Implementation of Shor's Algorithm for RSA Encryption

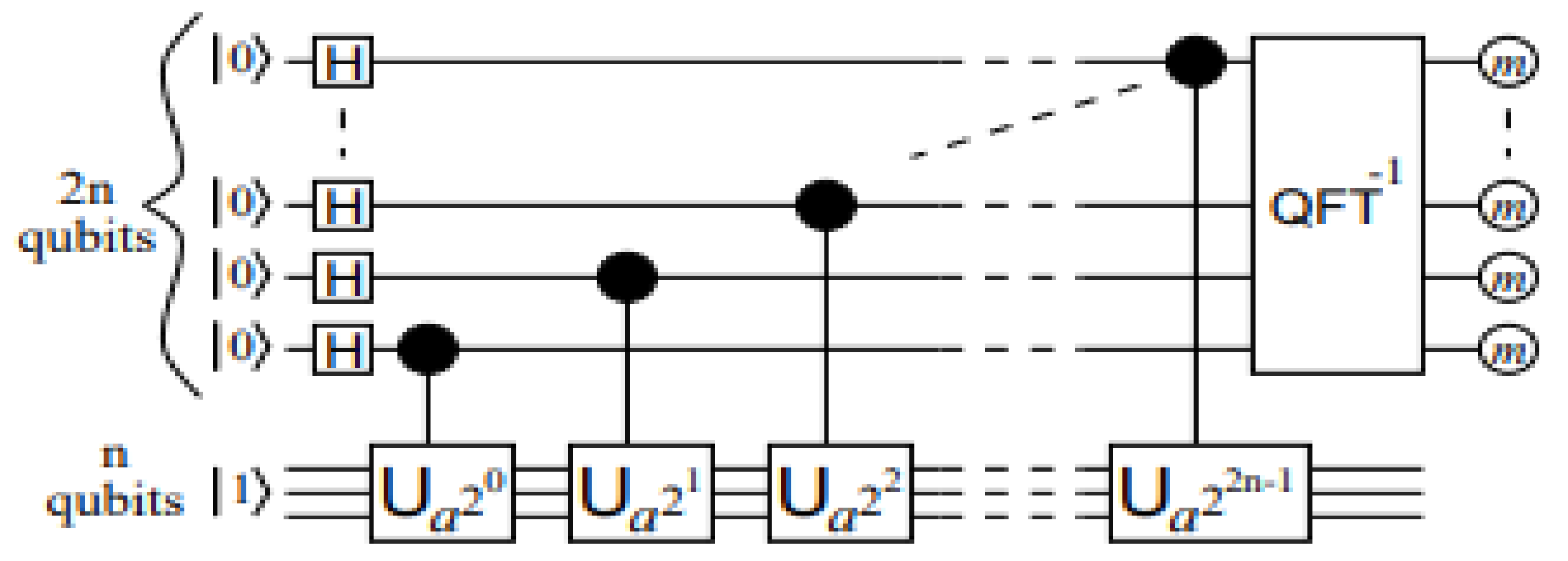
Ryan Willey, Lianxin Xin, and Jun Ren
Delaware State University

Abstract

Cryptography can be traced back thousands of years ago, and as technology has evolved so to have the methods used to encrypt and transmit messages. One of the most well-known encryption methods is RSA Encryption, whose strength relies on the computational intractability of factoring a product of two large prime numbers. With access to a sufficiently large quantum computers however, a user would be able to break this encryption schemes with relative ease. The purpose of this presentation is to introduce Shor's period-finding algorithm and its respective subroutines, demonstrating a quantum system's potential ability to factor these large numbers in polynomial time by taking advantage of quantum parallelism and interference. The examples of Shor's algorithm illustrated here are only proof of concept for now as quantum processors available to us are of insufficient size to rapidly factor large numbers and hence significantly prevail the classical encryption method. Using IBM's quantum computing platform Qiskit, we will show successful implementation of Shor's Algorithm on a quantum simulator then later direct the operation on a real quantum processor. Errors introduced by the noisy Qubits are briefly summarized at the end.

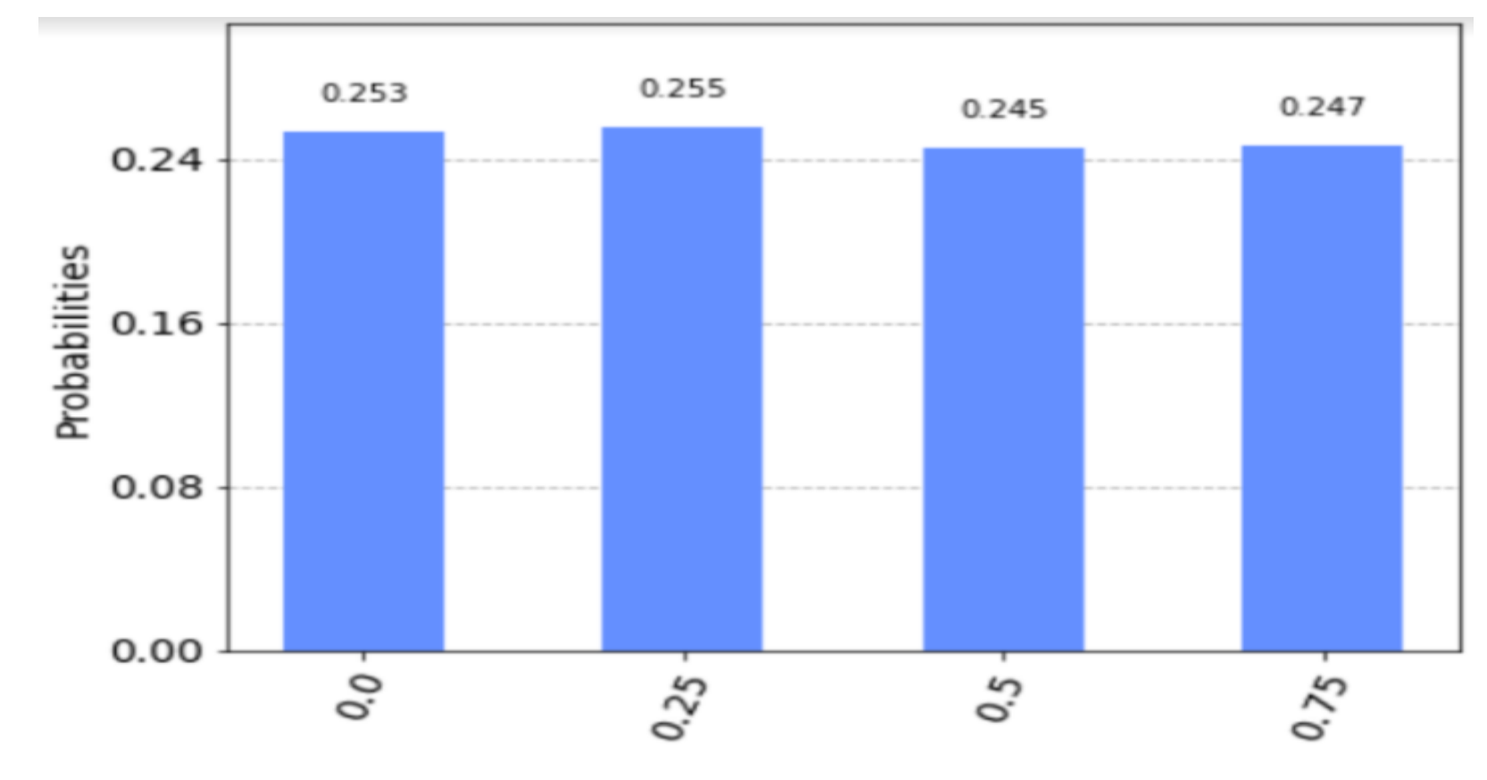
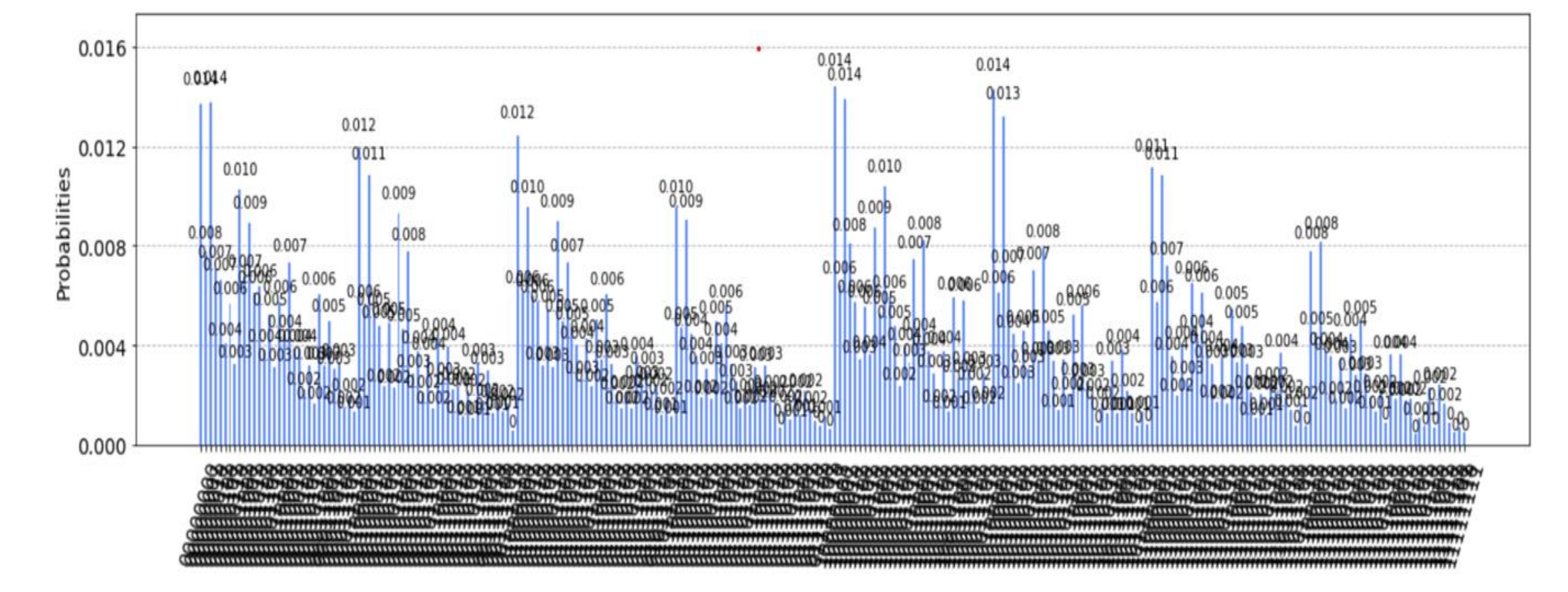
Factoring using Shor's Algorithm

1. Make a guess of the factor 'a', 1< a< N, where N is the number to be factored, must have gcd(a,N) = 1
2. Create unitary gate $U_a|x\rangle = |ax \bmod N\rangle$
3. Create target register of size n = $\log_2(N)$ and counting register of size 2n

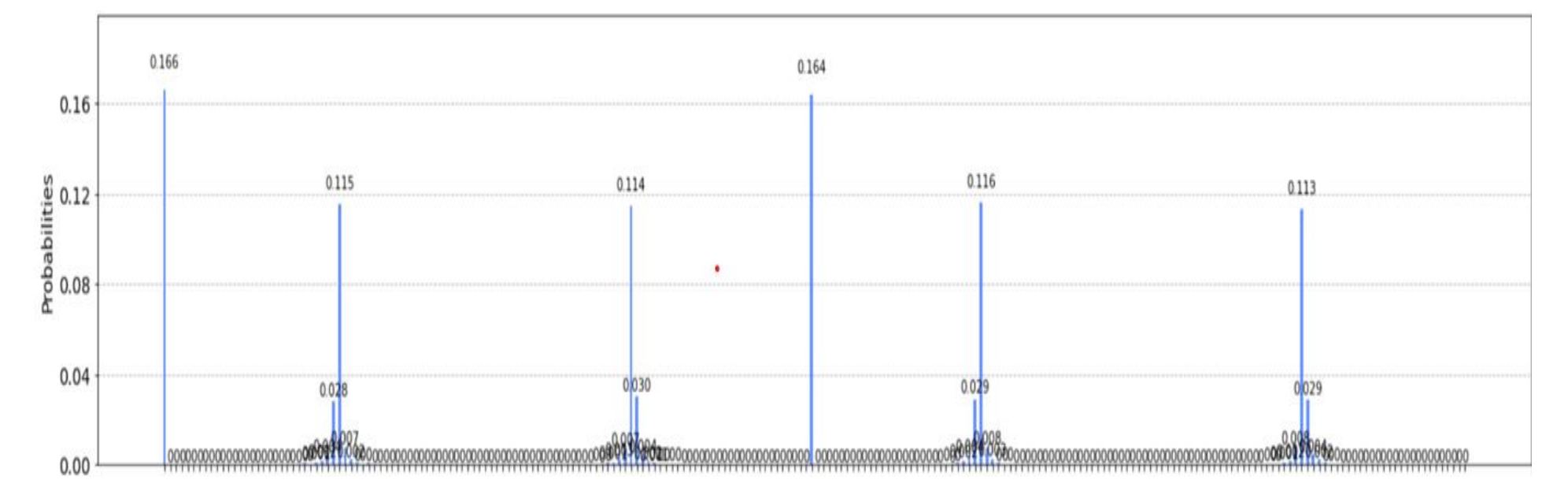
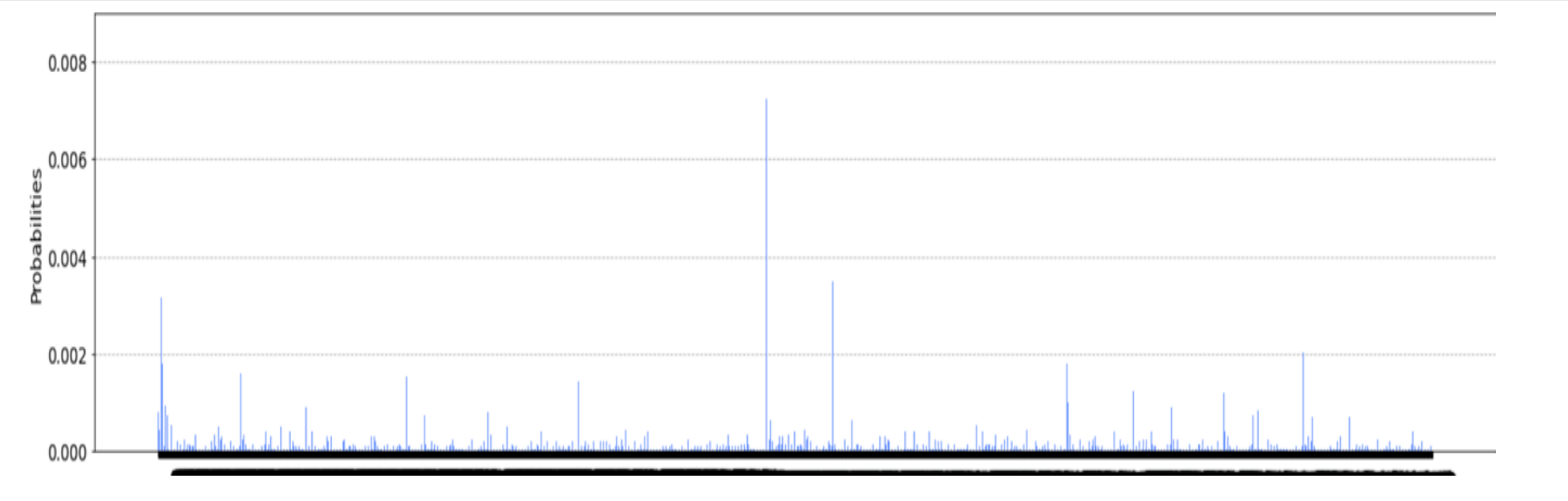


4. Encode phase of U_a into counting register using controlled U_a^(2i) operations
5. Perform QFT-inverse to extract the phase (s/r) from the counting register, order is given by r. 0<s<r.
6. $(a^r - 1) = (a^{r/2} + 1)(a^{r/2} - 1) \cong 0$ (mod N), if r is odd then fail, otherwise compute gcd[$(a^{r/2} + 1)$, N], gcd[$(a^{r/2} - 1)$, N], if they are coprime then fail, otherwise we have obtained the factors. (Classical Processing)

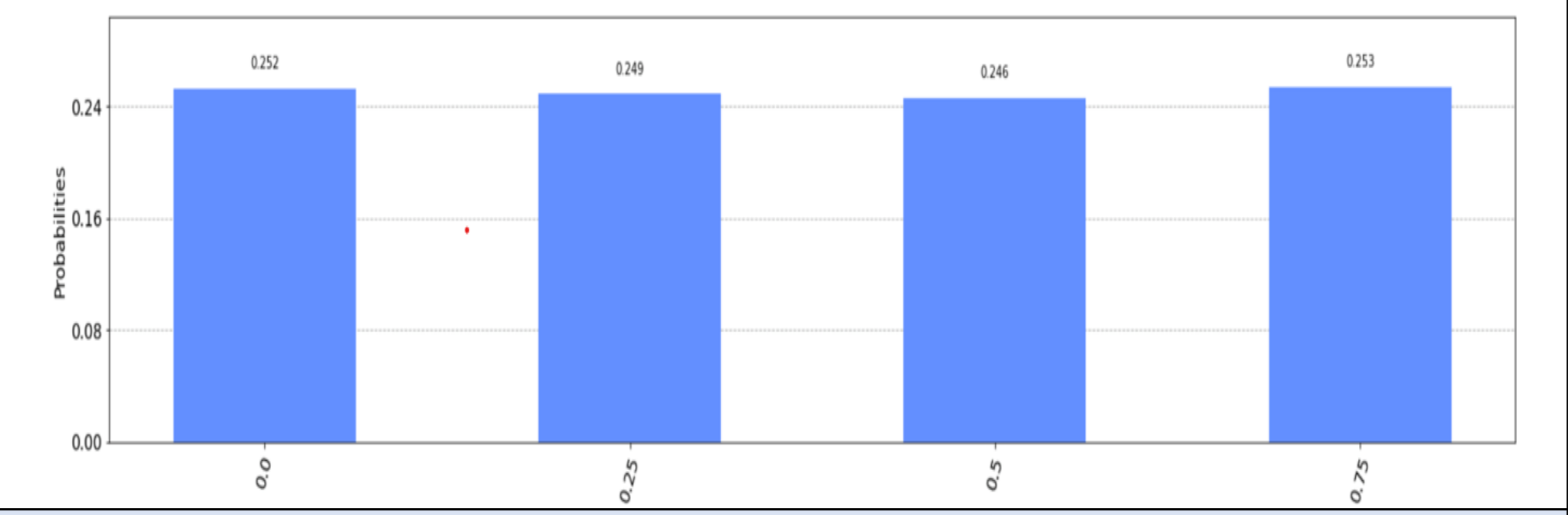
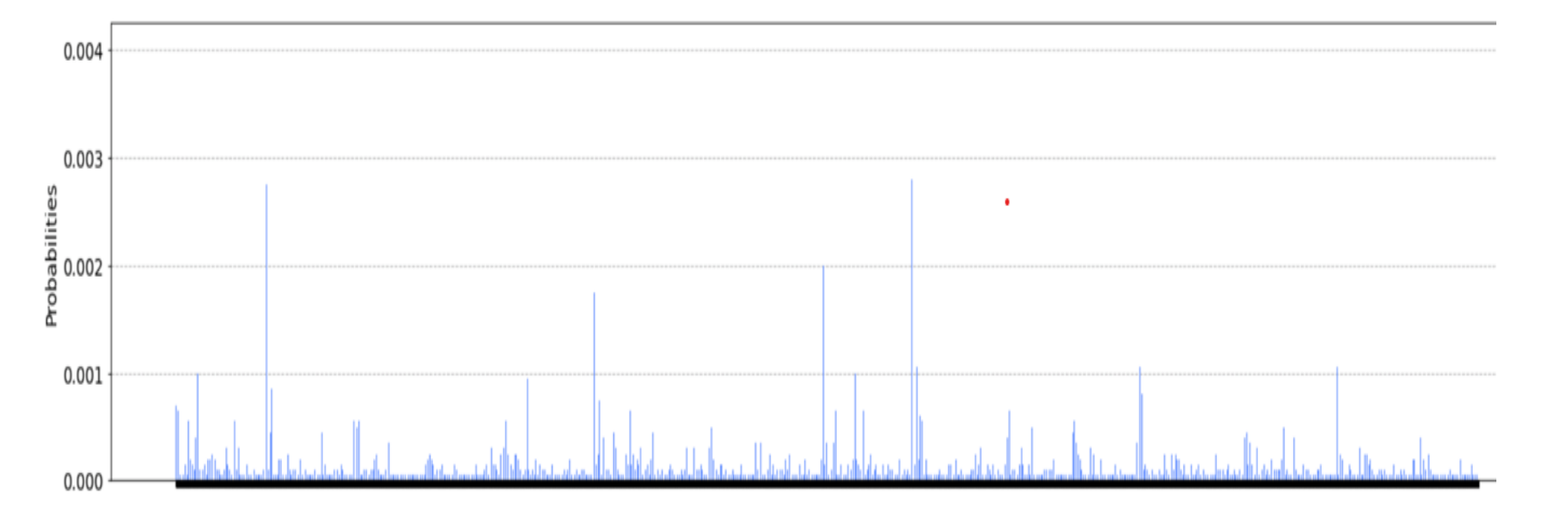
Data for factoring N = 15, a = 7

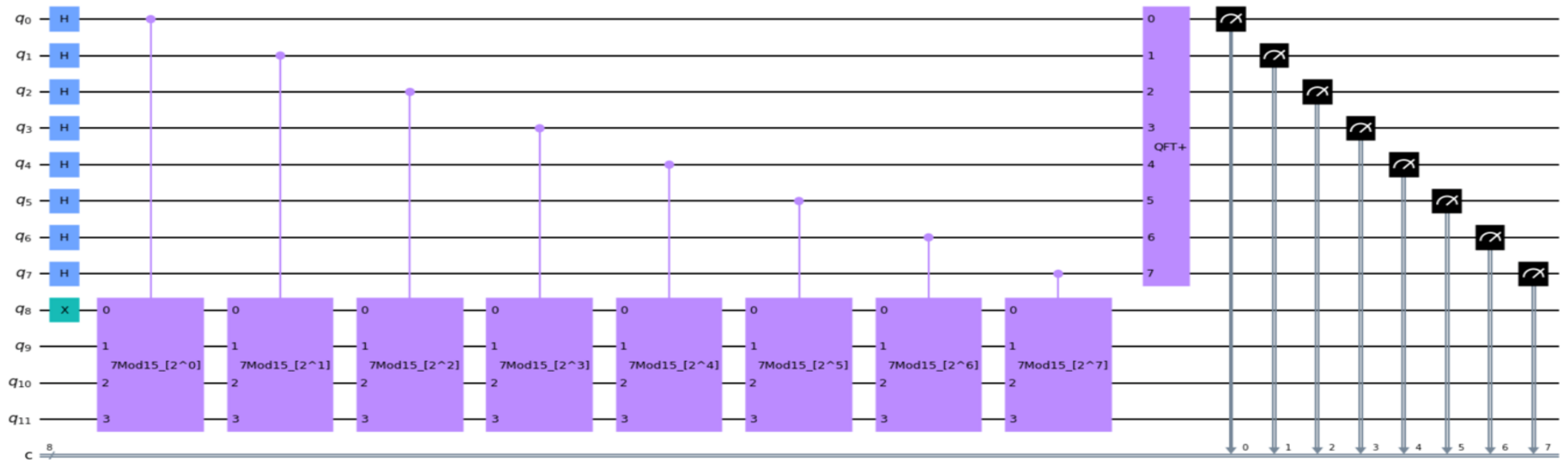
Simulator results	Ibmq-guadalupe
	
Probability of factoring = 74.70%	Probability of factoring = 3.80%, # qubits = 8

Data for factoring N = 119, a = 33

Simulator results	Ibm-cairo
	
Probability of factoring = 16.39%	Probability of factoring = 5.23%, # qubits = 14

Data for factoring, N = 119, a =13

Simulator results	Ibm-cairo
	
Probability of factoring = 74.70%	Probability of factoring = 3.75%, # qubits = 14



Quantum Circuit for finding r such that $7^r \cong 1$ mod (15), circuit uses four and eight qubits in the target and counting registers, respectively. We first create a superposition of all possible values using Hadamard Gates, write the phase of U onto the counting register, apply the inverse-QFT to extract the phase value. Measurements should be observed to peak at intervals of (1/r). Made using Qiskit

Hardware Limitations

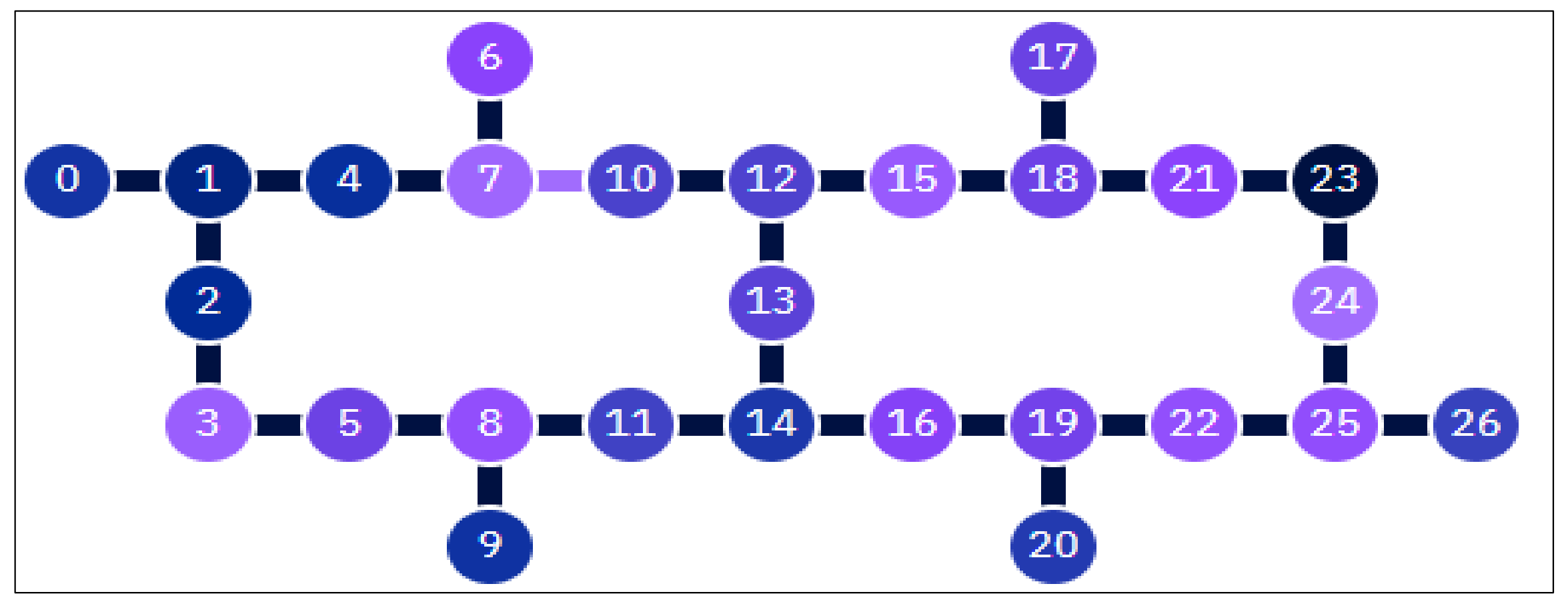


Figure: Physical qubit coupling map for ibm-cairo, average error rate for C-NOT gate average = 0.4%

Circuit (N,a)	Circuit Depth	Execution Time (s)
(15,7)	46,603	323.5
(119,33)	18,157	78.8
(119,13)	973	18.1

Figure: Circuit depth dependent guess selected and physical qubit assignment

Summary

Although at a low rate, our group was successful in factoring both 15 and 119 using both quantum simulators and real quantum computer systems. The results of this project demonstrate the ability quantum computers have today, but more importantly to give us an idea of the future roadmap of quantum computing. It is claimed that it would take many thousands of qubits to break RSA-2048 encryption in reasonable time while the largest quantum system, IBM's Eagle processor (127 qubits) would still take way too long to be of any use in cryptography. Within the next decade or two however there may exist quantum systems of such a scale that our classical encryption methods may be at high risk. Shor's algorithm has the potential to break RSA and other encryptions like Diffie-Hellman which relies on the difficulty of discrete logarithms

References

Beauregard, Stephane. (2003, Feb 21). *Circuit for shor's algorithm using 2n+3 qubits – arxiv.org*. arxiv. Retrieved March 22, 2022, from https://arxiv.org/pdf/quant-ph/0205095.pdf

IBM Quantum. (2017, Mar 7). *Open-source quantum development*. Qiskit. Retrieved March 22, 2022 from https://qiskit.org/

Acknowledgements

- Advanced Quantum Sensing Center (DOD contract W911NF2020276
- IBM Quantum Lab