AC/DC: Automated Compilation for Dynamic Circuits

Ed Younis

BQSKit

MACH-|Q

U.S. DEPARTMENT OF ENERGY
Office of Science

BERKELEY LAB
Bringing Science Solutions to the World

BQSKit's Amazing Multi-Institutional Team

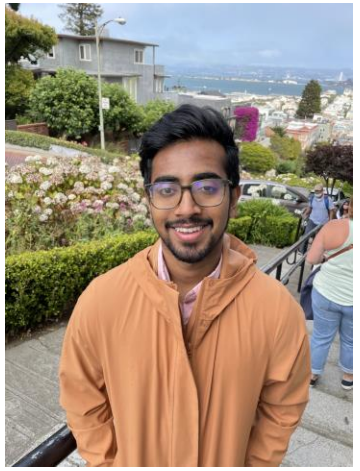
Costin Iancu

Ed Younis

Bert de Jong

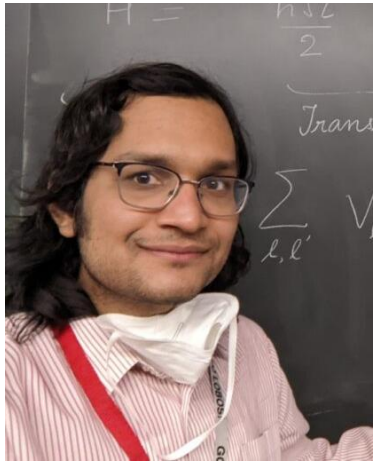
Efekan Kökcü

Mathias Weiden



Justin Kalloor

Siyuan Niu



Anupam Mitra

Ji Liu

Roel van Beeumen

Marc Davis

Past Members:

- Xin-Chuan Wu (Intel)
- Ethan Smith (NVIDIA)
- Tirthak Patel (Rice)
- Lindsay Bassman (CNR Nano)
- Aaron Szasz (Google)

Berkeley Quantum Synthesis Toolkit (BQSKit)

Powerful and portable quantum compiler framework built on numerical instantiation

Berkeley Quantum Synthesis Toolkit (BQSKit) [bis • kit]

Powerful and Portable Quantum Compiler Framework

- Gracefully handles every gate set and architecture
- Specialized pipelines for widely-used chips
- Generates resource efficient circuits (width, depth, timing, etc)
- Error mitigation
- Enables effective algorithm and hardware design exploration

- 20+ papers (3 IEEE QCE best papers: 2020, 2021, 2022)
- 500K+ downloads

BQSKit

bqskit.lbl.gov

Users:

- LANL
- Sandia
- ANL
- ORNL
- LLNL
- U Penn
- U Del
- U Chicago
- U Kansas
- U Tokyo
- NCSU
- Duke
- CUNY
- U Wisc
- Quantinuum
- Infleqtion
- AWS
- Microsoft
- NVIDIA
- UnitaryFund
- HSBC

Turn-key Functionality that Works Everywhere

```
pip install bqskit
```

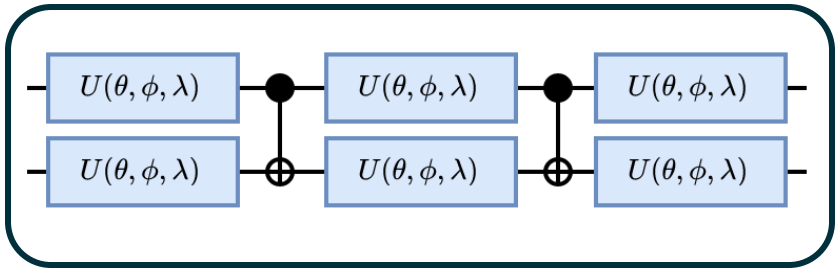
Python 3.8+ on Windows, Mac, Linux
(from laptop to supercomputer)

```
import bqskit
out_circuit = bqskit.compile(circuit)
```

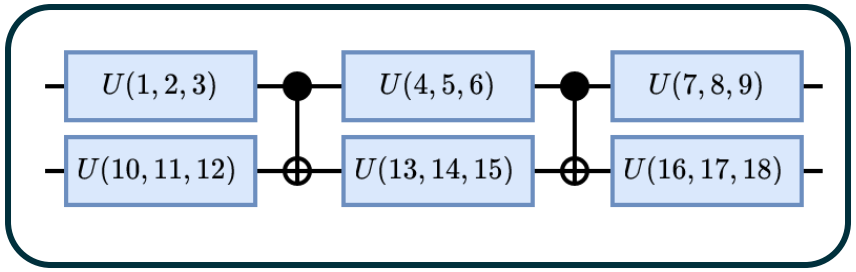
Any gate set or architecture in or out
qubits, qudits, states, many-states, unitaries, and more
(from experiment to real hardware)

Parameterized Circuit Instantiation: A Powerful Primitive

Parameterized Circuit



Instantiated Circuit



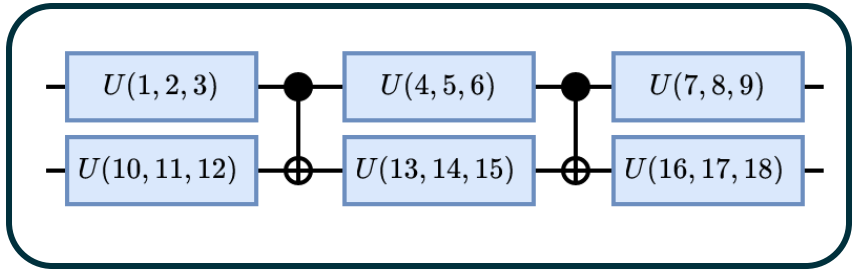
Parameterized Circuit Instantiation

Given a parameterized quantum circuit $C : \mathbb{R}^k \mapsto U(N)$ and a target unitary $V \in U(N)$, solve for

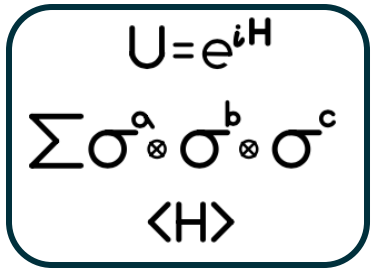
$$\underset{\alpha}{\operatorname{argmax}}\ tr(V^\dagger C(\alpha))$$

Instantiation Enables Function and Structure Separation

Instantiated Circuit



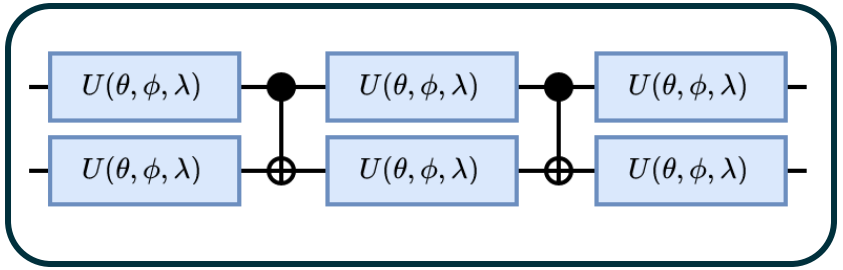
Mathematical Representations



Function

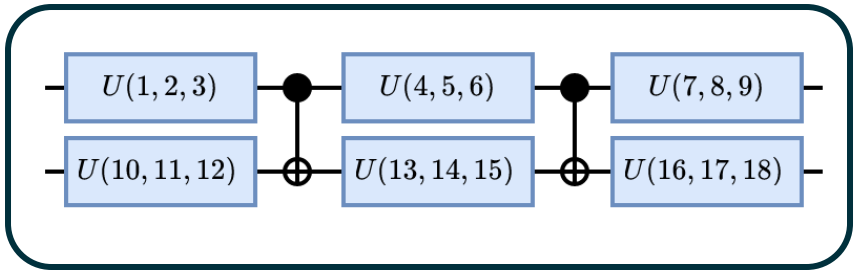
Can isolate and vary circuit function and structure

Parameterized Circuit

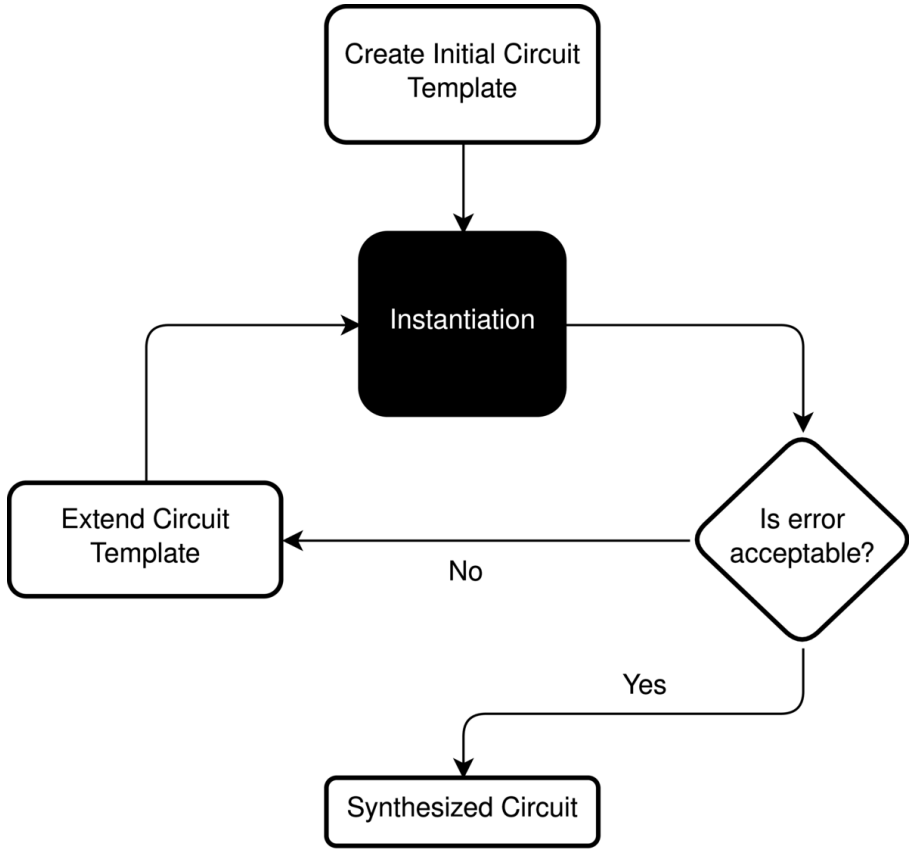


Structure

Instantiated Circuit



Instantiation is the Core of Bottom-Up Synthesis

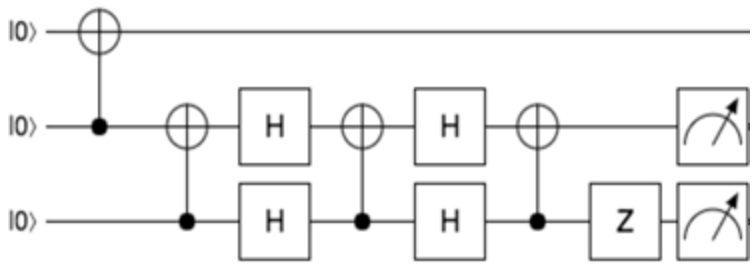


The standard bottom-up loop is to create an initial circuit template guess then continue to instantiate and extend it until a solution is found

Unitary

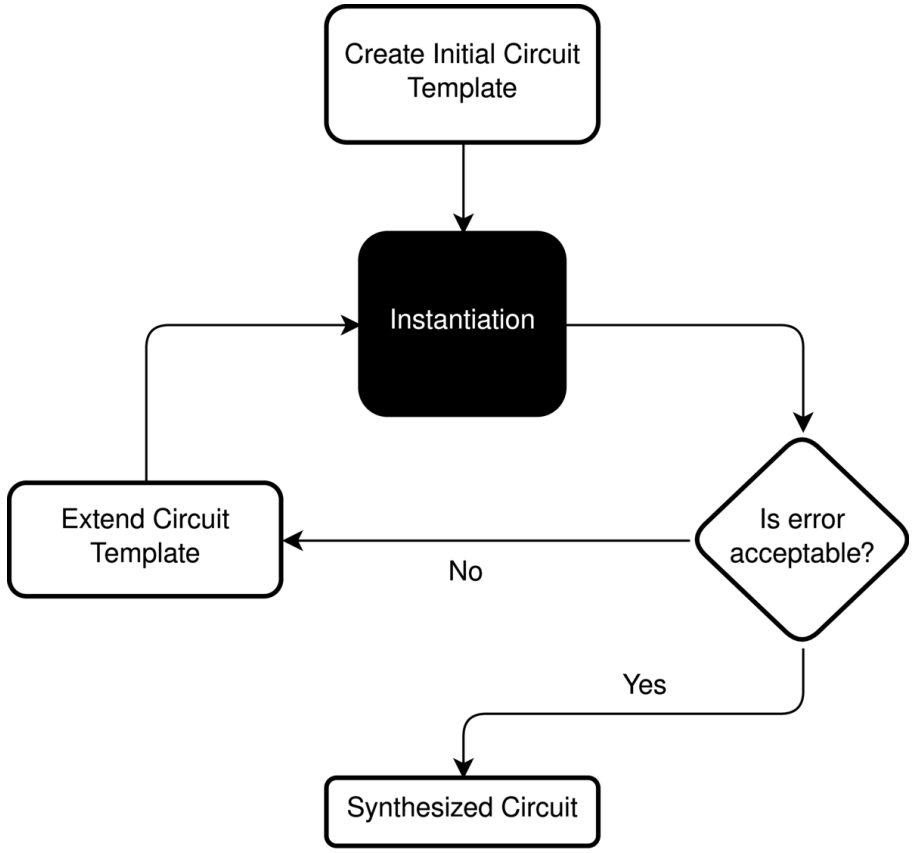
$$\frac{1}{\sqrt{2^3}}\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\
1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\
1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \\
1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} & \omega^{28} \\
1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} & \omega^{35} \\
1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} & \omega^{42} \\
1 & \omega^7 & \omega^{14} & \omega^{21} & \omega^{28} & \omega^{35} & \omega^{42} & \omega^{49}
\end{bmatrix}$$

Circuit



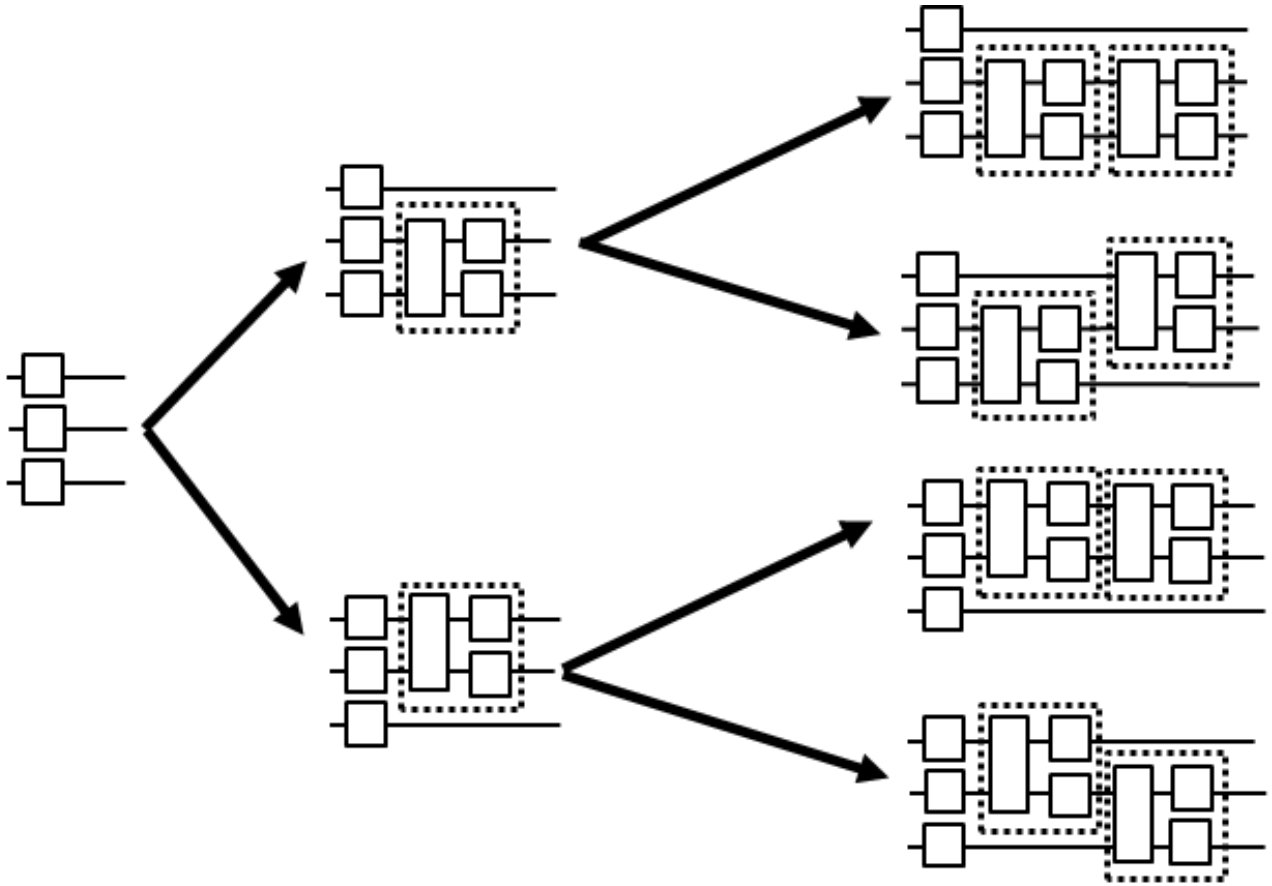
E. Younis et al., "QFast: Conflating Search and Numerical Optimization for Scalable Quantum Circuit Synthesis." 2021 *IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2021. (Best Paper Award)

Instantiation is the Core of Bottom-Up Synthesis



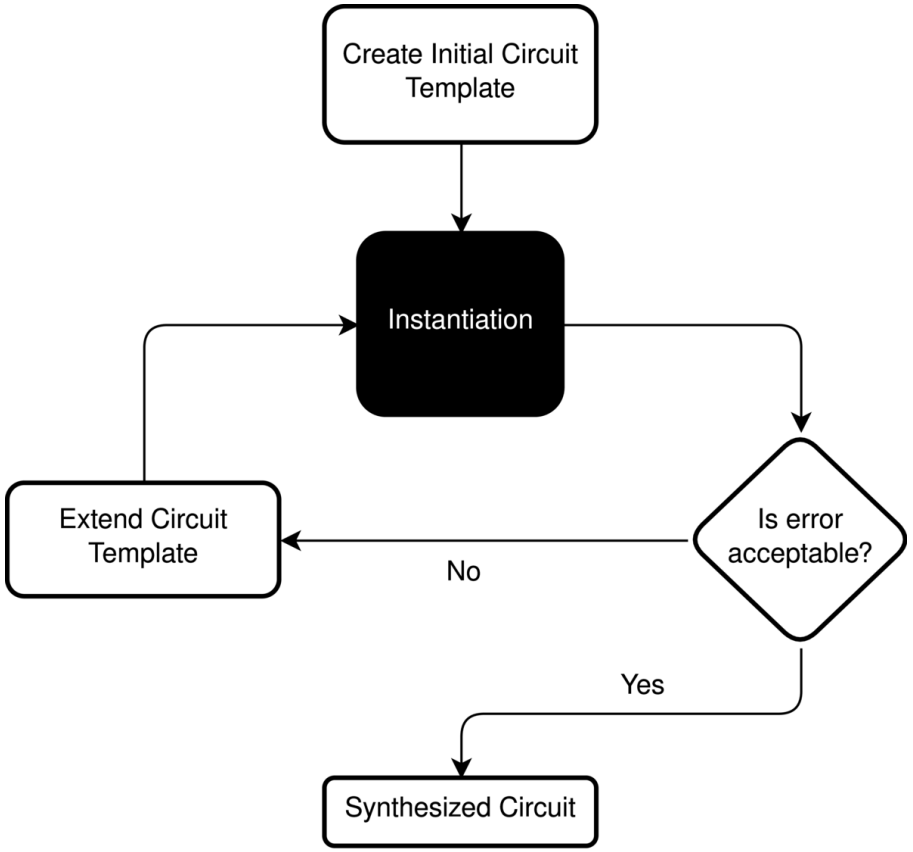
The standard bottom-up loop is to create an initial circuit template guess then continue to instantiate and extend it until a solution is found

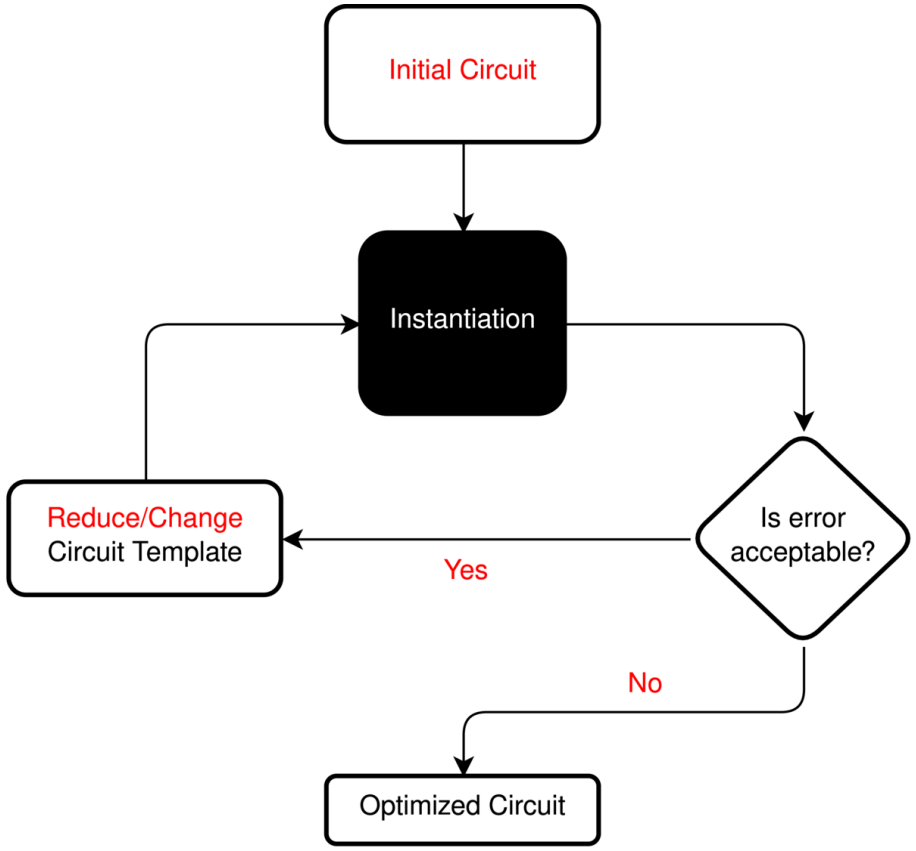
QSearch is a canonical bottom-up synthesis algorithm that produces optimal circuits.



M. Davis et al., “Towards Optimal Topology Aware Quantum Circuit Synthesis.” 2020 *IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2020. (Best Paper Award)

Instantiation-Based Circuit Transformation Flow

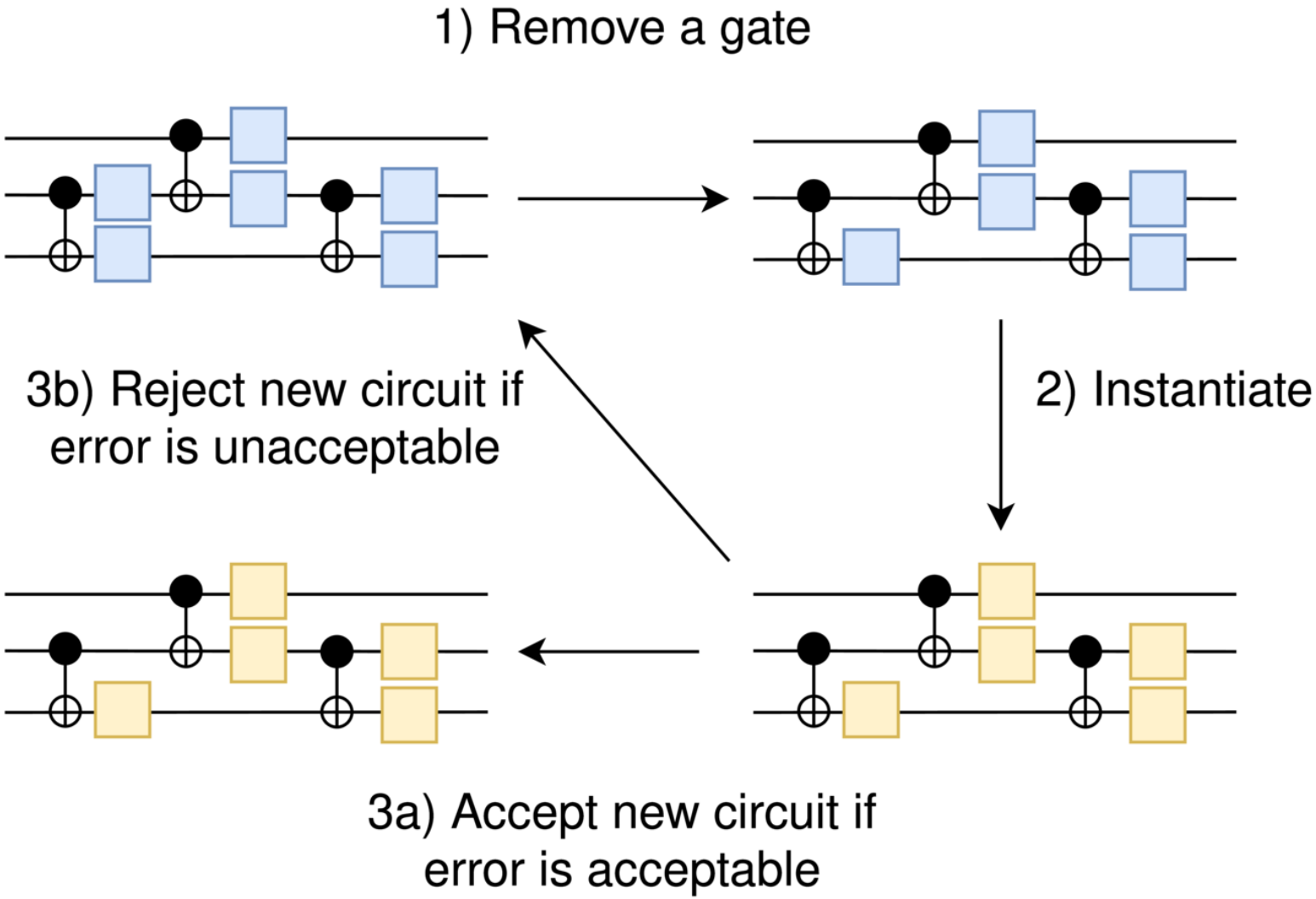




We have found Instantiation can also be used to transform or optimize a quantum circuit. Instead of extending a circuit template like in bottom-up synthesis, we reduce or transform a circuit template and continue until it is no longer successful or necessary.

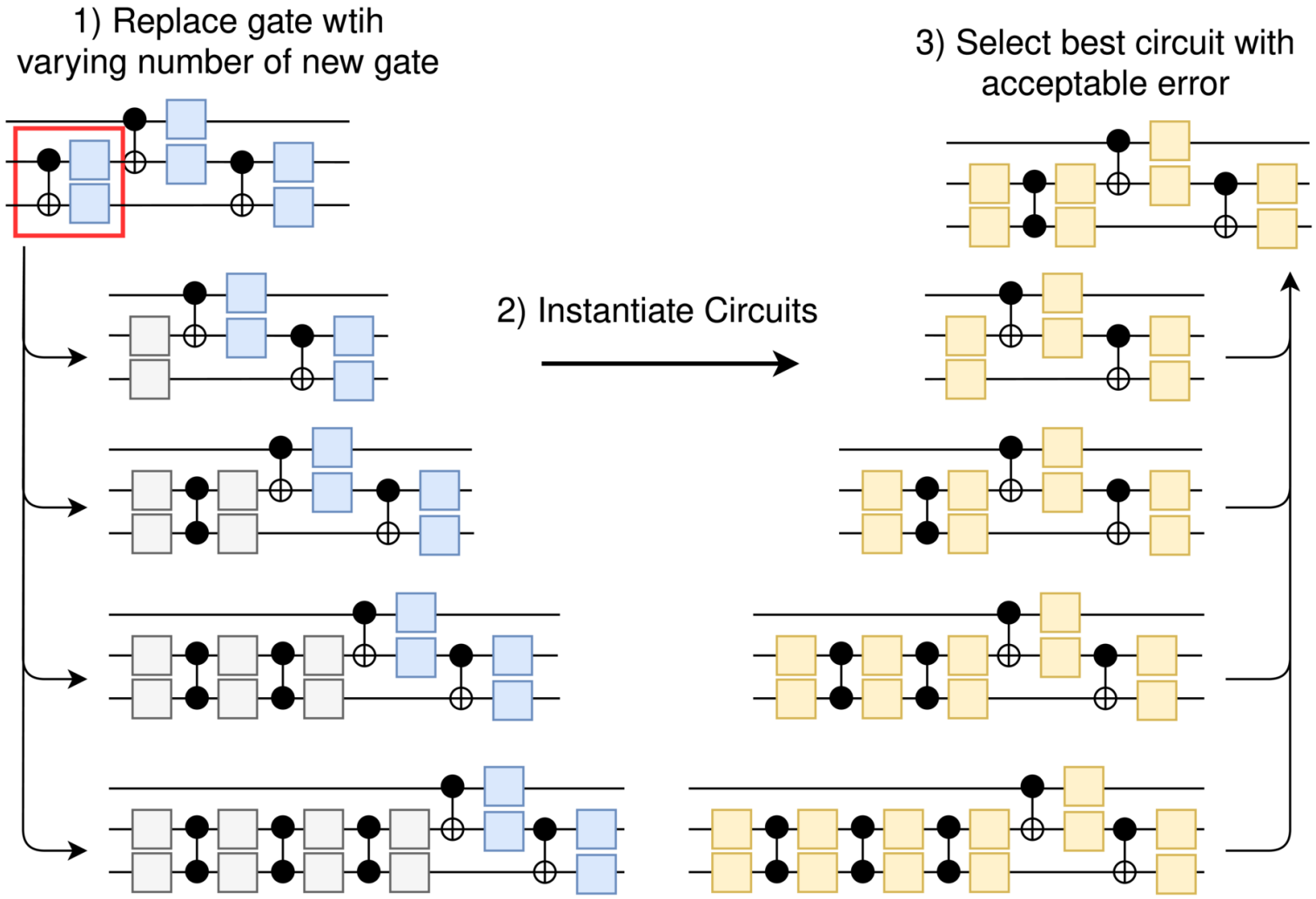
E. Younis et al., "Quantum Circuit Optimization and Transpilation via Parameterized Circuit Instantiation."
2022 IEEE International Conference on Quantum Computing and Engineering (QCE). IEEE, 2022

Gate Deletion is Straight Forward with Instantiation



- Suboptimal circuits can have gates removed and others reinstantiated to accommodate the loss
- Gate deletion + Candidate selection = Optimization algorithm
- Extremely portable and flexible
- Very effective due to global transformations

Instantiation can Retarget Circuits without Rules

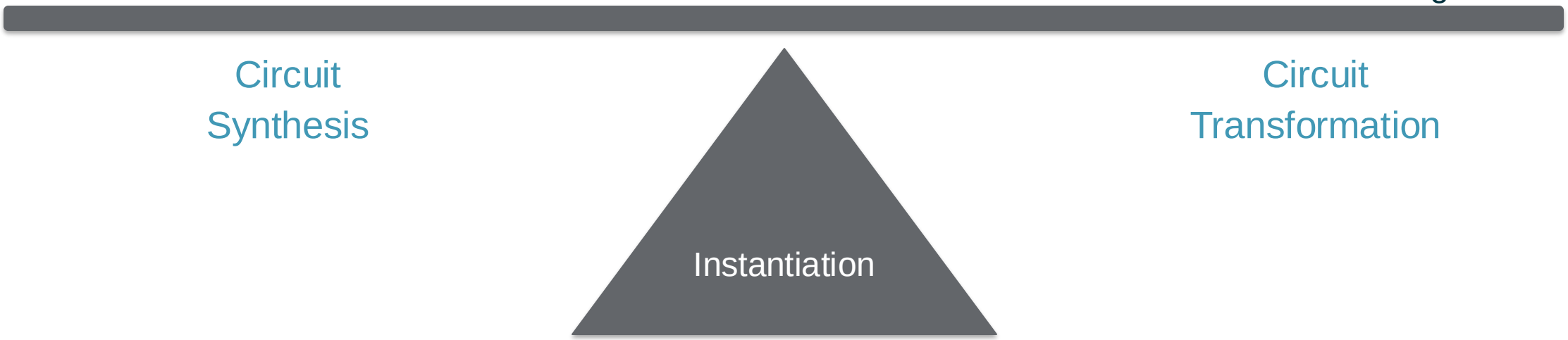


- Typically gate rebasing is done with fixed rules or analytic instantiation (KAK) that only consider local circuit changes
- Numerical instantiation-based rebasing is more effective, portable and flexible
- Works with gate sets contain multiple entangling gates

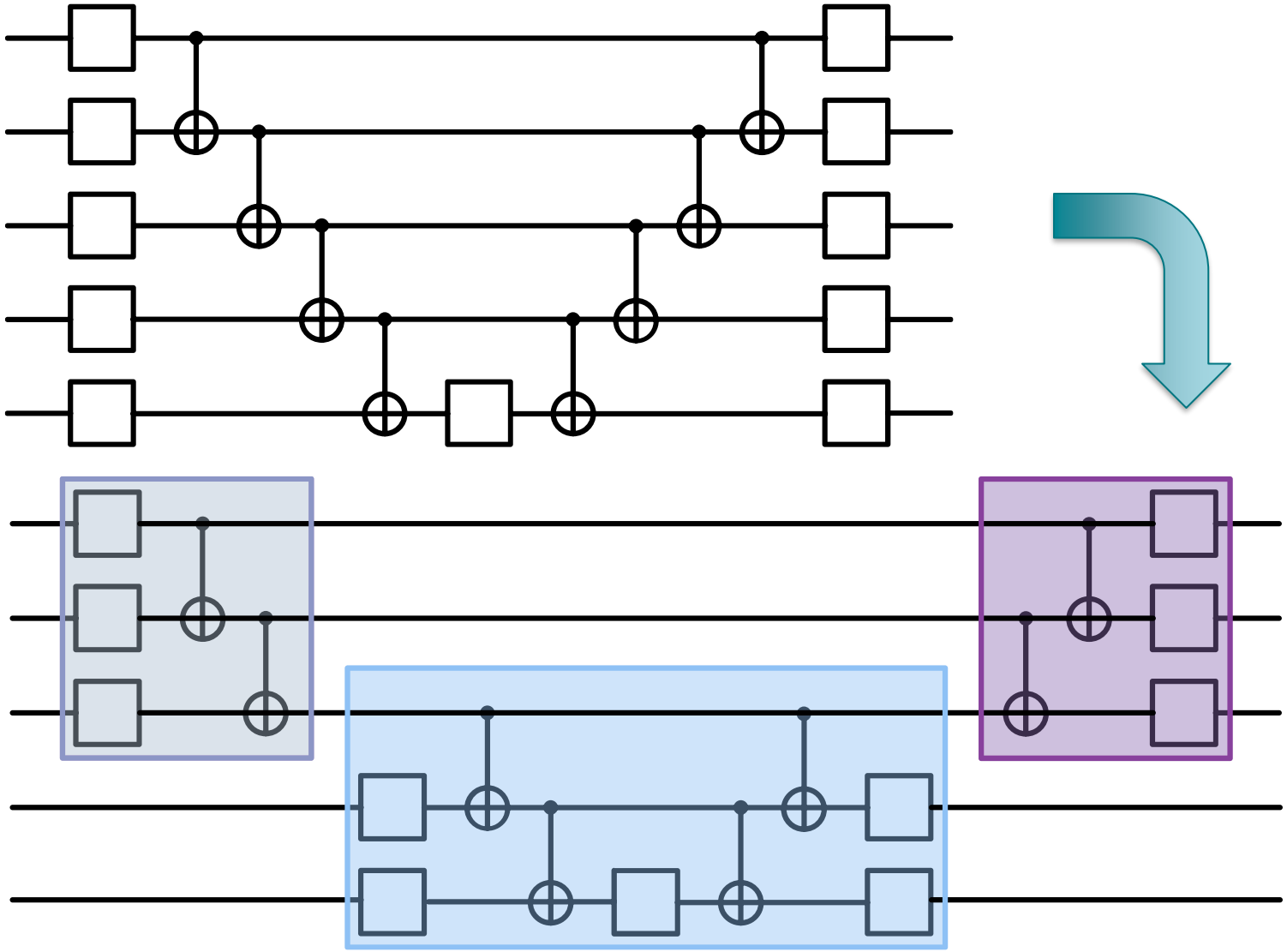
Instantiation-Based Algorithms in BQSKit

- QSweep – 1 qudit, pulse-optimal
- QSearch – 2-4 qudits, optimal
- PAS – 3-4 qudits, better-than-optimal + post-processing
- LEAP – 4-6 qudits
- QFAST – 5-8 qudits
- QPredict – 6-12 qudits

- Retargeting
- Gate Deletion
- Template Substitution
- Circuit Resizing



Partitioning Scales Instantiation to 1000s of qubits



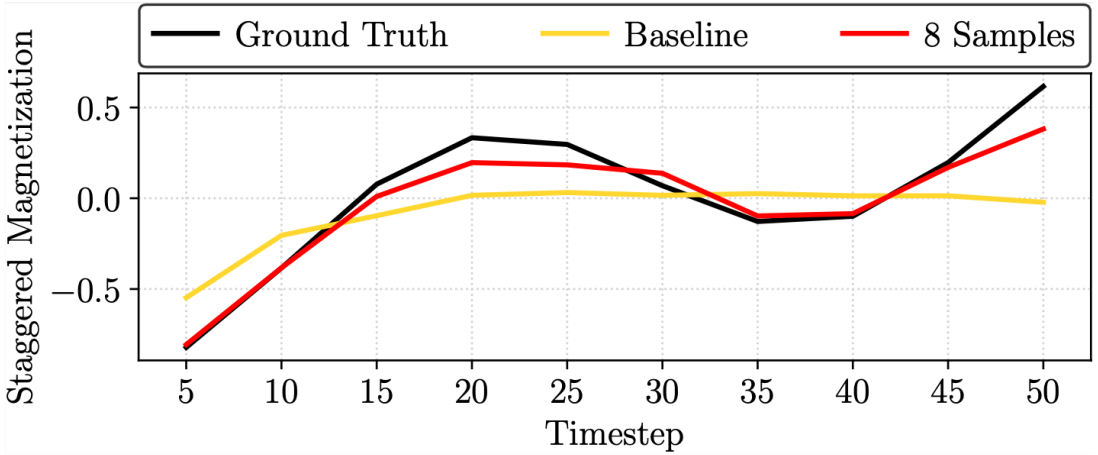
- Circuit on the left is partitioned into 3-qubit blocks below
- Partitioning overcomes the exponential scaling of numerical instantiation
- Restricts global view; can adjust block size for better performance or quality
- Creates an extremely parallel workflow

X.C. Wu et al. “Reoptimization of Quantum Circuits via Hierarchical Synthesis.” *2021 IEEE International Conference on Rebooting Computing (ICRC)*. IEEE, 2021.

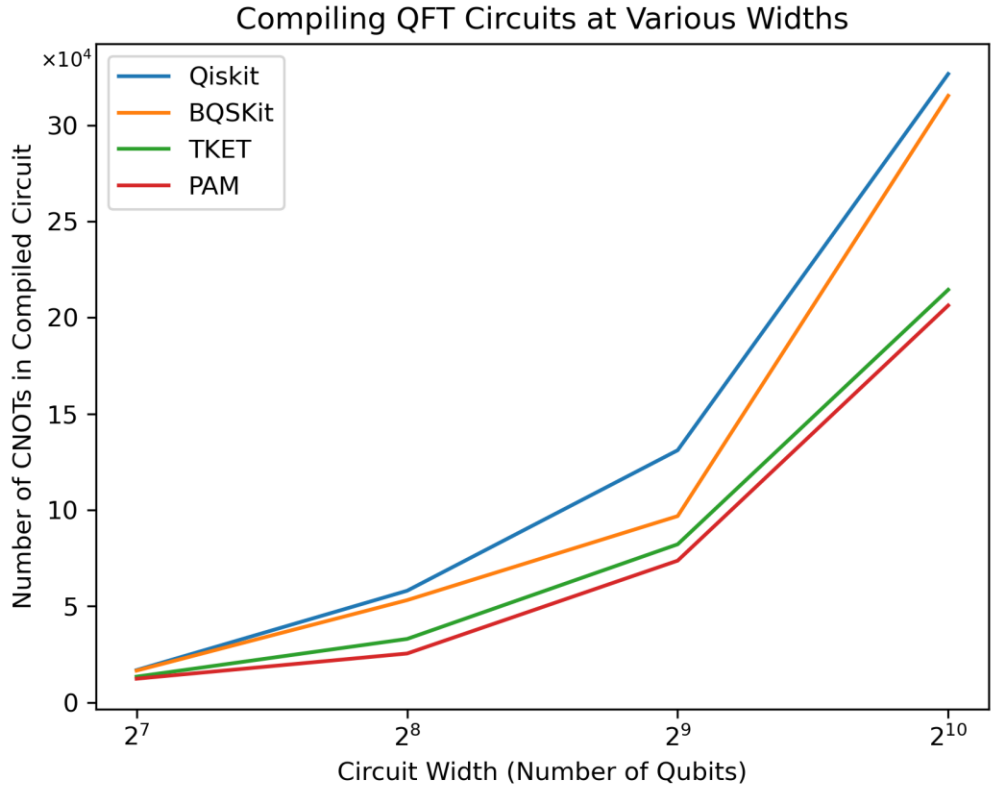
Instantiation-Based Workflows in BQSKit

Workflows

- Optimization (QGO, TopAS, Gate Deletion)
- Mapping/Routing (Generalized SABRE, PAM)
- Approximations (QuEST)
- Gate Set Retargeting
- Circuit Resizing

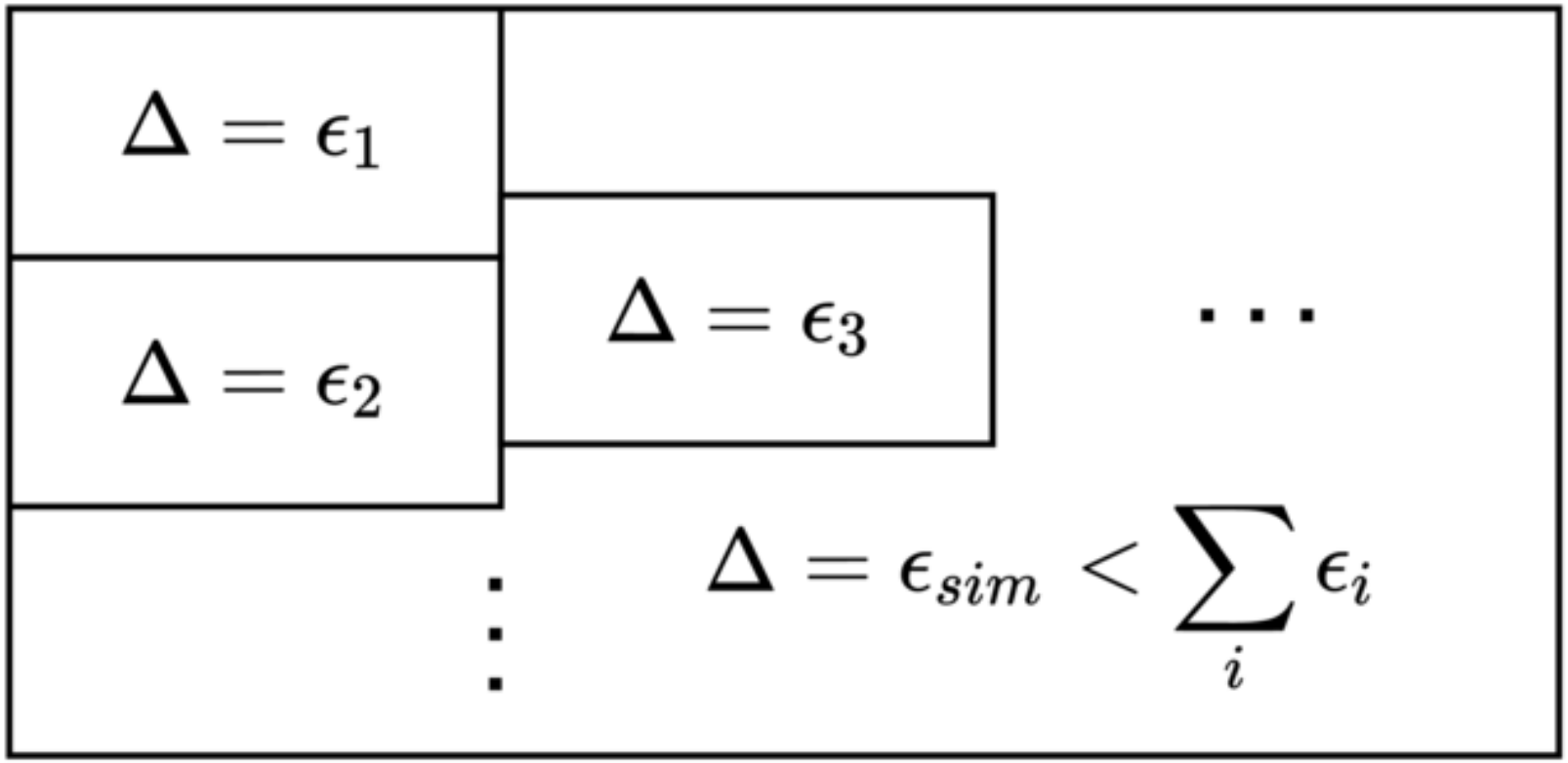


Approximations workflows better track ground truth on NISQ devices



PAM maps circuits with fewer final gates than commercial compilers

Push-button Compiler Verification



- Small circuits can have their error directly calculated
- Larger circuits can be first partitioned into large simulatable sections, then summing the section errors gives an upper bound on total error
- Unless specified, we target 10^{-8} to 10^{-10} instantiation verification error, most often it is less

Patel, Tirthak, et al. "Quest: systematically approximating quantum circuits for higher output fidelity." Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. 2022.

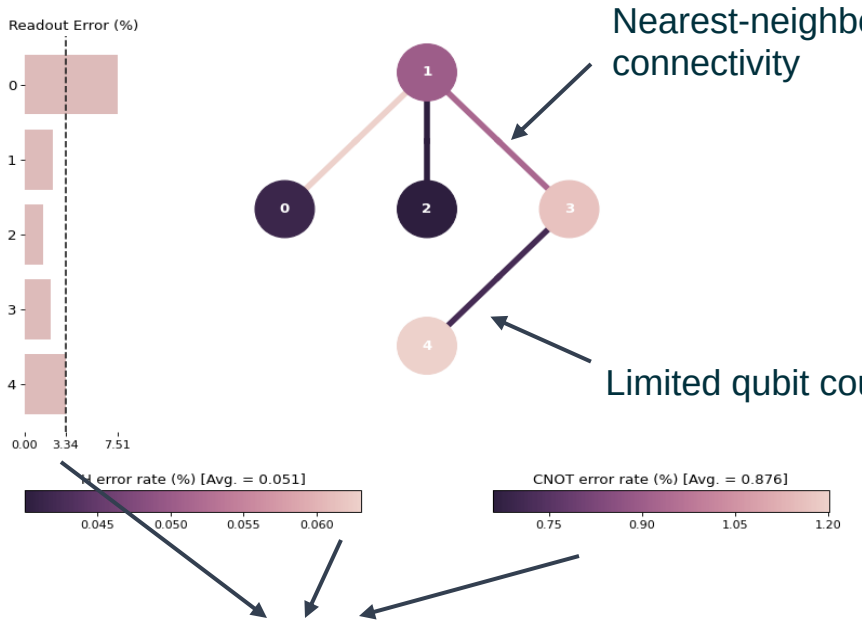
Effective Quantum Resource Optimization via Circuit Resizing in BQSKit

Discovering qubit reuse potential in previously impossible circuits via instantiation

Niu, Siyuan, et al. *Proceedings of the 61st ACM/IEEE Design Automation Conference*. 2024.

Quantum Computing Requires Resource Optimization

Why do we need resource optimization?

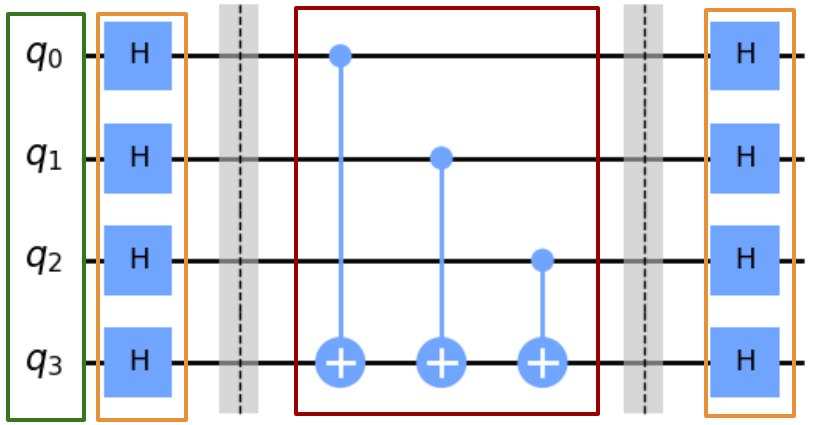


Nearest-neighbor connectivity

Limited qubit count

Different errors

How to characterize resource?

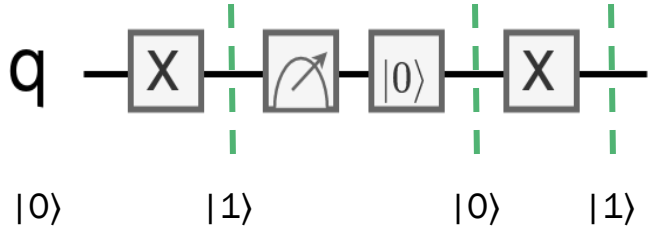


Depth

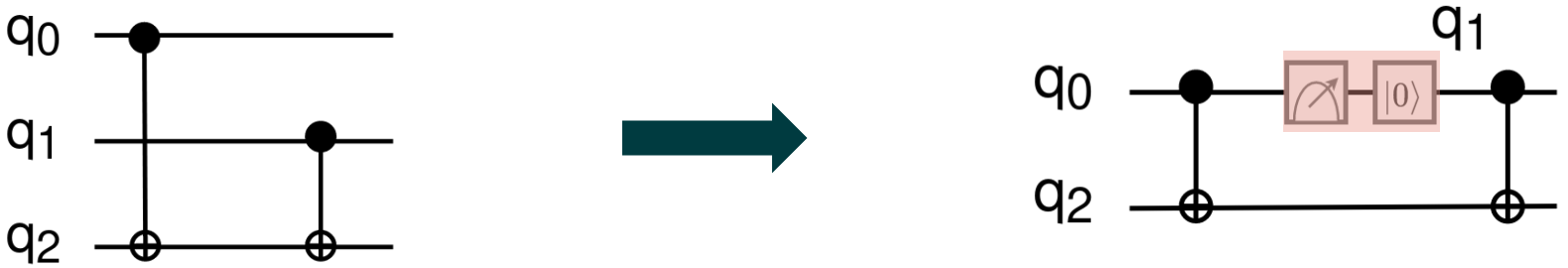
Qubits: 4
1-q gates: 8
2-q gates: 3
Depth: 5

Mid-circuit Measurement and Reset enables Resizing

- What is mid-circuit measurement and reset (MMR)?



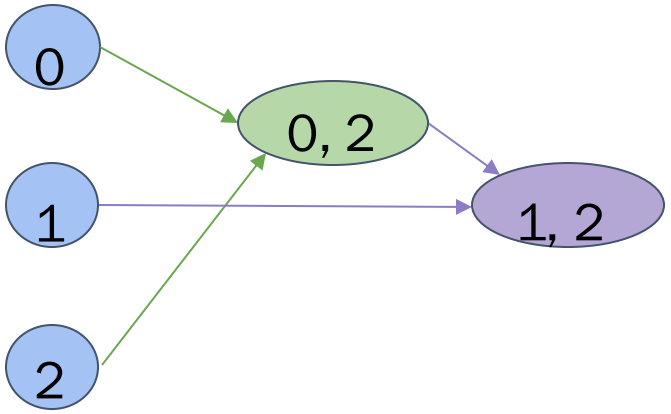
- How to use MMR for circuit resource optimization?



How to Realize Circuit Resizing?

- Toy example:

- By gate dependency graph!

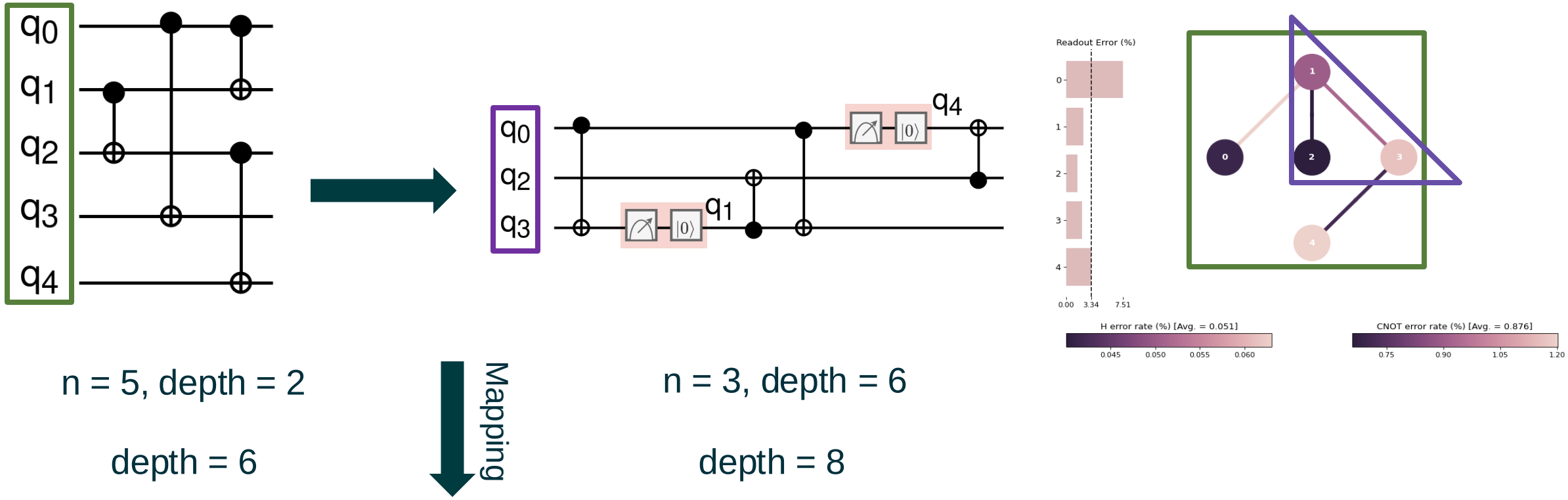


Qubit	Dependent	Independent
0	0, 2	1
1	0, 1, 2	None
2	0, 1, 2	None

- Not all the circuits are resizable by gate-dependency graph!

Gate-Dependency Based Resizing

- What can we get from circuit resizing?
  - Use qubits with higher fidelity
  - Reduce the mapping overhead

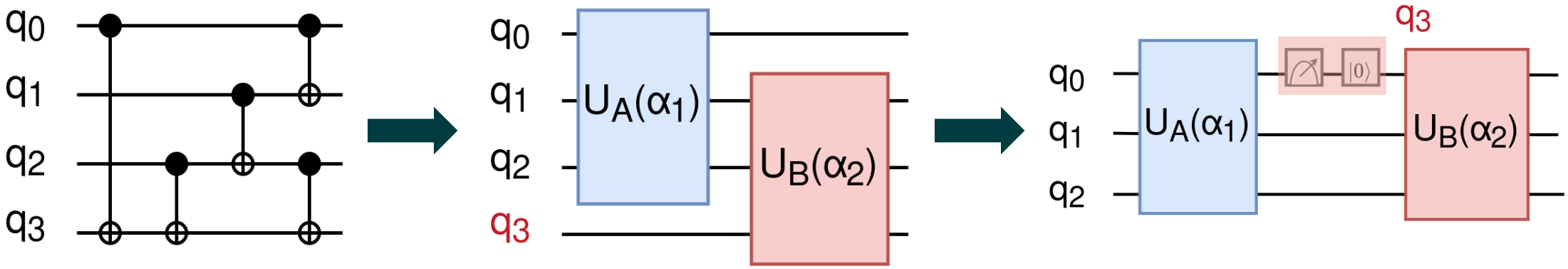


Resizable Checking via Instantiation

Can non-resizable circuit become resizable?

Resizable Checking via Instantiation

Can non-resizable circuit become resizable? - Yes, under constraints!



U_1

Not resizable based on gate dependency graph!

$U_2(\alpha_1, \alpha_2)$

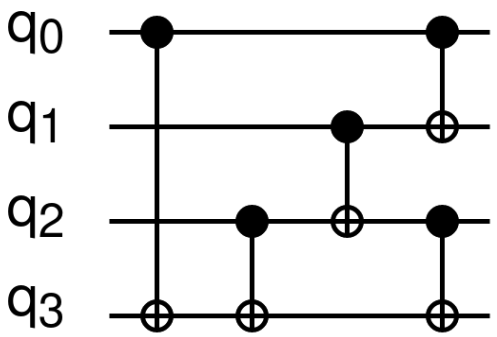
$\equiv$

$I \otimes U_B(\alpha_2)\ U_A(\alpha_1) \otimes I$

Check if q_i can be reused by q_j via **instantiation**

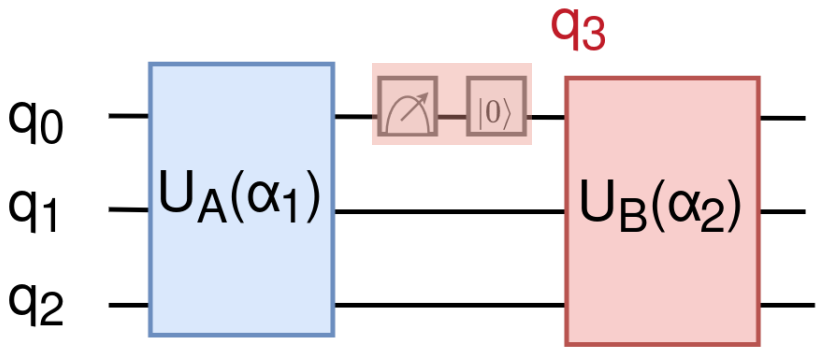
Instantiation-Based Resizing

U_1

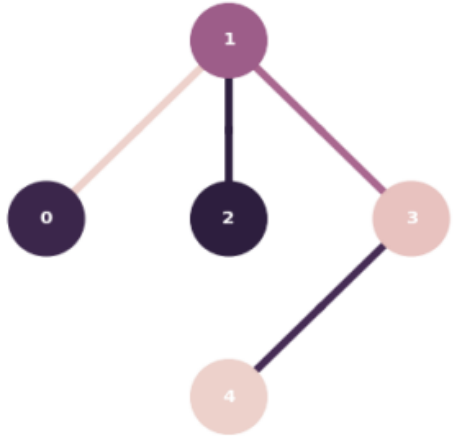


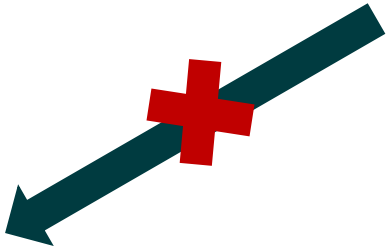
$U_1 = U_2$

$U_2(\alpha_1, \alpha_2)$



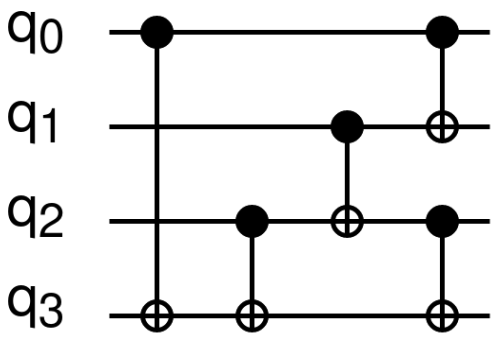
Only accepts two-qubit and single-qubit gates!



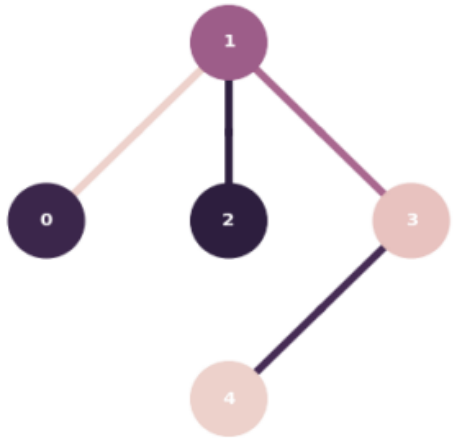


Instantiation-Based Resizing

U_1

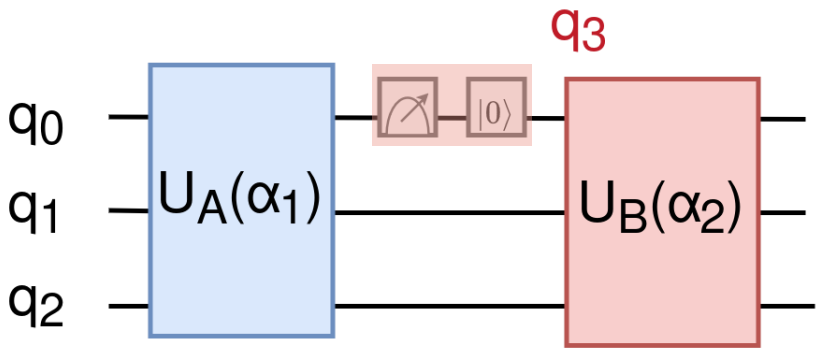


Only accepts two-qubit and single-qubit gates!



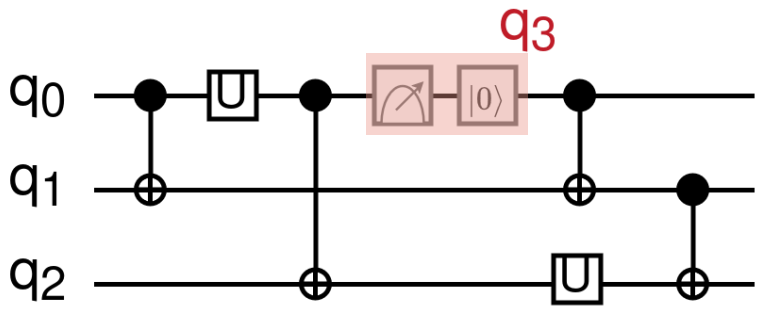
$U_1 = U_2$

$U_2\ (\alpha_1, \alpha_2)$



Unitary synthesis (search-based)

U_3

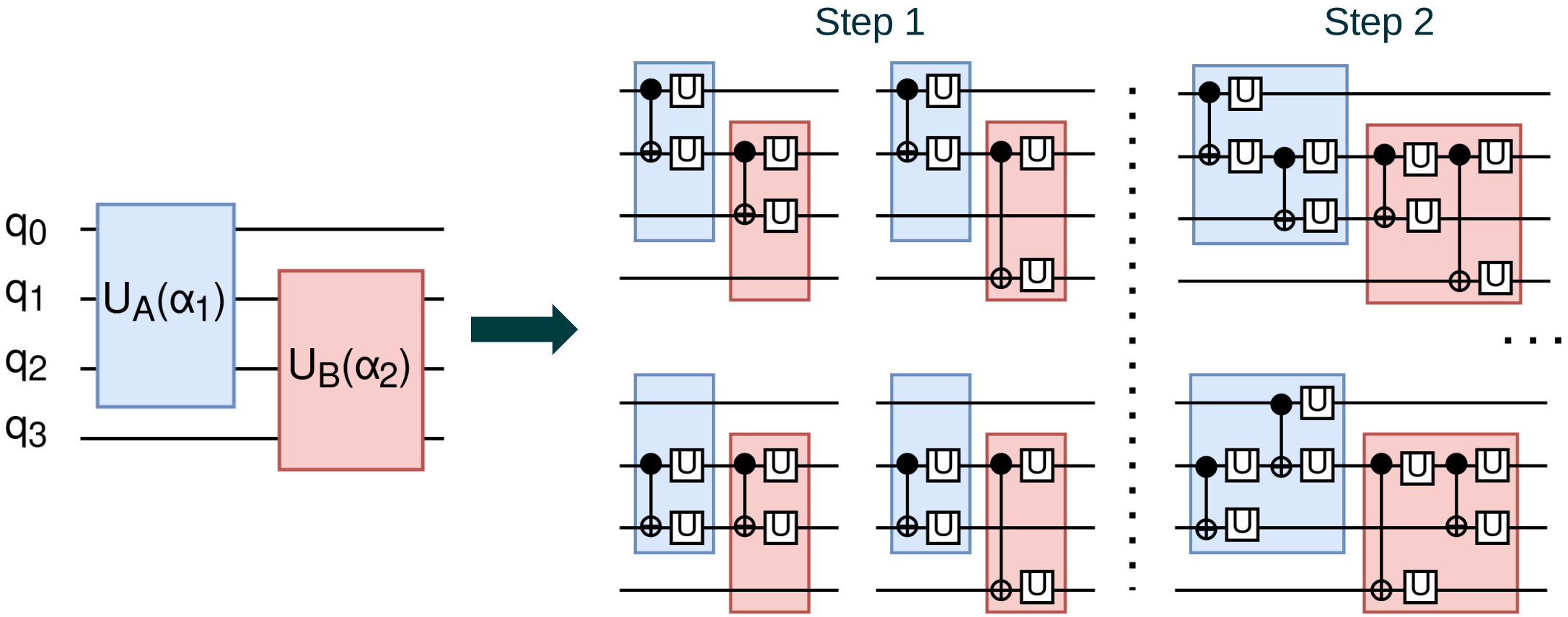


$U_1 = U_2 = U_3$

Executable!

Two-block Simultaneous Unitary Synthesis

- Synthesize two n-1 blocks simultaneously with block structure constraints
- Topology aware (no need mapping or routing)

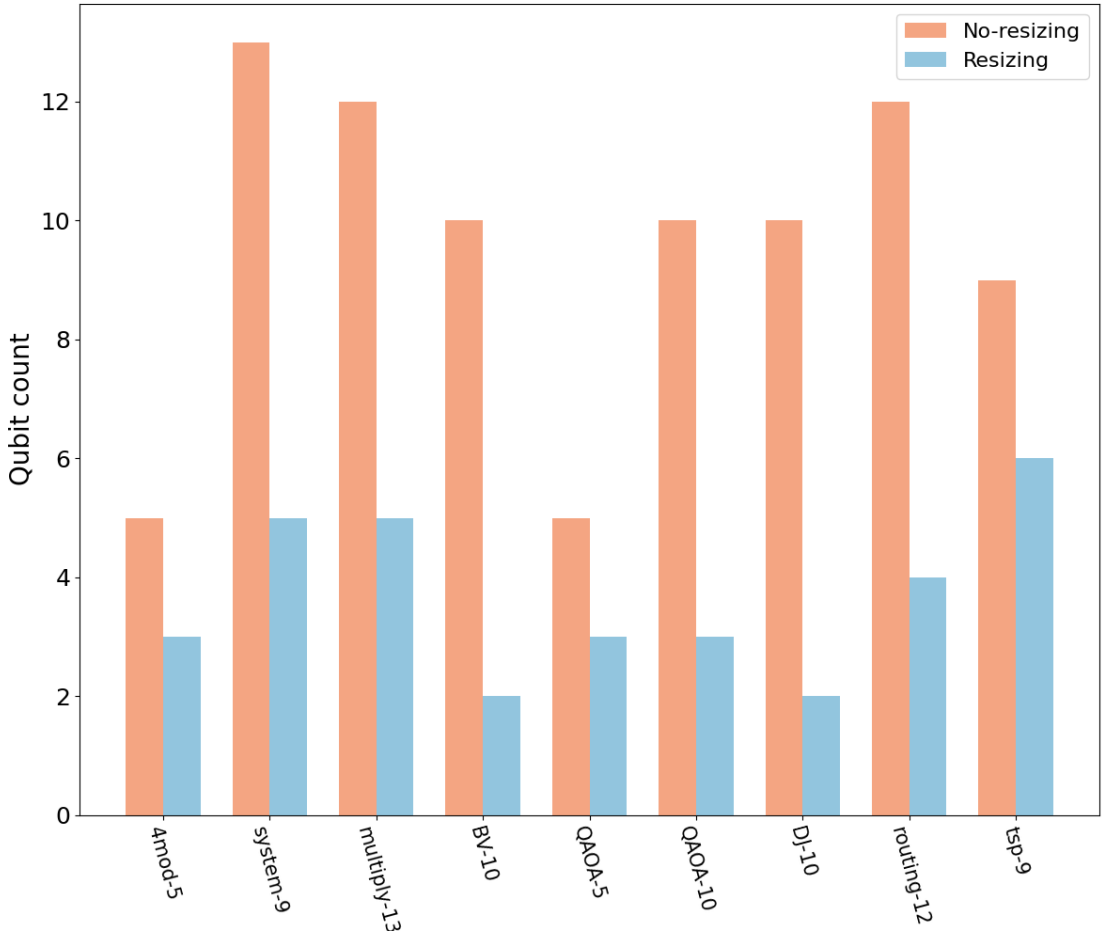


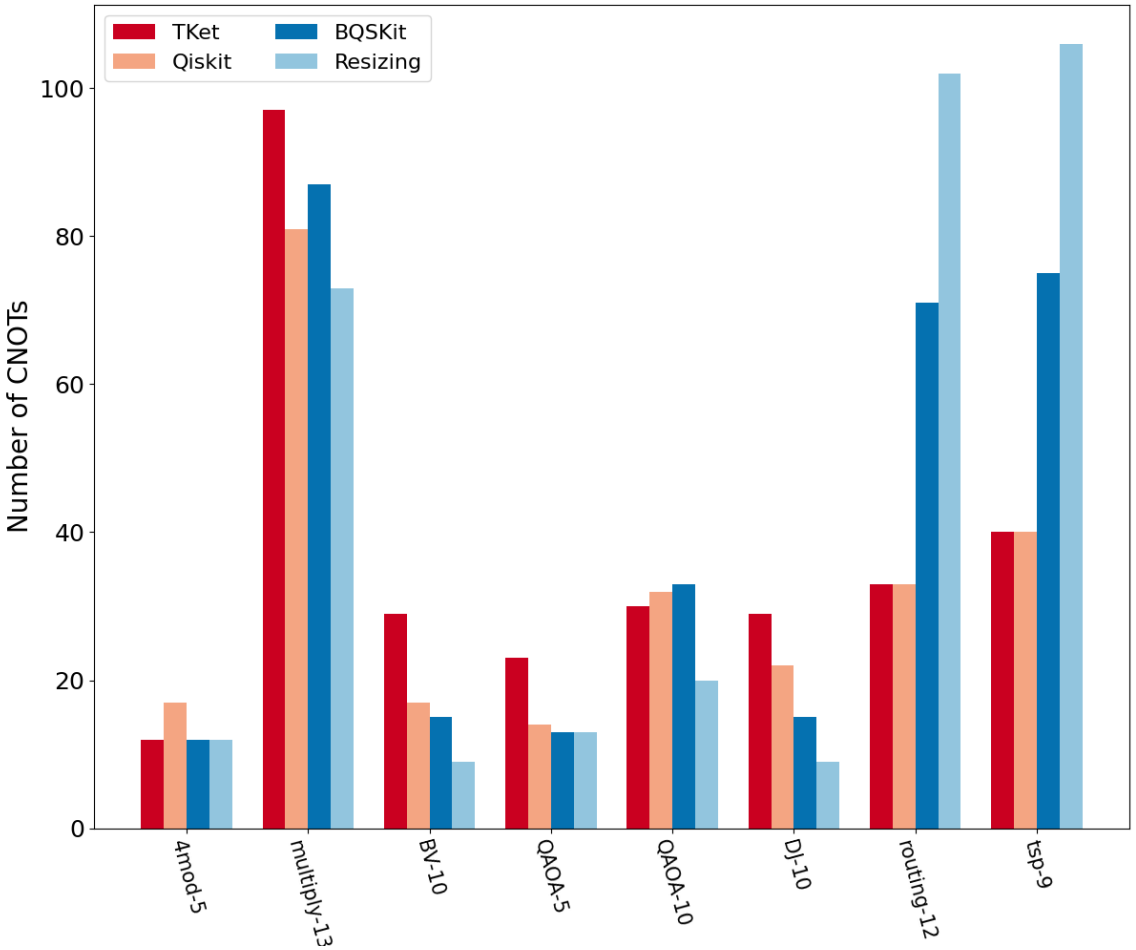
Experimental Setup

- Our resizing algorithms are built on top of BQSKit using Python 3.11.4.
- Comparison
  - Qiskit
  - Tket
  - BQSKit
- Benchmarks
  - VQE, QAOA, Bernstein-Vazirani, ...
- Evaluations on two IBM Quantum hardware
- Our code is available on Github! Try it out!

github.com/bqskit/bqskit-resize

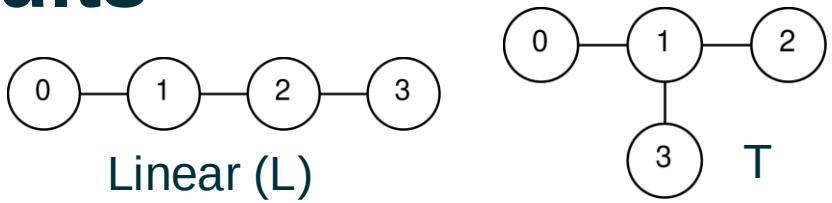
Gate-Dependency Resizing Results

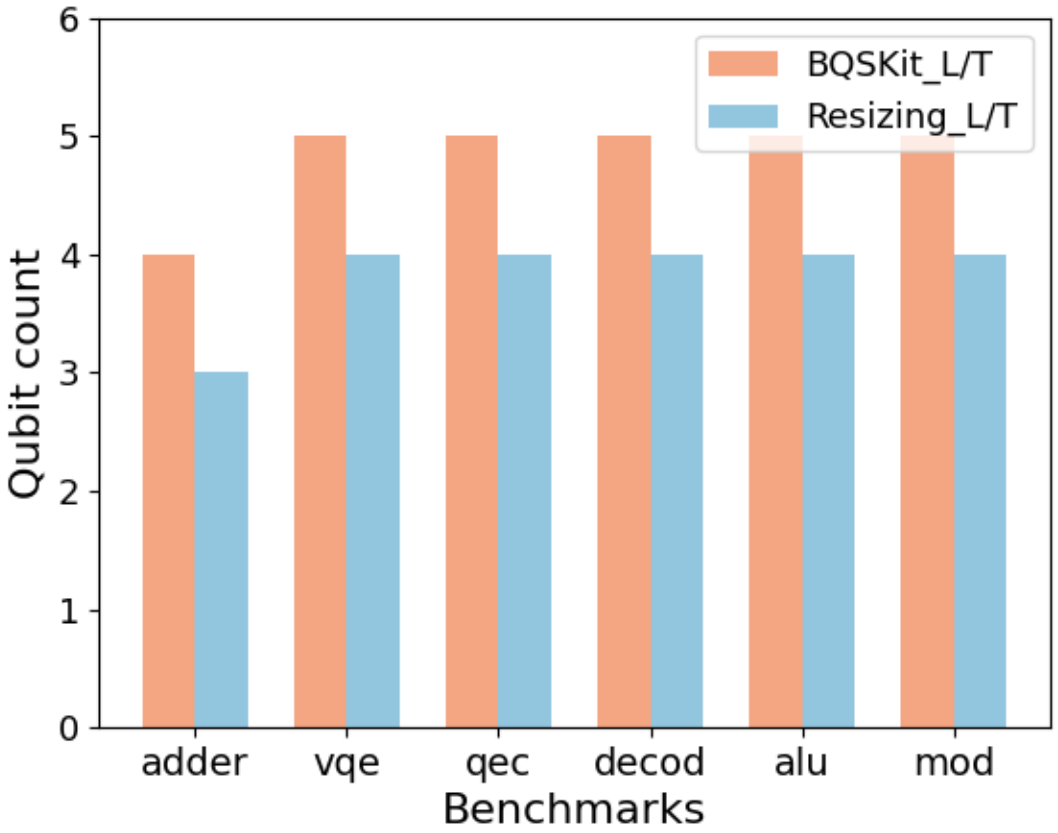


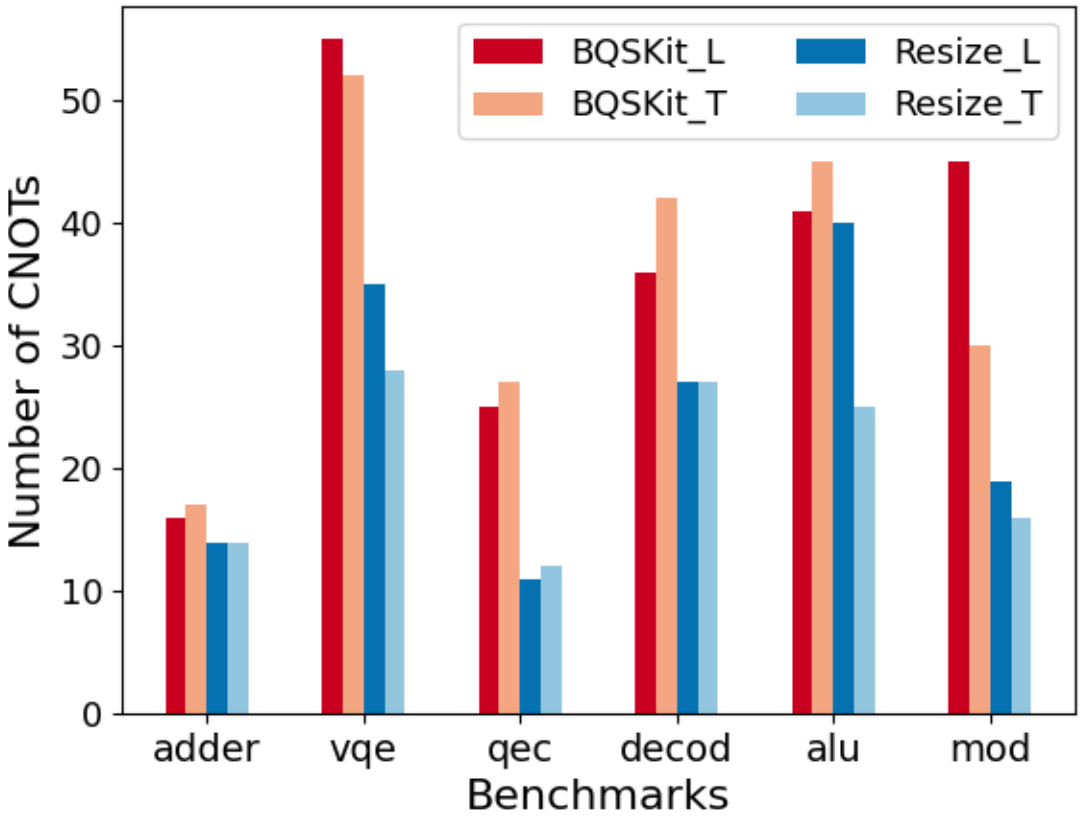


We reduce qubit count by **61.6**% and #CNOT by **11.4**%.

Instantiation-Based Resizing Results

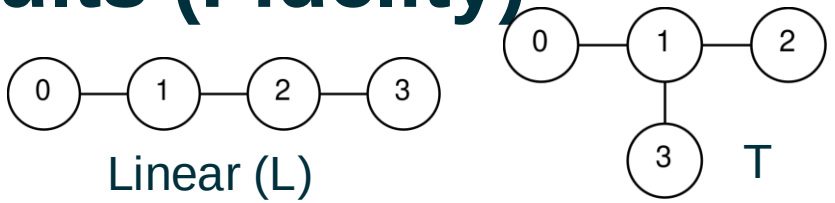


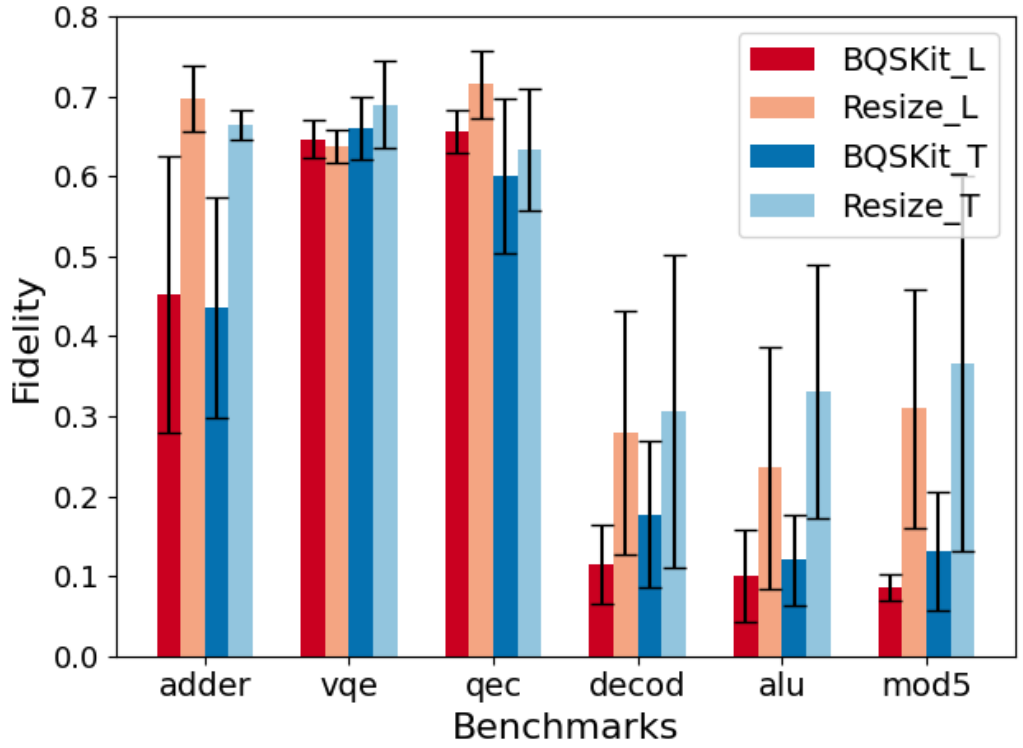




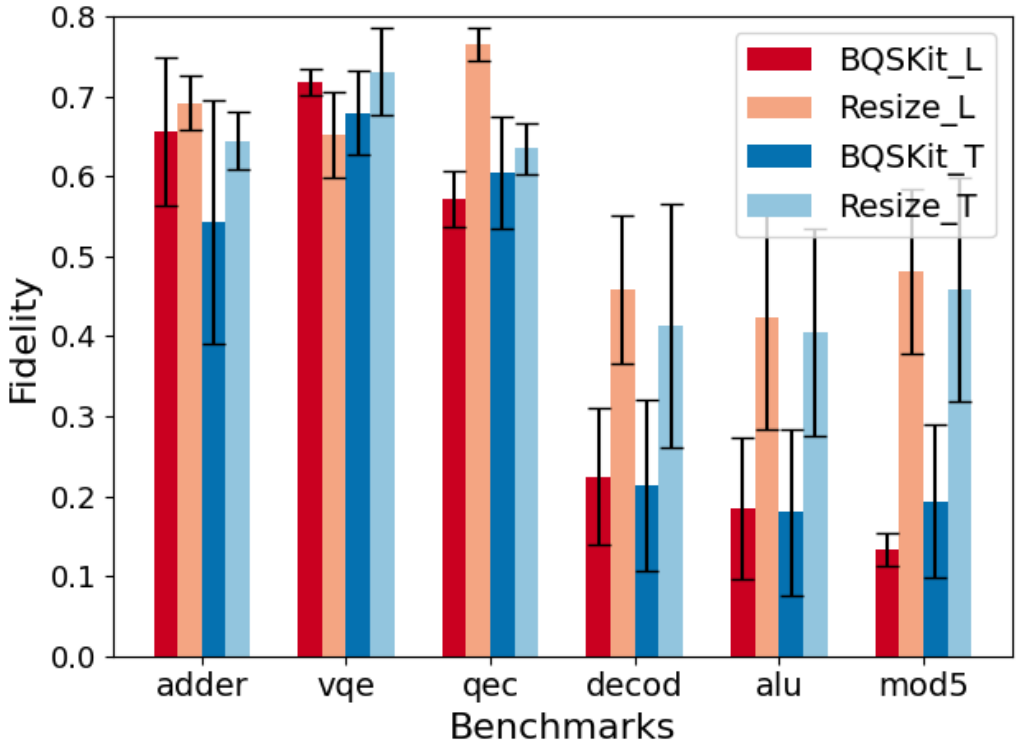
We reduce qubit count by 20.7% and #CNOT by 33% and 42.7% when compiled to linear (L) or T topology.

Instantiation-Based Resizing Results (Fidelity)





(a) ibmq_aukland



(b) ibmq_hanoi

We improve circuit fidelity by **28.5%** and **28.4%** for two chips with linear topology, and by **28.9%** and **26.6%** with T topology.

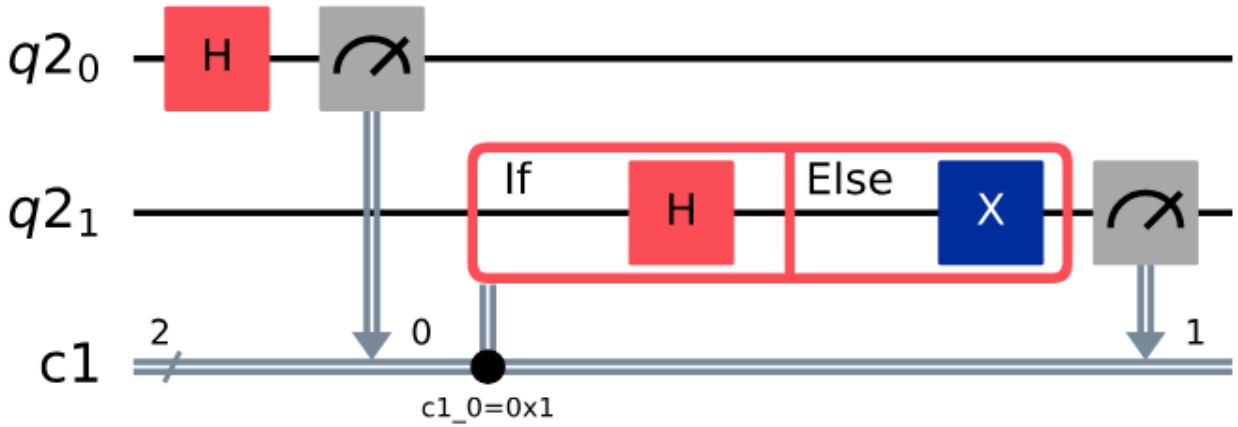
AC/DC: Automated Compilation for Dynamic Circuits

Empowering instantiation with dynamic circuit capabilities

Niu, Siyuan, et al. arXiv preprint arXiv:2412.07969 (2024).

Dynamic Circuits for Resource Optimization

- Dynamic circuits include mid-circuit measurement and feed-forward.
- Measuring some qubits and applying feed-forward operations to the un-measured qubits.



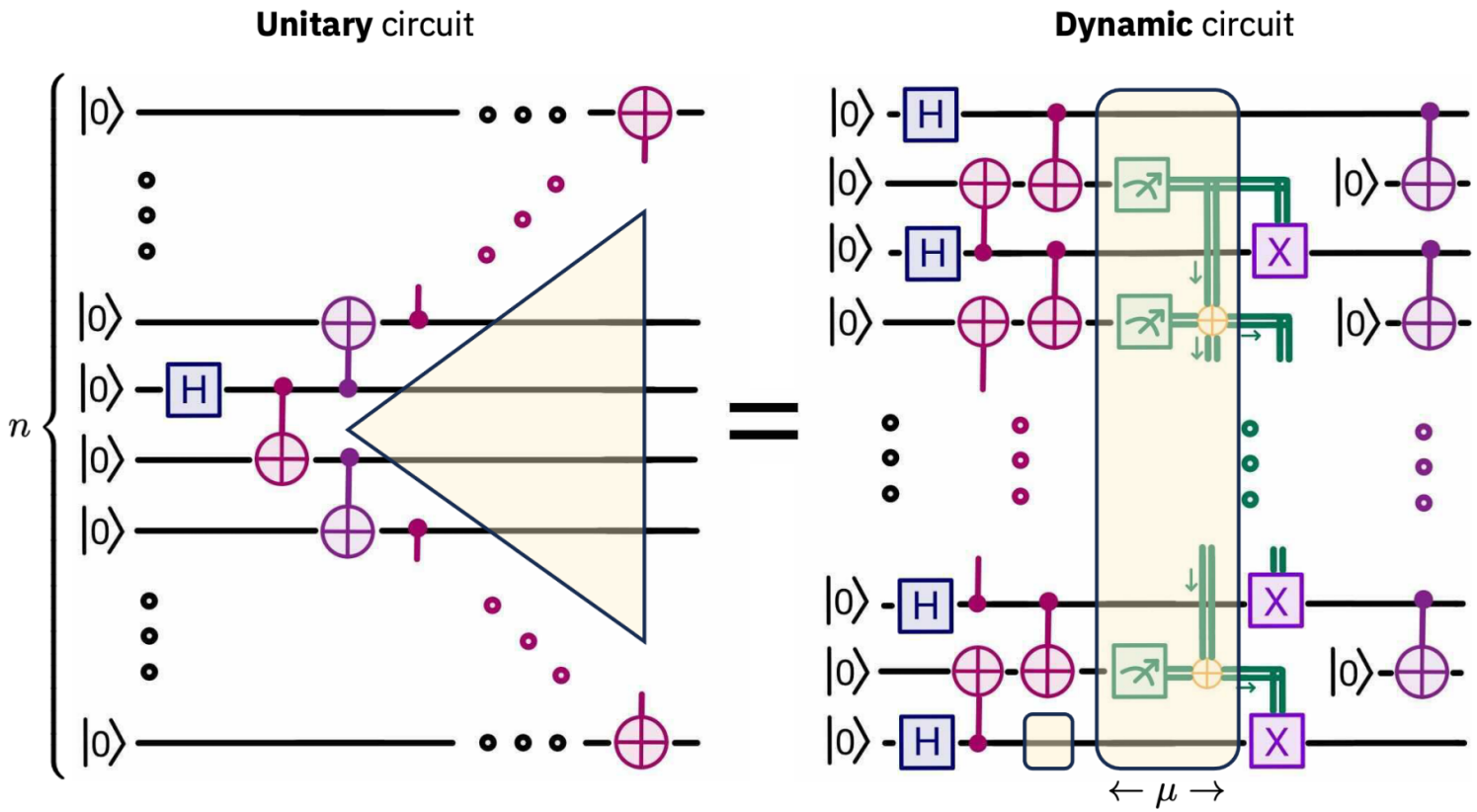
If c1 = 1, apply H to q21

If c1 = 0, apply X to q21

- Quantum resource optimization: reduce circuit depth and #CNOTs

Constant-Depth GHZ State Preperation

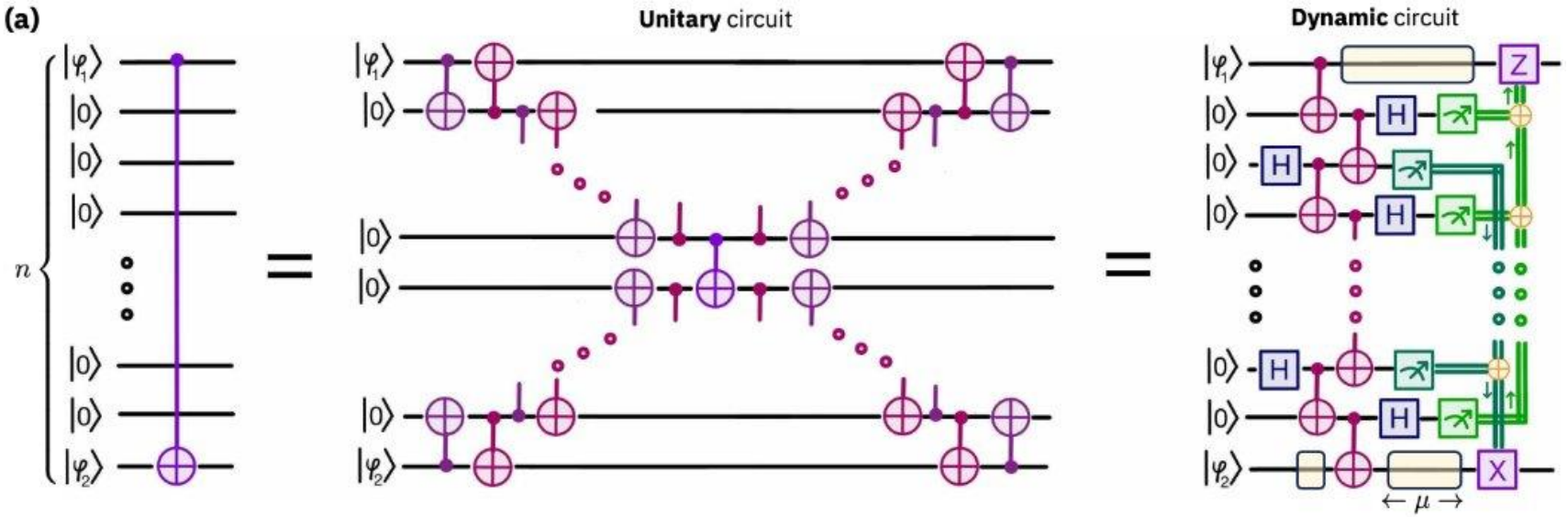
- Achieving GHZ state preperation on linear topology with constant depth **manually**.



Bäumer, Elisa, et al. "Efficient long-range entanglement using dynamic circuits." PRX Quantum 5.3 (2024): 030339.

Optimizing long-range CNOT

- Achieving long range CNOT on linear topology with constant depth **manually**.

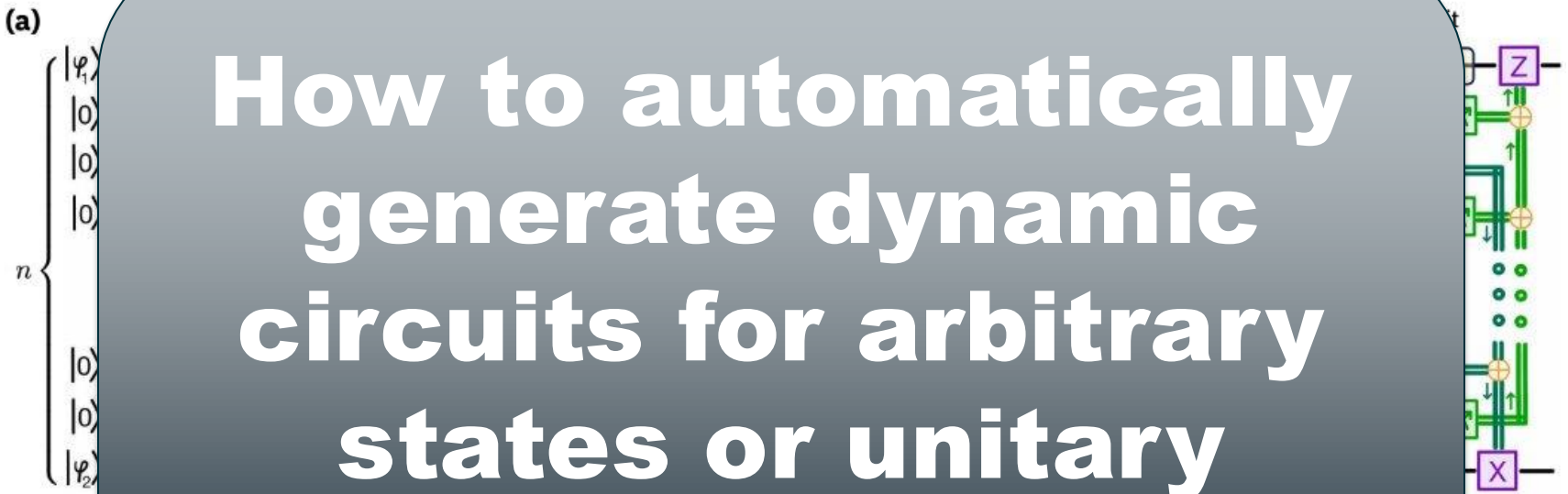


- Other applications in resource optimization
  - Quantum Fourier transformation
    - Quantum phase estimation
    - Quantum teleportation in circuit cutting

Corcoles et al. PRL 2021, Baumer et al. PRL 2024, Baumer et al. PRX Quantum 2024, Nature 636, 75–79 (2024)

Optimizing long-range CNOT

- Achieving long range CNOT on linear topology with constant depth **manually**.



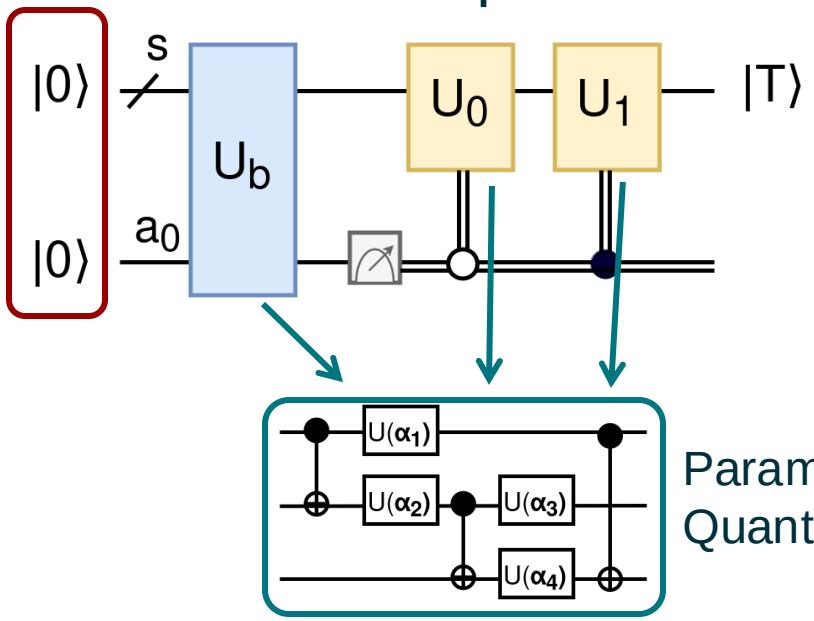
How to automatically generate dynamic circuits for arbitrary states or unitary operators?

- Other applicat
  - Quantum Fou
    - Quantum phase estimation
    - Quantum teleportation in circuit cutting

Corcoles et al. PRL 2021, Baumer et al. PRL 2024, Baumer et al. PRX Quantum 2024, Nature 636, 75–79 (2024)

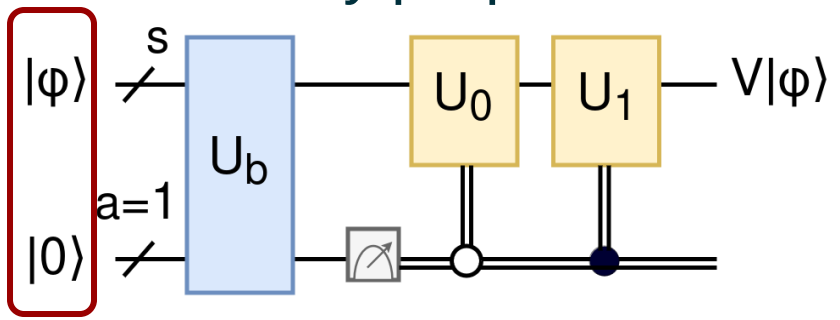
Automated Instantiation of Dynamic Circuits

State Preparation



Parameterized Quantum Circuit

Unitary preparation



Goal: Find parameters in U_b and U_i to prepare the target state or unitary V.

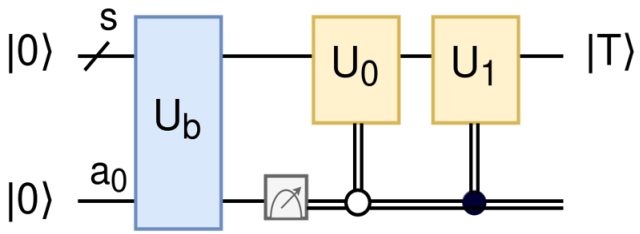
Challenge: How to deal with the non-unitary operations in synthesis?

Cost Functions for Dynamic State Preparation

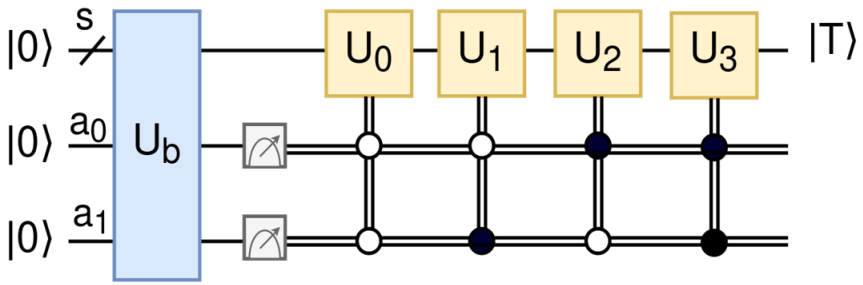
- Parameters
  - s - System qubits
  - a - Ancilla qubits
  - U_b - Unitary applied to all the qubits
  - U_i - Conditional unitary applied to system qubits
  - T - Target state

- **Cost Function (C_T):**

$$C_T = 1 - \sum_i F_T(|\phi_i\rangle) = 1 - \sum_i |\langle T \otimes i | \phi_i \rangle|^2.$$



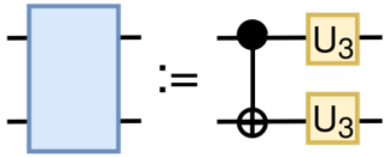
(a)

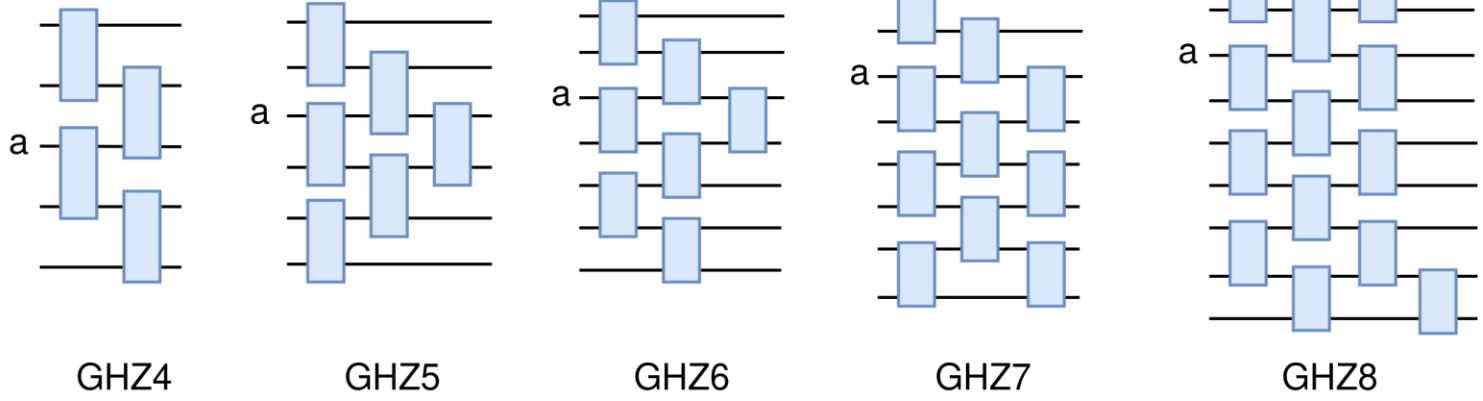


(b)

State Prep for GHZ State with One Ancilla

$$|\mathrm{GHZ}_n\rangle = \frac{|0\rangle^{\otimes n} + |1\rangle^{\otimes n}}{\sqrt{2}}$$



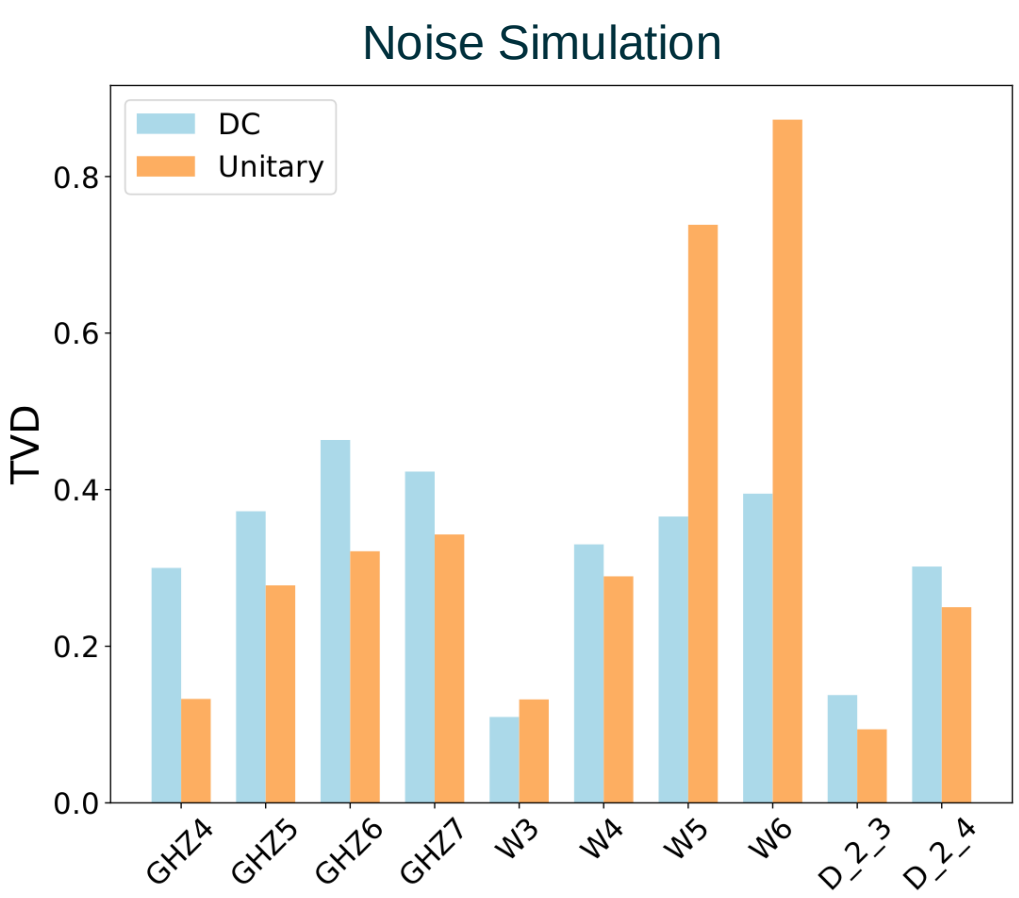


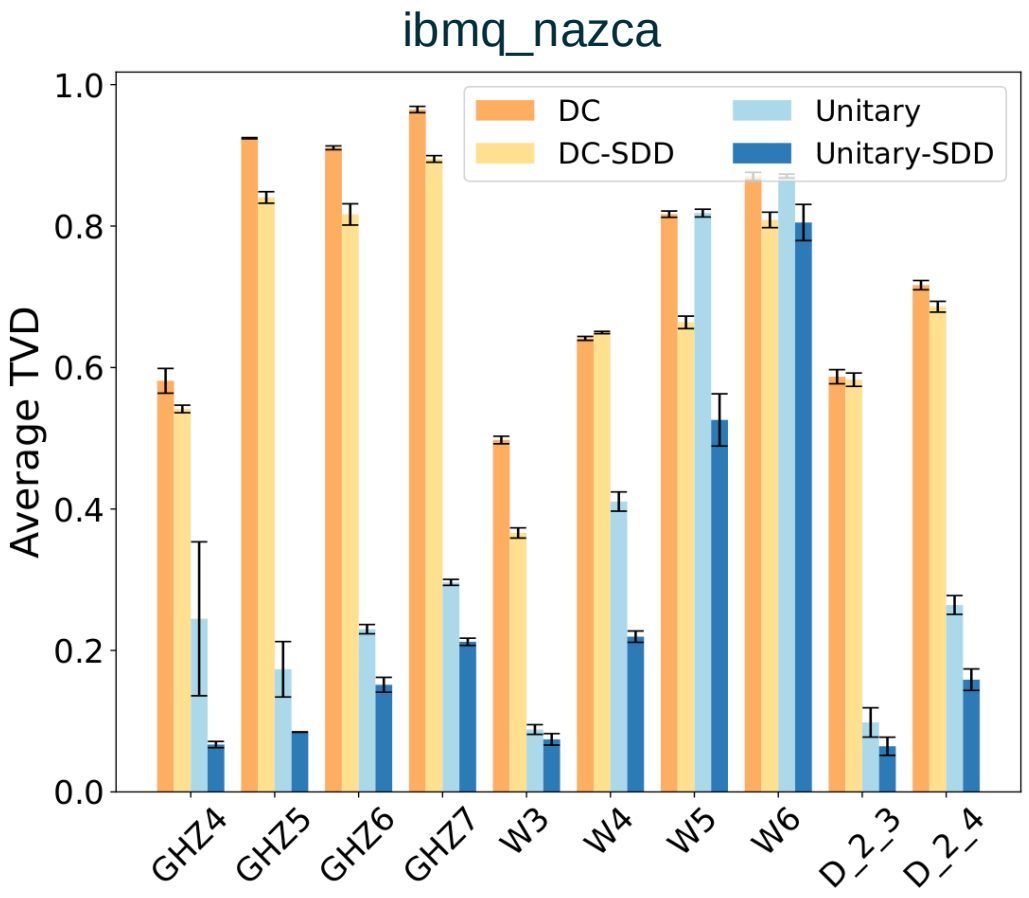
Dynamic circuits achieve **shorter circuit depth**

States	n	CNOT		Depth	
		Unitary	Dynamic	Unitary	Dynamic
GHZ4	4	3	4	3	**2**
GHZ6	6	5	7	4	**3**
GHZ8	8	7	13	5	**4**

More ancillas in dynamic circuits can further reduce the circuit depth.

State Prep Validation on IBM Quantum Hardware





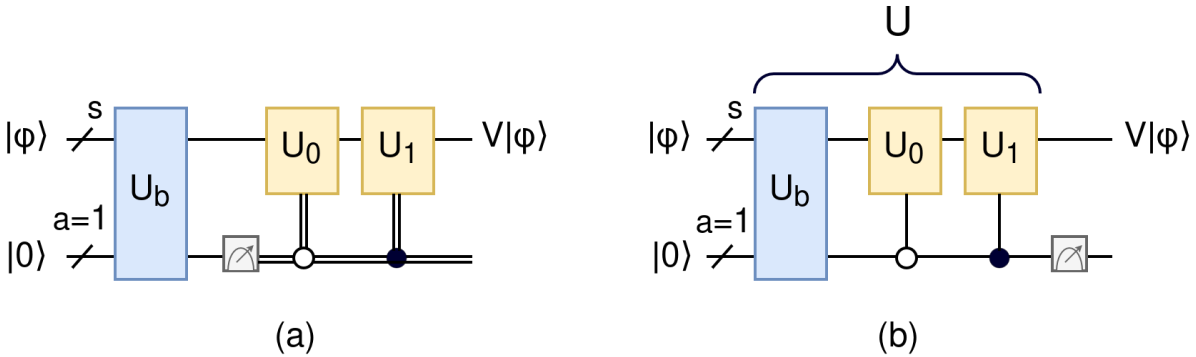
Noisy simulation results show that dynamic circuits sometimes obtain better fidelity (W states).

Cost Functions for Dynamic Unitary Instantiation

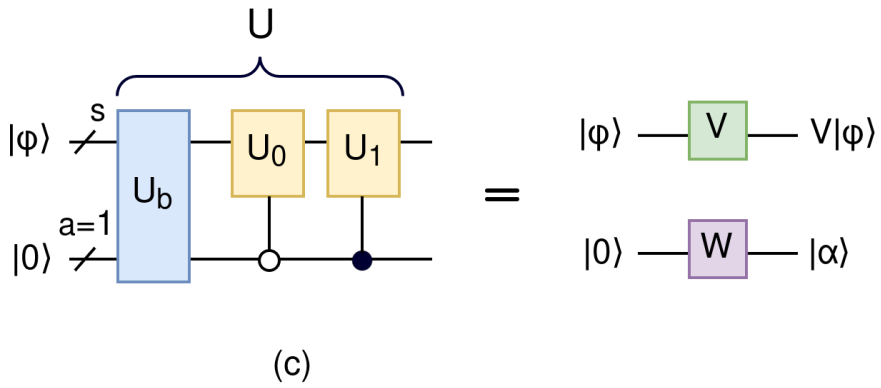
Unitary circuit - Hilbert Schmidt distance: $1 - \frac{1}{2^n}\left|\mathrm{Tr}\left(V^\dagger U\right)\right|$

Cost Functions for Dynamic Unitary Instantiation

Unitary circuit - Hilbert Schmidt distance: $1 - \frac{1}{2^n}\left|\mathrm{Tr}\left(V^\dagger U\right)\right|$



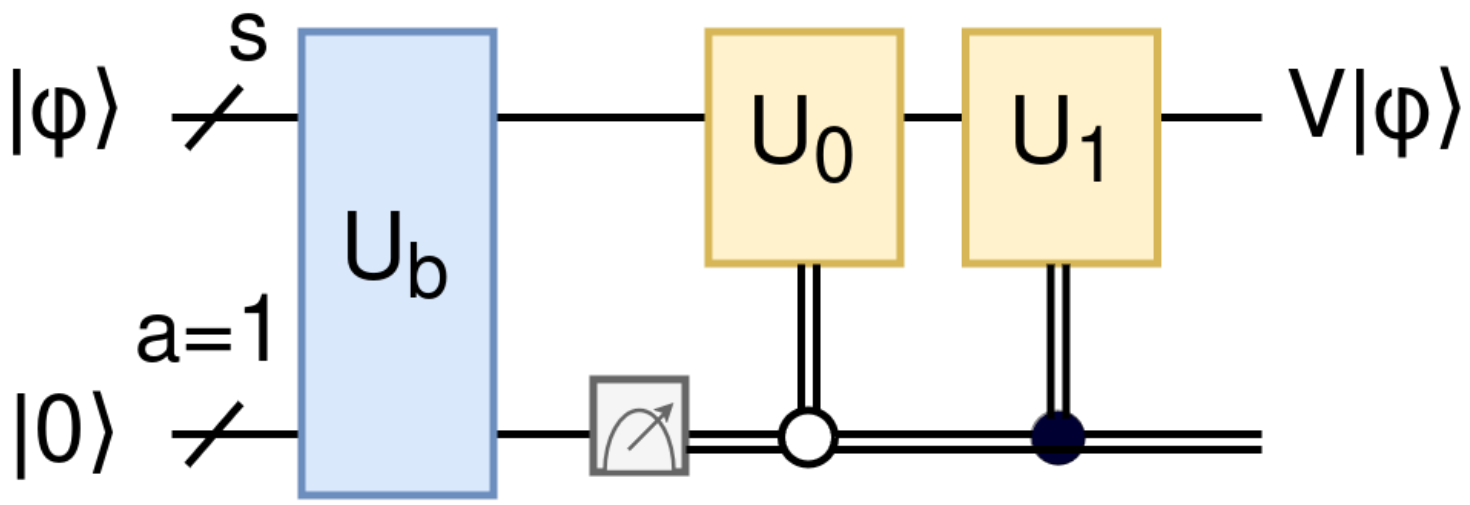
$$f_0 = I \otimes I \otimes \ldots I \otimes |0\rangle\langle 0|$$



$$C_{\mathrm{dyn1}}(U,W) = 1 - \frac{1}{2^s}\left|\mathrm{Tr}\left(f_0(V\otimes W)^\dagger U\right)\right|$$

First method for unitary preparation

Cost Functions for Dynamic Unitary Instantiation



$$C_{\text{dyn2}}(U) = 1 - \frac{1}{4^s} \sum_i \left| \text{Tr}\left(U_{i,0} V^\dagger \right) \right|^2$$

Ancilla System

$$\begin{aligned} U =& |0\rangle\langle 0| \otimes U_{00} \\ &+ |0\rangle\langle 1| \otimes U_{01} \\ &+ |0\rangle\langle 0| \otimes U_{10} \\ &+ |1\rangle\langle 1| \otimes U_{11} \\ =& \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \end{aligned}$$

The ordering of ancilla and system qubits determines elements

Cost Functions for Dynamic Unitary Instantiation

$$C_{\text{dyn1}}(U, W) = 1 - \frac{1}{2^s} \left| \text{Tr} \left(f_0 (V \otimes W)^\dagger U \right) \right|$$

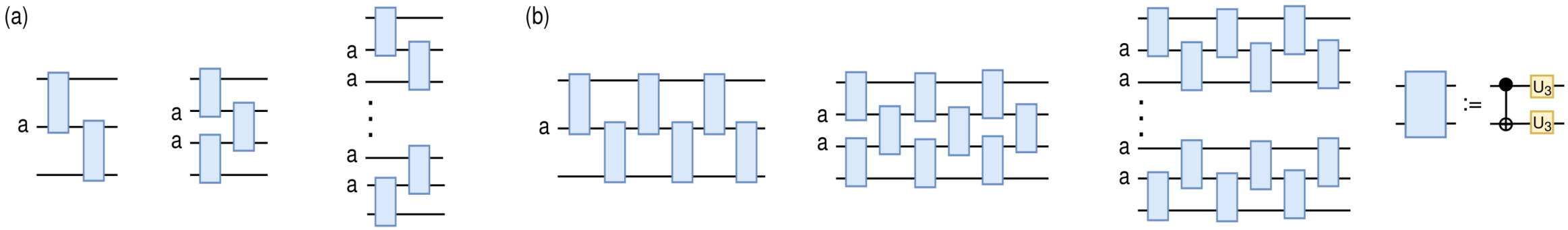
More parameters to optimize but less matrix multiplication

$$C_{\text{dyn2}}(U) = 1 - \frac{1}{4^s} \sum_i \left| \text{Tr} \left(U_{i,0} V^\dagger \right) \right|^2$$

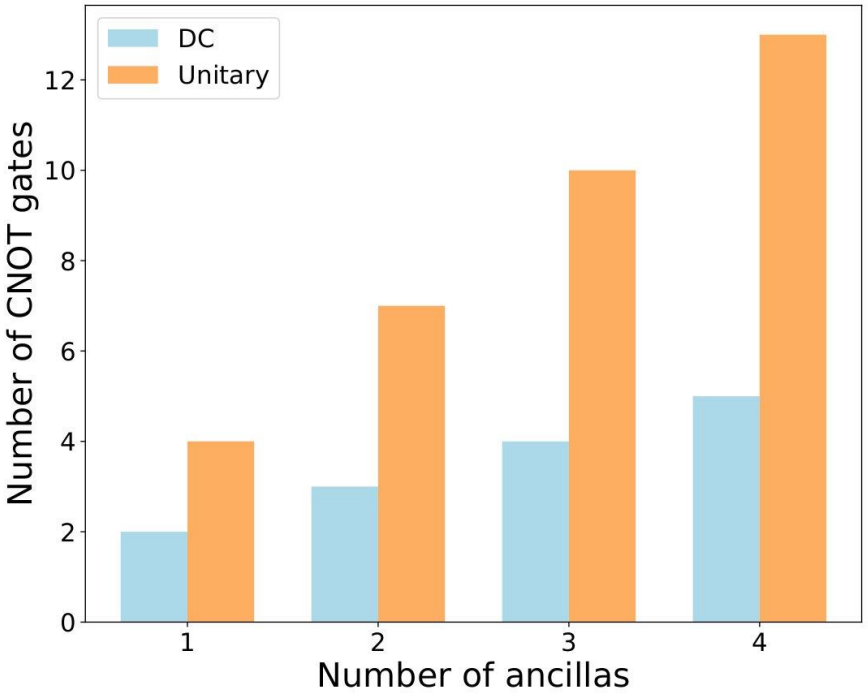
Less parameters to optimize but more matrix multiplication

Based on empirical experimental results **C_{dyn1}** is preferred

Dynamic Unitary Instantiation - Long Range Gates



- Dynamic circuit
  - (a) Long-range CNOT/CZ/CS/Rzz/Rxx/Ryy
    - Depth: constant 2 *O(1)*
    - CNOT: n - 1 *O(n)*
  - (b) Long-range generic two-qubit gate
    - Depth: constant 6 *O(1)*
    - CNOT: (n-1) x 3 *O(n)*
- Unitary circuits
  - circuit depth *O(n)*
  - CNOT: *O(n)*

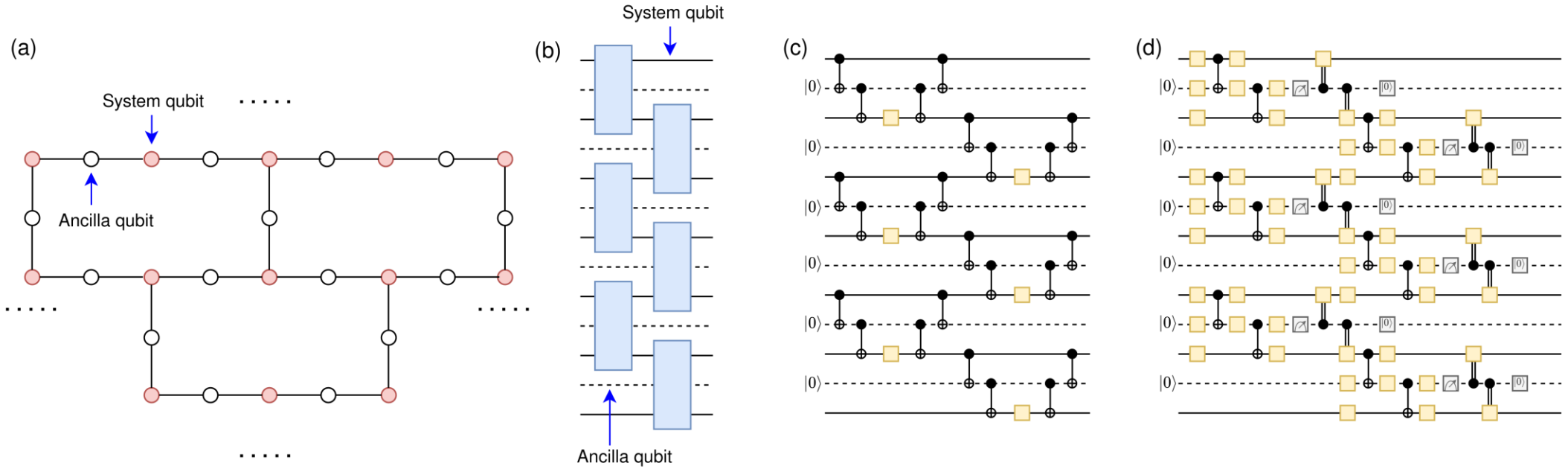


Dynamic Circuit Compilation for Lattice Simulation

Problem: Simulate time evolution of TFIM on a hexagonal lattice

$$H = J \sum_{(i,j)\in E} Z_i Z_j + B \sum_{i \in V} X_i$$

First scalable demonstration of dynamic circuit for lattice simulation!



IBM Heavy-Hex | Trotter Circuit | Unitary Circuit | Dynamic Circuit

Dynamic Circuit Compilation for Lattice Simulation

Problem: S

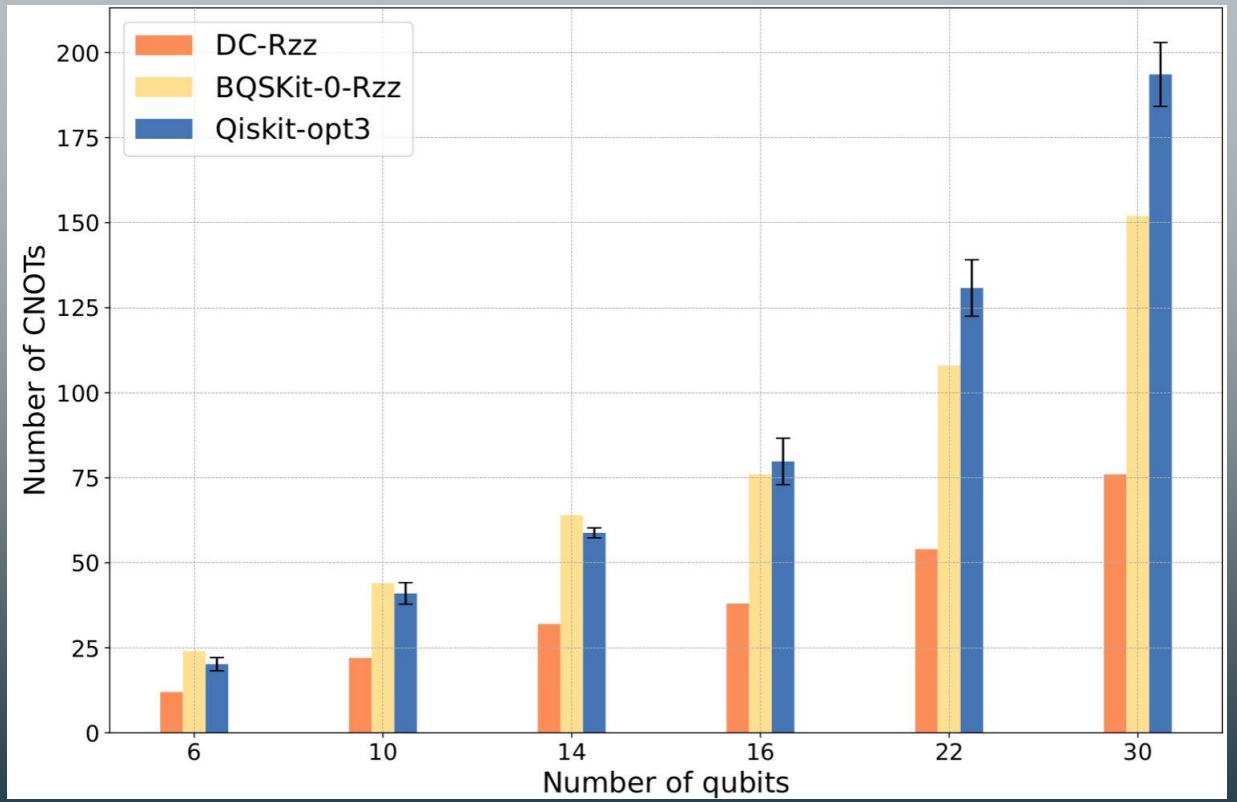
$H =$

(a)

DC: constant depth of 4, the least number of CNOTs

BQSKit-0 (unitary): constant depth of 7

Qiskit (unitary): depth increases as the number of qubits



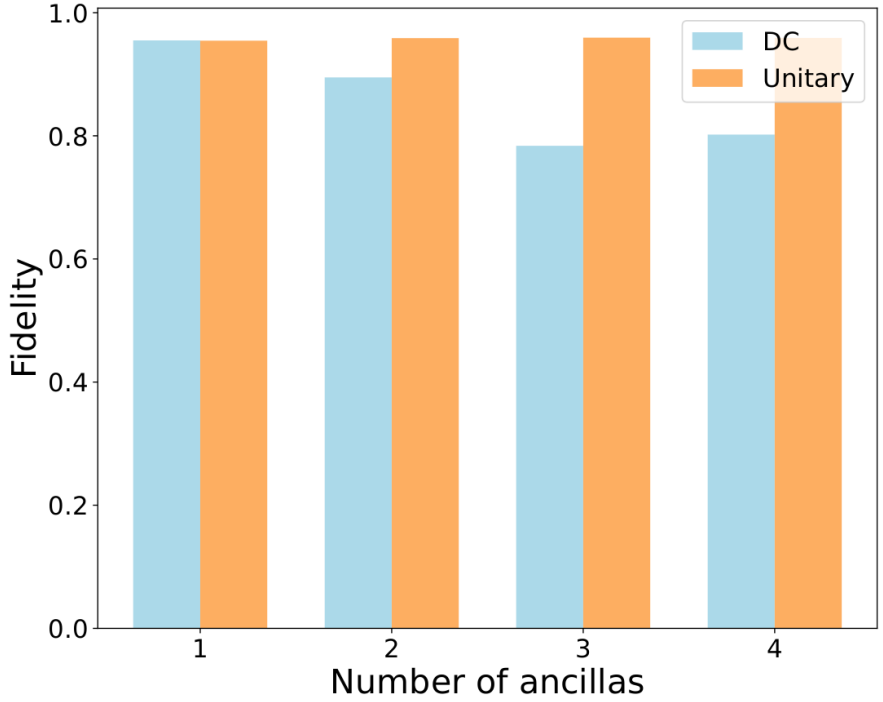
of ulation!

rcuit

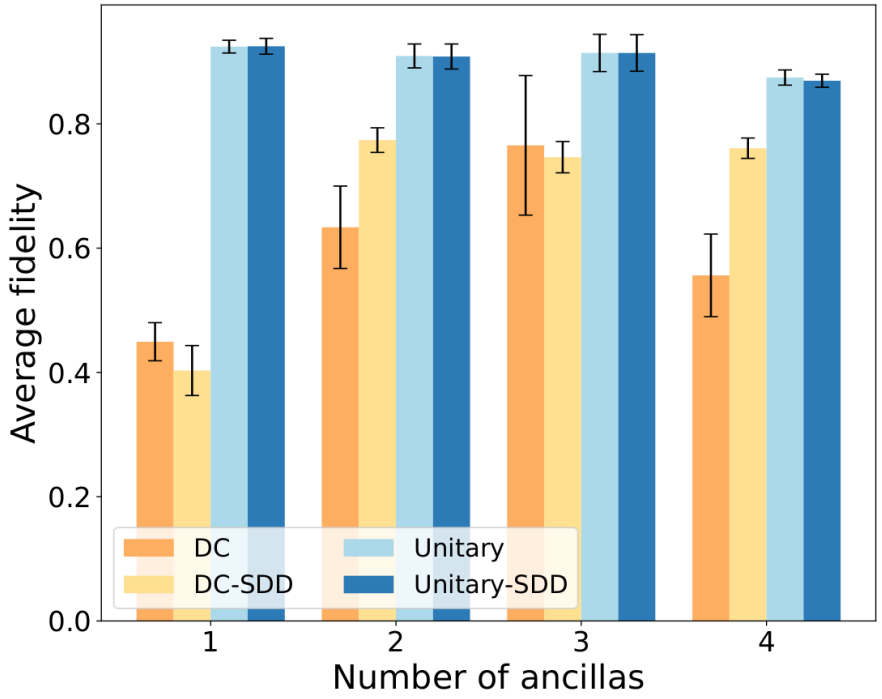
Quantum Hardware Results - Long Range Gates

- Hardware characteristics
  - Measurement error: ~10x more than 2-q gate error, ~20x longer than 2-q gate duration
  - Feed-forward loop: ~10x as long as 2-q gate duration
- **Unitary circuits obtain higher fidelity due to the large error in mid-circuit measurement.**

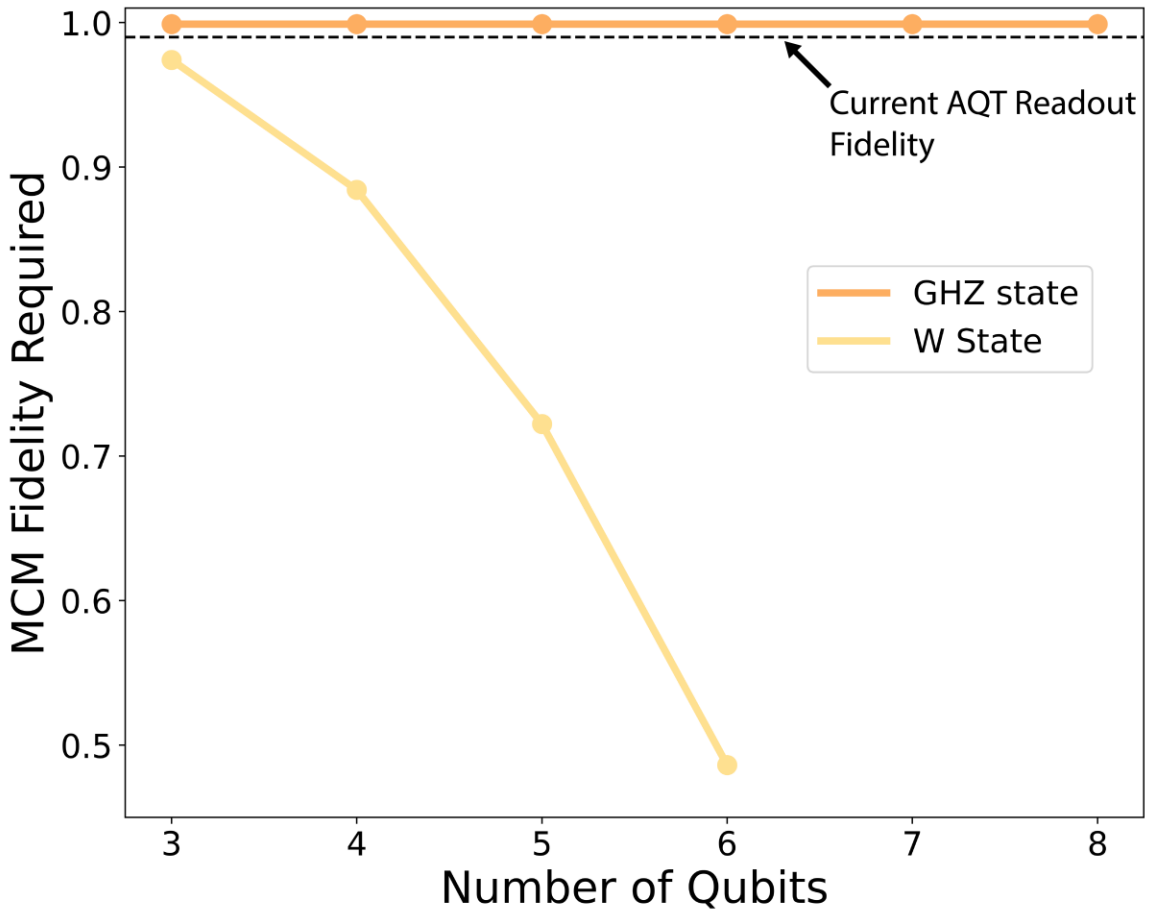
Noise simulation

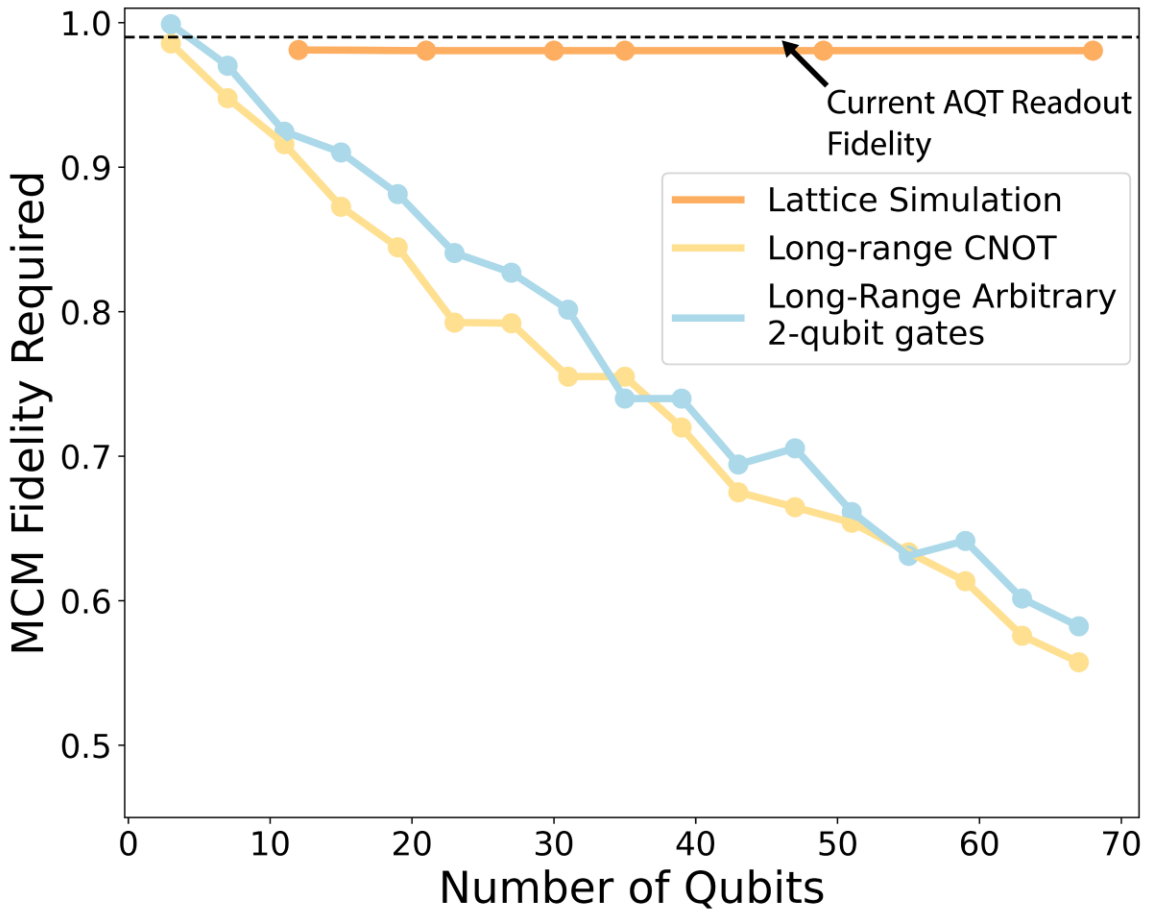


ibmq_nazca



MCM Fidelity Projection Analysis

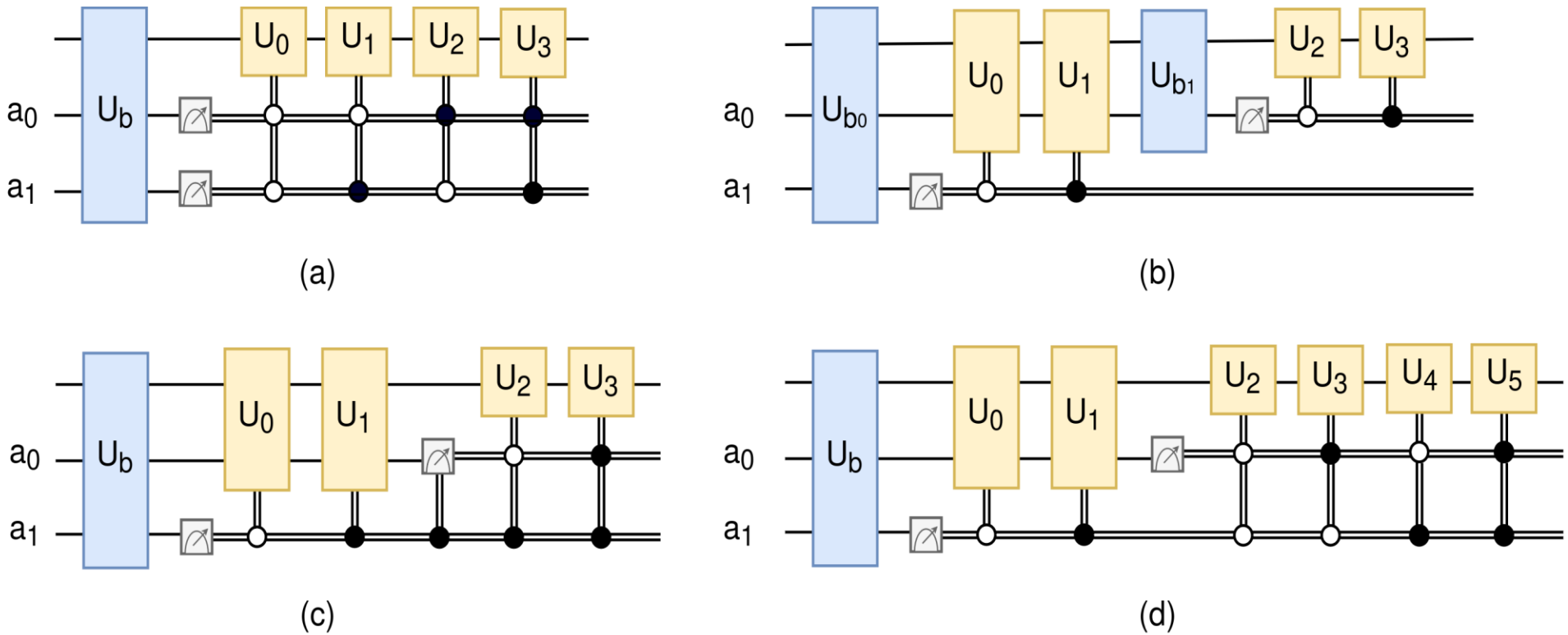




$$\mathbf{F} = f_2^{n_2} \cdot f_1^{n_1} \cdot (f_{\mathrm{MCM}})^{N \cdot M} \cdot \exp\left(-N \cdot \frac{C_T}{T_1}\right)$$

Kalloor, Justin, et al. "Quantum Hardware Roofline: Evaluating the Impact of Gate Expressivity on Quantum Processor Design." 2024 IEEE International Conference on Quantum Computing and Engineering (QCE). Vol. 1. IEEE, 2024.

Customizable Circuit Structure and Ancilla Count



(a) Simultaneous measurement
(b) Independent measurement
(c) Asymmetric Nested measurement
(d) Symmetric Nested measurement

Conclusions and Future Work

- Introduced BQSKit and the ongoing dynamic circuit work in the framework
- Next steps are to accelerate the math and release a major software update to BQSKit
- Upgrade the search step of synthesis to work on branched circuits
- Upgrade the partitioning methods in BQSKit to handle ancillas

BQSKit

bqskit.lbl.gov

Thank You!

```
from bqskit import Circuit, compile
circuit = Circuit.from_file('in.qasm')
out_circuit = compile(circuit)
out_circuit.save('out.qasm')
```

BQSKit is a powerful and portable quantum compiler framework

Tutorial and documentation online

Acknowledgements: DOE ASCR, ARQC, NERSC

We're always happy to collaborate

BQSKit

bqskit.lbl.gov