits to physical implementable primitives is not yet mature, but it is developing very rapidly.
- Liblsqecc: 2D lattice surgery routing, improved on Litinski's GoSC
- TQEC/Topols: 3D topological lattice surgery routing



Source: Zhou, Junyu, et al. "TopoLS: Lattice Surgery via Topological Program Transformations." arXiv preprint arXiv:2601.23109 (2026Compilation).

4.2 Cross-layer Optimizer

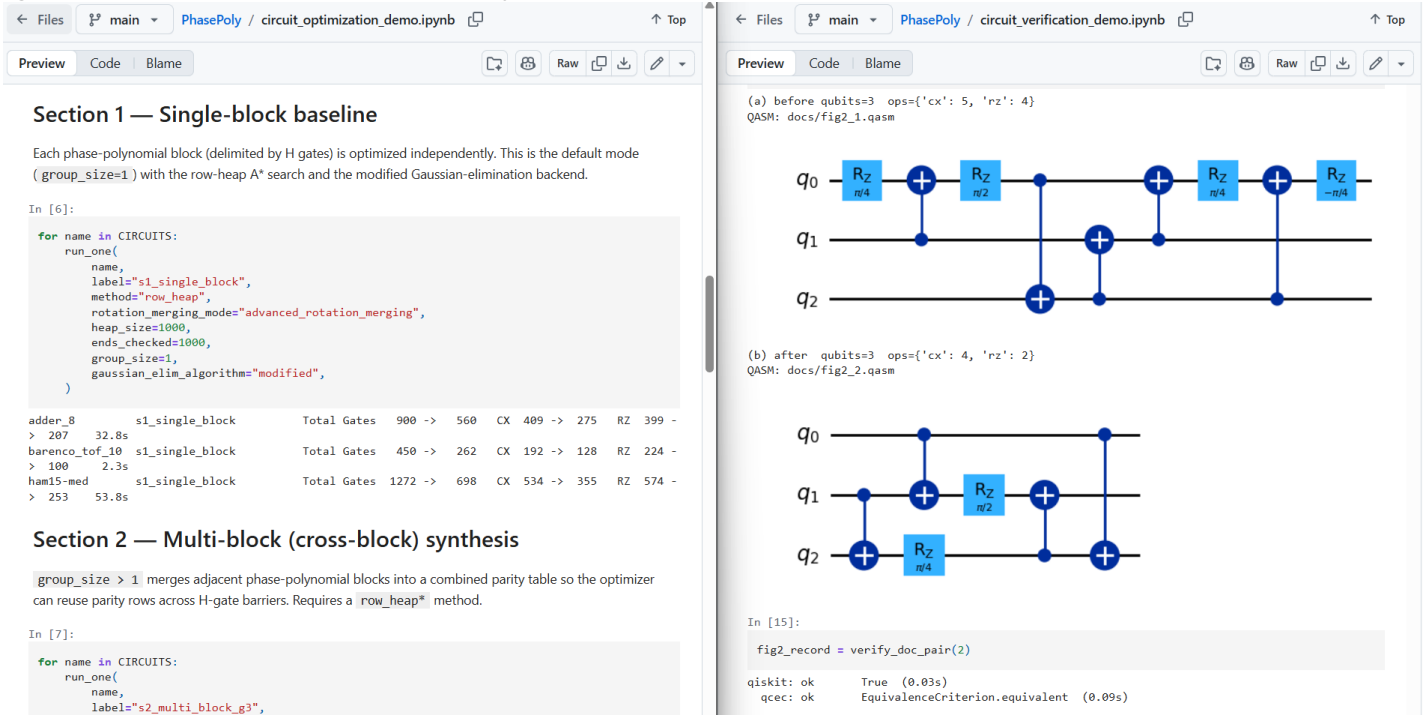
- Rotation gate is important, but CNOT cost is also non-trivial!
- Rotation merging's reduction: 20.45%(Liblsqecc), 19.06%(TopoLS)
- PhasePoly's reduction: 26.02%(Liblsqecc), 27.07%(TopoLS)



RUTGERS | Yonsei University | Carnegie Mellon University

4.3 Better Software and Implementation

- Phase-polynomial optimization should not be just an auxiliary tool inside rewriting optimizers.
- https://github.com/ruadapt/PhasePoly



Files main PhasePoly / circuit_optimization_demo.ipynb Top

Preview Code Blame Raw

Section 1 — Single-block baseline

Each phase-polynomial block (delimited by H gates) is optimized independently. This is the default mode (`group_size=1`) with the row-heap A* search and the modified Gaussian-elimination backend.

In [6]:

```
for name in CIRCUITS:
    run_one(
        name,
        label="s1_single_block",
        method="row_heap",
        rotation_merging_mode="advanced_rotation_merging",
        heap_size=1000,
        ends_checked=1000,
        group_size=1,
        gaussian_elim_algorithm="modified",
    )
```

```
adder_8         s1_single_block          Total Gates   900 ->   560   CX  409 ->  275   RZ  399 -
> 207    32.8s
barenco_tof_10  s1_single_block          Total Gates   450 ->   262   CX  192 ->  128   RZ  224 -
> 100     2.3s
ham15-med       s1_single_block          Total Gates  1272 ->   698   CX  534 ->  355   RZ  574 -
> 253    53.8s
```

Section 2 — Multi-block (cross-block) synthesis

`group_size > 1` merges adjacent phase-polynomial blocks into a combined parity table so the optimizer can reuse parity rows across H-gate barriers. Requires a `row_heap*` method.

In [7]:

```
for name in CIRCUITS:
    run_one(
        name,
        label="s2_multi_block_g3",
```

Files main PhasePoly / circuit_verification_demo.ipynb Top

Preview Code Blame Raw

```
(a) before qubits=3  ops={'cx': 5, 'rz': 4}
QASM: docs/fig2_1.qasm
```

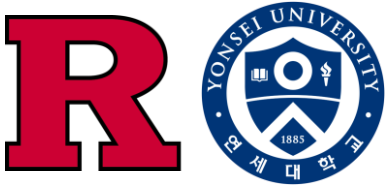


```
(b) after  qubits=3  ops={'cx': 4, 'rz': 2}
QASM: docs/fig2_2.qasm
```


In [15]:

```
fig2_record = verify_doc_pair(2)
```

```
qiskit: ok       True  (0.03s)
  qcec: ok       EquivalenceCriterion.equivalent  (0.09s)
```



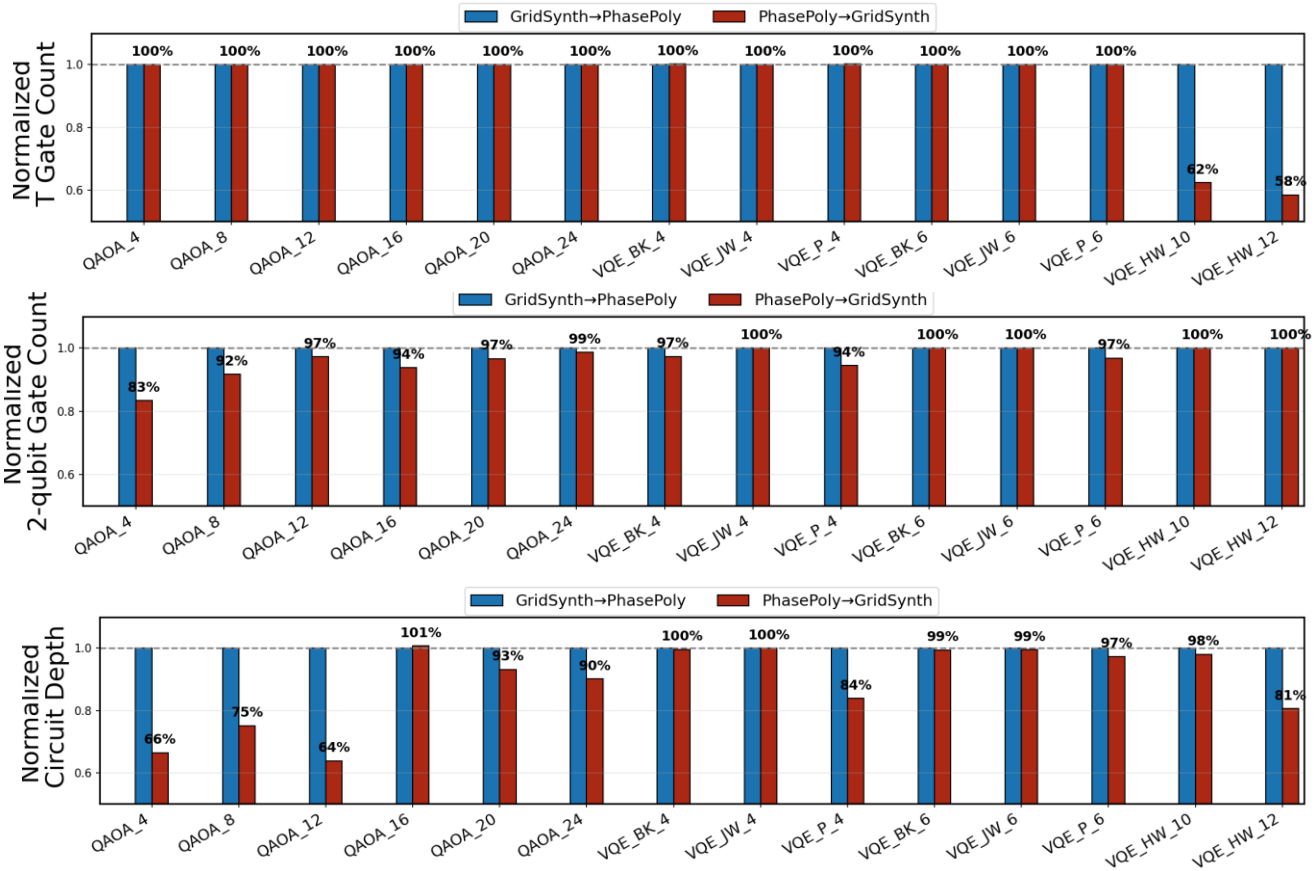
Carnegie Mellon University

Thank You for Your Time !

Q & A

zihan.chen.cs@rutgers.edu

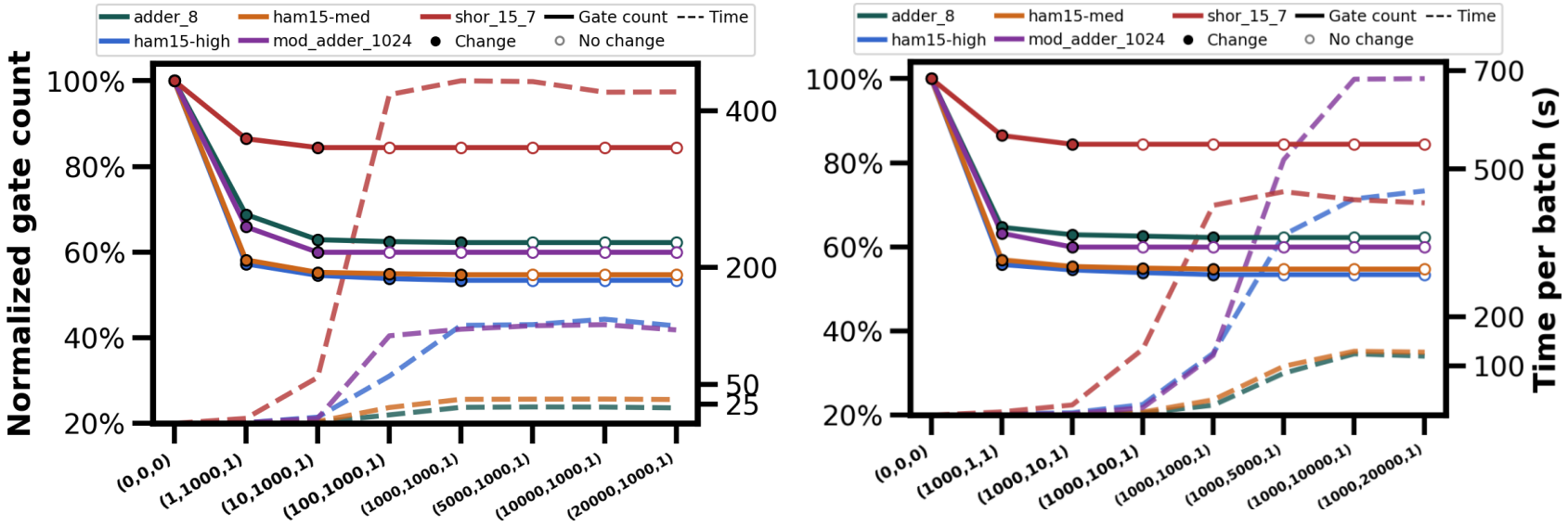
B1 Fault-Tolerant Compilation Integration



- Running PhasePoly before GridSynth produces the lowest depth on most circuits.
- Because large {CNOT,Rz} regions are simplified before GridSynth introduces additional H gates that split phase-polynomial blocks and limit rotation-merging opportunities.
- Optimization opportunities also depends on circuit structure a lot in VQE

B2 Parameter Sensitivity

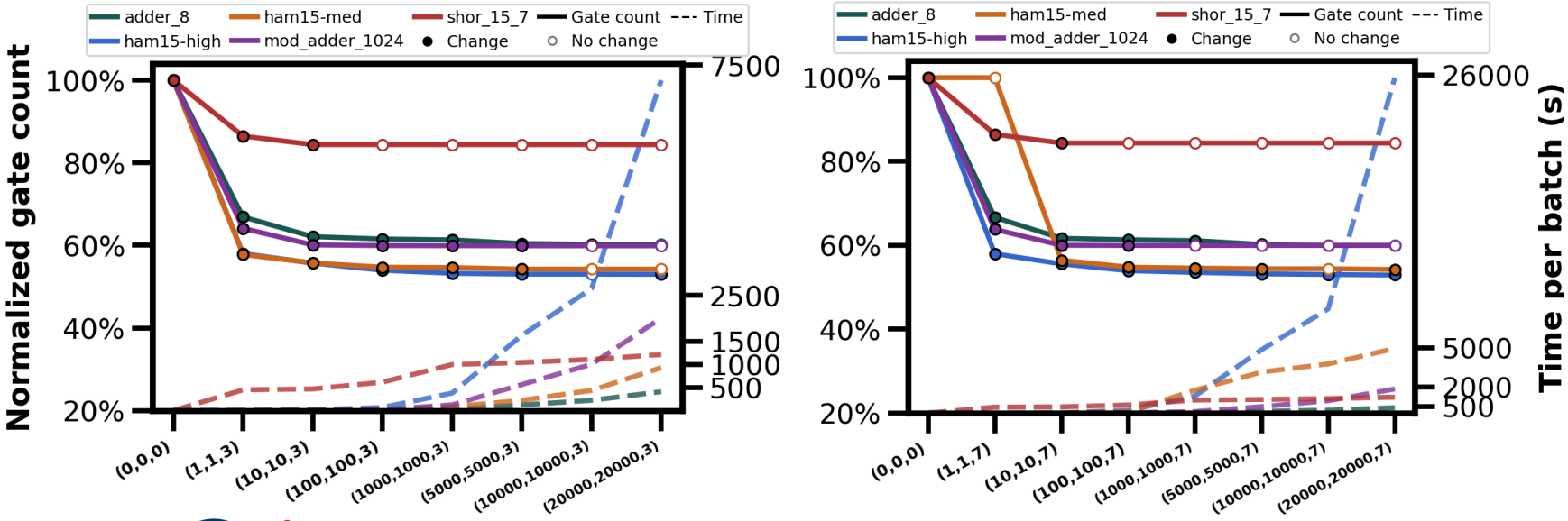
- Moderate bounds (Q=P=1000) consistently achieve near-highest reductions, while larger search spaces mainly increase compilation time with diminishing returns.



RUTGERS
Carnegie Mellon University

B2 Parameter Sensitivity

- Jointly scaling (Q,P,G) confirms the same robustness trend.
- Results for G ∈ {1,5} follow the same pattern.



RUTGERS

Carnegie Mellon University

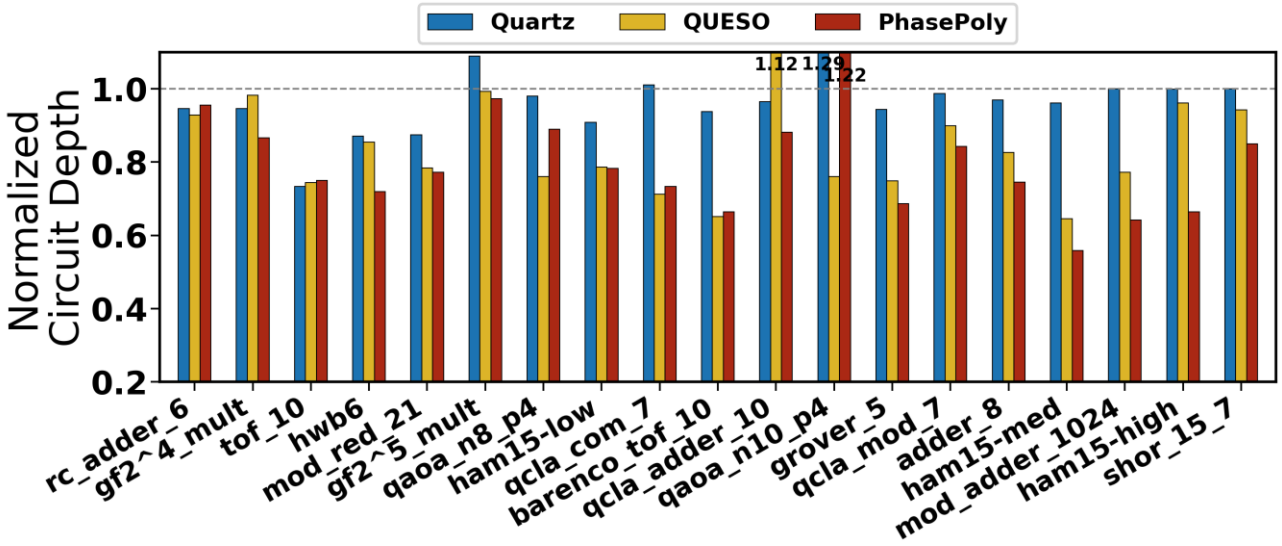
B3 Incremental Block Merging Strategy

- Overall, Incremental Block Merging offers a robust approach that captures large gains while avoiding over-merging regressions.

Circuit	barenco_tof_10		adder_8		ham15_med	
	# Gates	# CXs	# Gates	# CXs	# Gates	# CXs
Org.	450	192	900	409	1272	534
Group 1	262	128	557	274	696	353
Group 3	**248**	**114**	542	259	695	352
Group 5	**248**	**114**	**540**	**257**	693	350
Group 7	**248**	**114**	**540**	**257**	694	351
Incremental	**248**	**114**	542	259	**656**	**325**

B4 Better Circuit Depth

- Quartz reduces depth by 13.26%, QUESO by 19.64%, and PhasePoly achieves the largest reduction of 22.47%, corresponding to a 1.14–1.69× improvement.
- The gap widens on large circuit families, where Quartz reduces depth by 4.03%, QUESO by 14.45%, and PhasePoly by 40.91%, corresponding to a 2.83–10.15× improvement.



RUTGERS | Carnegie Mellon University