# A Brief Overview of Quantum Algorithm

Mohd Harith Akmal Zulfaizal Fadillah, Bahari Idrus,
Mohammad Khatim Hasan

*Centre for Artificial Intelligence Technology (CAIT),*
*Faculty of Information Science & Technology,*
*Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor*

**Abstract**

Quantum computing and quantum algorithms have been said to be able to provide significant speed up over classical computation with classical algorithms by exploiting the nature of quantum mechanical systems. We will go through briefly the prerequisite in understanding quantum algorithm before getting to know more about existing quantum algorithms. This report will look at some of the most famous algorithms – Grover's searching algorithm, Shor's period-finding algorithm and Harrow-Hassidim-LLoyd (HHL) algorithm for quantum linear systems problems.



## 1 Introduction

Quantum computing is not necessarily something new. It has been stipulated since the late 1970s by none other than Richard Feynman, where he deduced that quantum systems cannot be simulated reliably by classical computers without a trade off in its run time [1]. He speculated that a computer that obeys the law of quantum mechanics will be able to break this trade off and offer exponentially more powerful and reliable.

Classical computation infrastructures are dictated by the laws of classical physics which grows exponentially in power and capability every two years as per Moore's law [2]. A bottleneck of classical computational power is bound to happen as transistors and microchips get smaller and the growth will eventually stop as it reaches quantum scale where physics do not behave in the same way. Quantum computing could be the next alternative for a computation infrastructure where it is fundamentally built with quantum mechanical properties which can be taken advantage of.

Ever since the idea of quantum computing has been brought up, many physicists and computer scientist have tried taking a shot at cracking this enigma and came up with potential hardware configurations and quantum algorithms. These algorithms take advantage of quantum mechanical properties to provide speed up as opposed to their classical counterparts. 50 years have passed since Feynman's conception and the field of quantum computing has never been bigger, and this is partly due to the fact that we are now seeing actual reliable quantum computers being developed and are able to compute these theoretical quantum algorithms on actual quantum machines.

In this report, we will look briefly at the fundamentals of quantum computing needed to be able to construct a quantum algorithm circuit. We will also look at some of the most commonly known quantum algorithms such as the Grover's algorithm, Shor's algorithm and HHL algorithm for solving quantum linear system problems. This study will be the basis for our project *"Implementing quantum algorithm for least squares data fitting through IBM-Q"*.

# 2 Fundamentals of Quantum Computing

Quantum bits, like their classical counterparts, are the information representation for quantum systems. The smallest two-level device, the simplest quantum states system capable of showing quantum mechanical properties, is referred to as a qubit. Classical bits are discrete, consisting of 0s and 1s, while qubits are continuous, allowing them to exist in the states of $|0\rangle$ and $|1\rangle$ or in the superposition of the two. Qubit can be described by the Bloch sphere which is a geometrical representation of the pure state space of a two-level quantum system [3].

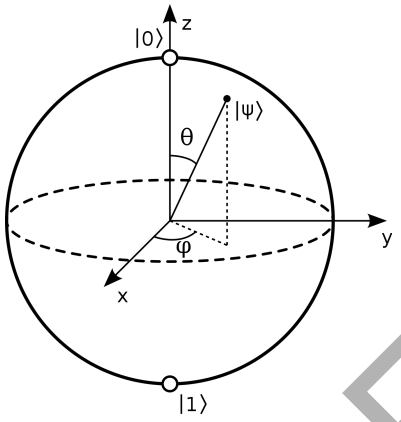



Figure 1: Bloch sphere

A single qubit can be represented in the basis of $|0\rangle$ and $|1\rangle$ such that the state of the two-dimensional qubit can be expressed as the linear combination of both bases. The quantum state then depends on the values of $a$ and $b$ to determine whether its in the computational basis state or a superposition between them,

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle \qquad \text{such that } |a|^2 + |b|^2 = 1$$

where $a$ and $b$ are the amplitude of the states. Measuring the qubit with quantum state $|\psi\rangle$ will collapse the wavefunction and yield the classical value of 0 with probability $|a^2|$ or 1 with probability $|b^2|$. Multiple qubits can be represented by taking the tensor product of multiple single qubits,

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1\begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0\begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$|01\rangle = |0\rangle \otimes |1\rangle \begin{pmatrix} 1\begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0\begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

One of the quantum mechanical properties that makes quantum computing beneficial is superposition. Classically as seen with the state vector $|\psi\rangle$, the state could be either $|0\rangle$ or $|1\rangle$, but by utilizing superposition, there exist a state where it could be either of the two and it cannot be definitely identified until measured. For example,

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

this is a superposition state with equal amplitude of $1/\sqrt{2}$ for both of the states. This property of superposition allows for quantum parallelism which provides exponential speedup.

Another quantum mechanical property that is of use in quantum computation and information is entanglement. An entangled system is inseparable such that it cannot be factored as the tensor product of its individual subsystems [4]. Entanglement is a physical phenomenon that affects the measurement of the system as a whole even when they are separated by a large distance. When an entangled qubit is measured, the wave function of the other entangled qubit also instantaneously collapses and provide a measurement that satisfies the entangled states. For example, say we have the entangled state known as the Bell state,

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

When the first qubit is measured to be $|0\rangle$, the second qubit that is measured at exactly the same time also yield a measurement of $|0\rangle$ which initially puzzled physicists as they presumed that there was some kind of faster-than-light transmission phenomenon interacting between those particles, which is not exactly true. The entangled state is essentially linked and correlated together such that measurement towards one of them collapses the other and this is the property that can be taken advantage of in quantum computation and information. Some use of quantum entanglement includes quantum teleportation and quantum superdense coding among others.

Quantum mechanics dictates such that the evolution of any quantum state is restricted to its property of unitarity represented by a unitary operator that is the sum of probabilities of all possible states is exactly 1. The unitary operator is also inner product and norm preserving. For a quantum computing machine, the algorithms and logic gates that construct the machine must therefore also be unitary, and thus reversible where the information is not lost such that the input can be yielded from the output. The reversibility of a unitary operator, $U$ can be shown where $V$ is a vector,

$$UU^\dagger = U^\dagger U = I$$

$$V \to UV \to U^\dagger UV \to IV \to V$$

by applying the conjugate transpose, $U^\dagger$ to the vector $V$ applied with $U$, the initial vector $V$ can be obtained. Quantum computational algorithms and quantum gates will have to obey this principle where they should be able to yield the input once the operations are reversed.

## 2.1 Quantum Gates

As reversibility is an important property of a quantum computer, the logic gates used in this machine differs from the conventional classical logic gates. Quantum logic gates must be unitary reversible while classical gates do not necessarily have to be. The classical NOT gate is reversible as we can retain the input information from the output but classical gates such as AND, NAND, OR, NOR and XOR are non-reversible [3]. There are two types of quantum gates: single qubit gates which act on single qubit and multi-qubit gates which act on multiple qubits. We will first look into single qubit gates [4],

i. **Hadamard gate** maps the basis state to a superposition state.

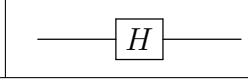
| | Gate notation | Matrix representation |
|---|---|---|
| *Hadamard* gate |  | $\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$ |

Table 1: Hadamard gate

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

ii. **Pauli-X gate** is the quantum equivalent of a NOT gate that rotates the basis state around the X-axis of the Bloch sphere by $\pi$ radians.

| | Gate notation | Matrix representation |
|---|---|---|
| *Pauli-X* gate | —[X]— | $\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ |

Table 2: Pauli-X gate

$$X|0\rangle = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} = |1\rangle$$
$$X|1\rangle = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} = |0\rangle$$

iii. **Pauli-Y gate** rotates the basis state around the Y-axis of the Bloch sphere by $\pi$ radians mapping $|0\rangle$ to $i|1\rangle$ and $|1\rangle$ to $-i|0\rangle$.

| | Gate notation | Matrix representation |
|---|---|---|
| *Pauli-Y* gate | —[Y]— | $\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}$ |

Table 3: Pauli-Y gate

$$Y|0\rangle = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\i\end{pmatrix} = i|1\rangle$$
$$Y|1\rangle = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}-i\\0\end{pmatrix} = -i|0\rangle$$

iv. **Pauli-Z gate** is a phase flip gate that rotates the basis state around the Z-axis of the Bloch sphere by $\pi$ radians that only flips the phase of $|1\rangle$ to $-|1\rangle$ and remains unchanged when applied to $|0\rangle$.

| | Gate notation | Matrix representation |
|---|---|---|
| *Pauli-Z* gate | —[Z]— | $\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$ |

Table 4: Pauli-Z gate

$$Z|0\rangle = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} = |0\rangle$$

$$Z|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -|1\rangle$$

v. **Phase shift ($R_\phi$) gates** map the basis state of $|0\rangle$ to $e^{i\phi}|1\rangle$. The Pauli-Z gate is a phase shift gate where $\phi = \pi$. Some other examples of phase shift gates are the S and T gates where $\phi = \pi/2$ and $\phi = \pi/4$ respectively.

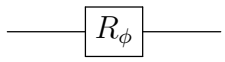
| | Gate notation | Matrix representation |
|---|---|---|
| $R_\phi$ gate |  | $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$ |

Table 5: Phase shift, $R_\phi$ gate

$$S = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

$$S|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

$$S|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ i \end{pmatrix} = i|1\rangle$$

The multi-qubits gates are usually acted upon two (binary) to three (tenary) different qubits. In general, the size of the matrix gate operator is in the form of $2^n \times 2^n$ where $n$ is the number of qubits. Some example of the most commonly used multi-qubit quantum gates are [4],

i. **CNOT/CX gate** is the controlled NOT gate which acts on 2 qubits performing the NOT operation to the target qubit when the control qubit is $|1\rangle$. The first qubit is the control qubit whereas the second is the target qubit.

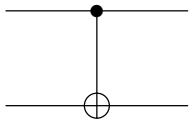
| | Gate notation | Matrix representation |
|---|---|---|
| *Controlled-NOT* gate |  | $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ |

Table 6: CNOT/CX gate

$$CX|10\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle$$

$$CX|11\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

$$|00\rangle \mapsto |00\rangle \qquad |10\rangle \mapsto |11\rangle$$
$$|01\rangle \mapsto |10\rangle \qquad |11\rangle \mapsto |10\rangle$$

ii. **Controlled-U gate** is a general controlled gate which acts on 2 qubits where the matrix $U$ can be either the single qubit gate X, Y, Z or the phase shift matrix. When the $U = X$, the gate becomes the CNOT/CX gate. For example if $U = R_\phi$, the controlled phase shift gate can be represented as,

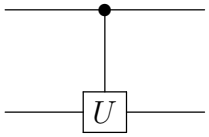
| | Gate notation | Matrix representation |
|---|---|---|
| *Controlled-U* gate |  | $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix}$ |

Table 7: Controlled-U gate

$$CR_\phi = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{pmatrix}$$

$$CR_\phi \left|10\right\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \left|10\right\rangle$$

$$CR_\phi \left|11\right\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ e^{i\phi} \end{pmatrix} = \left|1, e^{i\phi}1\right\rangle$$

The controlled phase shift gate only shifts $\phi$ when applied to the state $|11\rangle$. In general, the controlled-U gate maps the basis states as follows,

$$\left|00\right\rangle \mapsto \left|00\right\rangle \qquad \left|10\right\rangle \mapsto \left|1\right\rangle \otimes U\left|0\right\rangle$$

$$\left|01\right\rangle \mapsto \left|01\right\rangle \qquad \left|11\right\rangle \mapsto \left|1\right\rangle \otimes U\left|1\right\rangle$$

iii. **SWAP gate** swaps the first qubit with the second and vice versa, $|xy\rangle \to |yx\rangle$.

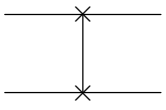
| | Gate notation | Matrix representation |
|---|---|---|
| *SWAP* gate |  | $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ |

Table 8: SWAP gate

$$\text{SWAP}\left|01\right\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \left|10\right\rangle$$

$$\text{SWAP}\,|10\rangle = \begin{pmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} = |01\rangle$$

$$|00\rangle \mapsto |00\rangle \qquad |10\rangle \mapsto |01\rangle$$
$$|01\rangle \mapsto |10\rangle \qquad |11\rangle \mapsto |11\rangle$$

iv. **Toffoli gate** also known as the CCNOT gate is a ternary quantum gate which acts on 3 qubits where the first and the second qubits are the control qubits and the third one is the target. The Toffoli gate applies a NOT gate to the target only when both of the control qubits are in the state $|1\rangle$.

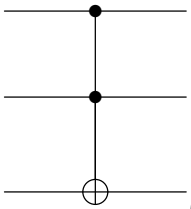
| | Gate notation | Matrix representation |
|---|---|---|
| *Toffoli/CCNOT* gate |  | $\begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{pmatrix}$ |

Table 9: Toffoli/CCNOT gate

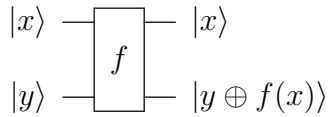
$$\text{CCNOT}\,|110\rangle = \begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} = |111\rangle$$

$$\text{CCNOT}\,|111\rangle = \begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix} = |110\rangle$$

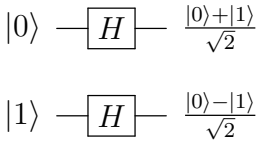
$$|000\rangle \mapsto |000\rangle \qquad |100\rangle \mapsto |100\rangle$$
$$|001\rangle \mapsto |001\rangle \qquad |101\rangle \mapsto |101\rangle$$
$$|010\rangle \mapsto |010\rangle \qquad |110\rangle \mapsto |111\rangle$$
$$|011\rangle \mapsto |011\rangle \qquad |111\rangle \mapsto |110\rangle$$

Now that we have seen some of the widely used quantum gates, we are going to be able to construct quantum algorithms utilizing them. Here are examples of a simple quantum circuit and some gate operations,

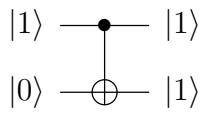
Simple quantum circuit




Hadamard transform




CNOT




# 3 Deutsch & Deutsch-Josza's algorithms

The **Deutsch's algorithm** is one of the simplest algorithm that utilizes quantum parallelism and interference that is able to show a speed up via quantum mechanical operations as opposed to classical computation. Suppose a function, $f : \{0, 1\} \to \{0, 1\}$ where if $f(0) = f(1)$, it is a constant function and if $f(0) \neq f(1)$, it is a balanced function [3]. Given function, $f$ as a black box, we are to determine whether the function is constant or balanced.

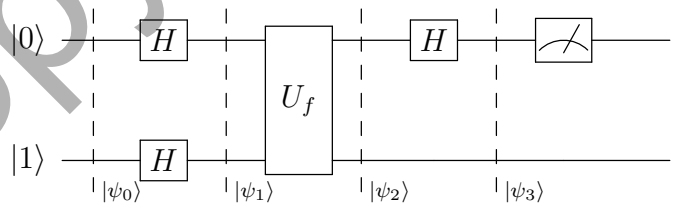
Classically, a minimum of two queries is needed to solve the black box – what is $f(0)$ and $f(1)$ – respectively. In quantum computing, we are able to solve the problem in one query. Let $U_f$ be the quantum counterpart of the function,

$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$$

$$U_f : |x, y \oplus f(x)\rangle \mapsto |x, (y \oplus f(x)) \oplus f(x)\rangle = |x, y\rangle$$

If $f$ is constant, $f(0) \oplus f(1) = 0$ whereas if $f$ is balanced, $f(0) \oplus f(1) = 1$. The Deutsch's algorithm can then be simplified such that it solves for $f(0) \oplus f(1)$.

The quantum circuit to solve this black box can be represented below, with two input qubits initialized as $|0\rangle$ and $|1\rangle$.




where,

$$|\psi_0\rangle = |01\rangle$$
$$|\psi_1\rangle = (\frac{|0\rangle + |1\rangle}{\sqrt{2}})(\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$

To evaluate $|\psi_3\rangle$, it is useful to look at the phase kickback trick which is commonly used in various quantum algorithms. Consider for any $x \in \{0, 1\}$,

$$|\psi\rangle = U_f |x\rangle |-\rangle = \frac{1}{\sqrt{2}}(U_f |x\rangle |0\rangle - U_f |x\rangle |1\rangle) = \frac{1}{\sqrt{2}}(|x\rangle |f(x)\rangle - |x\rangle |1 \oplus f(x)\rangle)$$

Considering the two outcomes of $f(x)$, if $f(x) = 0$, the equation becomes,

$$|\psi\rangle = |x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |x\rangle\,|-\rangle$$

and if $f(x) = 1$,

$$|\psi\rangle = |x\rangle \otimes \frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) = -\,|x\rangle\,|-\rangle$$

We observe that there is a -1 phase change between the two outcomes on the first qubit while the second qubit remains exactly the same. The phase kickback trick can then be simplified as,

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

Applying the phase kickback trick to $|\psi_2\rangle$, we can obtain,

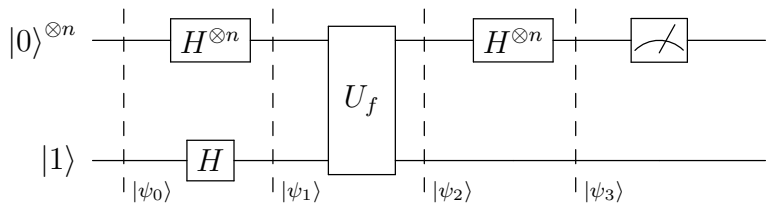
$$\begin{aligned}
|\psi_2\rangle &= \left(\frac{(-1)^{f(0)}\,|0\rangle + (-1)^{f(1)}\,|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\
&= \begin{cases} \pm\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), & \text{if f is constant} \\ \pm\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), & \text{if f is balanced} \end{cases} \\
|\psi_3\rangle &= \begin{cases} \pm\,|0\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), & \text{f is constant} \\ \pm\,|1\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right), & \text{f is balanced} \end{cases}
\end{aligned}$$

Thus, from the observation at $|\psi_3\rangle$ we are able to solve the black box problem with only one query by only measuring the first qubit. If the first qubit state is $|0\rangle$, the function is constant and if its $|1\rangle$, the function is balanced.

The Deutsch's algorithm we have seen is actually a general simple case of the **Deutsch-Josza algorithm** which acts on $n$ inputs. In this case, suppose a function $f : \{0,1\}^n \to \{0,1\}$, it takes n binary values as its input and outputs either 0 or 1. If all the outputs are either 0 or 1 respectively, the function is constant where as if exactly half of the outputs is 0 while the other half is 1, the function is balanced.

Classically, the solution requires $2^{n-1} + 1$ queries. The Deutsch-Josza algorithm solves the black box with exactly one evaluation of $f$. Let $U_f$ be the quantum counterpart of the function,

$$U_f : |x\rangle^{\otimes n}\,|y\rangle \mapsto |x\rangle^{\otimes n}\,|y \oplus f(x)\rangle$$




Much like the Deutsch'algorithm, all $n$ of the input qubits, initialized as $|0\rangle^{\otimes n}$ and the answer register, $|1\rangle$ are put into superposition states before passed through the function, $U_f$. The query register of $n$ qubits is then applied with a Hadamard gate before being measured.

$$|\psi_0\rangle = |0\rangle^{\otimes n} |1\rangle$$
$$|\psi_1\rangle = \sum_{x\in\{0,1\}^n} \frac{|x\rangle}{\sqrt{2^n}}(\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$
$$|\psi_2\rangle = \sum_{x\in\{0,1\}^n} \frac{(-1)^{f(x)}|x\rangle}{\sqrt{2^n}}(\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$

The Hadamard transform, $H^{\otimes n}$ on $|x_1, x_2, ..., x_n\rangle$, can be summarised as,

$$\begin{aligned} H^{\otimes n}|x\rangle &= H|x_1\rangle \otimes H|x_2\rangle \otimes ... \otimes H|x_n\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_1}|1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_2}|1\rangle) \otimes \\ &\quad \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_n}|1\rangle) \\ &= \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} (-1)^{x_1z_1+x_2z_2+...+x_nz_n}|z_1\rangle|z_2\rangle ...|z_n\rangle \end{aligned}$$

$$H^{\otimes n}|x\rangle = \frac{(-1)^{xz}|z\rangle}{\sqrt{2^n}}$$

Using this equation, it is now possible to evaluate $|\psi_3\rangle$,

$$|\psi_3\rangle = \sum_{z\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} \frac{(-1)^{xz+f(x)}|z\rangle}{2^n}(\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$

The amplitude of $|z\rangle = |0\rangle^{\otimes n}$ is $\sum_x (-1)^f(x)/2^n$. If the function is constant, the amplitude will be either +1 or -1. If $f$ is balanced, the positive and negative amplitudes cancel each other and the total amplitude of the state $|0\rangle^{\otimes n}$ is 0. Hence, we have shown that it is possible to solve the black box with just one evaluation. However nice of a speedup both of the Deutsch and Deutsch-Josza's algorithm have shown, they have no real practical application. They serve as a proof of concept that quantum algorithms are able to solve certain problems much faster compared to classical computing.

# 4 Grover's algorithm

**Grover's algorithm**, devised by Lov Grover back in 1996, is a quantum algorithm that searches an index or key value in an unstructured database using $O(\sqrt{N})$ evaluations where $N$ is the size of the domain. Classically, the searching algorithm takes linear time, $O(N)$ to evaluate as opposed to the exponential speedup provided by Grover's algorithm.

Suppose a function, $f : \{0,1\}^n \to \{0,1\}$ such that $f(x) = 0$ but for one key value, $x^* \in \{0,1\}^n$ where $f(x^*) = 1$. Assuming that $N = 2^n$ where there are $N$ many entries, we need to find $x^*$. Given a quantum oracle $O$,

$$O : |x\rangle|q\rangle \to |x\rangle|q \oplus f(x)\rangle$$

where $|x\rangle$ is the index register and $|q\rangle$ is the oracle qubit. The Grover search can implemented by first applying the Hadamard transformation on the first register,

$$\frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} |x\rangle|0\rangle$$

Applying the oracle operator, $O$,

$$\frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x, f(x)\rangle$$

where all the basis states has the same amplitude $1/\sqrt{N}$. Measuring the first register will give one of the basis states and measuring the second qubit will give $|0\rangle$ with probability $(N-1)/N$ or $|1\rangle$ with probability $1/N$.

There are two tricks in implementing the Grover search. Essentially what the Grover's algorithm tries to achieve is to flip and amplify the amplitudes of the states some number of times such that the only significant amplitude left is of the key value basis state while the amplitudes of the rest of the states have diminished. The first trick is the **phase inversion of** $x^*$. Firstly, we set the control qubit of $|0\rangle$ to its superposition state [5]. By using the phase kickback identity seen from the Deutsch's algorithm, we are able to get,

$$O : |x\rangle \, (\frac{|0\rangle - |1\rangle}{\sqrt{2}}) \rightarrow (-1)^{f(x)} \, |x\rangle \, (\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$

$$O : |x\rangle \rightarrow (-1)^{f(x)} \, |x\rangle$$

what this does is essentially flipping the the amplitude of only the key value while leaving the rest unchanged. Now that we have achieved flipping, we need to amplify the amplitude of the key value [5].

The second trick, **inversion about the mean**, does exactly what we need. To understand exactly what is meant by this, we consider a sequence of integers, $[23, 18, 34, 56, 39]$ where the mean, $\mu = 34$. The inversion about the mean of a particular integer from the sequence can be defined as the difference between the mean and the initial value plus the mean.

$$\alpha^{'} = \mu + (\mu - \alpha) = 2\mu - \alpha$$

Given a general state,

$$\sum_k \alpha_k \, |k\rangle$$

the inversion about the mean will produce $\sum_k (2\mu - \alpha_k) \, |k\rangle$ where

$$\mu = \frac{1}{N} \sum_k \alpha_k$$

In Grover search, we must first change the basis to a general superposition state before applying the inversion about $|\psi\rangle$ before reverting to the computational basis. Consider the operator, $U_0^{\perp}$,

$$U_0^{\perp} = \begin{cases} |x\rangle \rightarrow - |x\rangle \, , & x \neq 0 \\ |0\rangle \rightarrow |0\rangle \end{cases}$$

$U_0^{\perp}$ inverts the phase of all states that are orthogonal to $|0\rangle$. In terms of matrices,

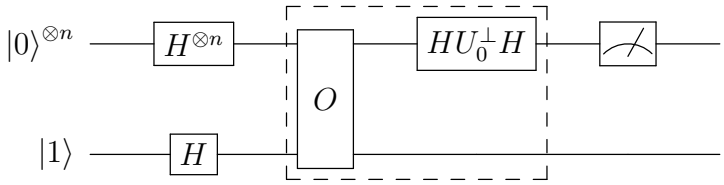
$$U_0^{\perp} = (2 \, |0\rangle \, \langle 0| - I)$$

To do an inversion about the superposition state $|\psi\rangle = 1/\sqrt{N} \sum_k |k\rangle$, some fashion of Hadamard transformations must be applied to $U_0^{\perp}$. For this basis change, $HU_0^{\perp}H$ is,

$$HU_0^{\perp}H = (2 \, |\psi\rangle \, \langle \psi| - I)$$

This matrix essentially inverts any vector applied to it around the mean. Now that we have studied these two tricks, we are able to flip and amplify the amplitude of the key value. These steps are done in iterations until the amplification of the amplitude of the key value state is significant. This step, which is represented in the dashed box in the quantum circuit, is called the Grover's iteration and are usually repeated for $O(\sqrt{N})$ times.

Grover's algorithm quantum circuit




The Grover's algorithm start with the state $|000...0\rangle$ in the first input register. Both the first and the second registers are applied with the Hadamard transformation. The next step is applying the Grover's iteration which is usually applied for $\frac{\pi\sqrt{N}}{4}$ times. In this iteration, the oracle or the phase inversion operator, $O$ and the inversion about the mean or the Grover's diffusion operator, $HU_0^{\perp}H$ are applied. The first register's qubits are then measured which will show an amplified amplitude of the key value state.

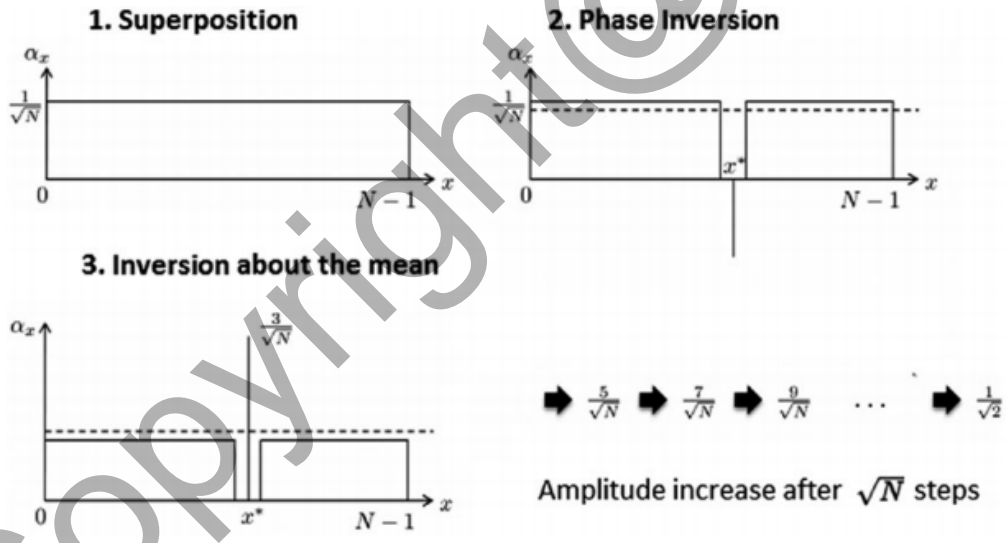



Figure 2: Grover's iteration effect on the amplitudes of states [6].

# 5 Shor's algorithm

The invention of a quantum algorithm for factorization that operates in polynomial time in $n$ is one of the most well-known quantum computing achievements. **Shor's algorithm**, named after Peter Shor, manages just that in $O(n^3 \log n)$ time using $O(n^2 \log n \log\log n)$ gates. The most widely used public-key cryptosystem, RSA, lies solely on the basis that these big numbers are impossible to factored classically. However, this algorithm shows that it may be feasible to do so on a quantum computer which will render the cryptosystem useless. This has led to the birth of a new sub-field called post-quantum cryptography which is devising quantum-proof cryptographic algorithms.

Given a large integer $N = p \cdot q$ for some prime integers $p$ and $q$, we want to find the prime factors $p$ and $q$. Shor's algorithm reduces this problem to a period-finding problem in a function with cyclic property. Suppose the function $f_{a,N}(x) = a^x \pmod N$ where $a$ and $N$ are relatively prime. The period of $f_{a,N}$, $r$, is the least number of $x$ such that $a^r \pmod N = 1$.

To understand the algorithm for integer factorization, it is useful to look at number theory [3, 7]. Below is a simplified algorithm of integer factorization,

i. Pick a number $a$ that is coprime with $N$.

ii. Find the period $r$ of the function $a^r \pmod N$ where it is the smallest $r$ such that $a^r \equiv 1 \pmod N$.

iii. If $r$ is even: $x \equiv a^{r/2} \pmod N$.
$\longrightarrow$ If $x + 1 \not\equiv 0 \pmod N$: At least one of $\{p, q\}$ is in $\gcd(x + 1, N)$ and $\gcd(x - 1, N)$.
Else: Find another $a$.

For example, consider factorizing the integer 15 with $a = 13$ where the period, $r$ of $13^r \pmod{15} = 1$ is 4.

$$x = a^{r/2} \pmod N = 13^2 \pmod{15} = 4$$
$$x + 1 = 5 \equiv 5 \pmod{15}$$
$$\gcd(5, 15) = 5 \qquad \gcd(3, 15) = 3$$
$$p, q = 5, 3$$

The problem of finding the period can be exploited by quantum computing using Quantum Fourier Transform providing polynomial runtime. The period of the modular exponentiation sequence can be found using QFT (see Appendix A) in the encoded amplitudes of the quantum state.

A Hadamard transform is applied to the first register of $m$ qubits producing an equal superposition state.

$$\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \, |\mathbf{0}\rangle$$

Next is a modular exponentiation of the function $U_{f_{a,N}}(x) = a^x \pmod N$ on the second register of qubits where $a$ is chosen randomly,

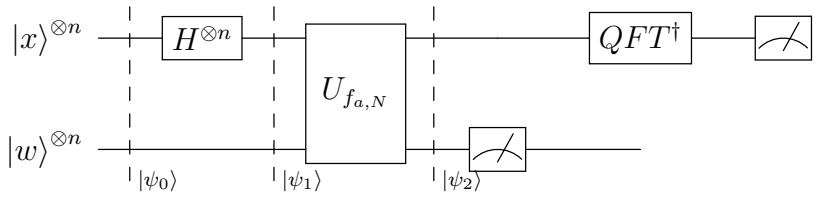
$$\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \, |f(x)\rangle$$

The second register is measured, the only states left in the superposition are those integers $x$ whose $a^x \pmod N$ is the same as the measured value. The only non-zero states in the first register will be the multiples of $r$ with some offset $t$ [8]. Applying QFT on the first register, the state becomes,

$$\frac{1}{\sqrt{N/r}} \sum_{j=0}^{\frac{N}{r}-1} |jr + t\rangle \xrightarrow{QFT} \sum_{j=0}^{N-1} \frac{1}{\sqrt{r}} |j\rangle$$

where $j = \frac{kN}{r}$. Now that we have a superposition state with period $N/r$ with no offset where the only basis states with non-zero amplitudes are those which are multiples of $N/r$. Measuring the first register, we will obtain $\frac{kN}{r}$ where computing $\gcd(\frac{kN}{r}, N)$ will give us $N/r$ and thus we can obtain the period, $r$.

Shor's algorithm quantum circuit




Let's try solving the factorization of 15 using the quantum period-finding algorithm [7]. The binary representation of 15 is 1111, hence we will need 4 qubits for our first register, $|x\rangle$ and another 4 qubits for the second register, $|w\rangle$ which we will apply the modular exponentiation on where $|x\rangle|w\rangle \rightarrow |x\rangle|w \oplus f_{a,N}(x)\rangle$.

$$
\begin{aligned}
|\psi_0\rangle &= |0\rangle^{\otimes 4}|0\rangle^{\otimes 4} \\
|\psi_1\rangle &= [H^{\otimes 4}|0\rangle]|0\rangle^{\otimes 4} = \frac{1}{4}(|0\rangle_4 + |1\rangle_4 + |2\rangle_4 + ... + |15\rangle_4)|0\rangle_4 \\
|\psi_2\rangle &= \frac{1}{4}(|0\rangle_4 \left|13^0 \pmod{15}\right\rangle + |1\rangle_4 \left|13^1 \pmod{15}\right\rangle + \\
&\quad \cdots + |15\rangle_4 \left|13^{15} \pmod{15}\right\rangle) \\
&= \frac{1}{4}\begin{pmatrix} |0\rangle_4|1\rangle_4 + |1\rangle_4|13\rangle_4 + |2\rangle_4|4\rangle_4 + |3\rangle_4|7\rangle_4 + \\ |4\rangle_4|1\rangle_4 + |5\rangle_4|13\rangle_4 + |6\rangle_4|4\rangle_4 + |7\rangle_4|7\rangle_4 + \\ |8\rangle_4|1\rangle_4 + |9\rangle_4|13\rangle_4 + |10\rangle_4|4\rangle_4 + |11\rangle_4|7\rangle_4 + \\ |12\rangle_4|1\rangle_4 + |13\rangle_4|13\rangle_4 + |14\rangle_4|4\rangle_4 + |15\rangle_4|7\rangle_4 \end{pmatrix}
\end{aligned}
$$

Measuring the $|w\rangle$ register will result in the output between $1, 13, 4, 7$ with equal probabilities. Let assume we measure 7. After $|w\rangle = |7\rangle_4$, $|x\rangle$ becomes,

$$|x\rangle = \frac{1}{2}(|3\rangle_4 + |7\rangle_4 + |11\rangle_4 + |15\rangle_4)$$

Applying $\text{QFT}^{\dagger}$ on the $|x\rangle$ register,

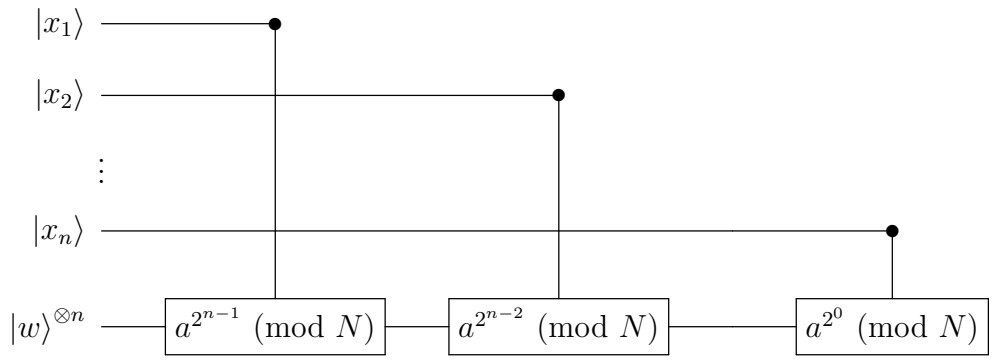
$$
\begin{aligned}
\text{QFT}^{\dagger}|x\rangle &= \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} e^{\frac{-2\pi i x y}{N}}|y\rangle \\
&= \frac{1}{4}\sum_{y=0}^{15}(e^{\frac{-3\pi i y}{8}} + e^{\frac{-7\pi i y}{8}} + e^{\frac{-11\pi i y}{8}} + e^{\frac{-15\pi i y}{8}})|y\rangle \\
&= \frac{1}{4}(4|0\rangle_4 + 4i|4\rangle_4 - 4|8\rangle_4 - 4i|12\rangle_4)
\end{aligned}
$$

Measuring the $|x\rangle$ register will give 0, 4, 8, 12 with equal probabilities. 3/4 of the outputs are multiples of $N/r$. Now that we have obtained $r$, the prime factors can be find classically.

The function $U_{f_{a,N}}$ in the quantum circuit is implemented using phase rotation gates which is akin to the ones in quantum phase estimation (QPE). Letting $f_{a,N}(x) = a^x \pmod{N}$, the function $U_{f_{a,N}}$ can be represented by using controlled-U gates,

$$x = [x_1, x_2, ..., x_n] = 2^{n-1}x_1 + 2^{n-2}x_2 + ... + 2^0 x_n$$

$$f_{a,N}(x) = a^{2^{n-1}x_1}a^{2^{n-2}x_2}...a^{2^0 x_n} \pmod{N}$$




We can see that the Shor's algorithm circuit now looks similar to the Quantum Phase Estimation circuit (see Appendix B). Shor's period-finding algorithm is essentially a special case of phase estimation.

# 6 Harrow-Hassidim-Lloyd (HHL) algorithm

In 2009, Aram Harrow, Avinatan Hassidim, and Seth Lloyd devised the quantum algorithm for linear systems of equations which known as the **HHL algorithm** [9]. This algorithm yields a scalar measurement of the solution vector as its output and does not give the actual values of the solutions.

This algorithm provides a significant speedup over its classical counterpart given that the linear system is sparse and the condition number $\kappa$ which is the ratio of the maximum eigenvalue and the minimum is low. The algorithm runs at time $O(\frac{\kappa^2 s^2}{\epsilon} \log_2 N)$ where $s$ is the sparseness and $\epsilon$ is the accuracy, which trumps over the fastest classical runtime of $O(N\kappa)$ providing exponential speedup.

Classically, the problem of linear system $A\mathbf{x} = \mathbf{b}$ can be solved by finding the values of $\mathbf{x} = A^{-1}\mathbf{b}$. In quantum computation, we are still trying to find the solution vector $|x\rangle$ given an $N \times N$ Hermitian matrix, $\mathbf{A}$ and the unit vector, $|b\rangle$.

$$A|x\rangle = |b\rangle \qquad |x\rangle = A^{-1}|b\rangle$$

Since A is Hermitian, its spectral decomposition is,

$$A = \sum_{j=0}^{N-1} \lambda_j |u_j\rangle \langle u_j|$$

$$A^{-1} = \sum_{j=0}^{N-1} \lambda_j^{-1} |u_j\rangle \langle u_j|$$

where $\lambda_j$ is the eigenvalues of A and $|u_j\rangle$ is its eigenvectors. The unit vector $|b\rangle$ can be defined in the eigenbasis of A as,

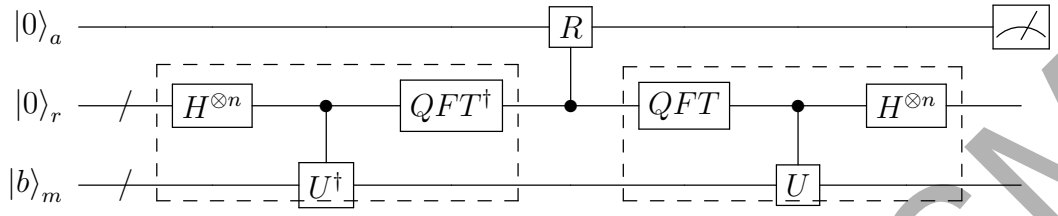
$$|b\rangle = \sum_{j=0}^{N-1} b_j |u_j\rangle$$

Hence, we can obtain $|x\rangle$ such that,

$$|x\rangle = A^{-1}|b\rangle = \sum_{j=0}^{N-1} \lambda_j^{-1} b_j |u_j\rangle$$

The HHL algorithm requires three registers, initialized to $|0\rangle$. One register, $|0\rangle_r$ is used to store the binary representation of A and another one, $|b\rangle_m$ carries the vector solution [7] acts as the memory register which will output $|x\rangle$. The third register is the ancilla or auxiliary qubit which is used in the intermediate steps of the algorithm, $|0\rangle_a$. The quantum circuit for HHL is as follows where the dashed boxes are QPE and QPE$^\dagger$ respectively,

HHL algorithm quantum circuit




Loading the data $b$ to the memory register, $|b\rangle_m$.

$$|0\rangle_{n_b} \rightarrow |b\rangle_{n_b}$$

We can apply the Quantum Phase Estimation (QPE) where an estimate of the eigenvalues can be represented in quantum states for $|0\rangle_r$,

$$U = e^{iAt} := \sum_{j=0}^{N-1} |u_j\rangle \langle u_j|$$

such that the quantum state of the registers expressed in the eigenbasis of A is,

$$|0\rangle_a |0\rangle_r |b\rangle_m \rightarrow \sum_{j=0}^{N-1} b_j |0\rangle_a |\lambda_j\rangle_r |u_j\rangle_m$$

where $|\lambda_j\rangle_r$ is the $n$ bit binary representation of $\lambda_j$. A rotation conditioned with $|\lambda_j\rangle_r$ is applied to the ancilla qubit, $|0\rangle_a$:

$$\sum_{j=0}^{N-1} b_j |\lambda_j\rangle_r |u_j\rangle_m \left( \sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle \right)$$

where $C$ is the normalization constant. By applying QPE$^\dagger$, we can get rid of the eigenvalues states, $|\lambda_j\rangle_r$.

$$\sum_{j=0}^{N-1} b_j |0\rangle_r |u_j\rangle_m \left( \sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle \right)$$

We will only get $|x\rangle$ when the measured outcome of the ancilla qubit is $|1\rangle$. Otherwise, the algorithm has to be repeated and some sort of an amplitude amplication subroutine can be used to increase the probability of measuring $|1\rangle$ [4].

$$|x\rangle \approx \left( \sqrt{\frac{1}{\sum_{j=0}^{N-1} |b_j|^2 / |\lambda_j|^2}} \right) \sum_{j=0}^{N-1} \frac{b_j}{\lambda_j} |0\rangle_r |u_j\rangle_m$$

Since then, there are several improvements that has been made in favor of the HHL algorithm that improves the run time by either replacing the QPE with Quantum Fourier Transform or

by using a hybrid classical-quantum algorithm [10, 11]. Improvements on algorithm that solves the linear system have also been shown experimentally for $8 \times 8$ matrix via adiabatic quantum computing inspired approach which is based on Hamiltonian simulation. Recently, a variational approach for solving quantum linear system problems has been demonstrated to perform well in noisy intermediate-scale quantum (NISQ) computers up to $1024 \times 1024$ problem size (10 qubits) [12].

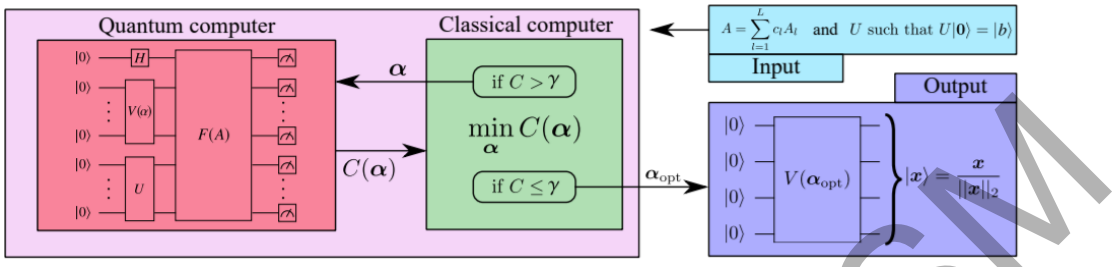



Figure 3: A schematic overview of the variational quantum linear solver (QLSP) [12].

The inputs are the linear combination of unitaries $A_l$ where the cost function will be estimated via an ansatz $V(\alpha)$ with parameters $\alpha$ in a quantum-classical loop. Once the cost reaches the desired threshold, the optimal parameters $\alpha_{\text{opt}}$ will be passed through the ansatz to form $|x\rangle$. Since this is essentially a different algorithm altogether from the HHL algorithm, we will not go much further than this.

# 7 Conclusion

Quantum computing and algorithms have shown promising results in providing speedup when compared to its classical counterpart for certain cases. The algorithms that have been discussed in this report are some of the most important groundbreaking quantum algorithms with great potential to be utilized in more complex and grounded problems. The only obstacle for quantum computing before we are able to fully utilize these algorithms are the hardware limitations on the machines themselves. In this report, we have discussed Grover's searching algorithm, Shor's period-finding algorithm and HHL algorithm for solving linear systems. We are particularly interested in the last algorithm as it serves as the foundation for our study in implementing least squares data fitting.

# Acknowledgements

This work is related to the ongoing research by Ts. Dr. Bahari Idrus and Assoc. Professor. Dr. Mohammad Khatim Hasan funded by GUP-2020-061.

# References

[1] Richard P Feynman. "Simulating physics with computers". In: *Int. J. Theor. Phys* 21.6/7 (1982).

[2] Gordon E Moore et al. *Cramming more components onto integrated circuits*. 1965.

[3] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. 2002.

[4] J Abhijith et al. "Quantum algorithm implementations for beginners". In: *arXiv e-prints* (2018), arXiv–1804.

[5] Noson S Yanofsky and Mirco A Mannucci. *Quantum computing for computer scientists*. Cambridge University Press, 2008.

[6] Vladimir Silva. *Practical quantum computing for developers: programming quantum rigs in the cloud using Python, quantum assembly language and IBM QExperience*. Apress, 2018.

[7] Abraham Asfaw et al. *Learn Quantum Computation Using Qiskit*. 2020. URL: `http://community.qiskit.org/textbook`.

[8] Jozef Gruska et al. *Quantum computing*. Vol. 2005. McGraw-Hill London, 1999.

[9] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations". In: *Physical review letters* 103.15 (2009), p. 150502.

[10] Andrew M Childs, Robin Kothari, and Rolando D Somma. "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision". In: *SIAM Journal on Computing* 46.6 (2017), pp. 1920–1950.

[11] Yonghae Lee, Jaewoo Joo, and Soojoon Lee. "Hybrid quantum linear equation algorithm and its experimental test on ibm quantum experience". In: *Scientific reports* 9.1 (2019), pp. 1–12.

[12] Carlos Bravo-Prieto et al. "Variational quantum linear solver: A hybrid algorithm for linear systems". In: *Bulletin of the American Physical Society* 65 (2020).

# A Quantum Fourier Transform (QFT)

The quantum Fourier transform is a linear transformation of qubits which essentially changes the basis of qubits from computational, $\{|0\rangle, |1\rangle\}$ to Fourier basis, eg: $\{|+\rangle, |-\rangle\}$ for single qubit.

$$|\tilde{x}\rangle = \mathrm{QFT}\,|x\rangle$$

$$\mathrm{QFT}\,|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi i x y}{N}} |y\rangle$$

The single qubit QFT is essentially the Hadamard transform. Suppose for a single qubit case with $N = 2$,

$$\begin{aligned}\mathrm{QFT}\,|x\rangle &= \frac{1}{\sqrt{2}} \sum_{y=0}^{2-1} e^{\frac{2\pi i x y}{N}} |y\rangle \\ &= \frac{1}{\sqrt{2}}[|0\rangle + e^{\pi i x} |1\rangle]\end{aligned}$$

$$\mathrm{QFT}\,|0\rangle = \frac{1}{\sqrt{2}}[|0\rangle + |1\rangle] = |+\rangle$$

$$\mathrm{QFT}\,|1\rangle = \frac{1}{\sqrt{2}}[|0\rangle + e^{i\pi} |1\rangle] = \frac{1}{\sqrt{2}}[|0\rangle - |1\rangle] = |-\rangle$$

In QFT, each qubit state is changed from $|x_k\rangle$ to $|0\rangle + \exp\{2\pi i x/2^k\} |1\rangle$ [3]. This can be easily shown where,

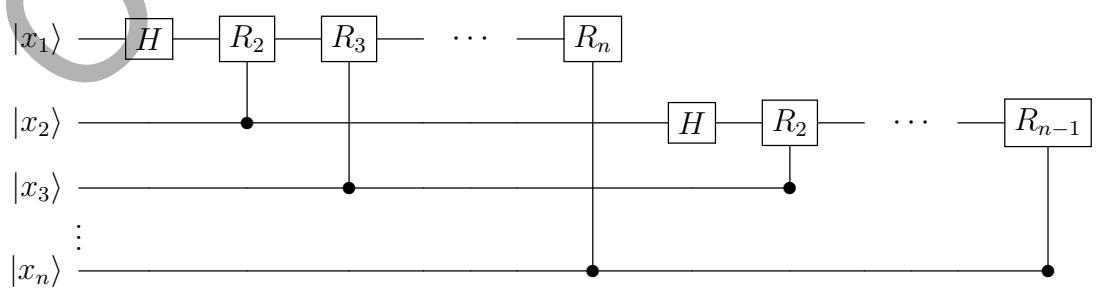
$$|x\rangle = |x_1, x_2, x_3, ..., x_n\rangle = |x_1\rangle \otimes |x_2\rangle \otimes ... \otimes |x_n\rangle$$

$$\downarrow \mathrm{QFT}$$

$$|\tilde{x}\rangle = \frac{1}{\sqrt{N}}(|0\rangle + e^{\frac{2\pi i x}{2^1}} |1\rangle) \otimes (|0\rangle + e^{\frac{2\pi i x}{2^2}} |1\rangle) \otimes ... \otimes (|0\rangle + e^{\frac{2\pi i x}{2^n}} |1\rangle)$$

this product representation of QFT is useful as it gives us ideas as how the quantum circuit is constructed. Besides the Hadamard gate that gives the superposition of states, the quantum circuit for QFT is constructed by using the unitary rotation gate, $R_k$,

$$R_k \equiv \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$$



# B Quantum Phase Estimation (QPE)

Quantum phase estimation utilizes QFT as one its main subroutine. Remember that a unitary operator has eigenvalues of the form $e^{i\theta}$ and eigenvectors that form an orthonormal basis,

$$U\left|\psi\right\rangle = e^{i\theta_\psi}\left|\psi\right\rangle$$

The goal of QPE is to estimate $\theta_\psi$ given the ability to prepare $\psi$ and the ability to apply $U$. One of the main uses of QPE includes simulating Hamiltonian evolution. The procedure uses two registers – first register of $n$ qubits initialized as $|0\rangle$ and the second register of $|\psi\rangle$. The precision and accuracy for $\theta_\psi$ can be obtained by increasing $n$ number of qubits. This procedure starts off by applying Hadamard transformation on the first register before applying the controlled-$U^{2^j}$ operation on the second. The state of the first register can be represented as,

$$\left|0\right\rangle^{\otimes n} \xrightarrow{U^{2^j}} \frac{1}{\sqrt{2}}^n (\left|0\right\rangle + e^{2\pi i\theta_\psi 2^{n-1}}\left|1\right\rangle) \otimes (\left|0\right\rangle + e^{2\pi i\theta_\psi 2^{n-2}}\left|1\right\rangle) \otimes \cdots \otimes (\left|0\right\rangle + e^{2\pi i\theta_\psi 2^{0}}\left|1\right\rangle)$$

The inverse QFT is then applied to the first register by reversing the QFT circuit seen in Appendix A. This operation will yield a good estimate for $\theta_\psi$ [3]. If we carefully examine the form of the first register prior to the inverse QFT operation, it is similar to that of the product of QFT where $\theta_\psi \to 2\pi/2^x$,

$$\text{QFT} : \left|\tilde{x}\right\rangle = \frac{1}{\sqrt{N}}(\left|0\right\rangle + e^{\frac{2\pi ix}{2^1}}\left|1\right\rangle) \otimes (\left|0\right\rangle + e^{\frac{2\pi ix}{2^2}}\left|1\right\rangle) \otimes ... \otimes (\left|0\right\rangle + e^{\frac{2\pi ix}{2^n}}\left|1\right\rangle)$$

From this observation, it is trivial that applying an inverse QFT to the first register will give $\left|\tilde{\theta}\right\rangle$ which is an estimate of $\theta_\psi$ [3],

$$\frac{1}{2^{n/2}}\sum_{t=0}^{2^n-1} e^{2\pi i\theta_{\psi_t}}\left|t\right\rangle\left|\psi\right\rangle \to \left|\tilde{\theta_\psi}\right\rangle\left|\psi\right\rangle$$

The circuit for this phase estimation procedure can be represented as,