Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 8th, 2020

Different Quantum Computing Architectures Affect Resulting Probabilistic Outputs

Ashish Kumar, Paula Medina, Lewis Westfall, Avery Leider, and Charles C. Tappert
Seidenberg School of Computer Science and Information Systems, Pace University
Pleasantville, NY 10570, USA
Email: {ak33894n, ph79486p, lwestfall, aleider, ctappert}@pace.edu

***Abstract*—Quantum computers have achieved a solid base in both physics and mathematics. Many people believe that a quantum computer is like a classical computer – once you have seen one, you have seen them all, and they all give the same results. Maybe someday all quantum computers will work the same way but not today. Unlike classical computers where the bits have a similar relationship between individual computers, the quantum bits (qubits) of a quantum computer are connected in many different patterns. Reviewing the architecture of quantum computers used in the IBM Quantum Experience shows the variety of ways that the qubits are connected to each other. This study used the superdense coding algorithm to show that different quantum computing architects affect the resulting probabilistic outputs.**

***Index Terms*—Quantum Computing, Qubits, Superdense Coding, IBM Quantum Computers**

I. Introduction

Quantum computing is one of the leading applications of quantum physics. It has the potential to solve some of the most complex problems which are beyond the reach of even today's most powerful supercomputers. Quantum computers are not going to replace classical computers but a different way of operating enables them to perform some calculations that classical computers cannot perform efficiently [1]. Classical computers encode data in bits and each bit can represent a 0 or a 1. Each bit acting as an on-off switch. The bits ultimately translate into computing functions That can perform a simple calculation like solving a maze. A classical computer will test each possible route one at a time to find the correct one [2].

Just as the classical computer has bits quantum computers have quantum bits, called qubits. Qubits are the foundation for Quantum computing. They make use of two key principles of quantum physics superposition and entanglement [3]. Superposition means that each qubit can represent a zero, a one or both at the same time . Entanglement connects two qubits in such a way that any operation on one qubits affects the other. Using these two principles qubits can act as much more sophisticated version of switches helping quantum computers solve difficult problems that are virtually impossible for classical computers.

Superdense coding, or Dense coding, is a quantum information process that allows one person to send two classical bits to another person using only a single qubit of a pair of entangled qubits. [4] This protocol was first proposed by Bennett and Wiesner in 1992 [5] and experimentally actualized in 1996 by Mattle, Weinfurter, Kwiat, and Zeilinger using entangled photon pairs. [6] Superdense coding is one of the underlying principle of secure quantum secret coding. The necessity of having both qubits to decode the information being sent eliminates the risk of eavesdroppers intercepting messages.

Superdense coding starts with a third party called Eve. Eve has two quantum bits which are assume going to start with zero state i.e is $|0\rangle$. Then she applies Hadamard gate to the first qubit to create superposition and then apply the CNOT gate to both qubits were the first qubit act as a control and second qubit act as a target. Applying the CNOT gate causes two qubits to become entangled, also called a Bell pair. Now Eve sends one of the qubits to another person we call him Alice and the other qubit to another person call him Bob. The idea of the superdense coding protocol is that Alice wants to share two classical bits of information to Bob using the qubit sent by Eve. But before she does she needs to apply a set of quantum gates depending on 2 bit of information she wants to send.

TABLE I
Encoding Rules for Superdense Coding (For Alice)

Message	Quantum Gate	Resulting state
00	I	$\|00\rangle+\|11\rangle$
10	X	$\|01\rangle+\|10\rangle$
01	Z	$\|00\rangle-\|11\rangle$
11	ZX	$\|10\rangle-\|01\rangle$

So if Alice wants to send 00 then all she needs to do is to apply I gate. To send 10 then she has to apply X gate. Depending on the bit of information she wants to send she can apply the appropriate gate and send it to Bob for the final process.

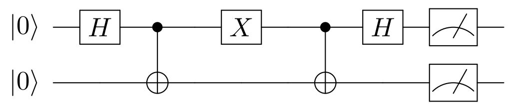
The final process is when Bob receives the qubit sent by Alice.

TABLE II
Decoding Rules for Superdense Coding (For Bob)

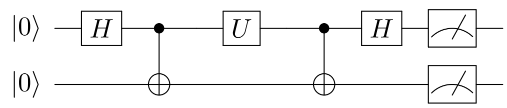
Bob Receives	After CNOT Gate	After H-Gate
$\|00\rangle+\|11\rangle$	$\|00\rangle+\|10\rangle$	$\|00\rangle$
$\|01\rangle+\|10\rangle$	$\|10\rangle+\|11\rangle$	$\|10\rangle$
$\|00\rangle-\|11\rangle$	$\|00\rangle-\|01\rangle$	$\|01\rangle$
$\|01\rangle-\|10\rangle$	$\|11\rangle-\|10\rangle$	$\|11\rangle$

To decode the information Bob needs to reapply CNOT and Hadamard gate to qubits. Bob applies the CNOT gate using the rightmost qubit as control and the leftmost qubit as target and finally applies Hadamard gate to extract the information. Table II shows the qubits when Bob receives Alice's qubit, after he applies the CNOT, and the final results after the Hadamard gate is applied.

Circuit 1 shows the complete quantum circuit diagram for Alice using the X gate to encode the message. We see the Hadamard and CNOT gates used by Eve to entangle, the X gate used by Alice to encode the message, and finally, the CNOT and Hadamard gates used by Bob to extract the message. Circuit 11 shows the general case using the U gate to symbolize any unity gate.



(1)



(2)

The IBM Quantum Experience (IBM QX) is IBM's cloud-based quantum computing service. As of now, there are seven processors on IBM QX having four different architectures. The most complicated architecture is that of IBM's 16 qubit computer in Melbourne, Australia Figure 1. The 16 qubit computer can have 2^{16} possible states. The circles indicates qubits and the arrow shows the connectivity between the qubits. In 16 qubit processor, each qubit is connected in squared shape.

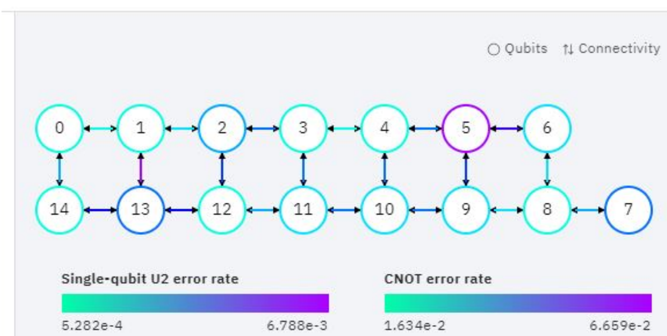


Fig. 1. IBM 16 qubit Melbourne

The rest of the IBM QX quantum computers have 5 qubit processors and are deployed in a different parts of the world. All of them have 5 qubits and 2^5 possible states.

Below is the 5 qubit processor in Yorktown Heights, where each qubit is connected in a star shape. By looking at Fig 2 we can see that there is direct connectivity between qubits q[0] q[1], qubits q[0] q[2], qubits q[2] q[3], qubits q[2] q[4] and qubits q[3] q[4] and connection between qubits q[1] q[3], qubits q[0] q[3] and qubits q[0] q[4] are not direct and in order for them to form connection they have to pass through another qubit.

Below the architecture for the 5 qubit processor located in London, Burlington, Essex, Ourense, and Vigo. In each of

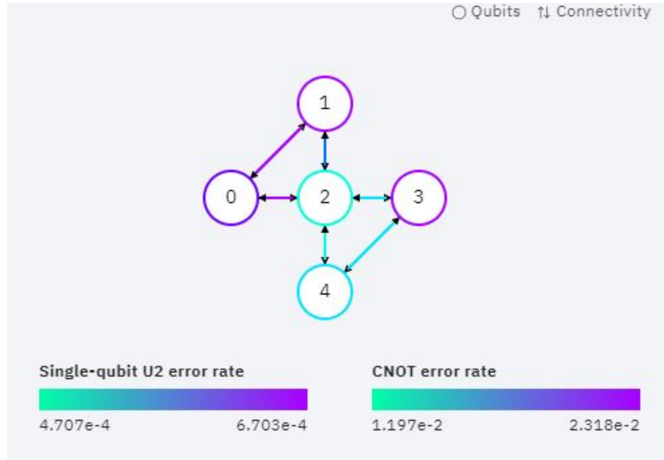


Fig. 2. IBM 5 qubit Yorktown

these the qubits are connected in T shape. By looking at Fig 3 we can see that there is direct connectivity between qubits q[0] q[1], qubits q[1] q[2], qubits q[1] q[3], and qubits q[3] q[4] and connection between qubits q[0] q[2], qubits q[0] q[3] and qubits q[0] q[4] are not direct and in order for them to form connection they have to pass through at least one other qubit.

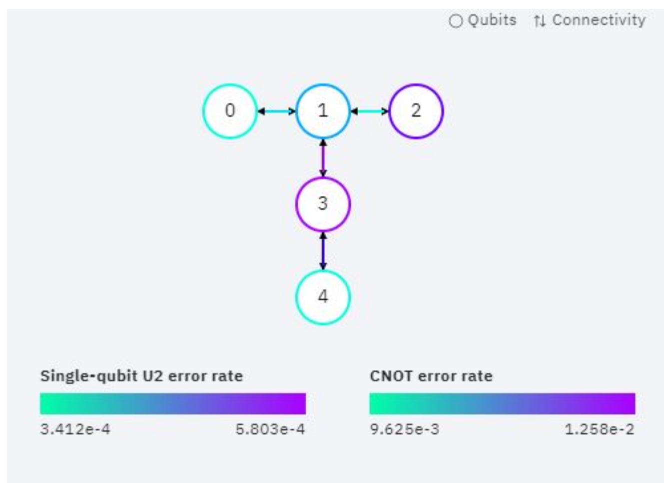


Fig. 3. IBM 5 qubit London

Below Figure is a 5 qubit processor in Rome where each qubit is connected linearly. By looking at Fig 4 we can see that there is direct connectivity between qubits q[0] q[1], qubits q[1] q[2], qubits q[2] q[3], and qubits q[3] q[4] and connection between qubits q[0] q[2], qubits q[0] q[3] and qubits q[0] q[4] are not direct and in order for them to form connection they have to pass through at least one qubit.

Our goal is to show how the computer's architecture affects the performance of superdense coding. For this paper, we are going to run the circuit on IBM 5 qubit London and IBM 5 qubit Rome processor and will be connecting different qubits

Fig. 4. IBM 5 qubit Rome

to see if the result differs and also check for error or noise rates.

II. Literature Review

Jozef Gruska [7] book, on quantum computing begins with essential concepts, models, strategies and results displayed in a efficient way. This book emphasis more on computational aspects, models, strategies and issues than on the details on technological features which are of significance for implementation of quantum information processing systems.

Nielsen and Chuang [8]This book gives an presentation to the most thoughts and methods of the field of quantum computation and quantum information. Rapid progress in this field and its cross-disciplinary nature have made it difficult for newcomers to get a broad outline of the foremost vital method and results of the field.This book contains twofold. Firstly, it present the fundamental concept related to computer science, mathematics and physics necessary to understand quantum computation and quantum information and second purpose of the book is to develop in detail the central results of quantum computation and quantum information.A review in the November 2001 edition of Foundations of Physics says, ”Among the handful of books that have been written on this new subject, the present volume is the most complete and comprehensive” [9].

III. Project Requirements

We Will Need Anaconda environment [10], and a Jupyter Notebook containing QISKit, which can be downloaded with Anaconda from QISKit. Each individual user will need an API key obtained at no cost from IBM's Q Experience [11].

The following histograms used in this study are all direct output from our Jupyter Notebook connected to the the IBM Q experience quantum computer [11]. The Jupyter Notebook implements the open source software development kit (SDK), QISKit, that includes a Python API [12] that translates the Python into Quantum Assembly Language (QASM) [13], which is then processed by an IBM quantum computer [14].

IV. Methodology

As we mentioned earlier about the different architecture of the quantum processor. For this paper, we will examine the result of the circuit transferring 10 state on 5 qubit processor that IBM has deployed in London and Rome. Initially, we required to have an IBM Q account to run the circuit. Once we logged in we can create a circuit using a circuit composer. Below is the first complete circuit for the 10 states.

Figure 5 shows the composer diagram, circuit 1, used to transfers 10 messages using qubits 0 and 1. Figure 6 Shows the diagram for qubit 0 and 2, and Figure 7 shows the diagram for qubit 0 and 3. The message encoded is 10, but because of the different qubits used, the primary results are 00010 for circuit 1, 00100 for circuit 2 and 01000 for circuit 3. In the results they will all be identified as 10. Each of these circuits were run on IBM's London and Rome quantum computers to get the results on two separate architectures.

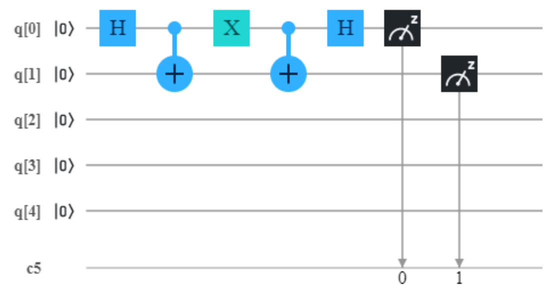


Fig. 5. Circuit 1

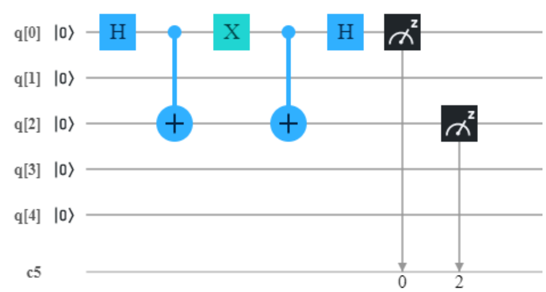


Fig. 6. Circuit 2

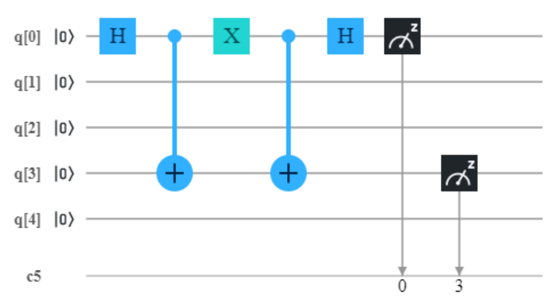


Fig. 7. Circuit 3

A. Linear Algebra

The language quantum mechanics is linear algebra. Linear algebra is a study of linear transformation and the entities they act on vectors [1].

Since the qubit is spoken to by a vector and the operations on the qubit or different qubits is done by matrix manipulation, a few information of direct variable based math is required. Don't stress, the sum you would like is only a little portion of linear variable based math additionally the simpler portion. [15].

The essentials [16]

Scalar A scalar is one dimensional having only a magnitude.

Vector A vector is two dimensional quantity which has a magnitude and a direction. Each element of a vector is scalar.

- Ket - $|\varphi\rangle$ - Ket vector are complex vector that is it can be real , imaginary or both. This is a column vector.
- Bra - $\langle\varphi|$ - for inner product of two vector we need Bra vector. It is the conjugate transpose of a ket and is a row vector.

$$|\varphi\rangle = \begin{bmatrix} a \\ b \end{bmatrix} \tag{3}$$

$$\langle\varphi| = \begin{bmatrix} a & b \end{bmatrix} \tag{4}$$

Matrix A matrix is n x m and every element is a scalar. Below is a 3 x 4 matrix.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \tag{5}$$

Scalar Multiplication To multiply by a scalar, simply multiply each element by the scalar.

$$s \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} = \begin{bmatrix} sa_1 & sb_1 \\ sa_2 & sb_2 \end{bmatrix} \tag{6}$$

Product of a vector and a matrix

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} a_1c + b_1d \\ a_2c + b_2d \end{bmatrix} \tag{7}$$

Inner product $\langle A \mid B\rangle$.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & g \\ f & h \end{bmatrix} = \begin{bmatrix} ae + bf & ag + bh \\ ce + df & cg + dh \end{bmatrix} \tag{8}$$

Tensor product

$$|V\rangle \otimes |W\rangle = |V\rangle\,|W\rangle = |VW\rangle \tag{9}$$

$$V = \begin{bmatrix} v_1 & v_2 \\ v_3 & v_4 \end{bmatrix}, W = \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} \tag{10}$$

$$|VW\rangle = \begin{bmatrix} v_1 & v_2 \\ v_3 & v_4 \end{bmatrix} \otimes \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix}$$

$$= \begin{bmatrix} v_1 \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} & v_2 \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} \\ v_3 \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} & v_4 \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} \end{bmatrix} \tag{11}$$

$$= \begin{bmatrix} v_1w_1 & v_1w_2 & v_2w_1 & v_2w_2 \\ v_1w_3 & v_1w_4 & v_2w_3 & v_2w_4 \\ v_3w_1 & v_3w_2 & v_4w_1 & v_4w_2 \\ v_3w_3 & v_3w_4 & v_4w_3 & v_4w_4 \end{bmatrix}$$

All this linear algebra are needed for Quantum computing.

Superdense coding has been presented in a theoretical manner using a quantum diagram and notation of qubit values. Now we will start with the unmodified qubits and add the steps one at a time until the complete process for one of the classical bit information communication messages has been described.

The five quantum gates we will use are the identity (I), Pauli-X (X), Pauli-Y (Y), Pauli-Z (Z), and the Hadamard (H).

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{12}$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{13}$$

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{14}$$

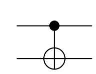
$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{15}$$

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{16}$$

We use one multiple qubit gate, the controlled not or CNOT gate. CNOT is a 2 qubit operator where one qubit with dark spot acts as a control and another with XOR symbol act as a target. If the control qubit is set to 1 then it flips the target qubit . The matrix operation for the CNOT is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{17}$$

and In Quantum circuit diagram CNOT gate black dot-connecting to qubit wire is for control bit and circle connecting to a wire is target qubit



(18)

The ket values $|0\rangle$ and $|1\rangle$ correspond to the column vectors $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ respectively.

To create entangled pair of qubits we combine the use of Hadamard gate with CNOT gate.Below figure shows circuit for Entanglement.

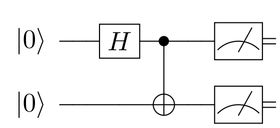


Fig. 8. Quantum Circuit for Entanglement

The diagram below shows the effect of combining Hadmard gate and CNOT gate to generate first Bell states given both qubit starts in the $|0\rangle$ initial state. If we wanted to generate different Bell states then all we need to do is change the initial state of the qubits.

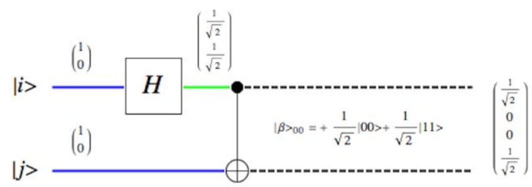


Fig. 9. Generating Bell State

V. RESULT

In figures 5, 6, and 7 we show how qubits 0 and 1 where connected. These circuits were run on IBM's London and Rome quantum computers. Figures 10, 11, 12, 13, 14, and Figure 15 are the resulting histograms from these runs. In the histogram, X-axis represents the classical bit or information and Y-axis shows a probability of the classical bit pattern.

Running circuit 1 on the London and Rome quantum computers gives similar results with probabilities of 91% and 92% respectively. Running circuit 2 also show similar results to each other but degraded from circuit 1. The probabilities are 87% and 88% respectively. The lower probabilities reflect that the two qubits used are separated by one qubit.

The results for circuit 3 show a significant difference between the London and Rome computers at 87% and 75%. The results for the London computer are similar to those for circuit 2. This is what was expected since circuit 2 and circuit 3 both use qubits that are separated by one qubit. The results for Rome are significantly different between circuit 2 and circuit 3. It is believed this is because, on the Rome computer, the qubits are separated by one qubit for circuit 2, while circuit 3 uses qubits that are separated by two qubits.

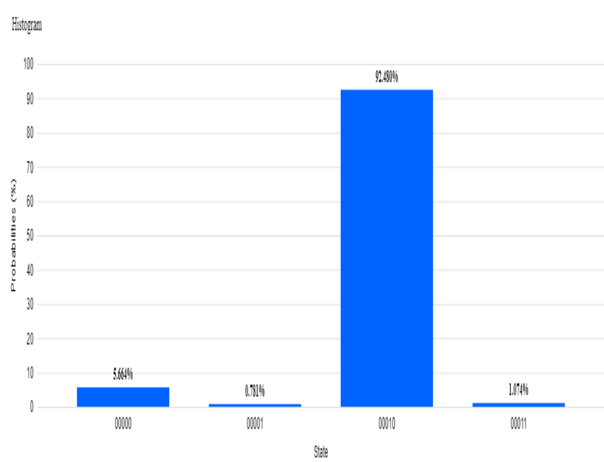


Fig. 10. Result 1 IBM Rome

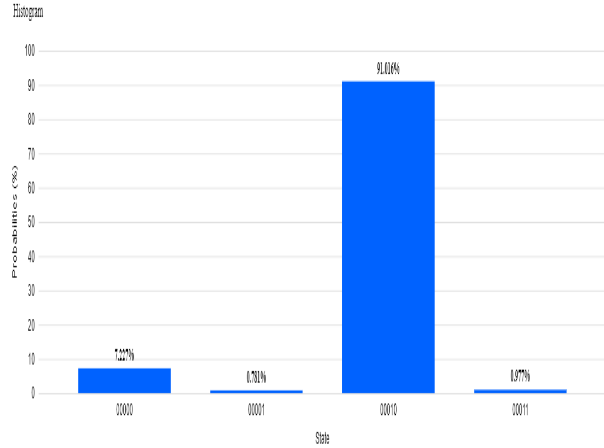


Fig. 11. Result 1 IBM London

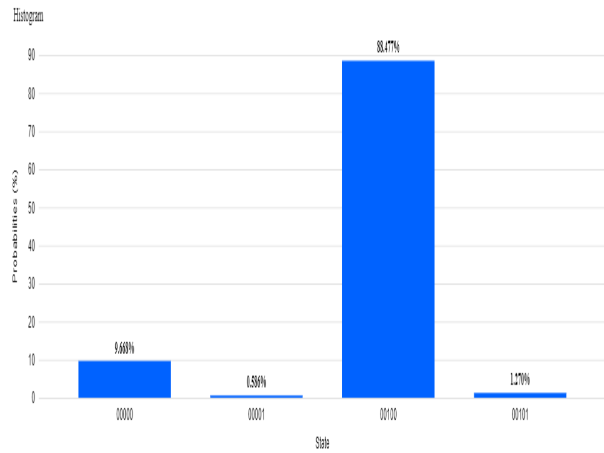


Fig. 12. Result 2 IBM Rome

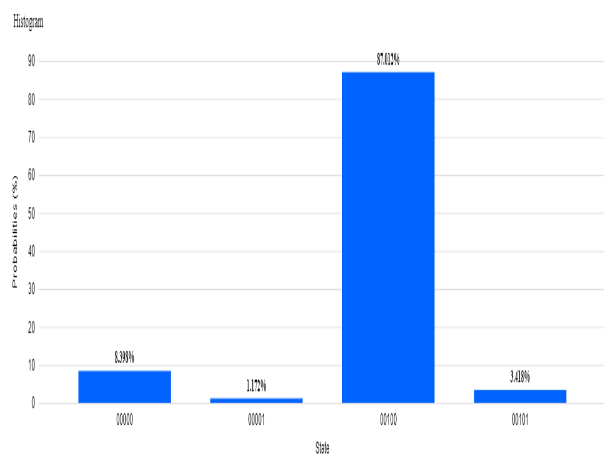


Fig. 13. Result 2 IBM London

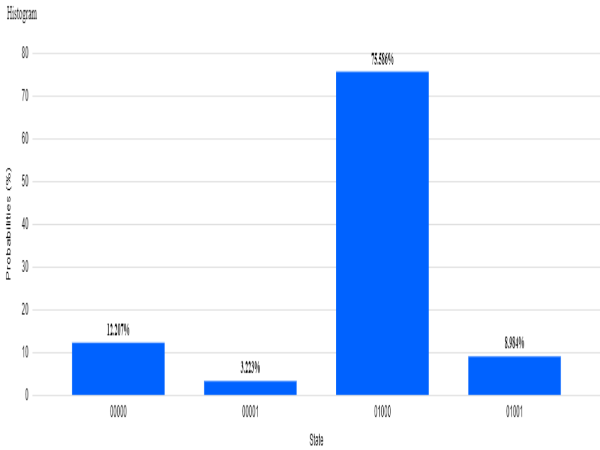


Fig. 14. Result 3 IBM Rome

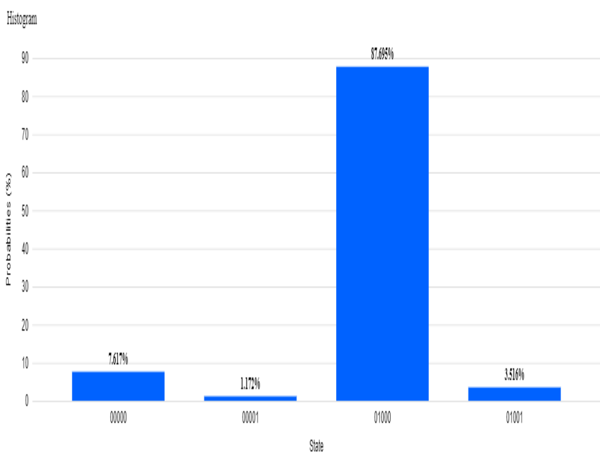


Fig. 15. Result 3 IBM London

Table III shows the resulting probabilities in tabular form. Any errors in the total of the probabilities is due to rounding error.

TABLE III
PROBABILITY RESULTS BY COMPUTER AND CIRCUIT

Circuit	**Computer**	**00**	**01**	**10**	**11**
Circuit 1	London	7.22	0.78	91.01	1.00
	Rome	5.66	0.78	92.48	1.07
Circuit 2	London	8.40	1.17	87.01	3.42
	Rome	9.67	0.59	88.48	1.27
Circuit 3	London	7.62	1.17	87.70	3.52
	Rome	12.20	3.22	75.59	8.98

VI. Conclusion and Future work

The three circuits were constructed and run on the IBM's London and Rome quantum computers. The results show that the accuracy depends on how many qubits are between the qubits that are entangled.

In the future, similar tests should be run on IBM's Yorktown and Melbourne computers to test these different architectures. The tests should also be extended to test the performance of using qubits 0 and 4 as this will further explore the effects of qubit separation.

References

[1] daytonellwanger, "Introduction to quantum computing (11) - the qubit," https://www.youtube.com/watch?v=Co1gWMjVPBI&list=PLIxlJjN2V90w3KBWpELOE7jNQMICxoRwc&index=13, 2017.
[2] I. T. Academy, "How it works: Quantum computing," https://www.youtube.com/watch?v=WVv5OAR4Nik, 2017.
[3] C. G. Almudever, L. Lao, X. Fu, N. Khammassi, I. Ashraf, D. Iorga, S. Varsamopoulos, C. Eichler, A. Wallraff, L. Geck *et al.*, "The engineering challenges in quantum computing," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 836–845.
[4] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2016.
[5] C. H. Bennett and S. J. Wiesner, "Communication via one-and two-particle operators on einstein-podolsky-rosen states," *Physical review letters*, vol. 69, no. 20, p. 2881, 1992.
[6] K. Mattle, H. Weinfurter, P. G. Kwiat, and A. Zeilinger, "Dense coding in experimental quantum communication," *Physical Review Letters*, vol. 76, no. 25, p. 4656, 1996.
[7] J. Gruska, *Quantum computing*. Citeseer, 1999, vol. 2005.
[8] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.
[9] S. P. Gudder, "Book review: Quantum computation and quantum information. by michael a. nielsen and isaac l. chuang. cambridge university press, cambridge, united kingdom, 2000, i–xxv+ 676 pp., (hardcover)," 2001.
[10] *Anaconda, Inc*, 2018, https://www.anaconda.com/.
[11] IBM, *Q Experience*, 2018, https://quantumexperience.ng.bluemix.net/.
[12] *Python Software Foundation*, 2018, https://www.python.org/.
[13] A. W. Cross, L. S. Bishop, J. A. Smolin, and J. M. Gambetta, "Open quantum assembly language," *arXiv preprint arXiv:1707.03429*, 2017.
[14] Steve Atkin, "Demystifying Superdense Coding," 2018. [Online]. Available: https://medium.com/qiskit/demystifying-superdense-coding-41d46401910e
[15] L. Westfall and A. Leider, "Teaching quantum computing," in *Proceedings of the Future Technologies Conference*. Springer, 2018, pp. 63–80.
[16] Ronald Frank PhD, *Linear Algebra Review*, class notes.