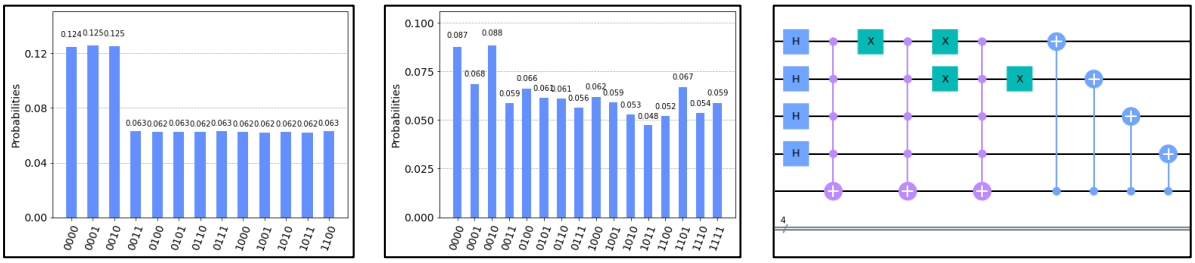


(a) (b) (c)

Figure 8: Step distribution (mod 13)

In this example we take a uniform distribution and eliminate the probability of the states 1101 - 1111 (13 to 15) in a 5-qubit circuit measuring only the first 4 qubits. This implementation can also be used for modular addition but require the use of an extra ancillary qubit as a carry bit that is not measured. We see in Figure 8 (a) the expected distribution using a simulation and in (b) the result from the quantum computer. The obvious discrepancy between the expected step distribution and the result in the quantum computer may correspond to how the multi controlled Toffoli gates are implemented in real quantum computers. Each multiple control gate is composed of a combination of single-qubit and two-qubit gates. These implementations create congruent results from the expected classical behavior but with different phases (Adriano Barenco, 1995). It has been proposed that this circuits be used in conjunction with other similar circuits that cancel out their phase differences.

Applications to cryptography

It has been proven that a quantum computer with enough power and low decoherence can solve many cryptographic schemes and algorithms. Such a system works by using the Quantum Fourier Transform to find factors of large integers. Although no such computer is widely available today, it is plausible that the systems will become commercially available in the near term (5 to 15 years). Many of the gates and implementations shown above could be used in schemes combining the power of different quantum algorithms such as Shor and Grover. We present what could be a piece of a larger quantum deciphering system.

A simple application for the field of cryptography can be demonstrated using the shift cypher. We can model the message as a 5-qubit quantum register with an extra qubit for addition mod 32, and another register to model the relative position of the shifted letter (Figure 9). The size of the key determines the number of qubits needed for the position register. One qubit yields at most a 2-bit key, whereas a key of size n needs approximately $\log_2(n)$ number of qubits. An encrypted message sent over a quantum channel can be decrypted with the known key.

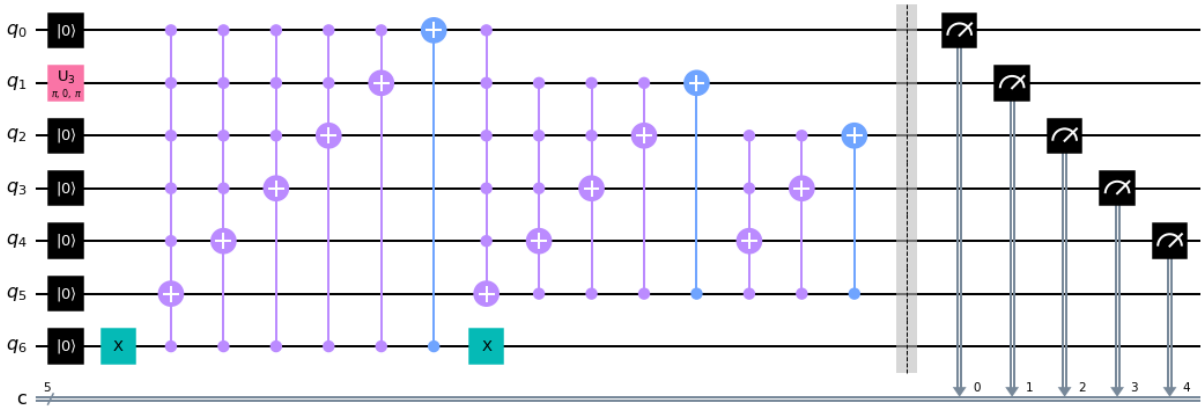


Figure 9 Proof of Concept Shift Cypher

In the proof-of-concept circuit we show an example application of the shift cypher. The circuit is initialized with the code representing a letter (01000) we add mod 26 the letter that represents the cypher key (0001). The measured outcome is expected to be the encrypted letter (00011). Qubit q5 represents the ancillary qubit for the carry-bit operation of mod 26, and q6 is the qubit for the position of the message. This circuit can be extended with a probability distribution as the initialization of the circuit and the letter as the shift. Adding more qubits for the position allows for an extension of the circuit for a Vigenère cypher. This method can be the basis of a more sophisticated quantum algorithm for frequency analysis by using the inverse alphabet gate to find either the key or decipher a message.

An extremely powerful tool in Quantum Computation is the Grover Algorithm which uses a technique called amplitude amplification to obtain an $O(\sqrt{N})$ speedup in an unstructured search. Using some of the previously discussed ideas we built an experiment to search for a marked element that represents the solution to a problem (Figure 10). For example, given a known expected distribution, such as the English alphabet frequency distribution, we may find a modified distribution with a hidden value that shifts or alters the expected result, in other words the key to the cypher.

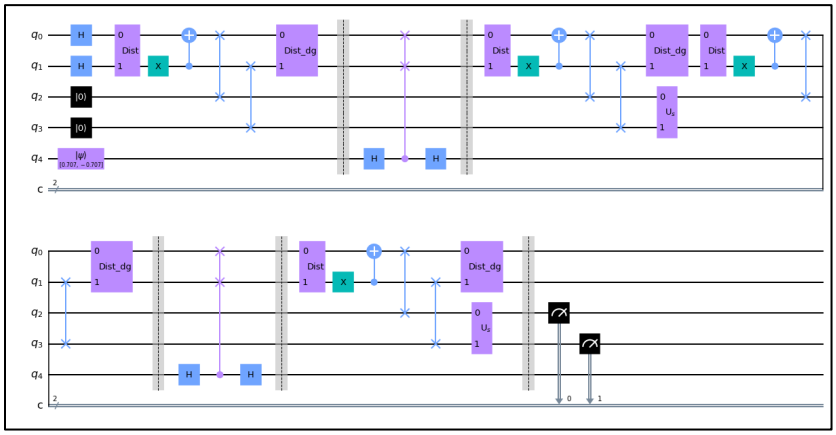
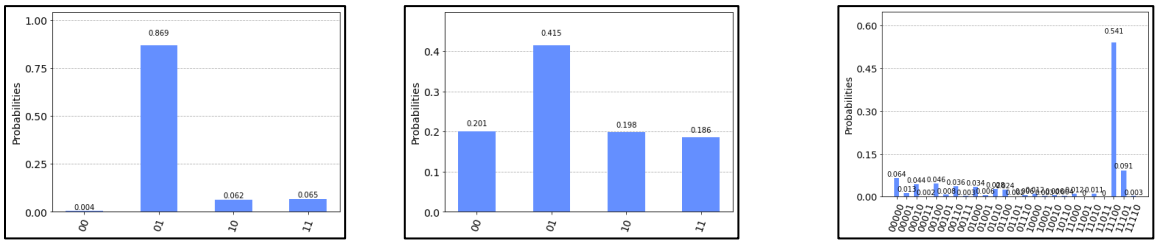


Figure 10 Grover Search for frequency analysis

In **Figure 10** we present a Grover Search to find the key that modifies an alphabet distribution as expected from the English language. We begin with a uniform distribution using the Hadamard gates. The gate called *Dist* is the alphabet distribution discussed previously. Next, we have the marked gates that represent the shift of the cypher. The combination of an alphabet distribution and a shift produces the distribution of the cypher. *Dist_dg* is the gate that represents the inverse of the alphabet distribution, which is the heuristic we know. We then do a test for a uniform distribution using the controlled swap gate test (between two vertical bars). The procedure is repeated until a solution is found by measuring qubits q2 and q3 which are connected to the top two qubits by swap gates but are also subject to the effect of the Grover coin. The Grover diffuser produces a uniform distribution but marks the solution state, the key to the cypher, with a phase shift.



(a) (b) (c)

Figure 11 Grover Search Result

In Figure 11 (a) we show the previous circuit run in the simulator backend while in (b) the same circuit was tested using IBM Quito. In (c) we present a generalization of the same circuit using 5 qubits per register, to approximate the English alphabet, with a success showing the marked state, representing the key, of 11100.

Conclusion

We explored several implementations of quantum circuits. Each of the quantum circuit can be built into gates of more complex circuit structures to solve problems in a quantum computer. The circuits were tested on real quantum computers with 5-qubits, but more qubits are needed to be used in more complicated algorithms. The biased distributions and Grover search described in the paper could be implemented as modules of a quantum deciphering tool. Quantum computing is fast becoming a reality and recent advances in the development of hardware implementations may bring access to quantum computation for many applications particularly in the field of security and cryptography. Quantum processes by their very nature are random processes and there are other direct applications to these circuits.

Acknowledgment

The work supported is based upon this material by, or in part by the National Centers of Academic Excellence (NCAE-C) under contract/award H98230-20-1-0411.

References

Aaronson, S. (2013). *Quantum Computing Since Democritus*. New York: Cambridge University Press.

Adriano Barenco, C. H. (1995, November). Elementary gates for quantum computation. *Phys. Rev. A, 52*(5), 3457-3467.

Aharonov, D. (1999). Quantum Computation. *Annual Reviews of Computational Physics VI*, 259-346.

Aharonov, Y. D. (1993). Quantum random walks. *Phys. Rev. A*, 1687-1690.

Arute, F. (2019, October 24). Quantum supremacy using a programmable superconducting processor. *Nature, 574*, 505-511.

Changpeng, S. Y. (2019). Quantum Algorithm Design: Techniques and Applications. *J Syst Sci Complex*, 375-452.

Childs, A. (2009). Universal computation by quantum walk. *Phys. Rev. Lett*(102), 180501.

Edward L. Wolf, G. B. (2017). *Josephson Junctions: History, Devices, and Applications*. Danvers: Pan Stanford Publishing.

Kempe, J. (2003). Quantum random walks - an introductory overview. *Contemporary Physics*, 302-327.

Nielsen, M. A. (2010). *Quantum Computation and Quantum Information*. Cambridge, UK: Cambridge University Press.

Venegas-Andraca, S. (2012). Quantum walks: a comprehensive review. *Quantum Inf. Process, 11*(5), 1015-1106.

Xia, F. L. (2019). Random Walks: A Review of Algorithms and Applications. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Yulin Wu, H. a.-S.-C.-H.-L. (2021, Jun 28). Strong quantum computational advantage using a superconducting quantum processor. *quant-ph*(2106.14734).

Zhao, Y.-G. Y.-Q. (2016, feb 04). Novel pseudo-random number generator based on quantum random walks. *Scientific Reports*, 6:20362.

Zickert, F. (2021). *Hands-on Quantum Machine Learning with Python* (Vol. 1 Get Started). PyQML.