Quantum Computing in the Biomedical Sciences; A Brief Introduction into Concepts and Applications

Casper van der Kerk[1], Attila Csala[1] & Aeilko H. Zwinderman[1]

[1] Amsterdam University Medical Center - Clinical Epidemiology, Biostatistics and Bioinformatics

Correspondence: Casper van der Kerk[†], Attila Csala[‡], Department Clinical Epidemiology, Biostatistics and Bioinformatics, AUMC Meibergdreef 9, 1105 AZ, Amsterdam, The Netherlands. E-mail: casper.kerk@gmail.com[†], a@csala.me[‡]

Received: March 29, 2019 Accepted: April 15, 2019 Online Published: July 30, 2019

doi:10.5539/cis.v12n3p104 URL: https://doi.org/10.5539/cis.v12n3p104

Abstract

Quantum computing is a field that aims to exploit the principles of superposition and entanglement to perform computations. By using quantum bits (qubits) a quantum computer is able to perform certain tasks more efficiently when compared to classical computers. While applied quantum computing is still in its early stages, quantum algorithms on simulated quantum computers have already been applied to certain problems in epidemics modeling and image processing. Furthermore, companies like Google and IBM continue to develop new quantum computers with an increasing number of qubits. While much progress has been made in the recent years, the so called ”quantum supremacy” has not yet been achieved, and quantum computing appears to be still unsuitable for most applications in biomedical sciences.

Keywords: Quantum Computing, Healthcare, Biostatistics

1. Introduction

In recent years the speed of data generation has increased rapidly. Such data, also called big data, is being characterized by a rapid acquisition of ubiquitous information. (Lv & et al., 2017) As the acquisition of data is speeding up, so too must the analysis of this data. According to Moore’s law, microprocessor chips have increased significantly in power the last 40-50 years (Moore, 1965). Moore’s law dictates that the amount of transistors on a microprocessor chip will double about every two years. However, Moore’s law is starting to falter (Waldrop, 2016). The microprocessor chips are now reaching the 2-3 nanometer limit, where the features are just 10 atoms across. On this scale the electron behaviour in the microprocessor chip is governed by quantum mechanics. This leads to challenges in engineering due to quantum uncertainties.

However, instead of trying to circumvent these quantum uncertainties, attempts are made to take advantage of them within the framework of Quantum Computing. Quantum computing (QC) is a field that aims to exploit the principles of *superposition* and *entanglement* to perform computations. These computations are conducted via quantum algorithms. An important aspect of these quantum algorithms is, when applied to a certain problem that they can be significantly more efficient than their classical counterparts (de Wolf, 2011).

This article will focus primarily on understanding the basics of QC, as well as some algorithms. Furthermore, we will discuss a few possible applications of QC in a real world examples in applied healthcare data science.

2. Method

2.1 Bits and Qubits

Classical computers work with bits. These bits indicate an OFF or ON state and are represented with a 0 or 1 respectively. These bits are a base-2 numeral system, e.g. 1 in base-2 is 1 in base-10, 10 in base-2 is 2 in base-10, 100 in base-2 is 4 in base-10, etc.

A quantum bit, or qubit for short, is much like a classical bit but also different. Qubits are represented in matrices of the form $a\binom{1}{0} + b\binom{0}{1}$, where $\binom{1}{0}$ represents the $|0\rangle$ qubit and $\binom{0}{1}$ represents the $|1\rangle$ qubit. Another way to look at this is by numbering the rows of the matrix starting a 0 for the first row. A zero has a 1 in the ”zero-row” and a one has a 1 in the ”one-row”. The values of a and b are complex numbers that satisfy $||a||^2 + ||b||^2 = 1$. Because of this rule qubits can take any value between 0 and 1, as long as the sum of the lentgh of their vectors equals to 1. This allows qubits to be in superposition. Note, that classical bits are just special cases of qubits since $\binom{1}{0}$ and $\binom{0}{1}$ fit within this definition. Finally, two or more qubits are presented by the tensor product of their respective matrices. For example, the qubits $|10\rangle$

are represented in equation 1.

$$|10\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \tag{1}$$

2.1.1 Superposition

A qubit in superposition is described by a wave function, which is not a stable state like classical bits can be described. Governed by the rules of quantum mechanics, when a qubit is measured it collapses into a value of either $|0\rangle$ or $|1\rangle$, essentially converting it into a classical bit. When a qubit has a value of $\left(\begin{smallmatrix}a\\b\end{smallmatrix}\right)$ it has a probability of $||a||^2$ to collapse into a $|0\rangle$ and a probability of $||b||^2$ to collapse into a $|1\rangle$. For example, a qubit with the value $\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$ has a probability of $||\frac{1}{\sqrt{2}}||^2 = \frac{1}{2}$ to collapse into either a $|0\rangle$ or a $|1\rangle$ (i.e. a fair coin flip). A qubit with this value is in a *balanced superposition*.

Other examples of qubits being in superposition are represented in equation 2.

$$\begin{pmatrix}\frac{1}{2}\\\frac{\sqrt{3}}{2}\end{pmatrix} \quad \begin{pmatrix}-1\\0\end{pmatrix} \quad \begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\end{pmatrix} \tag{2}$$

Note that a qubit with value $\left(\begin{smallmatrix}0\\1\end{smallmatrix}\right)$ thus has a 100% chance of collapsing into a $|1\rangle$.

2.1.2 Operations on a Single Classical Bit

On a single bit four main operations exist. These operations are represented in 3.

$$\begin{array}{rr} Identity = f(x) = x & Constant0 = f(x) = 0 \\ Negation = f(x) = \neg x & Constant1 = f(x) = 1 \end{array} \tag{3}$$

The identity and negation operations are *variable* operations, as their output is dependent on the input. They are also reversible, i.e. input can be determined from the output. The operations constant-0 and constant-1 are *constant* operations, they will always return the same output regardless of the input. These operations are not reversible.

These four operations will play a central role when we describe the *Deutsch oracle* in section 3.

2.1.3 Dirac Notation

The Dirac notation is often used in quantum mechanics, and is the standard notation in QC. Qubits are represented as $|A\rangle$. This is called a *ket* and its value is equal to A. The notation $\langle B|$ is known as a *bra*, it is the conjugate transpose of a ket. ($\langle B| = |B\rangle^{\dagger}$) These two notations together are known as a *braket*: $\langle B|\,|A\rangle$.

2.2 Quantum Gates

Quantum computing uses gates similar to boolean operation gates in classical computing, called quantum gates. One important aspect of quantum gates, as opposed to classical gates, is that every gate is reversible. This mean every gate and operator in quantum computing has its own inverse. In this section we will discuss some of the more common quantum gates.

2.2.1 Hadamard Gate

Perhaps the most important quantum gate is the Hadamard gate. This gate is used to bring classical bits into a balanced superposition. The Hadamard gate is represented in a quantum circuit by the symbol shown in 4 and is represented by the matrix shown in 5.

$$-\boxed{H}- \tag{4}$$

$$H = \begin{pmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}}\end{pmatrix} \tag{5}$$

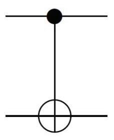
When a Hadamard gate is applied to a classical bit the transformation in 6 takes place, putting the classical bit into exactly equal superposition.

$$
\begin{aligned}
H|0\rangle &= \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \\
H|1\rangle &= \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{pmatrix}
\end{aligned}
\tag{6}
$$

These qubits have equal probability of collapsing into either a $|0\rangle$ or a $|1\rangle$ after measurement.

2.2.2 Controlled NOT-gate

The controlled NOT-gate (CNOT) is a gate similar to the NOT gate in classical computing, it negates the input (i.e. if the input is $|0\rangle$ then the output is $|1\rangle$, and vice versa). The CNOT is a NOT gate on two bits, the first bit is the *control bit* and the second bit is the *target bit*. If the value of the control bit is 0 then the CNOT has no effect. If the value of the control bit is 1 then a NOT-gate is applied to the target bit. The CNOT is represented in a quantum circuit as shown in 7.



(7)

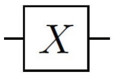
Here the upper full dot is the control bit, and the lower open circle is the target bit. The CNOT gate itself is represented by the matrix in 8.

$$
C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{8}
$$

An example of a CNOT being applied to a pair of bits can be found in Appendix 1.

2.2.3 Pauli X Gate

The Pauli X Gate is better known as the *Bitflip gate X*. This gate negates the input bit. When a 0 bit is the input, a 1 bit will be the output, and vice versa. This gate is essentially the quantum equivalent of a classical NOT gate. In a quantum circuit the Pauli X gate is drawn as shown in 9 and the matrix is represented as seen in 10.

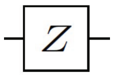


(9)

$$
X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{10}
$$

2.2.4 Pauli Z Gate

The Pauli Z Gate is also called the *Phaseflip gate Z*. This gate places a - in front of $|1\rangle$, flipping the phase of the bit. In a quantum circuit the Pauli Z gate is denoted as shown in 11 and its matrix is represented in 12.



(11)

$$
Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{12}
$$

2.3 Unit Circle State Machine

A useful tool to visualize the impact some of these gates have on qubits is by constructing circle state machines. These are circles with a radius of 1 around the origin. In qubits with the value $\binom{a}{b}$, a is the x-coordinate and b is the y-coordinate on this circle. Two examples of such circle state machines can be seen in figure 1.

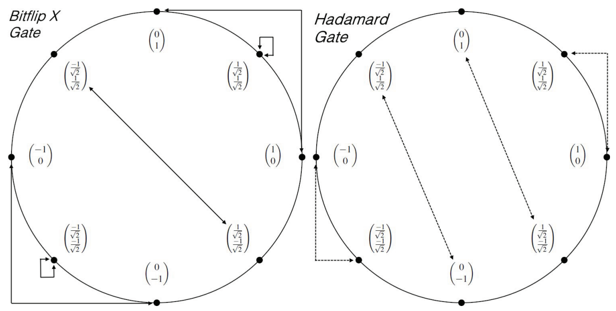
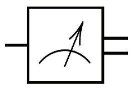


Figure 1. Unit circle state machine for the Bitflip X Gate (left) and the Hadamard Gate (right). Graphs are adapted from Andrew Helwer's Microsoft Research Talk (Helwer, 2018)

2.4 Measurement

At the end of a quantum circuit the qubits of interest are measured. This measurement is often denoted in a quantum circuit as seen in 13.



(13)

Measurement causes the qubits in superposition to collapse into classical bits. The single line on the left of the measurement symbol indicates the qubit and the double line on the right indicates the classical bit. After measurement a qubit with the value $\binom{a}{b}$ collapses into a 0 with a probability of $||a||^2$ or into a 1 with a probability of $||b||^2$.

2.5 Quantum Entanglement

In quantum computing two qubits can be in an *entangled state*. Two qubits are said to be entangled if their product state cannot be factored into two separate bits. An example of such an entanglement is shown in equation 14.

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} \qquad \begin{aligned} ac &= \tfrac{1}{\sqrt{2}} \\ ad &= 0 \\ bc &= 0 \\ bd &= \tfrac{1}{\sqrt{2}} \end{aligned} \tag{14}$$

In this example, if *ad* were indeed equal to 0 then *ac* and *bd* should also be equal to 0, which is not true. The same goes for *bc*. A 2-qubit state like this is called an *EPR-pair* named after Einstein, Podolsky, and Rosen who first described this state in 1935 (Einstein et al., 1935).

Entangled qubits hold no value on their own. However, they can still be measured. When the first qubit is measured and returns $|0\rangle$, then the second qubit automatically collapses into $|0\rangle$ as well. The same is true if $|1\rangle$ is measured. This collapse of the second qubit happens instantaneously when the first qubit is measured, even if the second qubit is unobserved. Moreover, when two qubits are entangled they can be physically separated over great distances while still maintaining this entanglement effect (Boone et al., 2015). A more detailed description of how two qubits can be entangled can be reviewed in appendix 3.

Entanglement is paramount in certain applications of quantum computing, one of these is quantum teleportation (Gisin, 2017).

2.6 Quantum Algorithms

The above concepts are used in quantum algorithms. Here we describe two of these algorithms in more detail.

2.6.1 The Deutsch Oracle

The Deutsch oracle is one of the first quantum algorithms and was first described by David Deutsch in 1985 (Deutsch, 1985). It is used to determine whether a function is variable or constant by observing only the input and output. The Deutsch oracle can therefore be described as a black box. Within this black box one of the four classical bit operations takes place on one bit. It is not known which functions are inside the black box, only the input and the output can be observed. While it still requires two queries on both a classical computer and a quantum computer to exactly pinpoint which function is inside the black box, it will take a quantum computer only a single query to determine if the function is variable or constant, whereas this is O(2^N) queries for a classical computer (where N denotes the size of the input).

2.6.1.1 The Black Box

On the classical computer the black box has a single input and a single output. However, this is insufficient if non-reversible functions must be written in a reversible way. Every function in QC must be reversible. Therefore, an extra output qubit is added to which the function is applied. The circuit representation of a black-box looks like this:

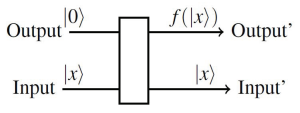


Figure 2. The Black Box problem in a quantum circuit

There are two important things to note here. First is that the output wire will always be initialized to $|0\rangle$. The second is that the black box leaves the input qubit unchanged, its value remains $|x\rangle$ at the input' wire. Instead the result of the black box is written to the output qubit and read at the output' wire.

2.6.1.2 Classical Operations in a Quantum System

The first operations are the *constant-0* and the *constant-1* operations. The output qubit is initialized as a $|0\rangle$. Therefore, the constant-0 operation will have no effect on whatever value we put into the black box. So there will be no operations on the qubit. Likewise, the constant-1 operation will always return the opposite of the starting state of the output qubit. This is the same as performing a bitflip (Pauli X Gate) on the output qubit (see Figure 3).

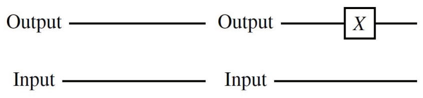


Figure 3. Constant-0 (left) and Constant-1 (right)

The other two operations are *identity* and *negation*. For the identity operations the input is equal to the output, which is $|x\rangle$ in this case. This is the same as applying a CNOT to the two qubits in our black box. If the input is $|0\rangle$, then no operations need to be applied to the output qubit since it was already initialized as a $|0\rangle$. If the input is $|1\rangle$ then the output qubit needs to be flipped. Likewise, if the output need to be the opposite of the input (negation) the output qubit need to be flipped after the CNOT gate has been applied (see Figure 4).

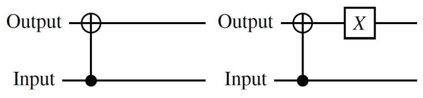


Figure 4. Identity (left) and Negation (right)

2.6.1.3 The Oracle

The circuit representation of the Deutsch Oracle looks as follows;

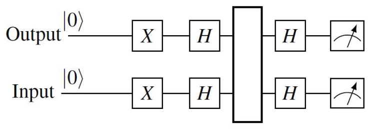


Figure 5. The Deutsch Oracle

With this oracle the following two statements can be made:

- *If the function inside the black box is constant, the system will be in the state* $|11\rangle$ *after measurement.*
- *If the function inside the black box is variable, the system will be in the state* $|01\rangle$ *after measurement.*

These statements can be proved by using the circle state machines from figure 1. Before the qubits enter the black box they both pass through a bitflip gate followed by a Hadamard gate. Following the corresponding lines in the circle state machines it can be observed that the qubits have gone through the states shown in 15 before the black box.

$$\begin{pmatrix}1\\0\end{pmatrix} - > \begin{pmatrix}0\\1\end{pmatrix} - > \begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix} \tag{15}$$

By looking at the circle state machines in figure 1 it can deduced that applying a bitflip and a Hadamard gate to a qubit with value ($\frac{1}{\sqrt{2}} \atop \frac{-1}{\sqrt{2}}$) does not affect its value after measurement. Therefore, the output qubit has value ($0 \atop 1$) if no bitflip is applied, and value ($0 \atop -1$) if a bitflip is applied. Regardless, after measurement the output qubit in a constant operation returns $|1\rangle$, and since the input qubit always remains untouched the final state is $|11\rangle$.

The difference between the constant and the variable circuit is the application of the CNOT gate. A CNOT applied to two qubit in superposition flips the target bit. In this case the transition shown in equation 16 happens.

$$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\end{pmatrix} - > \begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix} \tag{16}$$

This transition is described in more detail in appendix 2. Once again by looking at the circle state machines it can be observed that a Hadamard gate applied to a qubit with value ($\frac{1}{\sqrt{2}} \atop \frac{1}{\sqrt{2}}$) returns $|0\rangle$ after measurement. Therefore both the variable operations return $|01\rangle$ after the black box.

2.6.2 Grover's Search Algorithm

Grover's algorithm (GA) is used to find an item with a specific property, that satisfies a given condition, in an unstructured list with N items (Grover, 1997). A classical algorithm would perform O(N/2) on average, and O(N) in the worst case to find the marked item. The Grover's algorithm would find this marked item by checking approximately $\sqrt{N}$ items. GA has two parts; the *Oracle* and the *Amplitude amplification*.

2.6.2.1 The Oracle

The oracle is able to flip the amplitude of the marked item with property w. First the list of items is put in superposition, meaning all items have an equal amplitude ($\frac{1}{\sqrt{N}}$). Then a function is used that returns 1 for an item with property w and 0 otherwise. This function is described in 17.

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle \tag{17}$$

The function of this oracle does nothing to an item with property x, but an item with property $|w\rangle$ gets mapped to $U_f |w\rangle = -|w\rangle$. The marked item now has a negative amplitude.

2.6.2.2 Amplitude Amplification

In the amplitude amplification an inversion about the mean of the amplitudes is applied. This implies that every item that was below the mean gets flipped by the same amount above the mean. Since the marked item had a negative amplitude it gets increased by a greater amount than all the other items.

The oracle and the amplitude amplification are repeated approximately $\sqrt{N}$ times. The amplitude of the marked item $|w\rangle$ increases linearly with the number of applications of the algorithm until it is certain which of the items is the marked item.

3. Results

3.1 Application in Epidemics

In 2007 A. León and J. Pozo published an article in which they utilized quantum computing in an epidemic study (León & Pozo, 2007). In this study they predicted the infection rates and the spread of the HIV virus in Chile between 1986 and 2000. They did this by encoding each individual, in a population of N individuals, with a single qubit. If a person was infected with HIV they would be encoded as a 1, and if they were uninfected they were encoded with a 0. However, if it was unclear whether a person was infected they were encoded by a qubit in superposition, meaning that this individual had a value of 0 and 1 at the same time. Next, they ran all the qubits through a set of quantum gates depending on the sexual activity of those individuals. After measurement the qubits that were in superposition would collapse to either a $|0\rangle$ or a $|1\rangle$ state.

Using this method the authors were able to validate the model using the available data of HIV cases. However, projecting the number of people infected with asymptomatic HIV showed some significant differences when compared to the official predictions. In certain cases the model projected around 5904 cases of infection whereas the official predictions projected around 2618 cases. Therefore, this model need some further adjustments.

The missing infection states in this example are an obvious case to apply quantum computing concepts since missing outcome data occurs in many biomedical studies. Often patients with missing outcome data are discarded from the analysis and this quantum computing application may be an attractive alternative.

3.2 Application in Image Processing

In image processing meta-heuristic genetic algorithms have been improved by using quantum principles. A genetic algorithm (GA) is a class of *evolutionary algorithms*. They are commonly used to generate high-quality solutions to optimization and search problems. In 2011 Siddhartha Bhattacharyya and Sandip Dey published a paper in which they proposed a quantum inspired genetic algorithm (QIGA) (Bhattacharyya & Dey, 2011). With this algorithm they were able to determine the optimum threshold intensities of two real life gray level images. This image thresholding is an effective technique for segregating the foreground information from the background. When the performance of the QIGA was compared to the performance of the GA it was evident that the QIGA outperformed the classical GA. QIGA completed its calculations in about 2554 to 2666 seconds, whereas the GA required 6559 to 6384 seconds. This improvement was attributed, by the authors, to the inherent parallelism induced by the QIGA.

In 2014 Siddhartha Bhattacharyya and Sandip Dey published a follow-up paper in which they further improved on their QIGA and also introduced a quantum inspired particle swarm optimization for bi-level thresholding (QIPSO) (Dey et al., 2014). Again, the experiments showed that the quantum inspired meta-heuristic algorithms were more accurate and robust than their classical counterparts. In general, quantum computing was able to reduce the time complexity required in image thresholding when compared to the classical algorithms. The authors of these articles are currently investigating the possibility of using quantum-inspired meta-heuristic algorithms for pure and true color images.

Further improvement on using QC in image processing has been made by Li *et al* (Li & Zhang, 2014). In their article published in 2014 they used four different quantum algorithms to improve palm-print identification. By using these algorithms they were able to get a better filtering result by using a quantum filtering algorithm, speed-up feature extraction exponentially by using quantum Fourier transform, and speed-up palm-print matching quadratically by using quantum set operations and Grover's algorithm. Additionally, the matching accuracy increased to almost 100%.

This set of algorithms may in the future be used for other types of image processing like tumor screening or identifying hairline cracks in bone structures.

3.3 Potential New Applications

The following two applications are ideas that we consider potentially interesting problems to be solved by QC. While they have not been realized yet, these applications could utilize QC in healthcare.

3.3.1 Finding Errors in Genome Sequencing

In genomics it is possible that a nucleotide in a gene is sequenced incorrectly. To check whether this is the case a classical algorithm has to test each nucleotide against a reference template. However, in QC the *Deutsch-Jozsa* algorithm can be used. This is a deterministic algorithm, relying on the Deutsch oracle, it returns $|0\rangle$ if the output is constant, meaning all the outputs are 0 or all outputs are 1, and it returns a $|1\rangle$ is the output is variable, meaning half the outputs are 0 and half the outputs are 1.

All four possible nucleotides can be encoded by two qubits. The newly sequenced gene can then be encoded and compared to a template of that same gene which is known to be correct. When these two encoded strings are added with a modulo

2 the resulting value will be either 0, if the two nucleotides are equal, not 1 if they differ. The Deutsch-Jozsa algorithm can then be used to determine in a single query whether the values are constant (i.e. all values are 0). If the values are constant, then the sequenced gene contains no errors.

3.3.2 Localization of a Codon

Another application of QC in genomics could be the localization of a codon in a genetic sequence using, Grover's algorithm. If the four different nucleotides can be encoded by two qubits, then a codon can be encoded by six qubits. Grover's algorithm can then search for this specific qubit sequence in a gene or genome. For example, the codon 'ATG' needs to be found. These nucleotides could then be encoded as $|00\rangle + |01\rangle + |10\rangle$. The Grover's algorithm can then search where this specific sequence of six qubits are in the gene or genome.

To demonstrate this application a three qubit oracle was constructed as a proof of concept, based on an elaboration by the QC community and translated to R by the authors (Bick, 2018). This R code can be used to successfully mark a three qubit sequence which can then be found with high probability using the Grover's algorithm. The constructed three qubit oracle can be found in Appendix 4.

4. Discussion

Future of Quantum Computing

As discussed in section 4 of this article there already exists a number of applications for quantum computing in the biomedical research field, and many more applications are envisioned that have not been realized yet. In the U.S. around \$250 million is currently spent per year on quantum computing, and in 2018 the U.S. Department of Energy invested another \$40 million to further develop useful quantum algorithms in the field of chemistry, materials science, nuclear physics, and particle physics (Cho, 2018).

In 2018 Boixo *et al.* published an article in *Nature Physics* in which they propose a benchmark for tasks that could demonstrate *quantum supremacy* (Boixo et al., 2018). Quantum supremacy is achieved when a quantum device can perform well-defined computational tasks, without error correction, that cannot be performed by any known algorithms on a classical supercomputer in a reasonable amount of time. Furthermore, they postulate that quantum supremacy can be achieved with circuits in a two-dimensional lattice of 7x7 qubits.

In March of 2018 Google announced the superconducting Bristlecone chip (Terhal, 2018). This chip consists of a two-dimensional array of 72 superconducting qubits, which can be utilized to achieve quantum supremacy according to Boixo *et al.* Currently Google is looking to achieve the similar performance of their 9-qubit device in their 72 qubits Bristlecone (Kelly, 2018).

IBM launched their IBM Q Experience platform in 2016 (room, 2016). With it users can already freely experiment with a five qubit quantum computer. Recently, IBM introduced an universal 16 qubit quantum computer, which could be fully entangled, which is important for various quantum algorithms and super-dense coding (Wang et al., 2018).

These are just a few examples of numerous advancements made in the field of quantum computing. Considering these developments quantum computing appears to have a rapidly developing research field. However, QC is not an all powerful technology, certain problems cannot be sped up by QC. For example, James Amundson, a computational physicist, claims that QC can't speed up the analysis process of the Large Hadron Collider in Switzerland (Cho, 2018). Furthermore, all the algorithms discussed in section 4 were running on quantum simulations as opposed to running on actual quantum computers. This is further indication that applied quantum computing is still in its early stages. More development is necessary for QC to be more accessible than quantum simulation. Additionally, quantum-based approaches have yet to outperform classical heuristics (Mandrà & Katzgraber, 2017).

In conclusion, quantum computing is an active, rapidly developing research area that provides many interesting potential applications. Some of these applications have already been proven to be effective in image processing and epidemics, while other quantum-based solutions have yet to outperform classical approaches. With new developments on the horizon from companies like Google and IBM the quantum supremacy appears to be within reach. However, quantum computing will not be able to make all computations faster, and certain problems will be kept for classical computers.

Acknowledgements

CK conducted the literature study and drafted the manuscript. AC supervised the literature study and provided feedback on the drafts. AZ provided final critical feedback of the manuscript. All authors gave final approval for publication.

References

Bhattacharyya, S., & Dey, S. (2011). *An efficient quantum inspired genetic algorithm with chaotic map model based interference and fuzzy objective function for gray level image thresholding.* International Conference on Computational Intelligence and Communication Systems. https://doi.org/10.1109/CICN.2011.24.

Bick. (2018). Implementation of the oracle of grovers algorithm on ibm q using three qubits. URL https://quantumcomputing.stackexchange.com/questions/2200/implementation-of-the-oracle-of- grovers-algorithm-on-ibm-q-using-three-qubits.

Boixo, S., Isakov, S. V., Smelyanskiy, V. N., Babbush, R., Ding, N., & Jiang, Z. (2018). Characterizing quantum supremacy in near-term devices. *Nature Physics, 14*, 595C600. https://doi.org/ 10.1038/s41567-018-0124-x.

Boone, K., Bourgoin, J. P., Meyer-Scott, E., Heshami, K., Jennewein, T., & Simon, C. (2015). Entanglement over global distances via quantum repeaters with satellite links. *Physical Review, 91*. https://doi.org/ 10.1103/PhysRevA.91.052325.

Cho, A. (2018). Doe pushes for useful quantum computing. *Science, 359*, 141C142. https://doi.org/10.1126/science.359.6372.141.

Deutsch, D. (1985). Quantum theory, the church-turing principle and the universal quantum computer. Proceedings of the Royal Society of London. *Series A, Mathematical and Physical Sciences, 400*(1818), 97C117. https://doi.org/10.1098/rspa.1985.0070.

Dey, S., Bhattacharyya, S., & Ijjwal, M. (2014). *Quantum inspired genetic algorithm and particle swarm optimization using chaotic map model based interference for gray level image thresholding.* Swarm and Evolutionary Computation. https://doi.org/10.1016/j.swevo.2013.11.002.

Einstein, A., Podolsky, B., & Rosen, N. (1935). Can quantum-mechanical description of physical reality be considered complete? *Physical Review, 47*, 777C780. https://doi.org/10.1103/PhysRev.47.777.

Gisin, N. (2017). *Quantum-teleportation experiments turn 20.* Nature, pages 42C43. https://doi.org/10.1038/d41586-017-07689-5.

Grover, L. K. (1997). Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*. https://doi.org/10.1103/PhysRevLett.79.325.

Helwer, A. (2018). *Quantum computing for computer scientists*. URL https://www.microsoft.com/en-us/research/video/quantum-computing-computer-scientists/.

IBM News room. (2016). Ibm makes quantum computing available on ibm cloud to accelerate innovation. URL https://www-03.ibm.com/press/us/en/pressrelease/49661.wss.

Kelly, J. (2018). A preview of bristlecone, googles new quantum processor. URL https://ai.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html.

Leon, A., & Pozo, J. (2007). *Model based on a quantum algorithm to study the evolution of an epidemics*. Computers in Biology and Medicine, 2007. https://doi.org/10.1016/j.compbiomed.2006.03.005.

Li, H., & Zhang, Z. (2014). *Research on palmprint identification method based on quantum algorithms.* The Scientific World Journal. https://doi.org/10.1155/2014/670328.

Mandrà, S., & Katzgraber, H. G. (2017). A deceptive step towards quantum speedup detection. *IOP Science*. https://doi.org/10.1088/2058-9565/aac8b2.

Moore, G. E. (1965). Cramming more components onto integrated circuits. *Electronics, 38*(8). https://doi.org/10.1109/NSSC.2006.4785860.

Terhal, B. M. (2018). Quantum supremacy, here we come. *Nature Physics, 14*, 530C531. https://doi.org/10.1038/s41567-018-0131-y.

Wang, Y., Li, Y., Yin, Z., & Zeng, B. (2018). 16-qubit ibm universal quantum computer can be fully entangled. *Cornell University Library*. https://doi.org/10.1038/s41534-018-0095-x.

Waldrop, M. M. (2016). More than moore. *Nature, 530*. https://doi.org/10.1038/530144a.

Wolf, R. de. (2011). Quantum computing: Lecture notes. URL https://homepages.cwi.nl/ rdewolf/qcnotes.pdf.

Zhihan, L., Houbing, S., Basanta-Val, P., Steed, A., & Minho, J. (2017). Next-generation big data analytics: State of the art, challenges, and future research topics. *IEEE Transactions on Industrial Informatics, 13*(4), 1891C1899. https://doi.org/10.1109/TII.2017.2650204.

1. CNOT Gate

$$C\,|10\rangle = C\left(\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |11\rangle$$

$$C\,|01\rangle = C\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |01\rangle$$

2. CNOT on qubits in superposition

$$C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\end{pmatrix}\right) = C\begin{pmatrix}\frac{1}{2}\\\frac{-1}{2}\\\frac{-1}{2}\\\frac{1}{2}\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\-1\\-1\\1\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\end{pmatrix}$$

3. Quantum Entanglement

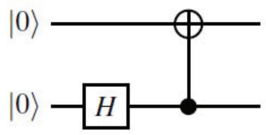


Figure 6. A quantum circuit for producing entangled qubits

$$C\left(H\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\\frac{1}{\sqrt{2}}\\0\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\0\\\frac{1}{\sqrt{2}}\end{pmatrix}$$

Figure 7. The mathematics behind the entanglement of qubits

4. Three Qubit Oracle

```
library(Matrix)
i <- complex(real = 0, imaginary = 1)

XX_gate <- function(input1){

   output_1 <- diag(x = 4)*cos(input1)

   v1 <- c(4,3,2,1)
   m1 <- sparseMatrix(seq_along(v1), v1, x=1)
   m1 <- as.matrix(m1)
   m1*-i*sin(input1)

   output_1 <- output_1+m1

   return(output_1)

}
R_gate <- function(theta, phi){

   output_1 <- diag(x = 2)*cos(theta/2)

   v1 <- c(2,1)
   m1 <- sparseMatrix(seq_along(v1), v1, x=1)
   m1 <- as.matrix(m1)
   m1* (-i*exp(1)^(i*phi) * sin(theta/2))

   output_1 <- output_1+m1

   return(output_1)

}
Rx_gate <- R_gate(theta, phi=0)
Ry_gate <- R_gate(theta, phi= (pi/2))


#START

a = TensorProd(TensorProd(Hadamard(I2),Hadamard(I2)),Hadamard(I2))
# 1st CNOT
a1= CNOT3_12(a)
# 2nd composite
# I x I x T1Gate
b = TensorProd(TensorProd(I2,I2),T1Gate(I2))
b1 = DotProduct(b,a1)
c = CNOT3_02(b1)

# 3rd composite
# I x I x TGate
d = TensorProd(TensorProd(I2,I2),TGate(I2))
d1 = DotProduct(d,c)
e = CNOT3_12(d1)

# 4th composite
# I x I x T1Gate
f = TensorProd(TensorProd(I2,I2),T1Gate(I2))
```

```
f1 = DotProduct(f,e)
g = CNOT3_02(f1)

#5th composite
# I x T x T
h = TensorProd(TensorProd(I2,TGate(I2)),TGate(I2))
h1 = DotProduct(h,g)
i = CNOT3_01(h1)

#6th composite
j = TensorProd(TensorProd(I2,T1Gate(I2)),I2)
j1 = DotProduct(j,i)
k = CNOT3_01(j1)

#7th composite
l = TensorProd(TensorProd(TGate(I2),I2),I2)
l1 = DotProduct(l,k)

#8th composite
n = TensorProd(TensorProd(Hadamard(I2),Hadamard(I2)),Hadamard(I2))
n1 = DotProduct(n,l1)
n2 = TensorProd(TensorProd(PauliX(I2),PauliX(I2)),PauliX(I2))
a = DotProduct(n2,n1)

#repeat the same from 2st not gate

a1= CNOT3_12(a)
# 2nd composite
# I x I x T1Gate
b = TensorProd(TensorProd(I2,I2),T1Gate(I2))
b1 = DotProduct(b,a1)
c = CNOT3_02(b1)

# 3rd composite
# I x I x TGate
d = TensorProd(TensorProd(I2,I2),TGate(I2))
d1 = DotProduct(d,c)
e = CNOT3_12(d1)

# 4th composite
# I x I x T1Gate
f = TensorProd(TensorProd(I2,I2),T1Gate(I2))
f1 = DotProduct(f,e)
g = CNOT3_02(f1)

#5th composite
# I x T x T
h = TensorProd(TensorProd(I2,TGate(I2)),TGate(I2))
h1 = DotProduct(h,g)
i = CNOT3_01(h1)

#6th composite
j = TensorProd(TensorProd(I2,T1Gate(I2)),I2)
j1 = DotProduct(j,i)
k = CNOT3_01(j1)

#7th composite
```

```
l = TensorProd(TensorProd(TGate(I2),I2),I2)
l1 = DotProduct(l,k)

#8th composite
n = TensorProd(TensorProd(PauliX(I2),PauliX(I2)),PauliX(I2))
n1 = DotProduct(n,l1)
n2 = TensorProd(TensorProd(Hadamard(I2),Hadamard(I2)),Hadamard(I2))
n3 = DotProduct(n2,n1)


result=measurement(n3)
plotMeasurement(result)
```

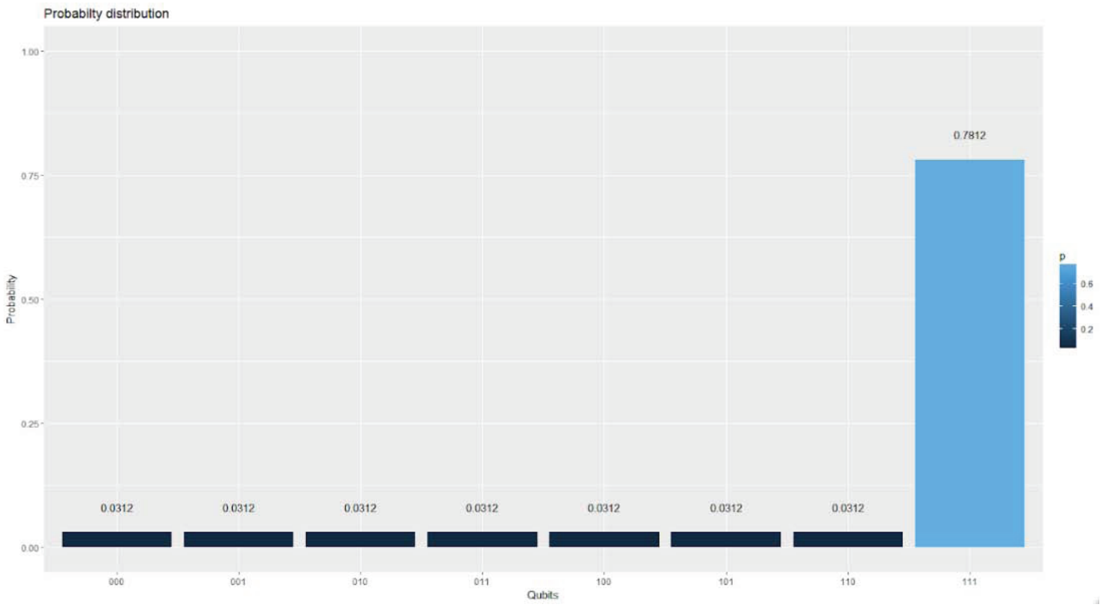


Figure 8. Result of the three qubit oracle

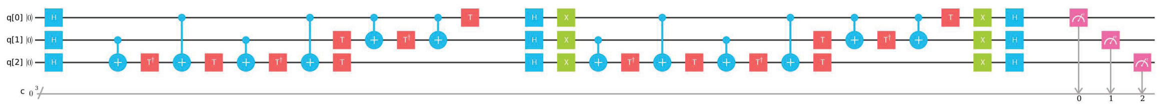


Figure 9. Visual representation of the three qubit oracle (Bick, 2018)

Copyrights

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/4.0/).