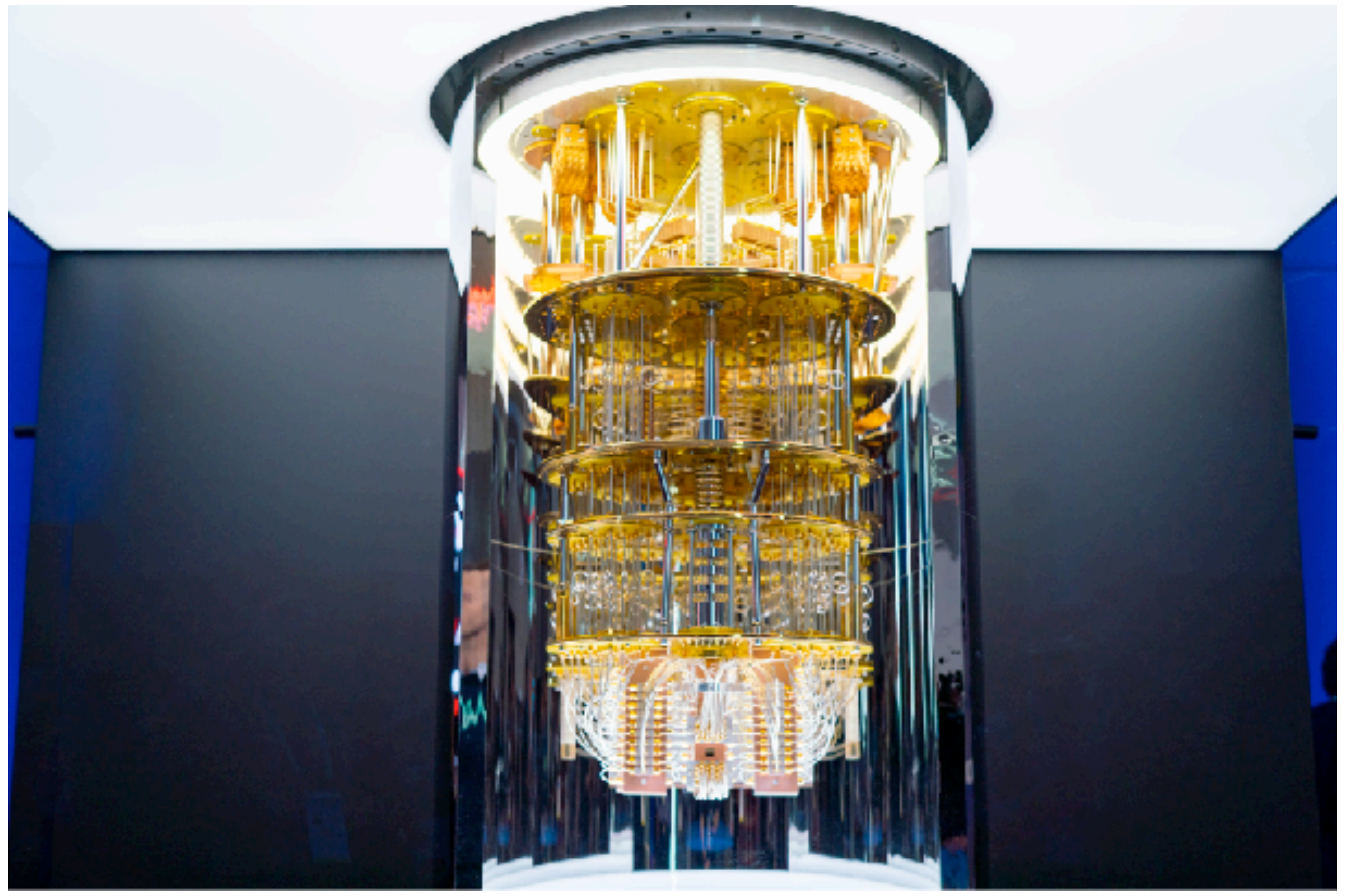
Quantum-Circuit Optimization

Native Basis Gate Set



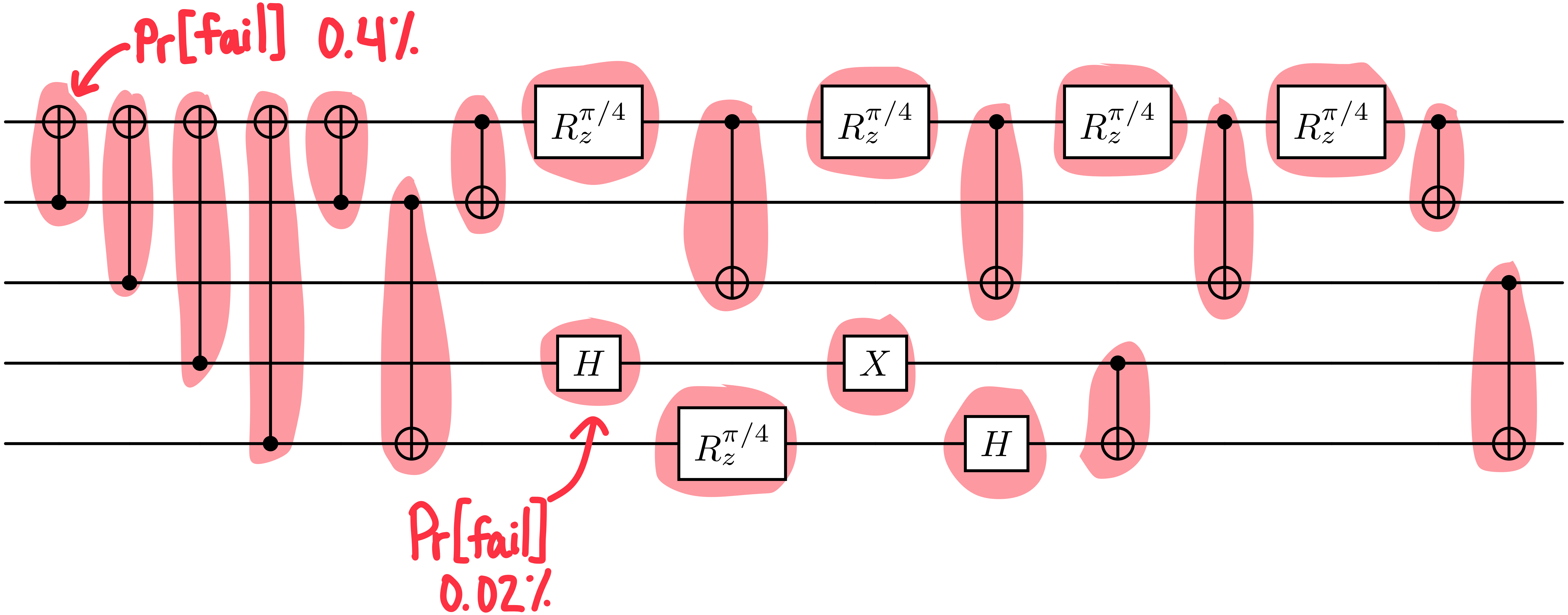
IBM

$$\left\{ R_z^{\theta},\ SX,\ X,\ CX \right\}$$

"universal"

i.e. approximate any computation to arbitrary precision

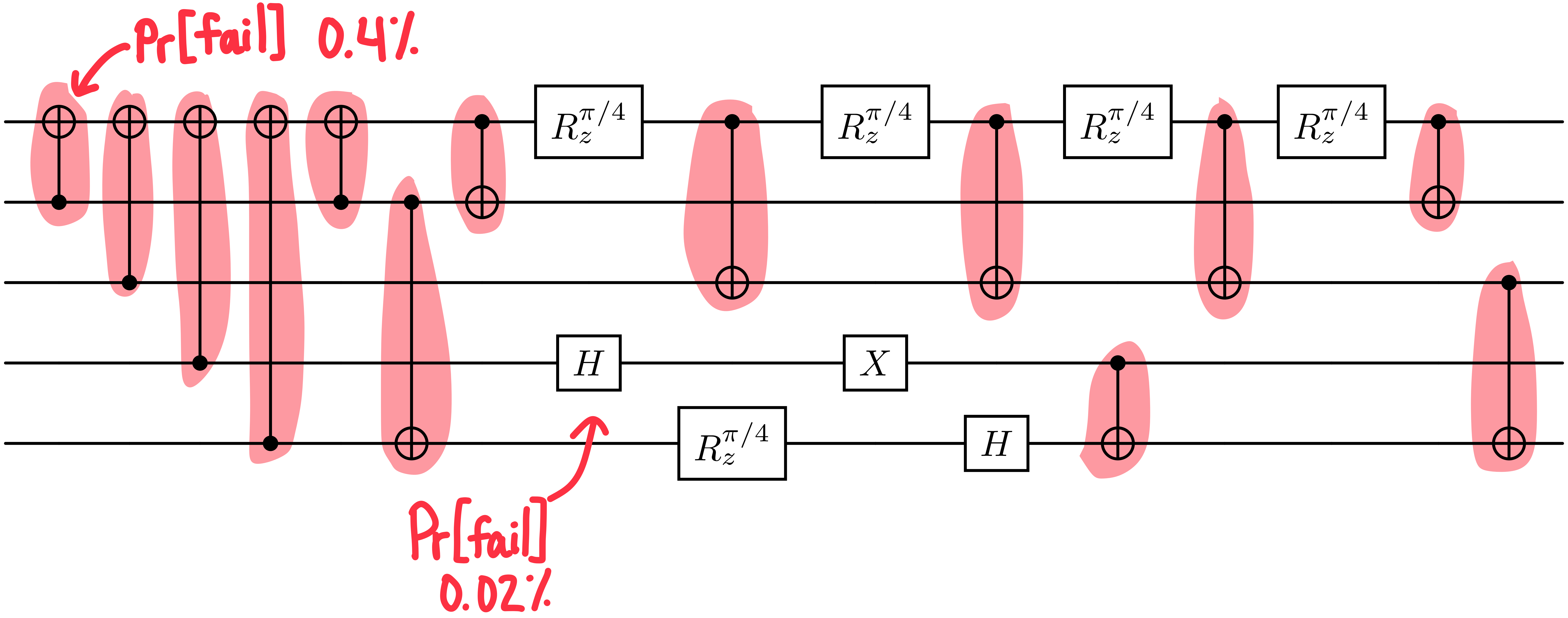
Optimization Objectives



<u>NISQ</u>

- total gate count

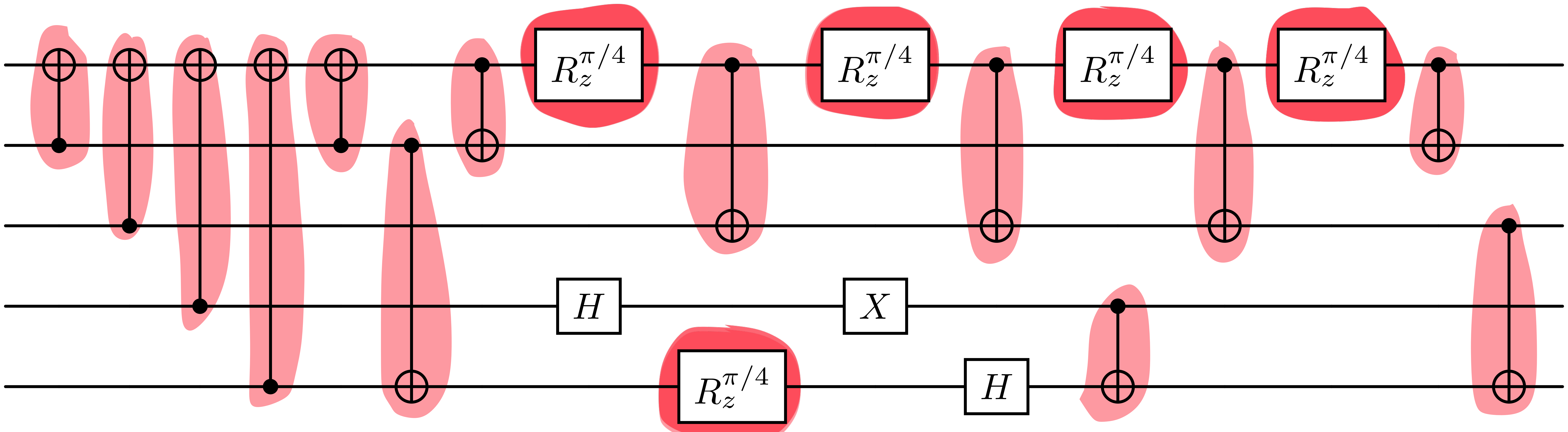
Optimization Objectives



<u>NISQ</u>

- total gate count
- 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

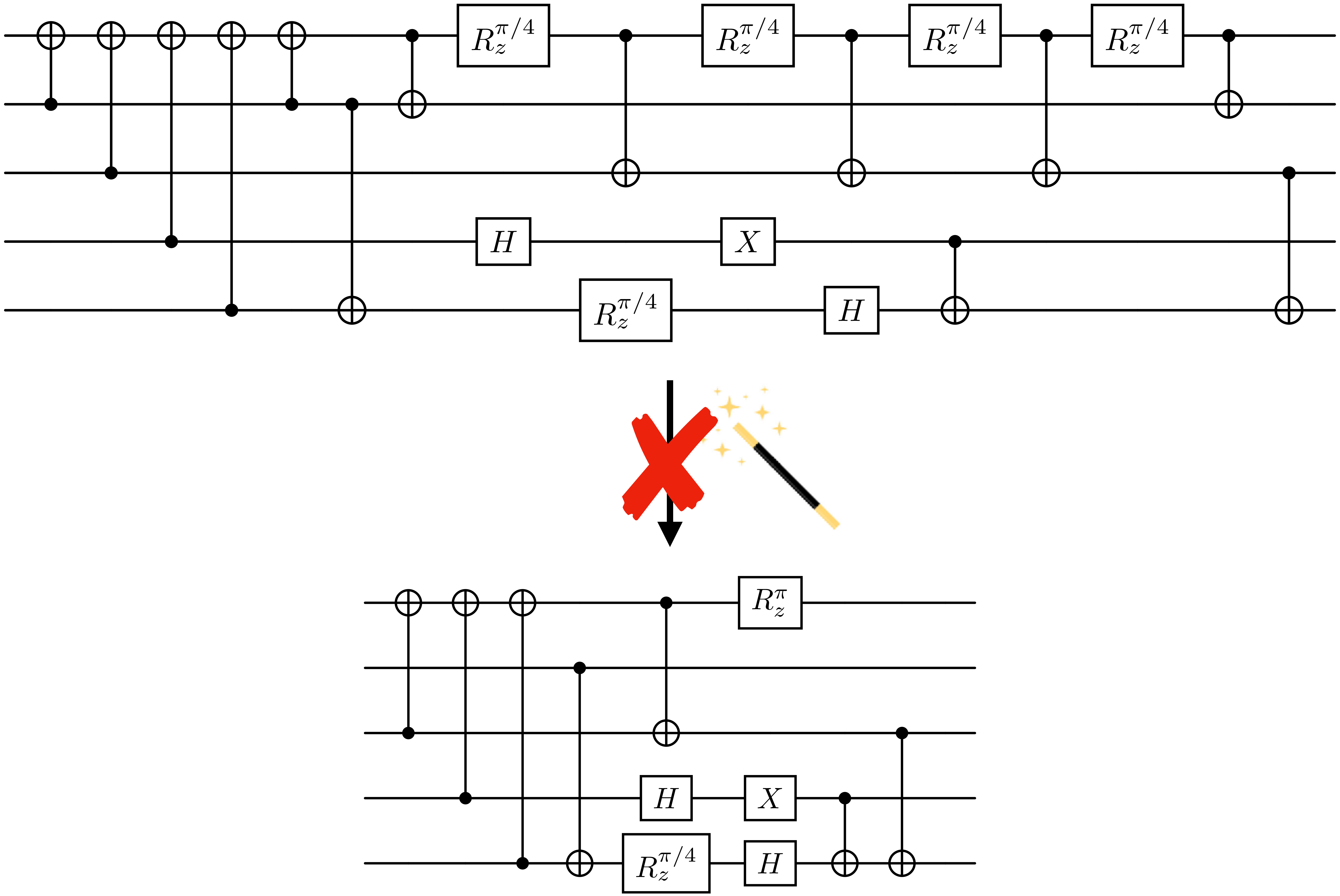
Given a circuit and optimization objective, output an equivalent circuit that minimizes the optimization objective.

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

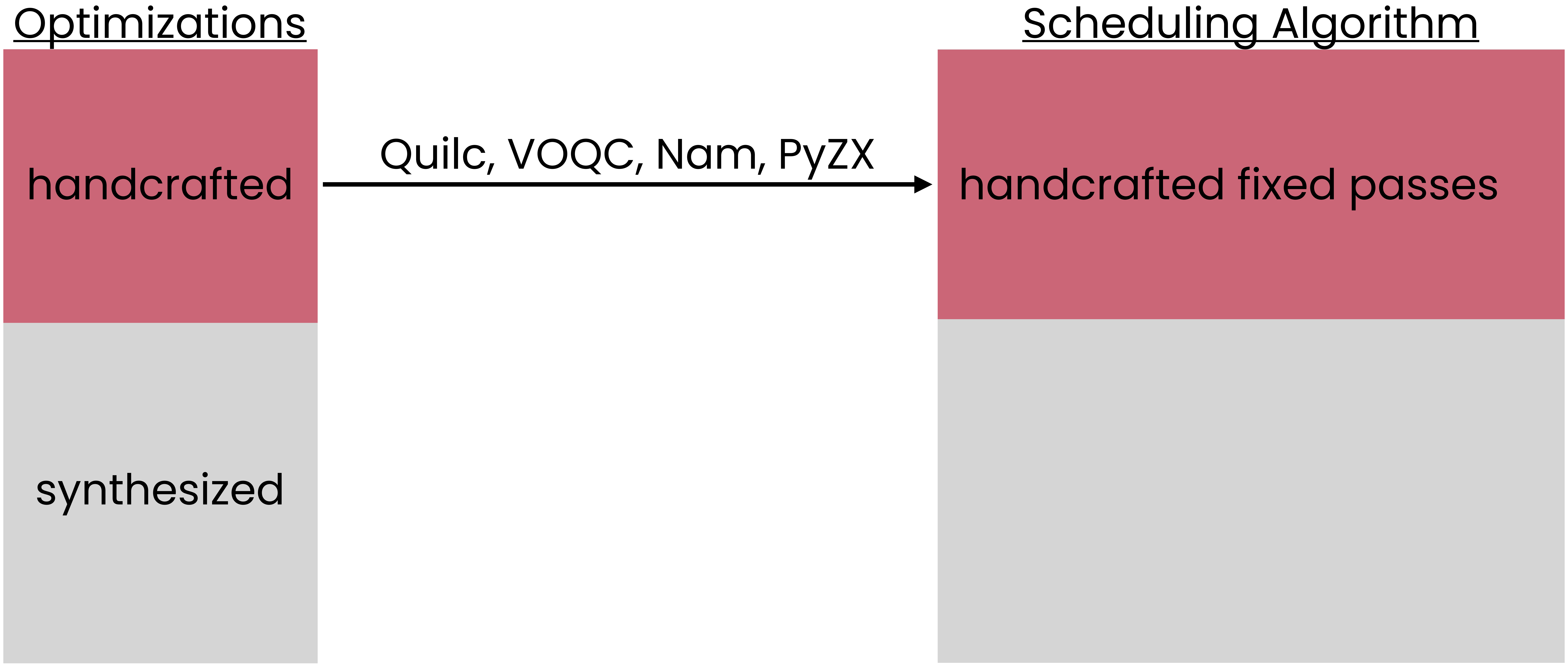


General Approach

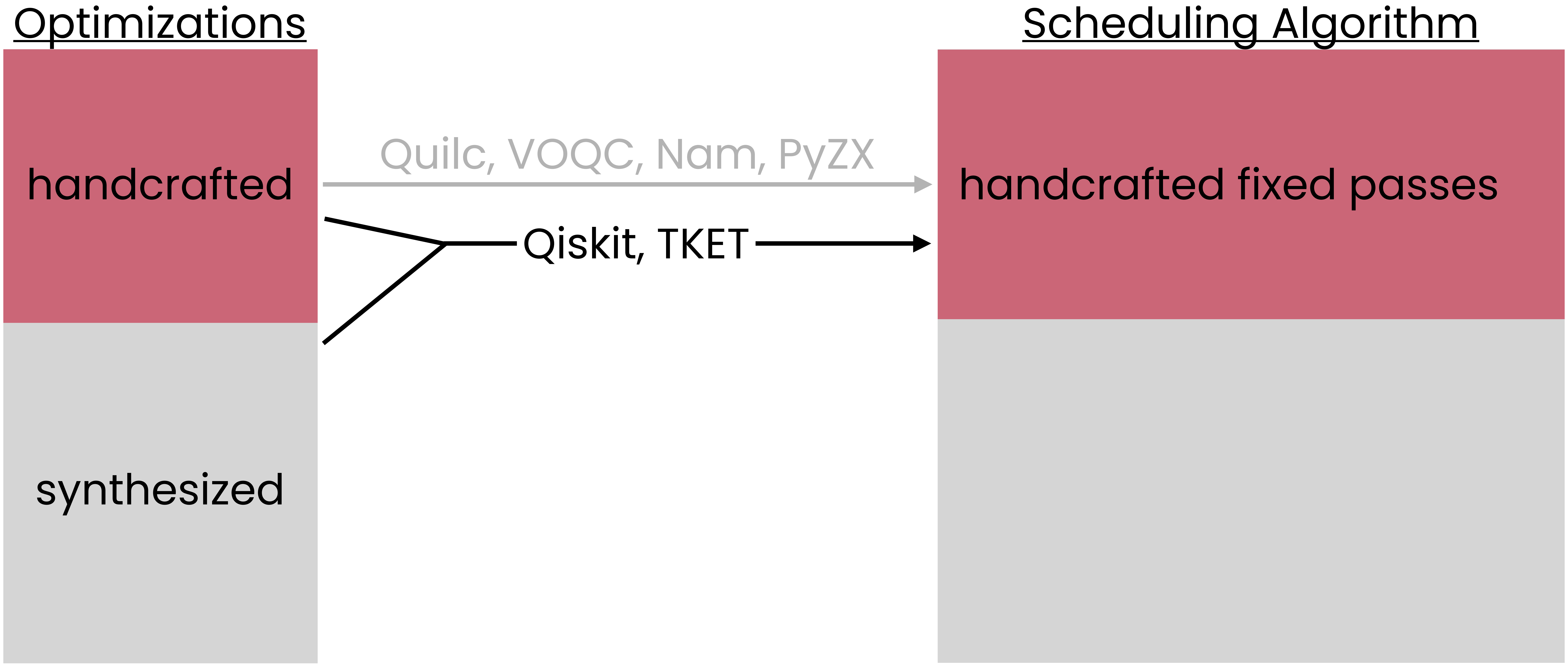
Quantum-Circuit Optimizer

Optimizations + Scheduling Algorithm = Optimizer

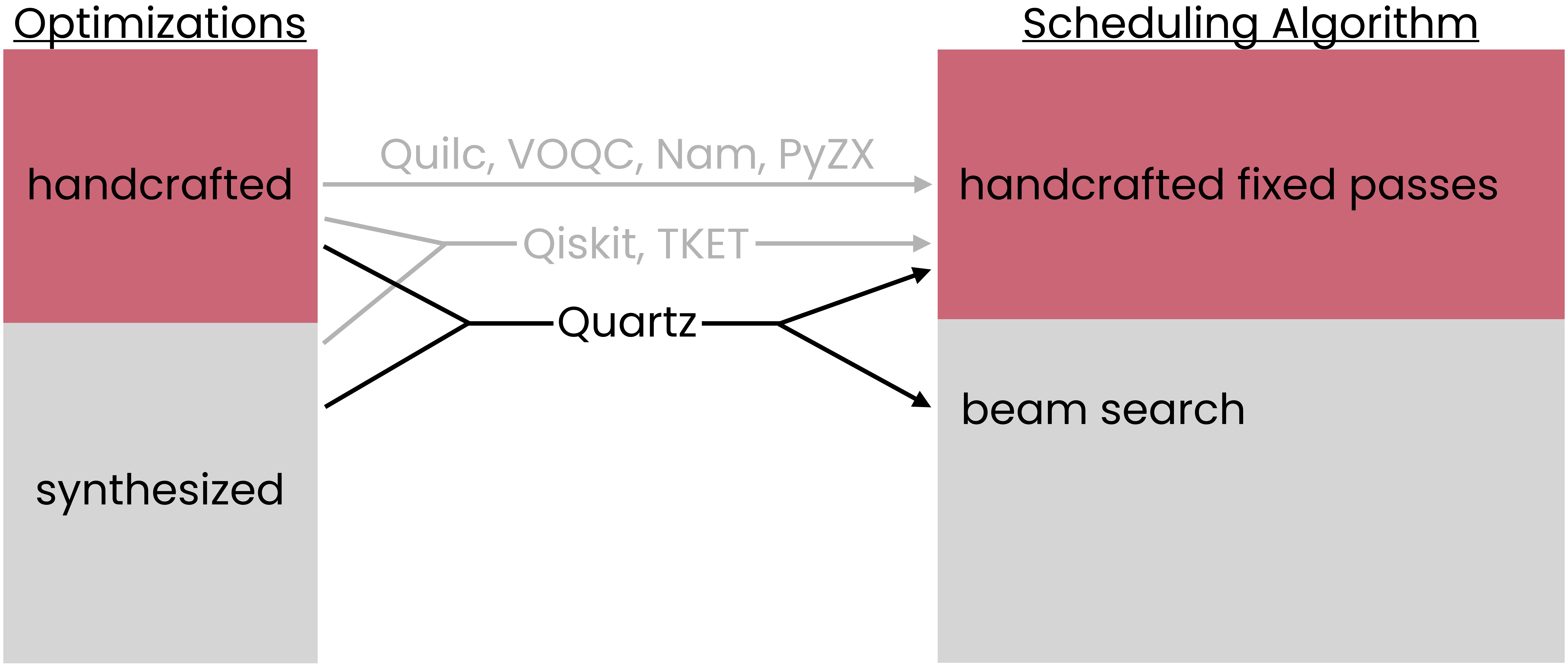
Quantum-Circuit Optimizer Landscape



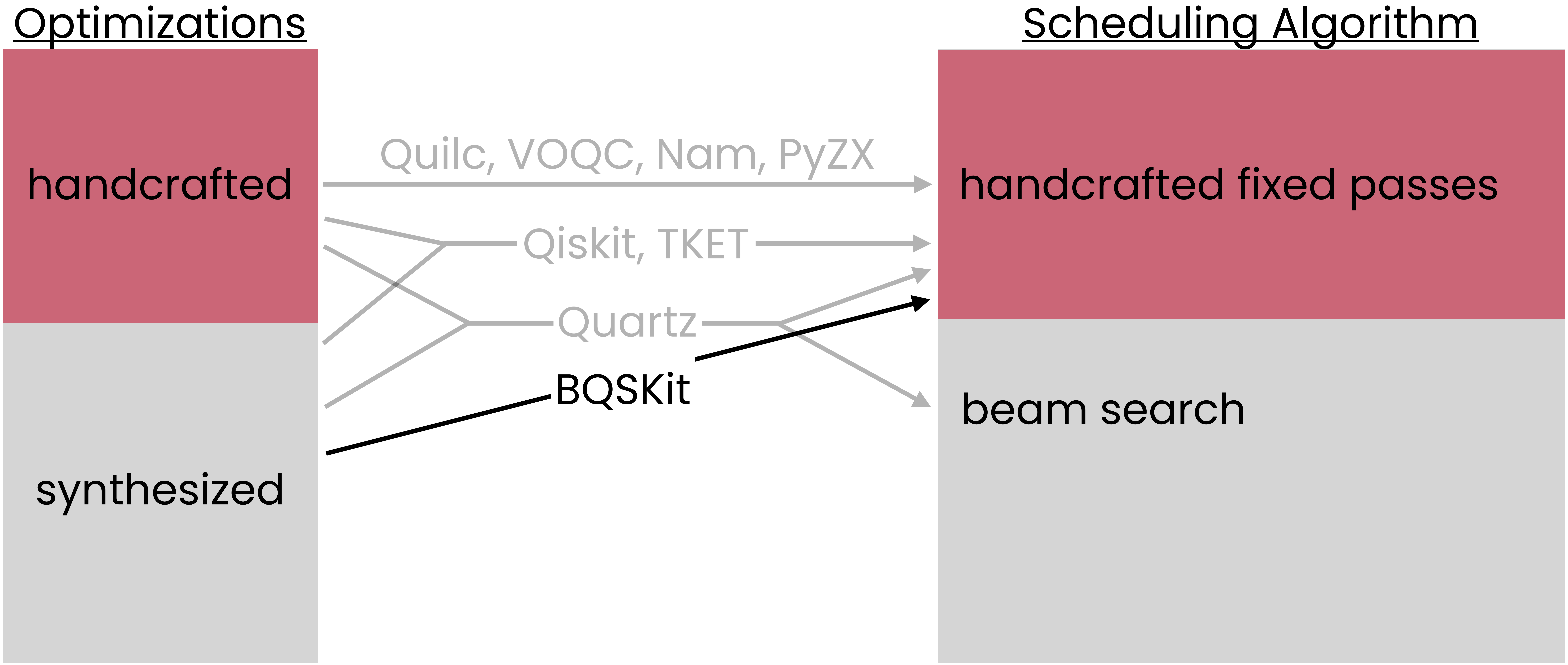
Quantum-Circuit Optimizer Landscape



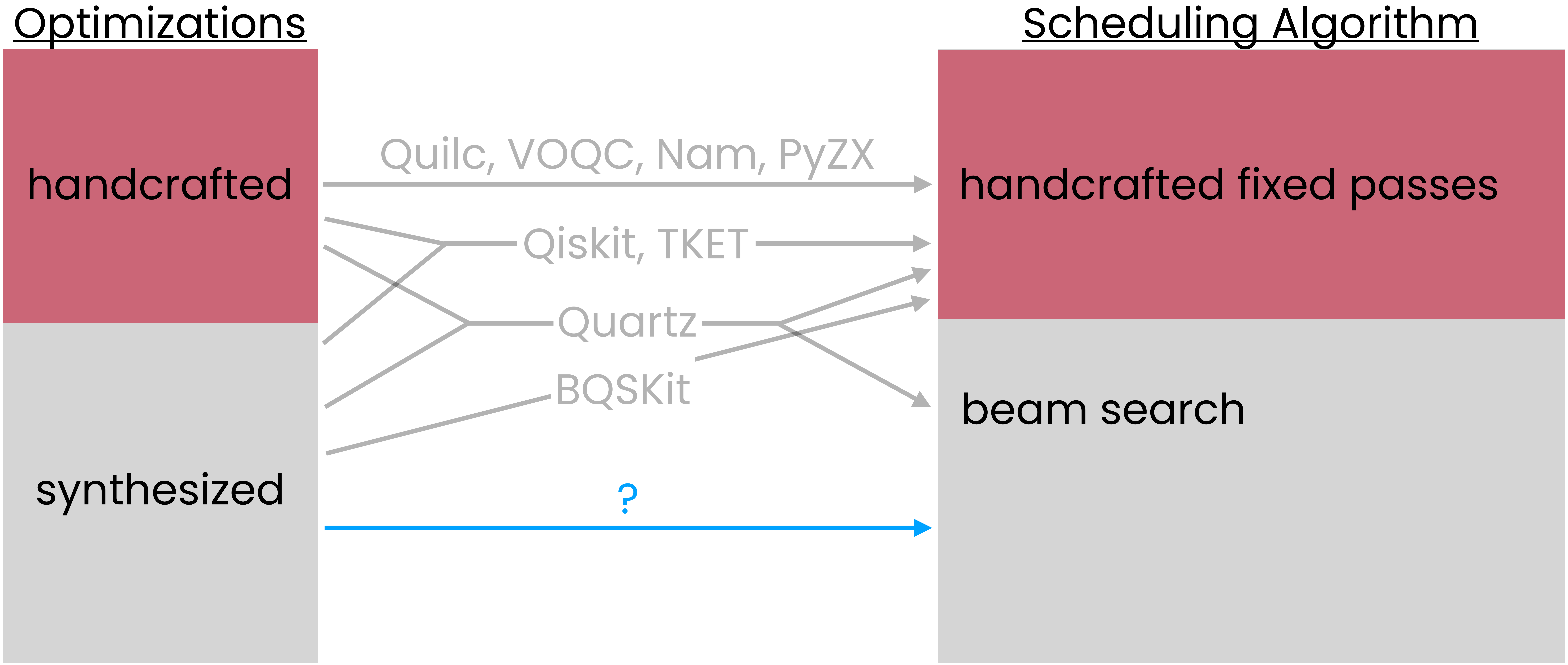
Quantum-Circuit Optimizer Landscape



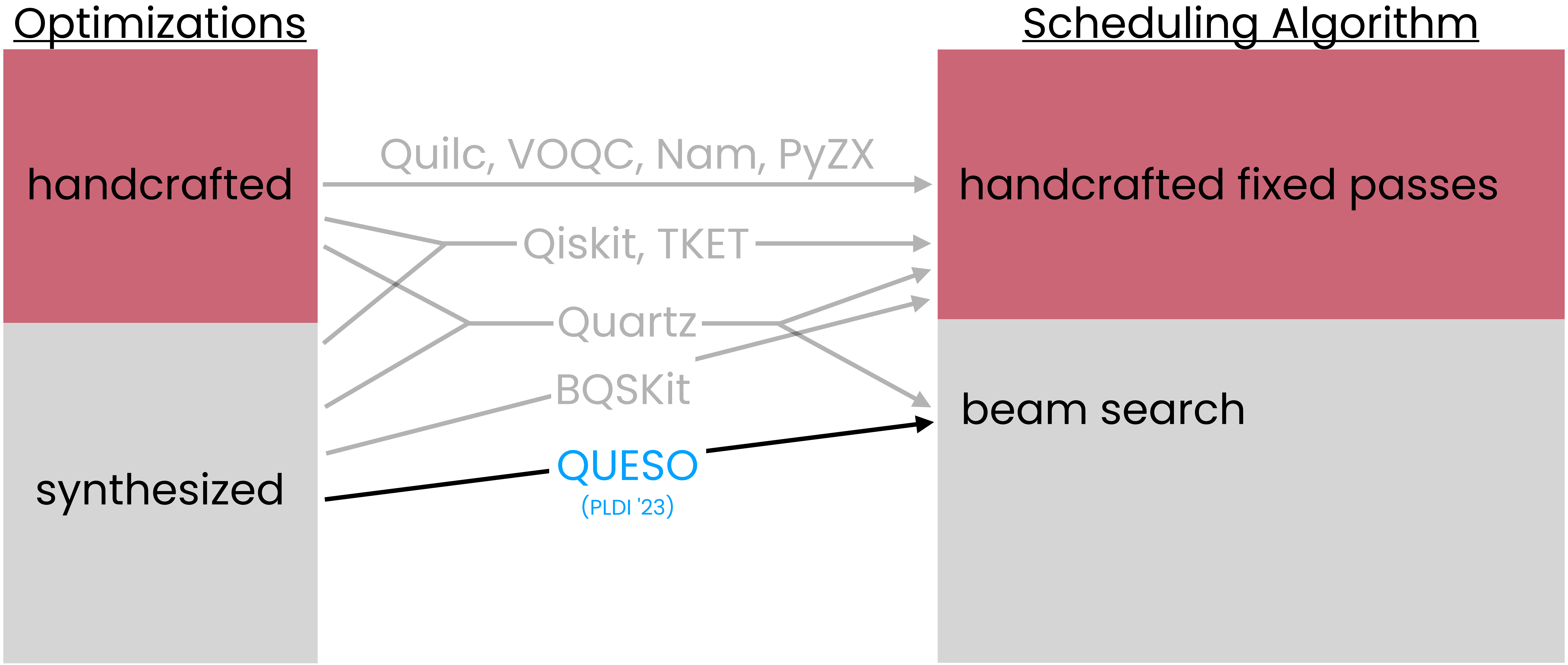
Quantum-Circuit Optimizer Landscape



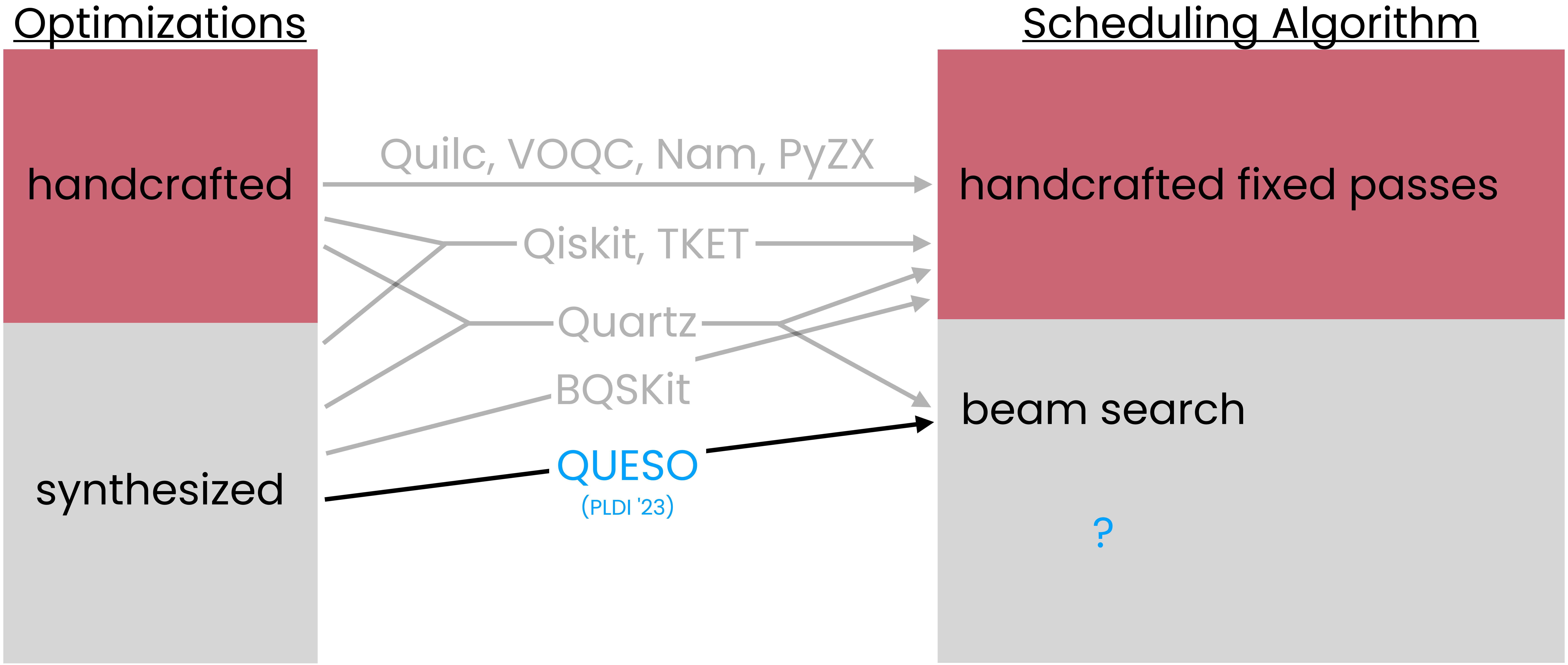
Quantum-Circuit Optimizer Landscape



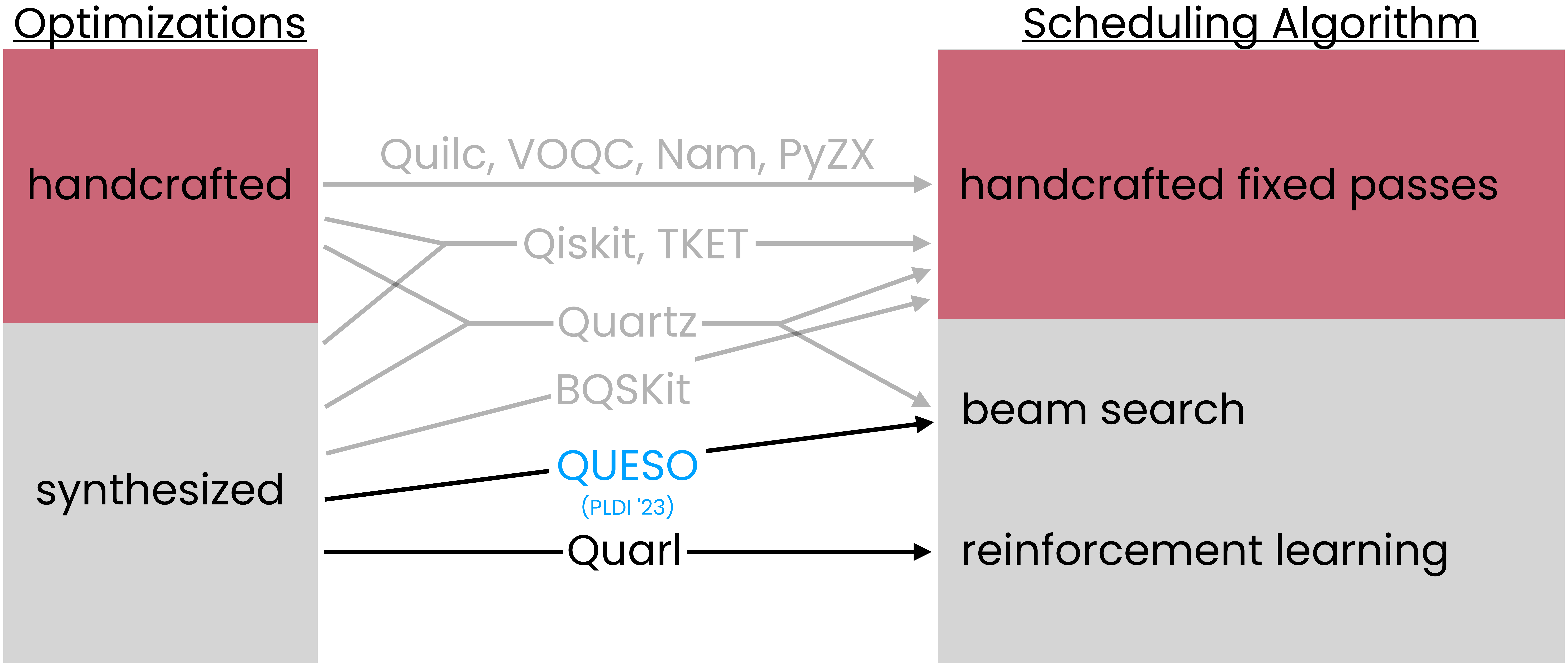
Quantum-Circuit Optimizer Landscape



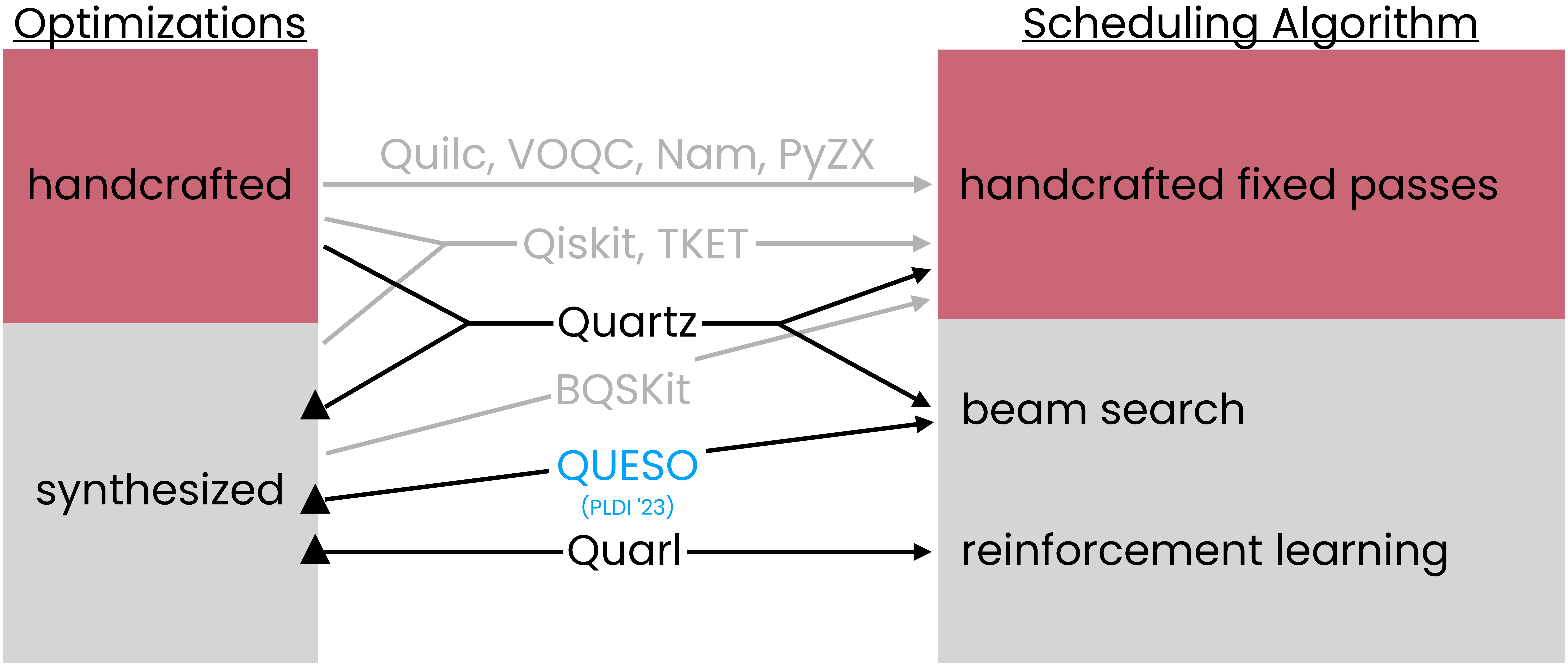
Quantum-Circuit Optimizer Landscape



Quantum-Circuit Optimizer Landscape

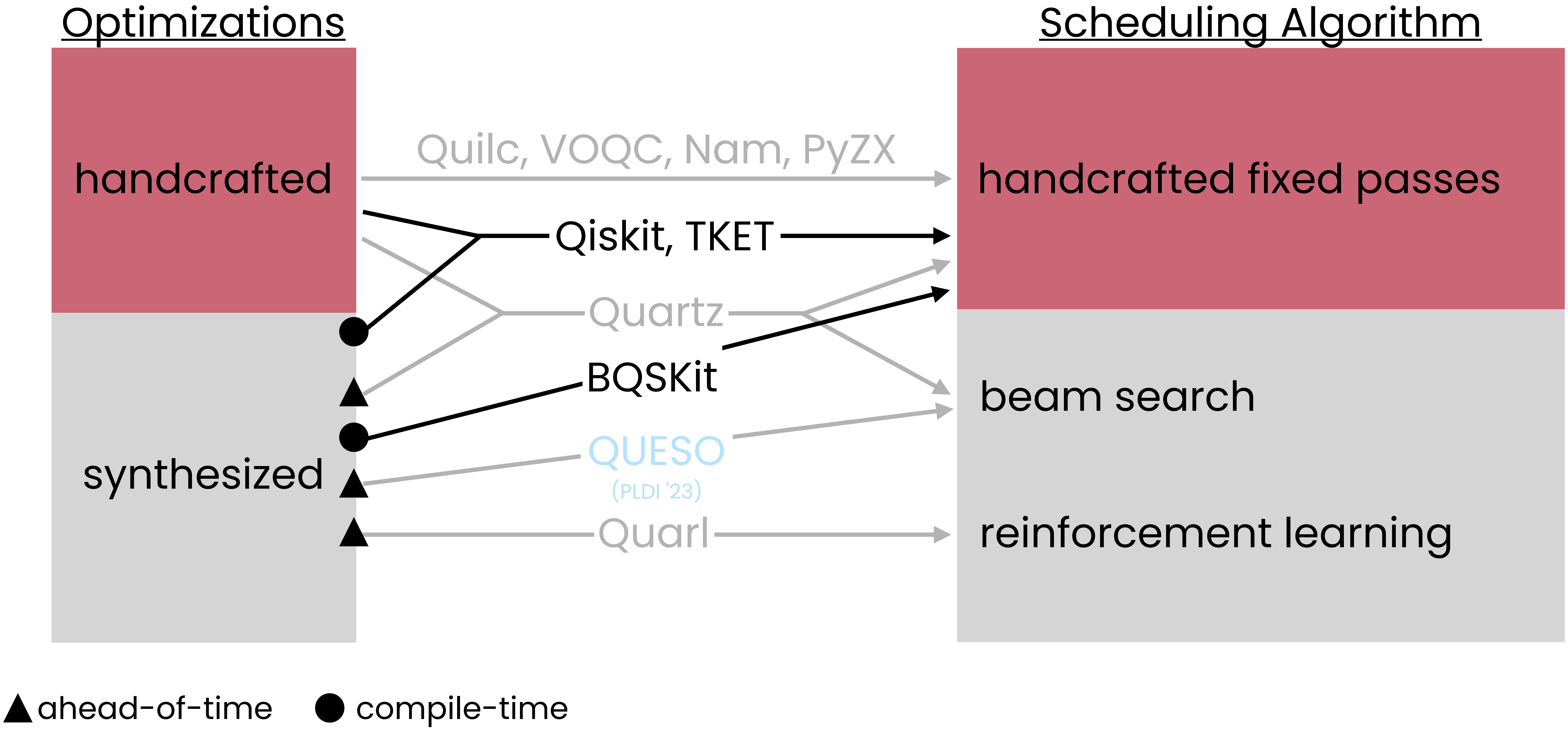


Quantum-Circuit Optimizer Landscape

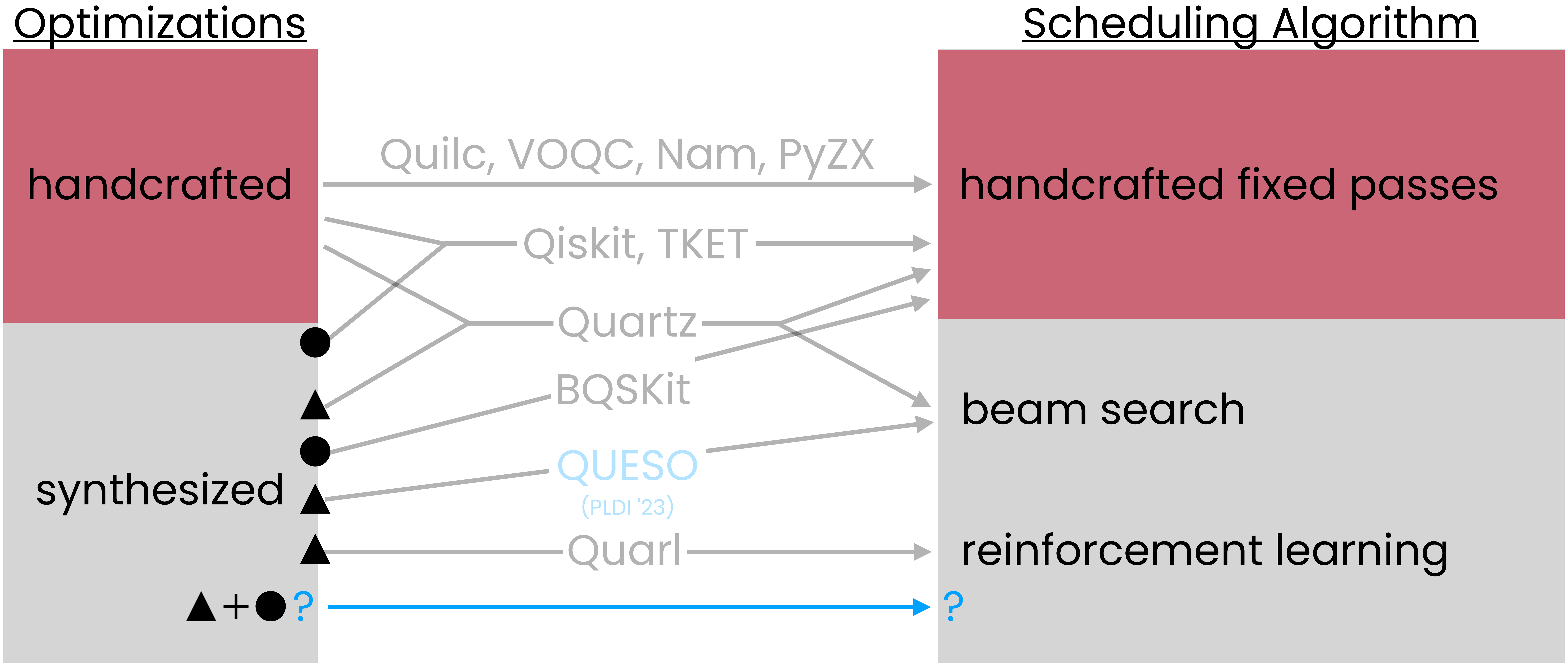


▲ ahead-of-time

Quantum-Circuit Optimizer Landscape

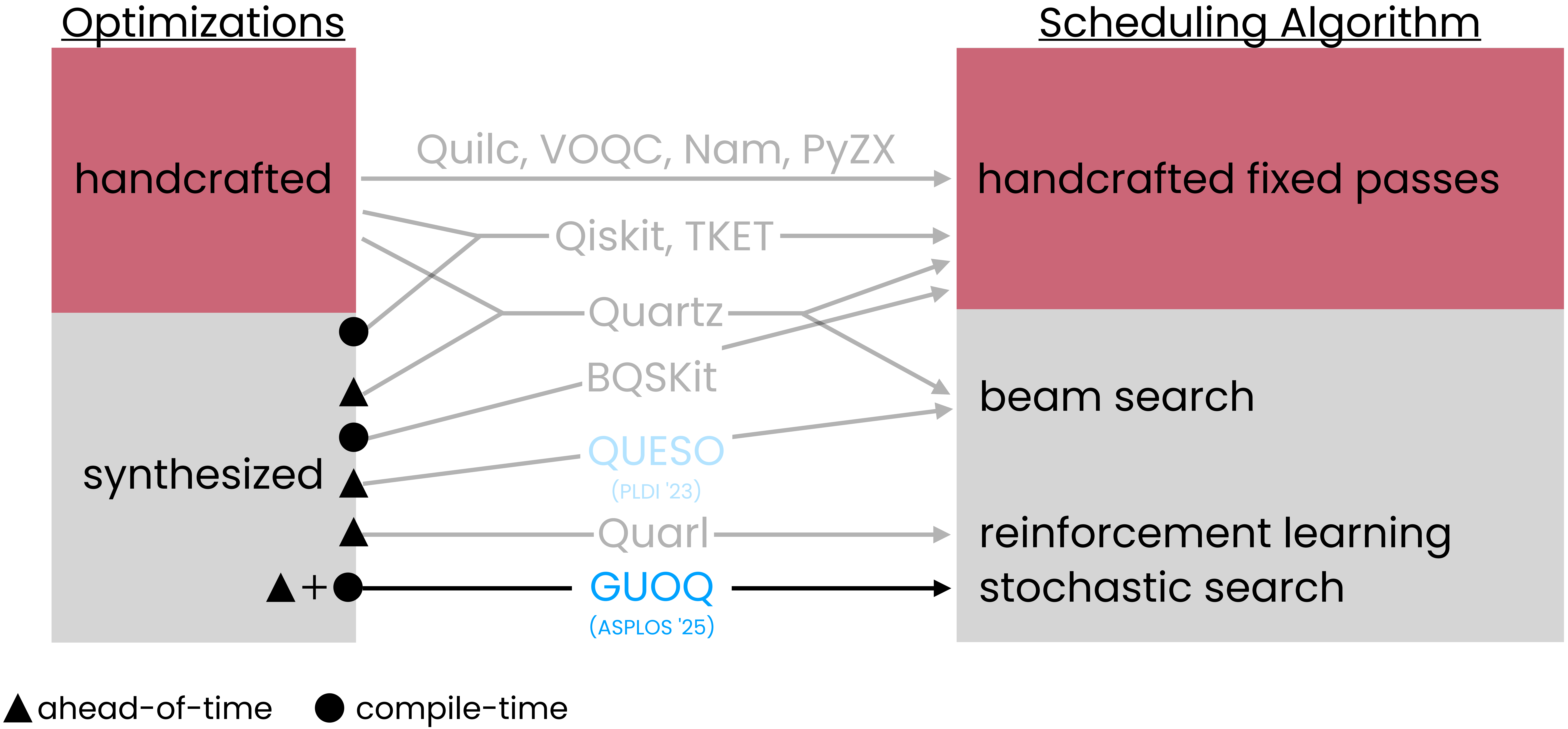


Quantum-Circuit Optimizer Landscape

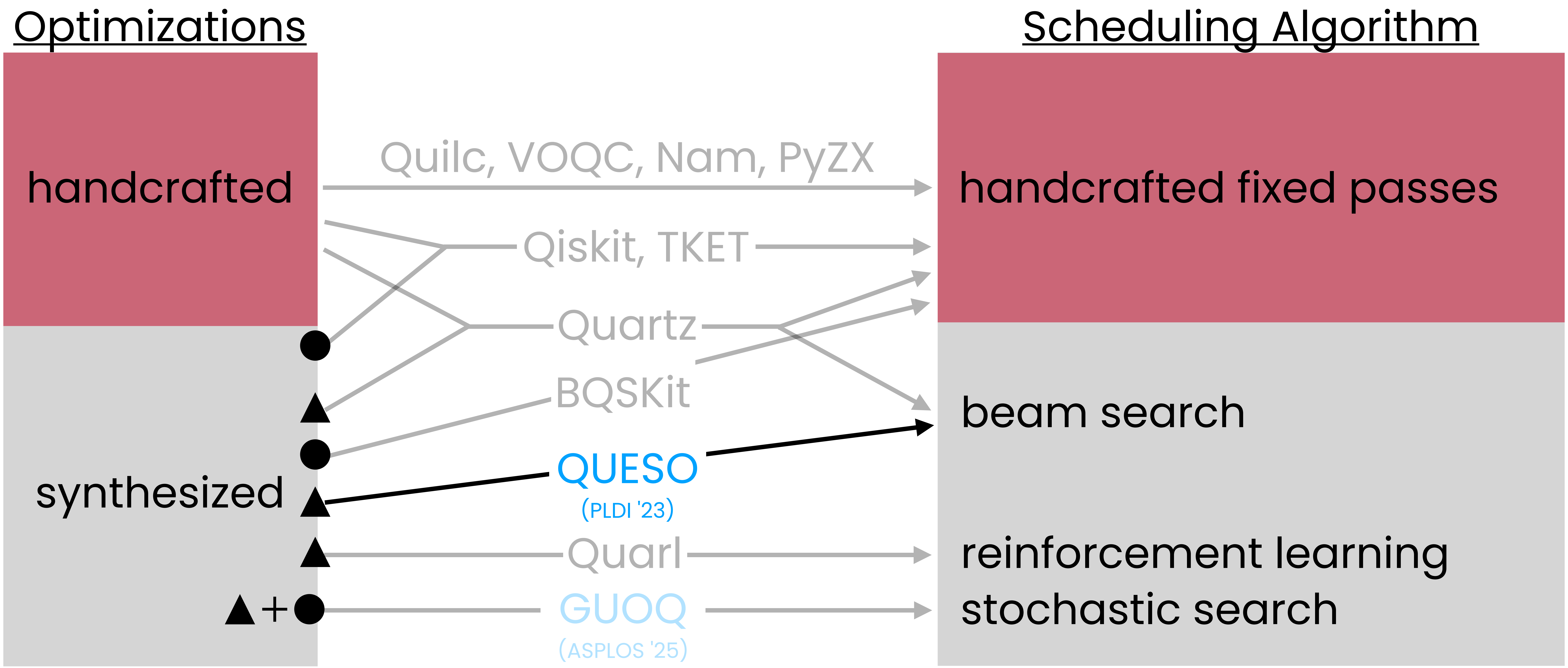


▲ ahead-of-time ● compile-time

Quantum-Circuit Optimizer Landscape



Quantum-Circuit Optimizer Landscape

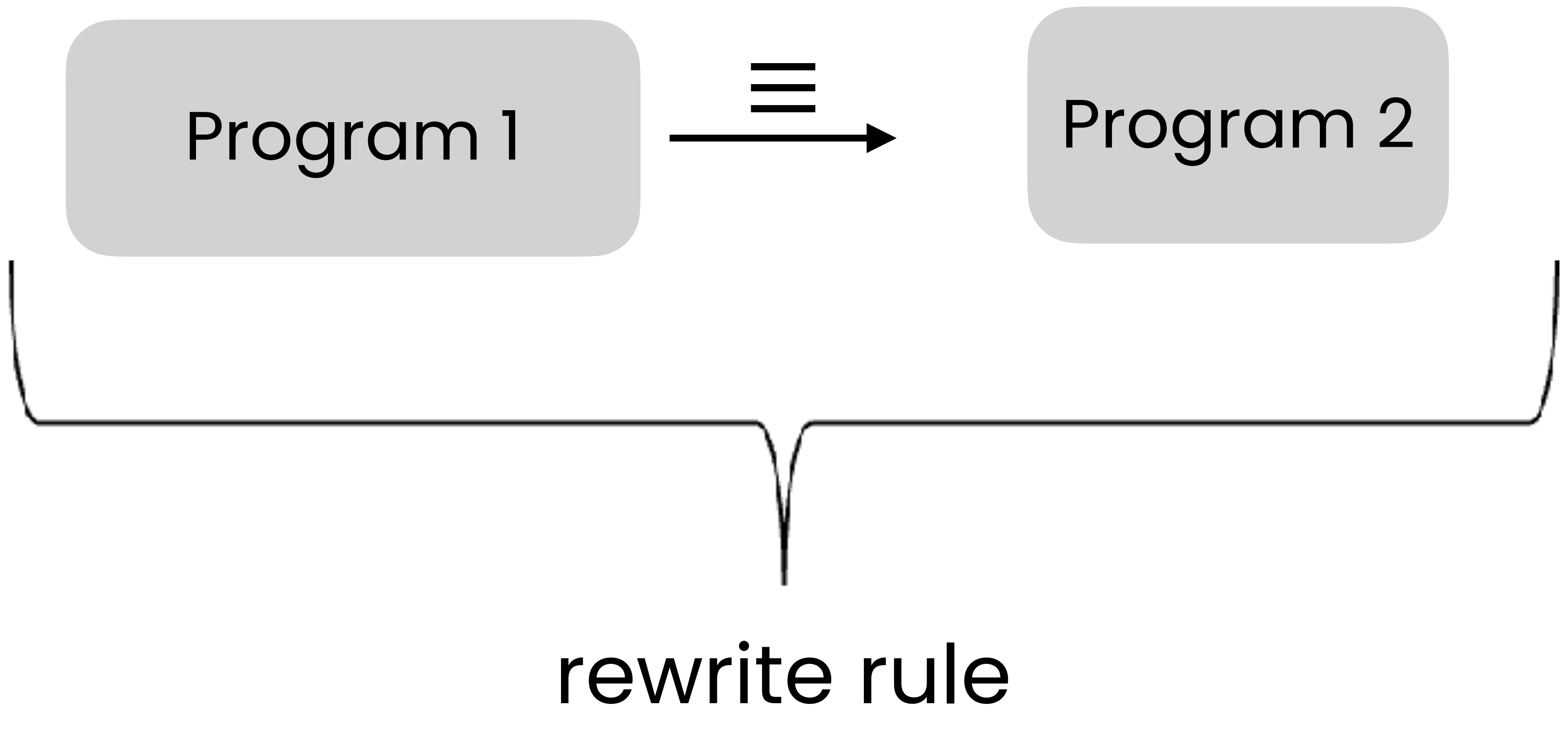


▲ ahead-of-time ● compile-time

Rewrite Rules

Rewrite Rules

"peephole optimization"



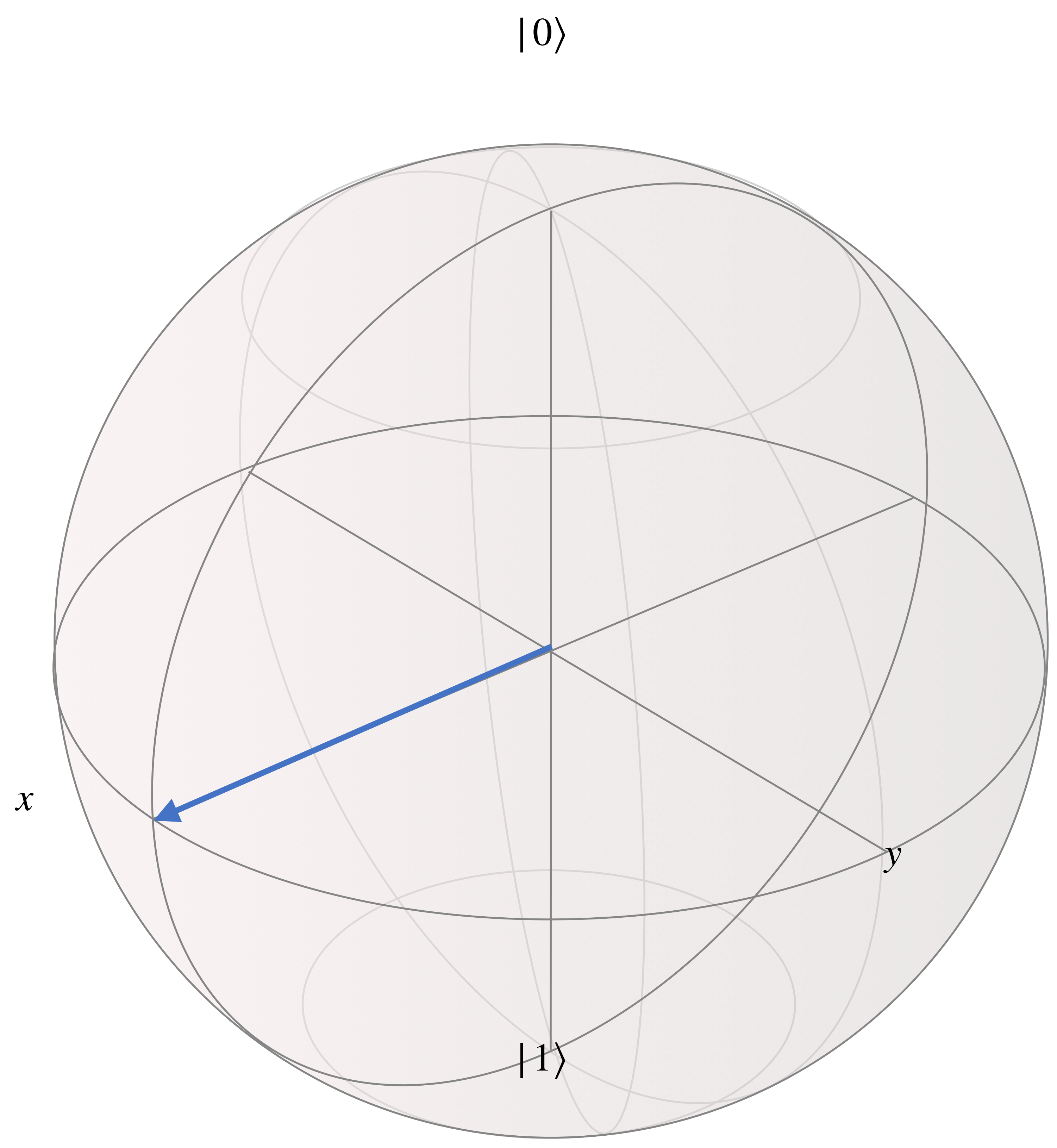
Circuit Equivalence

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

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

"global phase"

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

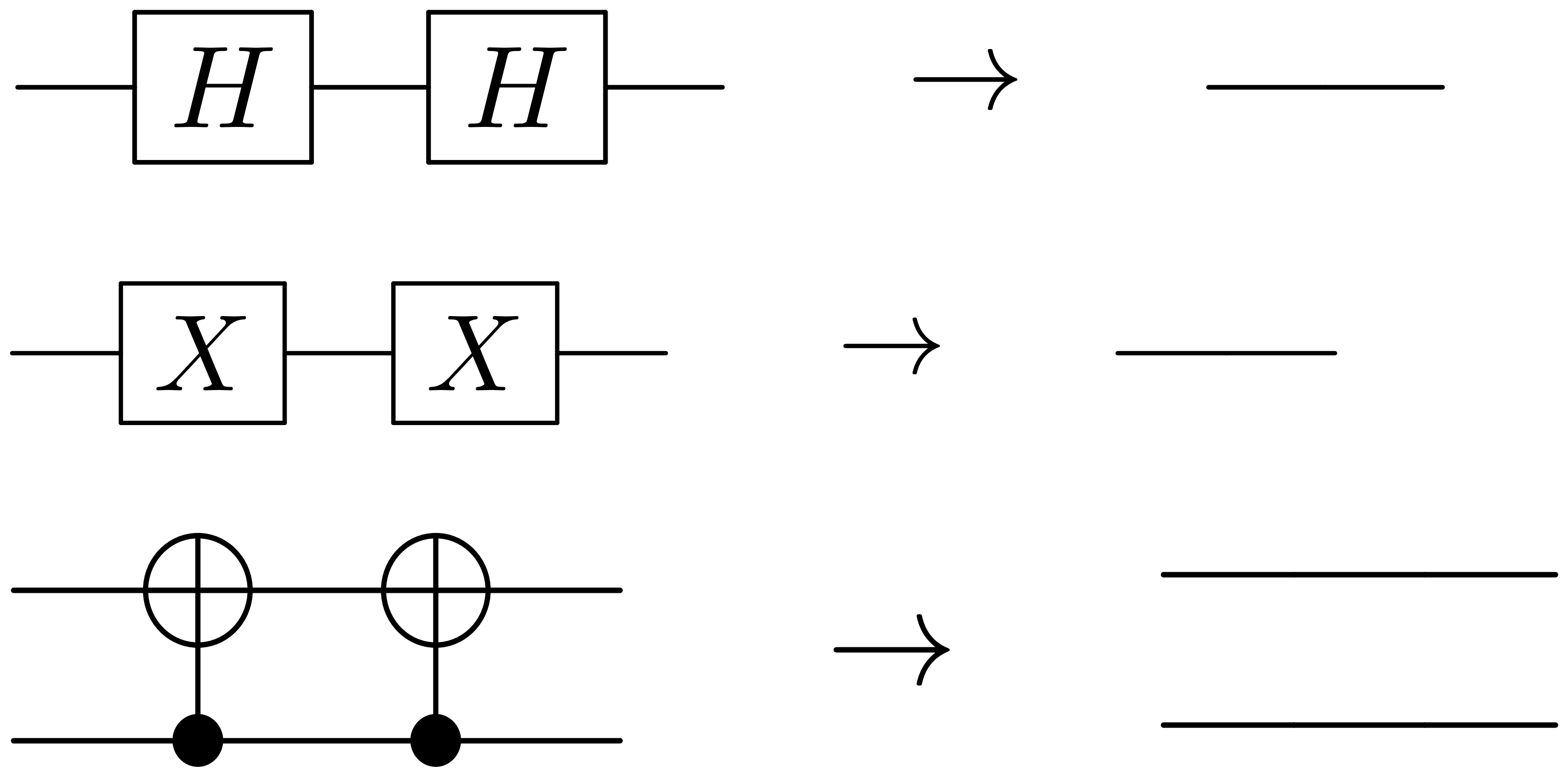


Hard for even a quantum computer to verify

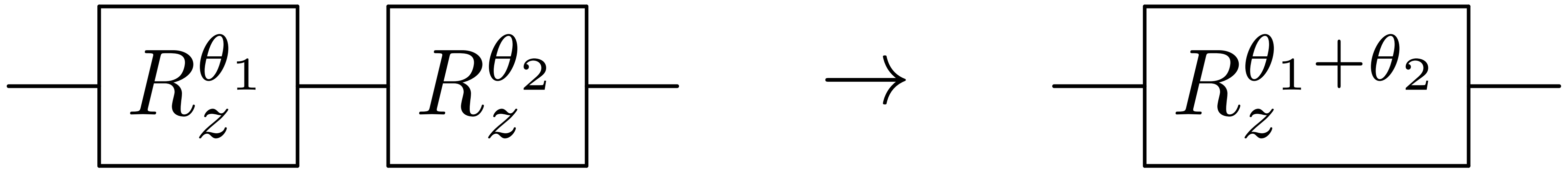
(QMA-Complete)

Simple Rewrite Rules

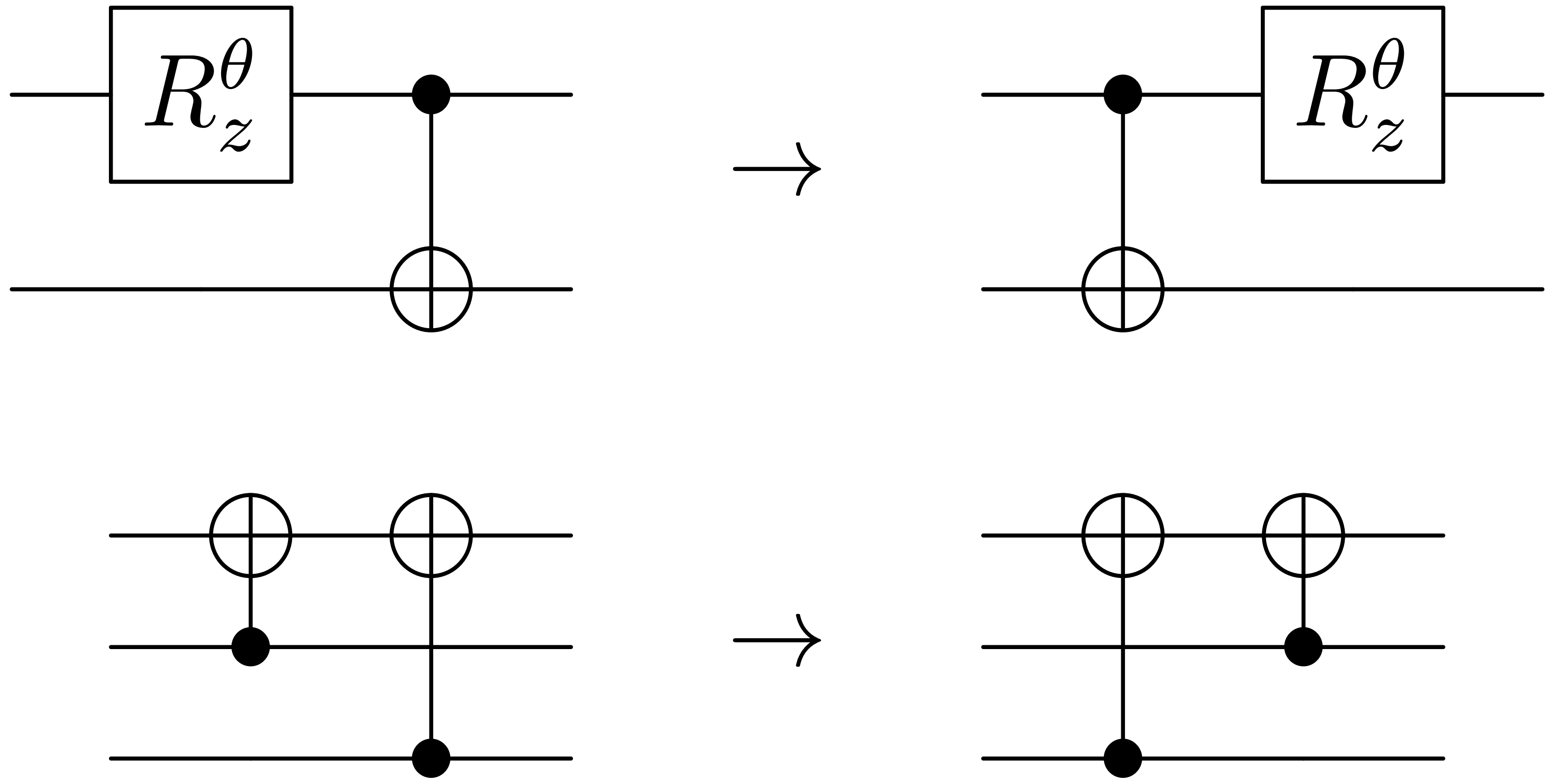
Cancel



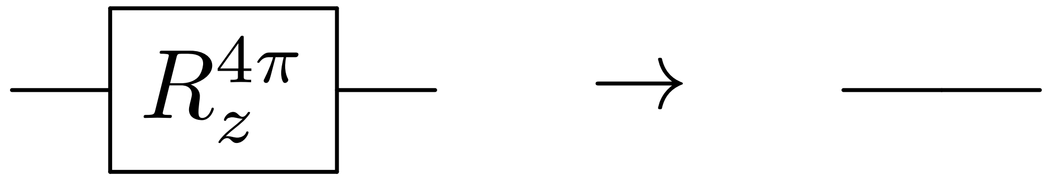
Merge



Commute



Global Phase



Diverse Hardware

Ion Trap

IONQ

QUANTINUUM

Topological

Hardware

Superconducting

rigetti

OQC

Google Quantum AI

IBM

aws

Neutral Atom

|QuEra>

COMPUTING INC.

Diverse Hardware

Ion Trap

IONQ

QUANTINUUM

Topological

Hardware

Superconducting

rigetti

OQC

Google Quantum AI

IBM

aws

Neutral Atom

|QuEra>

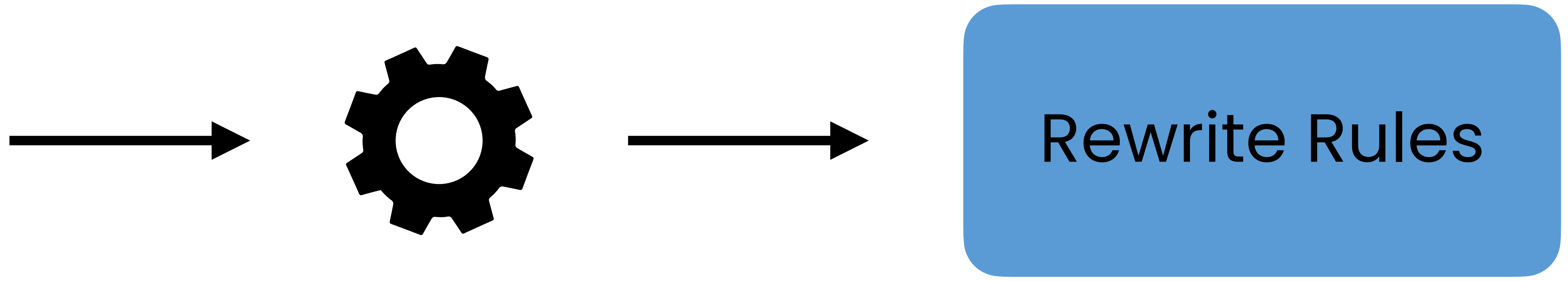
COMPUTING INC.

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

→ Rz, Rx, SX, X, CZ, Rzz

Synthesis!

<u>Arbitrary gate set</u>



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...

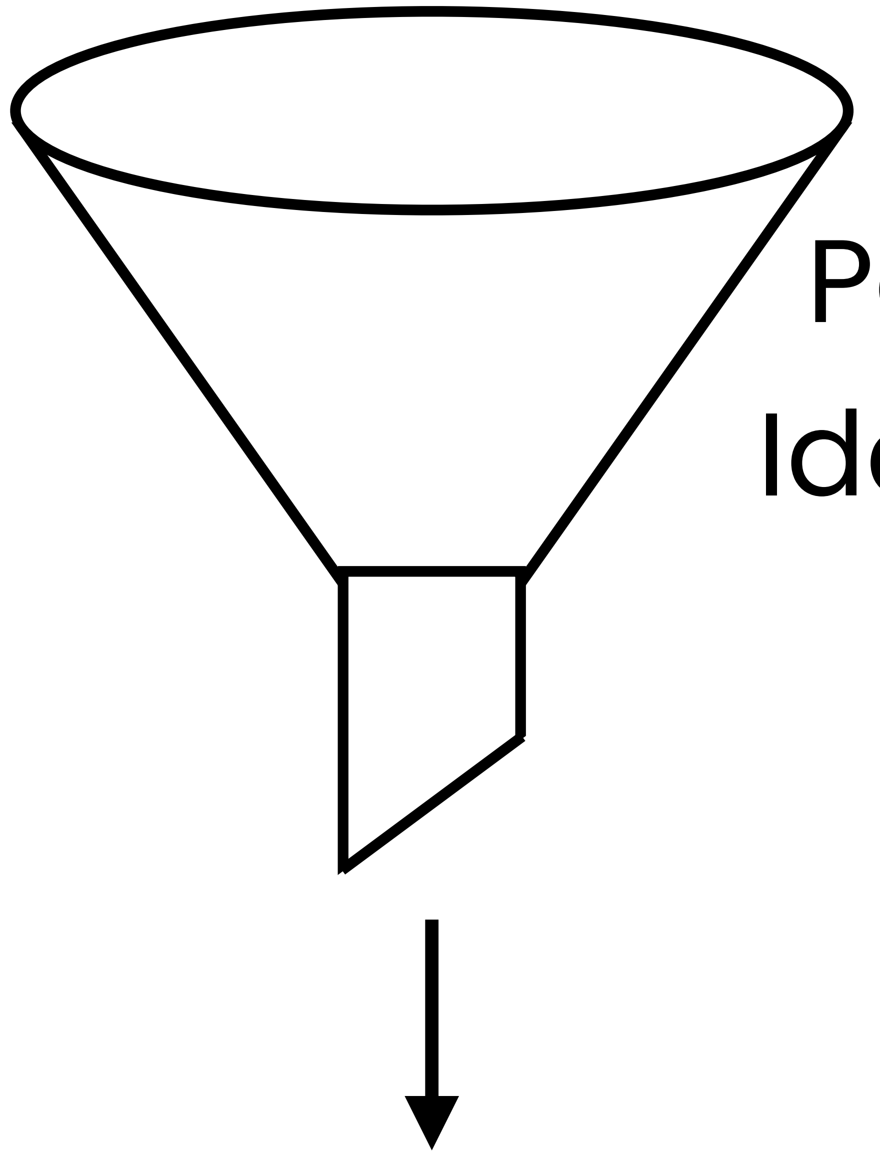
expensive

QUESO

Schwartz-Zippel Lemma for polynomial identity testing (PIT)

Feynman's path integral formulation for quantum mechanics

(Symbolic) Circuit → Polynomial →



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

Schwartz-Zippel Lemma

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

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

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

Schwartz-Zippel Lemma

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

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

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

finite $R \subset \mathbb{C}$

Schwartz-Zippel Lemma

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

$$p_1(\mathbf{x}) = x_1x_2$$

$$p_2(\mathbf{x}) = x_1x_2 + x_3x_2$$

finite $R \subset \mathbb{C}$

α_{x_1}

Schwartz-Zippel Lemma

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

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

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

finite $R \subset \mathbb{C}$

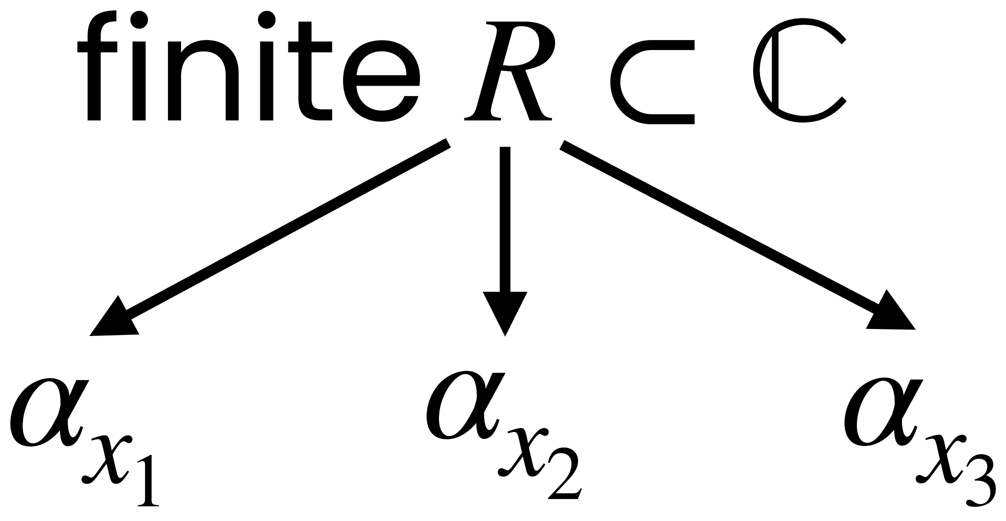
α_{x_1} α_{x_2}

Schwartz-Zippel Lemma

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

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

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



Schwartz-Zippel Lemma

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

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

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

finite $R \subset \mathbb{C}$

α_{x_1} α_{x_2} α_{x_3}

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

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

Schwartz-Zippel Lemma

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

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

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

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

Schwartz-Zippel Lemma

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

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

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

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

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

Schwartz-Zippel Lemma

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

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

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

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

QUESO's Improvements

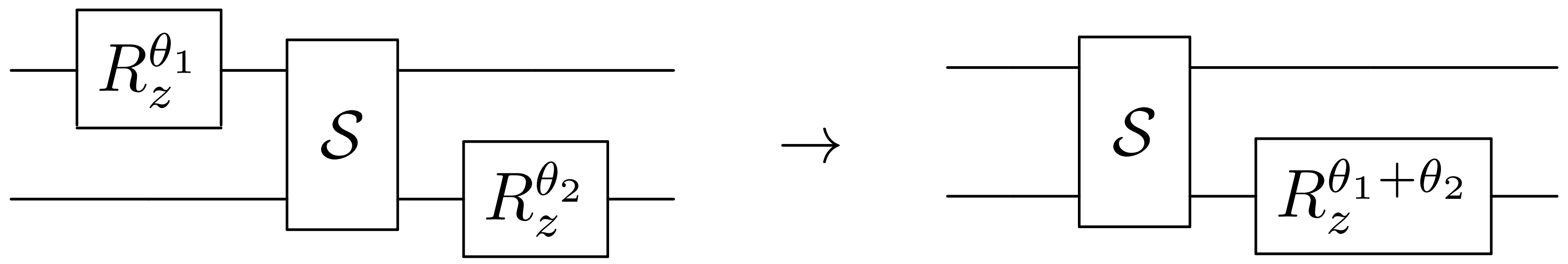
```
rules = []
circuits = enumerate(max_qubits, max_size)
for c1 in circuits:
  for c2 in circuits:
    if verify(c1, c2):
insert c1 polynomial into PIF
      rules.append(c1 → c2)
```

big

uh oh...

~~expensive~~

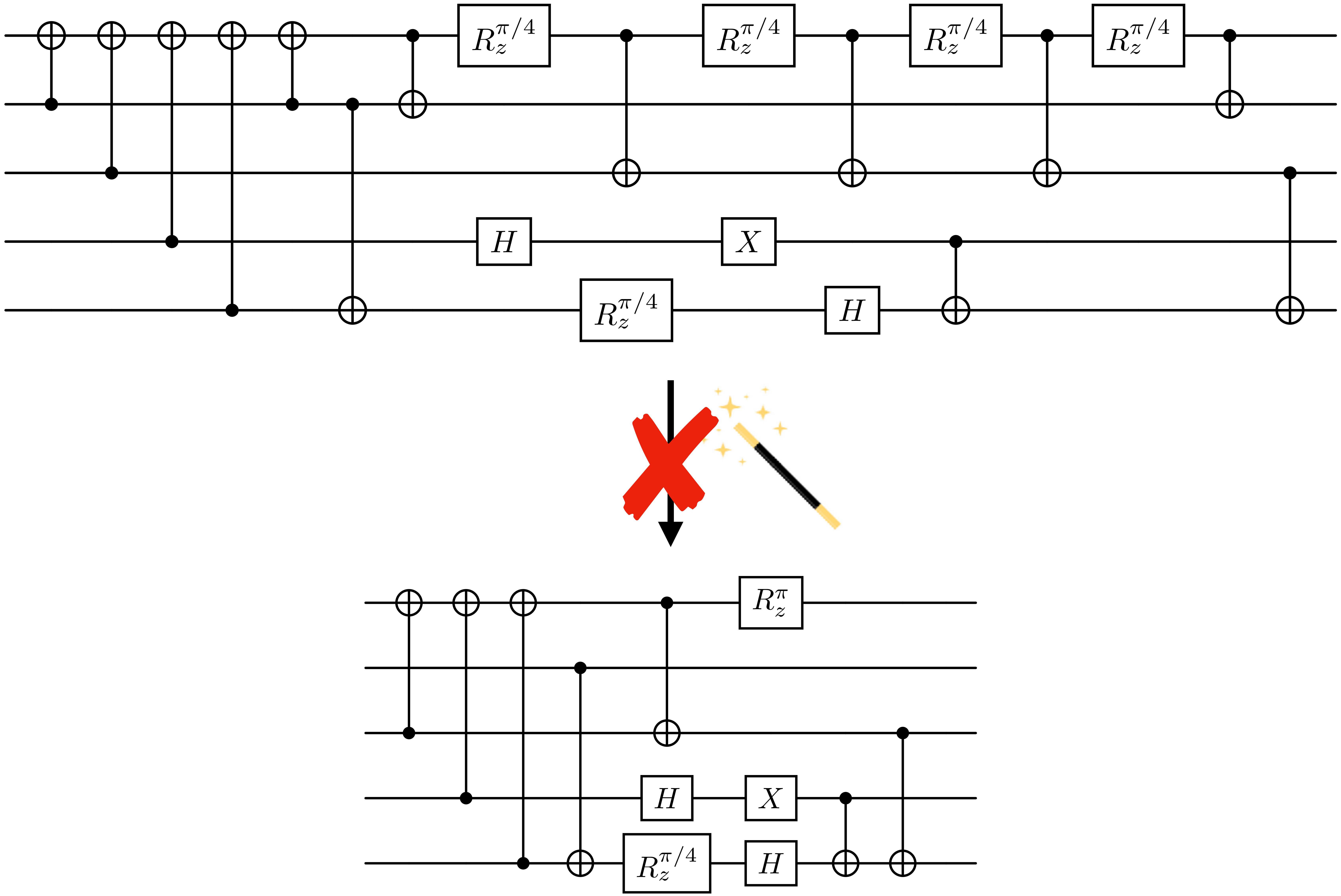
Efficient equivalence check between circuit pairs!

bonus: 

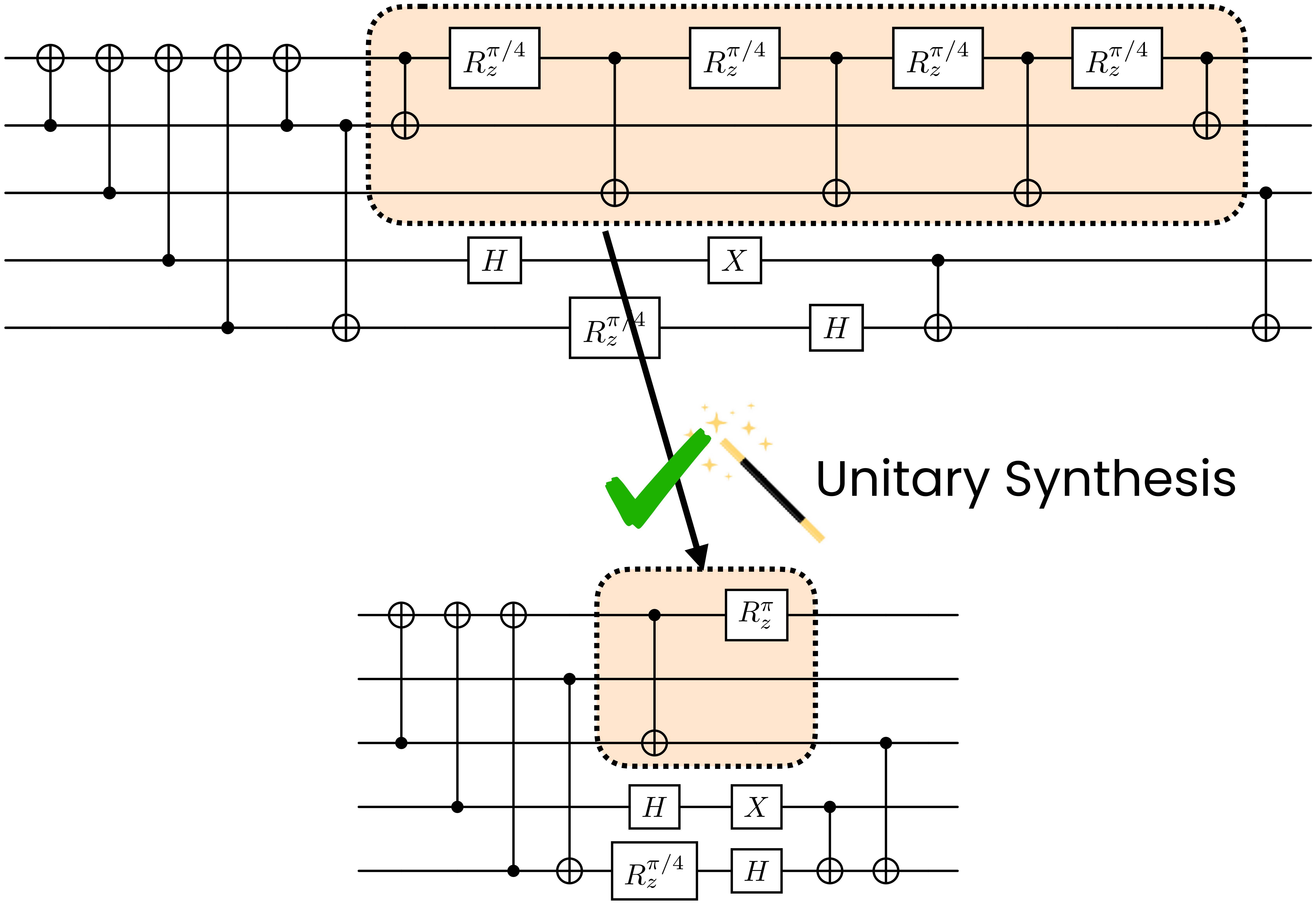
(advanced symbolic rules)

Circuit Resynthesis

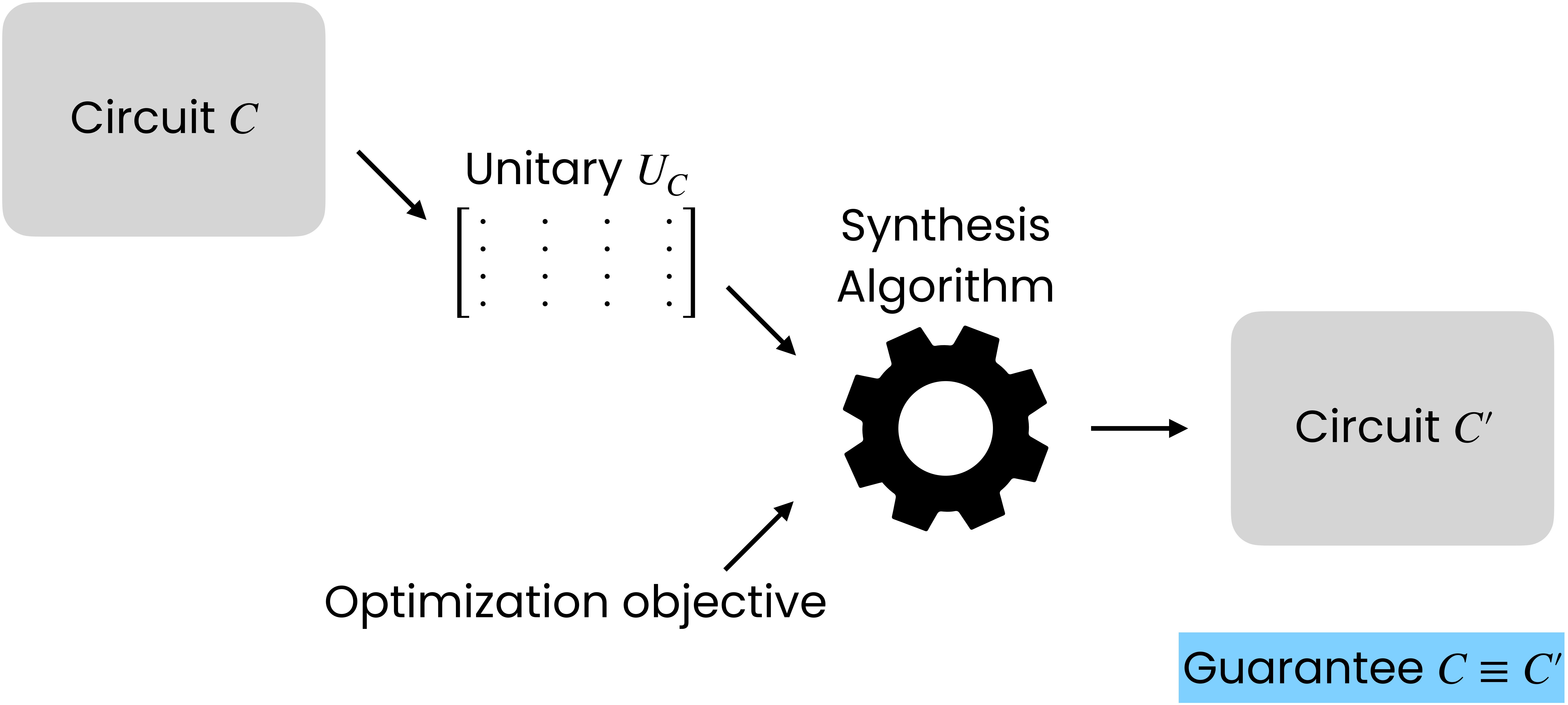
Circuit Resynthesis



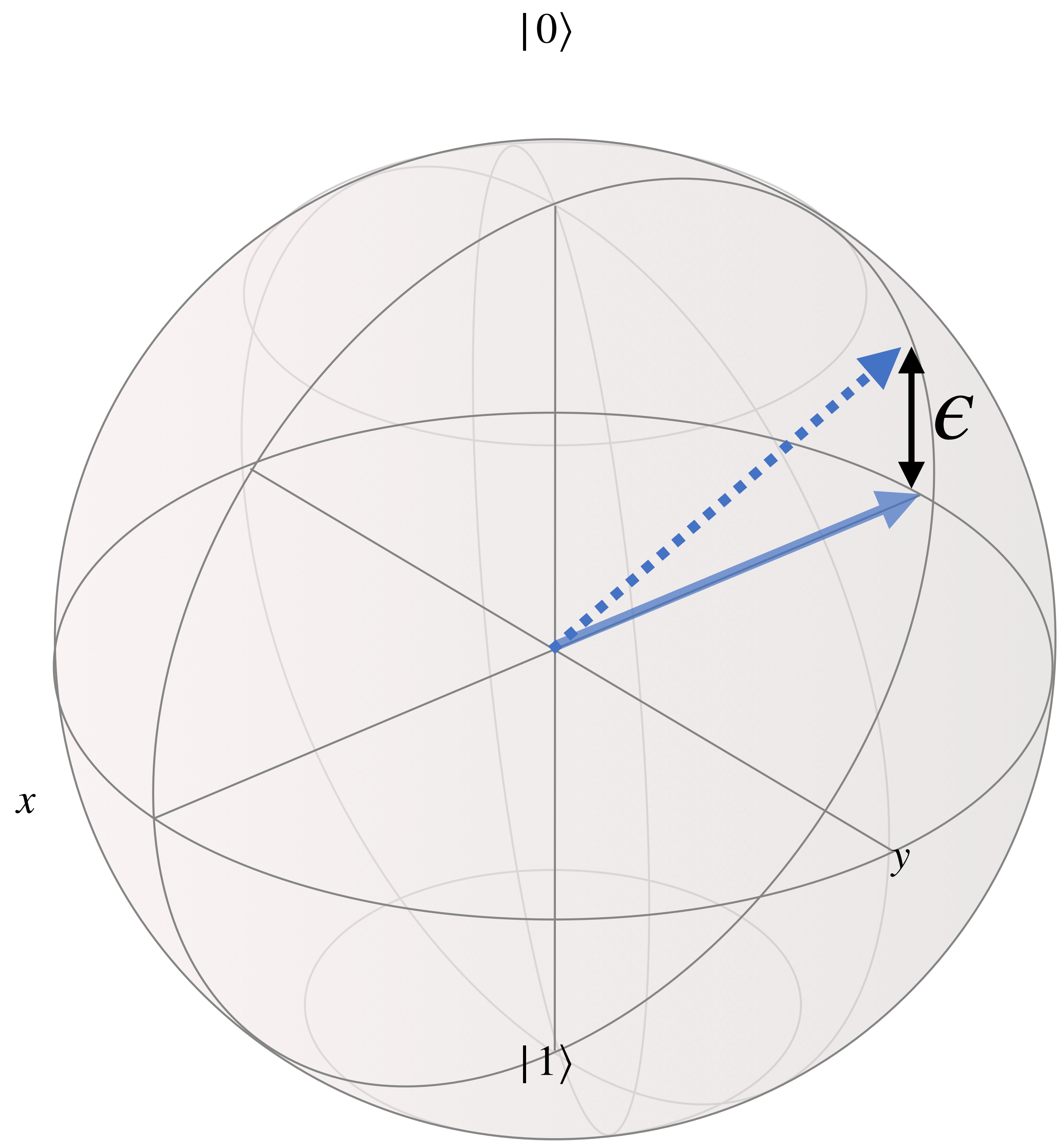
Circuit Resynthesis



Unitary Synthesis



Approximate Circuit Equivalence



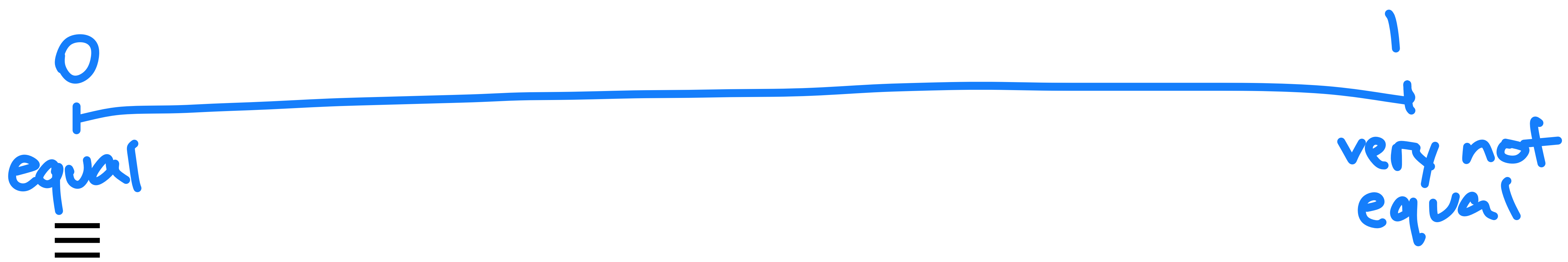
$$\Delta(U_C, U_{C'}) \leq \epsilon \iff C \equiv_\epsilon C'$$

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

Ideal

Circuit C + No noise $= U_C$

Reality

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

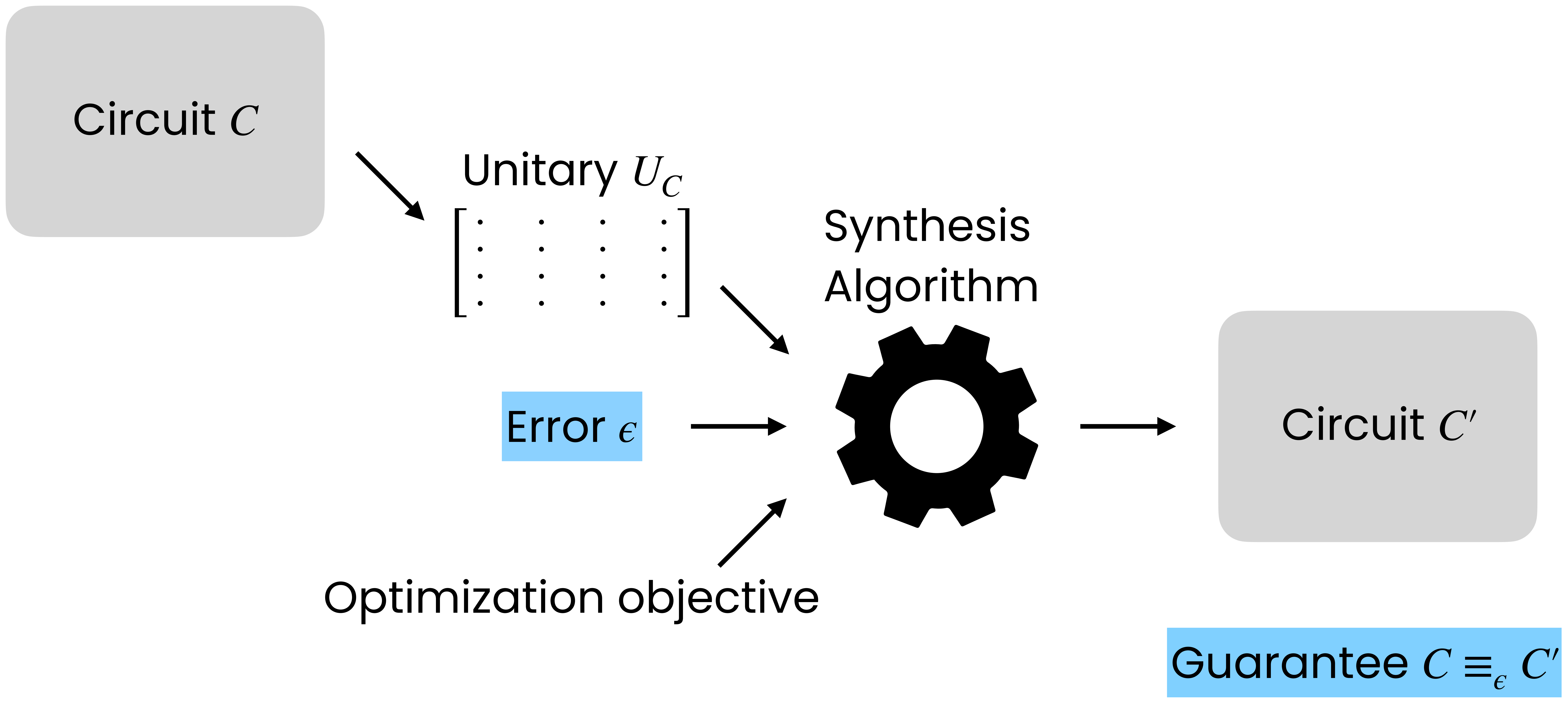
Circuit $C' \equiv_\epsilon C$

k fewer gates than C

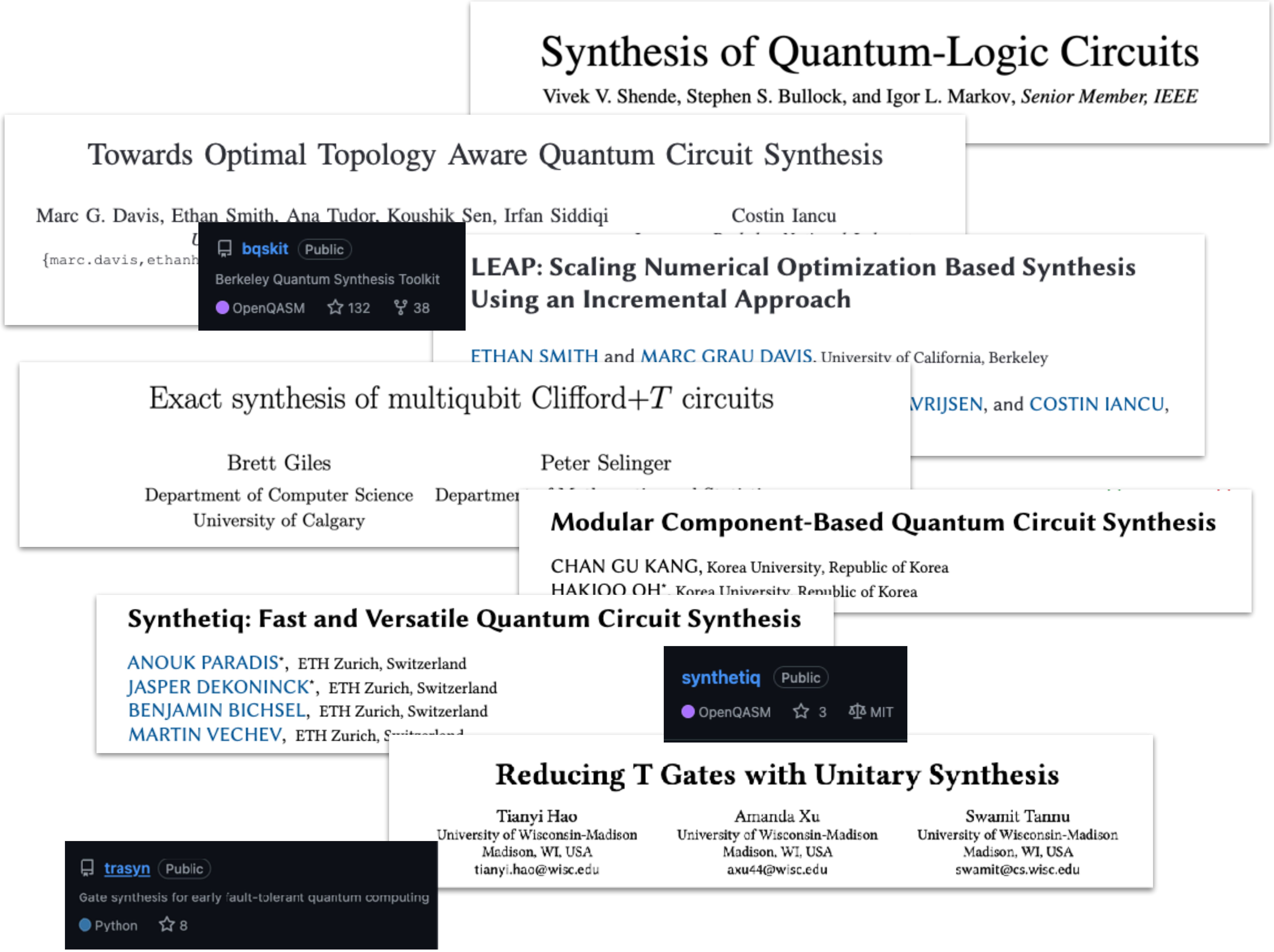
The hope: $U_{C'_{noisy}}$ closer to U_C than $U_{C_{noisy}}$

Noise from k gates $> \epsilon$

Unitary Synthesis



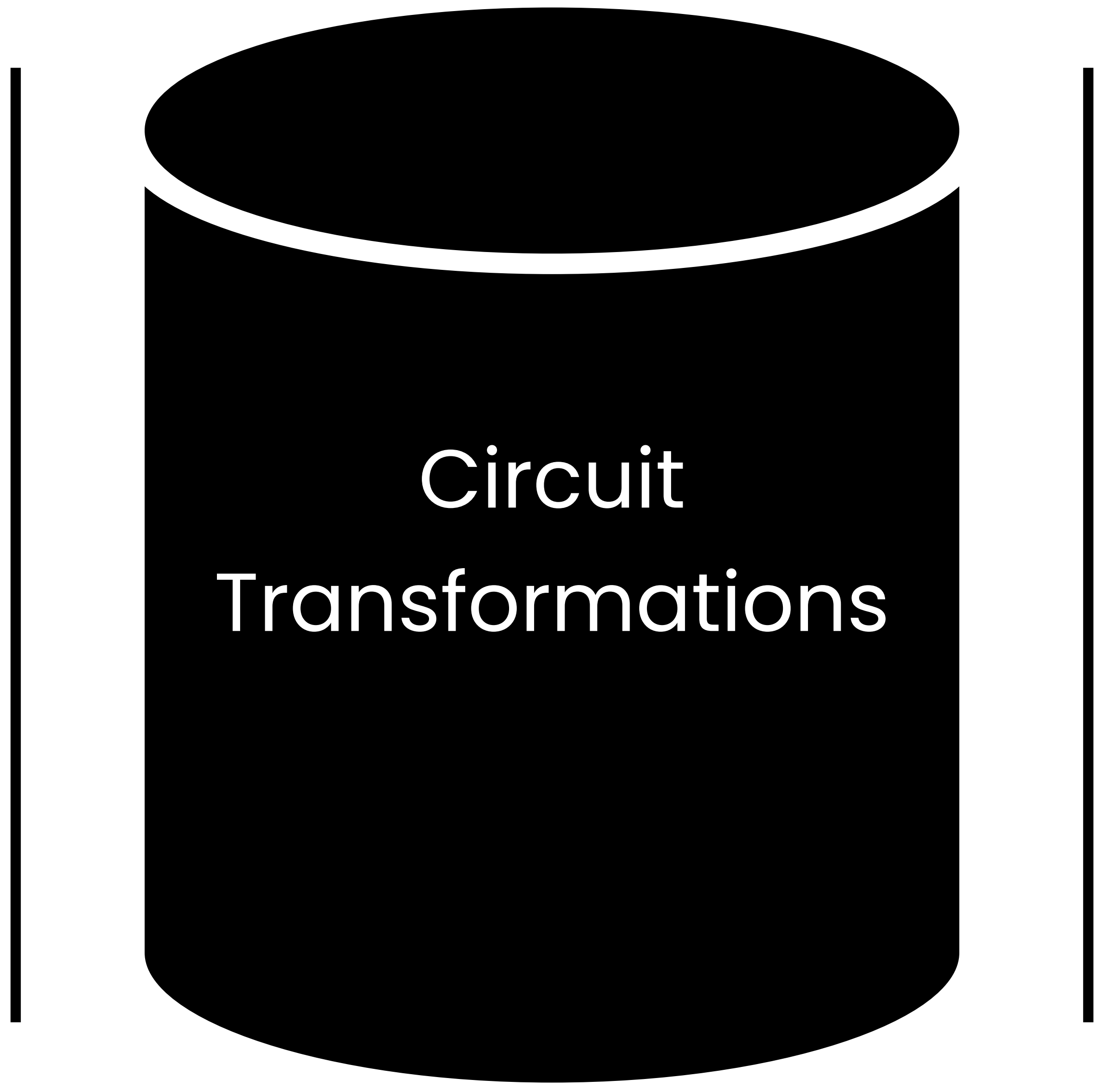
Rich Research Area



Scheduling Transformations

Scheduling is hard

In what order to apply???

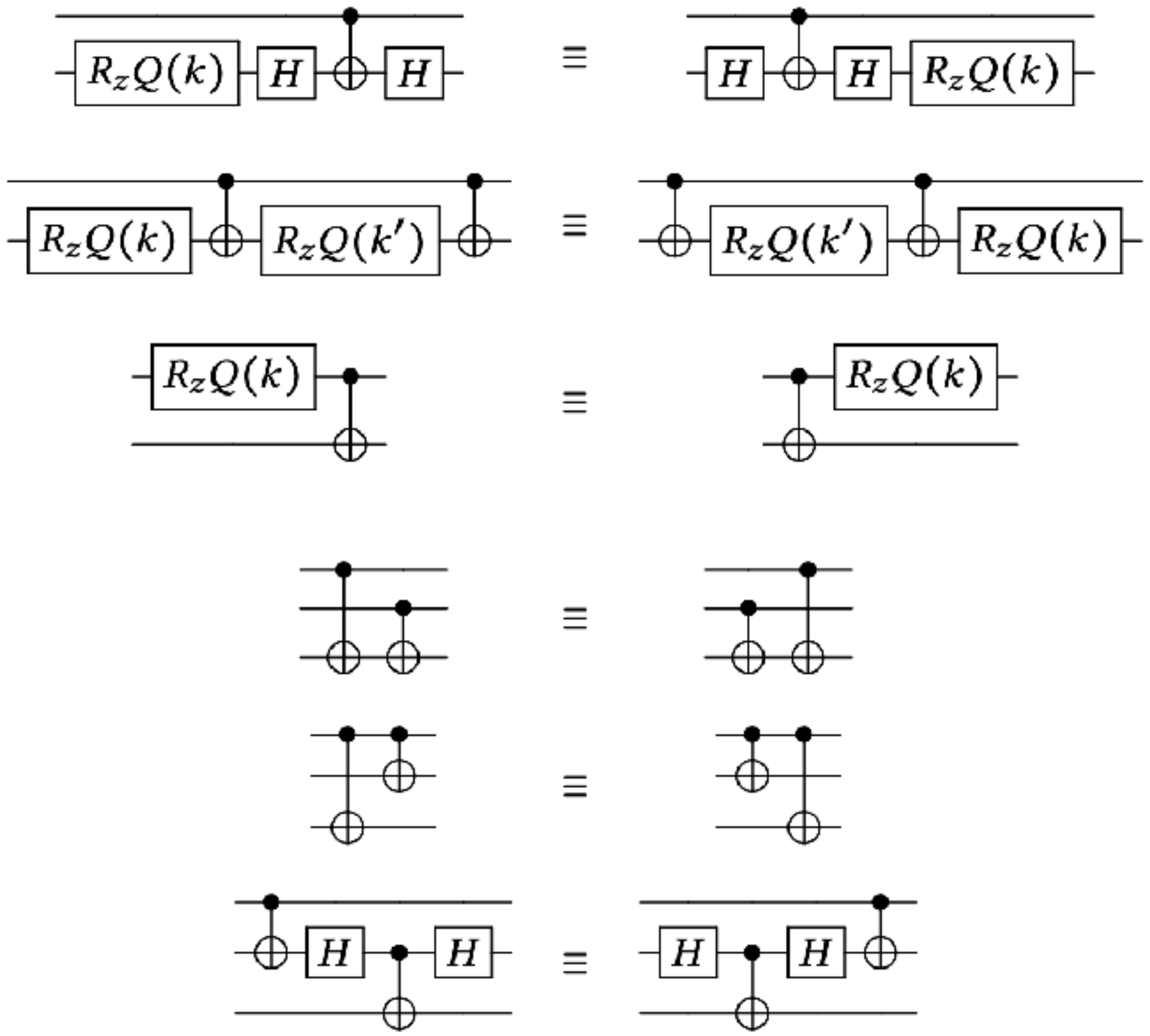


= big

"phase-ordering problem"

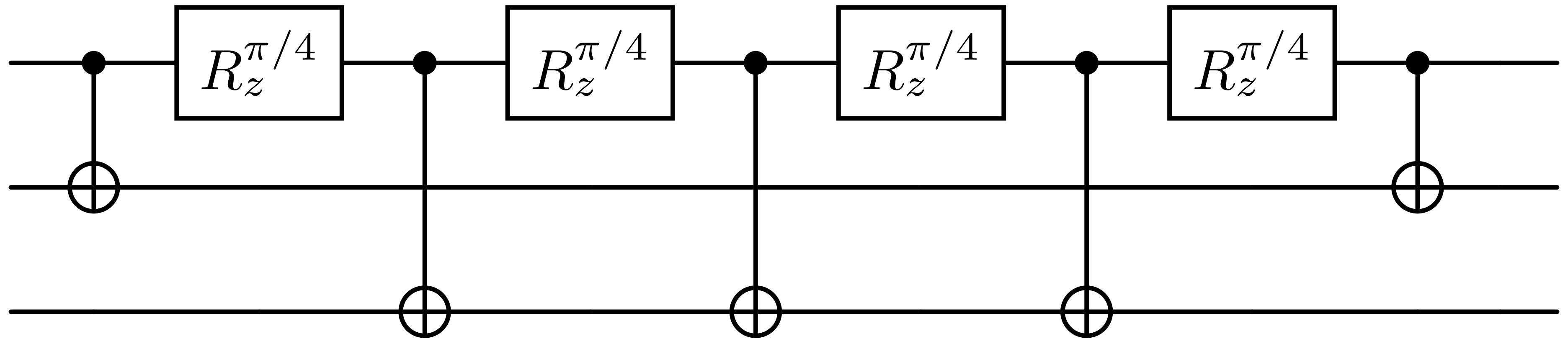
Two Disparate Techniques

Rewrite Rules

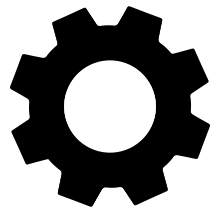


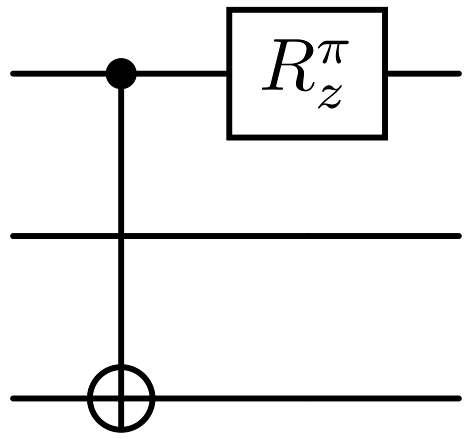
• • •

Circuit Resynthesis



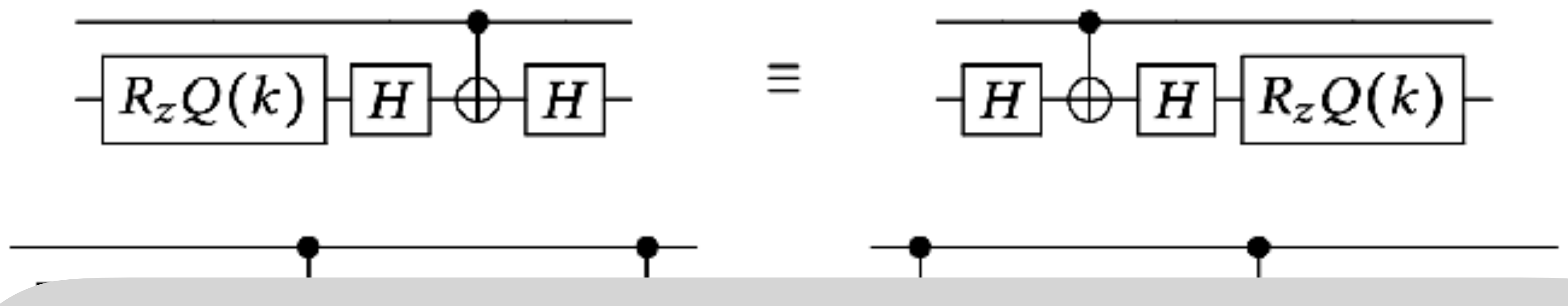
specification

synthesis 



Two Disparate Techniques

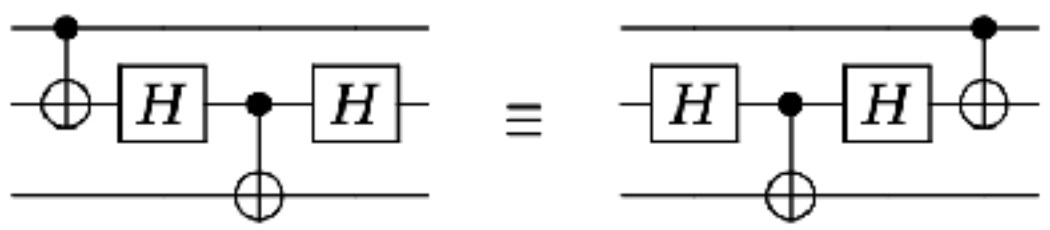
Rewrite Rules



Fast

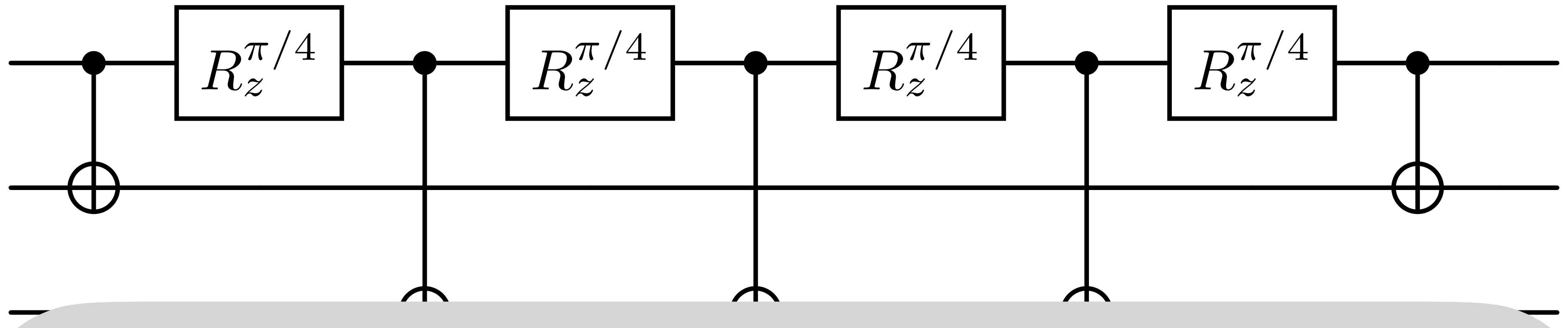
Limited by # gates

Exact



. . .

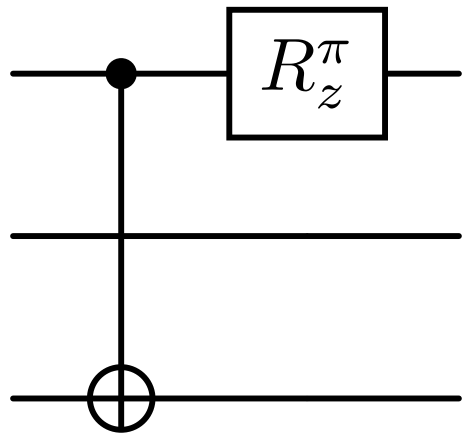
Circuit Resynthesis



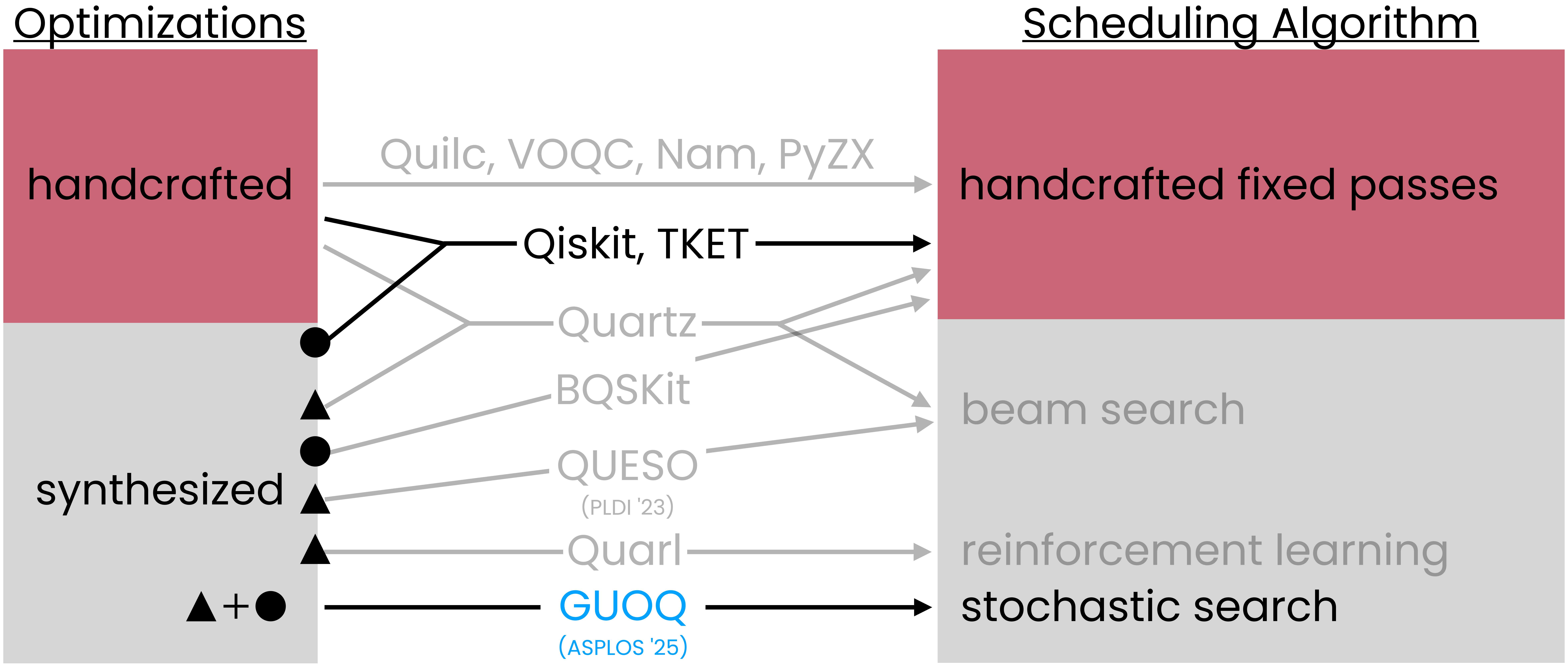
Slow

Limited by # qubits

Approximate

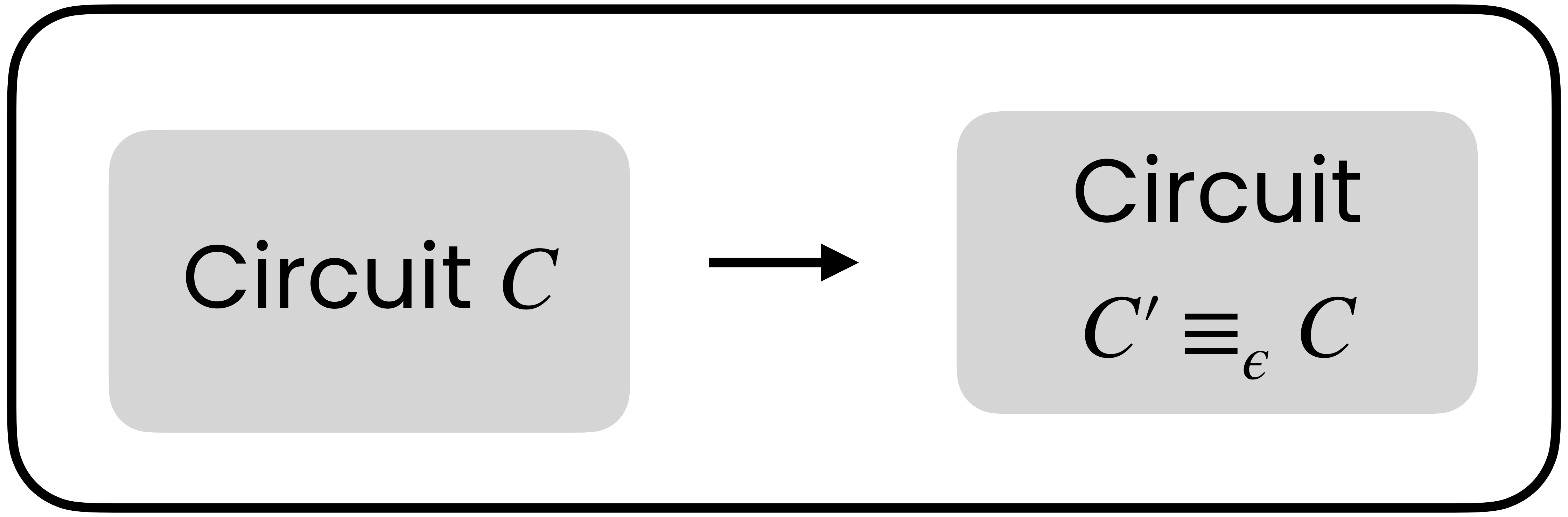


Revisiting The Landscape

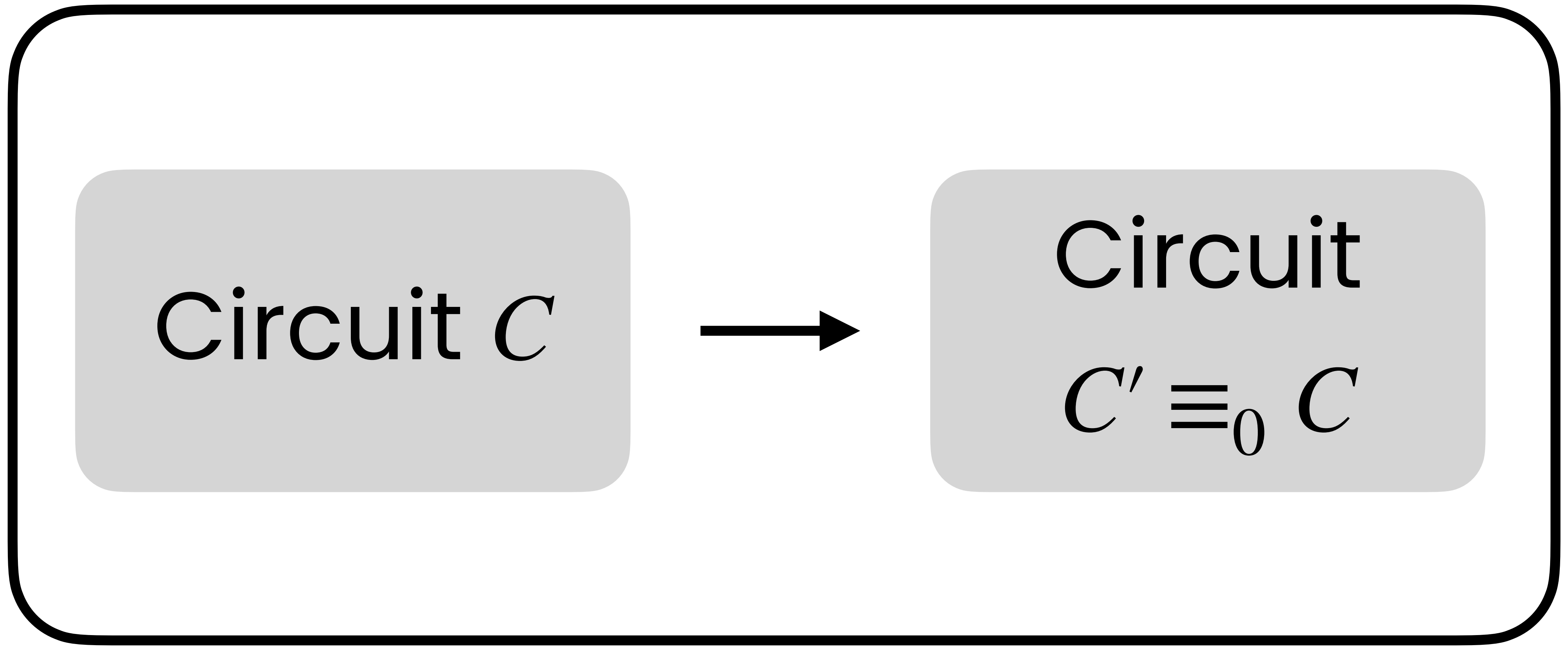


▲ ahead-of-time ● compile-time

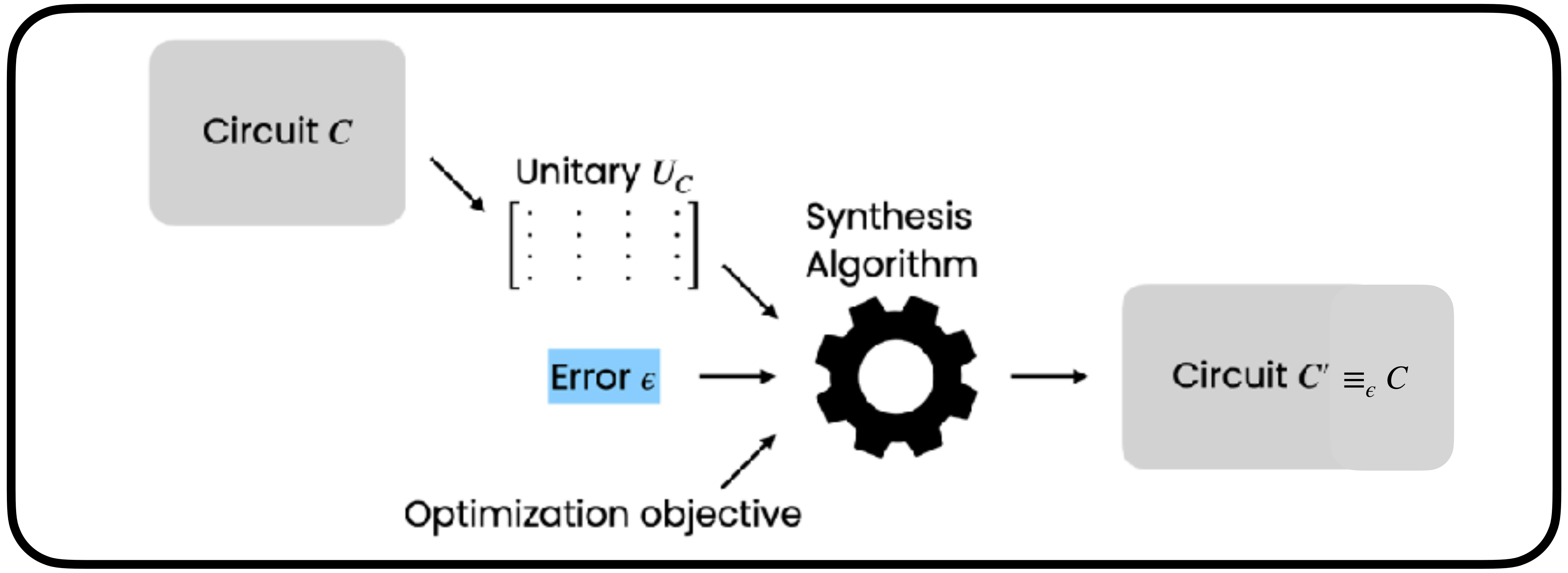
Unifying Framework



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

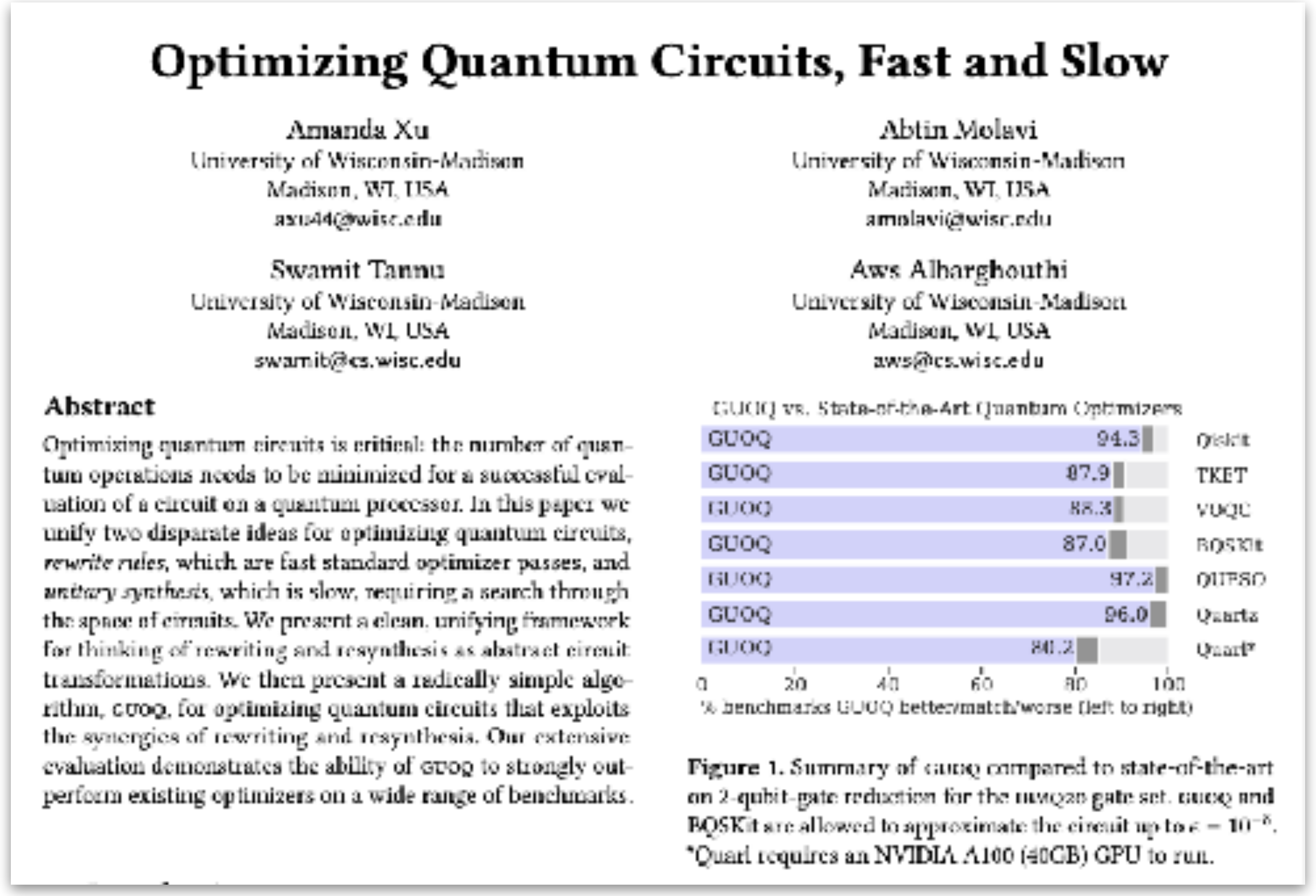


rewrite rule



resynthesis

The GUOQ Algorithm



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

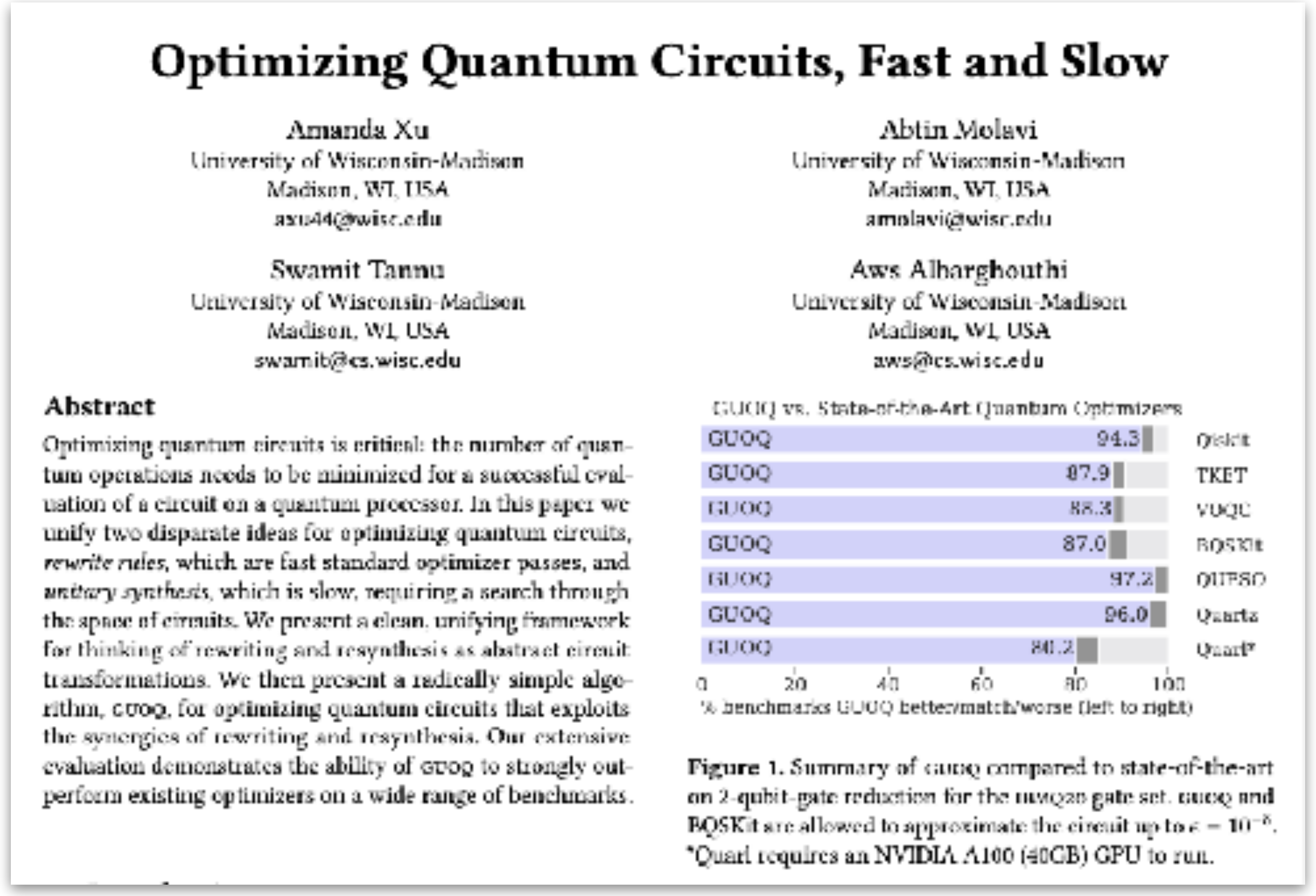
Optimizing quantum circuits is critical: the number of quantum operations needs to be minimized for a successful evaluation of a circuit on a quantum processor. In this paper we unify two disparate ideas for optimizing quantum circuits, *rewrite rules*, which are fast standard optimizer passes, and *unitary 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 '25

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.

The GUOQ Algorithm



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

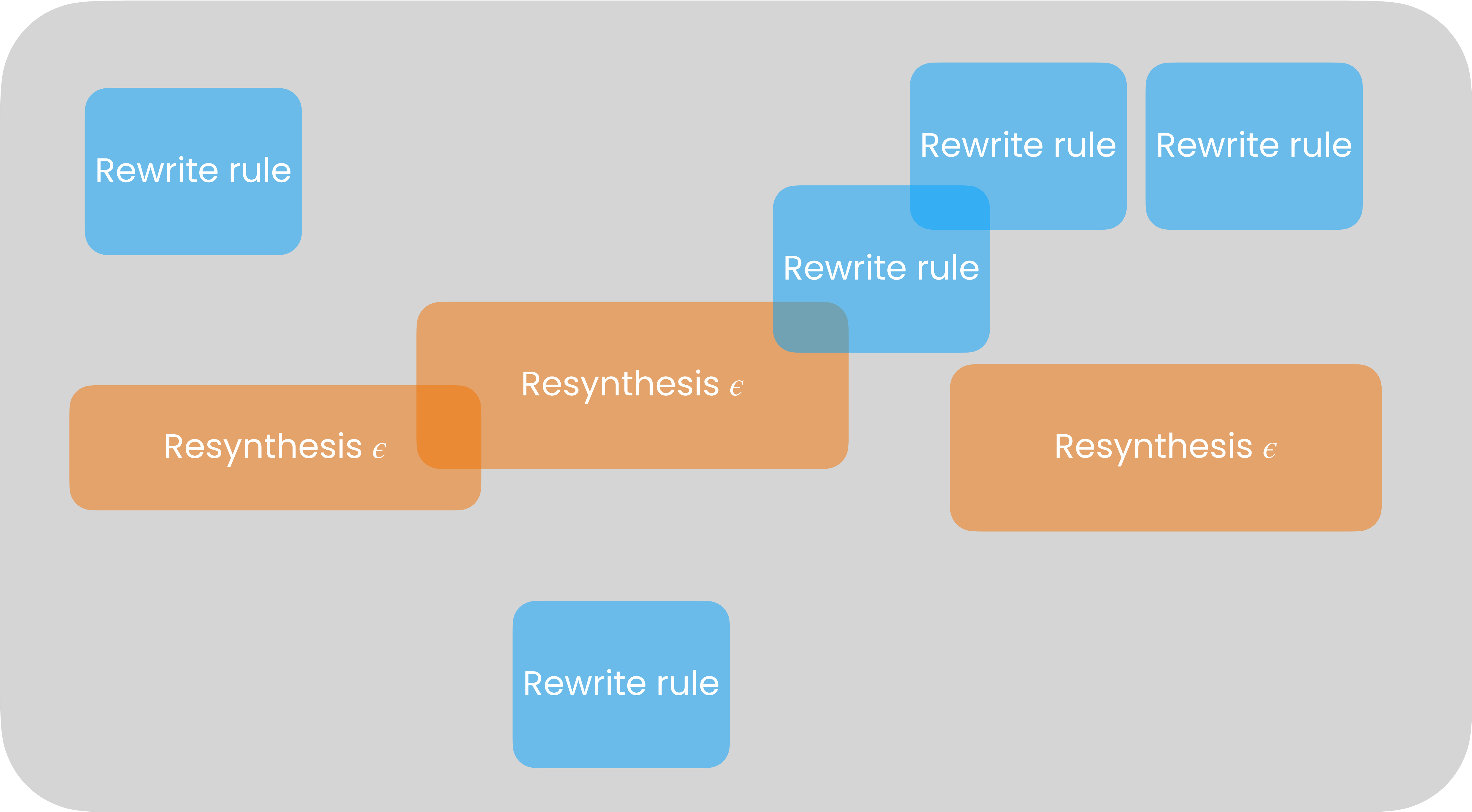
Abstract

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.

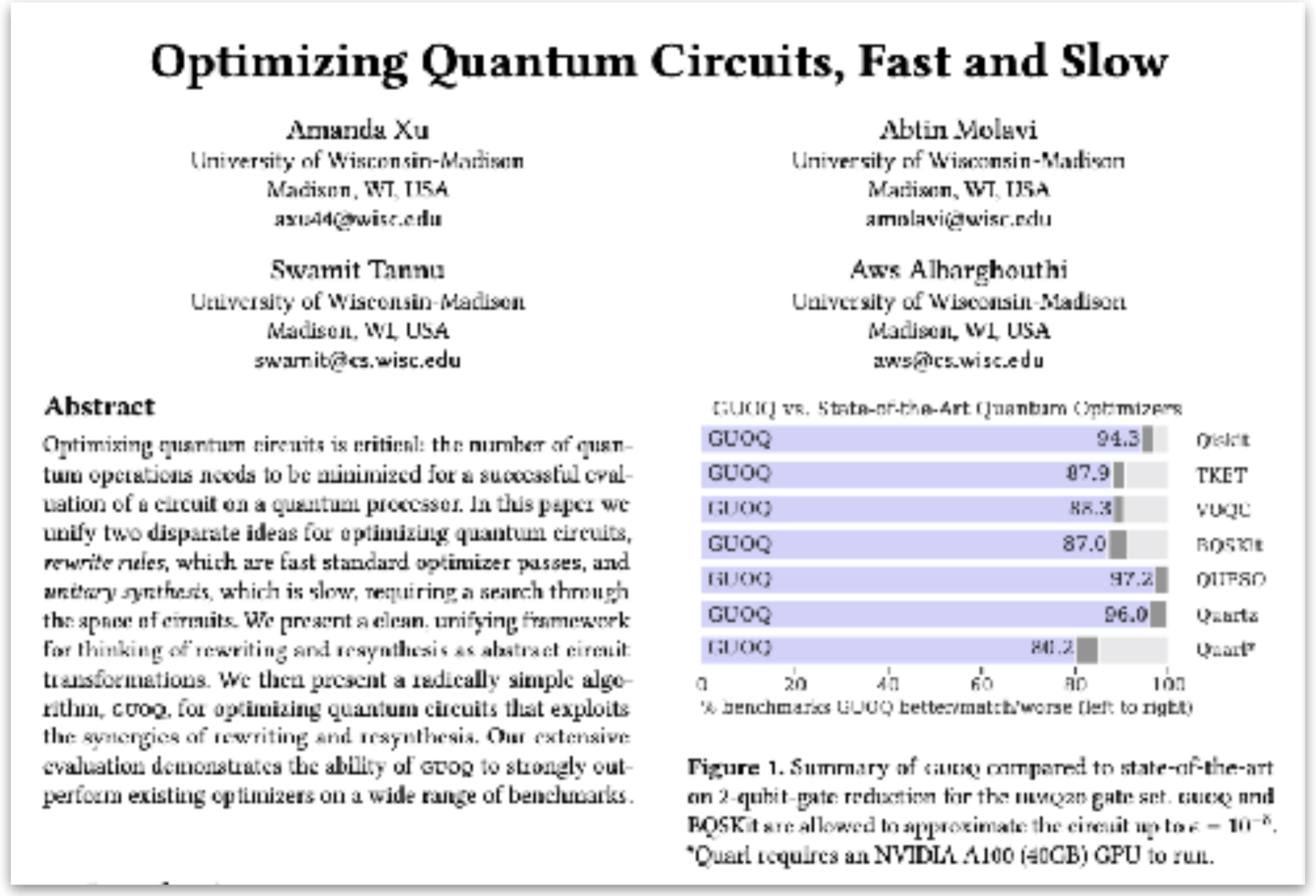
@ ASPLOS '25

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.



The GUOQ Algorithm

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.



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.

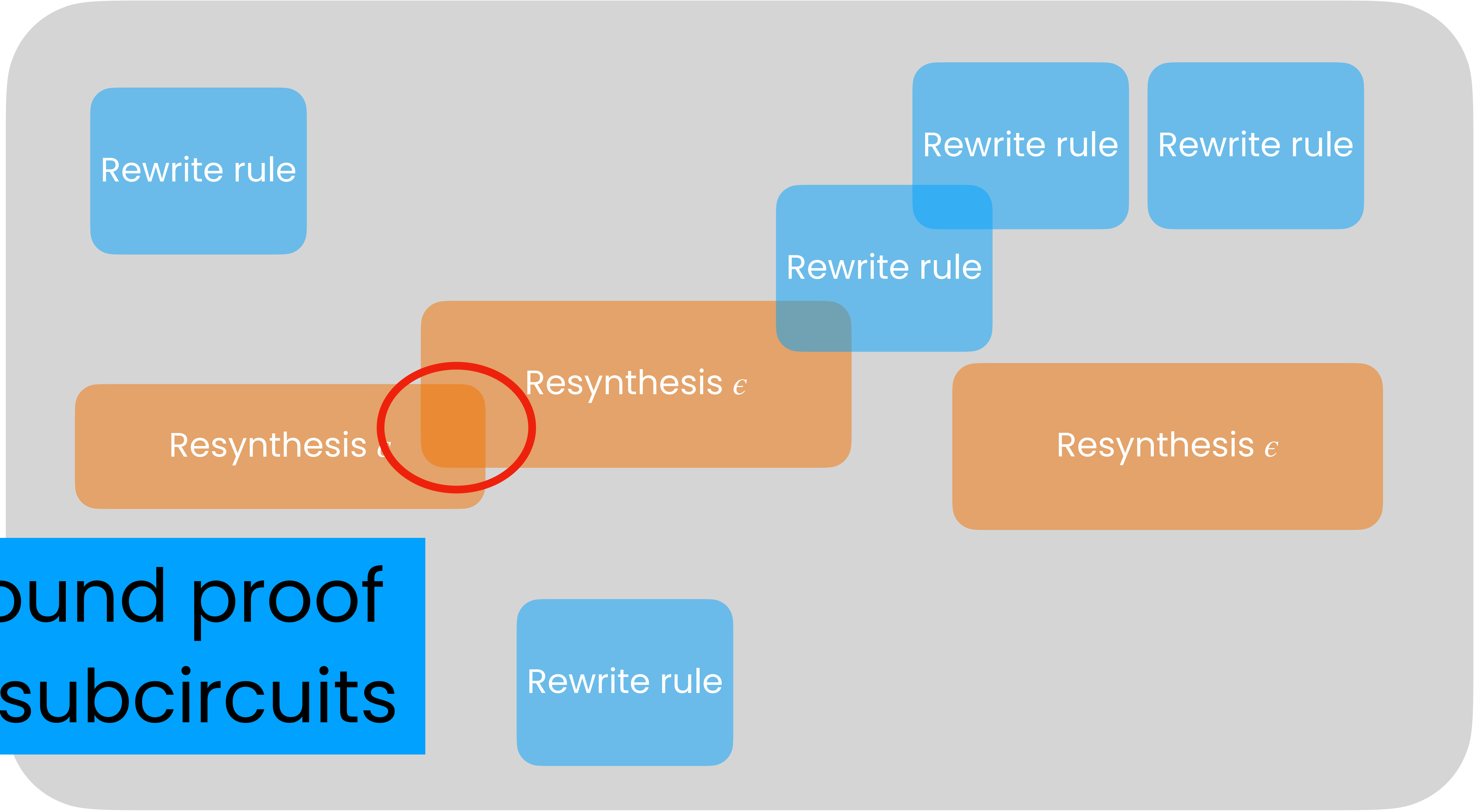
GUOQ vs. State-of-the-Art Quantum Optimizers

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

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

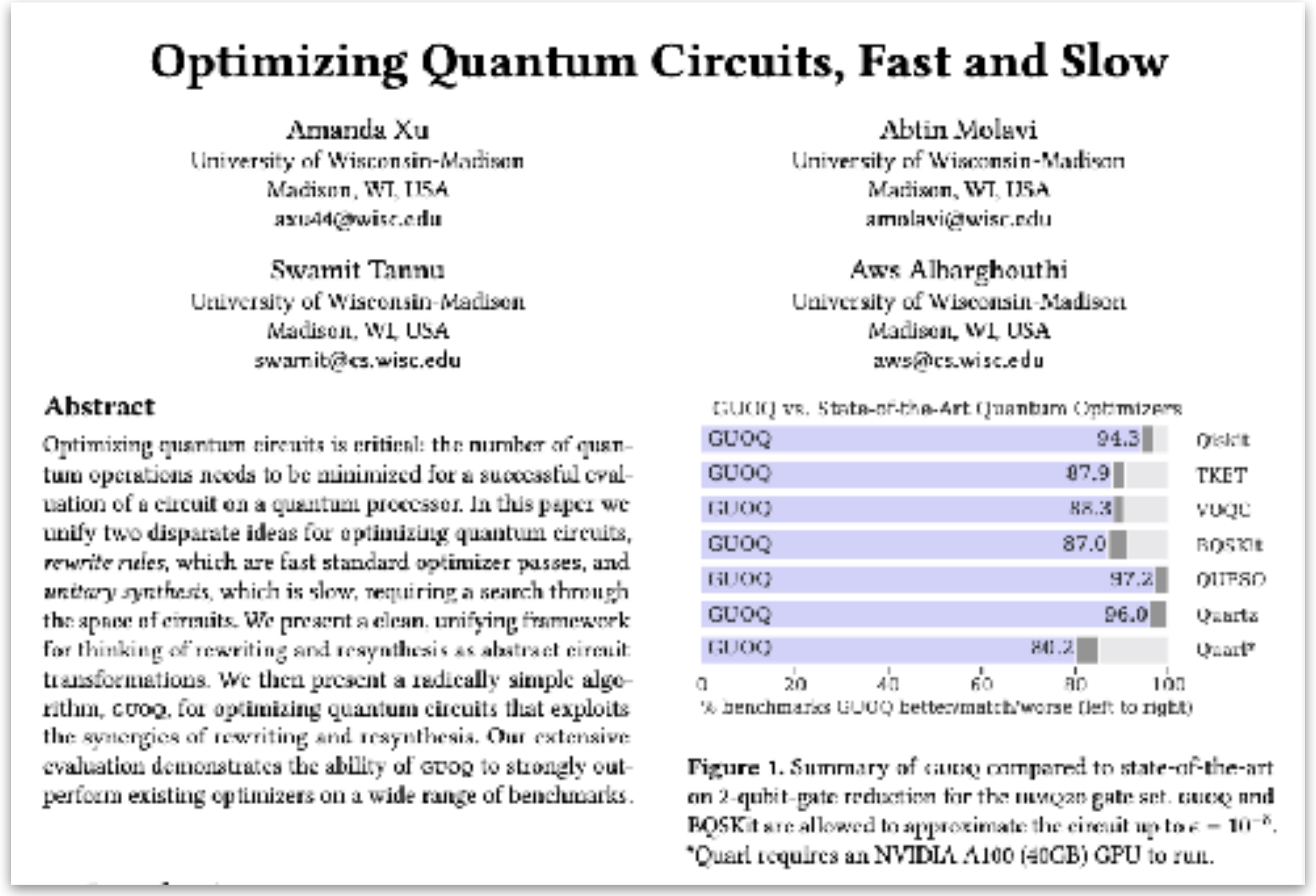
Figure 1. Summary of GUOQ compared to state-of-the-art on 2-qubit-gate reduction for the IBMQ20 gate set. GUOQ and BQSKit are allowed to approximate the circuit up to $\epsilon = 10^{-8}$. *Quarl requires an NVIDIA A100 (40GB) GPU to run.

@ ASPLOS '25



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

The GUOQ Algorithm


Optimizing Quantum Circuits, Fast and Slow

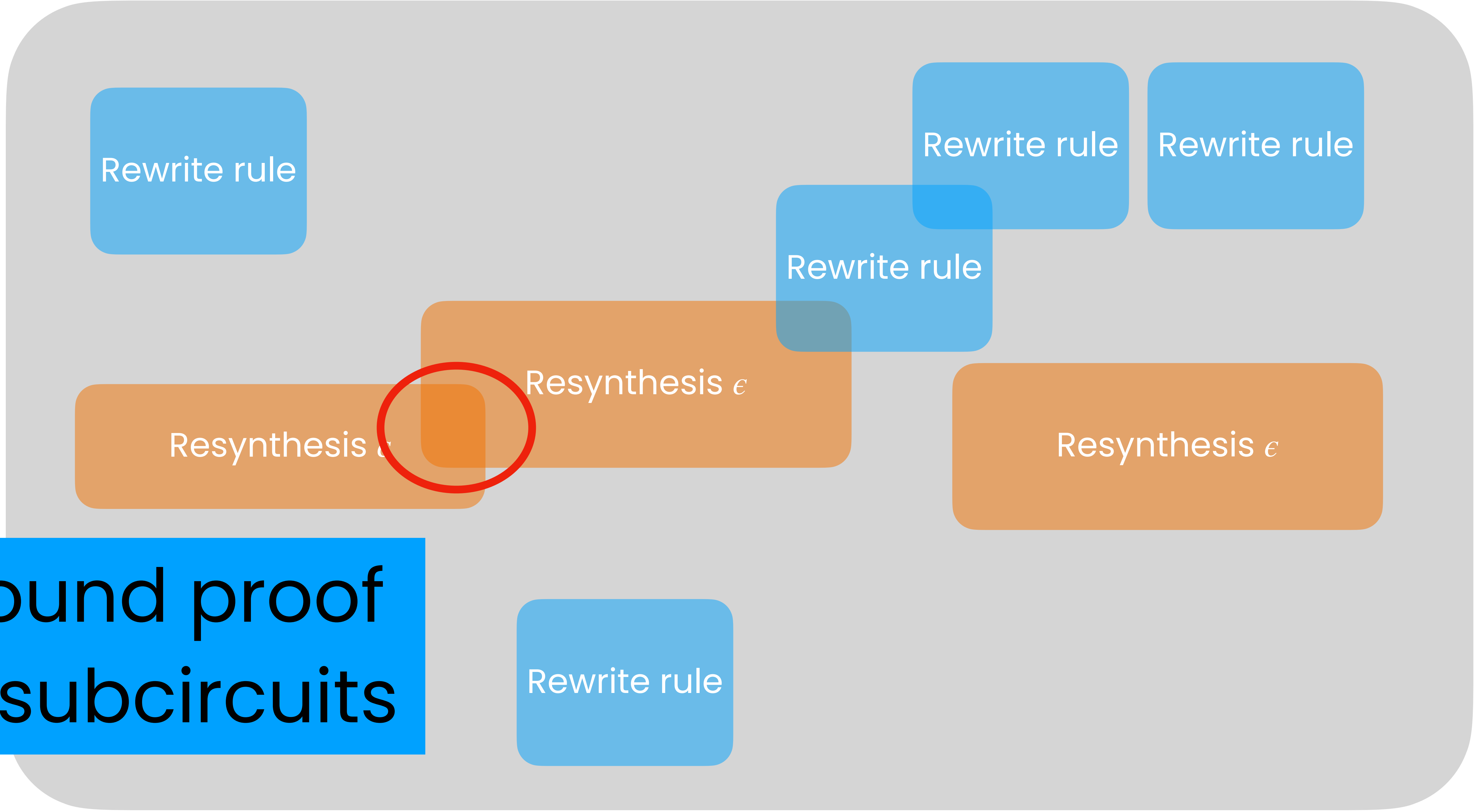
Amanda Xu, University of Wisconsin-Madison, Madison, WI, USA, axu44@wisc.edu
Abtin Molavi, University of Wisconsin-Madison, Madison, WI, USA, amolavi@wisc.edu
Swamit Tannu, University of Wisconsin-Madison, Madison, WI, USA, swamit@cs.wisc.edu
Aws Albarghouthi, University of Wisconsin-Madison, Madison, 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 '25

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.



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

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

We extended to handle arbitrary subcircuits

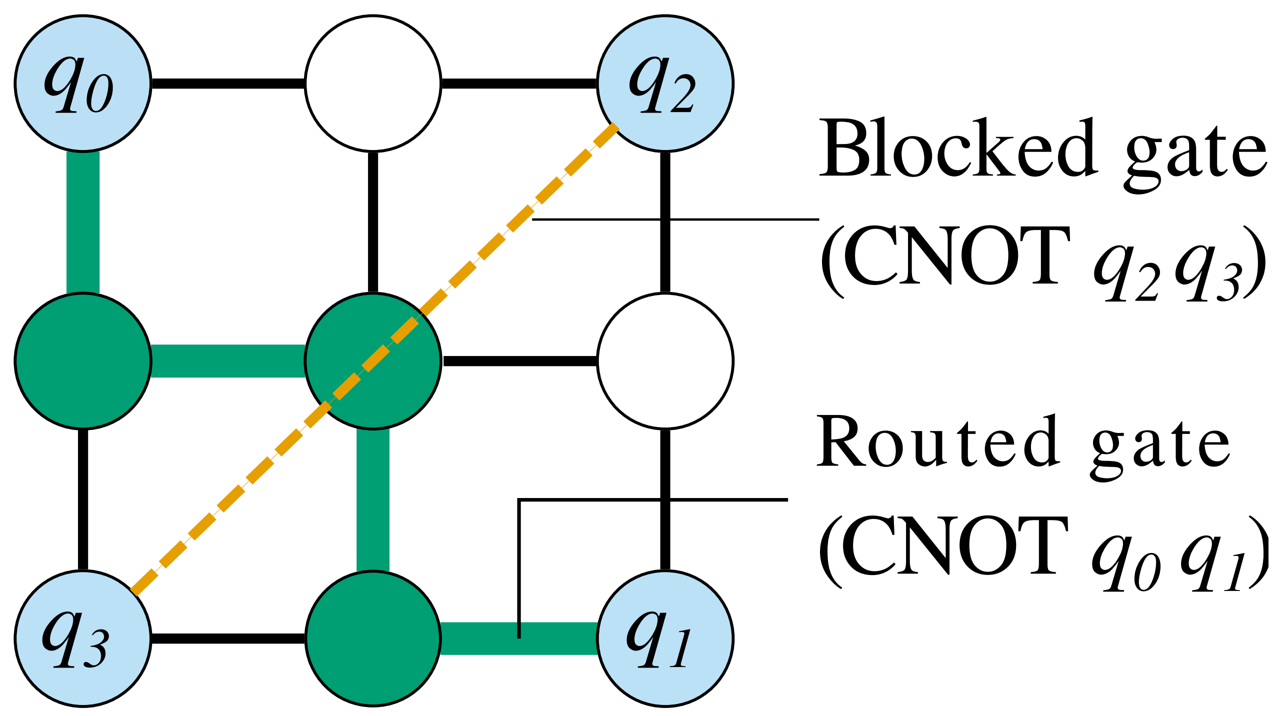
Optimizing for FTQC

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

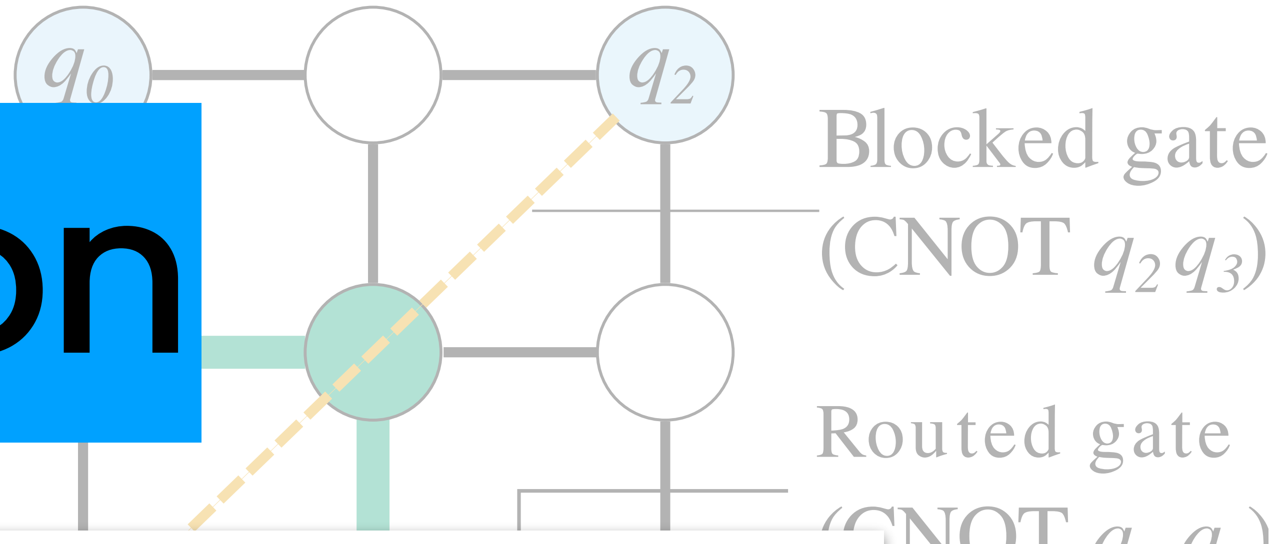
QUESO

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