

# **(IQUHACK 2026 | SUPERQUANTUM)**

## **67 QUBITS**

ADAM BANKS, RUBEN CARPENTER, RICHARD CHEN, GEORGE HUO, AND JUSTIN SATO

### 1. INTRODUCTION

In principle, quantum computing is highly expressive: any unitary evolution can be realized, allowing a wide range of problems to be modeled. In practice, however, constructing circuits can be fragile: small changes in coefficients or phases can make exact implementation impossible or extremely costly.

During iQuHack 2026, we explored this tradeoff between expressiveness and realizability in the concrete setting of Clifford+ $T$  circuits (using only the  $\{H, T, T^\dagger, \text{CNOT}\}$ ). In particular, we consider 11 special cases of unitary matrices, and for each construct a Clifford+ $T$  circuit either representing  $U$ , or an approximation  $\tilde{U}$  that balances two complementary metrics:

- **Operator norm distance:**

$$d(U, \tilde{U}) = \min_{\phi \in [0, 2\pi)} \|U - e^{i\phi} \tilde{U}\|_{\text{op}},$$

This quantifies the expressiveness of the circuit.

- **T-count:** The total number of  $T/T^\dagger$  gates in the circuit. This quantifies the instability and error-prone-ness of the circuit.

Many important operators, such as two-dimensional quantum Fourier transforms (challenge 11) and exponentials of certain Hamiltonians (challenge 5), can be decomposed exactly in this gate set. In these cases (which we color-code in green), finding the decomposition often felt like solving a puzzle, requiring intuition, pattern recognition, and some creative insights.

However, because Clifford+ $T$  contains only countably many elements, it cannot represent “almost all” unitary operators, even though it is dense in  $U(2)$ . In the cases where only an approximation can be found (which we color-code in blue), we applied systematic approximation frameworks, relying on classical computation brute force methods, an algorithm developed by Ross and Selinger [SR18], and several ad-hoc techniques we discuss throughout our report.

Across both exact and approximate constructions, we were struck by how few  $T$  gates were sufficient to achieve high accuracy, and are excited about a future with algorithms and infrastructure to push the boundaries of this trade-off between expressiveness and realizability.

We have made our code available on a public GitHub [repository].

### 2. CHALLENGES

#### 1. Controlled-Y Gate.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{bmatrix}$$

*Solution.* Observe that we can rewrite

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -i \end{bmatrix} \\ = (S \otimes I) \text{CNOT}(S^\dagger \otimes I).$$

This gives the quantum circuit:

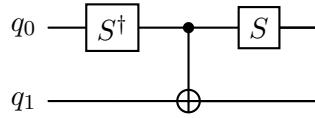


FIGURE 1. Quantum Circuit for Challenge 1

## 2. Controlled- $R_y(\pi/7)$ .

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sin(\pi/14) & -\cos(\pi/14) \\ 0 & 0 & \cos(\pi/14) & \sin(\pi/14) \end{bmatrix}$$

*Solution.* In terms of the two “conjugate” classes of rotations

$$R_y(\theta) := \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}, \quad R_z(\theta) := \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$

we aim to implement a Controlled gate  $\text{C-}R_y(\pi/7)$  rotation. We first notice that, if a Controlled  $\text{C-}R_z(\theta)$  rotation were available, then we could also obtain a  $[C - R_y(\theta)]$  rotation by conjugation:

$$\begin{aligned} & \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sin(\theta/2) & -\cos(\theta/2) \\ 0 & 0 & -\cos(\theta/2) & \sin(\theta/2) \end{bmatrix} \\ &= \frac{1}{2} \begin{bmatrix} e^{i\pi/4} & e^{-i\pi/4} & 0 & 0 \\ e^{-i\pi/4} & e^{i\pi/4} & 0 & 0 \\ 0 & 0 & e^{i\pi/4} & e^{-i\pi/4} \\ 0 & 0 & e^{-i\pi/4} & e^{i\pi/4} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix} \frac{1}{2} \begin{bmatrix} e^{-i\pi/4} & e^{i\pi/4} & 0 & 0 \\ e^{i\pi/4} & e^{-i\pi/4} & 0 & 0 \\ 0 & 0 & e^{i\pi/4} & e^{-i\pi/4} \\ 0 & 0 & e^{-i\pi/4} & e^{i\pi/4} \end{bmatrix} \\ &= \left( I \otimes \frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\pi/4} & e^{-i\pi/4} \\ e^{-i\pi/4} & e^{i\pi/4} \end{bmatrix} \right) \text{Controlled-}R_z(\theta) \left( I \otimes \frac{1}{\sqrt{2}} \begin{bmatrix} e^{-i\pi/4} & e^{i\pi/4} \\ e^{i\pi/4} & e^{-i\pi/4} \end{bmatrix} \right) \\ &= (I \otimes HSH) \cdot \text{C-}R_z(\theta) \cdot (I \otimes HS^\dagger H). \end{aligned}$$

Moreover, if we could implement  $R_z(\theta)$ , then we could also implement C- $R_y(\theta)$  as

$$\begin{aligned}
 & (I \otimes R_z(\theta/2)) \text{CNOT}(I \otimes R_z(-\theta/2)) \text{CNOT} \\
 &= \begin{bmatrix} e^{-i\frac{\theta}{4}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{4}} & 0 & 0 \\ 0 & 0 & e^{-i\frac{\theta}{4}} & 0 \\ 0 & 0 & 0 & e^{i\frac{\theta}{4}} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} e^{i\frac{\theta}{4}} & 0 & 0 & 0 \\ 0 & e^{-i\frac{\theta}{4}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{4}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{4}} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix}.
 \end{aligned}$$

This leads to the quantum circuit in Figure 2

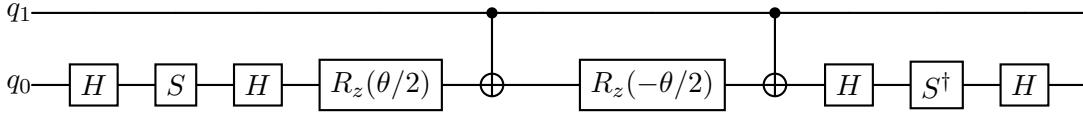


FIGURE 2. Quantum Circuit for Challenge 2

However, this relies on an exact implementation of  $R_z(\pi/7)$  with Clifford+T gates, which unfortunately is not possible: by a result of Kliuchnikov, Maslov, and Mosca [KMM13], a matrix is constructible if and only if its coefficients lie in the ring  $\mathbb{Z}[1/\sqrt{2}, i]$ , which is not the case for  $R_z(\pi/7)$ . To remedy this, we use the *gridsynth* package of Ross and Selinger [RS16, SR18] to approximate it. Requiring 10 decimals of accuracy, we achieved an operator norm distance of  $1.44 \cdot 10^{-14}$  with 104 T-gates. Increasing to 20 decimals, we achieved an operator norm distance of  $5.47 \cdot 10^{-14}$  with 380 T-gates.

### 3. Exponential of a Pauli String.

$$e^{i\frac{\pi}{7}Z \otimes Z}$$

*Solution.* From the same logic in Challenge 2, the  $\pi/7$  makes us suspect that we might not be able to find an exact solution. To that end, we will try to write this matrix in terms of rotations, which we can approximate efficiently and arbitrarily well using the *gridsynth* package.

We begin by computing

$$e^{i\frac{\pi}{7}Z \otimes Z} = e^{i\frac{\pi}{7}} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} e^{i\frac{\pi}{7}} & 0 & 0 & 0 \\ 0 & e^{-i\frac{\pi}{7}} & 0 & 0 \\ 0 & 0 & e^{-i\frac{\pi}{7}} & 0 \\ 0 & 0 & 0 & e^{i\frac{\pi}{7}} \end{pmatrix}.$$

This means the matrix acts by multiplication by  $e^{i\pi/7}$  when the parity of the bits is even, and by multiplication by  $e^{-i\pi/7}$  when it is odd. Therefore we can write it more naturally as

$$e^{i\frac{\pi}{7}Z \otimes Z} = \text{CNOT} \left( e^{i\frac{\pi}{7}} I \otimes \begin{pmatrix} e^{-i\frac{2\pi}{7}} & 0 \\ 0 & e^{i\frac{2\pi}{7}} \end{pmatrix} \right) \text{CNOT} = e^{i\frac{\pi}{7}} \text{CNOT}(I \otimes R_z(-2\pi/7)) \text{CNOT}.$$

This confirms our suspicion that an exact solution is impossible (as  $\sin(-2\pi/7) \notin \mathbb{Z}[1/\sqrt{2}, i]$ ). Up to a global phase factor, we approximate it using the *gridsynth* package [RS16, SR18] up to 20 digits of precision, achieving an operator norm distance of  $2.79 \cdot 10^{-14}$  with a  $T$ -count of 200.

#### 4. Exponential of a Hamiltonian.

$$e^{i\frac{\pi}{7}H_1},$$

where  $H_1 = X \otimes X + Y \otimes Y$ .

*Solution.* We begin by noting that  $X \otimes X$  and  $Y \otimes Y$  commute

$$\begin{aligned}(XX)(YY) &= (XY) \otimes (XY) = (iZ) \otimes (iZ) = -ZZ \\ (YY)(XX) &= (YX) \otimes (YX) = (-iZ) \otimes (-iZ) = -ZZ\end{aligned}$$

Thus,  $e^{i\frac{\pi}{7}H_1} = (e^{i\frac{\pi}{7}X \otimes X})(e^{i\frac{\pi}{7}Y \otimes Y})$ , and we can consider each term separately:

- For  $e^{i\frac{\pi}{7}X \otimes X}$ , we use the identity  $HZH = X$  to conjugate

$$e^{i\frac{\pi}{7}X \otimes X} = (H \otimes H)e^{i\frac{\pi}{7}Z \otimes Z}(H \otimes H)$$

- For  $e^{i\frac{\pi}{7}Y \otimes Y}$ , we use the identity  $XS^\dagger = S(HZH)S^\dagger = Y$  to conjugate

$$e^{i\frac{\pi}{7}Y \otimes Y} = (SH \otimes SH)e^{i\frac{\pi}{7}Z \otimes Z}(HS^\dagger \otimes HS^\dagger)$$

Putting everything together

$$e^{i\frac{\pi}{7}H_1} = (H \otimes H)e^{i\frac{\pi}{7}Z \otimes Z}(H \otimes H)(SH \otimes SH)e^{i\frac{\pi}{7}Z \otimes Z}(HS^\dagger \otimes HS^\dagger).$$

In Challenge 3 we approximated as

$$e^{i\frac{\pi}{7}Z \otimes Z} = e^{i\frac{\pi}{7}} \text{CNOT}(I \otimes R_z(-2\pi/7)) \text{CNOT}.$$

Using that result to 20 decimals of accuracy, we achieve operative norm distance of  $5.07 \times 10^{-14}$  with a  $T$ -count of 400.

#### 5. Exponential of a Hamiltonian.

$$e^{i\frac{\pi}{4}H_2},$$

where  $H_2 = X \otimes X + Y \otimes Y + Z \otimes Z$ .

*Solution.* We begin by explicitly computing

$$\begin{aligned}H_2 &= X \otimes X + Y \otimes Y + Z \otimes Z \\ &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.\end{aligned}$$

We recognize this as  $2 \cdot \text{SWAP} - I$ , which implies that  $e^{i\frac{\pi}{4}H_2} = e^{-\frac{i\pi}{4}} e^{i\frac{\pi}{2}} \text{SWAP}$ . Since  $\text{SWAP}^2 = I$ ,  
 $e^{i\frac{\pi}{2}} \text{SWAP} = \cos(\pi/2) + i \sin(\pi/2) \text{SWAP} = i \text{SWAP}$ ,

we conclude that  $e^{i\frac{\pi}{4}H_3} = e^{i\pi/4} \text{SWAP}$ . Up to a global factor, this is just a SWAP gate, which can be implemented with three CNOT gates.

**6. Transverse Field Ising Model.** Let  $H_3 = XX + ZI + IZ$ :

$$e^{i\frac{\pi}{7}H_3}$$

*Note.* This is a time evolution under a 2-qubit transverse field Ising model.

*Solution.* We note that  $ZI + IZ$  does not commute with  $XX$ , thus the Hamiltonian cannot be decomposed exactly. Recall the Trotter product formula, which states that for  $A, B \in \text{Mat}(\mathbb{C})$

$$e^{A+B} = \lim_{k \rightarrow \infty} \underbrace{(e^{A/k} e^{B/k}) \cdots (e^{A/k} e^{B/k})}_{k \text{ times}}.$$

Inspired by it, for sufficiently large  $k$  we estimate

$$\begin{aligned} e^{i\frac{\theta}{2}H_3} &\approx \underbrace{\left( e^{\frac{\theta X \otimes X}{2k}} e^{\frac{\theta Z \otimes I}{2k}} e^{\frac{\theta I \otimes Z}{2k}} \right)}_{k \text{ times}} \cdots \underbrace{\left( e^{\frac{\theta X \otimes X}{2k}} e^{\frac{\theta Z \otimes I}{2k}} e^{\frac{\theta I \otimes Z}{2k}} \right)}_{k \text{ times}} \\ &= \left( (H \otimes H) e^{\frac{\theta Z \otimes Z}{2k}} (H \otimes H) e^{\frac{\theta Z \otimes I}{2k}} e^{\frac{\theta I \otimes Z}{2k}} \right)^k \\ &= ((H \otimes H) \text{CNOT}(I \otimes R_z(-\theta/k)) \text{CNOT}(H \otimes H)(R_z(-\theta/k) \otimes I)(I \otimes R_z(-\theta/k))^k. \end{aligned}$$

which means

$$e^{i\frac{\pi}{7}H_3} \approx ((H \otimes H) \text{CNOT}(I \otimes R_z(-2\pi/7k)) \text{CNOT}(H \otimes H)(R_z(-2\pi/7k) \otimes I)(I \otimes R_z(-2\pi/7k))^k$$

Naturally, the approximation is more accurate for larger  $k$ . For our purposes, given the constraints of the challenge, we chose  $k = 5$  with 12 decimals of precision for our approximation of  $R_z(-\frac{2\pi}{7})$ , achieving an operator norm distance of  $2.9 \times 10^{-3}$  with 3710 T-gates.

**7. State Preparation.** Design  $U \in \mathbb{C}^{4 \times 4}$  such that

$$\begin{aligned} |00\rangle &\mapsto (0.1061479384 - 0.679641467i) |00\rangle \\ &\quad + (-0.3622775887 - 0.453613136i) |01\rangle \\ &\quad + (0.2614190429 + 0.0445330969i) |10\rangle \\ &\quad + (0.3276449279 - 0.1101628411i) |11\rangle. \end{aligned}$$

*Solution.* Since the target state  $|\psi\rangle$  is normalized, it can be chosen as the first column of a unitary matrix  $U \in \mathbb{C}^{4 \times 4}$ . The remaining three columns can be any vectors that complete an orthonormal basis of  $\mathbb{C}^4$ . The behavior of  $U$  on the remaining basis states do not matter, since  $U$  only acts on the first column.

We construct such a unitary using an isometry. Conceptually, we define a linear map  $V : \mathbb{C} \rightarrow \mathbb{C}^4$  by  $V(1) = |\psi\rangle$ . This map preserves inner products and therefore extends to a full unitary by completing an orthonormal basis of the output space. This is computationally implemented using Qiskit's Isometry package, which takes  $|\psi\rangle$  as the image of  $|0\rangle$ , fills in the rest of the basis states, and generates the remaining columns automatically to make the 4 by 4 unitary. At this stage the circuit is correct but not optimized for any particular cost metric.

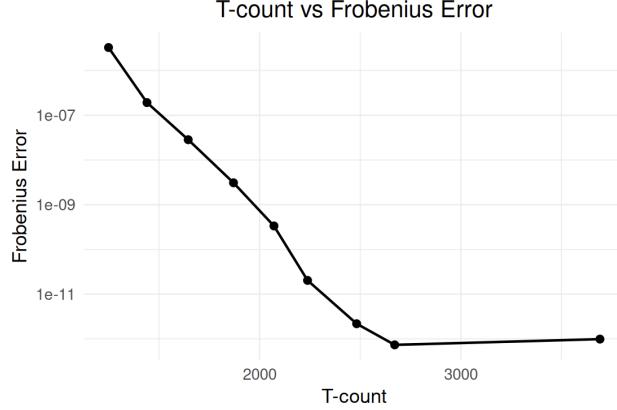


FIGURE 3. T-Gates vs Log Error. The Frobenius error is equivalent to the operator norm.

Now, to minimize T-count, the optimization starts by transpiling the isometry circuit into a gate set of 1-qubit rotations and CX gates. Aggregated 1-qubit operations are expressed as Clifford gates. If  $\theta$  is an integer multiple of  $\pi/4$ , the rotation is implemented exactly using Clifford and  $T$  gates. The remaining non-Clifford  $R_z(\theta)$  rotations are approximated using the `gridsynth` algorithm from problem 10, which makes Clifford+ $T$  circuits to an error tolerance, which can then be fit to find the optimal error bound per T-count.

Above is a plot of how the planned T-count that this problem was iteratively run in affected the error in the approximated state vector. It looks linear in log-space and tapers off at a T-count of 2500, which is at an error of about  $1e-12$ . The sudden flatline in approximation power is likely due to the approximator's built in floating point truncation, which serves as a good example of computational challenges that could limit approximation capabilities.

### 8. Structured Unitary 1.

$$\frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}$$

*Solution.* Looking for structure, we recognized that the entries in column 1 are multiplied by  $i$  each time, the entries in column 2 are multiplied by  $-1$  each time, and the entries in column 3 are multiplied by  $-i$  each time. This means that

$$U_{j,k} = \left( e^{i\pi/4} \right)^{jk}, \quad 0 \leq j, k \leq 3.$$

This helped us recognize  $U$  as a quantum Fourier transform. There is a standard construction (e.g. see Nielsen and Chuang's textbook [NC00]) that builds  $U$  using  $H$  and the controlled

$$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\frac{2\pi i}{2^2}} \end{pmatrix}$$

gate, which we can construct using three  $T$ -gates as

$$C - R_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{pmatrix} = (T \otimes I) \text{CNOT}(I \otimes T^\dagger) \text{CNOT}(I \otimes T).$$

Putting everything together gives us the quantum circuit in Figure 4 with  $T$ -count 3.

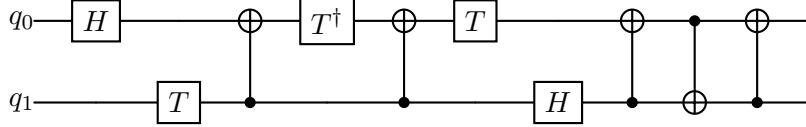


FIGURE 4. Quantum Circuit for Challenge 8

*Remark:* Before the last swap, we can observe that we need at least three  $T/T^\dagger$  gates using the phase-polynomial framework from [AM19]: from Figure 4 we find  $\varphi(x_1x_2) = x_1 + 7(x_1 \oplus x_2) + x_2$ , and the same argument from Challenge 11 carries through to argue any other representing polynomial with also have at least 3 odd coefficients.

*Remark:* In the spirit of this challenge, which tests the expressiveness of Clifford+ $T$ , we were wondering whether higher dimensional quantum Fourier transforms with  $n > 2$  can be expressed exactly. We believe the answer is no. Indeed, since

$$[\text{QFT} |x_0 \cdots x_{N-1}\rangle]_k = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} e^{\frac{2\pi i j k}{2^n}} x_j,$$

expressing QFT exactly would require  $e^{\frac{2\pi i j k}{N}} / \sqrt{N} \in \mathbb{Z}[1/\sqrt{2}, i]$ , which in particular includes  $e^{i\pi/2^{n-1}}$ . But when  $n \geq 3$  we know from field theory that  $\mathbb{Q}(\sqrt{2}, i) \subsetneq \mathbb{Q}(e^{i\pi/2^{n-1}})$ .

### 9. Structured Unitary 2.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} + \frac{i}{2} & \frac{1}{2} + \frac{i}{2} \\ 0 & i & 0 & 0 \\ 0 & 0 & -\frac{1}{2} + \frac{i}{2} & -\frac{1}{2} - \frac{i}{2} \end{bmatrix}$$

*Solution* Modulo a SWAP gate, we aim to construct

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & -\frac{1}{2} + \frac{i}{2} & \frac{1}{2} + \frac{i}{2} \\ 0 & 0 & -\frac{1}{2} + \frac{i}{2} & -\frac{1}{2} - \frac{i}{2} \end{bmatrix}$$

We noticed that this resembled a Controlled- $H$  gate, but were not able to make this perspective work. Instead, some trial and error of conjugating expressions brought us to find the identity

$$\underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}}_{H \otimes I} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{pmatrix}}_{H \otimes I} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}}_{H \otimes I} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1+i}{2} & \frac{1-i}{2} \\ 0 & 0 & \frac{1-i}{2} & \frac{1+i}{2} \end{pmatrix}.$$

Now, we can turn this into  $V$  by scaling rows and rows and columns by  $i$ . In terms of

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad M_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad M_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

we find

$$(1) \quad U = \text{SWAP} \cdot V = \text{SWAP} \cdot M_2 \cdot M_4 \cdot (H \otimes I) \cdot M_4 \cdot (H \otimes I) \cdot M_3 \cdot M_4.$$

Thus it suffices to construct  $M_i$ , which we do via

$$\begin{aligned} M_4 &= (T \otimes I) \text{CNOT} (I \otimes T^\dagger) \text{CNOT} (I \otimes T) \\ M_2 &= \text{CNOT}_{0 \rightarrow 1} \cdot M_4 \cdot \text{CNOT}_{0 \rightarrow 1} \\ M_3 &= \text{CNOT} \cdot M_4 \cdot \text{CNOT} \end{aligned}$$

Unfortunately, this involves lots of  $T$  gates, which makes the total  $T$ -count high (this was our first submission with a  $T$ -count of 15). However, we can simplify the expression in (1) extensively using

$$\text{SWAP} = (T \otimes I) \text{CNOT}(I \otimes T^\dagger) \text{CNOT}(I \otimes T).$$

Products of tensor-products collapse, giving a final answer of

$$U = (H \otimes I) \text{CNOT}(I \otimes T) \text{CNOT}(I \otimes T^\dagger) \text{CNOT}(H \otimes I).$$

which has a  $T$ -count of 3.

## 10. Random Unitary.

$$\begin{bmatrix} 0.1448081895 + 0.1752383997i & -0.5189281551 - 0.5242425896i & -0.1495585824 + 0.312754999i & 0.1691348143 - 0.5053863118i \\ -0.9271743926 - 0.0878506193i & -0.1126033063 - 0.1818584963i & 0.1225587186 + 0.0964028611i & -0.2449850904 - 0.0504584131i \\ -0.0079842758 - 0.2035507051i & -0.3893205530 - 0.0518092515i & 0.2605170566 + 0.3286402481i & 0.4451730754 + 0.6558933250i \\ 0.0313792249 + 0.1961395216i & 0.4980474972 + 0.0884604926i & 0.3407886532 + 0.7506609982i & 0.0146480652 - 0.1575584270i \end{bmatrix}$$

*Solution.* Any two-qubit unitary can be decomposed, up to a global phase, into local 1-qubit unitaries using the KAK decomposition given by

$$(K_1 \otimes K_2) A (K_3 \otimes K_4) = U$$

where  $K$  are 1-qubit unitaries and  $A$  contains the non-local entangling component of the qubits. We implement this decomposition using Qiskit's `TwoQubitBasisComposer(CXGate())`, which converts the random unitary into a quantum circuit that has arbitrary 1-qubit unitaries that are tied together with  $CX$  gates. This step isolates all 2-qubit entanglement into the  $CX$  gates, which just leaves optimizing the 1-qubit unitaries to have minimal error without using too many T gates.

$$U_{1q} = R_z(\alpha) H R_z(\beta) H R_z(\gamma)$$

up to a global phase. This representation is nice because it provides a structured way to create unitaries by implementing rotations about the Z axis together with Hadamard gates.

To minimize gate count and approximation error, single-qubit operations are accumulated as matrix products until a CNOT gate is encountered. At that point, the aggregated 1-qubit unitary must be flushed, where it is decomposed into the  $ZXZ$  form and multiplied into the circuit before being hit with the entangling  $CX$  gate.

We use the Python package `rmsynth` to minimize the number of T gates when creating the 1-qubit unitaries. Lastly, for decompositions that are close to being rotations about the Z axis of  $\frac{\pi}{2}$

or  $\frac{\pi}{4}$ , which would basically be T or S gates, those calculations are avoided.

See problem\_10.py in the GitHub repository for more details. Below is pseudocode for an algorithm that searches for a low T gate Clifford+T circuit, inspired by combinatorial brute-forcing:

```

Compute all 24 one-qubit Cliffords C
Store for each C:
    - matrix M(C)
    - inverse circuit C_dag

for eps_total in EPS_GRID do
    frame_pool = empty list

    for i = 1 to N do
        sample L0, L1, R0, R1 from Cliff(1)

        U_prime = (L0 kron L1) * U * (R0 kron R1)

        angles = extract Rz angles from KAK(U_prime)

        score = 0
        for theta in angles do
            k = nearest integer to theta / (pi/4)
            score += 1
            score += abs(theta - k * pi/4)
            score += abs(sin(theta / 2))
        end for

        store (score, L0, L1, R0, R1, U_prime) in frame_pool
    end for

    select K frames with lowest score from frame_pool

    for each selected frame (L0, L1, R0, R1, U_prime) do
        angles = extract Rz angles from KAK(U_prime)

        eps_list = RMS_allocate(angles, eps_total)

        base = KAK(U_prime) as CX and 1-qubit gates

        for each qubit q in {0, 1} do
            acc[q] = identity_2
        end for

        for each gate g in base do
            if g is CX(a, b) then
                for q in {a, b} do
                    decompose acc[q] into ZXZ
                    for each Rz(theta) in decomposition do
                        if abs(theta - k*pi/4) < tolerance then

```

```

        emit exact Clifford+T Rz
    else
        emit gridsynth(theta, next eps)
    end if
    end for
    reset acc[q] to identity_2
end for
emit CX(a, b)

else
    q = target qubit of g
    acc[q] = matrix(g) * acc[q]
end if
end for

flush remaining acc[q] in same way

for each candidate circuit C_prime from PyZX(base) do
    if C_prime adds no non-Clifford Rz gates then
        C = (L0_dag kron L1_dag) * C_prime * (R0_dag kron R1_dag)
        compute error(C, U)
        compute T_count(C)
    end if
end for
end for
end for

return circuit with smallest T_count satisfying a specific error target
(to snipe the leaderboard placement)

```

**11. 4-Qubit Diagonal Unitary.** Consider  $U$  acting on 4 qubits such that  $U|x\rangle = e^{i\phi(x)}|x\rangle$  and  $x \in \{0, 1\}^4$ . The phase  $\varphi(x)$  is the following:

$$\begin{aligned}
\varphi(0000) &= 0, & \varphi(0001) &= \pi, & \varphi(0010) &= \frac{5}{4}\pi, & \varphi(0011) &= \frac{7}{4}\pi, \\
\varphi(0100) &= \frac{5}{4}\pi, & \varphi(0101) &= \frac{7}{4}\pi, & \varphi(0110) &= \frac{3}{2}\pi, & \varphi(0111) &= \frac{3}{2}\pi \\
\varphi(1000) &= \frac{5}{4}\pi, & \varphi(1001) &= \frac{7}{4}\pi, & \varphi(1010) &= \frac{3}{2}\pi, & \varphi(1011) &= \frac{3}{2}\pi, \\
\varphi(1100) &= \frac{3}{2}\pi, & \varphi(1101) &= \frac{3}{2}\pi, & \varphi(1110) &= \frac{7}{4}\pi, & \varphi(1111) &= \frac{5}{4}\pi,
\end{aligned}$$

Compile this circuit with as few T and CNOT gates as possible.

*Solution.* Our approach follows the phase-polynomial framework of Amy and Mosca [AM19], which was discussed during the workshop. As a diagonal unitary operator with phases in multiples of  $\pi/4$ , they show there is a presentation

$$\varphi(x_3x_2x_1x_0) = \frac{\pi}{4}P(x_0, x_1, x_2, x_3)$$

where

$$P(x_0, x_1, x_2, x_3) = \sum_{S \subseteq \{0,1,2,3\}} a_S \bigoplus_{i \in S} x_i \in \mathbb{Z}_8[x_0, x_1, x_2, x_3]$$

is a weighted sum of linear Boolean functions. We first find such a presentation, and then use it to build a quantum circuit whose number of  $T$  gates equals the number of odd coefficients of  $P$ .

A naive search over all possible phase polynomials is computationally infeasible: there are  $8^{2^4-1} \approx 3.5 \cdot 10^{13}$  candidates. A key observation we made is that the given function  $\phi$  is invariant under any permutation of  $x_1, x_2, x_3$ . Motivated by this, we made the leap of faith to search for a candidate  $P$  which itself is symmetric in  $x_1, x_2, x_3$  as a polynomial. This reduces the number of free parameters down to 7, and now the search space of  $8^7$  can be exhaustively checked in under a second, yielding a unique (symmetric) solution

$$P(x_3, x_2, x_1, x_0) = 7 \cdot x_0 \oplus x_1 \oplus x_2 + 7 \cdot x_0 \oplus x_1 \oplus x_3 + 7 \cdot x_0 \oplus x_2 \oplus x_3 + 7 \cdot x_0 \oplus x_1 \oplus x_2 \oplus x_3.$$

This representation directly translates into a quantum circuit implementing  $U$ . In Figure 5 we present a construction which we believe minimizes the number of CNOT gates.

We now justify the optimality in the number of  $T$ -type gates.

**Claim.** Any  $P(x) \in \mathbb{Z}_8[x_0, x_1, x_2, x_3]$  representing  $\phi(x)$  has at least four odd coefficients.

*Proof.* Let  $\tilde{P} \in \mathbb{Z}_8[x_0, \dots, x_3]$  be any other phase polynomial satisfying

$$\tilde{P}(x) = P(x) \pmod{8}, \quad \text{for all } x \in \{0, 1\}^4.$$

Reducing modulo 2, we obtain two polynomials over  $\mathbb{F}_2$  that agree as functions  $\mathbb{F}_2^4 \rightarrow \mathbb{F}_2$ . Since  $\mathbb{F}_2$  is a field, this implies (e.g. by Lagrangian interpolation) they agree as polynomials in  $\mathbb{F}_2[x_i]$ . In particular, since  $P$  has 4 odd coefficients,  $\tilde{P}$  must too.  $\square$

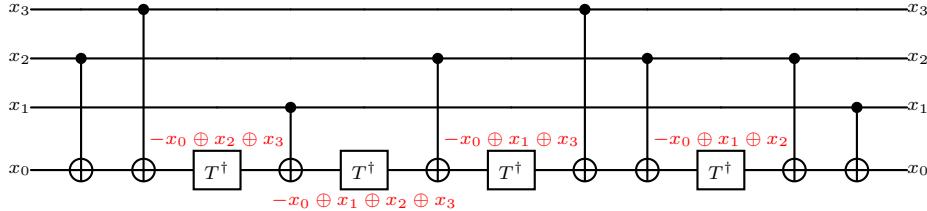


FIGURE 5. Quantum Circuit for Challenge 11

As shown in [AM19], the number of  $T$  gates is bounded below by the number of odd coefficients in any phase-polynomial representation, which completes the argument.

**12. Bonus - Commuting Pauli Phase Diagram.** In quantum simulation and fault-tolerant compilation, one often needs to implement products of small-angle exponentials of Pauli strings. You are given a dense list of commuting Pauli strings with fixed  $(\frac{\pi}{8})$ -quantized angles, and you must compile the resulting unitary into Clifford+T with as few T gates as possible.

$$U := \prod_{j=1}^m \exp \left( -i \frac{\pi}{8} k_j P_j \right)$$

*Solution.* While we did not have time to fully solve this bonus challenge during iQuHACK, we believe the approach is as follows:

Because all the Pauli strings commute, we can apply a global Clifford change of basis without changing the unitary  $U$ . Using  $H$  and  $S^\dagger$  gates, we conjugate each Pauli string so that every  $X$  and  $Y$  is mapped to a  $Z$ , leaving each term as a tensor product of only  $Z$  and  $I$  operators. Terms that act on the same Z-parity can then be combined by adding their phase coefficients mod 8, which causes many non-Clifford phases to cancel. The remaining odd-parity phases would be handled using a CNOT parity gadget and a single  $T$  or  $T^\dagger$  gate, after which the basis change is undone.

This approach is also well suited for implementation, since the required manipulations can be handled using existing tools such as PennyLane’s `clifford_t_decomposition` Python package.

## REFERENCES

- [AM19] M. Amy, M. Mosca, T-Count Optimization and Reed–Muller Codes. *IEEE Trans. Inform. Theory* **65** (2019), 4771–4784.
- [KMM13] V. Kliuchnikov, D. Maslov, and M. Mosca, Fast and efficient exact synthesis of single qubit unitaries generated by Clifford and T gates. *Quantum Inf. Comput.* **13** (2013), no. 7–8, 607–630. Also available at arXiv:1206.5236v4.
- [NC00] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000.
- [RS16] N. J. Ross and P. Selinger, Optimal ancilla-free Clifford+T approximation of z-rotations. *Quantum Inf. Comput.* **16** (2016), no. 11&12, 901–953.
- [SR18] P. Selinger and N. J. Ross, Exact and approximate synthesis of quantum circuits (newsynth package), Software package for quantum circuit synthesis, 2018. <https://www.mathstat.dal.ca/~selinger/newsynth/>
- [repository] LittleHalf, -67-Qubits-Superquantum, GitHub repository, <https://github.com/LittleHalf/-67-Qubits-Superquantum>