Notes on basic notions of basic Quantum algorithms

A mathematical perspective

Leo Kruglikov

March 26, 2023

1 Concepts and notions

Fundamentals

Here I non-rigourously describe the main notions that are used during the algorithm discussions.

Concept 1.1. *A quantum cirquit is composed of a (multiple) qubit(s) input and a measurable output of the same size. If multiple qubits are involved, the inputs are mathematically defined:*

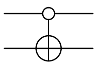
$$\textit{input of n qubits } |\psi_i\rangle = \bigotimes_{i=1}^{n} |\psi_i\rangle \tag{1.1}$$

Concept 1.2. *A quantum circuit is composed of different and various gates, that can be applied on multiple qubits. Every gate is an operator. If a gate $\widehat{O}$ is applied only on state $|\psi_i\rangle$, the operator on the whole system is given by*

$$\widehat{O}_{on\ state\ i} = \mathbb{1}_{(1)} \otimes \mathbb{1}_{(2)} \cdots \otimes \mathbb{1}_{(i-1)} \otimes \widehat{O} \otimes \mathbb{1}_{(i+1)} \cdots \otimes \mathbb{1}_{(n)} \tag{1.2}$$

Main Circuits

Let's discuss main representation of the circuits.

Operator	Bra-Ket representation	Matrix	Other math properties	Qiskit
Identity op. $I = \mathrm{Id} = \mathbb{1}$	Closure relation: $\forall \{a_i\} \in V$ $\mathbb{1} = \sum_i \|v_i\rangle \langle v_i\|$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\forall \|\psi\rangle \in V, \mathbb{1} \|\psi\rangle = \|\psi\rangle$	
Hadamard op. $H = \mathcal{H}$	In the Z/X basis $\mathcal{H} = \|+\rangle \langle 0\| + \|-\rangle \langle 1\|$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	Change of basis from Z to X basis.	
Z-gate $\sigma_z = Z$	$\sigma_z \equiv \|0\rangle \langle 1\| - \|1\rangle \langle 1\|$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	Eigen-op. of the $\left\|+1; \hat{S}_z \right\rangle = \|0\rangle$ state	
X-gate $\sigma_x = X$	$\sigma_x \equiv \|0\rangle \langle 1\| + \|1\rangle \langle 0\|$ $\sigma_x \equiv \|+\rangle \langle +\| - \|-\rangle \langle -\|$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	Eigen-op. of the $\|+\rangle \equiv \frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$	
phase gate $P(\phi)$	$P(\phi) \equiv \|0\rangle \langle 0\| + e^{i\phi} \|1\rangle \|1\rangle$	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$		
Controlled-NOT 	CNOT = $= \|0\rangle \langle 0\| \otimes \mathbb{1} + \|1\rangle \langle 1\| \otimes X$ $\|x_c\rangle \|x_t\rangle \xrightarrow{CNOT} \|x_c\rangle \|x_c \oplus x_t\rangle$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	Acting on a 4x1 vec. spanned by $\{\|00\rangle, \|01\rangle, \|10\rangle, \|11\rangle\}$	

2 Deutsch-Jozsa algorithm

The Deutsch-Jozsa algorithm was the first algorithm to be proposed. The goal of the Deutsch's quantum algorithm is to determine a concrete property of a funciton. In this case, we use the quantum algorithm, to determine the properties of the function. In the case of the

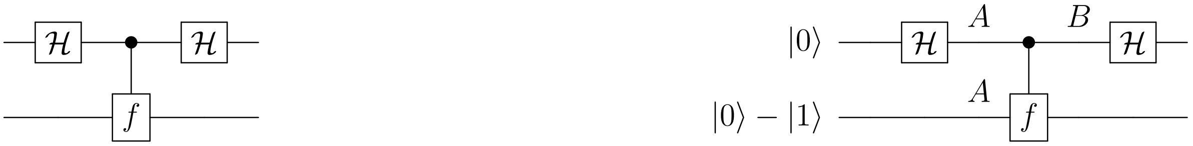
Definition 2.1 (Constant and balanced)**.** The function used here is a function from $\{0,1\}^N \mapsto \{0,1\}$. The outputs of it can be either constant or balanced. We say that the function is *constant* if for any input the output is constant, that is, whether all ones or all zeroes. We say that the function is *balanced* if it outputs ones half of the times and zeroes half of the other times (that is, out of all possible inputs, it will output a 1 in the half of the input cases and 0 in the other).

We will consider the case where n (the number of inputs) is equal to one. That is, the function $f(x)$ has one input bit. $f(x)$, with $x \in \{0,1\}$. Let's see examples of functions that are either balanced or constant. Let $f_1(x)$. The function happens to output the following: $f_1(0) = 0$ and $f_1(1) = 0$. We see that the function has zeroes for any input. Therefore, we can conclude that the function f_1 is constant. Consider now the function f_2, satisfying the following: $f_2(0) = 1$ and $f_2(1) = 0$. As there are 2 kinds of inputs (0 and 1), there are only 2 possible outputs (2 different outputs for 2 different inputs). We see that for the function f_2, half of the outputs (i.e. only one out of two) is 0 and half of the inputs is 1. Therefore, the function f_2 is balanced.

The example was a simple case, where the input was simply either 0 or 1. The definition of this constant/balanced can be extended to a more complex case of an input $\{0,1\}^n$, which is the Deutsch-Jozsa algorithm.

Deutsch's algorithm

Let's first consider the Deutsch's algorithm [2], that is, with one input bit. Note that in this algorithm we assume or we are promised that the given function is either balanced or constant (it can be neither). The circuit is *postulated* to be. In order for it to have a working mechanism, we need to prepare the initial states.



(a) Deutsch's circuit

(b) Initialized states

Figure 1: Deutsch's circuit

Having the circuit with prepared initial states (Figure 1) we're now in the situation in carefully initialize the cirquit. We've identified 2 states of the cirquit: `A` and `B`, which we'll quickly discuss [1].

The first register begins with the state $|0\rangle$. As usual, applying the Hadamard gate $\mathcal{H}$ on the first register gives us $|0\rangle \xrightarrow{\mathcal{H}} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ as usual. Thus, the total state at the point **A** is given by the tensor product of both registers, which is $(|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle)$.

After the point A we apply the *function evaluation* (TODO). Let's consider the rigourous operation:

state at **A**: $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Now, the state after the function f, using the concept of the evaluation function, we have the *general* expression of $|x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes |y \oplus f(x)\rangle$, which, in the case of our circuit Figure 1, gives the state $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ for $|y\rangle$. Thus, for the circuit between **A** and **B**:

$$|\psi_B\rangle = \left|\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right\rangle \otimes \left|f(x) \oplus \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right\rangle \tag{2.1}$$

$$\frac{1}{2}\Big|\,|0\rangle + |1\rangle\,\Big\rangle \otimes \Big|\,|f(x)\rangle - |1 \oplus f(x)\rangle\,\Big\rangle \tag{2.2}$$

$$\frac{1}{2}\Big|\,|0\rangle + |1\rangle\,\Big\rangle \otimes \Big|(-1)^{f(x)}(|0\rangle - |1\rangle)\Big\rangle \tag{2.3}$$

Now we can expand this further, still for the state at the point B. When we explicit the expression in order to see the input of the function $f(x)$. What I mean is that in Equation 2.3, the function $f(x)$ it accepts the superposition, as the input is an entangled state.

$$|\psi_B\rangle = \frac{1}{2}\Big|\,|0\rangle + |1\rangle\,\Big\rangle \otimes \Big|(-1)^{f(x)}(|0\rangle - |1\rangle)\Big\rangle \tag{2.4}$$

$$|\psi_B\rangle = \frac{1}{2}\Big|(-1)^{f(x)}(|0\rangle + |1\rangle)\Big\rangle \otimes \Big|\,|0\rangle - |1\rangle\,\Big\rangle \tag{2.5}$$

$$|\psi_B\rangle = \frac{1}{2}\Big|(-1)^{f(0)}\,|0\rangle + (-1)^{f(1)}\,|1\rangle\,\Big\rangle \otimes \Big|\,|0\rangle - |1\rangle\,\Big\rangle \tag{2.6}$$

now we apply an unusual trick is not that simple to see but simple to check that it is true. What we will use here is the fact that

$$f(0) \oplus f(0) \oplus f(1) = f(1) \tag{2.7}$$

This can be seen as the term $f(0) \oplus f(0)$ will always give zero and thus, $0 \oplus f(x) = f(x)$ and therefore,

[1] The $|0\rangle - |1\rangle$ state can be created using the Hadamard gate applied on the $|0\rangle$ *state.TODO*

$$|\psi_B\rangle = \frac{1}{2}\left|(-1)^{f(0)}|0\rangle + (-1)^{f(0)\oplus f(0)\oplus f(1)}|1\rangle\right\rangle \otimes \left||0\rangle - |1\rangle\right\rangle \tag{2.8}$$

$$|\psi_B\rangle = \frac{1}{2}(-1)^{f(0)}\left||0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle\right\rangle \otimes \left||0\rangle - |1\rangle\right\rangle \tag{2.9}$$

We know however that the global phase do not matter at all. In addition to that, we see that in Figure 1, after the point **B**, we're not interested in the state $|y\rangle$ anymore. Thus, we can simply ignore both the global phase $(-1)^{f(0)}$ and the second state $|0\rangle - |1\rangle$. Thus, we're left with the state $\left|\tilde{\psi_B}\right\rangle \equiv |0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle$, on which we apply the $\mathcal{H}$ operation. This gives us

$$\mathcal{H}\left|\tilde{\psi_B}\right\rangle \overset{*}{=} |+\rangle + (-1)^{f(0)\oplus f(1)}|-\rangle = \tag{2.10}$$

$$(1 + (-1)^{f(0)\oplus f(1)})|0\rangle + (1 - (-1)^{f(0)\oplus f(1)})|1\rangle \tag{2.11}$$

Therefore, we obtain that if $f(0) \oplus f(1) = 0$, the measurement of the top qubit will be $|0\rangle$ and if the result $f(0) \oplus f(1) = 1$, the result of the measurement will be given by $|1\rangle$. Therefore, we will measure the $|0\rangle$ bit if the value of the function is constant (either all ones or all zeroes), and similarly, we will measure the value $|1\rangle$ bit if the value of the function is balanced. In both cases, we're making the measurements with probability 1.

Deutsch-Jozsa's algorithm

The Deutsch-Josza's algorithm is similar to the Deutsch's one. That is, it is a generalization from one bit input (i.e. $\{0, 1\}$) to the n bit input (i.e. $\{0, 1\}^n$, which is nothing but a bit string).

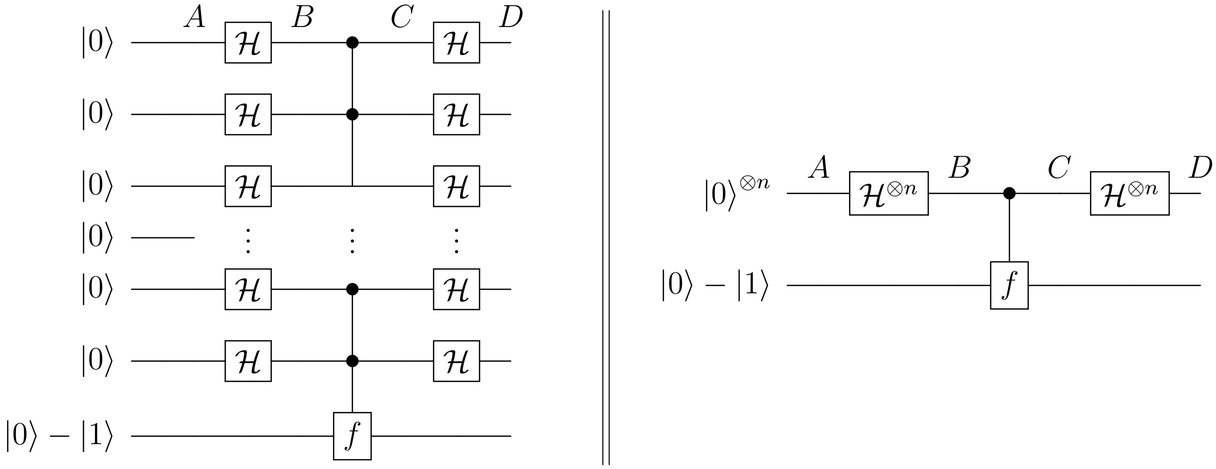


Table 1: The table illustrates 2 schematic circuits for the Deutsch-Josza algorithm. Both of them are equivalent. We prefer the second one (the right one) as it is simply shorter.

The working principle of this generalized version is harder to show and harder to understand. The proof is nicely shown in [2]. Lets start from the initial state, as shown in Table 1. As usual, we sometimes omit the global normalization constant (we thus write $\overset{*}{=}$).

$$|\psi_A\rangle \overset{*}{=}(|0\rangle \otimes |0\rangle ... \otimes |0\rangle) \otimes (|0\rangle - |1\rangle) = |0\rangle^{\otimes n} \otimes (|0\rangle - |1\rangle) \tag{2.12}$$

$$|\psi_B\rangle \overset{*}{=} \bigotimes_{i=0}^{n} \mathcal{H} |\psi_A\rangle \overset{*}{=} \mathcal{H}^{\otimes n} |0\rangle^{\otimes n} \otimes (|0\rangle - |1\rangle) \tag{2.13}$$

$$\overset{*}{=} |+\rangle^{\otimes n} (|0\rangle + |1\rangle) = \sum_{x}^{2^n-1} |x\rangle \otimes (|0\rangle - |1\rangle) \tag{2.14}$$

Note that $|x\rangle$ here represents a binary string. Indeed, a $\mathcal{H}$ acting on $|0\rangle$ can represent the sum of numbers from 0 to 1. ($\mathcal{H} |x\rangle \overset{*}{=} |0\rangle + |1\rangle$). When acting on a 2D space, $(\mathcal{H} \otimes \mathcal{H})(|0\rangle \otimes |0\rangle) \overset{*}{=} |0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle$. We therefore see that the Hadamard gate on the n-dimensional space will give the sum of binary strings ranging from 0 to $2^n - 1$. This is what is meant in the sum over $|x\rangle$. Then, the quantum function evaluation, as defined, will create the transformation $|x\rangle \otimes |y\rangle \xrightarrow{f} |x\rangle \otimes |f(x) \oplus y\rangle$, by definition. Therefore, at the point C (as on Table 1), the transformation give:

$$|\psi_C\rangle \xleftarrow{f} |\psi_B\rangle \tag{2.15}$$

$$|\psi_C\rangle \overset{*}{=} \sum_{x}^{2^n-1} |x\rangle (|f \oplus 0\rangle - |f \oplus 1\rangle) \tag{2.16}$$

$$= \sum_{x}^{2^n-1} |x\rangle (-1)^{f(x)} (|0\rangle - |1\rangle) \tag{2.17}$$

now, we can "ignore" the $|0\rangle - |1\rangle$ part, as after the C point, we're only applying the $\mathcal{H}$ to the top part (i.e. without the $|0\rangle - |1\rangle$). We also remember our important result

$$H |x\rangle \equiv \bigotimes^{n} \mathcal{H} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle \tag{2.18}$$

with the $x \cdot y$ being the vector product between two bitstrings. The bitstrings can be represented as vectors, e.g. $|x\rangle \equiv (0, 1, 1, 0, 0, ..., 1)$ and $|y\rangle \equiv (1, 0, 1, 1, ..., 1)$. Their dot product is defined as $x \cdot y = (x_1 \cdot y_1) \oplus (x_2 \cdot y_2) \oplus (x_3 \cdot x_3) \oplus ... \oplus (x_n \cdot x_n)$. Therefore, by applying the $\mathcal{H}$ on $|\psi_C\rangle$ given in Equation 2.17,

$$\bigotimes^{n} \mathcal{H} |\psi_C\rangle \overset{*}{=} \mathcal{H} \sum_{x}^{2^n-1} |x\rangle (-1)^{f(x)} \tag{2.19}$$

$$= \sum_{x}^{2^n-1} \mathcal{H} |x\rangle (-1)^{f(x)} \overset{\text{Equation 2.18}}{=} \tag{2.20}$$

$$= \sum_{x}^{2^n-1} \sum_{y}^{2^n-1} |y\rangle (-1)^{f(x)} (-1)^{x \cdot y} \tag{2.21}$$

$$= \sum_{y}^{2^n-1} \left[\sum_{x}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot y} \right] |y\rangle \tag{2.22}$$

which is the (almost) final result. This is what we get, when we'll obtain after applying the $\mathcal{H}$ on the $|\psi_C\rangle$ state. Now, we can see that the $|y\rangle$ are linear combinations of probability amplitudes $\sum_x^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot y}$ for each $|y\rangle$. We now have a look at the state $|y\rangle = 0$, that is, $|y\rangle = |0, 0, ..., 0\rangle$. The probability of measuring it is given by $\left| \sum_x^{2^n-1} (-1)^{f(x)} \right|^2$. The probability of measuring this $|0\rangle = |y\rangle$ will be given by 1, if the function is constant. Indeed, if $f(x)$ is constant (either always ones or always zeroes), then we will make the sum of -1 if it is always one, and sum of all $(-1)^0$ if always zeroes. If it is balanced, then we will make the sum of (-1) and 1, which will give us 0 probability if the function is balanced.

Thus, we've showed that the algorithm can determine, whether the function is constant or balanced.

Example

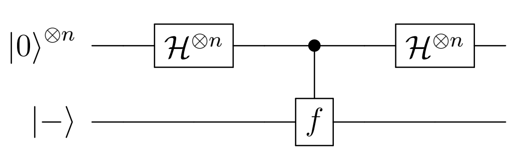
The "problem" with this algorithm is that we need to have a very specific oracle. To be more precise, one may thing that we constructed the concept of solving the algorithm, based on the given oracle that "comes from nowhere". This is not false. For an arbitrary function, we can't construct the algorithm right away.

In order to ckeck the algorithm, one can simply check the algorithm for specific inputs, and whether it represents the specific outputs.

Bernstein-Vazirani algorithm

The next algorithm is the Bernstein-Vazirani algorithm, which also gives a speedup over the classical solution. The technique to solve such problem is similar to the Deutsch-Jozsa's one, that is, uses a phase kick-back trick.

The problem that the Bernstein-Vazirani algorithm aims to solve is the following: we have a function $f(x)$ which has the form $f(x) : \{0,1\}^n \to \{0,1\}$ and $f(x) : x \mapsto s \cdot x = s_1x_1 \oplus s_2x_2 \oplus ... \oplus s_nx_n$, for some "secret" string s, for which we denote the corresponding function $f_s(x)$. The goal of the Bernstein-Vazirani problem is to determine this "secret" string s.



As usual, in order to implement this algorithm, we need a quantum oracle for $f_s(x)$. In this case, the quantum oracle will be defined as for the Deutsch-Jozsa's algorithm as $|x\rangle \otimes |y\rangle \xrightarrow{U_{f_s}} |x\rangle \otimes |y \oplus f_s(x)\rangle$. Similar to the case of the Deutsch-Jozsa, the state that will get through the oracle will be $|x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. The result will be given by $|x\rangle\,(-1)^{f_s(x)}$ as shown in Equation 2.17, where we decided to "drop" the second part with $|-\rangle$. The circuit is given in Table 2.

Let's now show mathematically the Bernstein-Vazirani algorithm:

$$|0\rangle^{\otimes n} \xrightarrow{\mathcal{H}^{\otimes n}} \sum_{x}^{2^n} |x\rangle \tag{2.23}$$

which is the state of the top-register before we reach the quantum function evaluation. Further we write:

$$\sum_{x}^{2^n-1} |x\rangle\,(|0\rangle - |1\rangle) \xrightarrow{U_f} \sum_{x}^{2^n-1} (-1)^{f_s(x)}\,|x\rangle \tag{2.24}$$

$$= \sum_{x}^{2^n-1} (-1)^{s\cdot x}\,|x\rangle \tag{2.25}$$

Then, we need to apply the Hadamard gate to this state. We will then write the following:

$$\sum_{x}^{2^n-1} (-1)^{s\cdot x}\,|x\rangle \xrightarrow[\text{Equation 2.18}]{\mathcal{H}^{\otimes n}} \sum_{x}^{2^n-1} (-1)^{x\cdot s} \sum_{y}^{2^n-1} (-1)^{x\cdot y}\,|y\rangle = \tag{2.26}$$

$$\sum_{x}^{2^n-1} \sum_{y}^{2^n-1} (-1)^{x\cdot s}(-1)^{x\cdot y}\,|y\rangle = \tag{2.27}$$

$$\sum_{x,y}^{2^n-1} (-1)^{(x\cdot s + x\cdot y)}\,|y\rangle \tag{2.28}$$

We now claim that the last expression $\sum_{x,y}(-1)^{x\cdot s+x\cdot y}\left|y\right\rangle = \left|s\right\rangle$. It is not straightforward to see so we can try to prove it (as in [1]). We first can rewrite $x \cdot s + x \cdot y = x \cdot (s \oplus y)$. We can now take one fixed $\left|y\right\rangle$ and sum over the $\left|x\right\rangle$'s:

$$\sum_{x}(-1)^{x\cdot(s\oplus y)} \tag{2.29}$$

Now, if the chosen $\left|y\right\rangle$ is equal to the s, the term $s \oplus y$ will be clearly zero. Therefore, if the chosen y happens to be the same as s, the probability of obtaining it will be maximum (we're adding all ones, as $(-1)^{x\cdot(s\oplus y)} = \sum(-1)^0$). Consequently the only non-zero term is associated to the state $\left|y\right\rangle = \left|s\right\rangle$.

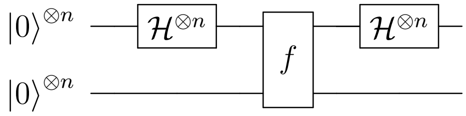
Thus, using this algorithm, we're able to find the state $\left|y\right\rangle = \left|s\right\rangle$ related to this "secret" string.

Simons algorithm

Let's now consider the next algorithm, commonly known as the Simon's algorithm or the Simon's problem. As usual we will treat the concept mathematically, where it is nicely shown in Wikipedia [9].

We first want to determine the problem statement. We are given some kind of function $f(x)$ Often, we define this problem through the concept of the whether this function is *one-to-one* or *two-to-one*. Both of these notions can be formally related to some function properties, namely surjection and injection. Anyways, the *one-to-one* function means that for any input x to the function $f(x)$, there's only one possible output. The two-to-one means that there's possibly several inputs for one single output. For example, $f(0) = 0, f(1) = 1, f(2) = 2, f(3) = 3$ - which is a one-to-one function. Now, an example of a two-to-one function is given by $f(0) = f(1) = 0, f(2) = f(3) = 1$. Note that for the inputs, they can be represented as binary strings, representing numbers.

One can define it in another way, which is fully equivalent. Indeed, the problem now goes as follows: we're given a function $f(x)$ and the number/binary string s. We're now given a promise for $f(x)$: given $f(x)$ and s, $f(x) = f(y)$ if and only if $x \oplus y \in \{0^{\otimes n}, s\}$. That is, the output will be same for two different inputs if and only if $x \oplus y$ is either $0^{\otimes n}$ or s. Here, we define the operation $x \oplus y$ to be the bitwise XOR. That is, $x \oplus y \equiv (x_1 \text{ XOR } y_1, x_2 \text{ XOR } y_2, ..., x_n \text{ XOR } y_n)$.



The goal of the algorithm is to determine the bit-string s. To schematically show that these two formulations (in terms of one-to-one & two-to-one) and the last one are equivalent, we first note that $a \oplus b = 0^{\otimes n} \iff a = b$. Another fact that we notice is the following: for some x and s, we have that $x \oplus y = s$ is unique for x if and only if $s \neq 0^{\otimes n}$. In other words, the output of the operation s is uniquely determined by the inputs only if $s = 0^{\otimes n}$. Therefore, we say that if $s \neq 0^{\otimes n}$, the function is two-to-one and if $s = 0^{\otimes n}$, the function is one-to-one.

Let's now consider the circuit of the algorithm. The circuit is given in Table 2. As usual, we're considering the initial state, which will go through the first $\mathcal{H}$ and we'll obtain the usual result

$$\mathcal{H}^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \tag{2.30}$$

Then we will apply our quantum function evaluation to the state on both registers:

$$\sum_{x=0}^{2^n-1} |x\rangle |0\rangle^{\otimes n} \xrightarrow{f(x)} \sum_{x=0}^{2^n-1} |x\rangle |f(x) \oplus 0\rangle = \sum_{x=0}^{2^n-1} |x\rangle |f(x)\rangle \tag{2.31}$$

Which is nothing but the state after the function evaluation and before the second $\mathcal{H}^{\otimes n}$ on the first register. We will then obtain as usual:

$$\sum_{x=0}^{2^n-1} |x\rangle |f(x)\rangle \xrightarrow{(\mathcal{H}^{\otimes n})\otimes(\mathbb{1}^{\otimes n})} \tag{2.32}$$

$$\sum_{x=0}^{2^n-1} \left[\sum_{y=0}^{2^n-1} (-1)^{x\cdot y} |y\rangle\right] |f(x)\rangle = \sum_{y=0}^{2^n-1} |y\rangle \left[\sum_{x=0}^{2^n-1} (-1)^{x\cdot y} |f(x)\rangle\right] \tag{2.33}$$

which is our state that will be measured. In fact, as usual, we've simplified the multiplicative factor. Therefore, the "correct" state that we'll be measuring, will be as the one described above times the factor $^1/_{(2^n)}$. Therefore for a certain $|y\rangle$, the probability of this happening will be given by

$$\left|\left|\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{x\cdot y} |f(x)\rangle\right|\right|^2 \tag{2.34}$$

Now we consider the 2 possible cases (whether f will be one-to-one or two-to-one).

If the function $f(x)$ is one-to-one, the Equation 2.34 will give us the probability of measuring some ket $|y\rangle$. An another way to write this probability in Equation 2.34, we can write the probability for a state $|y\rangle$ to be measured will be given by $|\langle y|x\rangle|$. Therefore, we have that the probability is given by nothing but

Now we consider the second case. That is, the case, where the function is not one-to-one. We can follow the nice trick shown in Wikipedia [9]. When the function is not one-to-one, this means therefore that for some inputs x_1 & x_2, we have the same outputs $f(x_1) = f(x_2) = z$, where z is some kind of value in the range/domain of the function f. For the specific case, one may write for the probability for some chosen $|y\rangle$ as for Equation 2.34. [9].

$$\left|\left|\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{x\cdot y} |f(x)\rangle\right|\right|^2 = \left|\left|\frac{1}{2^n} \sum_{z\in\text{Range}(f)} ((-1)^{y\cdot x_1} + (-1)^{y\cdot x_2}) |z\rangle\right|\right|^2 \tag{2.35}$$

Note that here we've changed the sum over x i.e. over sum of the domain, to the sum over the range of its values z. We now note another important property for the XOR operation. The truth table of XOR is given by the

x_1	x_2	s
0	0	0
1	0	1
0	1	1
1	1	0

Table 2: Truth table for XOR.

Table 2. From which we clearly see that $x_1 \otimes x_2 = s \iff x_1 \otimes s = x_2$. Therefore, if this is valid for one bit string, it will also be valid for any bit strings of any length, as the bitwise XOR only operates on pairs of bits. Thus, by observing the property of $x_1 \otimes x_2 = s \iff x_1 \otimes s = x_2$, we can write the Equation 2.38 as

$$\left\| \frac{1}{2^n} \sum_{z \in \text{Range}(f)} ((-1)^{y \cdot x_1} + (-1)^{y \cdot x_2}) \left| z \right\rangle \right\|^2 = \tag{2.36}$$

$$= \left\| \frac{1}{2^n} \sum_{z \in \text{Range}(f)} ((-1)^{y \cdot x_1} + (-1)^{y \cdot (x_1 \otimes s)}) \left| z \right\rangle \right\|^2 \tag{2.37}$$

$$= \left\| \frac{1}{2^n} \sum_{z \in \text{Range}(f)} ((-1)^{y \cdot x_1} + (-1)^{y \cdot x_1 \otimes y \cdot s)}) \left| z \right\rangle \right\|^2 \tag{2.38}$$

$$= \left\| \frac{1}{2^n} \sum_{z \in \text{Range}(f)} (-1)^{y \cdot x_1} (1 + (-1)^{y \cdot s}) \left| z \right\rangle \right\|^2 \tag{2.39}$$

From where, we can observe some things: if the chosen y happens to be such that $y \cdot s = 1$, then the probability (the sum given right above) will be given by 0 (the factor in front of $\left| z \right\rangle$ will be 0). Now, if the value of the product $y \cdot s = 0$, it will not be zero. instead, the sum becomes of the form

$$\left\| \frac{1}{2^n} \sum_{z \in \text{Range}(f)} (-1)^{y \cdot x_1} (2) \left| z \right\rangle \right\|^2 \tag{2.40}$$

$$\left\| \frac{1}{2^{n-1}} \sum_{z \in \text{Range}(f)} (-1)^{y \cdot x_1} \left| z \right\rangle \right\|^2 \tag{2.41}$$

Using the same trick as in Equation 2.37, we can deduce that the probability will be given by $\frac{1}{2^{n-1}}$.

From these two cases, we can therefore deduce the following: the probability of measuring a certain y if $s = 0^{\otimes n}$ (the first case) will be given by $\frac{1}{2^n}$. And the probability for a certain y if $s \neq 0^{\otimes n}$, we have 2 (sub)cases. That is, depending on whether this y will either obey $y \cdot s = 0$ or $y \cdot s \neq 0$. If it is the case of $y \cdot s = 0$, we will have 0 probability of measuring this $\left| y \right\rangle$. If, on the other hand, $y \cdot s \neq 0$, it will be given by $\frac{1}{2^{n-1}}$.

Therefore, we deduce that in both cases, for some measured y, we will have that $\underline{y \cdot s = 0}$ (as in the first case, $s = 0$ by definition and in the second, if it is not the case, the probability of measuring the y state will be zero).

During the mathematical process, starting from Equation 2.34, we considered some specific $\left| y \right\rangle$ that we'll get at the end/output. So at the end we'll get several $\left| y \right\rangle$'s with probability, depending on different cases, as discussed above. The key point here is that these states, i.e. all y's obey $y \cdot s = 0$. As provided in [9], we use the classical post processing in order to obtain the necessary s as required in the problem statement. It is important to note that, it may happen that there are "missing states", as there are sometimes 0 probability of observing some variables.

Fourier transform

The quantum Fourier algorithm is, in my opinion, the ”first useful algorithm” considered here... Lets recall the (not very formal) definition of the classical Fourier transform. It is an operation $\mathcal{F}$ transforming some function $f(x)$ to an another function $f(y)$.

$$f(x) \xrightarrow{\mathcal{F}} f(y) \ : \ f(y) = \int_{-\infty}^{+\infty} dx f(x) e^{-i2\pi yx} \tag{2.42}$$

From that, one can define the concept of the discrete fourier transform. We use the analogy/mapping of the interval to a vector. That is, we can discretize some interval into a vector of N elements $\vec{v} = (v_0, v_1, ..., v_{N-1})$. We call the discretized version of the Fourier transform, without surprise, the Discrete Fourier Transform or DFT and defined by the mapping between two vectors $\vec{x} = (x_0, x_1, ..., x_{N-1})$ to $\vec{y} = (y_0, y_1, ..., y_{N-1})$.

$$y_k = \sum_{n=0}^{N-1} x_n e^{-\frac{i2\pi}{N}kn} \tag{2.43}$$

To a very similar manner, one can define the quantum fourier transform, which, instead of function-to-function and vector-to-vector mappings, there’s quantum state to quantum state mapping. Namely, the Fourier transform maps some quantum state $|x\rangle \equiv \sum x_i |i\rangle$ to $|y\rangle = \sum y_i |i\rangle$ for some basis vectors $\{|i\rangle\}_{i\in\mathbb{N}}$. Thus we obtain the quantum Fourier transform’s definition

$$|y\rangle \xrightarrow{\mathcal{F}} |x\rangle \tag{2.44}$$

$$y_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n \omega_N^{kn} \text{ , with } \omega_N \equiv e^{i\frac{2\pi}{N}} \text{ and } k \in [0, N-1] \tag{2.45}$$

Note that here the sign of ω’s power is unimportant and is nothing but a convention. We also note the inverse Fourier transform, given by

$$|x\rangle \xrightarrow{\mathcal{F}} |y\rangle \tag{2.46}$$

$$x_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} y_n \omega_N^{-kn} \text{ , with } \omega_N \equiv e^{i\frac{2\pi}{N}} \text{ and } k \in [0, N-1] \tag{2.47}$$

as shown in beautiful Wikipedia [5], we can represent the quantum Fourier transform with

$$|x\rangle \xmapsto{\mathcal{F}_{\text{quant}}} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{xk} |k\rangle \tag{2.48}$$

Or, similarly, we can use the nice ket-bra representation of the (unitary) QFT operator [4]:

$$\mathcal{F} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \sum_{k=0}^{N-1} \omega_N^{xk} |k\rangle \langle x| \tag{2.49}$$

In addition to all that, we can also represent the quantum Fourier transform as a $(N-1) \times (N-1)$ matrix operation, acting on a vector.

$$F_N = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix} \tag{2.50}$$

Now, before considering some general cases of the Quantum Fourier transform and circuits, let's consider first some basic examples with small number of qubits and other considerations.

First of all [4], we can consider the Fourier Transform as not changing the initial state, but rather changing it to a different representation in the so-called fourier basis. That is, $\mathcal{F}|x\rangle = |\tilde{x}\rangle$.

Now let's consider one qubit system given by the most general expression $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. The one qubit, as usual consists of 2 states (in the Z-basis). We can now look at all the previous expressions written above, to have that $N = 2$. Using that, we can use, for example, the Equation 2.51, or the component-by-component expression Equation 2.48. Thus, for the 1 qubit system, we have 2 components that we'll obtain. We will denote the components of the vector/state $|y\rangle$ by y_0 and y_1. The components of the "input vector" $|\psi\rangle$ with basis $|0\rangle$ and $|1\rangle$, are given by respectively α and β. Thus, using the Equation 2.48 and the definition of ω , we have by components:

$$y_0 = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \omega_N^{kn} x_n \overset{N=2,k=0}{=} \frac{1}{\sqrt{2}} \left(\alpha e^{\frac{i2\pi}{2} \cdot 0 \cdot 0} + \beta e^{\frac{i2\pi}{2} \cdot 0 \cdot 1} \right) = \frac{1}{\sqrt{2}} (\alpha + \beta) \tag{2.51}$$

$$y_1 = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \omega_N^{kn} x_n \overset{N=2,k=1}{=} \frac{1}{\sqrt{2}} \left(\alpha e^{\frac{i2\pi}{2} \cdot 1 \cdot 0} + \beta e^{\frac{i2\pi}{2} \cdot 1 \cdot 1} \right) = \frac{1}{\sqrt{2}} (\alpha + \beta e^{i\pi}) = \frac{1}{\sqrt{2}} (\alpha - \beta) \tag{2.52}$$

With that, we see that the state $\alpha|0\rangle + \beta|1\rangle$ gets transformed to $\frac{(\alpha+\beta)}{\sqrt{2}}|0\rangle + \frac{(\alpha-\beta)}{\sqrt{2}}|1\rangle$. One may notice that this can be rewritten as $\frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle)$, which is nothing but $\alpha|+\rangle + \beta|-\rangle$. In other words, $\alpha|0\rangle + \beta|1\rangle \xrightarrow{\mathcal{F}_{N=2}} \alpha|+\rangle + \beta|-\rangle$, which is nothing but our Hadamard gate $\mathcal{H}$. Thus, the Fourier transform of 2 state qubit is the $\mathcal{H}$. As it is mentioned in [4], we can write the obtained $\alpha|0\rangle + \beta|1\rangle \xrightarrow{\mathcal{F}_{N=2}} \tilde{\alpha}|0\rangle + \tilde{\beta}|1\rangle$, which we call the *Fourier basis.*

Now, still following the path given in [4], we can try to describe this transformation for a generic n qubit case. In this case, $N = 2^n$. Then, we can write for the generic case:

$$\mathcal{F}_N \left|x\right\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{yx} \left|y\right\rangle = \tag{2.53}$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{i\frac{2\pi xy}{2^n}} \left|y\right\rangle \tag{2.54}$$

, now, we can use the fact that $\left|y\right\rangle$ represents a tensor product $\bigotimes_i \left|y_i\right\rangle$, with $\left|y_i\right\rangle \in \{0, 1\}$. We elaborate then further on the Equation 2.57:

$$\mathcal{F}_N \left|x\right\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{i2\pi(\sum_k \frac{y_k}{2^k})x} \left|y_0 y_1 y_2 ... y_n\right\rangle \tag{2.55}$$

We note however that in Equation 2.58, we used the fact that $y/2^n = \sum_k^n y_k/2^k$. This is true due to the definition of a binary number notation. Indeed, some number s can be represented in binary in the form of $s = \sum_{k=1}^n s_k 2^k$, with n being the most signicative bit. Thus, by dividing the whole expression by 2^n, we will get $s/2^n = \sum_{k=1}^n s_k 2^{k-n}$. The desired expression expression is obtained using the difference in the position of the most/least significant bits, here, followed in [4][2]. Now, continuing to expand Equation 2.58, we have:

$$\mathcal{F}_N \left|x\right\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{i2\pi(\sum_k \frac{y_k}{2^k})x} \left|y_0 y_1 y_2 ... y_n\right\rangle \tag{2.56}$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \prod_{k}^{n} e^{i2\pi\frac{y_k}{2^k}} \left|y_0 y_1 y_2 ... y_n\right\rangle \tag{2.57}$$

Now, one can write the sum $\sum_{y=0}^{N-1}$ over the $\left|y_0 y_1 y_2 ... y_k\right\rangle$ as the sum over the components y_k of the $\left|y_0 y_1 y_2 ... y_k\right\rangle$ as $\sum_{y_0=0}^{1} \sum_{y_1=0} \sum_{y_2=0} ... \sum_{y_n=0}$. The last step [3] is to simply rewrite as a product of different states. That is,

$$\mathcal{F}_N \left|x\right\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \prod_{k}^{n} e^{i2\pi\frac{y_k}{2^k}} \left|y_0 y_1 y_2 ... y_n\right\rangle = \bigotimes_{k=1}^{n} \left(\left|0\right\rangle + e^{2i\pi\frac{x}{2^k}} \left|1\right\rangle \right) \tag{2.58}$$

$CROT_k = \begin{bmatrix} \mathbb{1} & 0 \\ 0 & UROT_k \end{bmatrix}$	$UROT_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix}$

Table 3: The definitions of the $CROT_k$ and $UROT_k$ operators.

Now, we're in position to consider the Quantum circuit of the fourier transform. Before that, we'll first consider the building blocks of the QFT, that is, the 2 gates-the $CROT_k$ and the

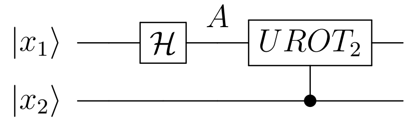
[2] It is possible to check that this is true for any representations of binary numbers not depending on LSB/MSB's position

$UROT_k$. Their representations are given in Table 3. The action of the $CROT_k$ operator can also be described as follows [4]:

$$CROT_k\,|0\psi\rangle = |0\psi\rangle \tag{2.59}$$

$$CROT_k\,|1\psi\rangle = e^{\frac{2\pi i}{2^k}\psi}\,|1\psi\rangle\,,\ \text{with}\ \ |\psi\rangle\ \ \text{is a binary string} \tag{2.60}$$

We can now consider the circuit of the QFT. First, for 2 qubits, the 2-qubit QFT: Let's now



write our usual description of the circuit, starting at some states $|x_1\rangle \otimes |x_2\rangle$. The first gate will give the state at A: $|\psi_A\rangle = \mathcal{H} \otimes \mathbb{1}(|x_1\rangle \otimes |x_2\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{2\pi i}{2}x_1}\,|1\rangle) \otimes |x_2\rangle$. Now, we can apply the $CROT_2$ operator, which, in bra-ket notation gives $CROT_2 = \mathbb{1} \otimes |0\rangle\,\langle 0| + UROT_2 \otimes |1\rangle\,\langle 1|$. Thus, by applying this operator to the state at A, we get

$$CROT_2\Bigg(\frac{1}{\sqrt{2}}\big(|0\rangle + e^{\frac{2\pi i}{2}x_1}\,|1\rangle\big) \otimes |x_2\rangle\Bigg) = \tag{2.61}$$

$$= \Bigg(\mathbb{1} \otimes |0\rangle\,\langle 0| + UROT_2 \otimes |1\rangle\,\langle 1|\Bigg)\Bigg(\frac{1}{\sqrt{2}}(|0\rangle \otimes |x_2\rangle + e^{\frac{2\pi i}{2}x_1}\,|1\rangle \otimes |x_2\rangle)\Bigg) = \tag{2.62}$$

$$= \frac{1}{\sqrt{2}}\Bigg(|0\rangle \otimes \langle 0|x_2\rangle + e^{\frac{2\pi i}{2}x_1}\,|1\rangle \otimes \langle 0|x_2\rangle)\Bigg) + \tag{2.63}$$

$$+\frac{1}{\sqrt{2}}\Bigg(|0\rangle \otimes \langle 1|x_2\rangle + [e^{\frac{2\pi i}{2}x_2} + e^{\frac{2\pi i}{2^2}x_2}] \otimes \langle 1|x_2\rangle\Bigg) \tag{2.64}$$

which will give different outcomees for different inputs. Equivalently, in a less messy manner, [4], we can write:

$$CROT_2\Bigg(\frac{1}{\sqrt{2}}\big(|0\rangle + e^{\frac{2\pi i}{2}x_1}\,|1\rangle\big) \otimes |x_2\rangle\Bigg) \overset{\text{ctrl.-2, targ.-1}}{=} \tag{2.65}$$

$$= \frac{1}{\sqrt{2}}\Bigg(|0\rangle + [e^{\frac{2\pi i}{2}x_1} + e^{\frac{2\pi i}{2^2}x_2}]\,|1\rangle\Bigg) \otimes |x_2\rangle \tag{2.66}$$

With this idea, let's finally try to generalize it for a more generic input of n qubits. This by *replacing* the $\otimes\,|x_2\rangle$ by $\otimes\,|x_2x_3...x_n\rangle$. Therefore, the state after the first $UROT_2$ will be given by $\frac{1}{\sqrt{2}}\Bigg(|0\rangle + [e^{\frac{2\pi i}{2}x_1} + e^{\frac{2\pi i}{2^2}x_2}]\,|1\rangle\Bigg) \otimes |x_2x_3...x_n\rangle$ Then, the next step will undergo the next n controlled $UROT_k$ operators with the k qubit being the controlled qubit and the first qubit being the target. After all the n gates (using the same logic as for the 2-qubit systems), we will get the state [4]:

$$\frac{1}{\sqrt{2}}\Bigg(|0\rangle + [e^{\frac{2\pi i}{2}x_1} + e^{\frac{2\pi i}{2^2}x_2}\ ... + ...\ e^{\frac{2\pi i}{2^{n-1}}x_{n-1}} + e^{\frac{2\pi i}{2^n}x_n}]\,|1\rangle\Bigg) \otimes |x_2x_3...x_n\rangle \tag{2.67}$$

Now, we can notice the already mentioned *trick* in Equation 2.58. That is, the fact that in binary representation, $x = 2^{n-1}x_1 + 2^{n-2}x_2 + ... + 2^1 x_{n-1} + 2^0 x_n$. Thus, the sum expression in the square brackets (factor of $x_j/2^j$) is nothing but $x/2^n$, giving us the state

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right) \otimes |x_2 x_3 ... x_n\rangle \tag{2.68}$$

So up to now we performed the following things on the 1-n qubits: Apply the $\mathcal{H}$ followed by n $UROT_k$ operators with the first qubit as the target **and** the $[\![1, n]\!]$ [3] being the control ones.

The next steps are exactly the same, but now, the 2nd qubit becoms the target and the $[\![2, n]\!]$ become the controls. This is performed then for $[\![3, n]\!]$, $[\![4, n]\!]$, etc... It can be easily shown from the Equation 2.71, that, by applying subsequent operations described above, we will subsequently get

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right) \otimes |x_2 x_3 ... x_n\rangle \rightarrow \tag{2.69}$$

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right) \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^{n-1}}x}|1\rangle\right) \otimes |x_3 ... x_n\rangle \rightarrow ... \tag{2.70}$$

$$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right) \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^{n-1}}x}|1\rangle\right) \otimes ... \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{\frac{2\pi i}{2^0}x}|1\rangle\right) \tag{2.71}$$

Let's finally try to implement and comment the QFT circuit and make brief comments on them (I'll write U_k instead of $UROT_k$ for space saving's reasons)

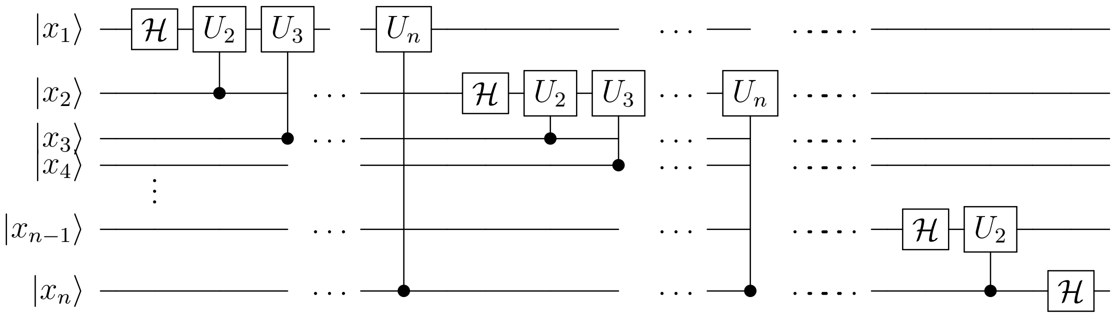


Table 4: The QFT algorithm, where successive $UROT_k$ operations are performed. The circuit in fact is composed of 2 types of building blocks. We first chose the target qubit and then apply the $UROT_k$ operation with k being the control qubit of those successive operations. This operation (where the 1 qubit is the target is described in the Equation 2.72). Then, we proceed with the second qubit to be the target and we perform the same operation now for the $[\![2, n]\!]$ qubits as control. (as per Equation 2.73). We proceed then for the rest of the qubits as targets, to finally arrive at the last qubit $n - 1$ and n. The result of these operations is given in Equation 2.74.

[3] The notation $[\![1, n]\!]$ means a discrete interval. For example, $[\![2, 5]\!] \equiv 2, 3, 4, 5$

Phase estimation

The phase estimation is an important algorithm, that does what its name suggests - to estimate the phase of a system. That it, suppose there exists a state $|\psi\rangle$. Suppose then there exists an operator $U|\psi\rangle = e^{2i\pi\theta}|\psi\rangle$. The goal of the algorithm is thus to find the eigenvalue's $e^{2i\pi\phi\theta}$ argument θ. The input of the algorithm is therefore the state $|\psi\rangle$, as well as an input vector $|0\rangle^{\otimes^m}$, which server as the counting vector. So this $|0\rangle^{\otimes^m}$ will accumulate the counting of the phase θ.

Let's consider the general circuit of the phase estimation algorithm [6] [7] and try to mathematically analyze the circuit in a more mathematical way.

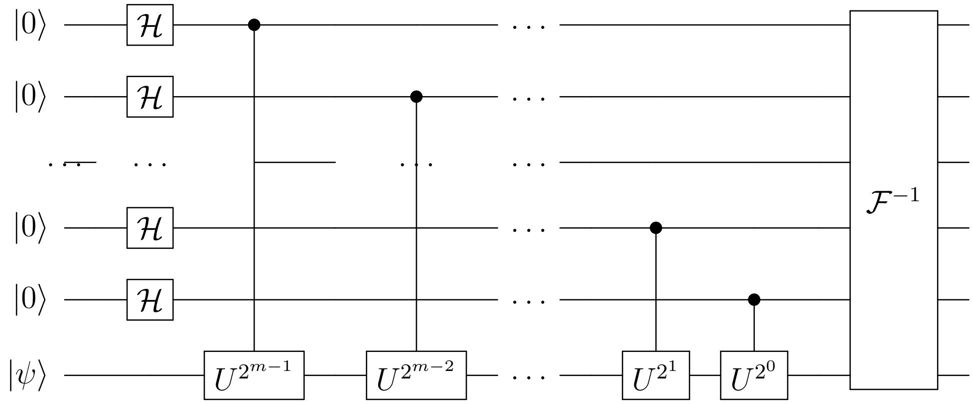


Table 5

Note that there are some similarities with the QFT, that is, the operator U raised to the powers 2^k.

So as usual, we have that the initial state is given by nothig but $|0\rangle^{\otimes^m} \otimes |\psi\rangle$. Then, the next step is to apply the $\mathcal{H}$ on the counting bits. We have then

$$|\psi\rangle^{\otimes m}|\psi\rangle \xrightarrow{\mathcal{H}^{\otimes^m}\otimes\mathbb{1}} \frac{1}{2^{2/n}}\big(|0\rangle + |1\rangle\big)^{\otimes^m} \otimes |\psi\rangle \tag{2.72}$$

Then, we sequencially apply the different U^{2^k} operators. This is similar to the operators found in the QFT. As the eigenvalue of the U operator is given by $e^{2\pi i\theta}$ the power of this operator can be expressed by:

$$U^{2^k}|\psi\rangle = e^{2\pi i 2^k\theta}|\psi\rangle \tag{2.73}$$

Therefore, the consecutive applications of these U^k are given below. We should also remember, that the controlled U operator in the case of e.g. 2 qubits with the 0th as the control and the 1st as the target, will be given by $U = |0\rangle \langle 0| \otimes \mathbb{1} + |1\rangle |1\rangle \otimes U$

$$\frac{1}{2^{\frac{n}{2}}}\big(|0\rangle + |1\rangle\big)^{\otimes^m} \otimes |\psi\rangle \xrightarrow{\text{apply the controlled } U^{2^{m-1}}} \tag{2.74}$$

$$\frac{1}{2^{\frac{n}{2}}}\left(|0\rangle \langle 0| \otimes (\mathbb{1}^{\otimes^{m-1}}) \otimes \mathbb{1} + |1\rangle \langle 1| \otimes (\mathbb{1}^{\otimes^{m-1}}) \otimes U\right)\frac{1}{2^{\frac{n}{2}}}\left[\frac{1}{2^{\frac{2}{n}}}\big(|0\rangle + |1\rangle\big) \otimes \big(|0\rangle + |1\rangle\big)^{\otimes^{m-1}} \otimes |\psi\rangle\right] = \tag{2.75}$$

$$= \frac{1}{2^{\frac{n}{2}}}\left[|0\rangle \otimes (|0\rangle + |1\rangle)^{\otimes^{m-1}} \otimes |\psi\rangle + |1\rangle \otimes (|0\rangle + |1\rangle)^{\otimes^{m-1}} \otimes U\,|\psi\rangle\right] = \tag{2.76}$$

$$\frac{1}{2^{\frac{n}{2}}}\left[|0\rangle \otimes (|0\rangle + |1\rangle)^{\otimes^{m-1}} \otimes |\psi\rangle + |1\rangle \otimes (|0\rangle + |1\rangle)^{\otimes^{m-1}} \otimes e^{2\pi i 2^{m-1}\theta}\,|\psi\rangle\right] = \tag{2.77}$$

$$= \frac{1}{2^{\frac{n}{2}}}\left[|0\rangle + e^{2\pi i 2^{m-1}\theta}\,|1\rangle\right] \otimes \left[|0\rangle + |1\rangle\right]^{\otimes^{m-1}} \otimes |\psi\rangle \tag{2.78}$$

This application can be performed several times. At the end, using the similar development, it is possible to show that after the consecutive applications of the U operator, we obtain at the end

$$\frac{1}{2^{\frac{2}{n}}}\left[|0\rangle + e^{2\pi i 2^{m-1}\theta}\,|1\rangle\right] \otimes \left[|0\rangle + e^{2\pi i 2^{m-2}\theta}\,|1\rangle\right] \otimes \cdots \otimes \left[|0\rangle + e^{2\pi i 2^{0}\theta}\,|1\rangle\right] \otimes |\psi\rangle = \tag{2.79}$$

$$= \frac{1}{2^{\frac{2}{n}}}\left[\sum_{k=0}^{2^n-1} e^{2\pi i k\theta}\,|k\rangle\right] \otimes |\psi\rangle \tag{2.80}$$

So this is the state before we apply the inverse Fourier transform. However, let's first quickly recall how does the simple Fourier transform work.

$$|y\rangle \mapsto \left(|0\rangle + e^{\frac{2\pi i}{2^1}x}\,|1\rangle\right) \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^2}x}\,|1\rangle\right) \cdots \left(|0\rangle + e^{\frac{2\pi i}{2^n}x}\,|1\rangle\right) = \tag{2.81}$$

$$|y\rangle \mapsto \left(\sum_{k=0}^{N-1} \omega_N^{yk}\,|k\rangle\right) \tag{2.82}$$

The inverse fourier transform simply swaps the sign of the exponential of the ω_N. That is, we can define the operator of the inverse Fourier transform:

$$|y\rangle \mapsto \left(\sum_{k=0}^{N-1} \omega_N^{-yk}\,|k\rangle\right) \tag{2.83}$$

and the operator:

$$\widehat{\mathcal{F}^{-1}} = \frac{1}{2^{\frac{n}{2}}}\sum_{x=0}^{2^n-1}\sum_{k=0} \omega_N^{-xk}\,|k\rangle \langle x| \tag{2.84}$$

$$\widehat{\mathcal{F}^{-1}} = \frac{1}{2^{\frac{n}{2}}}\sum_{x=0}^{2^n-1}\sum_{k=0} e^{-\frac{2\pi i x k}{2^n}}\,|k\rangle \langle x| \tag{2.85}$$

So we have that the state that the $\mathcal{F}^{-1}$ must be applied to is given by

$$\frac{1}{2^{\frac{n}{2}}}\left[\sum_{k=0}^{2^n-1} e^{2\pi i k\theta}\left|k\right\rangle\right]\otimes\left|\psi\right\rangle \equiv \frac{1}{2^{\frac{n}{2}}}\left[\sum_{y=0}^{2^n-1} e^{2\pi i y\theta}\left|y\right\rangle\right] \tag{2.86}$$

Let's now try to apply the inverse Fourier in it:

$$\begin{aligned}
\widehat{\mathcal{F}^{-1}} = \frac{1}{2^{\frac{n}{2}}}\left[\sum_{y=0}^{2^n-1} e^{2\pi i y\theta}\left|y\right\rangle\right] =\\
\frac{1}{2^n}\sum_{x=0}^{2^n-1}\sum_{k=0} e^{-\frac{2\pi i x k}{2^n}}\left|k\right\rangle\left\langle x\right|\left[\sum_{y=0}^{2^n-1} e^{2\pi i y\theta}\left|y\right\rangle\right] =\\
\frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{-\frac{2\pi i x k}{2^n}} e^{2\pi i y\theta}\left|k\right\rangle\left\langle x|y\right\rangle \overset{\langle x|y\rangle=\delta_{xy}}{=}\\
\frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{-\frac{2\pi i x k}{2^n}} e^{2\pi i y\theta}\left|k\right\rangle\left\langle x|y\right\rangle \overset{\langle x|y\rangle=\delta_{xy}}{=}\\
\frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{-\frac{2\pi i y k}{2^n}} e^{2\pi i y\theta}\left|k\right\rangle = \frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{-2\pi i y(\frac{k}{2^n}-\theta)}\left|k\right\rangle =\\
= \frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{\frac{-2\pi i y}{2^n}(k-\theta 2^n)}\left|k\right\rangle
\end{aligned} \tag{2.87}$$

Which is the last state after the inverse Fourier transofrm. To be more precise,

$$\frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{k=0}^{2^n-1} e^{\frac{-2\pi i y}{2^n}(k-\theta 2^n)}\left|k\right\rangle\otimes\left|\psi\right\rangle \tag{2.88}$$

Now, we can ask ourselves, how are distributed the probabilities of measuring this final state? It is clear, that if $k = \theta 2^n$, the probability of measuring the state $\left|k\right\rangle = \left|\theta 2^n\right\rangle$ is the maximum. Indeed, in this case, the exponential, $e^{\theta 2^n - (k=)\theta 2^n}$ will have the maximum value.

Therefore, at the end, we will know what is theta, by measuring the first m qubits.

3 Shor's algorithm

The Shor's algorithm is one of the most famous algorithms in quantum computing. It has the potential ability to factor integers with polynomial time. The process of integer factoring is very important in computer security, that is, in cryptography. We will however not go into great details into the deep mathematics behind cryptography, but only what it is necessary for the understanding of the algorithm.

To consider the Shor's algorihtm, we define the function that is working with natural numbers and with modular arithmetic [4]. We define therefore the function

$$f(x) = a^x \bmod N \tag{3.1}$$

$$f(x) \overset{\text{mod } N}{\equiv} a^x \tag{3.2}$$

We note that this function is clearly periodic. In addition to that, one also defines the notion of *period* for such function, denoting t or r [8], which is the smallest number t such that

$$f(x = t) = a^t \bmod N = 1 \tag{3.3}$$

$$f(x = t) = 1 \overset{\text{mod } N}{=} a^t \tag{3.4}$$

Now, we get to the quantum mechanical description of this algorithm. One defines as usual the (unitary) operator U:

$$U|y\rangle = |ay \bmod N\rangle \tag{3.5}$$

Let's check how this operator acts sequentially for the example of $a = 5$ and N=28...

$$\begin{aligned} U|1\rangle &= |5\rangle \\ U^2|1\rangle = U|5\rangle = |5 \cdot 5 \bmod 28\rangle &= |3\rangle \\ U^3|1\rangle = U|3\rangle = |5 \cdot 3 \bmod 28\rangle &= |13\rangle \\ U^4|1\rangle = U|13\rangle = |5 \cdot 13 \bmod 28\rangle &= |9\rangle \\ U^5|1\rangle = U|9\rangle = |5 \cdot 9 \bmod 28\rangle &= |17\rangle \\ U^6|1\rangle = U|17\rangle = |5 \cdot 17 \bmod 28\rangle &= |1\rangle \end{aligned} \tag{3.6}$$

We thus again came back to the state $|1\rangle$. This means that for the case of these numbers (i.e. $a = 5$ and $N = 28$), the period of the function is 6. That is, $5^6 \overset{\text{mod } 28}{\equiv} 1$.

[4] Without going into detail of modular arithmetic, let's give some examples, that will help us to understand it. $44 \overset{\text{mod } 7}{\equiv} 2$, as $44 = 6 \cdot 7 + 2$. $34 \overset{\text{mod } 9}{=} 7$, as $34 = 3 \cdot 9 + 7$. So this is the number that must be added to the maximum number of times that the first number will go into second (e.g. 7 or 9 will go into 44 or 34 respectively).

In a similar fashion, one can define a vector, having an interesting property. Namely, being the eigenvector of this operator. The eigenvector is in fact the superposition of the aformentioned vectors. The superposition is given by

$$|u\rangle = \sum_{k=0}^{r-1} \left|a^k \text{ mod } N\right\rangle \tag{3.7}$$

In the specific case of $a = 5$ and $N = 28$, we have that

$$|u\rangle = \frac{1}{4}\Big(|1\rangle + |13\rangle + |9\rangle + |17\rangle\Big) \tag{3.8}$$

It is indeed an eigenket:

$$\begin{aligned} U|u\rangle = U\frac{1}{4}\Big(|1\rangle + |13\rangle + |9\rangle + |17\rangle\Big) &= \\ \frac{1}{4}\Big(U|1\rangle + U|13\rangle + U|9\rangle + U|17\rangle\Big) &= \\ \frac{1}{4}\Big(|13\rangle + |9\rangle + |17\rangle + |1\rangle\Big) &= \\ \frac{1}{4}\Big(|1\rangle + |13\rangle + |9\rangle + |17\rangle\Big) &= |u\rangle \end{aligned} \tag{3.9}$$

We can show it for a general case for the $|u\rangle = \sum_{k=0}^{r-1} \left|a^k \text{ mod } N\right\rangle$. The main property of some state $|a^{r-1} \text{ mod } \rangle$ is $U|a^{r-1} \text{ mod } \rangle = |a^r \text{ mod } \rangle = |1\rangle$.

So we show that this is indeed the eigenket of the operator:

$$\begin{aligned} U|u\rangle = U\sum_{k=0}^{r-1}\left|a^k \text{ mod } N\right\rangle = \sum_{k=0}^{r-1}\left|a^{k+1} \text{ mod } N\right\rangle &= \\ = \sum_{k'=1}^{r-1}\left|a^k \text{ mod } N\right\rangle + \left|a^{r=(k+1)} \text{ mod } N\right\rangle &= \\ \sum_{k'=1}^{r-1}\left|a^k \text{ mod } N\right\rangle + \left|a^{0(=k)} \text{ mod } N\right\rangle = \sum_{k=0}^{r-1}\left|a^k \text{ mod } N\right\rangle \end{aligned} \tag{3.10}$$

We can now consider another state:

$$|u_1\rangle = \sum_{k=0}^{r-1} e^{-\frac{2\pi i k}{r}} \left|a^k \text{ mod } N\right\rangle \tag{3.11}$$

In fact, it is also an eigenstate of the operator U. One can show that the vector in Equation 3.11 is also an eigenket of the operator U.

$$\begin{aligned} U\sum_{k=0}^{r-1} e^{-\frac{2\pi ik}{r}} \left|a^k \bmod N\right\rangle = \sum_{k=0}^{r-1} e^{-\frac{2\pi ik}{r}} \left|a^{k+1} \bmod N\right\rangle = \\ = e^{\frac{2\pi i}{r}} \sum_{k=0}^{r-1} e^{-\frac{2\pi i(k+1)}{r}} \left|a^{k+1} \bmod N\right\rangle \end{aligned} \tag{3.12}$$

We now do the similar trick as for Equation 3.10. To obtain

$$e^{\frac{2\pi i}{r}} \sum_{k=0}^{r-1} e^{-\frac{2\pi i(k+1)}{r}} \left|a^{k+1} \bmod N\right\rangle = e^{\frac{2\pi i}{r}} \sum_{k=0}^{r-1} e^{-\frac{2\pi ik}{r}} \left|a^{k} \bmod N\right\rangle \tag{3.13}$$

In fact, in the general case, one can write the following:

$$U\sum_{k=0}^{r-1} e^{-\frac{2\pi isk}{r}} \left|a^k \bmod N\right\rangle \equiv U\left|u_s\right\rangle = e^{\frac{2\pi is}{r}} \left|u_s\right\rangle \tag{3.14}$$

The idea here is that the state $|u_s\rangle$ is an eigenvector of the operator U and its eigenvalue is given by $e^{\frac{2\pi is}{r}}$. Therefore, one can remember the fact that in section 2, we were estimating the phase of the eigenvalue of the operator U. In this case, the phase θ is given by $\theta = \frac{s}{r}$. Therefore, one can estimate the phase, thus obtain the so-wanted period r.

Grover's algorithm

The Grover's algorithm is the first algorithm to mention when discussing sorting. The Grover's algorithm is a sorting algorithm that uses a quantum oracle. We recall that the quantum oracle is a certain function/operator that we consider to exist. In the case of the Grover's algorithm, one defines the Grover's oracle:

$$U_\omega(x) = \begin{cases} |x\rangle \text{ if } x \neq \omega \\ -|x\rangle \text{ if } x = \omega \end{cases} \tag{3.15}$$

We remeber that an oracle is a function that is able to verify some kind of solution. From the Equation 3.15, one can say that the oracle operator corresponds to the function

$$U_\omega |x\rangle = (-1)^{f(x)} |x\rangle, \quad \text{with } f(x) = \begin{cases} 0 \text{ if } x \neq \omega \\ 1 \text{ if } x = \omega \end{cases} \tag{3.16}$$

Next, one defines the so-called *Grover's diffusion operator*, sometimes referred to as the phase amplification operator. It is defined as

$$U_s = 2|s\rangle\langle s| - \mathbb{1} \tag{3.17}$$

, where $|s\rangle$ is a superposition that will be defined further.

We also define the operator $\widetilde{U_s}$ to simplyfy notation:

$$\widetilde{U_s} = 2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - \mathbb{1} \tag{3.18}$$

So now, let's consider the circuit itself. We will try to give the mathematical proof of it based on the circuit itself.

On Table 6, there are 2 fully equivalent circuits. Namely, in the right circuit, the operator $\widetilde{U_s} = 2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - \mathbb{1}$. If we apply to some arbitrary state $|\psi\rangle$ first the Hadamard then the U_s then the Hadamard again, we get (we remember that an operator is applied to the right):

$$\begin{aligned} |\psi\rangle \xrightarrow{\mathcal{H}} \mathcal{H}|\psi\rangle \xrightarrow{U_s} \Big[2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - \mathbb{1}\Big]\mathcal{H}|\psi\rangle = \\ \Big[2|0\rangle^{\otimes n}\langle 0|^{\otimes n}\mathcal{H} - \mathcal{H}\Big]|\psi\rangle \overset{\mathcal{H}|0\rangle^{\otimes n}=|s\rangle}{=} \\ \Big[2|0\rangle^{\otimes n}\langle s| - \mathcal{H}\Big]|\psi\rangle \xrightarrow{\mathcal{H}\text{ again}} \\ \mathcal{H}\Big[2|0\rangle^{\otimes n}\langle s| - \mathcal{H}\Big] = \Big[2\mathcal{H}|0\rangle^{\otimes n}\langle s| - \mathcal{H}\mathcal{H}\Big] \overset{\mathcal{H}^2=\mathbb{1}}{=} 2|s\rangle\langle s| - \mathbb{1} \end{aligned} \tag{3.19}$$

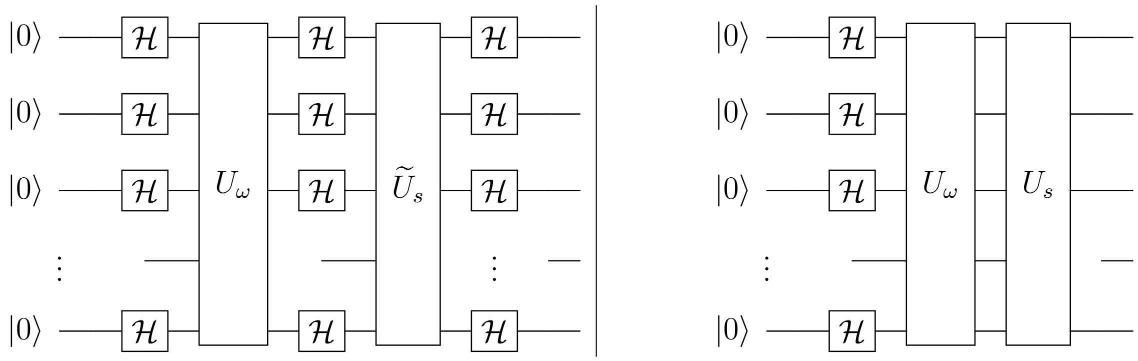


Table 6: The basic Grover's algorithm step. In order to get better results, i.e. to emphasize the element $|\omega\rangle$ that we're looking for, we would perform multiple of these Grover's steps. Note that there are 2 equivalent cirquits.

Lets now try to brake down the steps of the algorihtm. As usual, we're preparing the the initial state in the form $|0\rangle^{\otimes n}$. The application of the $\mathcal{H}^{\otimes n}$ gives the state $|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$. Now it's time to apply the Grover's operator. The Grover's operator (as defined in Equation 3.16) will act on the state $|s\rangle$.

$$U_\omega |s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}(-1)^{f(x)}|x\rangle = \frac{1}{\sqrt{N}}\Bigg(\sum_{x=0}^{\omega-1}|x\rangle + \sum_{x=\omega+1}^{N-1}|x\rangle \quad - |\omega\rangle\Bigg) \coloneqq \frac{1}{\sqrt{N}}|\phi\rangle \coloneqq \overline{|\phi\rangle} \tag{3.20}$$

Now it is time to apply the Grover's diffusion operator on the previously obtained state denoted as $\overline{|\phi\rangle}$.

The diffusion operator is defined in Equation 3.17. We recall that the state $|s\rangle$ is defined as $|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$. The operator is therefore,

$$\begin{aligned} U_s\overline{|\phi\rangle} = 2|s\rangle\langle s|\overline{|\phi\rangle} - \mathbb{1}\overline{|\phi\rangle}\frac{1}{\sqrt{N}}\Big(2|s\rangle\langle s|\phi\rangle - \mathbb{1}|\phi\rangle\Big) &\xrightarrow{\text{Equation 3.20}} \\ \frac{1}{\sqrt{N}}\left[\frac{1}{N}\cdot 2\Big(\sum_{x=0}^{N-1}|x\rangle\Big)\cdot\Big(\sum_{x=0}^{N-1}\langle x|\Big) - \mathbb{1}\right]|\phi\rangle &= \\ \frac{1}{\sqrt{N}}\left[\frac{1}{N}\cdot 2\Big(\sum_{x=0}^{N-1}|x\rangle\Big)\cdot\Big(\sum_{x=0}^{N-1}\langle x|\Big) - \mathbb{1}\right]|\phi\rangle &= \\ \frac{1}{\sqrt{N}}\left[\frac{1}{N}\cdot 2\Big(\sum_{x=0}^{N-1}|x\rangle\Big)\cdot\Big(\sum_{x=0}^{N-1}\langle x|\Big)|\phi\rangle - |\phi\rangle\right] & \end{aligned} \tag{3.21}$$

Let's try to break down the individual components. The term we want to break down is

$$\begin{aligned}\Big(\sum_{x=0}^{N-1}\langle x|\Big)|\phi\rangle = \Big(\sum_{x=0}^{N-1}\langle x|\Big)\cdot\Big(\sum_{x=0}^{N-1}(-1)^{f(x)}|x\rangle\Big) = \Big(\sum_{x=0}^{N-1}\langle x|\Big)\cdot\Big(\sum_{y=0}^{N-1}(-1)^{f(y)}|y\rangle\Big) = \\ = \Big(\sum_{x=0}^{N-1}\langle x|\Big)\cdot\Big(\sum_{y=0}^{\omega-1}|y\rangle + \sum_{y=\omega+1}^{N-1}|y\rangle \; - |\omega\rangle\Big) \overset{\langle x|y\rangle=\delta_{x,y}}{=} (\omega + (N-\omega-1)+1) = N\end{aligned} \tag{3.22}$$

The last equality is obtained using the orthogonality and the fact that we're summing from 0 to $\omega - 1$, giving ω and from $\omega + 1$ to $N - 1$ giving $N - \omega + 1$ and the last scalar product gives simply -1. Therefore, by plugging the result into Equation 3.21.

$$\frac{1}{\sqrt{N}}\left[\frac{1}{N}\cdot 2\Big(\sum_{x=0}^{N-1}|x\rangle\Big)\cdot N - |\phi\rangle\right] = \frac{1}{\sqrt{N}}\left[2\Big(\sum_{x=0}^{N-1}|x\rangle\Big) - |\phi\rangle\right] \tag{3.23}$$

Now, it is quite straightforward to simplify the equation, as we remember that $|\phi\rangle$ differs from $\sum_{x=0}^{N-1}|x\rangle$ by the factor in front of $|\omega\rangle$. We therefore obtain the resulting expression:

$$\begin{aligned}\frac{1}{\sqrt{N}}\left[2\Big(\sum_{x=0}^{N-1}|x\rangle\Big) - |\phi\rangle\right] = \\ \frac{1}{\sqrt{N}}\left[\Big(\sum_{x=0}^{N-1}|x\rangle\Big) + \Big(\sum_{x=0}^{N-1}|x\rangle\Big) - |\phi\rangle\right] = \frac{1}{\sqrt{N}}\left[\Big(\sum_{x=0}^{N-1}|x\rangle\Big) + 2|\omega\rangle\right]\end{aligned} \tag{3.24}$$

Now, based on Table 6 left circuit, we're done applying gates and we can now try to measure and try to understand what we get at the output. We can easily rewrite the last equation as

$$\begin{aligned}\frac{1}{\sqrt{N}}\left[\Big(\sum_{x=0}^{N-1}|x\rangle\Big) + 2|\omega\rangle\right] = \\ \frac{1}{\sqrt{N}}\left[\Big(\sum_{x=0}^{\omega-1}|x\rangle + \sum_{x=\omega+1}^{N-1}|x\rangle + |\omega\rangle\Big) + 2|\omega\rangle\right] = \\ \frac{1}{\sqrt{N}}\left[\Big(\sum_{x=0}^{\omega-1}|x\rangle + \sum_{x=\omega+1}^{N-1}|x\rangle\Big) + 3|\omega\rangle\right] := |\eta\rangle\end{aligned} \tag{3.25}$$

From that, we can obtain the probability of obtaining the state $|\omega\rangle$ by taking the inner product $|\langle\eta|\omega\rangle|^2$. We know that $\langle x|\omega\rangle$ will be zero if $|x\rangle \neq |\omega\rangle$. We see that the probability of observing the state $|\omega\rangle$ is greater than any other states. In fact, one can check (I am not covering it here) that by repeating this process more than once, the probability of observing the element $|\omega\rangle$ will be increase. There exists a theorem that states that the number of these kind iterations must be of order $\mathcal{O}(\sqrt{N})$, where N is the number of entry qubits.

References

[1] *Bernstein–Vazirani algorithm.* In: *Wikipedia.* Page Version ID: 1121289156. Nov. 11, 2022. URL: https://en.wikipedia.org/w/index.php?title=Bernstein%E2%80%93Vazirani_algorithm&oldid=1121289156 (visited on 01/31/2023).

[2] *Deutsch–Jozsa algorithm.* In: *Wikipedia.* Page Version ID: 1122661600. Nov. 18, 2022. URL: https://en.wikipedia.org/w/index.php?title=Deutsch%E2%80%93Jozsa_algorithm&oldid=1122661600 (visited on 01/26/2023).

[3] Artur Ekert {and} Tim Hosgood. *Introduction to Quantum Information Science.* URL: https://qubit.guide/ (visited on 01/26/2023).

[4] *Quantum Fourier Transform.* URL: https://community.qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html (visited on 02/13/2023).

[5] *Quantum Fourier transform.* In: *Wikipedia.* Page Version ID: 1123746557. Nov. 25, 2022. URL: https://en.wikipedia.org/w/index.php?title=Quantum_Fourier_transform&oldid=1123746557 (visited on 02/04/2023).

[6] *Quantum Phase Estimation.* URL: https://community.qiskit.org/textbook/ch-algorithms/quantum-phase-estimation.html (visited on 02/27/2023).

[7] *Quantum phase estimation algorithm.* In: *Wikipedia.* Page Version ID: 1133698428. Jan. 15, 2023. URL: https://en.wikipedia.org/w/index.php?title=Quantum_phase_estimation_algorithm&oldid=1133698428 (visited on 03/06/2023).

[8] *Shor's Algorithm.* URL: https://community.qiskit.org/textbook/ch-algorithms/shor.html (visited on 03/09/2023).

[9] *Simon's problem.* In: *Wikipedia.* Page Version ID: 1133798643. Jan. 15, 2023. URL: https://en.wikipedia.org/w/index.php?title=Simon%27s_problem&oldid=1133798643 (visited on 01/31/2023).