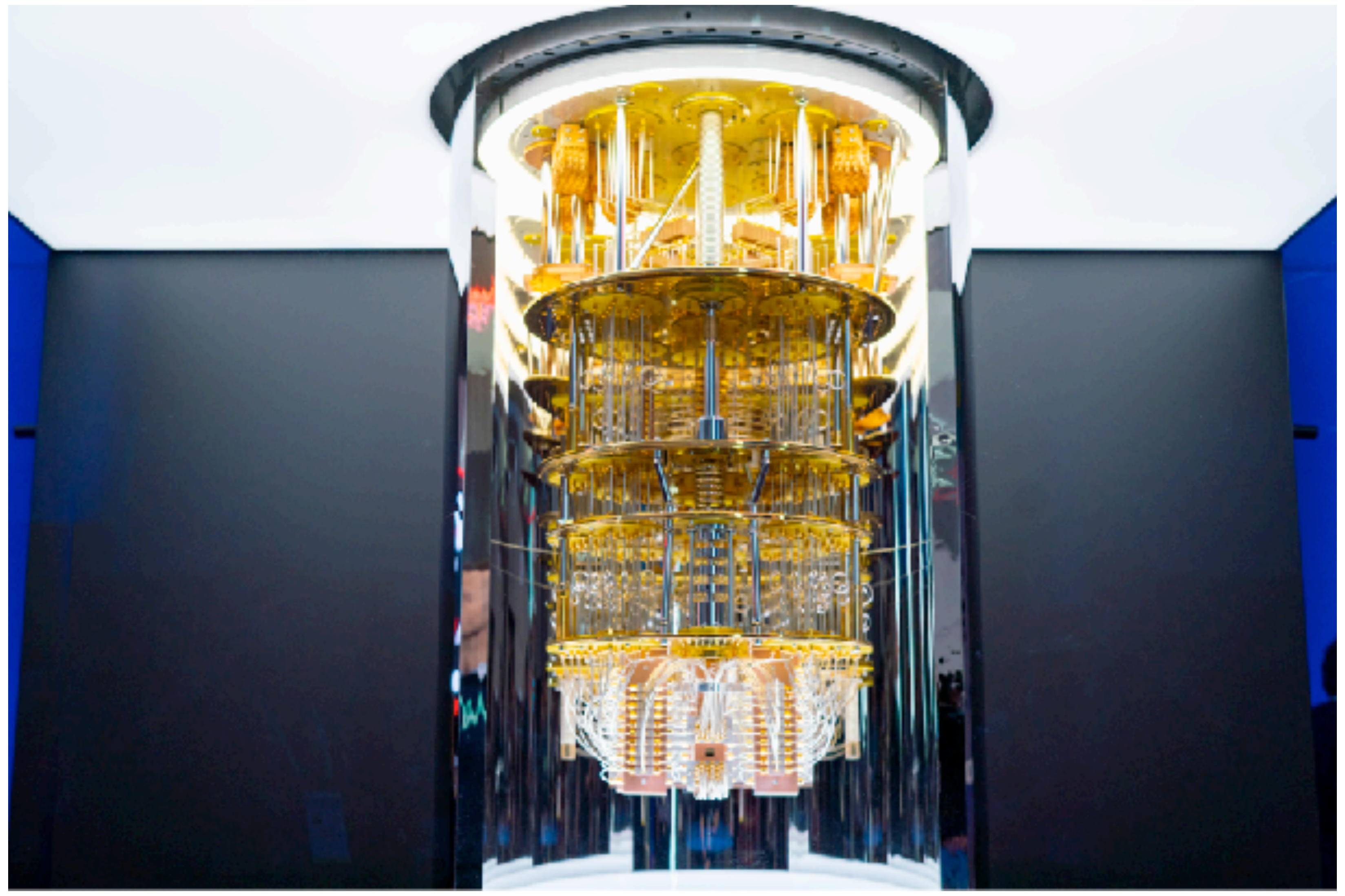
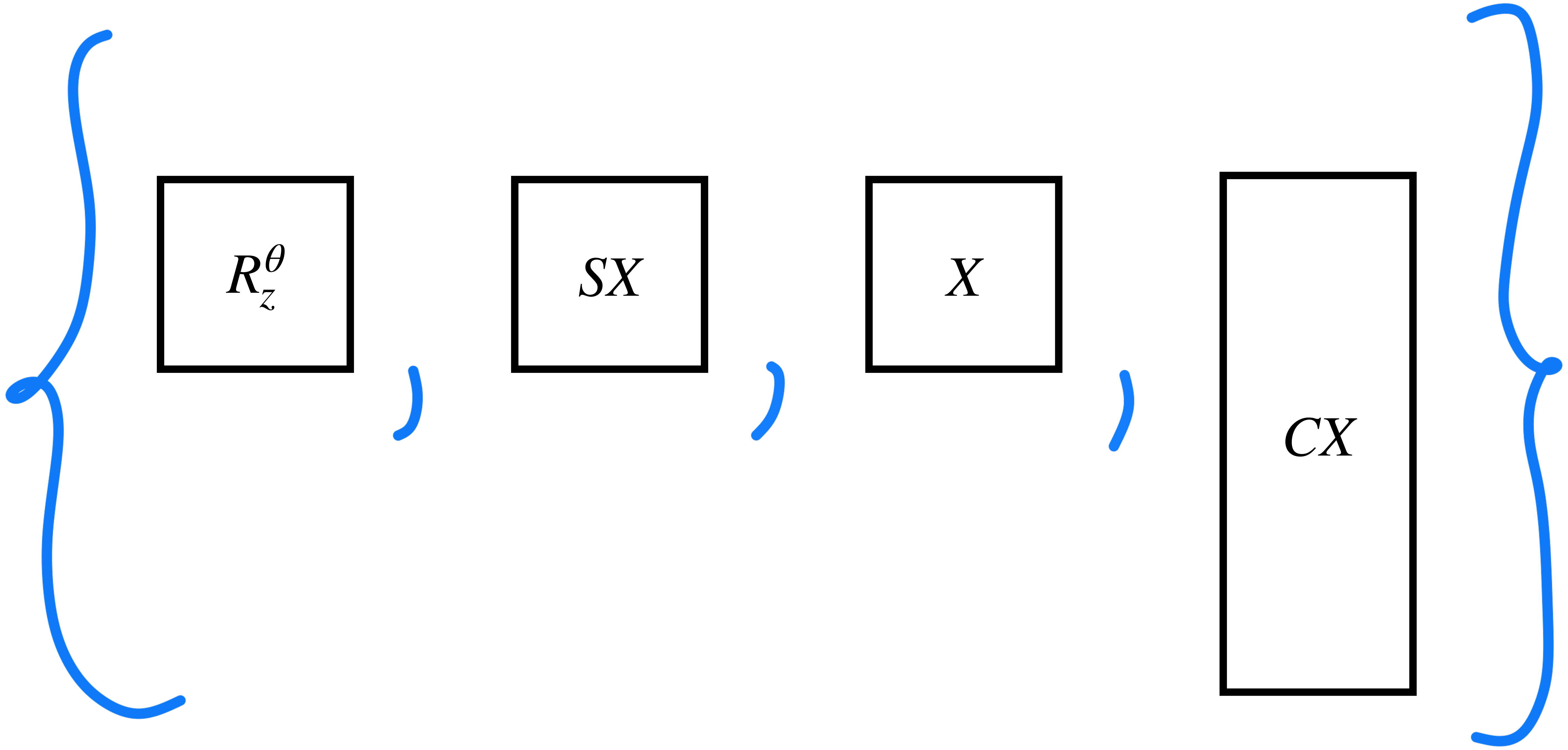
Quantum-Circuit Optimization

Native Basis Gate Set



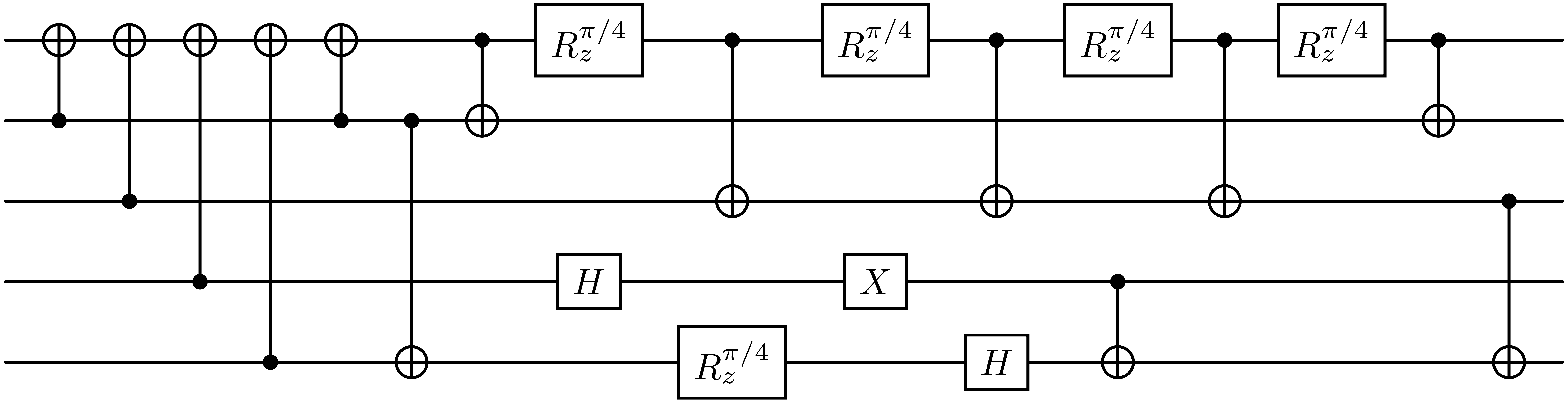
IBM



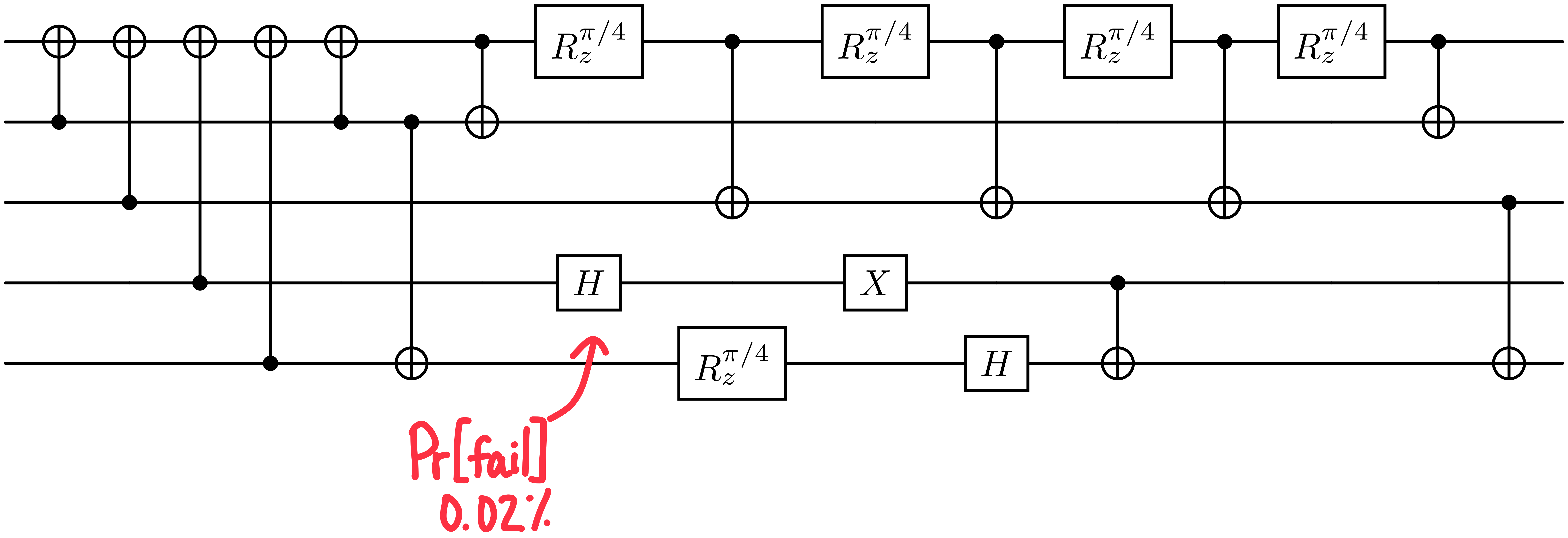
"universal"

i.e. approximate any computation to arbitrary precision

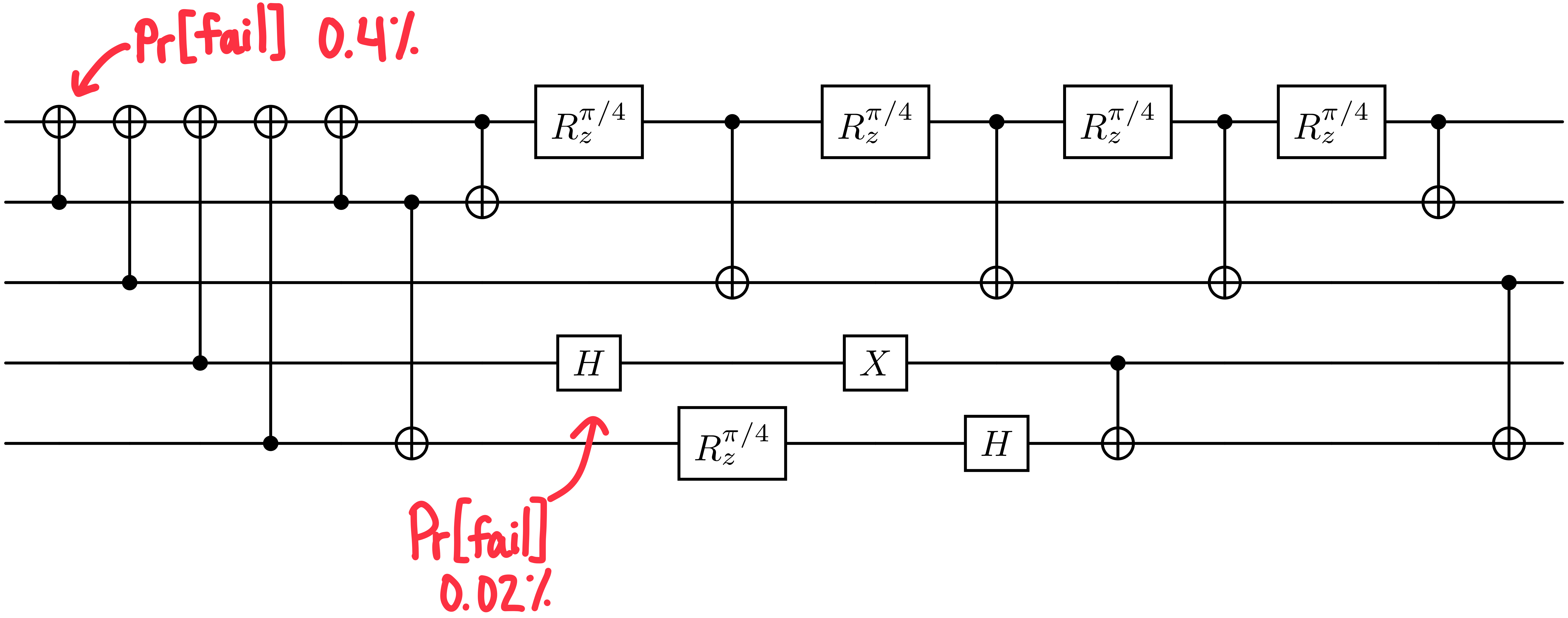
Optimization Objectives



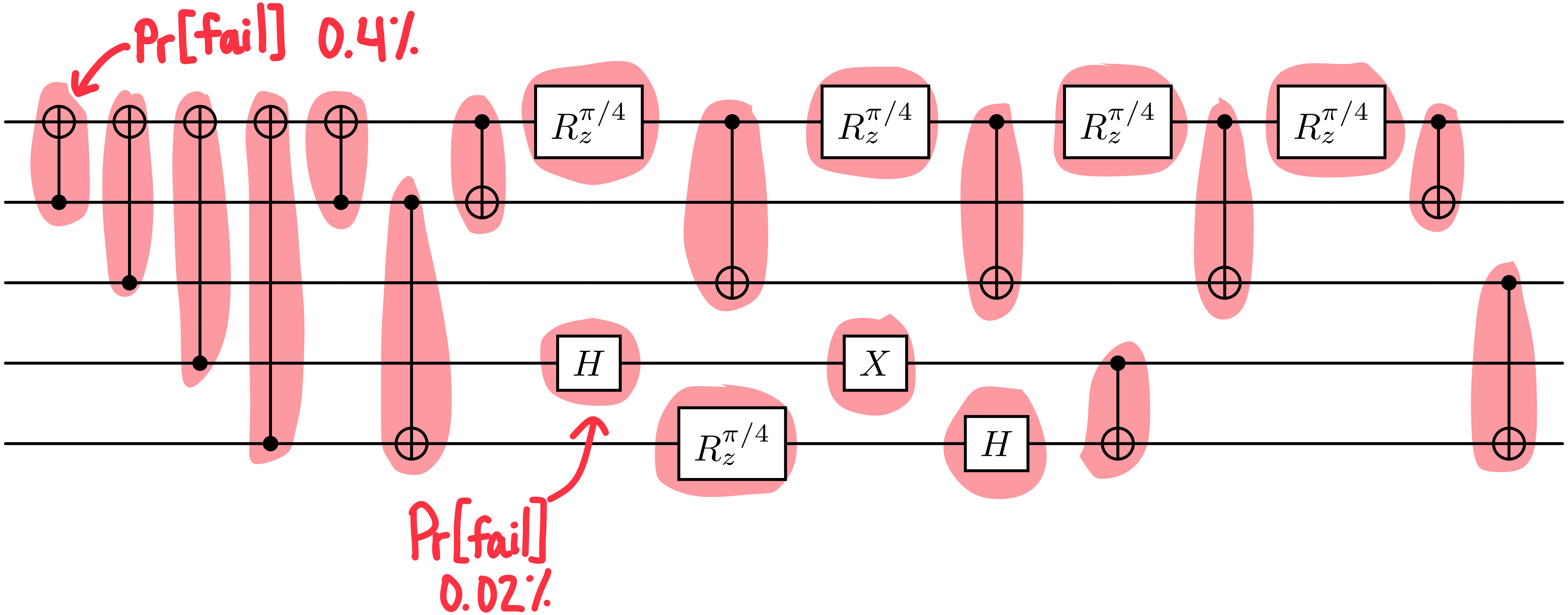
Optimization Objectives



Optimization Objectives



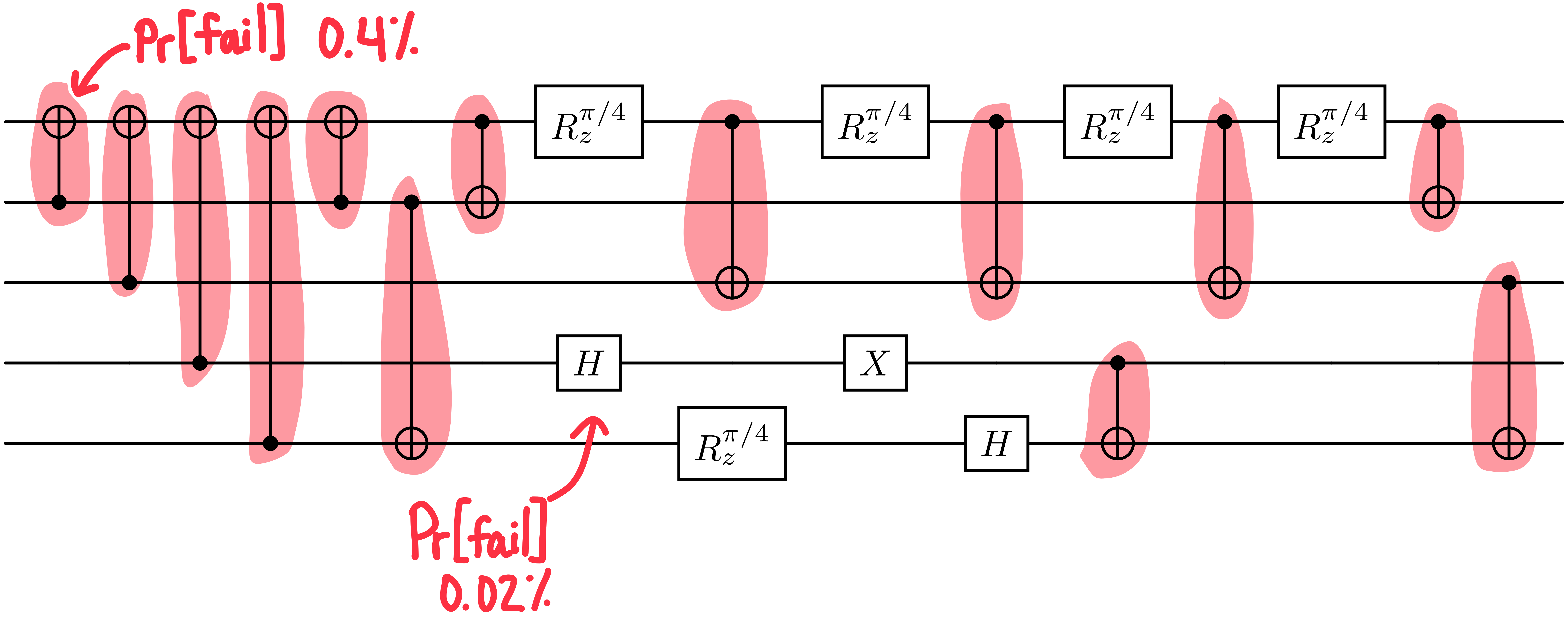
Optimization Objectives



NISQ

- total gate count

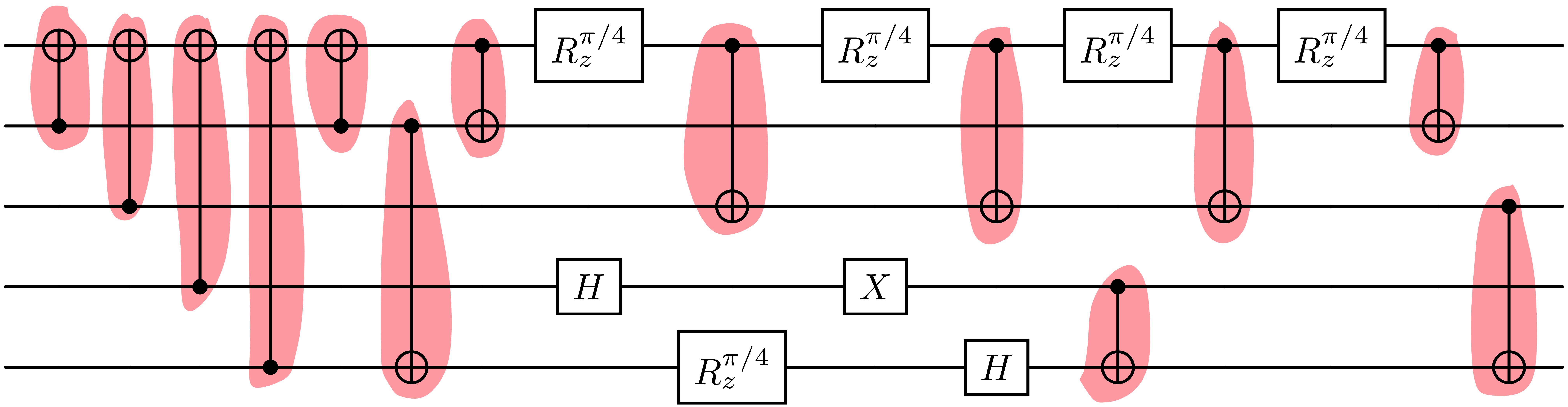
Optimization Objectives



NISQ

- total gate count
- 2q gate count

Optimization Objectives



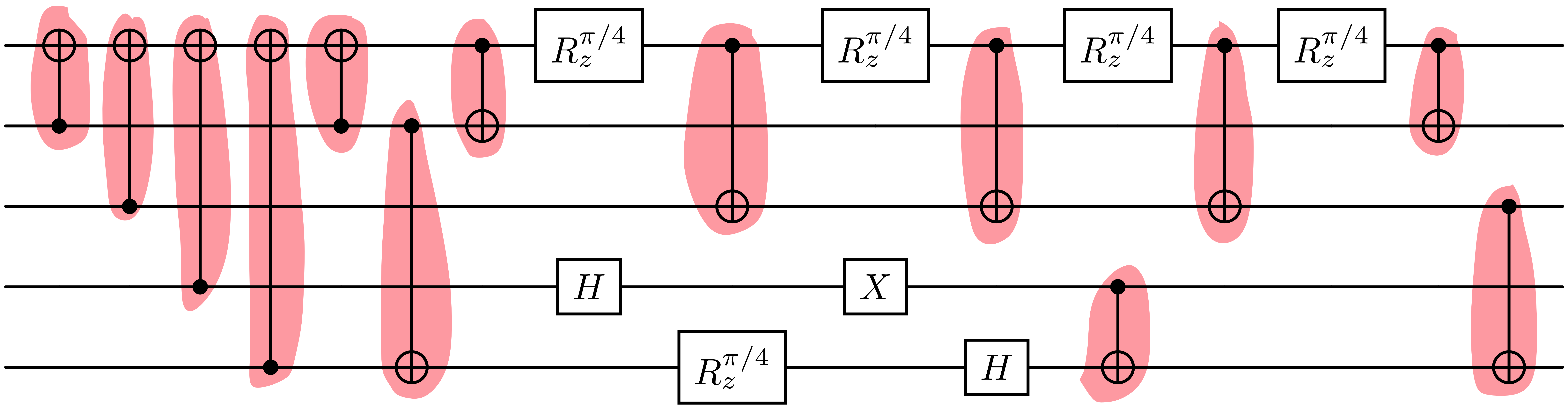
<u>NISQ</u>

- total gate count
- 2q gate count

<u>FTQC</u>

- T count and 2q gate count

Optimization Objectives



$$T = R_z^{\pi/4}$$

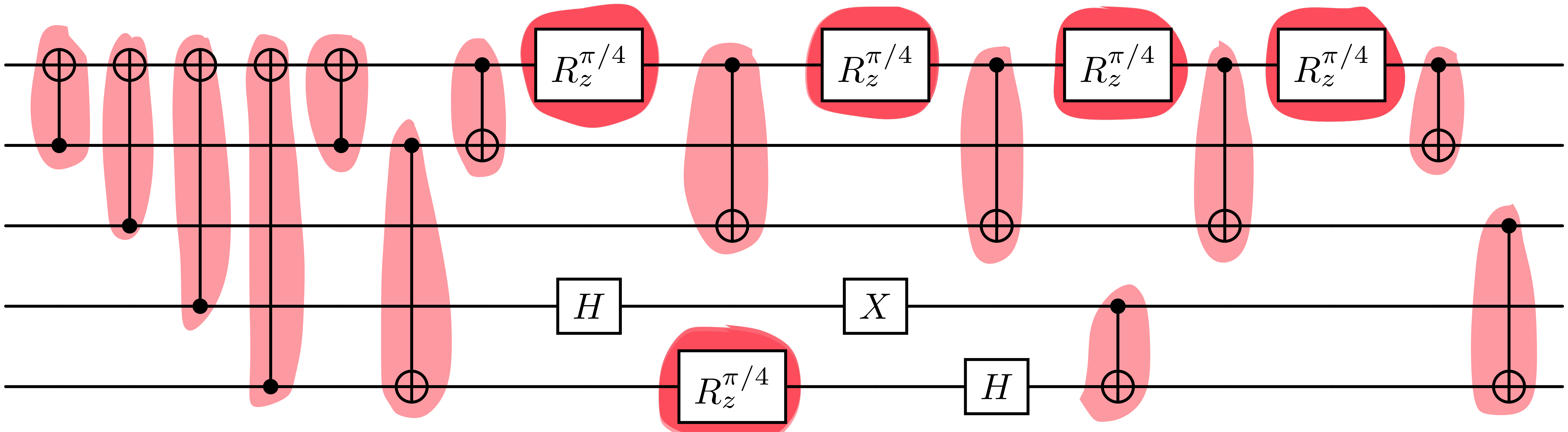
<u>NISQ</u>

- total gate count
- 2q gate count

<u>FTQC</u>

- T count and 2q gate count

Optimization Objectives



$T = R_z^{\pi/4}$

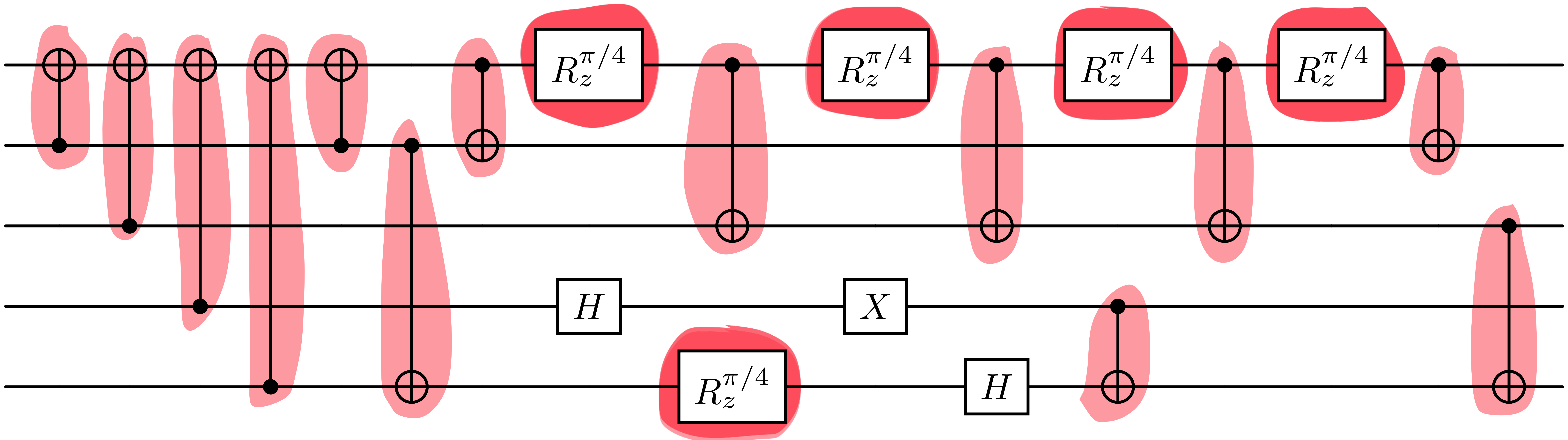
NISQ

- total gate count
- 2q gate count

FTQC

- T count and 2q gate count

Optimization Objectives



$T = R_z^{\pi/4}$

NISQ

- total gate count
- 2q gate count

FTQC

- T count and 2q gate count

and many more...

High-level problem

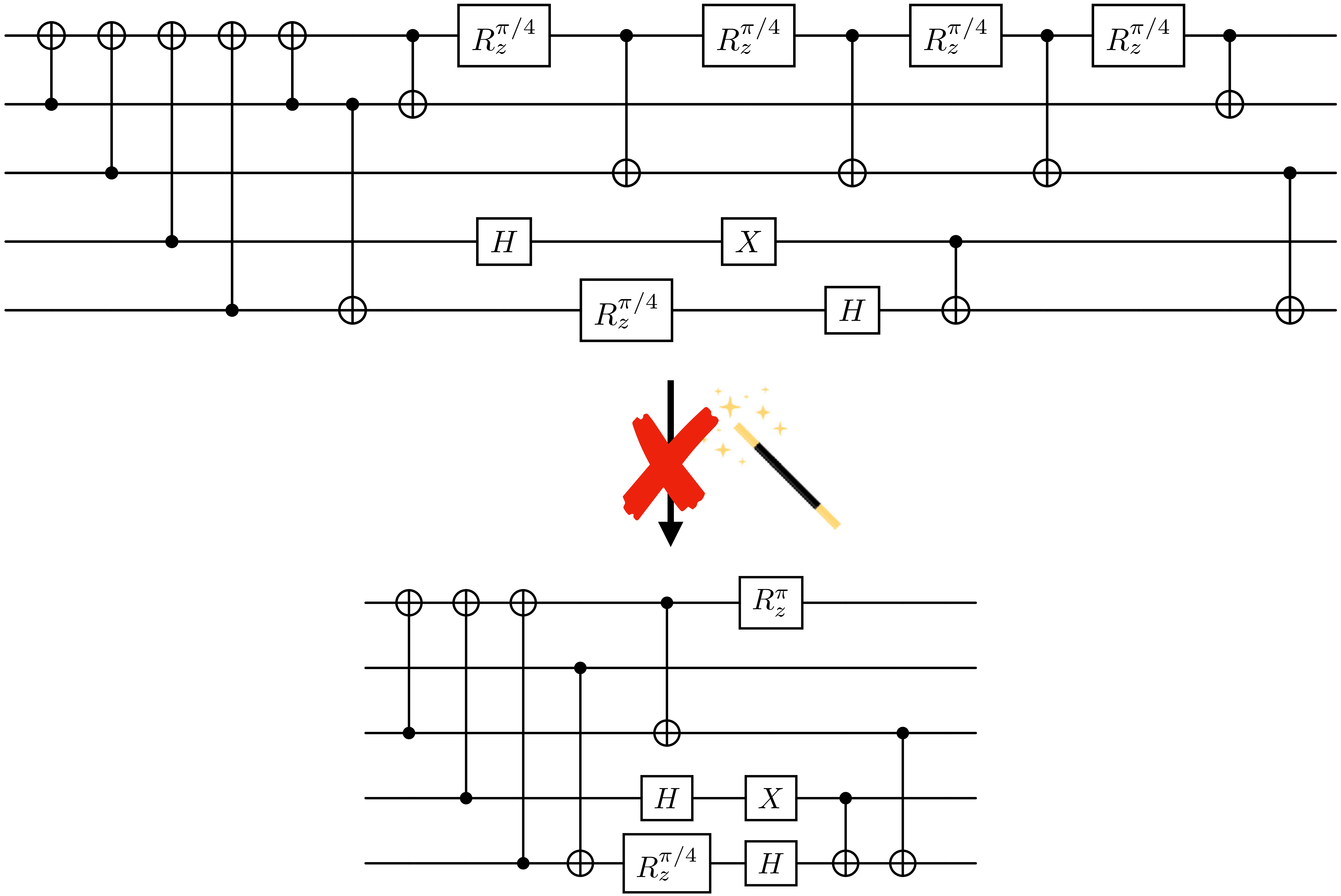
Given a circuit and optimization objective, output an equivalent circuit that minimizes the optimization objective.

Optimising quantum circuits is generally hard

John van de Wetering[1] and Matthew Amy[2]

[1]University of Amsterdam
[2]Simon Fraser University
July 28th 2024

General Approach



General Approach

Circuit Equivalence

Circuits are unitary matrices: $U^\dagger U = UU^\dagger = I$

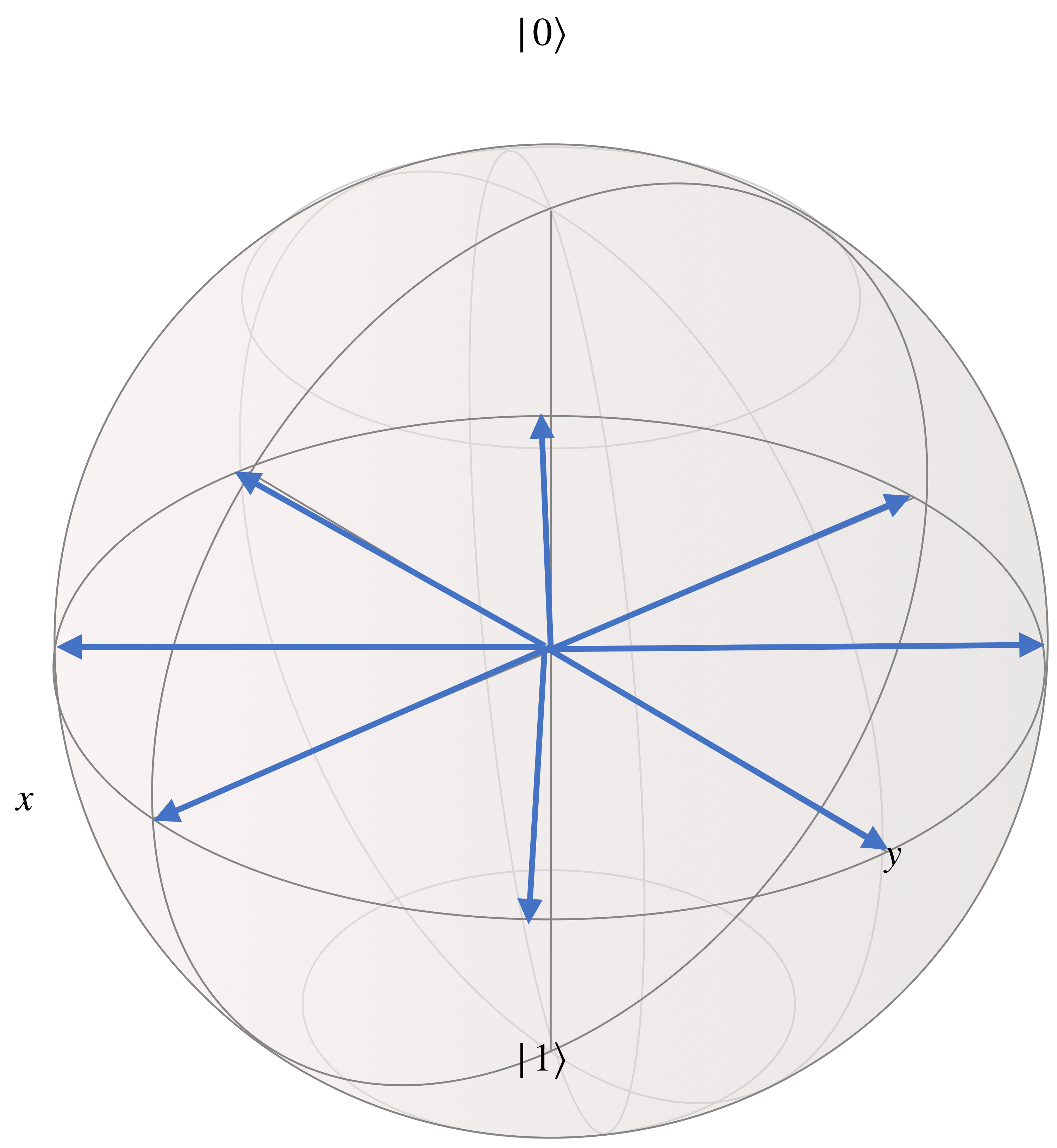
$$C_1 \equiv C_2 \iff U_{C_1} = U_{C_2}$$

"global phase"

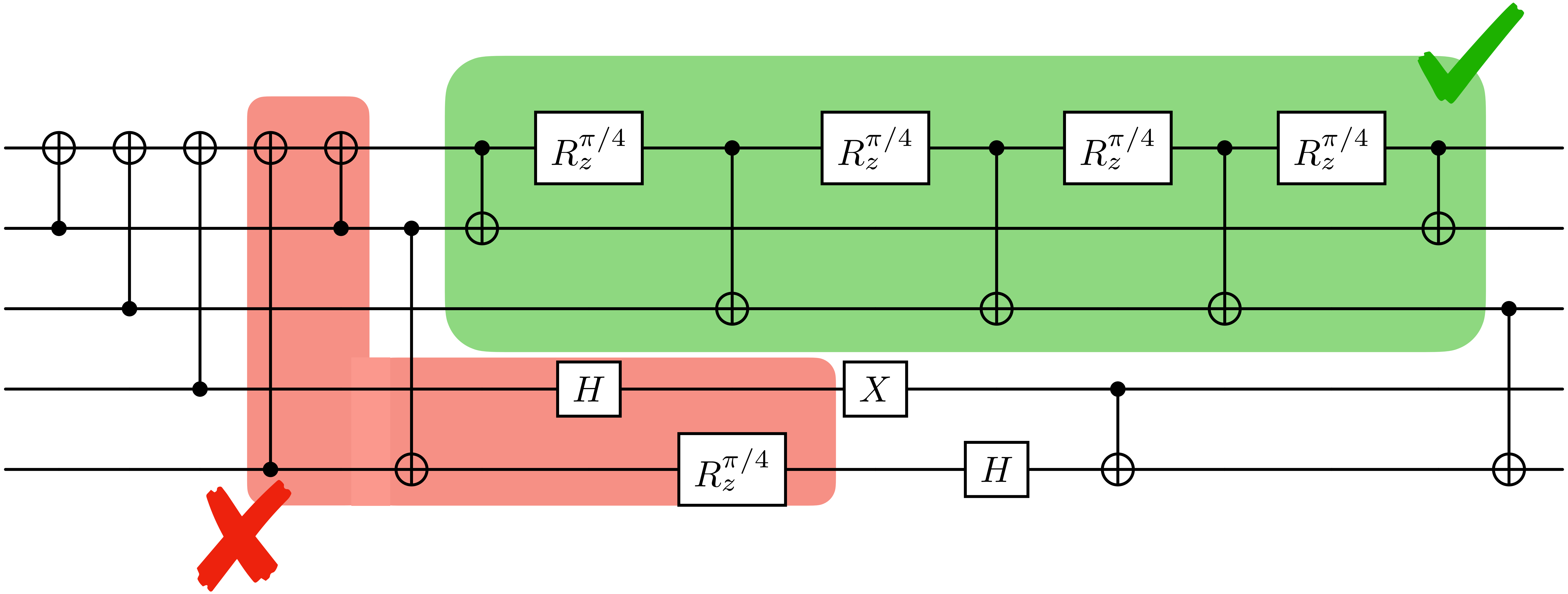
$$C_1 \equiv C_2 \iff \exists \varphi \,.\, U_{C_1} = e^{i\varphi} U_{C_2}$$

Hard for even a quantum computer to verify

(QMA-Complete)



Subcircuit



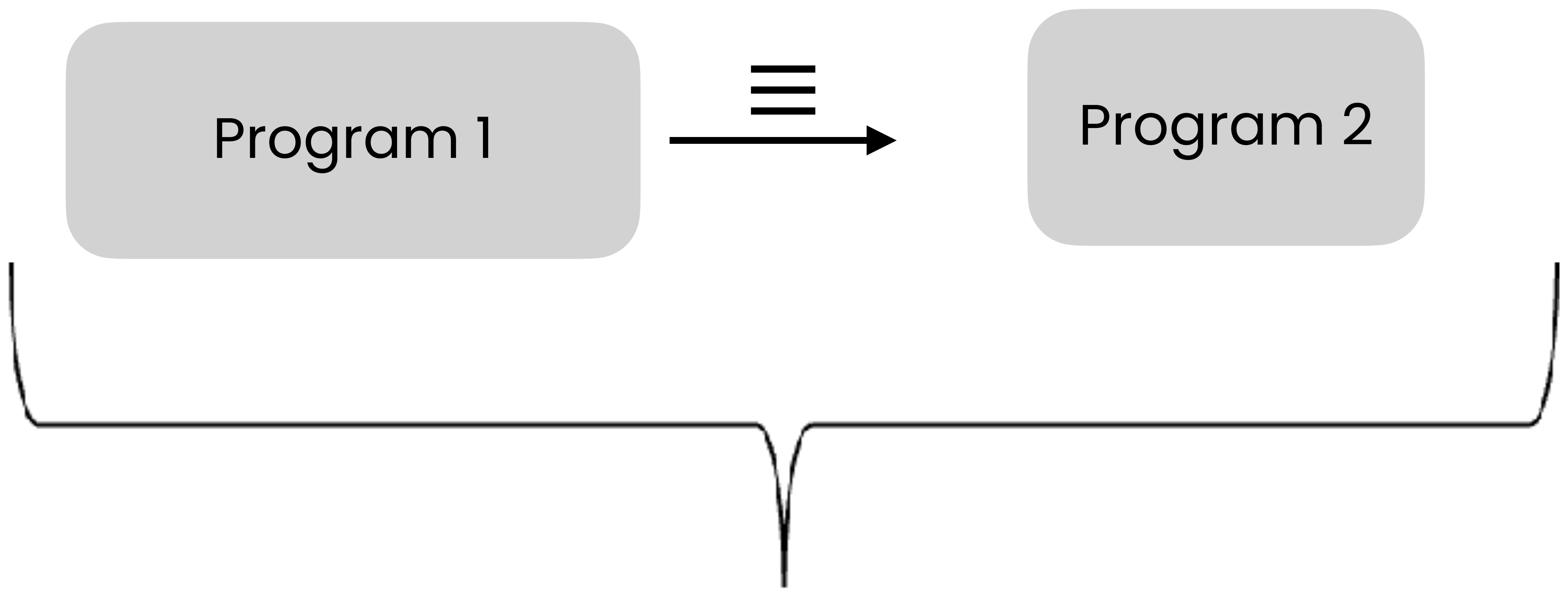
"convex subgraph"

i.e. any path between two vertices in the DAG also in subgraph

Rewrite Rules

Rewrite Rules

"peephole optimization"

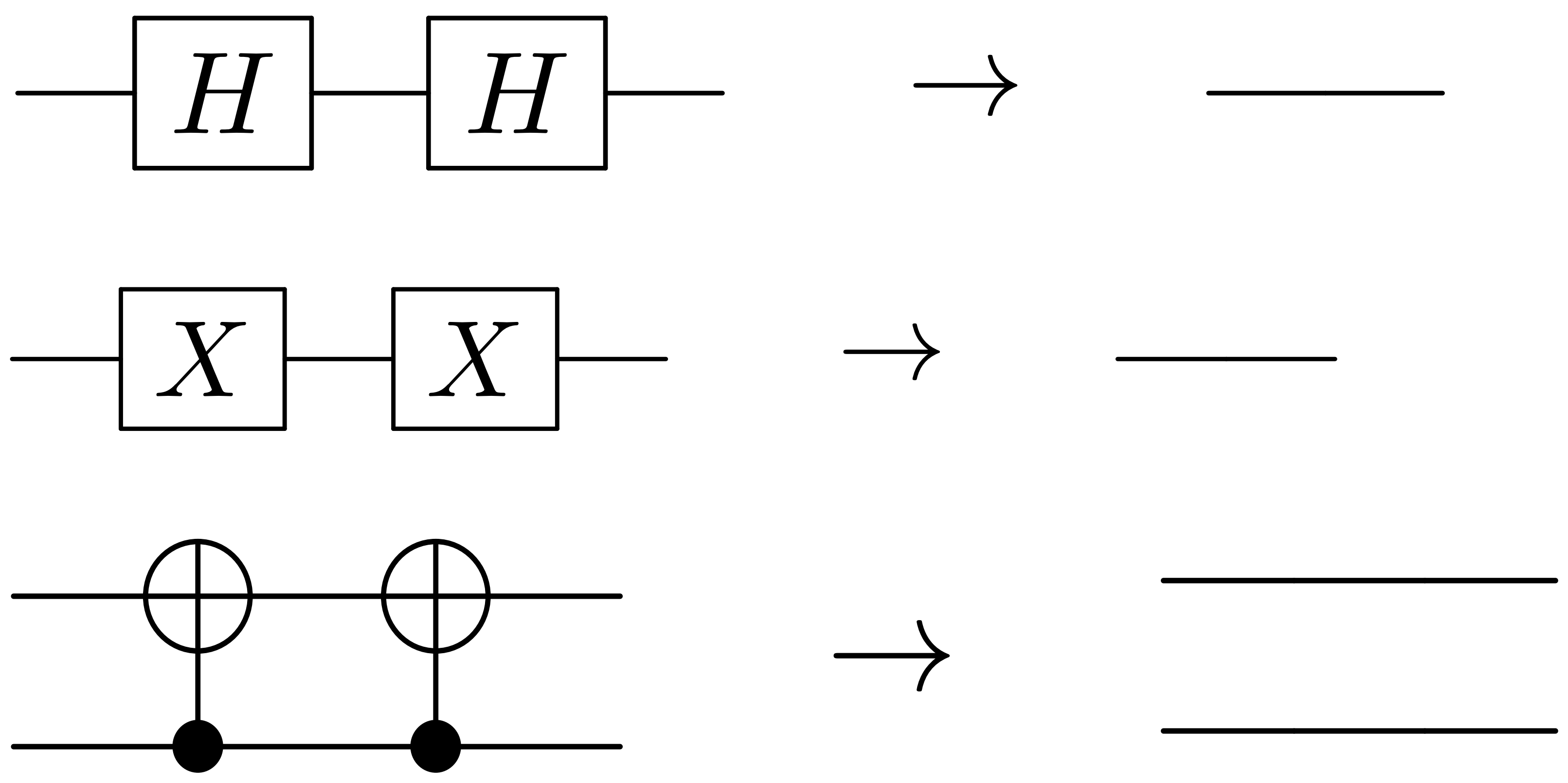


rewrite rule

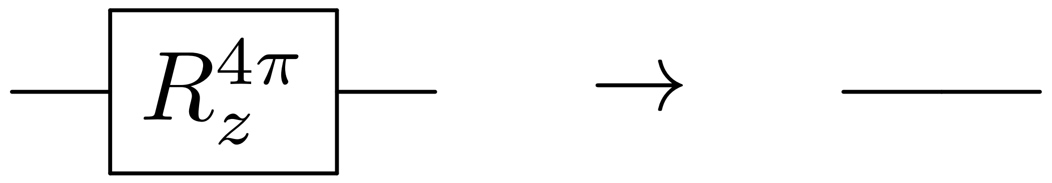
- Optimization: LHS and RHS same gate set
- Decomposition: LHS higher level than RHS

Simple Rewrite Rules

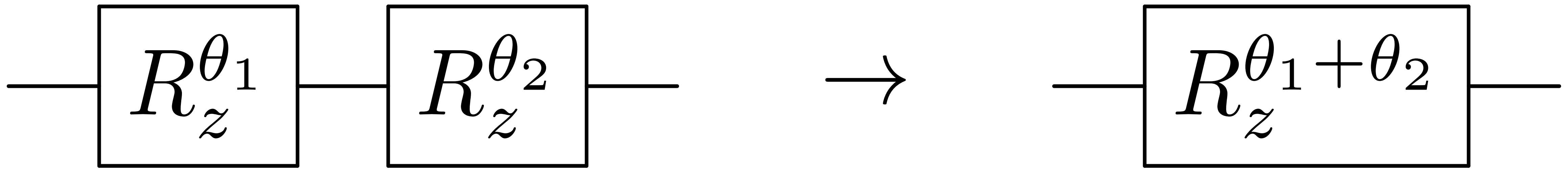
Cancel



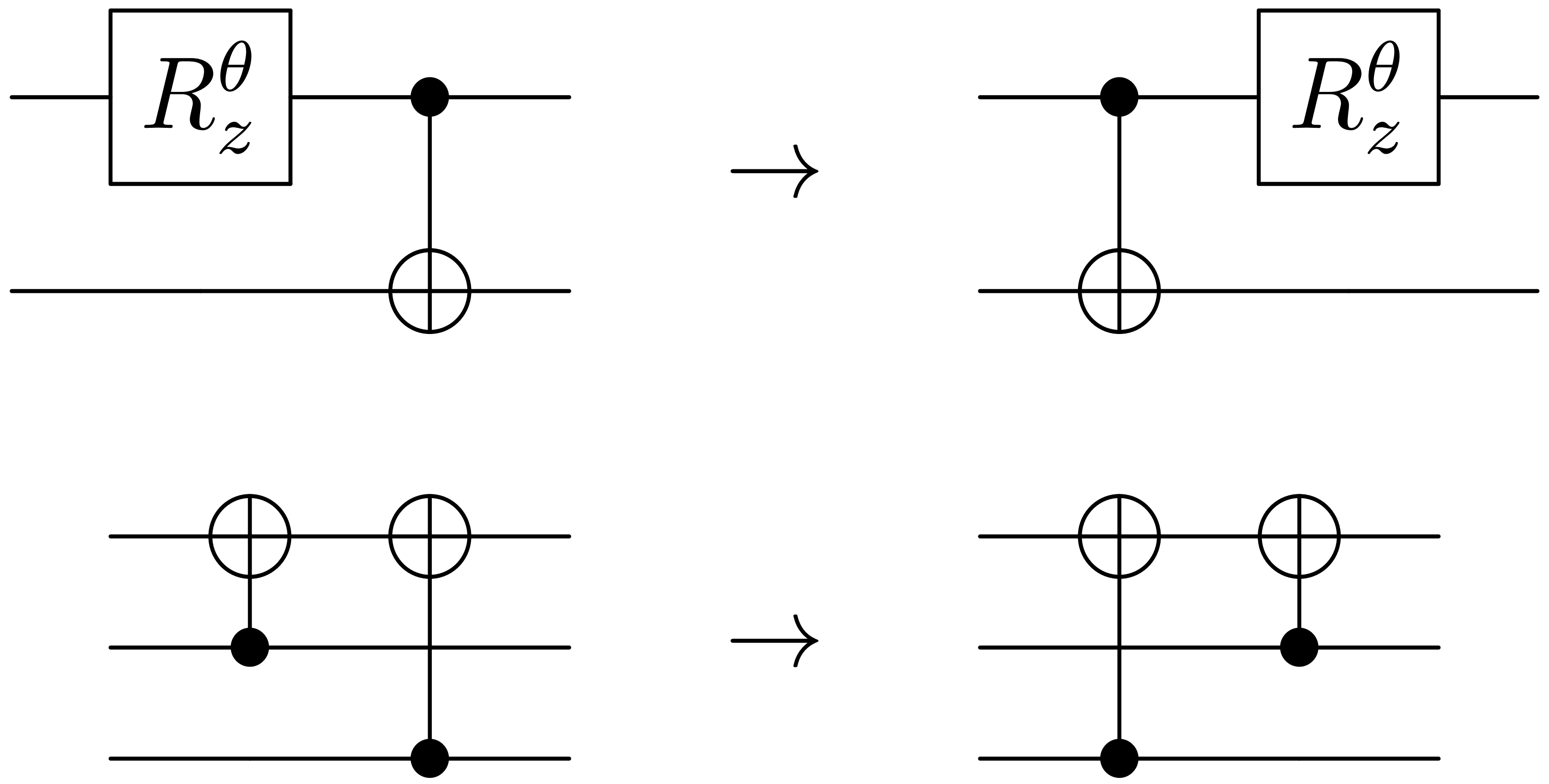
Global Phase



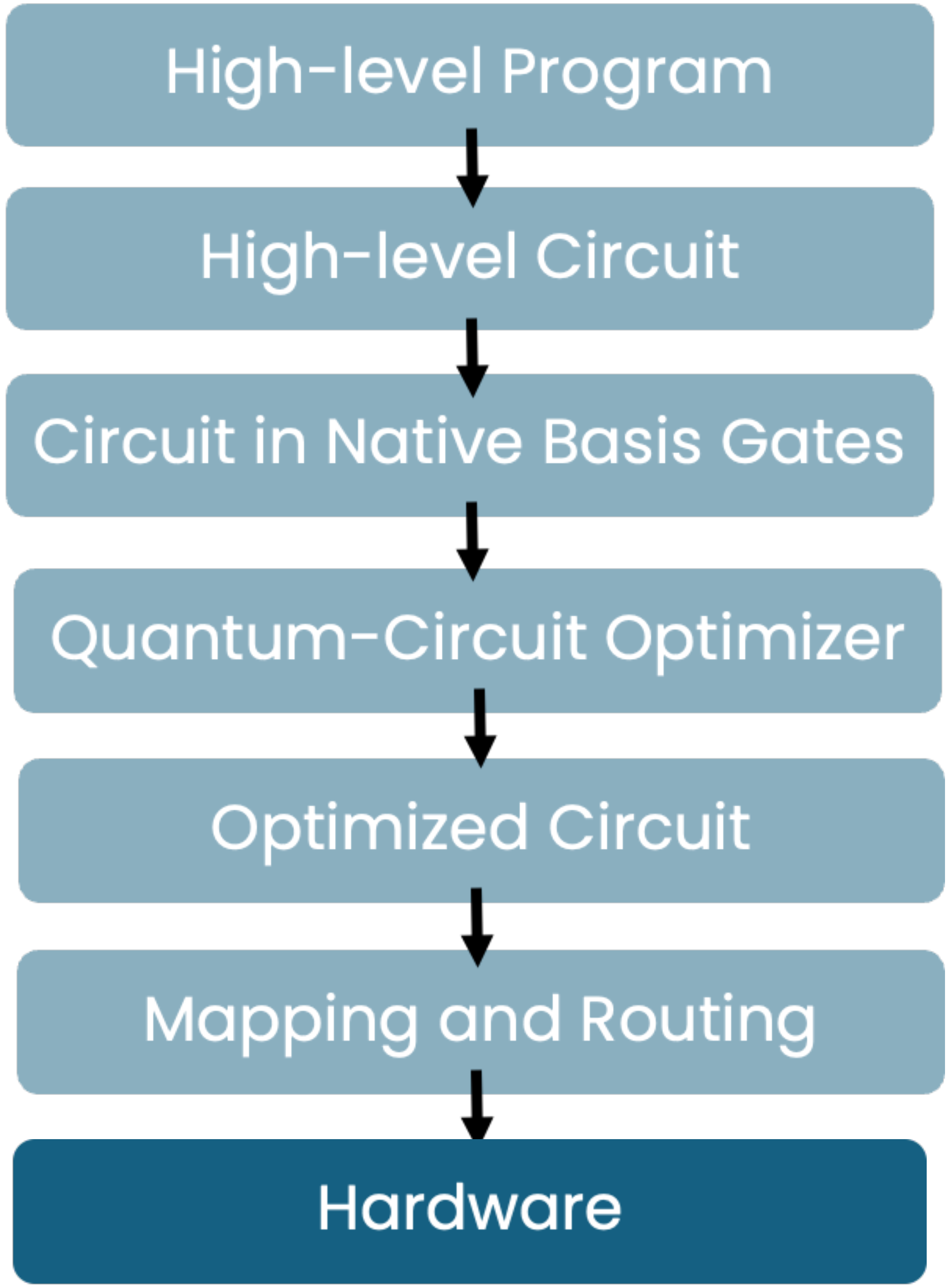
Merge



Commute



Diverse Hardware



Diverse Hardware

Ion Trap

IONQ

QUANTINUUM

Topological

Hardware

Superconducting

rigetti

OQC

Google Quantum AI

IBM

aws

~~U1, U2, U3, CX~~ → ~~Rz, SX, X, CX~~

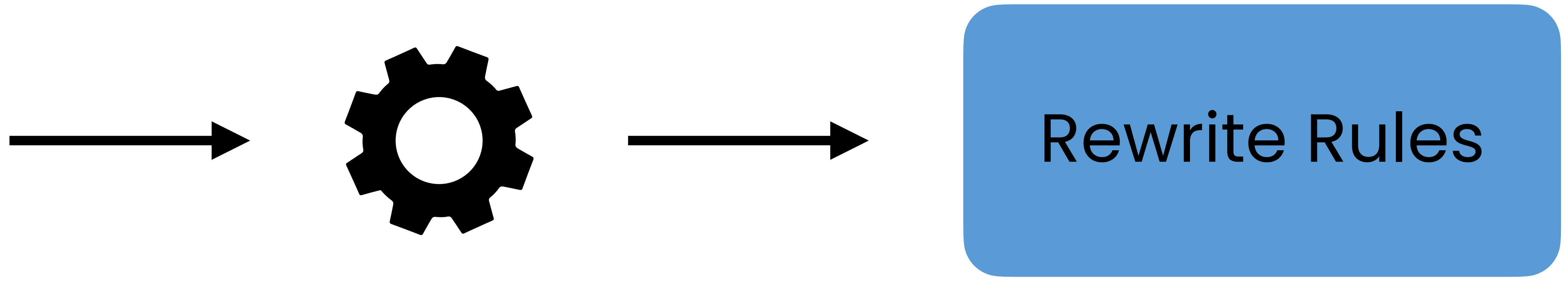
→ Rz, Rx, SX, X, CZ, Rzz

Neutral Atom

|QuEra⟩ Computing Inc.

Synthesis!

Arbitrary gate set



Naïve Synthesis

```
rules = []
circuits = enumerate(max_qubits, max_size)
for c1 in circuits:
  for c2 in circuits:
    if verify(c1, c2):
      rules.append(c1 → c2)
```

big

uh oh...

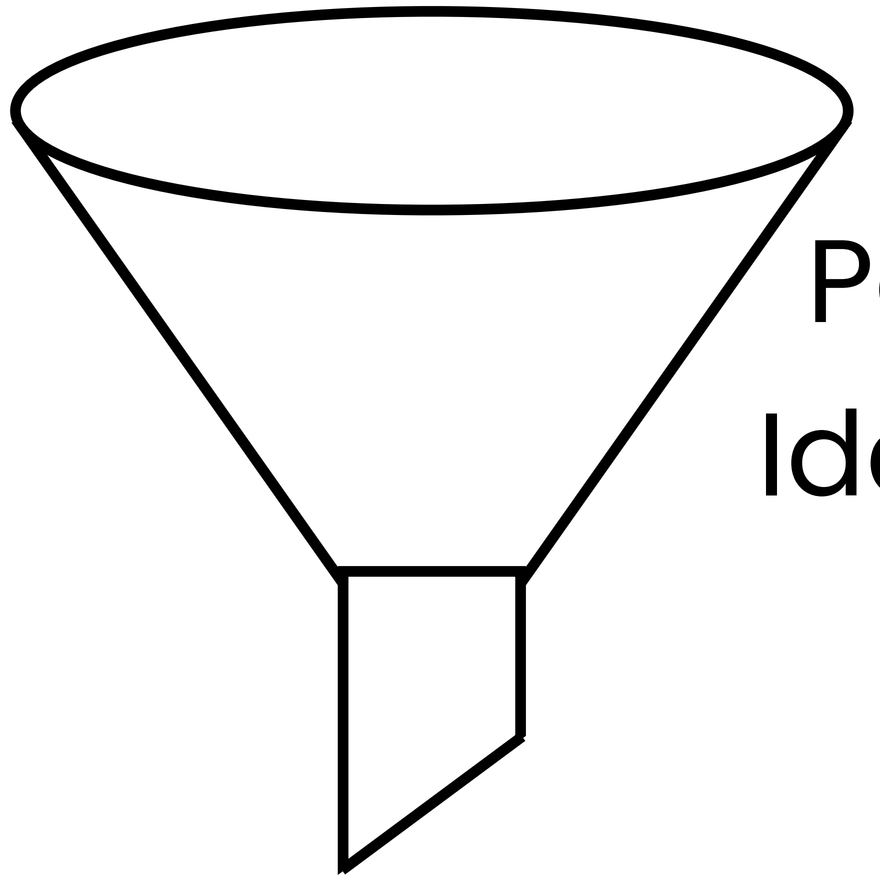
expensive

QUESO

Schwartz-Zippel Lemma for polynomial identity testing (PIT)

Feynman's path integral formulation for quantum mechanics

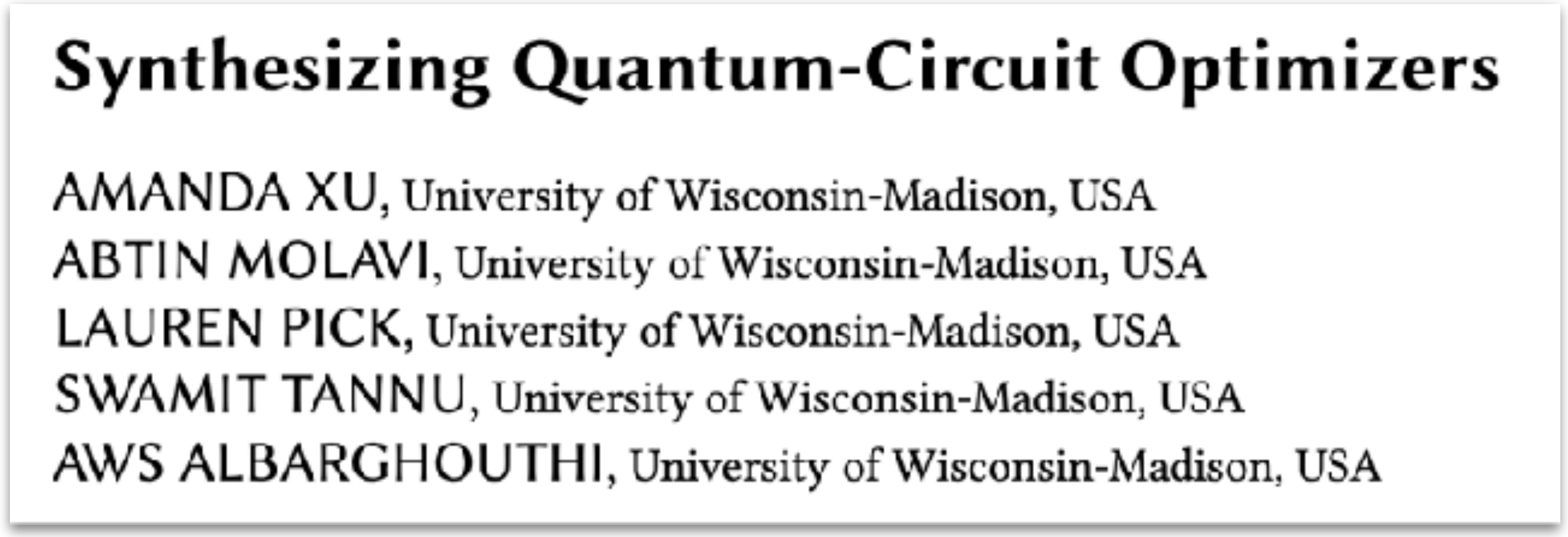
(Symbolic) Circuit → Polynomial →



Polynomial Identity Filter (PIF)

↓

Circuit Equivalence Classes



Synthesizing Quantum-Circuit Optimizers

AMANDA XU, University of Wisconsin-Madison, USA
ABTIN MOLAVI, University of Wisconsin-Madison, USA
LAUREN PICK, University of Wisconsin-Madison, USA
SWAMIT TANNU, University of Wisconsin-Madison, USA
AWS ALBARGHOUTHI, University of Wisconsin-Madison, USA

@ PLDI'23

Path-sum Semantics

Algebraically represent semantics of quantum gates

$$R_z(\theta) : |x\rangle \to e^{i(2x-1)\theta}|x\rangle$$

$$CNOT : |x_1x_2\rangle \to |x_1(x_1 \oplus x_2)\rangle$$

amplitude state

More generally: $|\mathbf{x}\rangle \to \phi(\mathbf{x}, \rho)\,|f(\mathbf{x})\rangle$

Simple Circuit as Polynomial

Circuit	Polynomial
`Rz(`θ`) q`$_1$`;`	$e^{i(2(0)-1)\theta}\|0\rangle$

Simple Circuit as Polynomial

<u>Circuit</u>

`Rz(`θ`) q`$_1$`;`

<u>Polynomial</u>

$e^{-i\theta}|0\rangle$

Simple Circuit as Polynomial

Circuit

`Rz(θ) q1;` — $\mathbf{Rz}(\theta)\ \mathbf{q}_1;$

Polynomial

$$e^{-i\theta}|0\rangle + e^{i\theta}|1\rangle$$

Polynomial Identity Testing (PIT)

$\mathbf{x} \in \mathbb{C}^3$

$$p_1(\mathbf{x}) = x_1 x_2$$

$$p_2(\mathbf{x}) = x_1 x_2 + x_3 x_2$$

How can we check $p_1 \equiv p_2$?

Schwartz-Zippel Lemma

$p_1(\mathbf{x}) = x_1 x_2$

$p_2(\mathbf{x}) = x_1 x_2 + x_3 x_2$

finite $R \subset \mathbb{C}$

α_{x_1} α_{x_2} α_{x_3}

$p_1(\boldsymbol{\alpha}) = \alpha_{x_1}\alpha_{x_2}$

$p_2(\boldsymbol{\alpha}) = \alpha_{x_1}\alpha_{x_2} + \alpha_{x_3}\alpha_{x_2}$

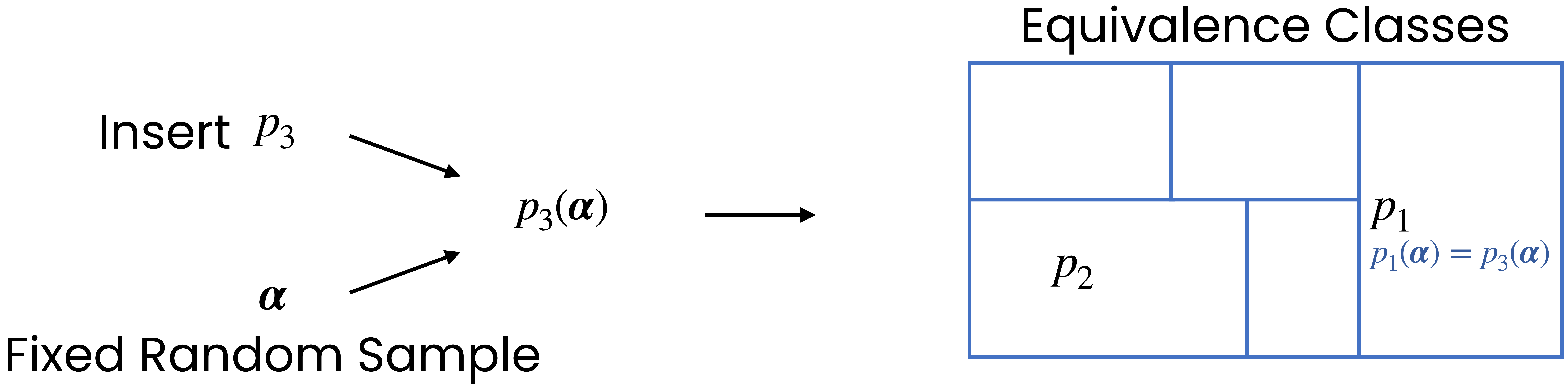
Check $p_1(\boldsymbol{\alpha}) = p_2(\boldsymbol{\alpha})$

small

If $p_1 \neq p_2$, then the probability that $p_1(\boldsymbol{\alpha}) = p_2(\boldsymbol{\alpha})$ is at most $\dfrac{d}{|R|}$ (max total degree)

very large

Polynomial Identity Filter (PIF)



QUESO's Improvements

```
rules = []
circuits = enumerate(max_qubits, max_size)
for c1 in circuits:
  for c2 in circuits:
    if verify(c1, c2):
      rules.append(c1 → c2)
```

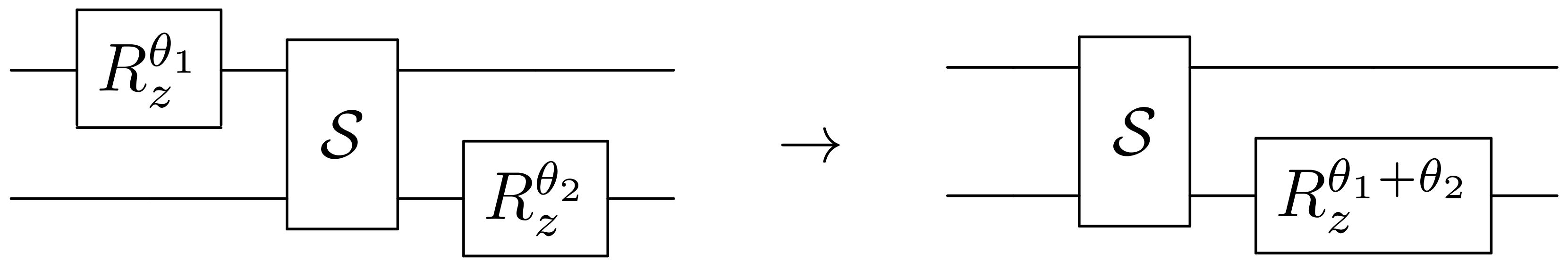
insert into PIF

big

uh oh...

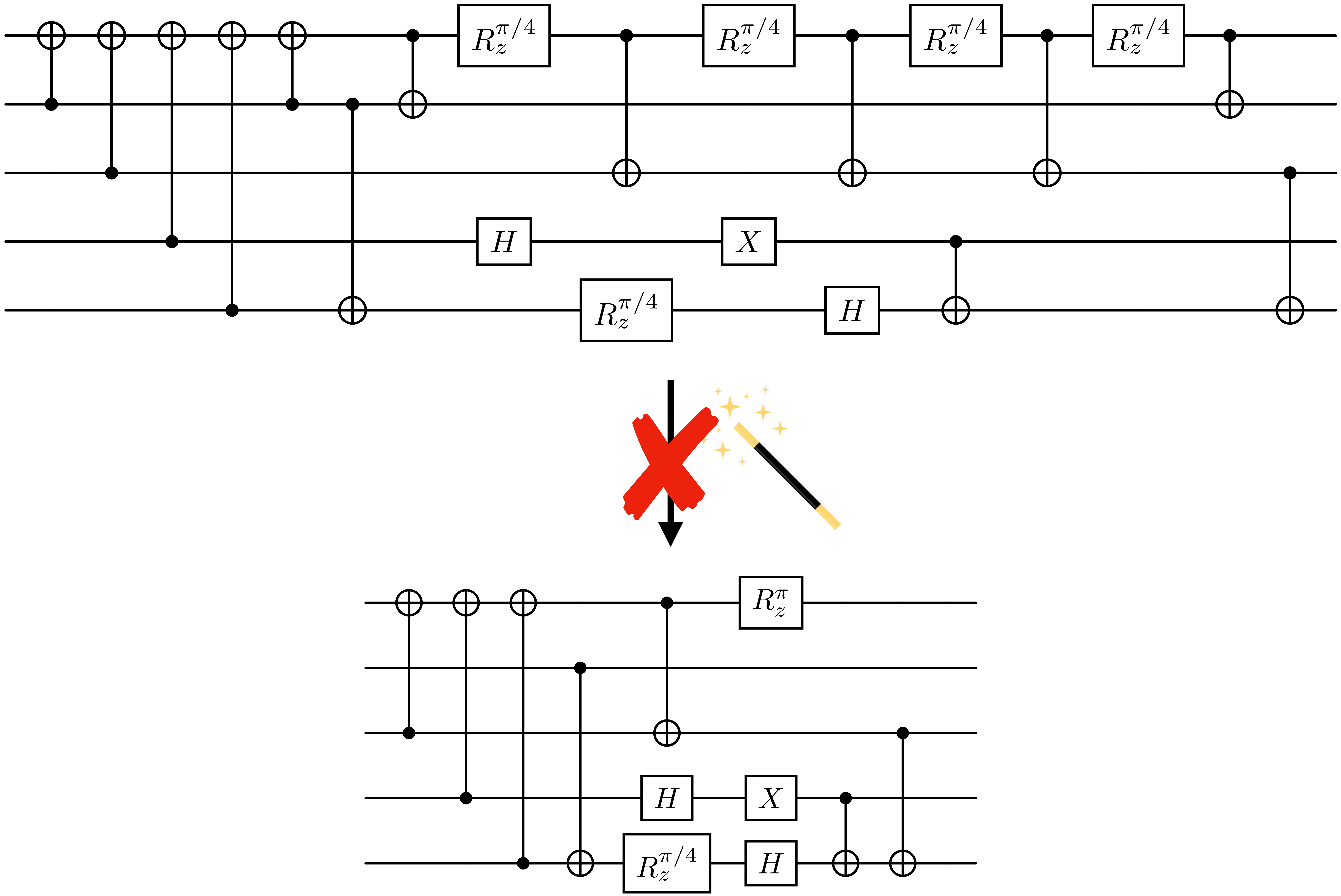
expensive

Efficient equivalence check between circuit pairs!

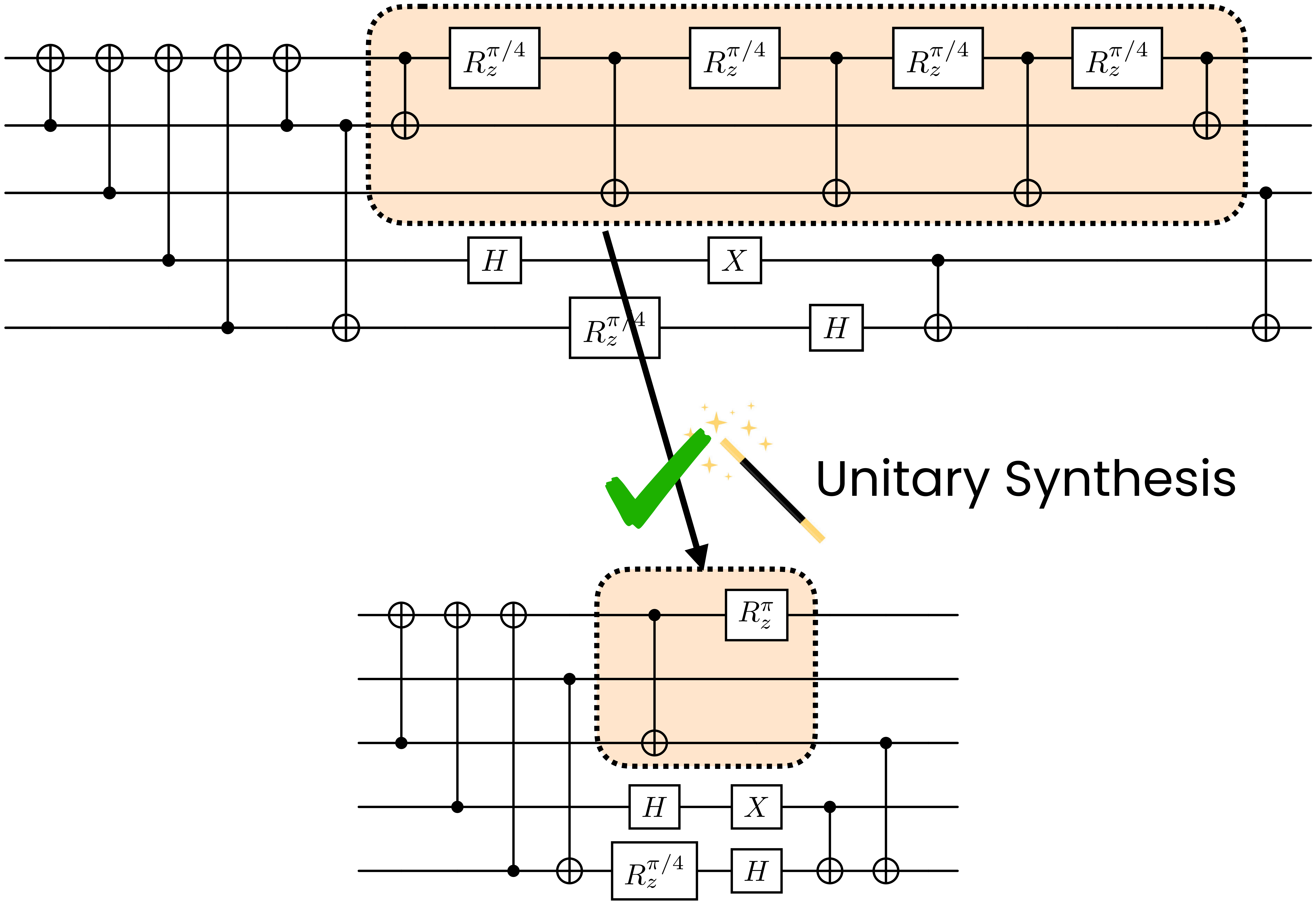
bonus:  (advanced symbolic rules)

Circuit Resynthesis

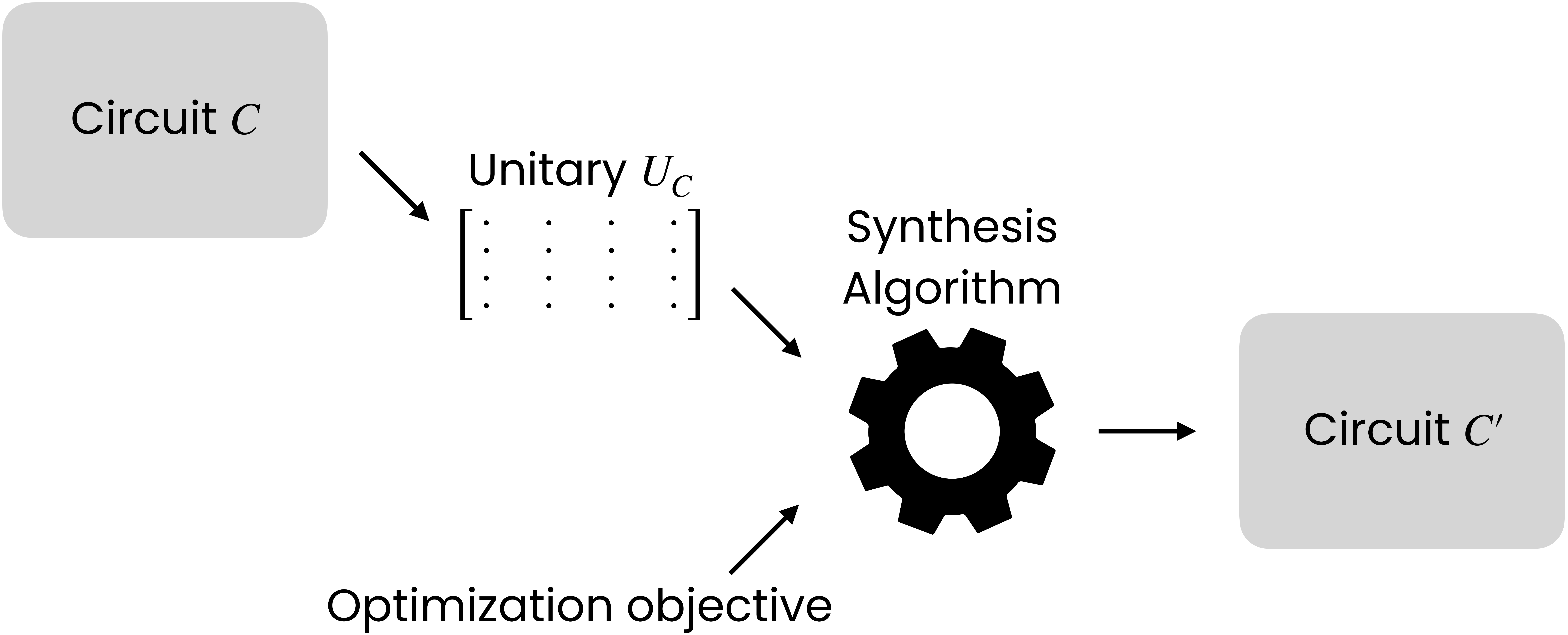
Circuit Resynthesis



Circuit Resynthesis

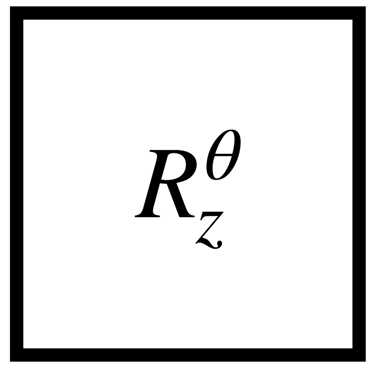


Unitary Synthesis

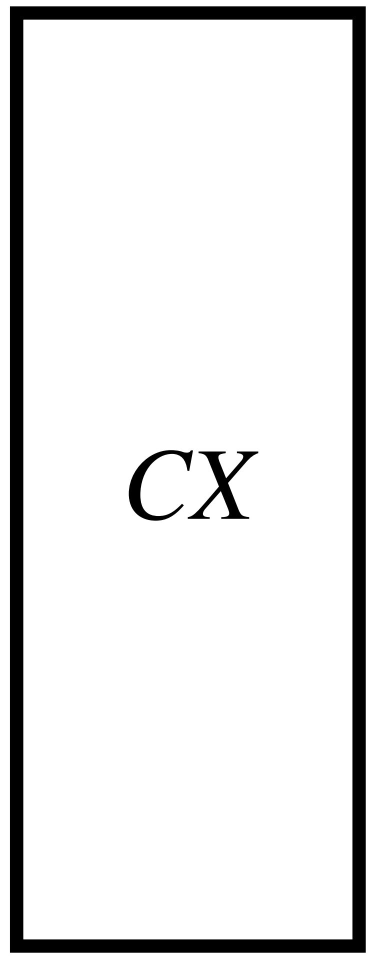


Guarantee $C \equiv C'$

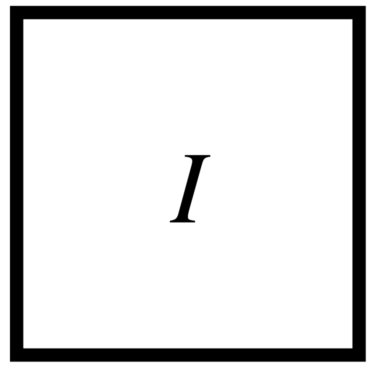
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

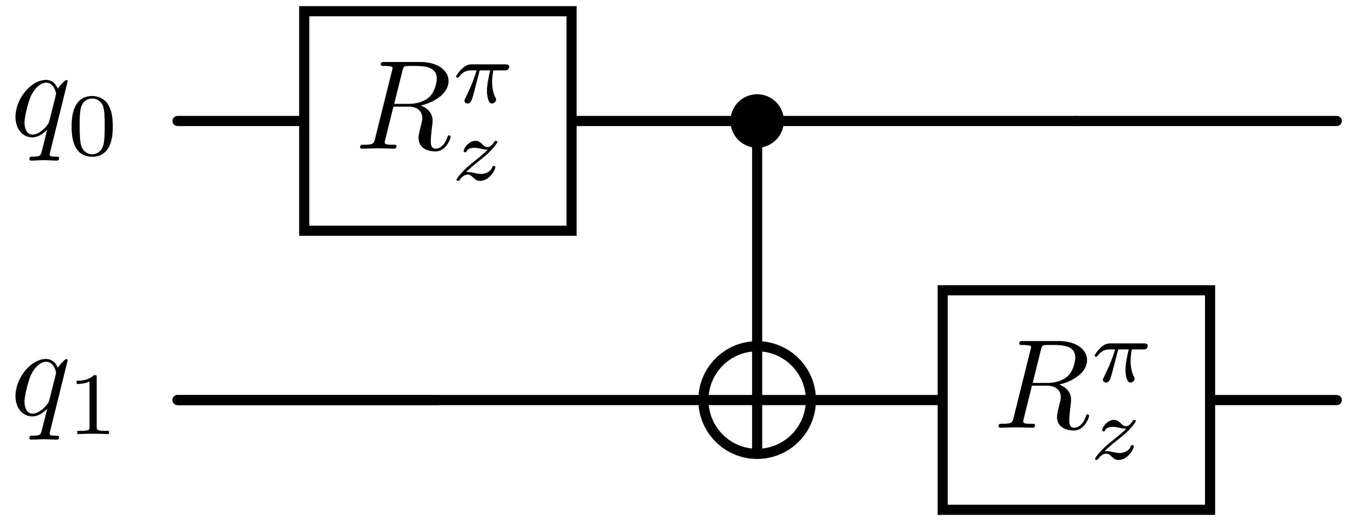


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

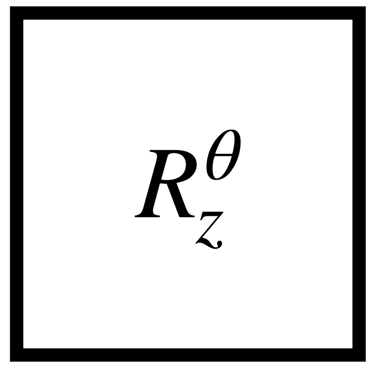


$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

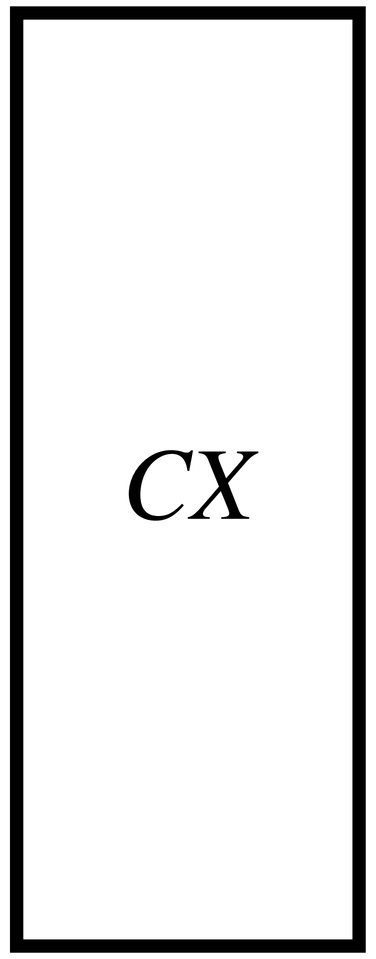
$C :=$



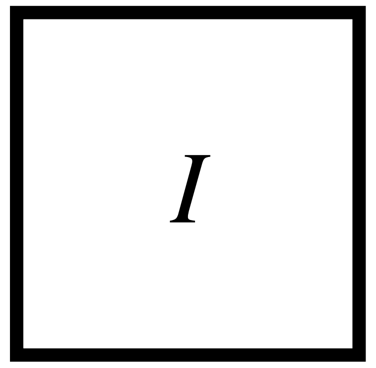
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

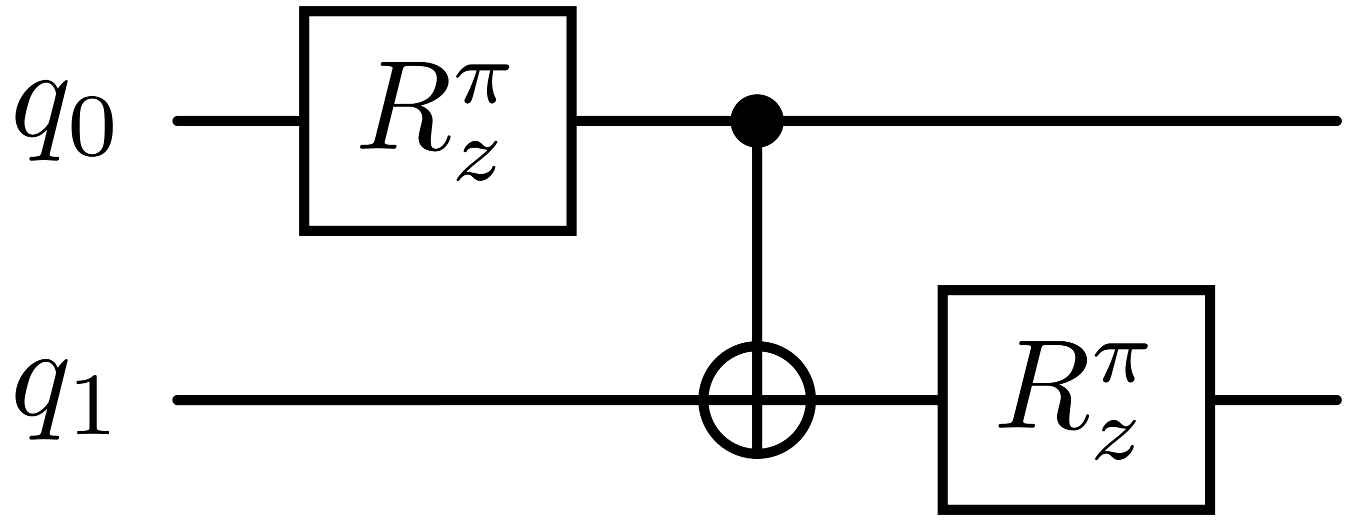


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$



$U_C =$ $U_{R_z^\pi}$

Circuit to Unitary

R_z^{θ}

$$U_{R_z^{\theta}} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

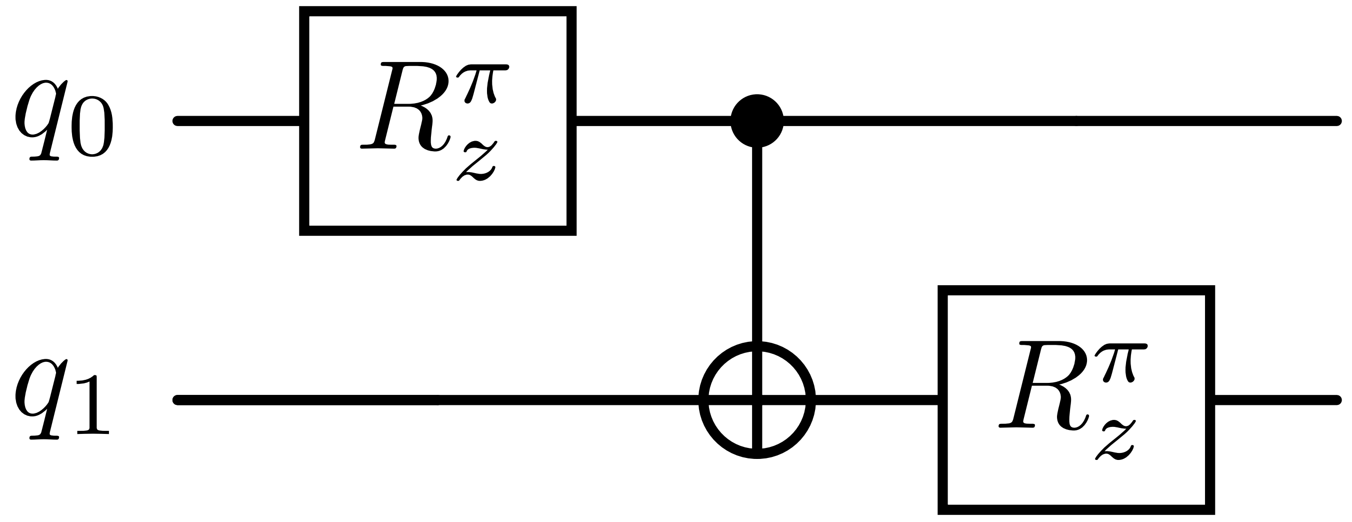
CX

$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

I

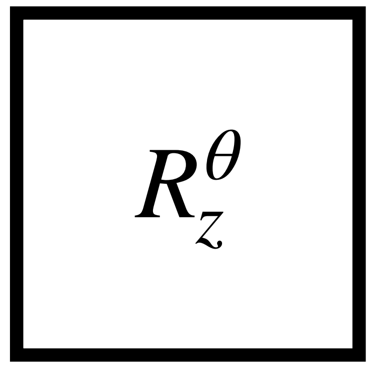
$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$

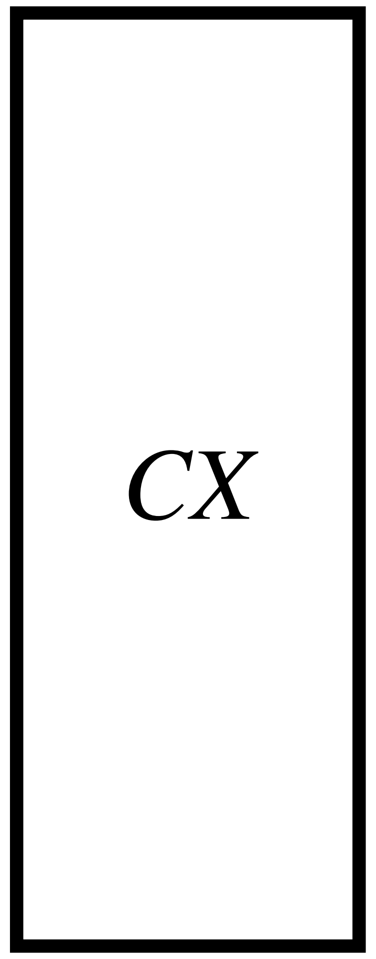


$\underset{2^2 \times 2^2}{U_C} =$ $U_{R_z^{\pi}}$

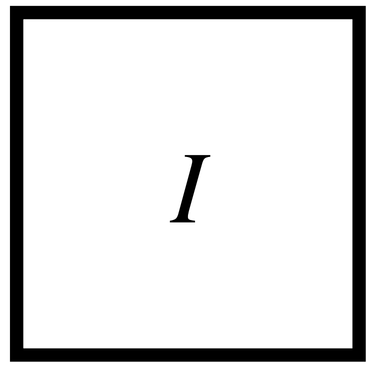
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

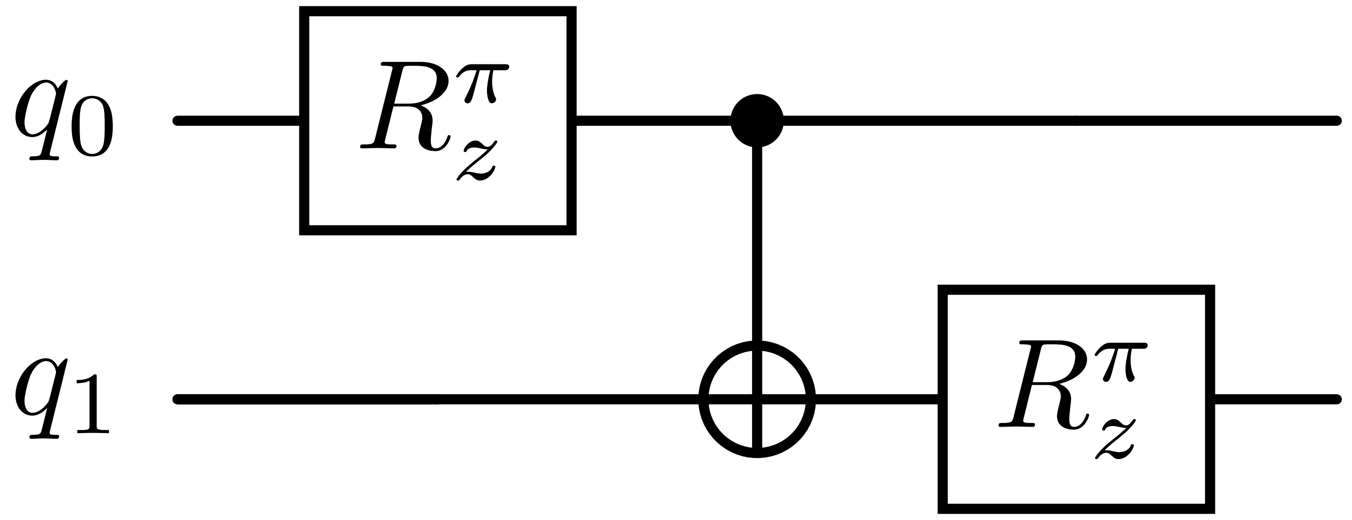


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$



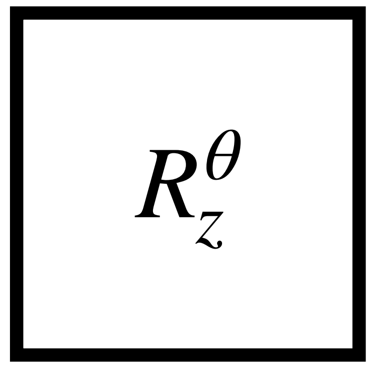
$U_C =$

$2^2 \times 2^2$

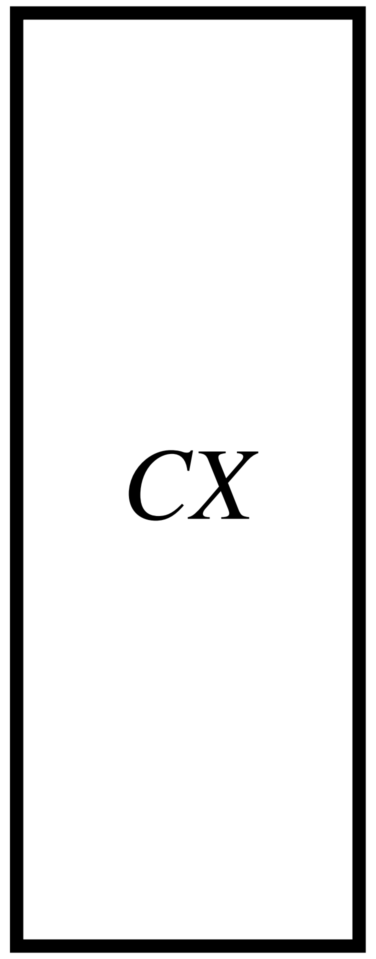
$U_{R_z^\pi}$

2×2

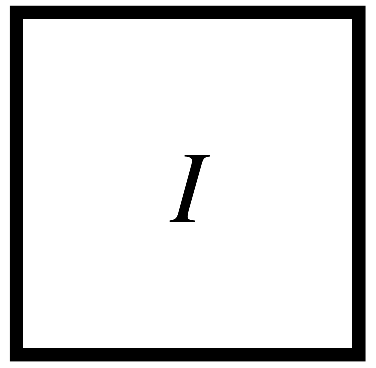
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

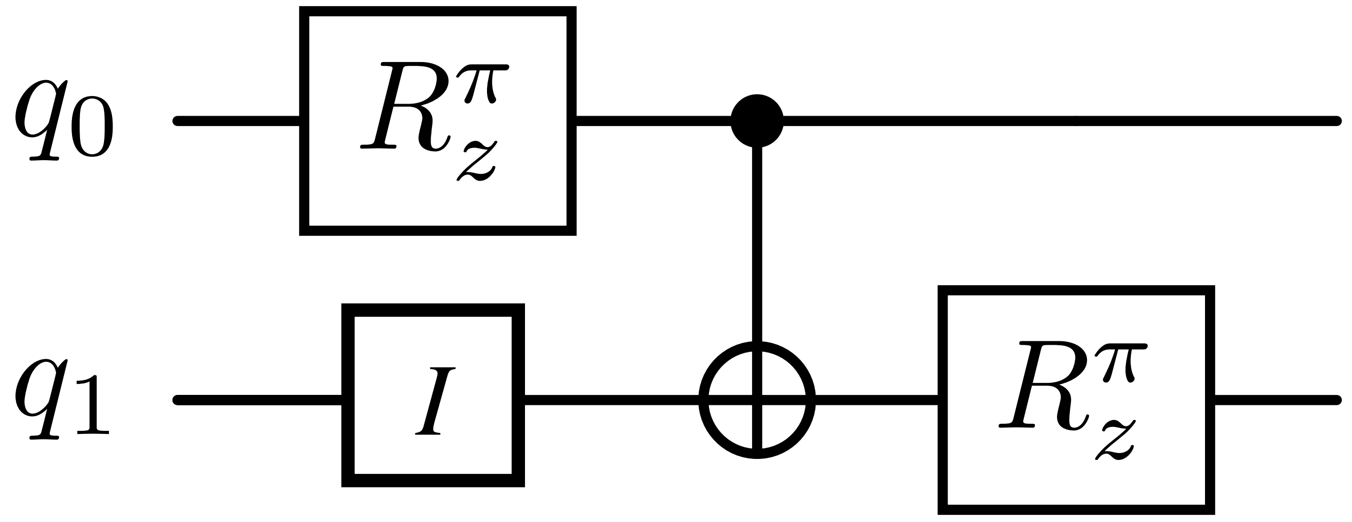


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



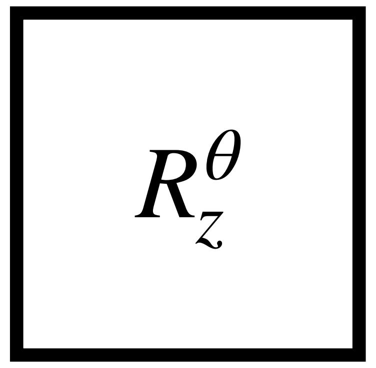
$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$

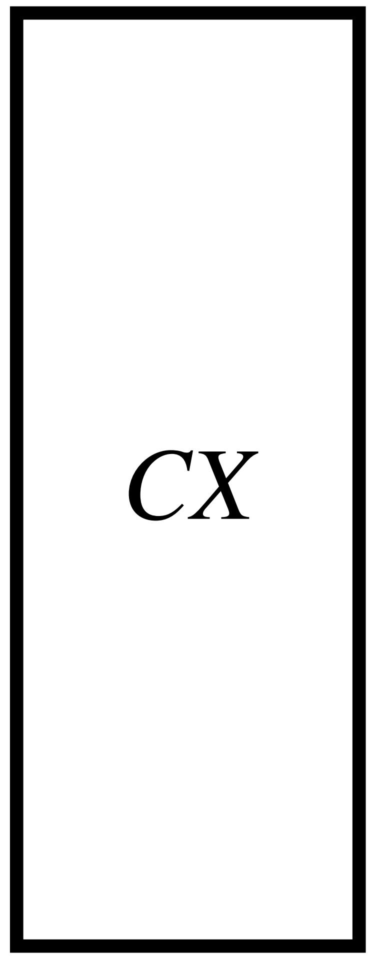


$\underset{2^2 \times 2^2}{U_C} =$ $\underset{2 \times 2}{U_{R_z^\pi}}$

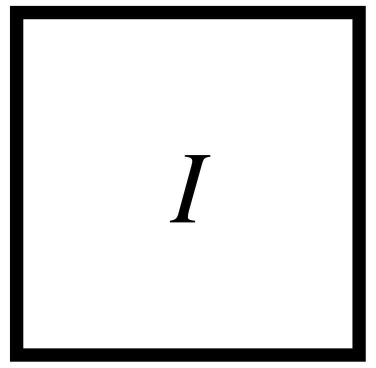
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

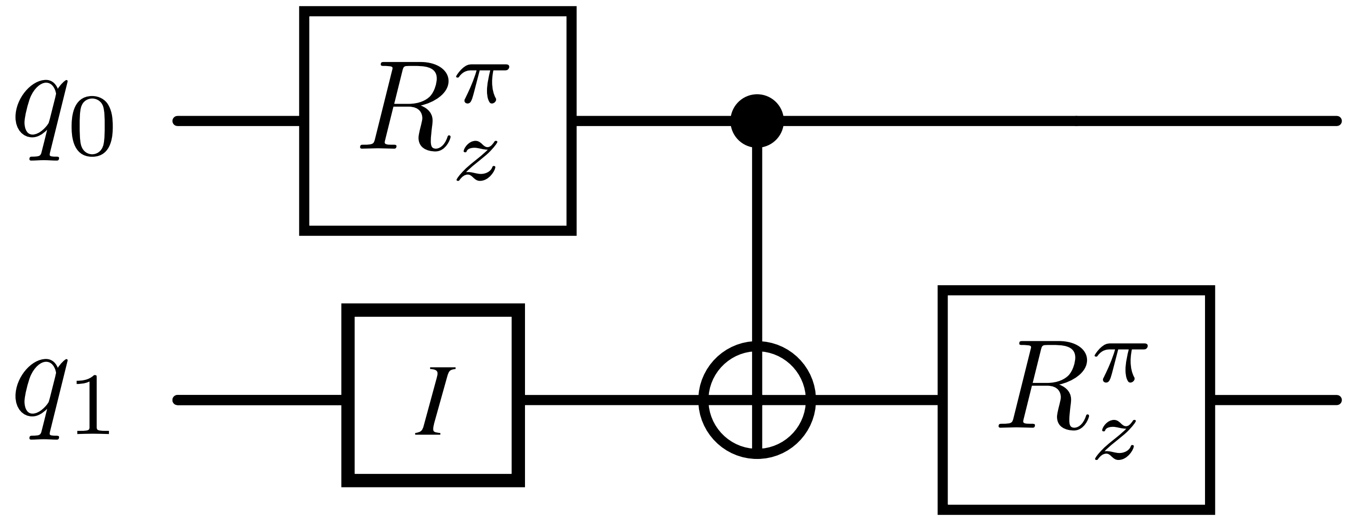


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



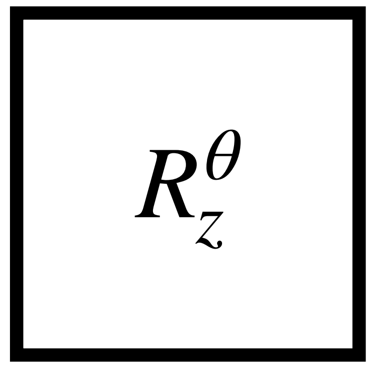
$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$

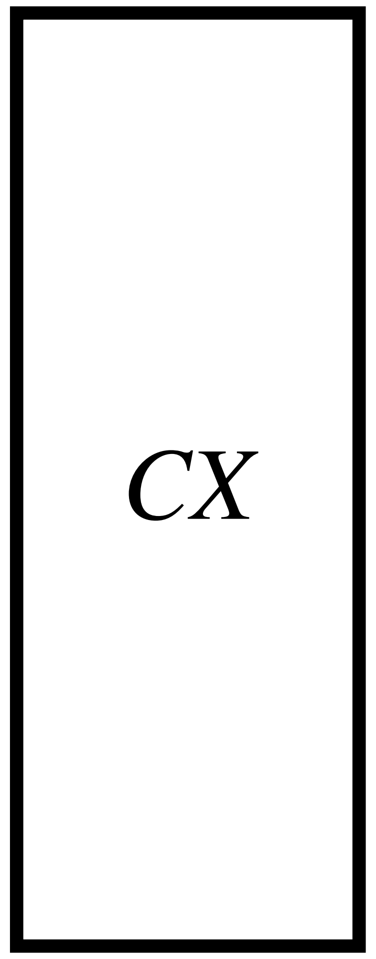


$$\underset{2^2 \times 2^2}{U_C} = \qquad (\underset{2 \times 2}{U_{R_z^\pi}} \otimes U_I)$$

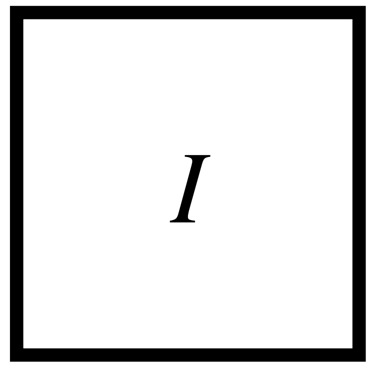
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

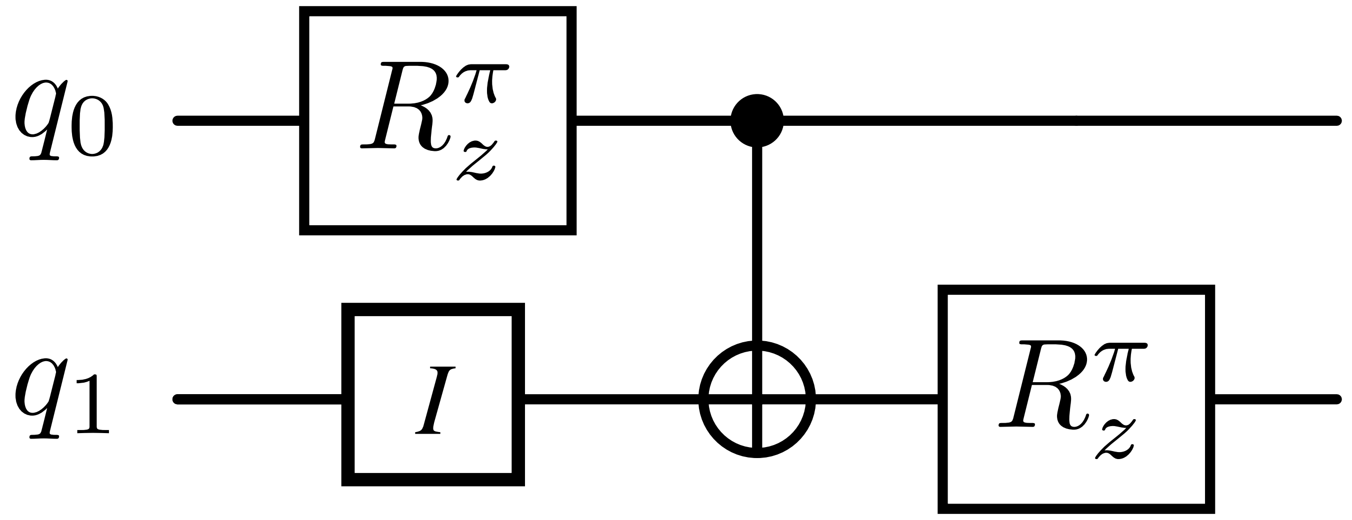


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



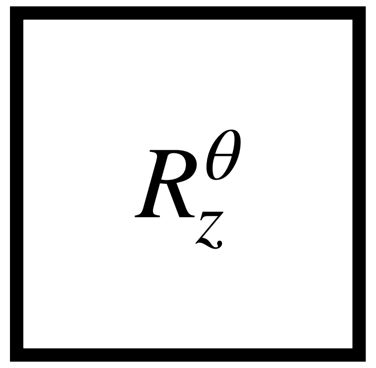
$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$C :=$

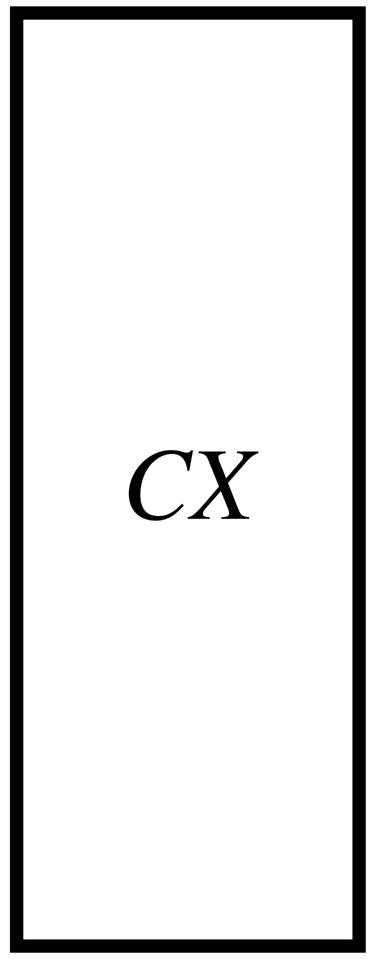


$$\underset{2^2 \times 2^2}{U_C} = \qquad \underbrace{(\underset{2\times 2}{U_{R_z^\pi}} \otimes \underset{2\times 2}{U_I})}_{2^2 \times 2^2}$$

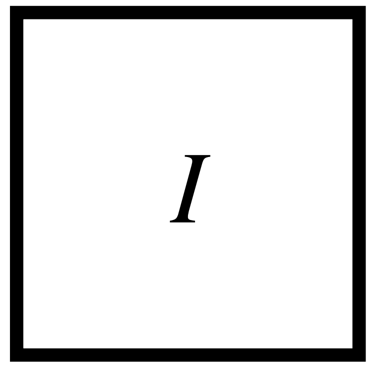
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

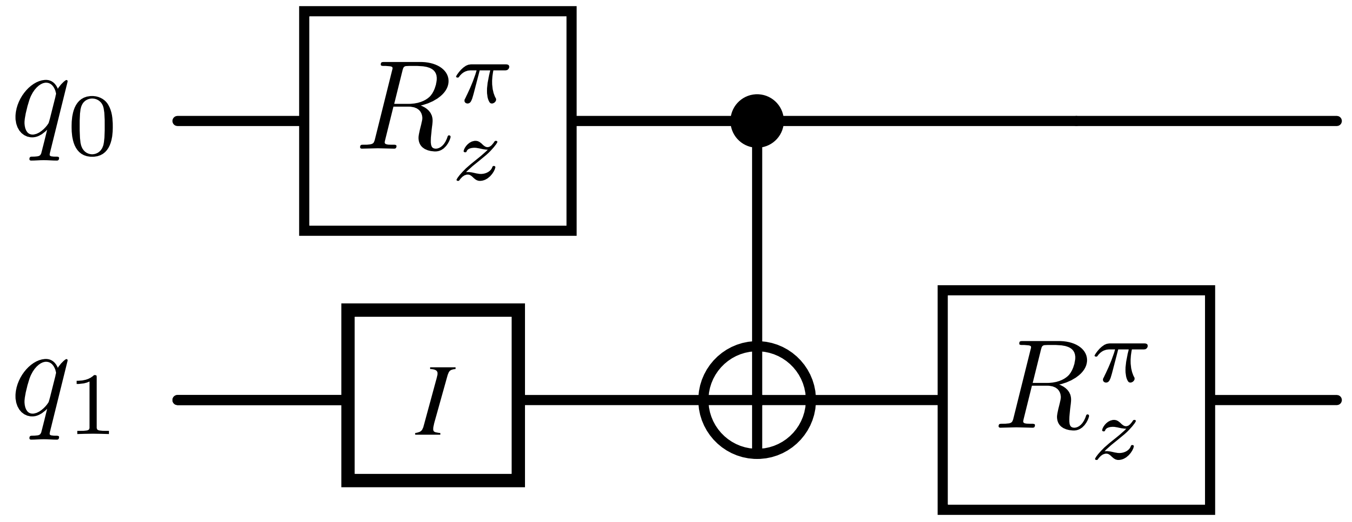


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

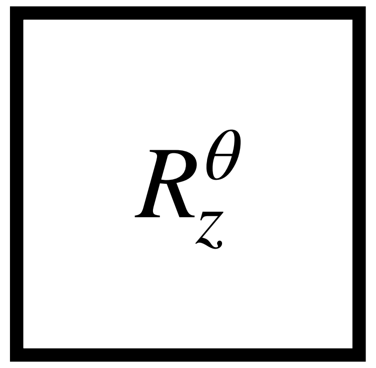
$C :=$



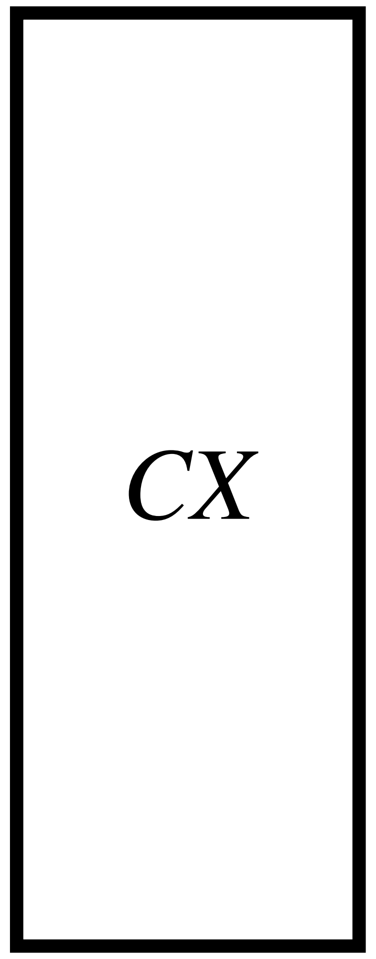
$$\underset{2^2 \times 2^2}{U_C} = \qquad \underbrace{(\underset{2\times 2}{U_{R_z^\pi}} \otimes \underset{2\times 2}{U_I})}_{2^2 \times 2^2}$$

$$(U_{R_z^\pi} \otimes U_I) = \begin{bmatrix} e^{-i\frac{\pi}{2}} \cdot U_I & 0 \cdot U_I \\ 0 \cdot U_I & e^{i\frac{\pi}{2}} \cdot U_I \end{bmatrix}$$

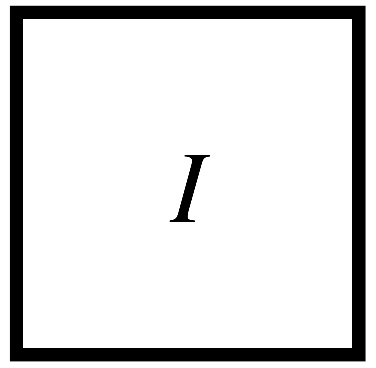
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

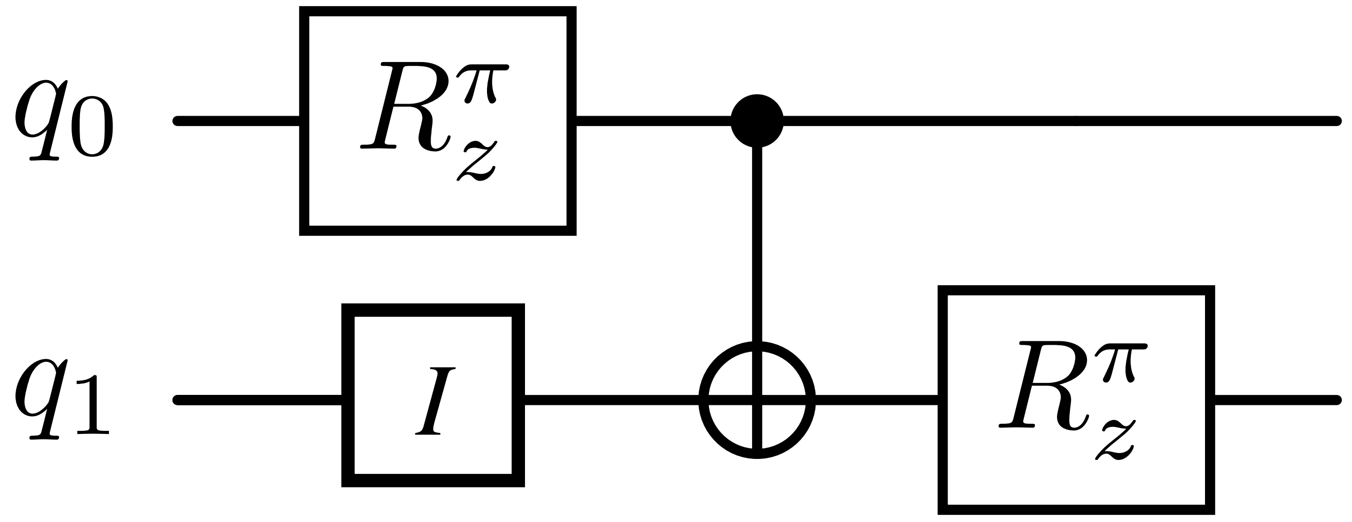


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

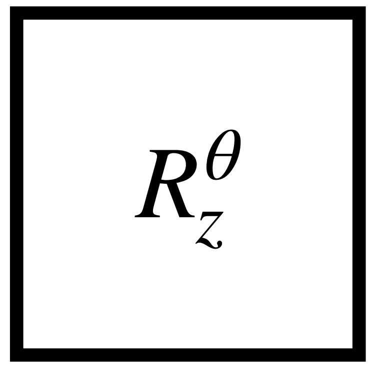
$C :=$



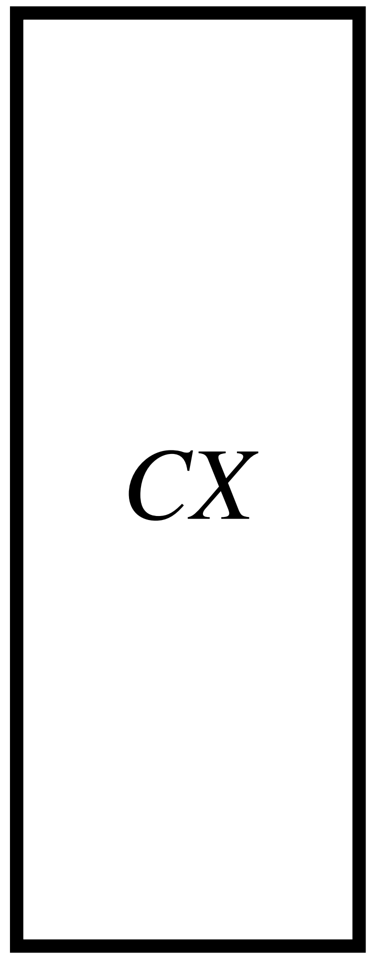
$$\underset{2^2 \times 2^2}{U_C} = \underbrace{(\underset{2\times 2}{U_{R_z^\pi}} \otimes \underset{2\times 2}{U_I})}_{2^2 \times 2^2}$$

$$(U_{R_z^\pi} \otimes U_I) = \begin{bmatrix} \overset{2\times 2}{e^{-i\frac{\pi}{2}} \cdot U_I} & \overset{2\times 2}{0 \cdot U_I} \\ \underset{2\times 2}{0 \cdot U_I} & \underset{2\times 2}{e^{i\frac{\pi}{2}} \cdot U_I} \end{bmatrix}$$

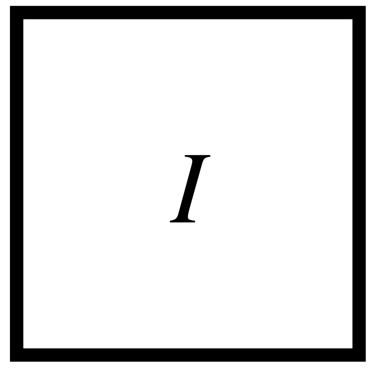
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

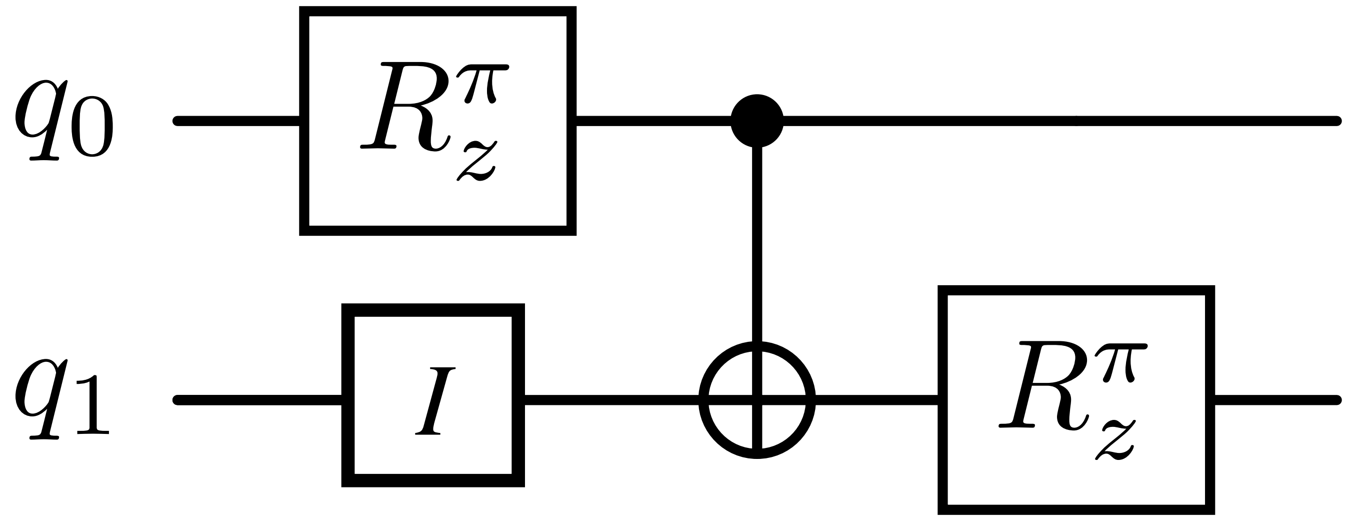


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

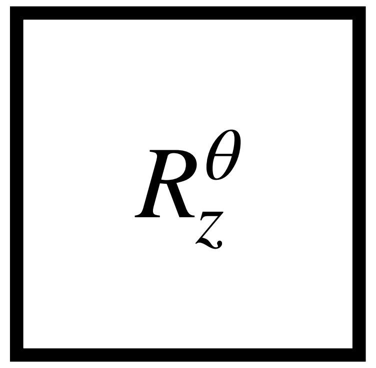
$C :=$



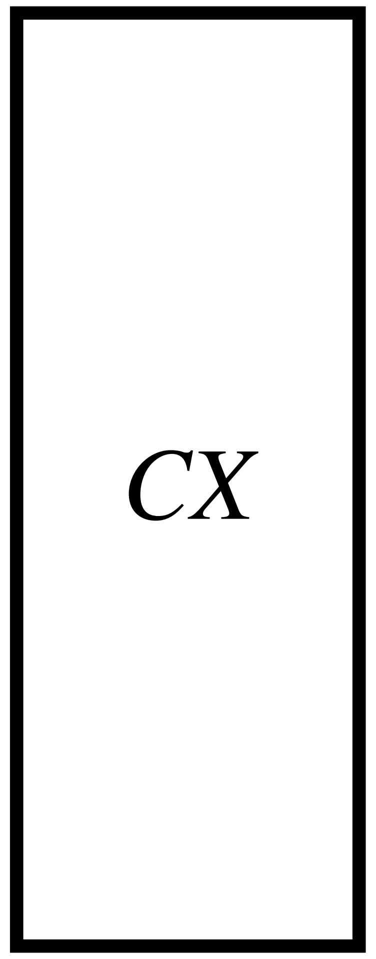
$$\underset{2^2 \times 2^2}{U_C} = U_{CX} \cdot \underbrace{(\underset{2\times 2}{U_{R_z^\pi}} \otimes \underset{2\times 2}{U_I})}_{2^2 \times 2^2}$$

$$(U_{R_z^\pi} \otimes U_I) = \begin{bmatrix} \overset{2\times 2}{e^{-i\frac{\pi}{2}} \cdot U_I} & \overset{2\times 2}{0 \cdot U_I} \\ \underset{2\times 2}{0 \cdot U_I} & \underset{2\times 2}{e^{i\frac{\pi}{2}} \cdot U_I} \end{bmatrix}$$

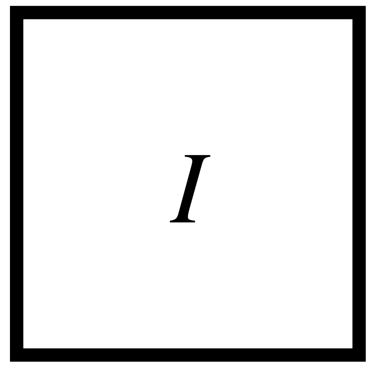
Circuit to Unitary



$$U_{R_z^\theta} := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

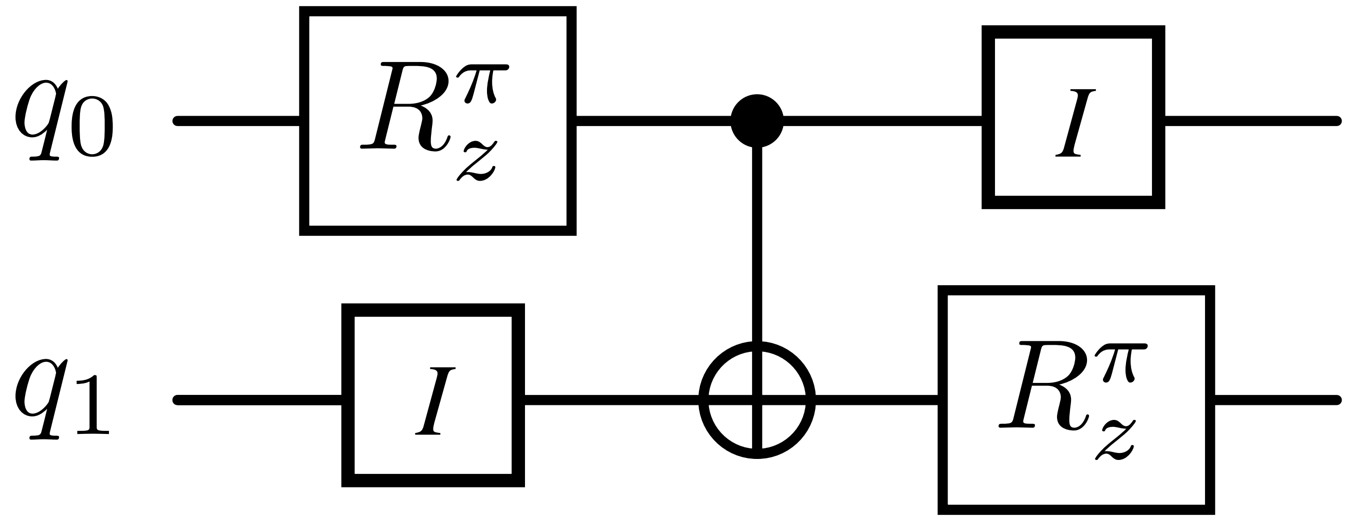


$$U_{CX} := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



$$U_I := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

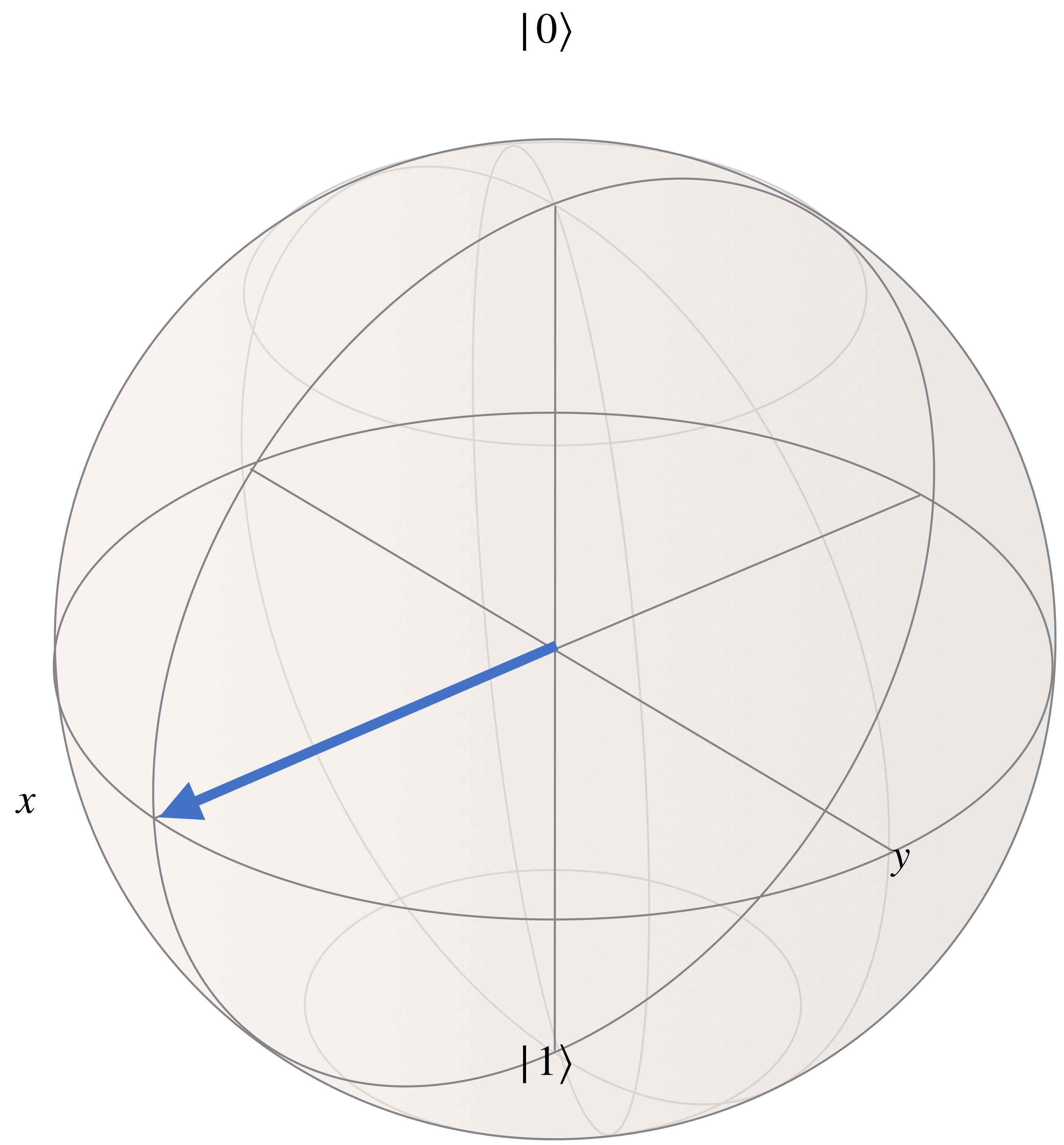
$C :=$



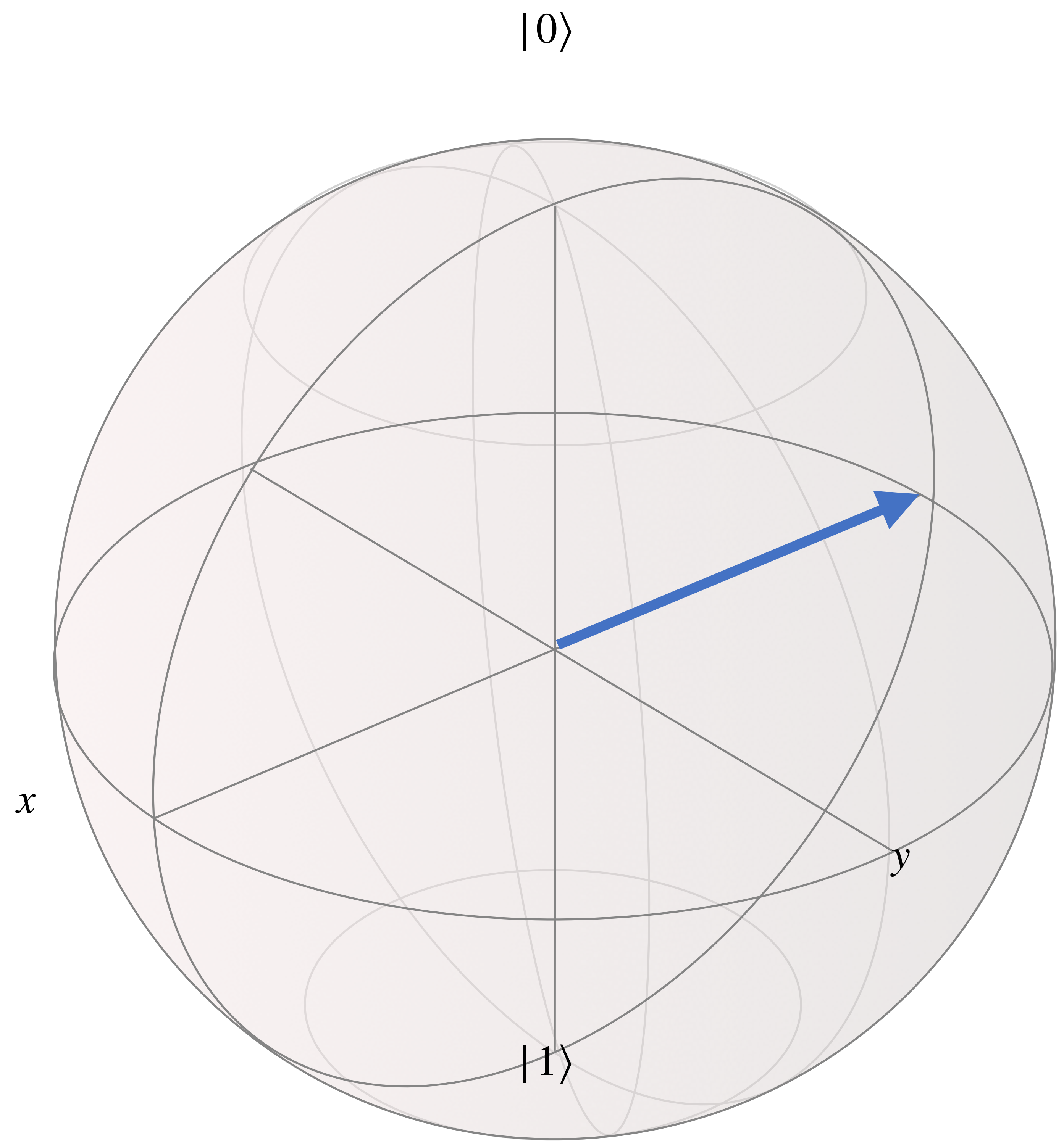
$$\underset{2^2 \times 2^2}{U_C} = (U_I \otimes U_{R_z^\pi}) \cdot U_{CX} \cdot \underbrace{(\underset{2 \times 2}{U_{R_z^\pi}} \otimes \underset{2 \times 2}{U_I})}_{2^2 \times 2^2}$$

$$(U_{R_z^\pi} \otimes U_I) = \begin{bmatrix} \overset{2 \times 2}{e^{-i\frac{\pi}{2}} \cdot U_I} & \overset{2 \times 2}{0 \cdot U_I} \\ \underset{2 \times 2}{0 \cdot U_I} & \underset{2 \times 2}{e^{i\frac{\pi}{2}} \cdot U_I} \end{bmatrix}$$

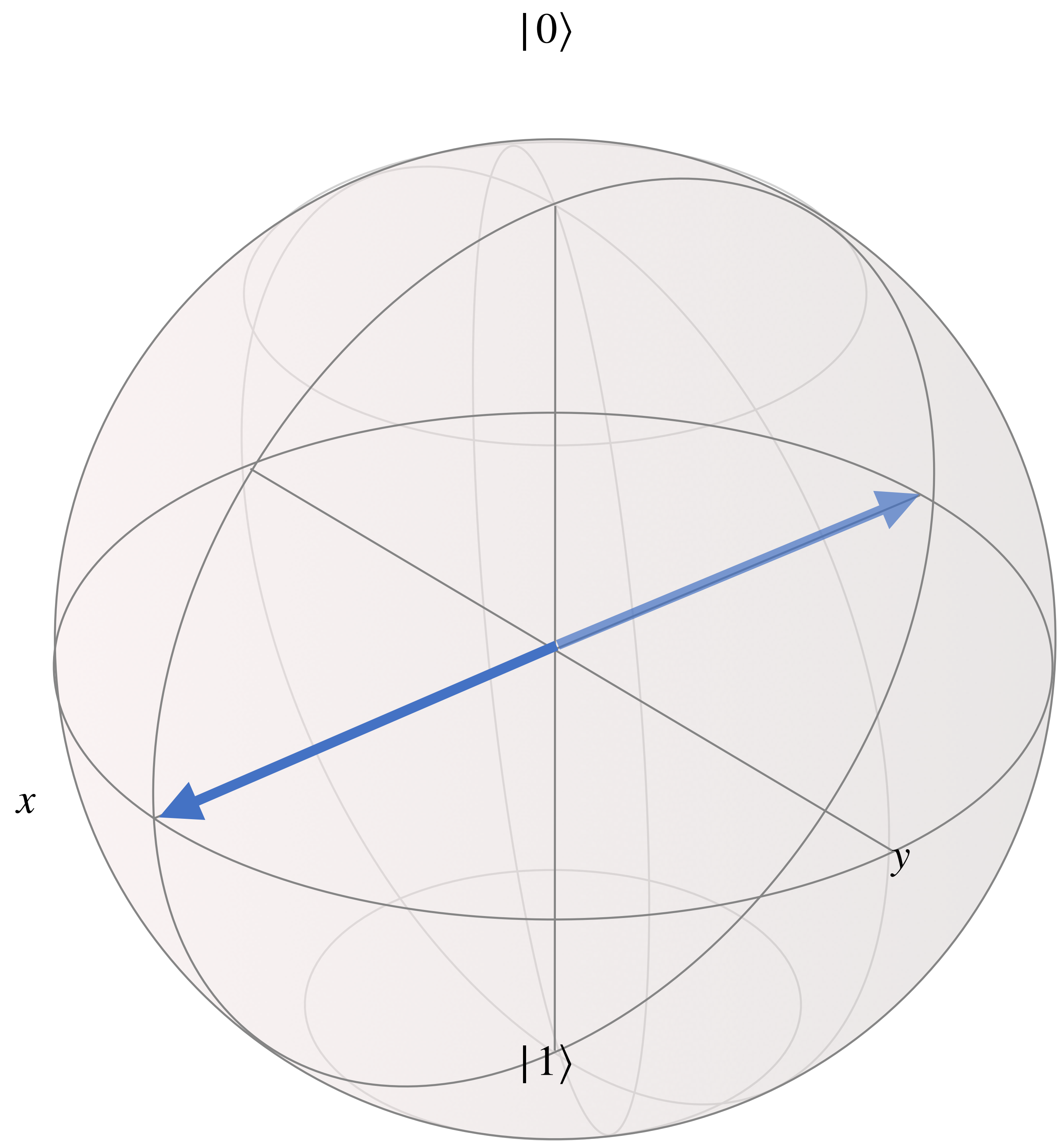
Approximate Circuit Equivalence



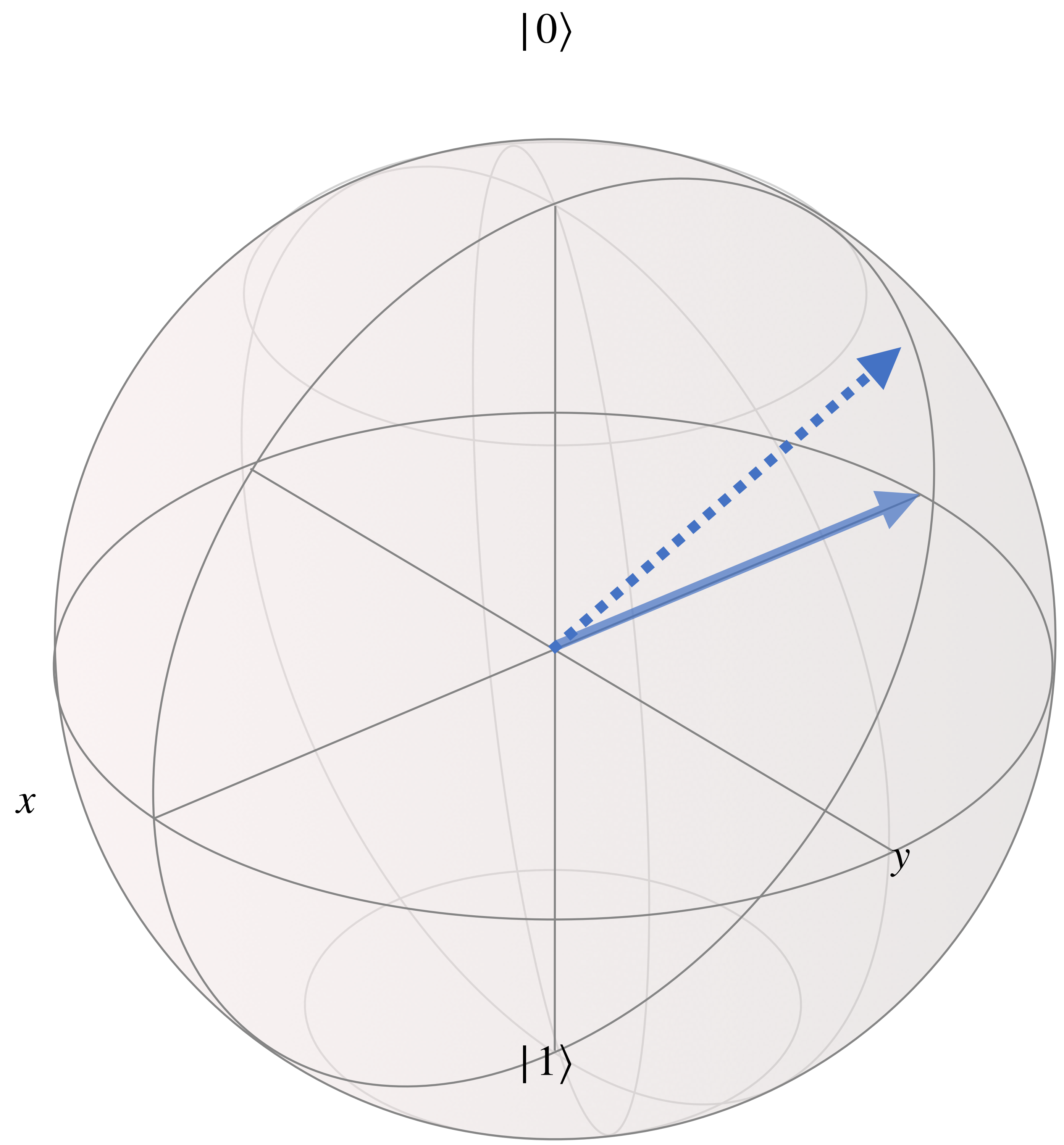
Approximate Circuit Equivalence



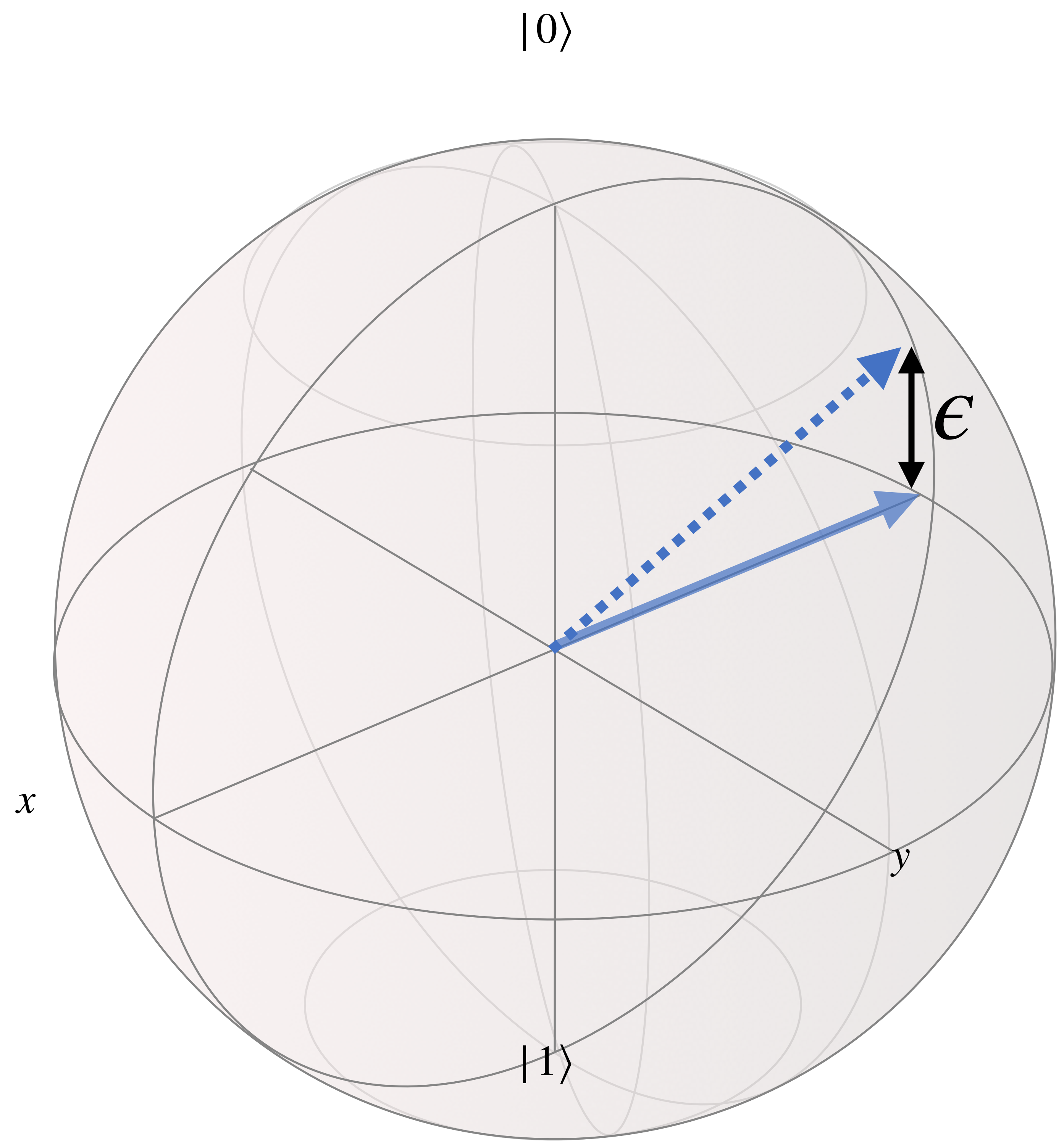
Approximate Circuit Equivalence



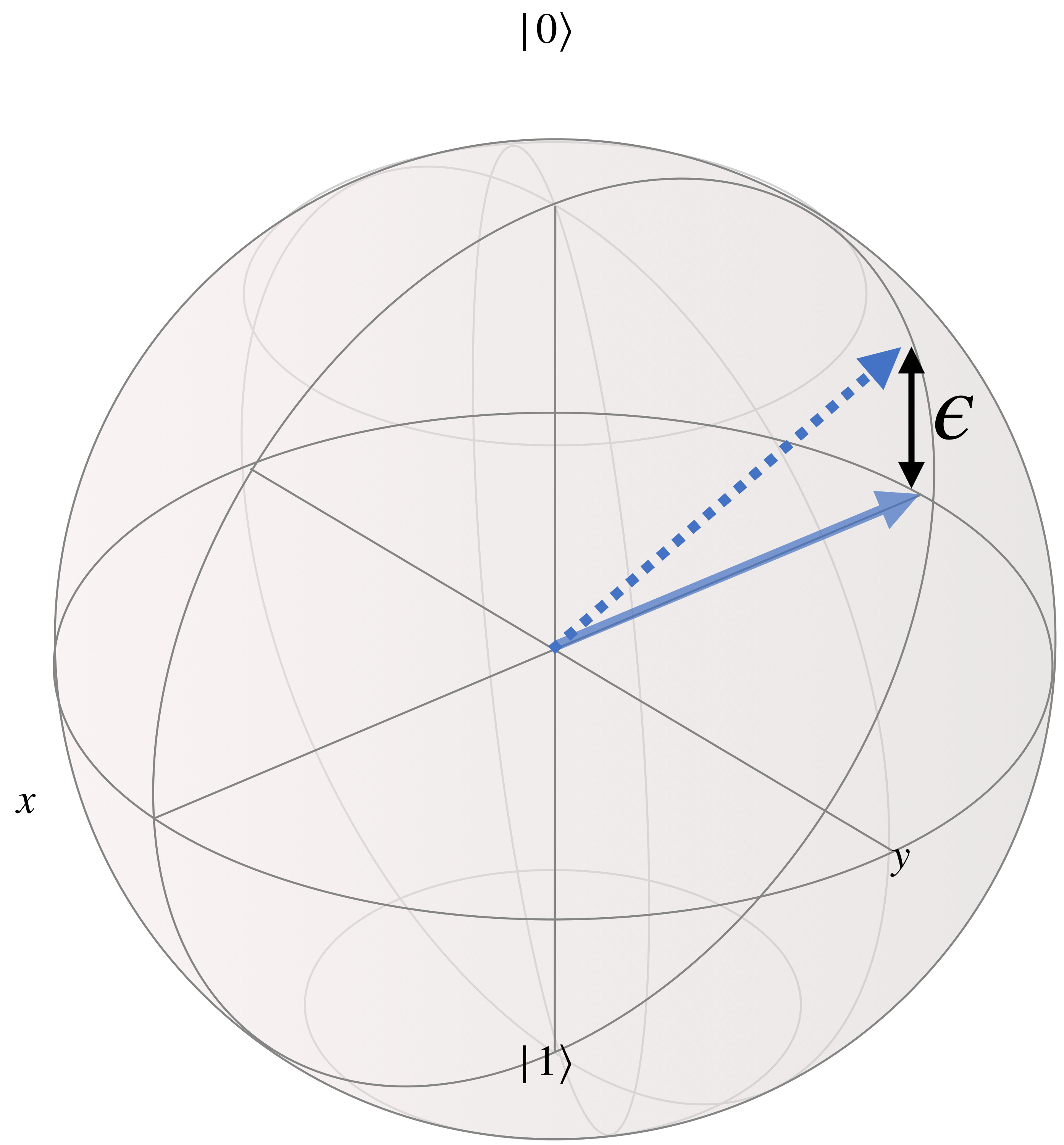
Approximate Circuit Equivalence



Approximate Circuit Equivalence



Approximate Circuit Equivalence



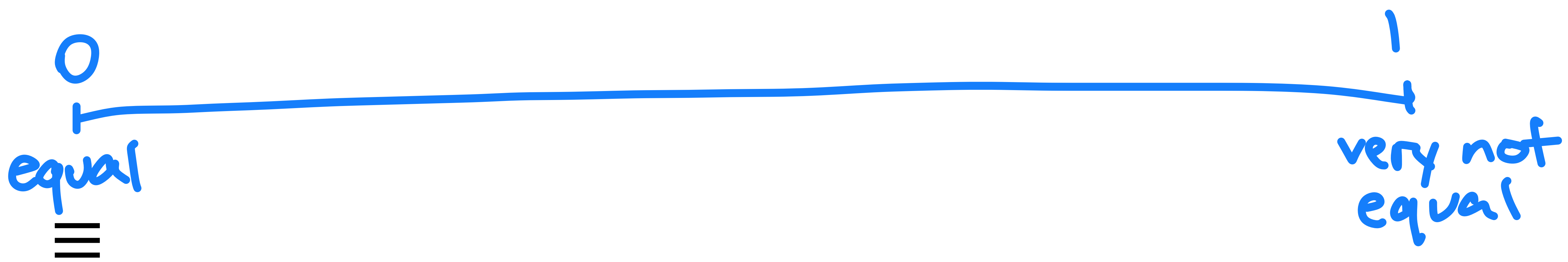
$$\Delta(U_C, U_{C'}) \leq \epsilon \iff C \equiv_\epsilon C'$$

Hilbert-Schmidt Distance

"easy" to compute

$$\Delta_{HS}(U, U') := \sqrt{1 - \frac{|Tr(U^\dagger U')|^2}{2^{2n}}}$$

qubits



Upper bound on total variation distance

Why Approximate Circuits?

FTQC **requires** approximations for arbitrary angle rotations

<u>Ideal</u>

Circuit C + No noise $= U_C$

<u>Reality</u>

Circuit C + Noise $= U_{C_{noisy}}$

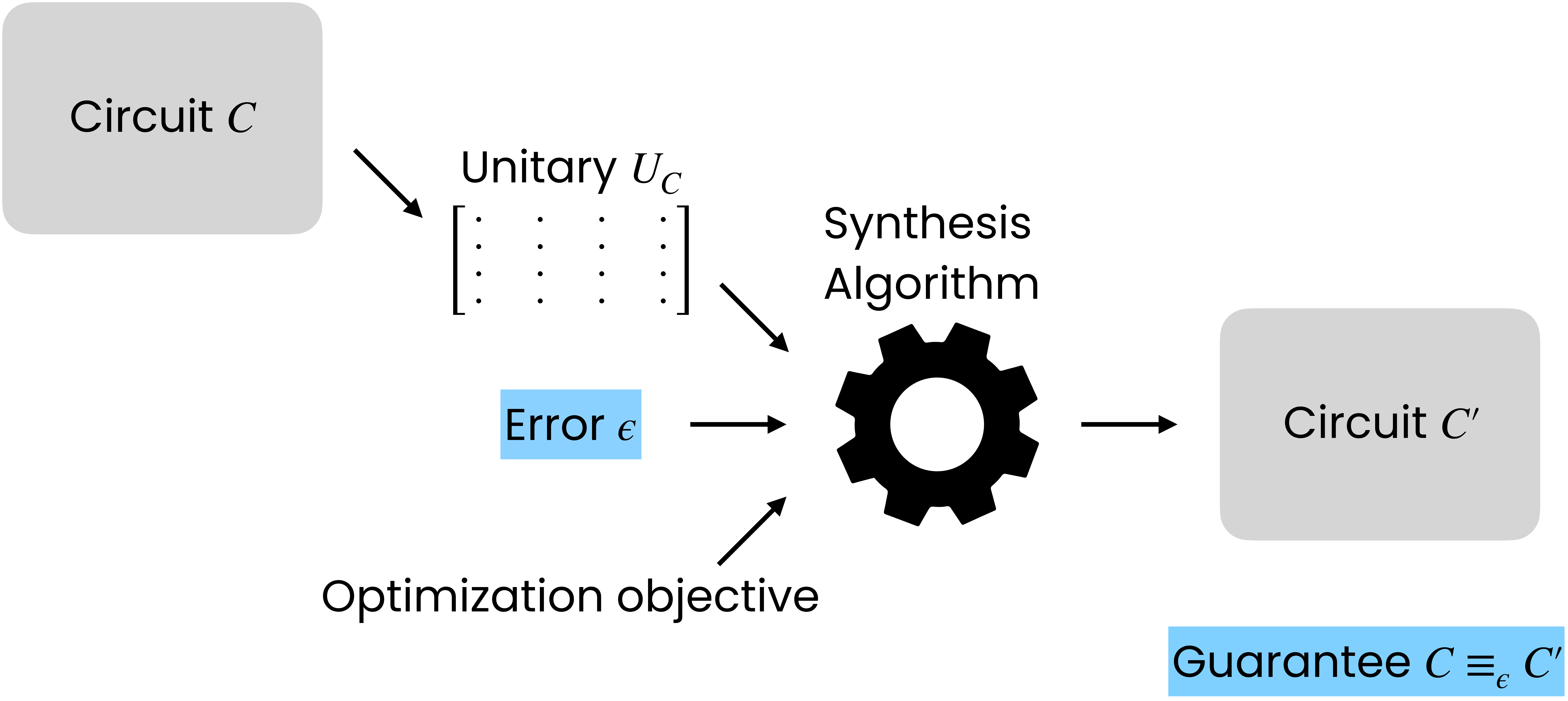
Circuit $C' \equiv_\epsilon C$

k fewer gates than C

The hope: $U_{C'_{noisy}}$ closer to U_C than $U_{C_{noisy}}$

Noise from k gates $> \epsilon$

Unitary Synthesis



Rich Research Area

Synthesis of Quantum-Logic Circuits

Vivek V. Shende, Stephen S. Bullock, and Igor L. Markov, *Senior Member, IEEE*

Towards Optimal Topology Aware Quantum Circuit Synthesis

Marc G. Davis, Ethan Smith, Ana Tudor, Koushik Sen, Irfan Siddiqi — Costin Iancu, *Lawrence Berkeley National Laboratory*

{marc.davis,ethanh

bqskit Public

Berkeley Quantum Synthesis Toolkit

OpenQASM ☆ 132 38

LEAP: Scaling Numerical Optimization Based Synthesis Using an Incremental Approach

ETHAN SMITH and MARC GRAU DAVIS, University of California, Berkeley

JEFFREY LARSON, Argonne National Laboratory

RIJSEN, and COSTIN IANCU,

Exact synthesis of multiqubit Clifford+T circuits

Brett Giles, Department of Computer Science, University of Calgary

Peter Selinger, Departm

Modular Component-Based Quantum Circuit Synthesis

CHAN GU KANG, Korea University, Republic of Korea

blic of Korea

Synthetiq: Fast and Versatile Quantum Circuit Synthesis

ANOUK PARADIS*, ETH Zurich, Switzerland

JASPER DEKONINCK*, ETH Zurich, Switzerland

BENJAMIN BICHSEL, ETH Zurich, Switzerland

MARTIN VECHEV, ETH Zurich, Switzerland

synthetiq Public

OpenQASM ☆ 3 MIT

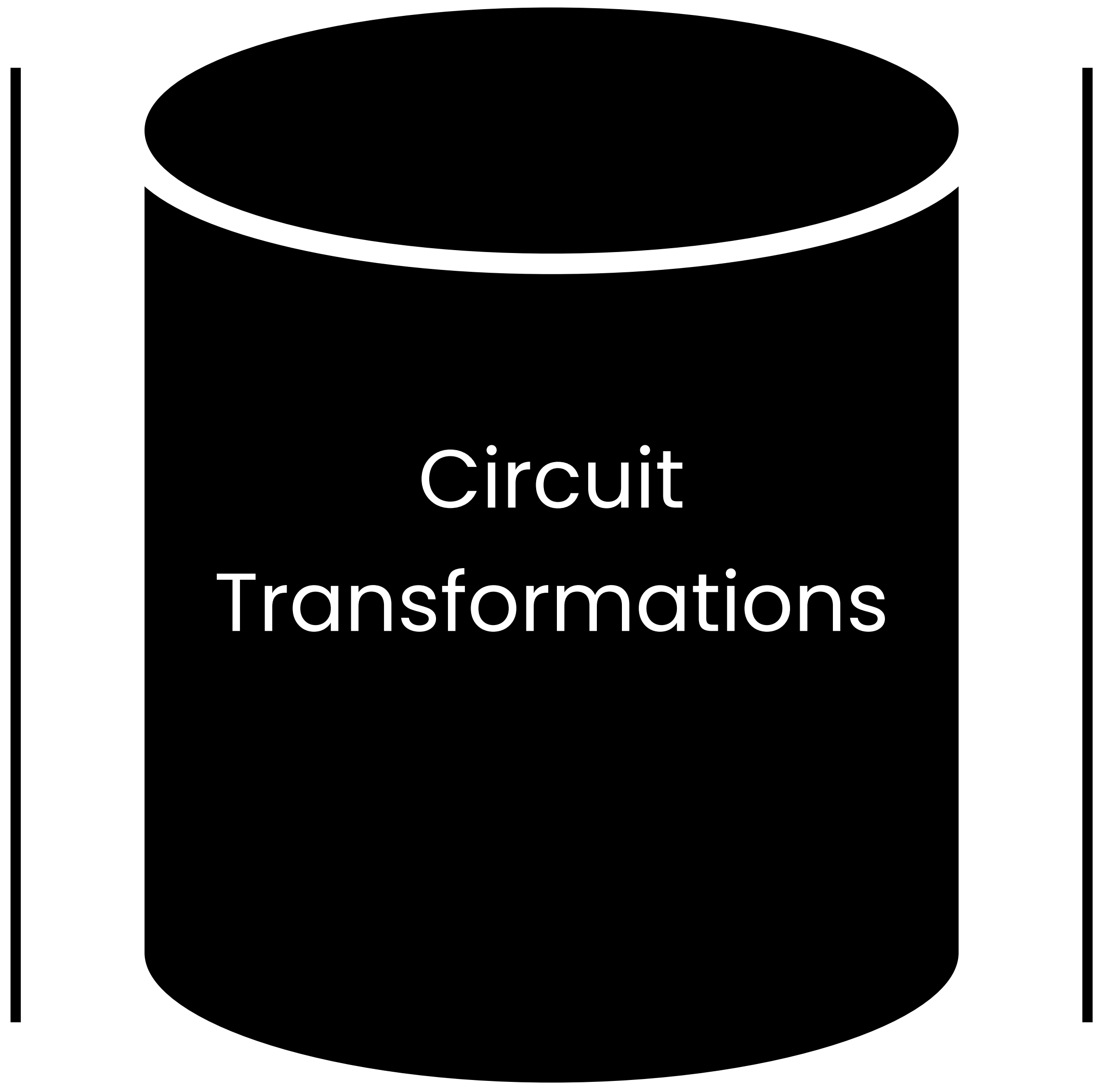
Techniques:

- analytical
- numerical optimization
- explicit search

Scheduling Transformations

Scheduling is hard

In what order to apply???



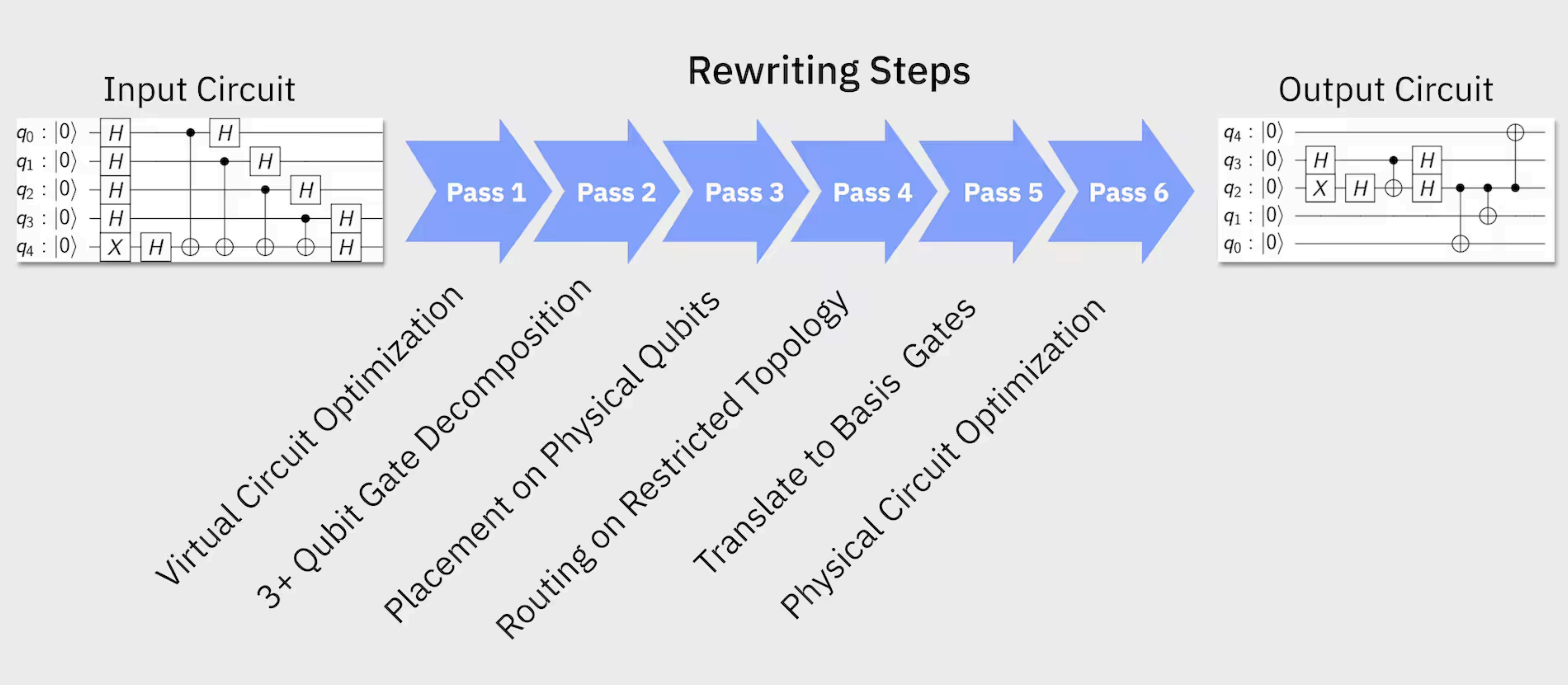
= big

"phase-ordering problem"

Qiskit

https://docs.quantum.ibm.com/api/qiskit/transpiler#optimization-stage

https://github.com/Qiskit/qiskit/blob/stable/1.4/qiskit/transpiler/preset_passmanagers/level3.py



Qiskit

https://docs.quantum.ibm.com/api/qiskit/transpiler#optimization-stage

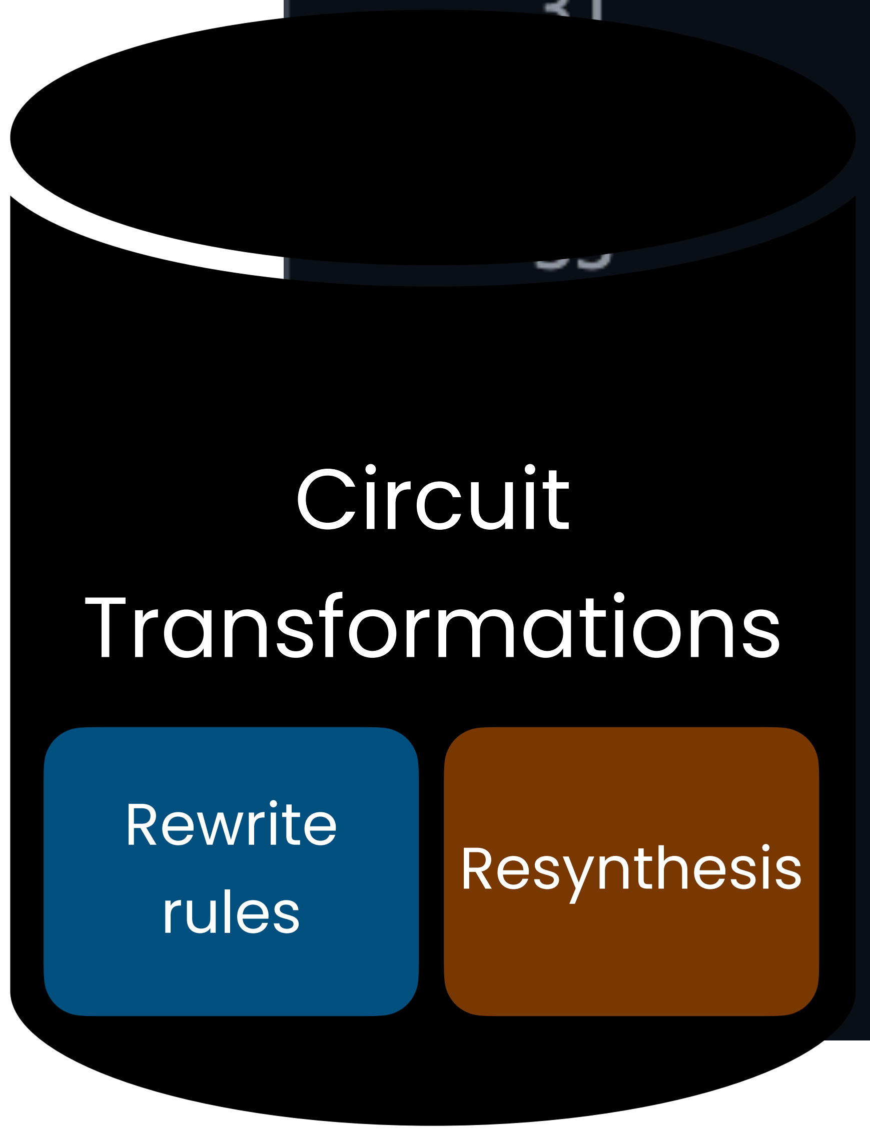
https://github.com/Qiskit/qiskit/blob/stable/1.4/qiskit/transpiler/preset_passmanagers/level3.py

qiskit / qiskit / transpiler / preset_passmanagers / **level3.py** ↑ Top

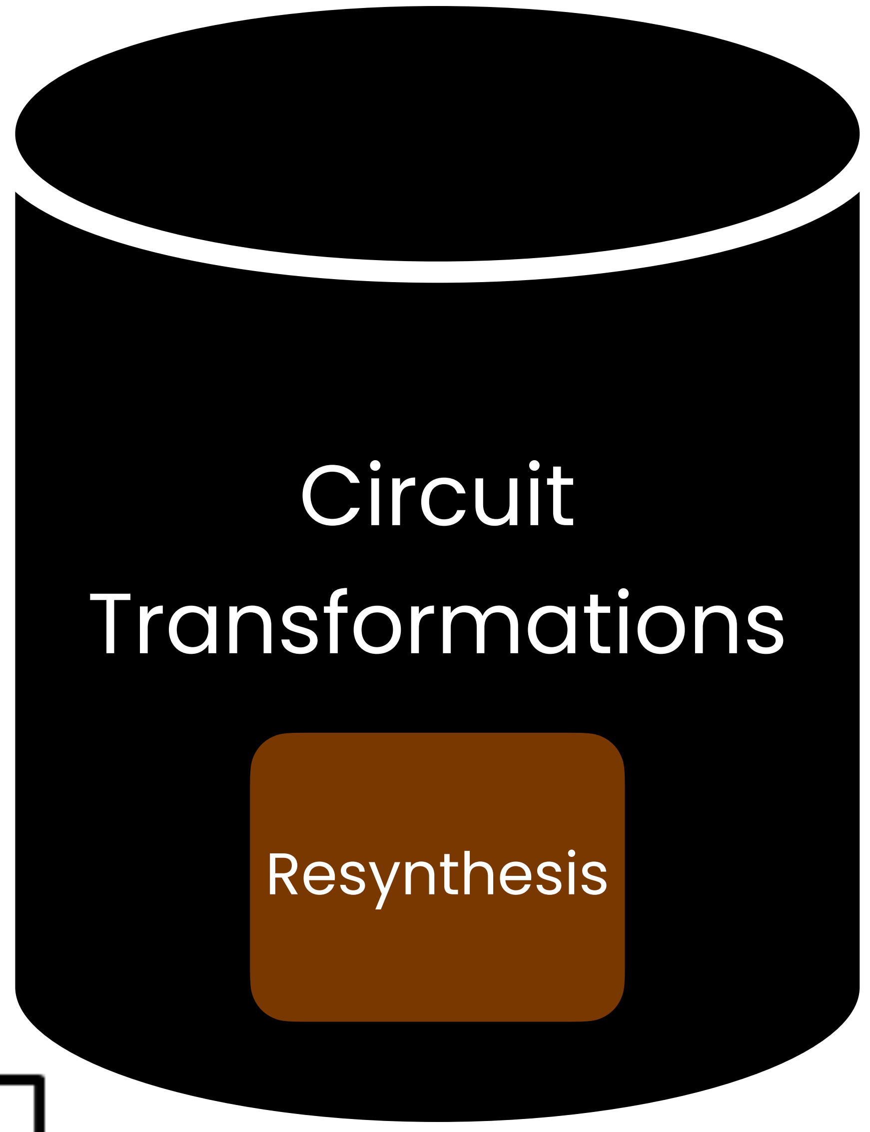
Code Blame 119 lines (104 loc) · 4.8 KB · Raw

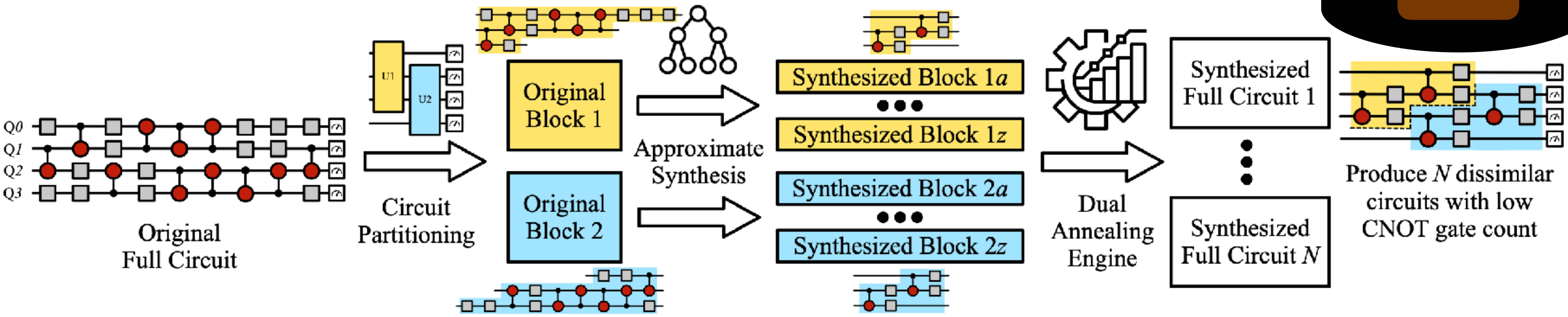
```
def level_3_pass_manager(pass_manager_config: PassManagerConfig) -> StagedPassManager:
    """Level 3 pass manager: heavy optimization by noise adaptive qubit mapping and
    gate cancellation using commutativity rules and unitary synthesis.

    This pass manager applies the user-given initial layout. If none is given, a search
    for a perfect layout (i.e. one that satisfies all 2-qubit interactions) is conducted.
    If no such layout is found, and device calibration information is available, the
    circuit is mapped to the qubits with best readouts and to CX gates with highest fidelit

    The pass manager then tra                           he coupling constraints.
    It is then unrolled to th                           rections are fixed.
    Finally, optimizations in the form of commutative gate cancellation, resynthesis
    of two-qubit unitary blocks, and redundant reset removal are performed.
```

coarse fixed passes



BQSKit

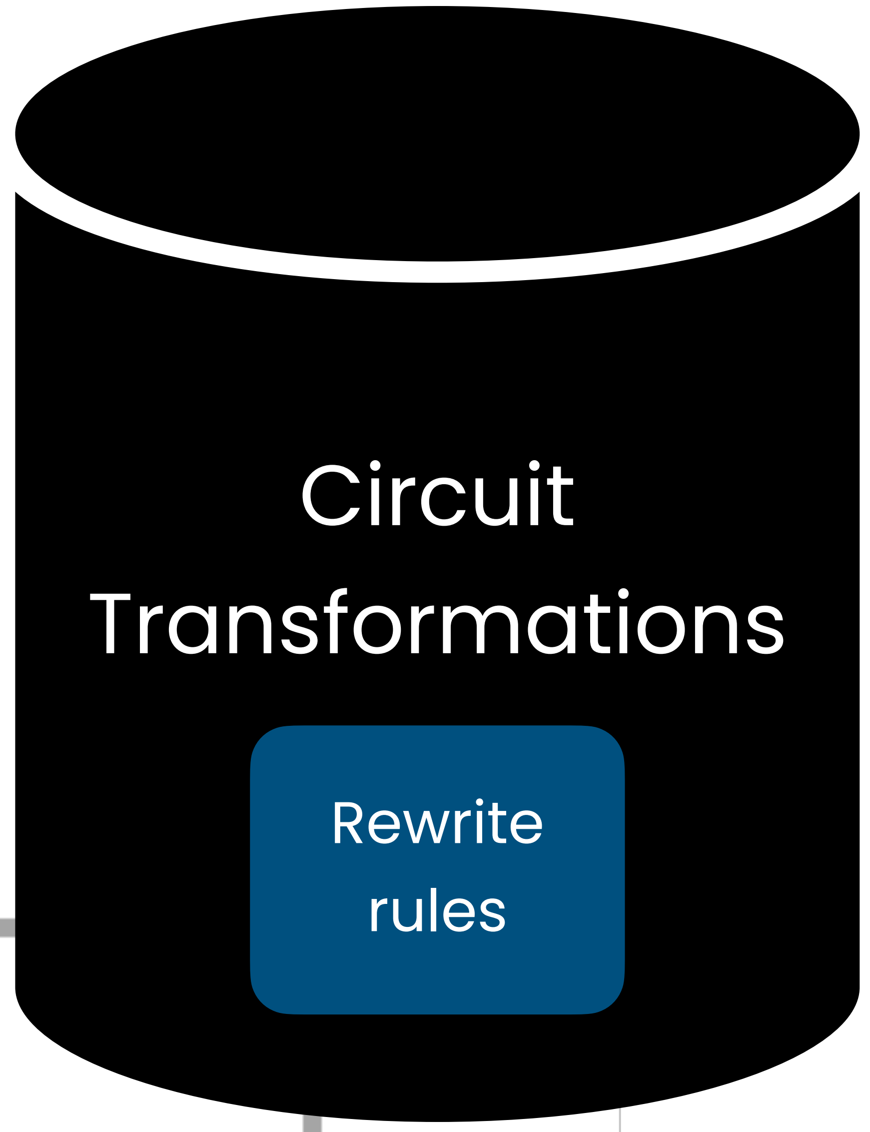


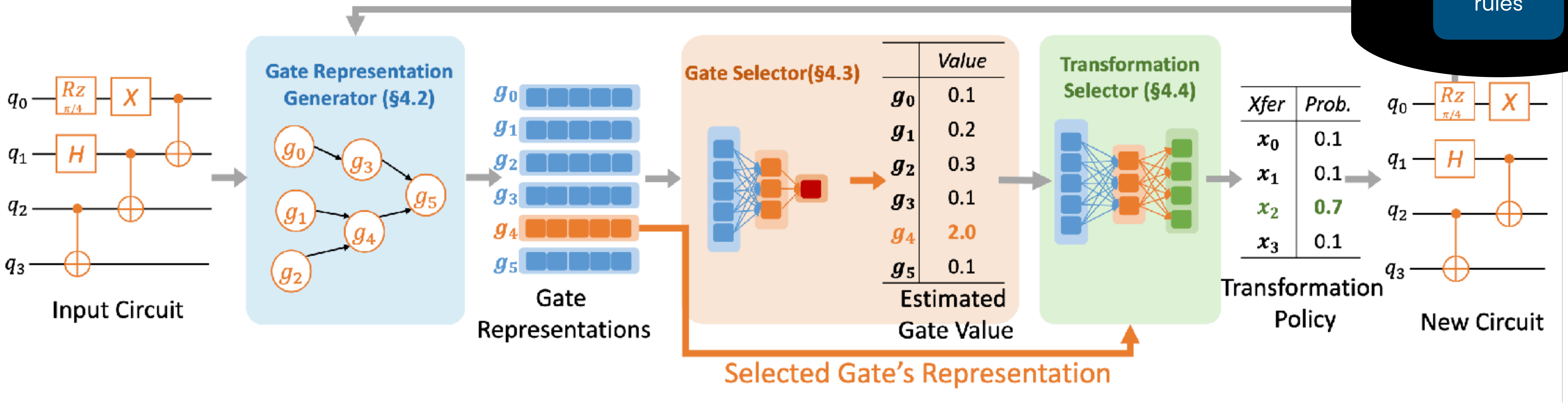


QUEST: systematically approximating quantum circuits for higher output fidelity [Patel et al. 2022]

https://github.com/BQSKit/bqskit

Quarl

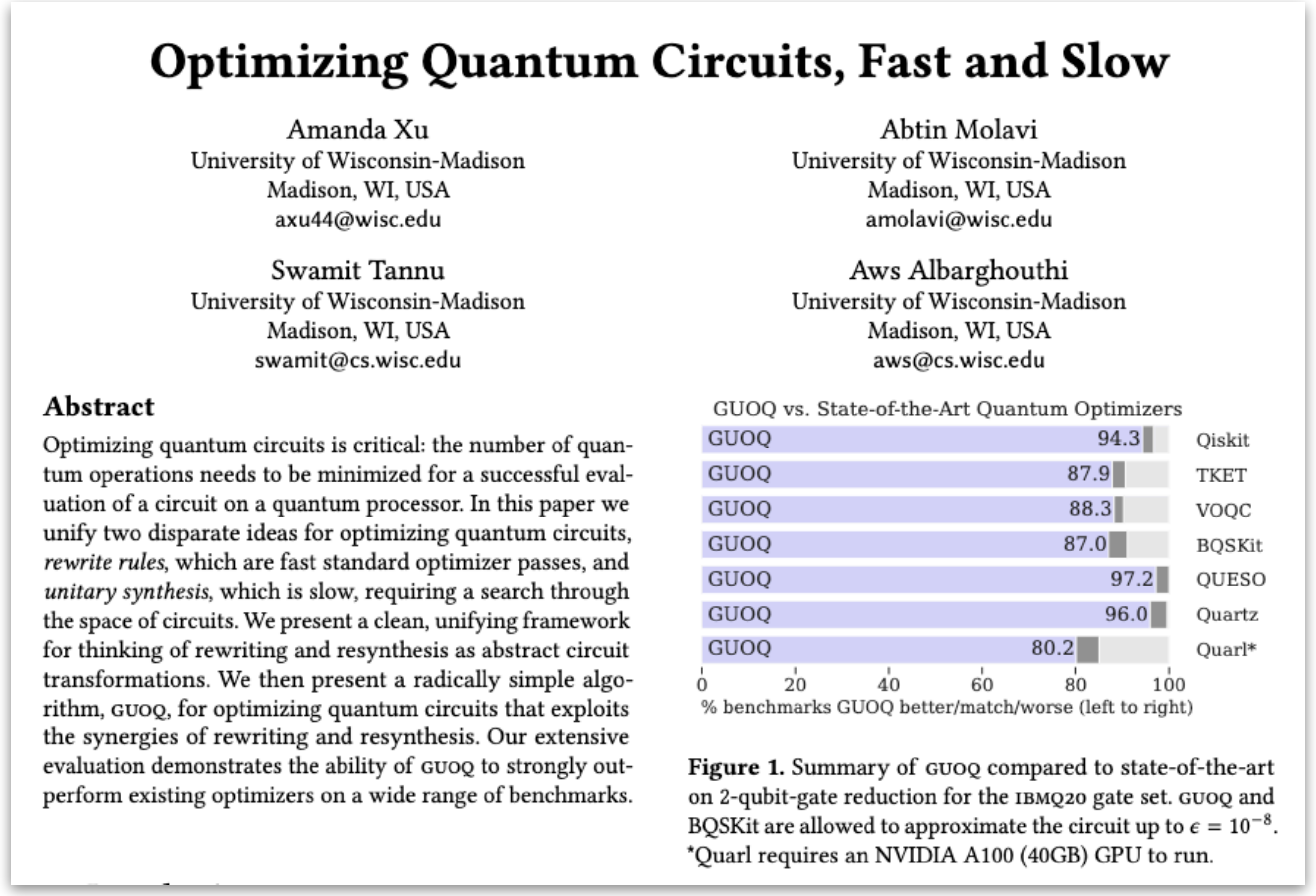




* requires an NVIDIA A100 GPU ($$$)

Quarl: A Learning-Based Quantum Circuit Optimizer [Li et al. 2024]

GUOQ



Optimizing Quantum Circuits, Fast and Slow

Amanda Xu
University of Wisconsin-Madison
Madison, WI, USA
axu44@wisc.edu

Abtin Molavi
University of Wisconsin-Madison
Madison, WI, USA
amolavi@wisc.edu

Swamit Tannu
University of Wisconsin-Madison
Madison, WI, USA
swamit@cs.wisc.edu

Aws Albarghouthi
University of Wisconsin-Madison
Madison, WI, USA
aws@cs.wisc.edu

Abstract

Optimizing quantum circuits is critical: the number of quantum operations needs to be minimized for a successful evaluation of a circuit on a quantum processor. In this paper we unify two disparate ideas for optimizing quantum circuits, *rewrite rules*, which are fast standard optimizer passes, and *unitary synthesis*, which is slow, requiring a search through the space of circuits. We present a clean, unifying framework for thinking of rewriting and resynthesis as abstract circuit transformations. We then present a radically simple algorithm, GUOQ, for optimizing quantum circuits that exploits the synergies of rewriting and resynthesis. Our extensive evaluation demonstrates the ability of GUOQ to strongly outperform existing optimizers on a wide range of benchmarks.

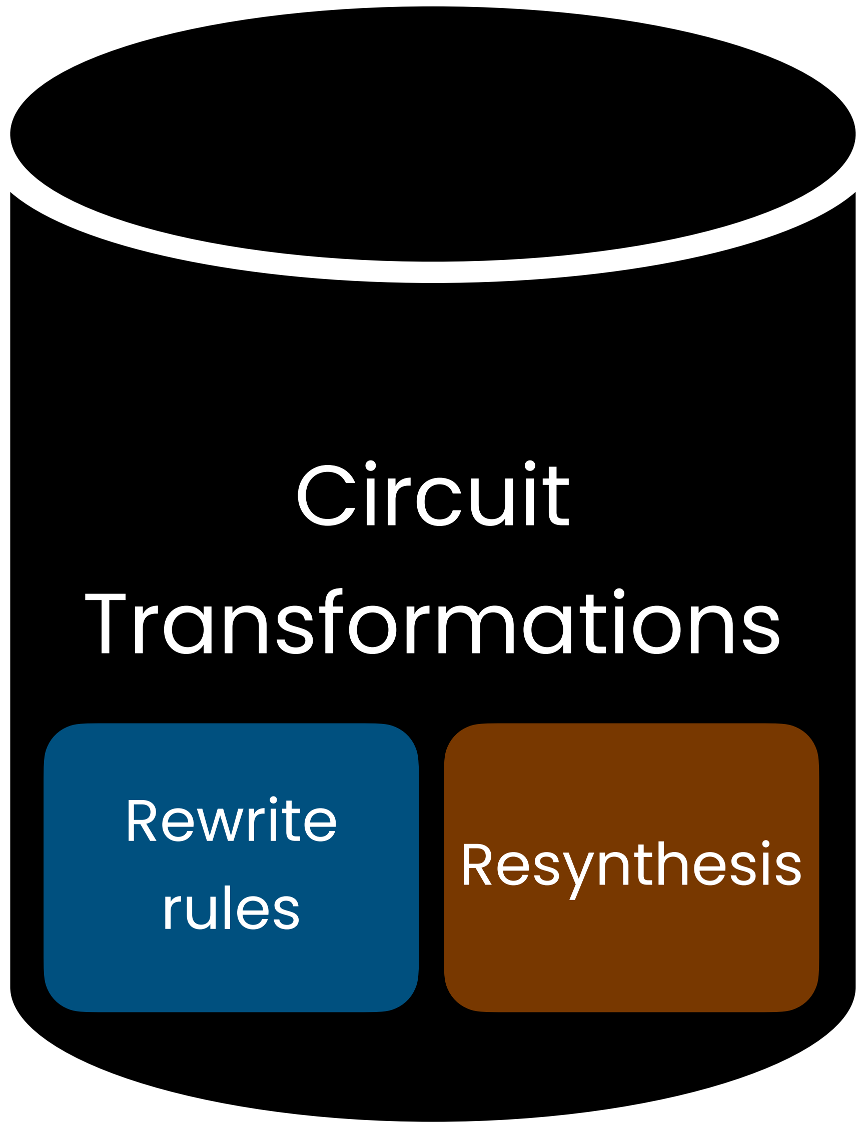
GUOQ vs. State-of-the-Art Quantum Optimizers

GUOQ	94.3	Qiskit
GUOQ	87.9	TKET
GUOQ	88.3	VOQC
GUOQ	87.0	BQSKit
GUOQ	97.2	QUESO
GUOQ	96.0	Quartz
GUOQ	80.2	Quarl*

0 20 40 60 80 100
% benchmarks GUOQ better/match/worse (left to right)

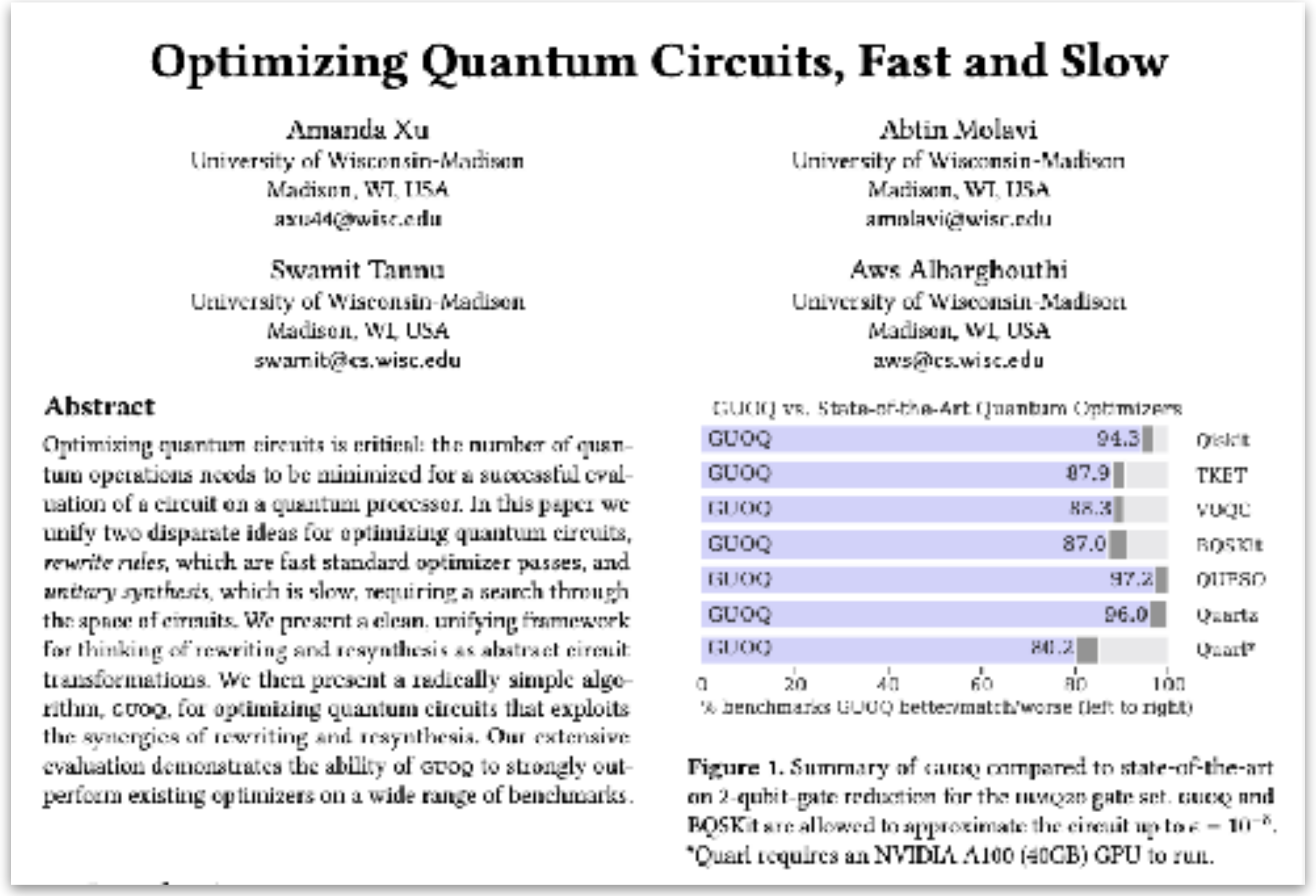
Figure 1. Summary of GUOQ compared to state-of-the-art on 2-qubit-gate reduction for the IBMQ20 gate set. GUOQ and BQSKit are allowed to approximate the circuit up to $\epsilon = 10^{-8}$. *Quarl requires an NVIDIA A100 (40GB) GPU to run.

ASPLOS talk on April 1st!



1. Randomly pick a transformation.
2. Randomly pick a subcircuit to transform.
3. Accept the result if better than current. Else, reject with high probability.

GUOQ


Optimizing Quantum Circuits, Fast and Slow

Amanda Xu
University of Wisconsin-Madison
Madison, WI, USA
axu44@wisc.edu

Abtin Molavi
University of Wisconsin-Madison
Madison, WI, USA
amolavi@wisc.edu

Swamit Tannu
University of Wisconsin-Madison
Madison, WI, USA
swamit@cs.wisc.edu

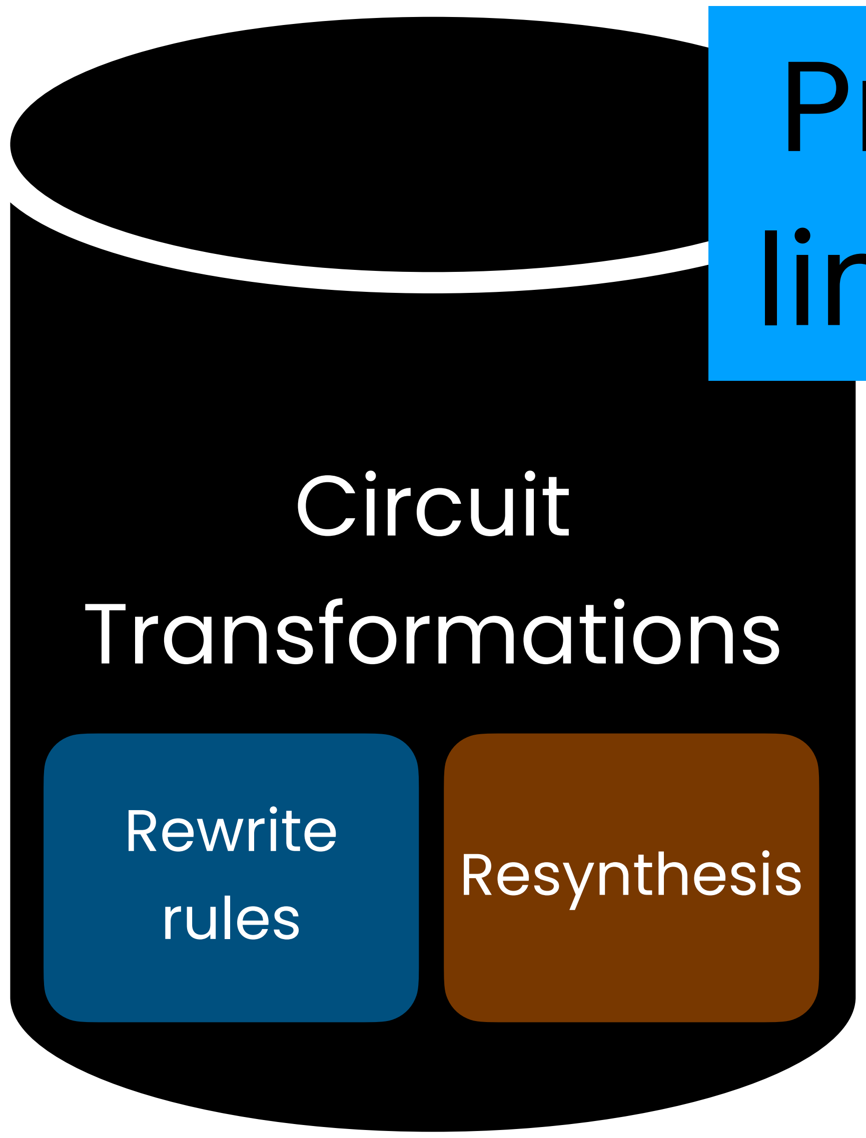
Aws Albarghouthi
University of Wisconsin-Madison
Madison, WI, USA
aws@cs.wisc.edu

Abstract

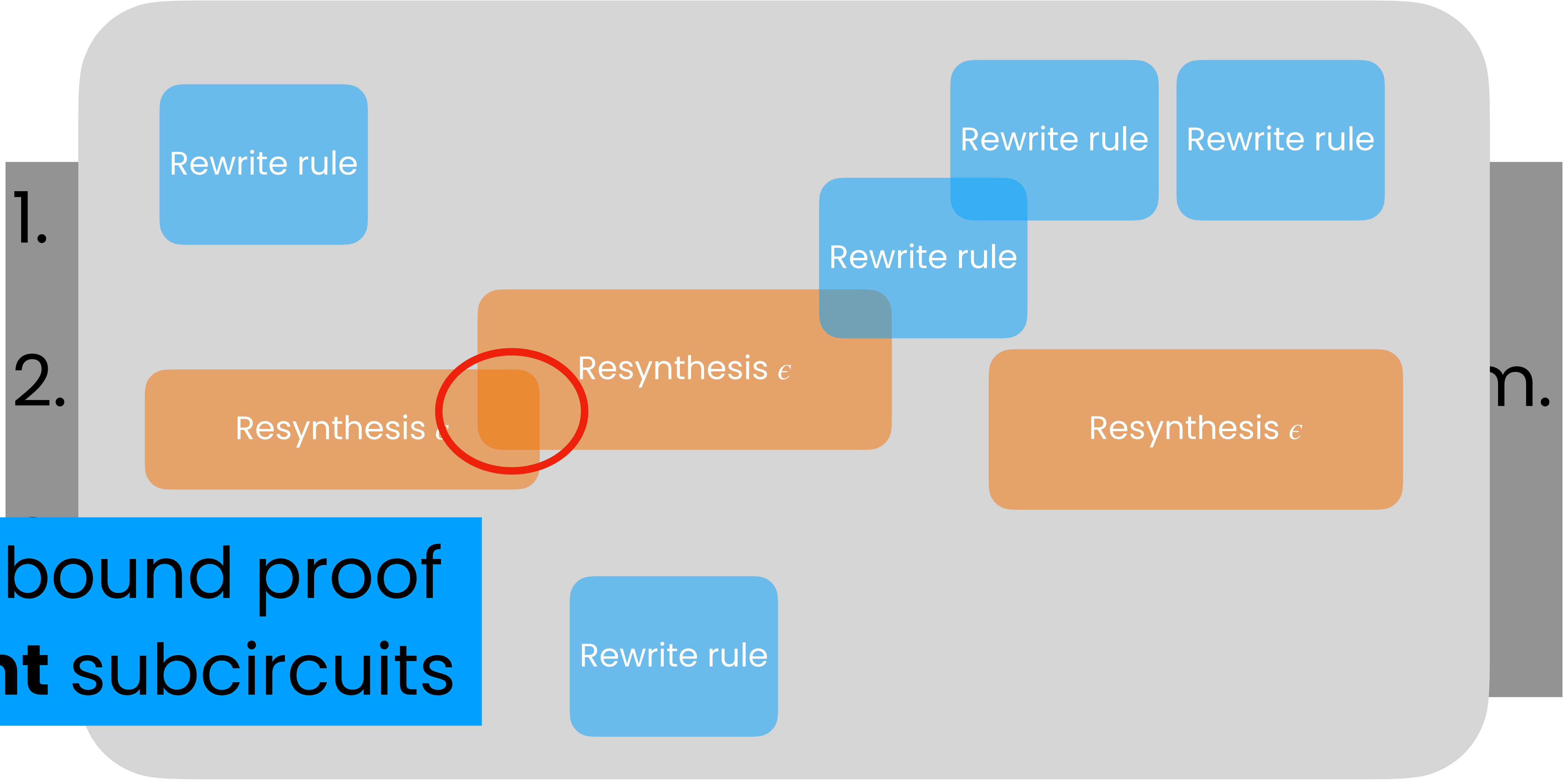
Optimizing quantum circuits is critical: the number of quantum operations needs to be minimized for a successful evaluation of a circuit on a quantum processor. In this paper we unify two disparate ideas for optimizing quantum circuits, *rewrite rules*, which are fast standard optimizer passes, and *unitary synthesis*, which is slow, requiring a search through the space of circuits. We present a clean, unifying framework for thinking of rewriting and resynthesis as abstract circuit transformations. We then present a radically simple algorithm, GUOQ, for optimizing quantum circuits that exploits the synergies of rewriting and resynthesis. Our extensive evaluation demonstrates the ability of GUOQ to strongly outperform existing optimizers on a wide range of benchmarks.

Figure 1. Summary of GUOQ compared to state-of-the-art on 2-qubit-gate reduction for the IBMQ20 gate set. GUOQ and BQSKit are allowed to approximate the circuit up to $\epsilon = 10^{-8}$. *Quarl requires an NVIDIA A100 (40GB) GPU to run.

ASPLOS talk on April 1st!



Prior work: error bound proof limited to **disjoint** subcircuits

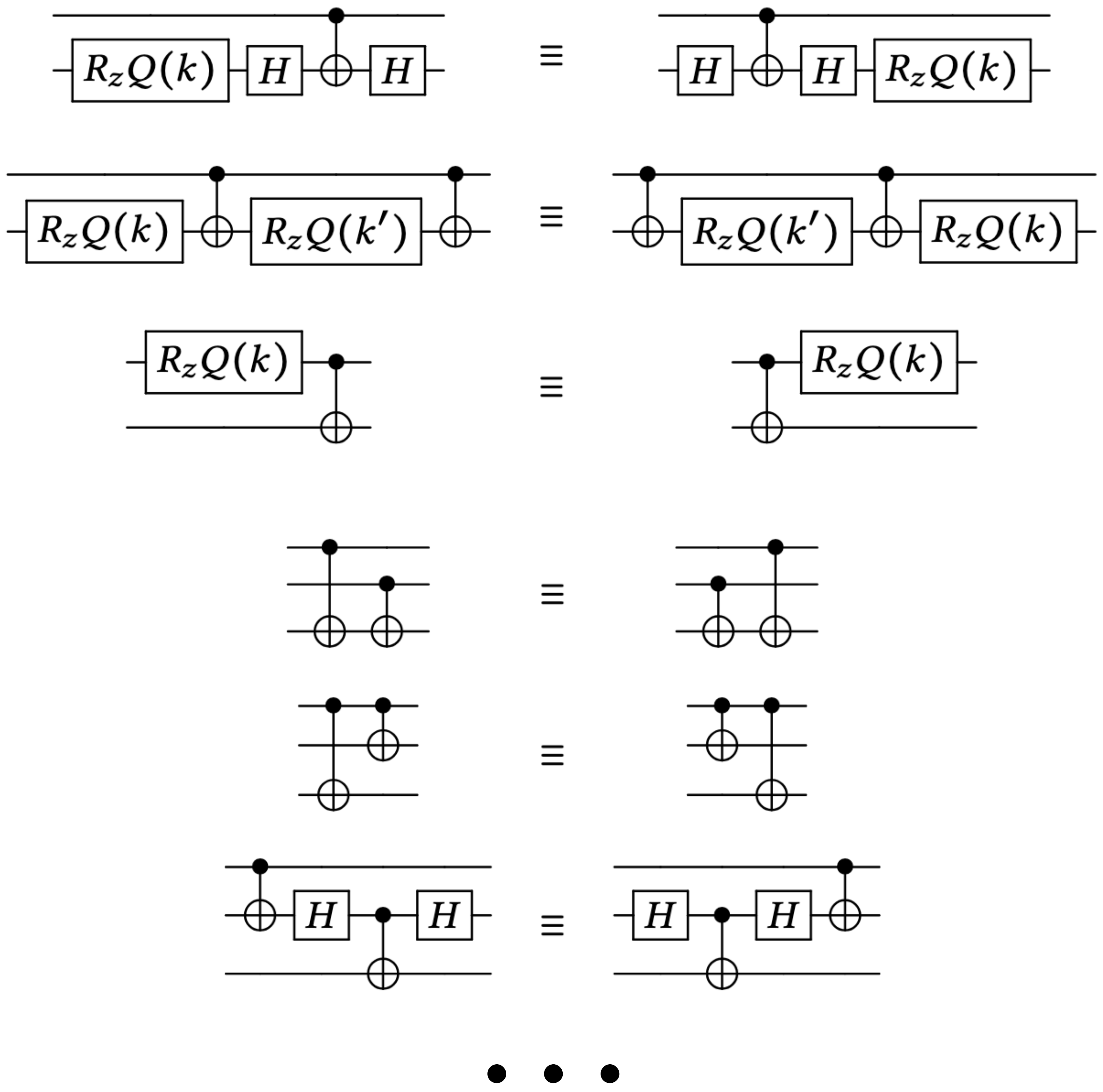


$$\epsilon_{total} \leq \epsilon + \epsilon + \epsilon$$

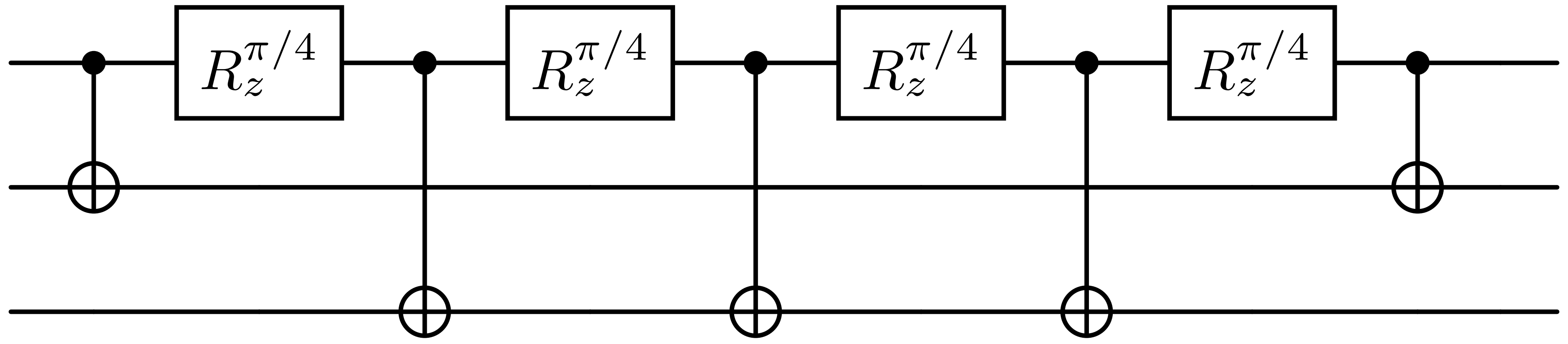
We extended to handle arbitrary subcircuits

GUOQ Motivation: Two Disparate Techniques

Rewrite Rules

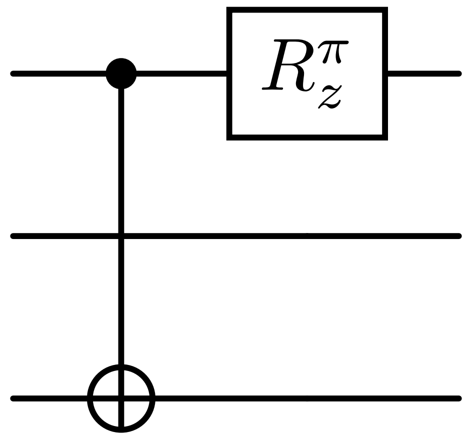


Circuit Resynthesis



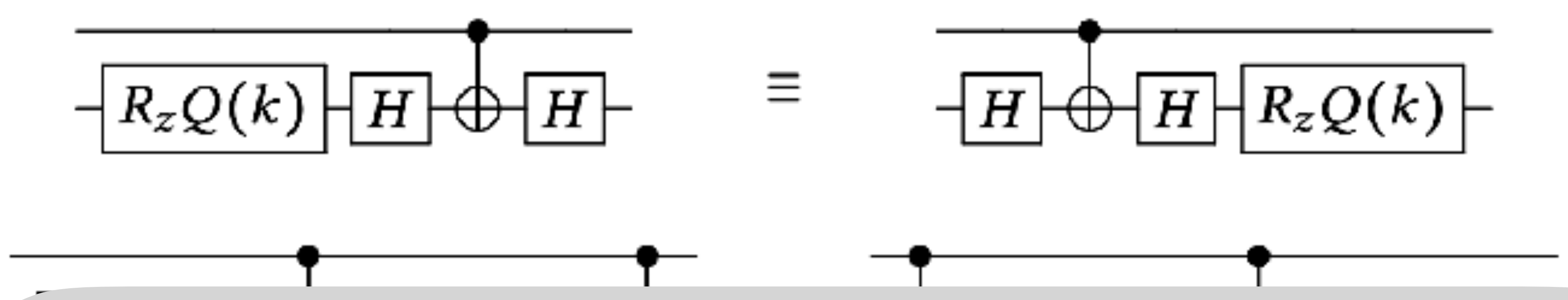
specification

synthesis



GUOQ Motivation: Two Disparate Techniques

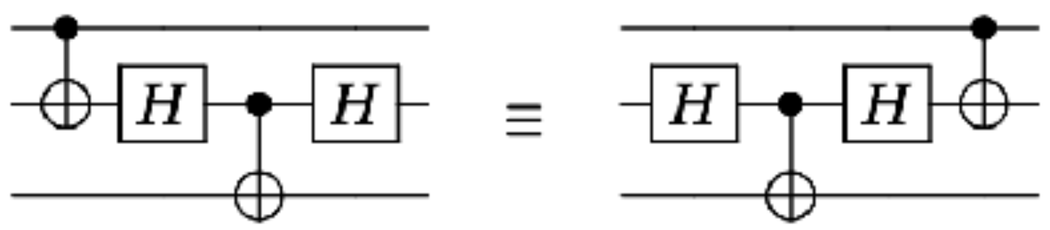
Rewrite Rules



Fast

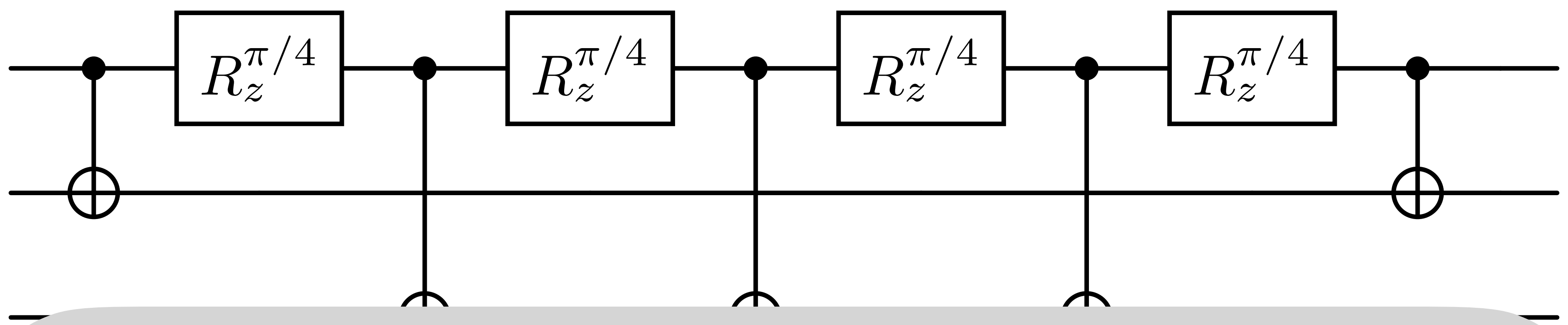
Limited by # gates

Exact



• • •

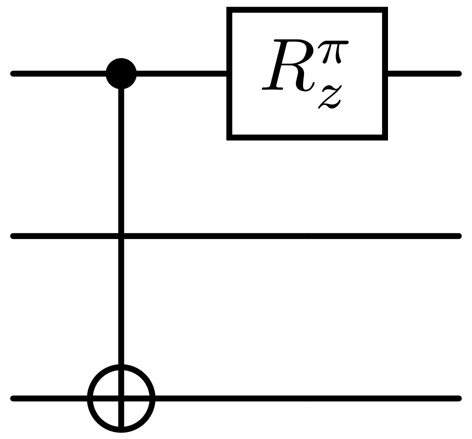
Circuit Resynthesis



Slow

Limited by # qubits

Approximate



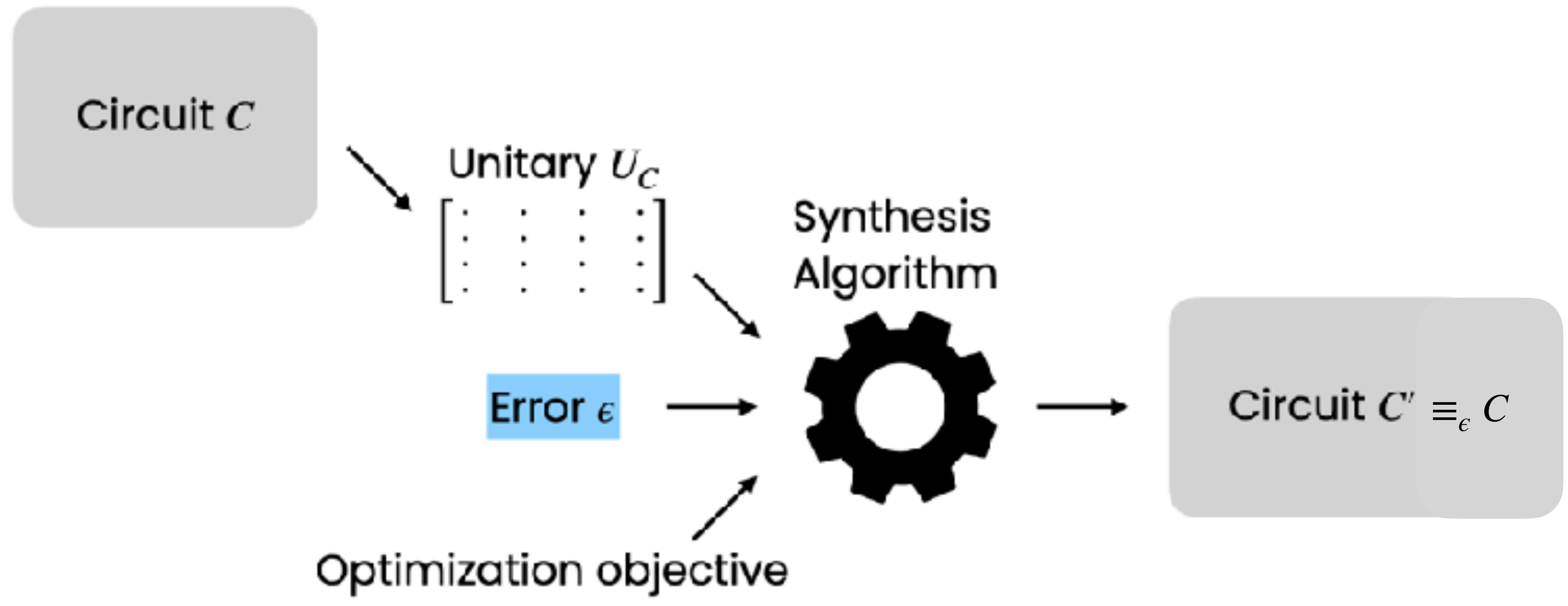
Unifying Abstract Circuit Transformations

Circuit C → Circuit $C' \equiv_\epsilon C$

transformation $\tau_\epsilon : \mathscr{C} \rightarrow \mathscr{C}$

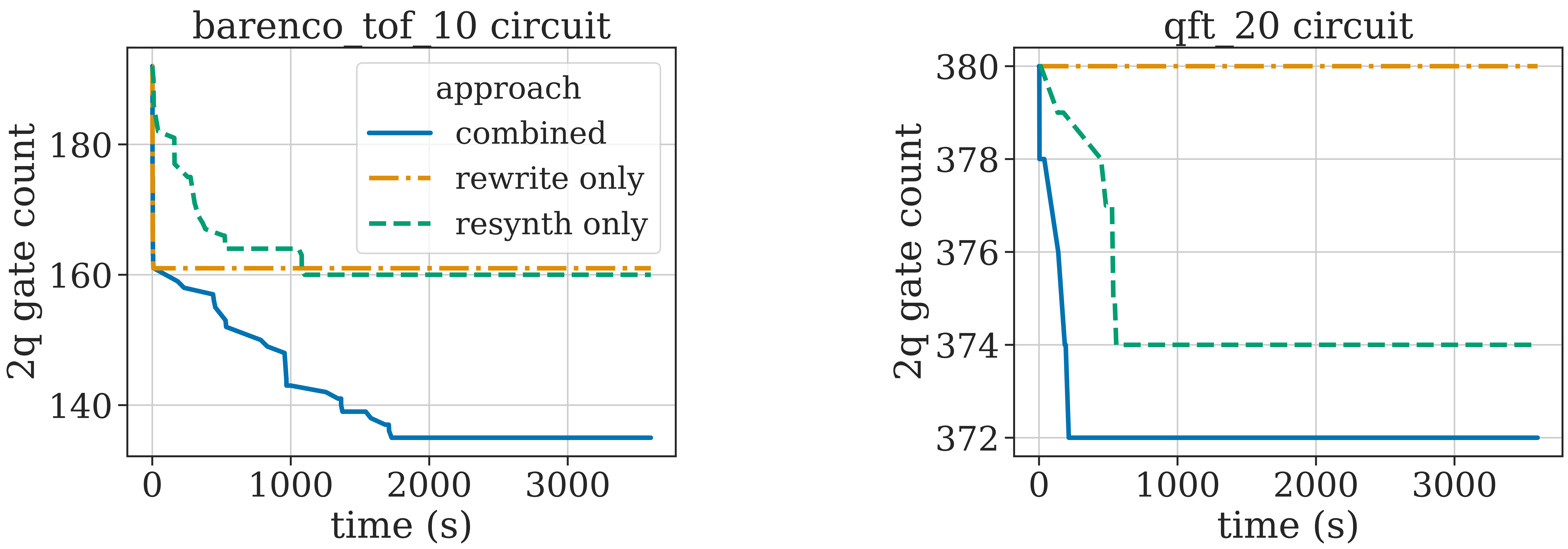
Circuit C → Circuit $C' \equiv_0 C$

rewrite rule



resynthesis

GUOQ Search Exploits Synergy



Also significantly outperforms state-of-the-art!

Come to the ASPLOS talk on April 1st!

Optimizing for FTQC

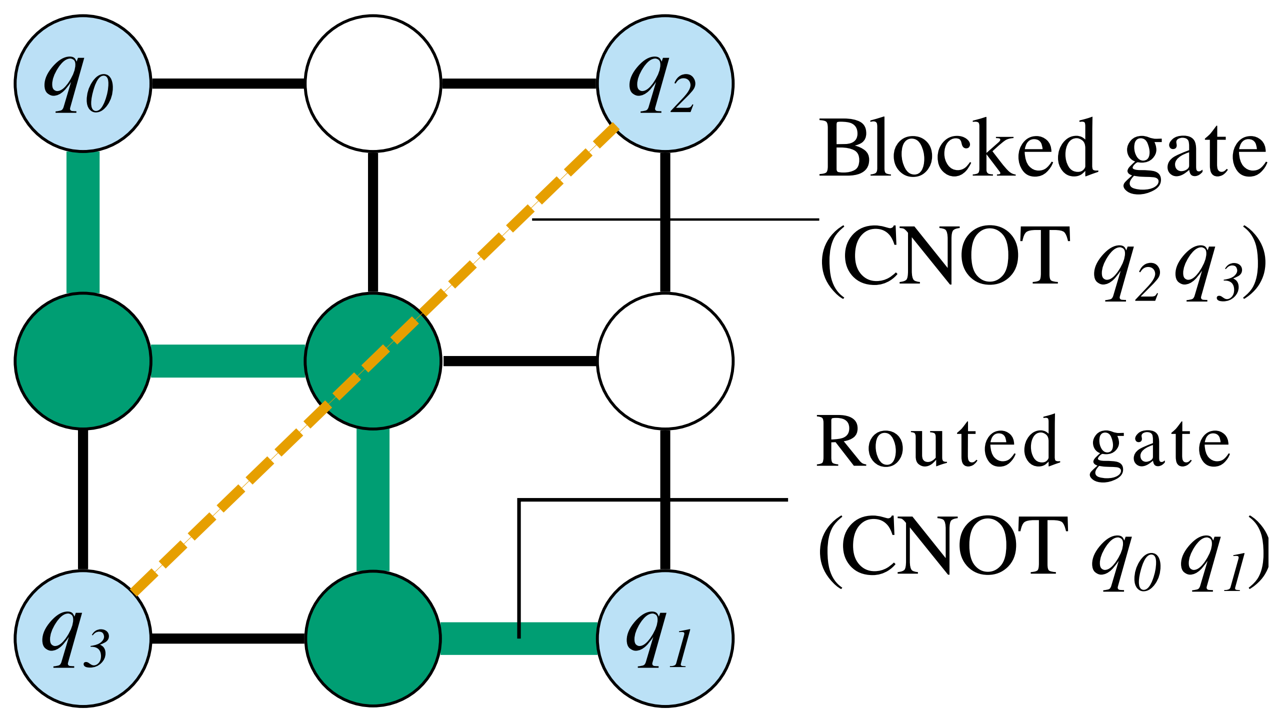
Optimizing for FTQC

discrete

Clifford + T gate set: $\{T, T^\dagger, S, S^\dagger, H, CX\}$

"magic state distillation"

Eastin-Knill theorem: no transversal universal set



BQSKit

Qiskit

GUOQ

Quarl

VOQC

Quartz

TKET

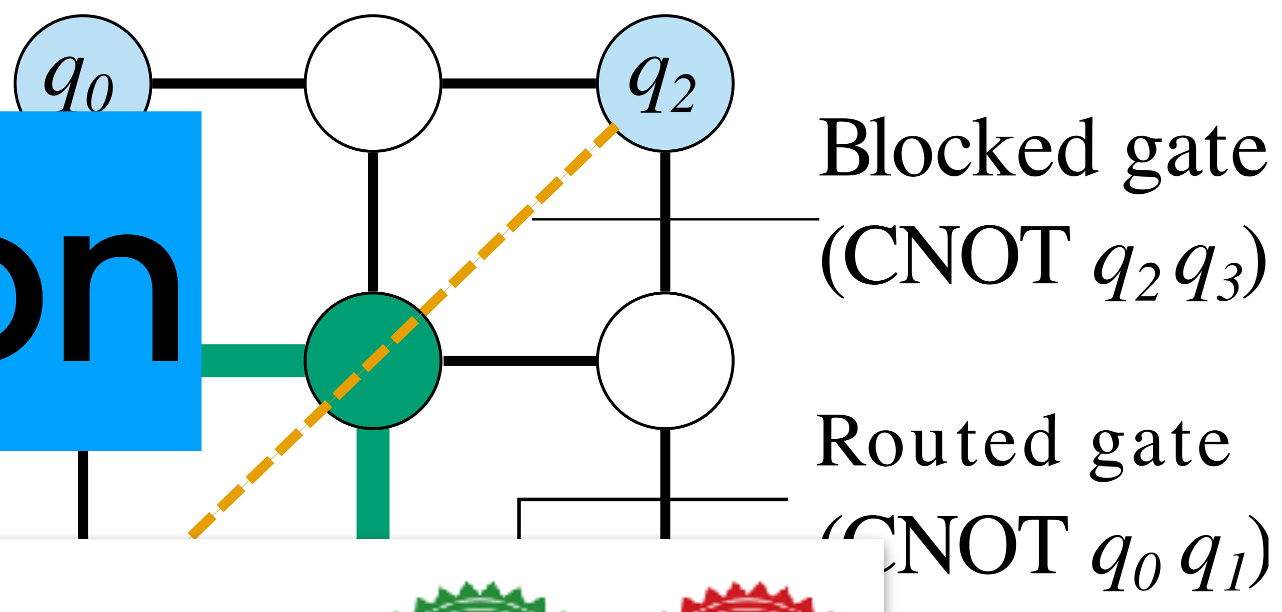
QUESO

Optimizing for FTQC

discrete

Clifford + T gate set: $\{T, T^\dagger, S, S^\dagger, H, CX\}$

"magic state distillation"



Eastin-Knill th

Open Research Question

Reducing T-count with the ZX-calculus

Aleks Kissinger and John van de Wetering

Radboud University Nijmegen

January 20, 2020

Linear and Non-linear Relational Analyses for Quantum Program Optimization

MATTHEW AMY, Simon Fraser University, Canada

JOSEPH LUNDERVILLE, Simon Fraser University, Canada

BQSKit

GUOQ

VOQC

TKET

Qiskit

Quarl

Quartz

QUESO