iQuHack 2026: Title

Adam Godel[1], Yebin Song[1], Nico Jackson[1], Travis Meyer[1], and Timothy Wright[1]

[1]Boston University, Boston, Massachusetts 02215, USA

February 1, 2026

Abstract

Placeholder

1   Introduction

The development and optimization of fault-tolerant quantum circuits has been at the forefront of quantum research in recent years. This weekend at iQuHack, Superquantum tasked us with implementing efficient approximate quantum circuit implementations of unitary gate operators using the Clifford + T gate set, where the latter is by far the most computationally difficult gate to execute and functions as a description of the "quantumness" of a circuit.

Our approach to implement these unitaries varied widely depending on the operator. Some have symmetries that can be exploited, while others are completely random and require brute-force optimization. Furthermore, there is a subjective tradeoff between minimizing the number of T gates and minimizing the distance between the approximate and the exact unitaries. In this writeup, we present our approaches to implement all eleven operators tasked in our challenge, considering how our implementations can be varied in both of these optimization regimes.

We will begin by establishing some background on quantum computing and circuit theory, then continue by discussing each of our implementations in detail, and conclude with our reflection and possible future work on these problems.

2   Background

A *qubit* is the fundamental bit of information in quantum computing. A *two-qubit quantum state* is represented as

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle, \qquad (1)$$

where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$. A two-qubit quantum state can be acted upon with *quantum gates*, which are represented as unitary matrices $A \in U(4)$ such that $|\psi\rangle \mapsto A|\psi\rangle$, which is a valid two-qubit quantum state. Similarly, one-qubit quantum gates are unitary matrices in $U(2)$.

The following quantum gates are known as the *Clifford gates*:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \; S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \; CX = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad (2)$$

When combined with the T gate, defined as

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}, \qquad (3)$$

they form a *universal gate set*, meaning that any unitary operation can be approximated to arbitrary precision using circuits built with just these four gates.

Moreover, the *Gottesman-Knill theorem* shows that the Clifford gates can be simulated in polynomial time on a classical computer. This reduces the "hardness" of a quantum circuit to be described in the number of T gates it contains.

1

3   Our Implementations

We now go through each of the eleven unitaries we implemented during the weekend, and the optimizations we made to leverage both low T gate counts and higher accuracy. We describe the implementations in order, although the difficulty in constructing each implementation varied widely.

3.1   Controlled-Y Gate

The *controlled-Y gate* is defined as

$$CY = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{bmatrix}. \tag{4}$$

To implement this gate, we consider that $(I \otimes S)CX(I \otimes S^\dagger) = CY$, where the $\dagger$ operator denotes the conjugate transpose. Since we can implement this gate using only Clifford gates, there is no tradeoff— we get high accuracy and no T gate cost.
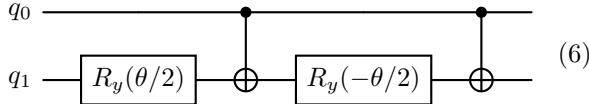
3.2   Controlled-$R_y(\pi/7)$ Gate

The *controlled-R_y gate* is defined as

$$CR_y(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos(\theta/2) & -\sin(\theta/2) \\ 0 & 0 & \sin(\theta/2) & \cos(\theta/2) \end{bmatrix}. \tag{5}$$

We want to consider the case where $\theta = \pi/7$, which does not map cleanly onto a Clifford $+$ T gate set. Therefore, we have to consider approximations. This is when we developed our central strategy for creating these implementations: ***modularity*** of gates.

Suppose we had a R_y gate in our toolbox. If this were the case, we could represent $CR_y(\theta)$ with the following circuit:

$$\tag{6}$$

We know that this is true by considering the state of q_1 based on the state of q_0. We know that $CR_y(\theta)$ should apply $R_y(\theta)$ if $q_0 = |1\rangle$ and do nothing if $q_0 = |0\rangle$. In Eq. 7, we can see that if $q_0 = |0\rangle$, nothing is done to q_1, while if $q_0 = |1\rangle$, we can see that a $R_y(\theta)$ gate is applied to q_1.

The problem now reduces to an implementation of $R_y(\theta)$.