Quantum Fully Homomorphic Encryption

6.5610 Final Project, Spring 2025

Rebecca Chang, Thomas Guo, and Evan Ren

Massachusetts Institute of Technology

(Dated: May 2025)

Abstract

Like their classical counterparts, quantum fully homomorphic encryption (QFHE) schemes would allow for outsourcing of quantum computations to an honest-but-curious server, allowing for efficient quantum computation on secure data. We demonstrate a fully secure QFHE scheme, its application to useful quantum circuits, and discuss possible steps towards secure schemes with less quantum communication requirements.

I. INTRO TO QUANTUM COMPUTING:

Quantum computing is a field that uses quantum mechanical phenomena to perform certain computations faster than classical computers. While classical computing uses bits, quantum computing uses qubits, which can exist in a superposition between two basis states. To get an output, such a qubit is measured, probabilistically yielding one of the two measurement results. Quantum computers can use quantum gates, physically acting as wave interference, to manipulate these probabilities [1].

Though an exciting technology, quantum computers still face many challenges before they can feasibly be widely used for computation. "DiVincenzo's criteria" outlines five conditions necessary for useful quantum computing – physical scability of the system, reliable state preparation (initialization), long quantum coherence times, universal quantum gates, and reliable qubit readout or measurement [2]. Many platforms have been proposed and developed to meet these requirements, such as superconducting qubits, neutral or Rydberg atoms, trapped ions, photonics, and more, each which present their own pros and cons in usage. Simultaneously, research in the theory side of quantum computing look at topics like quantum algorithms, complexity, simulation, machine learning, and, of course, cryptography, usually in hardware-agnostic ways. [1]

A. Useful Terminology: [1]

Dirac Notation: We will refer to states as $|\psi\rangle = a|0\rangle + b|1\rangle$, where when measuring the state $|\psi\rangle$, the probability of measuring $|0\rangle$ is $|a|^2$ and the probability of measuring $|1\rangle$ is $|b|^2$.

Unitary gates: All quantum gates acting on n qubits can be represented by a $2^n \times 2^n$ unitary matrix U. In other words, $U^\dagger U = I$, where $\dagger$ represents the conjugate transpose.

Since unitaries preserve norms, they preserve probability amplitudes.

Common gates: Some commonly used gates are as follows:

- Pauli gates: $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
- Other Clifford gates $H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
  - Note: We will use the names S and P interchangeably throughout this work; they refer to the same gate.
- T gates: $T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$

Each of these gate types is well-studied, with useful properties.

Universal Gate Sets: These are sets of gates which can be used to approximate any unitary gate to arbitrary (ϵ) precision. This is analogous to classical functional completeness with boolean operators. One commonly referenced universal gate set is the set of Clifford gates (CNOT, H, S) and the T gate. A couple other commonly referenced sets are the set of rotation operators with the phase shift gate and CNOT, and the set of the Toffoli (CCNOT) gate with the H gate.

The No-Cloning Theorem: It is not possible create a copy of an unknown quantum state. That is, a measurement scheme which does not disturb the system cannot exist. Notably, this only holds for unknown states – we must be able to reliably do state preparation for at least some known states to have useful quantum computation.

II. QUANTUM FULLY HOMOMORPHIC ENCRYPTION BACKGROUND:

Classical fully homomorphic encryption allows computations to be performed on encrypted data, the result of which is decrypted to get the same answer as performing those operations on the original data. Similarly, the quantum version allows for performing unitaries on encrypted states, then decrypting and performing phase and bit corrections to get the correctly evolved state. Both schemes allow for the outsourcing of computation to

honest-but curious servers without revealing information about the plaintext (or quantum state). This will become more useful as quantum computation hardware improves and moves to some form of cloud computing, where difficult or expensive gate operations are better done with outsourced hardware that may be more robust to noise, be more efficient with certain gate sets, or have better quantum coherence than what can be achieved locally. [3]

Quantum FHE is a part of a growing field called blind quantum computation (BQC), which studies how clients can have servers perform quantum computation without revealing the structure of the computation [4]. Many aspects are limited by what is known about quantum algorithms and the types of hardware it can be implemented on. We looked mainly at three papers.

The first paper [5] outlines a fully secure quantum FHE scheme for universal gate-based quantum computers, with security provided by a quantum one-time pad (QOTP) using Pauli X and Z gates. It requires a minimally-interactive quantum communication channel to send the initial encrypted state and ancilla qubits, and to receive the final state. It also requires the use of a classical FHE scheme for the Pauli decryption keys. Tham et al. demonstrate a practical implementation of this scheme with a photonic computing setup.

The second paper [6] improves the communication complexity of the quantum FHE scheme to nearly optimal, that is, within a factor of $1 + o(1)$ of the sum of the sizes of the initial state and final state. The reason this is difficult is because of the communication blow-up induced by the classical FHE scheme for the decryption keys. For example, with learning with errors (LWE), the blow-up is $\text{poly}(\lambda)$ where λ is the security parameter. Instead of using two QOTP bits per qubit (for X and Z), the paper describes how to leverage the structure of certain classical FHE schemes and rely on so-called spooky interactions to compress the QOTP. The packed bits are processed with a high-rate FHE scheme, yielding a leveled approach that can be made fully homomorphic with bootstrapping.

The third paper [7] provides a scheme relying only on a classical communication channel, but at the cost of relaxing information theoretic security to computational security. Its security relies on the security assumptions of the LWE problem. An additional benefit of this approach is that the client no longer needs to have quantum capabilities that allow it to prepare and send states.

III. OUR WORK:

Our implementation roughly follows the algorithm outlined by Tham et al. [5], between client Alice and server Bob.

First, Alice generates random Pauli keys, $\vec{a}, \vec{b} \in \{0,1\}^N$, as well as keys pk, sk for a classical FHE scheme. Then, she prepares $2N$ ancilla qubits, whose phases are dependent on the desired circuit and can be prepared with minimal use of H, Z, S, and $CNOT$ gates. She then encrypts the initial quantum state $|\phi\rangle$ as $|\psi\rangle = Z^{\vec{a}} X^{\vec{b}} |\phi\rangle$, and encrypts the Pauli keys using the classical FHE scheme. She sends the encrypted initial state, the ancilla qubits, and the encrypted Pauli keys to Bob.

Next, Bob, the honest-but-curious server, applies the sequence of intended gates in the circuit. If the gate is non-Clifford (in our case, a T gate), then phase correction must be applied using the ancilla qubits. Then for any gate, he homomorphically updates the encrypted Pauli keys. He then returns the udpated $|\psi\rangle$ and updated encrypted Pauli keys to Alice.

Finally, Alice decrypts the Pauli keys via the classical FHE scheme, and uses the decrypted Pauli keys $\vec{a}', \vec{b}'$ to decrypt the final state as: $Z^{\vec{a}'} X^{\vec{b}'} |\psi\rangle = \mathcal{U}|\phi\rangle$ for the intended sequence of unitaries (circuit) $\mathcal{U}$.

The security of this scheme has two components. The security of the Pauli key component is equivalent to the security of the classical FHE scheme used. The security of the encryption of the initial state follows from the fact that they appear like randomly generated [mixed] states. Formally, looking at the density matrix of the encrypted state for randomly selected $\vec{a}, \vec{b}$,

$$\sum_{\vec{a},\vec{b}} \frac{1}{2^n} Z^{\vec{a}} X^{\vec{b}} |\phi\rangle\langle\phi| Z^{\vec{a}} X^{\vec{b}} = \mathcal{I}_n / 2^n$$

implying a fully random state. Intuitively, this can be understood using a simple example with just the pure states $|0\rangle$ and $|1\rangle$. Even if the server knew that the client begins with one of these two states, say it measures $|1\rangle$ – then it cannot know if this came from $X|0\rangle$ or $|1\rangle$, which (along with these two states with Z), all have equal probability, implying equal probability the encrypted state came from the two pure states. Without any prior knowledge on the initial state, the only way to gain information is via repeated measurements of the same state to get the probability distribution of the encrypted state under the same keys

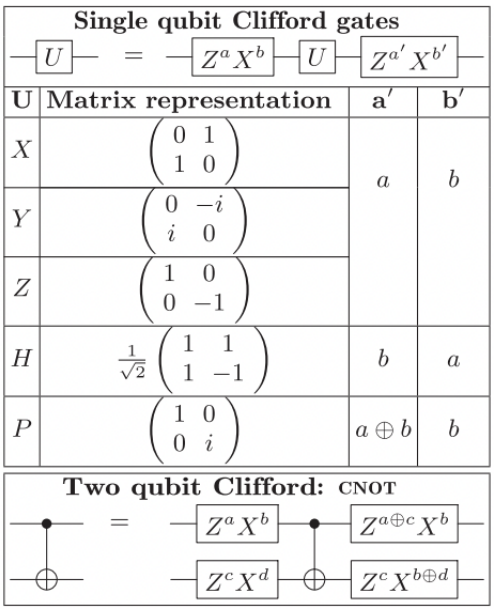

FIG. 1. Fig 1 in Tham et.al., homormophic key updates for some Clifford gates

– this is easily combated by regenerating $\vec{a}$ and $\vec{b}$ every shot, as due to the No-Cloning Theorem, Bob cannot copy the unknown state and must rely on Alice's encrypted messages.

The most interesting part of the scheme is how X and Z corrections can be applied. Because Pauli X and Z errors conjugate with Cliffords, they can be propogated to the end of the circuit/scheme and applied by Alice without needing any extra interactive steps between client and server. The homormophic updates can be derived by the effect of the conjugation and actually turn out to be quite simple updates for the Cliffords (see Fig. 1). Similarly, the homomorphic updates for additional useful gates can be derived (notated as encryption $\rightarrow$ gate $\rightarrow$ correction):

$$Z^a X^b \bigotimes Z^c X^d \rightarrow SWAP = CNOT(x,y) \cdot CNOT(y,x) \cdot CNOT(x,y) \rightarrow Z^c X^d \bigotimes Z^a X^b$$

as expected from the SWAP gate. Also,

$$Z^a X^b \rightarrow P^\dagger \rightarrow Z^{a \oplus b} X^b$$

the same as for P, which can be intuitively checked as applying the gates $P^\dagger$ and P do "cancel" and result in the original $Z^a X^b$.

Finally, the CZ gate:

$$Z^a X^b \bigotimes Z^c X^d \rightarrow CZ(x,y) = H(y)CNOT(x,y)H(y) \rightarrow Z^{a \oplus d} X^b \bigotimes Z^d X^{b \oplus c}$$

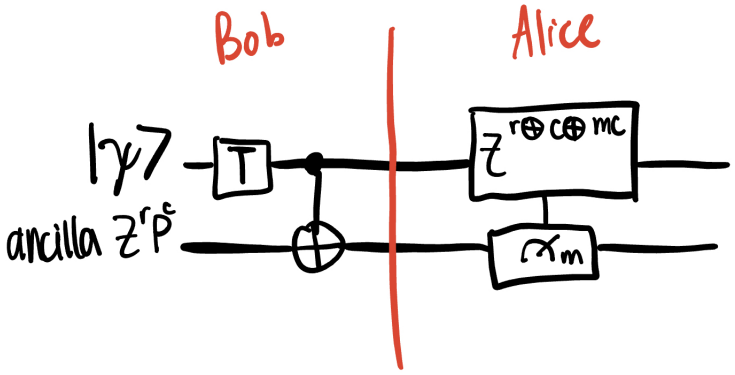

FIG. 2. Applying phase correction saved in ancilla qubits. Z errors can be propogated to the end, unlike phase errors, and Bob never knows r or c.

The more difficult case is when we have T gates. T gates introduce a phase when propogated across Paulis:

$$T|\psi\rangle = TZ^aX^b|\phi\rangle = P^bZ^aX^bT|\phi\rangle$$

We must correct this phase before we can continue with the rest of the circuit, every time there is a T gate. Therefore, Alice must use the ancilla circuits to save the necessary phase corrections using a process similar to quantum gate teleportation, which generally applies operators saved in ancilla qubits and is frequently seen in such contexts with T gates. In total, there are 2^{2N} possible ancillary qubits that could be needed because all addition is performed modulo 2 – these phases can be generated from the ancillas $Z^rP^{a_j}|+\rangle$ and $Z^sP^{b_k}|+\rangle$ for each a_j and b_j in $\vec{a}$ and $\vec{b}$, and random binary digits r and s which are saved by Alice. The security here, in terms of hiding the a_j and b_k, is the same as before – the use of the random bits and the fact that a malicious party can only measure once and cannot clone states to build an actual distribution, means that no information can be leaked through the ancilla states. So, finally, the phase correction can be applied and the necessary Pauli updates can be derived using the measurement of the ancilla qubits (circuit in Fig 2). This process can be slightly simplified because there are essentially three choices for what b' could be – a, b, or $a \bigoplus b$ – and casework can be done in each of these cases to determine exactly what ancillas are necessary. However, the number of ancillas is still $O(cN)$ with c between 1 and 2.

We implemented the full scheme with some simple circuits to test functionality. We used

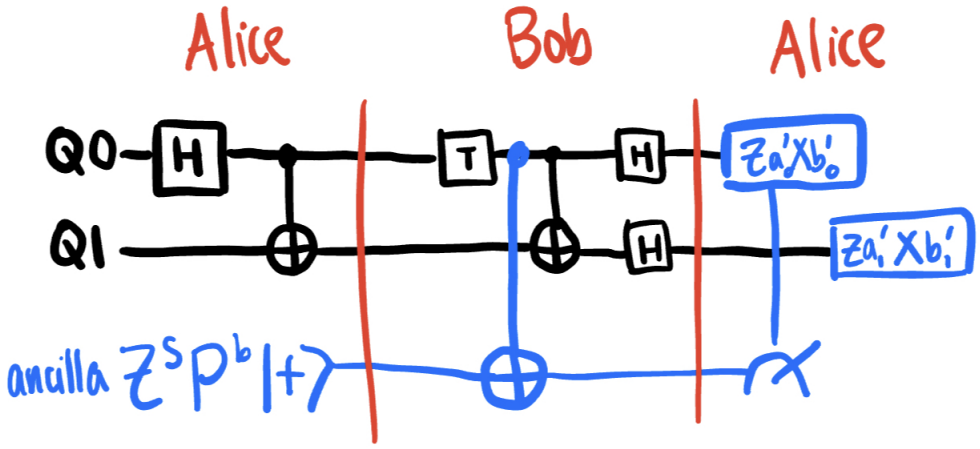

FIG. 3. Sample circuit to show working T gate. Black is the "normal" circuit, blue is what is added for QFHE

an existing classical FHE package to encrypt the Pauli keys, and performed the simulations using AerSimulator in Qiskit.

The first circuit we tested was a simple circuit to showcase the function of a T gate (Figure 1). The results are in Fig 3, where to read the results, the first two numbers in each decrypted count represent the qubit measurements and the last two represent the ancilla measurements, so an accurate result would just come from disregarding the ancilla bits and adding counts from the same main qubit measurements. This was compared to calculating the circuit via matrix multiplication in Mathematica and achieved the expected distribution of counts. Similarly correct results were reached with other simple compositions of gates. In addition, measurement in various parts of the circuit (what a curious party may try to do to gain information about the initial state) yielded near 50% measurements for each basis over many shots with random Pauli keys $\vec{a}, \vec{b}$ even with the same starting state, experimentally corroborating the theory behind the density matrix security argument from earlier.

Then, as a more useful example, we ran a small example of Grover's algorithm – using the initial state $|\phi\rangle = |00\rangle + |01\rangle + |10\rangle - |11\rangle$, we would expect the $|11\rangle$ component to be amplified, as Grover's amplifies the component for which the oracle returns -1. This code snippet better showcases the information that is shared between Alice and Bob. From Alice, Bob only takes in the encrypted version of the circuit and the Pauli keys and returns

```
out = ClassicalRegister(2, "out")
cipher_qc.add_register(out)
cipher_qc.measure([0, 1], out)
# ---------------- Simulation ----------------
simulator = AerSimulator()
result = simulator.run(cipher_qc, shots=10000).result()
counts = result.get_counts()

print("Final decrypted measurement counts:")
print(counts)
```

```
Final decrypted measurement counts:
{'01 10': 365, '11 10': 366, '10 10': 2116, '01 00': 367, '00 00': 2108, '00 10': 2149, '11 00': 375, '10 00': 2154}
```

FIG. 4. Sample T gate QFHE results

```
# ---------------- Simulation ----------------
cipher_qc, enc_a, enc_b, enc_c, enc_d, pk, sk = AliceEncryptGrover()
cipher_qc, enc_a, enc_b, enc_c, enc_d = BobGrover(cipher_qc, enc_a, enc_b, enc_c, enc_d)
cipher_qc = AliceDecryptGrover(cipher_qc, enc_a, enc_b, enc_c, enc_d, pk, sk)
cipher_qc.measure_all()

results11 = 0
for _ in range(10000):
  simulator = AerSimulator()
  result = simulator.run(cipher_qc, shots=1).result()
  counts = result.get_counts()
  if '11' in counts:
    results11 += counts['11']

print("Final decrypted measurement counts:")
print("11:"+str(results11))
```

```
Final decrypted measurement counts:
11:10000
```

FIG. 5. Running Grover's Algorithm

their updated versions, which is what Alice uses to decrypt, and a measurement at the end tallies to the counts we see, generating a new set of Pauli keys every shot. In this case, we do in fact get $|11\rangle$ every time using a two-qubit version of Grover's algorithm, the expected answer.

IV. CHALLENGES, CONCLUSIONS, FUTURE WORK:

One of the main downsides to this scheme is the required resource overhead – this formulation requires $2N$ ancilla qubits, for N the number of T gates. The Clifford and T gate set is universal. However, the Solovay-Kitaev Theorem states that single-qubit gate sets which densely generate $SU(2)$, such as our gate set, can approximate desired quantum gates in $O(\text{polylog}(1/\epsilon))$ for small error ϵ [8]. Having the number of qubits linearly depend on this $O(\text{polylog}(1/\epsilon))$ factor for each unitary can get extremely expensive, especially with the

extremely limited number of logical qubits currently used by quantum computing platforms. This is one of the reasons why Tham et al. were only able to experimentally demonstrate the use of a single T gate, rather than a complete useful algorithm. In our attempts to code circuits for algorithms as "basic" as low-qubit Quantum Fourier Transform, an algorithm commonly used as a base block for algorithms such as HHL, we quickly ran into an unmanageable number of T and $T^\dagger$ gates, and associated ancillas, which arise from the minimum decomposition of controlled-S and Toffoli gates. [9]

Also, note that $\vec{a}$ and $\vec{b}$ must be regenerated for every shot, and as is the difficulty in most quantum processes, Alice must run many shots to obtain a useful distribution from the final state. Thus, while the scheme is non-interactive for each shot, there must be the back and forth communication during each shot with different $\vec{a}$ and $\vec{b}$.

The other main issue that has begun to be addressed by other researchers is the difficulty in communication [7][6]. Currently, this scheme requires Alice to be able to locally prepare initial and ancilla states and send them via a quantum channel to Bob. A more feasibly implementable solution would see Alice only communicate classically with Bob, such as sending an encoding of the relative amplitudes and phase of the state and having Bob do all state and ancilla preparation. This is, however, extremely difficult to design while maintaining security.

Finally, this scheme is bottlenecked by all the expected difficulties of quantum computation. Some gates may be difficult to implement on certain platforms, despite being important for universal quantum computation (i.e. often T gates, Toffolis, and irrational gates). However, without these gates, the Gottesman-Knill Theorem states Clifford-only circuits can be efficiently simulated by a classical computer, and a quantum computer can only yield a speedup by a polynomial factor. In addition, the problems with error correction and fault tolerance, number of logical qubits, and other issues that generally currently limit quantum computing are still applicable to this scheme.

In the future, more work in measurement-based quantum computing and blind computation in general are needed for efficient and secure quantum FHE. One possible direction is using the quantum signal processing framework and its extension the quantum singular value transformation, which in the most popular formalism alternates uses a unitary and Z rotations to perform polynomial transformations on a linear operator encoded in that unitary. This framework has been shown to comprise many useful algorithms, including

search, phase estimation, and simulation [10], and work has been done considering the use of a discrete set of Z rotations that would be less expensive to implement but can still fully express all possible polynomials. The use of such a framework and more Solovay-Kitaev-type proofs could mean we only need to find homomorphic key updates for the unitary and the set of rotations.

V. WHO DID WHAT:

We all read and discussed the papers, Becca coded. `https://colab.research.google.com/drive/1UGBOdTqNAXGhlbRK8axV1QgZ_w3ppMs6?usp=sharing`

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2010).

[2] D. P. DiVincenzo, The physical implementation of quantum computation, Fortschritte der Physik **48**, 771–783 (2000).

[3] A. Broadbent and S. Jeffery, Quantum homomorphic encryption for circuits of low t-gate complexity, in *Advances in Cryptology – CRYPTO 2015* (Springer Berlin Heidelberg, 2015) p. 609–629.

[4] J. F. Fitzsimons, Private quantum computation: An introduction to blind quantum computing and related protocols (2016), arXiv:1611.10107 [quant-ph].

[5] W. Tham, H. Ferretti, K. Bonsma-Fisher, A. Brodutch, B. C. Sanders, A. M. Steinberg, and S. Jeffery, Experimental demonstration of quantum fully homomorphic encryption with application in a two-party secure protocol, Physical Review X **10**, 10.1103/physrevx.10.011038 (2020).

[6] O. Chardouvelis, N. Döttling, and G. Malavolta, Rate-1 quantum fully homomorphic encryption, in *Theory of Cryptography: 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8–11, 2021, Proceedings, Part I* (Springer-Verlag, Berlin, Heidelberg, 2021) p. 149–176.

[7] E. Davies and A. Kay, Efficient post-quantum secured blind computation (2024), arXiv:2404.07052 [quant-ph].

[8] C. M. Dawson and M. A. Nielsen, The solovay-kitaev algorithm (2005), arXiv:quant-ph/0505030 [quant-ph].

[9] T. Kim and B.-S. Choi, Efficient decomposition methods for controlled-r n using a single ancillary qubit, Sci Rep (2018).

[10] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang, Grand unification of quantum algorithms, PRX Quantum **2**, 10.1103/prxquantum.2.040203 (2021).