Shor's Algorithm and Grover's Algorithm in Quantum Computing

By

Chris Valle

Submitted to the graduate degree program in Mathematics and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Arts.

_____

Chairperson Dr. Hongguo Xu

_____

Dr. Erik Van Vleck

_____

Dr. Weizhang Huang

Date Defended: Feburary 4, 2011

The Thesis Committee for Chris Valle

certifies that this is the approved version of the following thesis:

Shor's Algorithm and Grover's Algorithm in Quantum Computing

_____

Chairperson Hongguo Xu

Date approved: April 25, 2011

Abstract

In this paper we will analyse two quantum algorithms that sparked interest in the potential of quantum computers. The first is Lov Grover's algorithm which may be used to conduct a type of database search. The second is Peter Shor's algorithm which may be used to factor large numbers and provides an exponential speed up over the best current classical algorithms. In the context of these two algorithms we will discuss the benefits and weaknesses of quantum computation. We will show that in exchange for a quantum computer's greater speed we must accept an inherent level of uncertainty in our results.

Contents

1 Introduction **4**
1.1 Preliminaries 5
1.2 The model 6

2 Some useful quantum logic gates **8**
2.1 The Hadamard gate 8
2.2 The controlled phase shift gate 13
2.3 The Toffoli gate 13

3 Grover's Algorithm **14**
3.1 Why it works 15
3.1.1 An outline 15
3.1.2 The details 15
3.2 Computational cost 17

4 Shor's algorithm **19**
4.1 The Algorithm 19
4.2 Determining a non-trivial factor of n from the order of $x \bmod n$. 19
4.3 Modular exponentiation 22
4.4 The quantum fourier transform 22
4.5 Determining the order of $x \pmod n$ 24
4.6 A lower bound for observing a sufficient state to compute the order of $x \pmod n$ 26
4.7 Computational cost 31

5 Conclusion **33**

Bibliography **33**

1 Introduction

The mid 1990's saw the discovery of two quantum algorithms that reignited interest in quantum computing and its emerging potential. The first was introduced in 1994 by Peter Shor and provided a quantum alternative to factoring large numbers. Shor's algorithm achieved an exponential speed up over the best known classical algorithm. The second was introduced by Lov Grover in 1997 as a remarkable search algorithm that achieved a quadratic speed up over its classical counterpart.

Each of these algorithms would operate on a fundamentally different level than their classical counterparts, exploiting the novel use of quantum superposition. We will see that the characteristic of superposition is the key to the power of quantum computation. This essentially allows a quantum computer to perform parallel processing with little additional resources. There is however a very profound trade-off: in exchange for the quantum speed up we must accept an inherent uncertainty in our results.

At first it was thought that this uncertainty would render quantum computers impractical. It was eventually discovered however that for certain types

of problems the power of quantum computation could be harnessed while overcoming the issue of uncertainty. Shor's and Grover's algorithms were some of the first algorithms that successfully achieved this goal. The details of these algorithms will be described in this paper.

1.1 Preliminaries

Below we take $\mathbb{N}$ to denote the set $\{0, 1, 2, 3, ...\}$, $\mathbb{R}$ to denote the set of real numbers, and $\mathbb{C}$ to denote the set of complex numbers.

For any set $\mathbb{F}$ let $\mathbb{F}^{m\times n}$ denote the set of $m \times n$ matrices with entries in $\mathbb{F}$ and let $\mathbb{F}^n := \mathbb{F}^{n\times 1}$.

Let $||\cdot||$ denote the Euclidean norm or matrix 2-norm.

Let $A \in \mathbb{C}^{m\times n}$ and let A' denote the conjugate transpose of A. A is Unitary if

$$A'A = AA' = I_n$$

where I_n denotes the $n \times n$ identity matrix. We will omit n if the dimension is clear.

Throughout this paper we will use Bra-ket notation for vectors in $\mathbb{C}^n$. If $|x\rangle, |y\rangle \in \mathbb{C}^n$ then $\langle x|y\rangle$ and $|x\rangle\langle y|$ denote the inner and outer product respectively. For the inner product we may also use the notation $\langle x, y\rangle$.

If $A \in \mathbb{C}^{n\times n}$ is unitary and $|x\rangle, |y\rangle \in \mathbb{C}^n$ then we have

$$\langle A|x\rangle, A|y\rangle\rangle = \langle x|A'A|y\rangle = \langle x, y\rangle\,, \tag{1}$$

thus the application of A preserves angles. Clearly the product of unitary matrices is also unitary.

For a vector $|q\rangle \in \mathbb{C}^n$ with $|||q\rangle|| = 1$ the Householder matrix H_q is defined to be

$$H_q := I - 2|q\rangle\langle q|.$$

This application of H_q may be interpreted as a reflection about the hyperplane orthogonal to $|q\rangle$, and the application of $-H_q$ may be interpreted as a reflection about $|q\rangle$. Note that $H_q^{-1} = H_q$ since

$$(I - 2|q\rangle\langle q|)(I - 2|q\rangle\langle q|) = I - 4|q\rangle\langle q| + 4|q\rangle\langle q| = I\,.$$

From this we also see that H_q is unitary since $H_q' = H_q$.

Note that $|q\rangle$ may be extended to an orthonormal basis $\{|q\rangle, |q_2\rangle, |q_3\rangle, ..., |q_n\rangle\}$ of $\mathbb{C}^n$ and that

$$H_q|q\rangle = (1 - 2|q\rangle\langle q|)|q\rangle = -|q\rangle$$

and

$$H_q|q_j\rangle = (1 - 2|q\rangle\langle q|)|q_j\rangle = |q_j\rangle\,.$$

for $j = 2, ..., n$. Thus the eigenspace corresponding to 1 is of dimension $n-1$, and so the eigenspace corresponding to -1 is of dimension 1, i.e a simple eigenspace. Likewise $-H_q$ has a simple eigenspace for 1.

1.2 The model

The internal state of a classical computer may be viewed as an l-tuple

$$(a_{l-1}, a_{l-2}, ..., a_0)$$

where each $a_i \in \{0, 1\}$ and l is the number of bits employed by the computer and so the total number of possible states is 2^l. A classical computer may be viewed as a collection of functions that operate on these internal states. We will abbreviate the expression of these states by using the the Bra-ket notation. Specifically for any integer a such that $0 \leq a \leq 2^l - 1$ we take

$$|a\rangle := (a_{l-1}, a_{l-2}, ..., a_0)$$

where

$$a = \sum_{k=0}^{l-1} a_k 2^k \,.$$

Throughout this paper we take a_k to denote the k-th binary digit of any $a \in \mathbb{N}$. We will either express explicit states in binary or in the abbreviate form $|a\rangle$, for example on a 4 bit machine the internal state $(0, 1, 0, 1)$ may be expressed as $|0101\rangle$ as well as $|5\rangle$. In the case of ambiguity the context should make it clear.

Suppose we have a machine that admits only one bit. A single bit of a classical computer may reside only in the states $|0\rangle$ or $|1\rangle$ at any given time. However, a quantum bit, or *qubit*, may reside in a linear combination (or superposition) of these states. That is to say if $|f\rangle$ is a qubit of our machine then

$$|f\rangle = \alpha|0\rangle + \beta|1\rangle$$

where $\alpha, \beta \in \mathbb{C}$. When a physical measurement is made to observe the value of $|f\rangle$ it will collapse to either $|0\rangle$ or $|1\rangle$. We may refer to $|0\rangle$ and $|1\rangle$ as the observable states. The probability of observing $|0\rangle$ is $|\alpha|^2$ and the probability of observing $|1\rangle$ is $|\beta|^2$. Hence we must have

$$|\alpha|^2 + |\beta|^2 = 1 \,.$$

Classically there is only a single operation that can act on a single bit, that is the operation of negation or NOT gate. If we denote this gate by X we have

$$\begin{array}{ccc} & X & \\ |0\rangle & \mapsto & |1\rangle \,, \\ |1\rangle & \mapsto & |0\rangle \,. \end{array}$$

What would happen if we applied X to a qubit $|f\rangle$ that is in a superposition of states $|0\rangle$ and $|1\rangle$? It turns out we may treat X and in fact any quantum gate as a linear transformation. This follows from the laws of quantum mechanics. Thus the action of X on $|0\rangle$ and $|1\rangle$ determines the action of X on $|f\rangle$. That is

$$X|f\rangle = X(\alpha|0\rangle + \beta|1\rangle) = \alpha X|0\rangle + \beta X|1\rangle = \alpha|1\rangle + \beta|0\rangle \,.$$

Thus the quantum NOT gate simply interchanges the respective probabilities of $|0\rangle$ and $|1\rangle$.

Suppose we wish to consider a 2-qubit gate. Let X_c denote the controlled-NOT gate or CNOT gate. This gate will operate on two qubits and negate the second qubit if and only if the first qubit is $|1\rangle$. Thus for the observable states we have

$$\begin{array}{ccc} & X_c & \\ |00\rangle & \mapsto & |00\rangle \\ |01\rangle & \mapsto & |01\rangle \\ |10\rangle & \mapsto & |11\rangle \\ |11\rangle & \mapsto & |10\rangle\,. \end{array}$$

Once again the action on the observable states completely determines the action on any superposition of states:

$$\begin{array}{rl} & X_c(\alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle) \\ = & \alpha_0 X_c|00\rangle + \alpha_1 X_c|01\rangle + \alpha_2 X_c|10\rangle + \alpha_3 X_c|11\rangle \\ = & \alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|11\rangle + \alpha_3|10\rangle\,. \end{array}$$

If a machine is capable of employing l bits then there are $N := 2^l$ possible internal states denoted by

$$|0\rangle, |1\rangle, ..., |N-1\rangle\,. \tag{2}$$

At any moment the state of our machine is a linear combination of these states:

$$\gamma_0|0\rangle + \gamma_1|1\rangle + \cdots + \gamma_{N-1}|N-1\rangle$$

where $\gamma_i \in \mathbb{C}$ and

$$\sum_{k=0}^{N-1} |\gamma_k|^2 = 1\,. \tag{3}$$

Due to (3) any quantum operator must be unitary. Since a unitary operation is invertible this implies that any quantum operation is reversible.

In the case of Shor's algorithm it will be helpful to divide the state of our machine into two distinct registers. Suppose the two registers can assume linear combinations of the states

$$A = \{|0\rangle, |1\rangle, ..., |N-1\rangle\}$$

and

$$B = \{|0\rangle, |1\rangle, ..., |M-1\rangle\}$$

respectively. Then the complete state of the machine may be described as

$$\sum_{a=0}^{N-1} \sum_{b=0}^{M-1} \gamma_{a,b}|a\rangle|b\rangle$$

where

$$\sum_{a=0}^{N-1} \sum_{b=0}^{M-1} |\gamma_{a,b}|^2 = 1$$

and juxtaposition of vectors $|a\rangle|b\rangle = |a,b\rangle$ denotes the tensor product.

If $U : A \mapsto A$ is unitary then we may define $U^* : A \times B \mapsto A \times B$ by

$$U^*|a\rangle|b\rangle = (U|a\rangle)|b\rangle\,.$$

Then

$$\begin{aligned} U^* \sum_{a=0}^{N-1}\sum_{b=0}^{M-1} \gamma_{a,b}|a\rangle|b\rangle &= \sum_{b=0}^{M-1}\left(U\sum_{a=0}^{N-1}\gamma_{a,b}|a\rangle\right)|b\rangle \\ &= \sum_{b=0}^{M-1}\left(\sum_{a=0}^{N-1}\gamma'_{a,b}|a\rangle\right)|b\rangle \\ &= \sum_{a=0}^{N-1}\sum_{b=0}^{M-1}\gamma'_{a,b}|a\rangle|b\rangle \end{aligned}$$

where

$$\sum_{a=0}^{N-1}\sum_{b=0}^{M-1}|\gamma'_{a,b}|^2 = \sum_{a=0}^{N-1}\sum_{b=0}^{M-1}|\gamma_{a,b}|^2\,.$$

Hence U^* is also unitary and so may be implemented on a quantum machine. This means that we may take any quantum gate designed for one quantum space and extend it to be applied to any larger space. This will allow us to consider quantum machines with multiple registers and apply quantum gates defined to act principally on a single register. In an abuse of notation we will denote U^* by U where the context is clear.

2 Some useful quantum logic gates

2.1 The Hadamard gate

The Hadamard gate H_k is one of the more useful quantum gates. On a single qubit it is defined as follows:

$$\begin{aligned} & \overset{H}{} \\ |0\rangle &\mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |1\rangle &\mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\,. \end{aligned}$$

Note this may also be represented as

$$H|a\rangle := \frac{1}{\sqrt{2}}\sum_{b=0}^{1}\exp(\pi i a b)|b\rangle$$

where $|a\rangle$ and $|b\rangle$ are the single qubit representations of a and b.

Suppose we have l qubits. To define the Hadamard gate on a multi-qubit state it is convenient to use a special characteristic function $\Phi_k : \mathbb{N}\times\mathbb{N} \mapsto \{0,1\}$. Suppose $a = \sum_{i=0}^{l-1} a_i 2^i$ and $b = \sum_{i=0}^{l-1} b_i 2^i$. We take

$$\Phi_k(a,b) := \begin{cases} 1 & \text{if } a_i = b_i \text{ for all } i \neq k \\ 0 & \text{otherwise.} \end{cases}$$

On an l qubit basis state $|a\rangle$ we define H_k to be

$$H_k|a\rangle = \frac{1}{\sqrt{2}} \sum_{b=0}^{2^l-1} \Phi_k(a,b) \exp(\pi i a_k b_k)|b\rangle \,. \tag{4}$$

We must show that H_k is unitary.

$$\begin{aligned} H_k \sum_{a=0}^{2^l-1} \gamma(a)|a\rangle &= \sum_{a=0}^{2^l-1} \gamma(a) H_k|a\rangle \\ &= \sum_{a=0}^{2^l-1} \sum_{b=0}^{2^l-1} \frac{1}{\sqrt{2}} \gamma(a) \Phi_k(a,b) \exp(\pi i a_k b_k)|b\rangle \\ &= \sum_{b=0}^{2^l-1} \delta(b)|b\rangle \,, \end{aligned}$$

where

$$\delta(b) = \sum_{a=0}^{2^l-1} \frac{1}{\sqrt{2}} \gamma(a) \Phi_k(a,b) \exp(\pi i a_k b_k) \,.$$

Taking $\bar{b}$ to be such that $\Phi_k(b,\bar{b}) = 1$ and $\bar{b}_k = 1 - b_k$ we have

$$\begin{aligned} \delta(b) &= \frac{1}{\sqrt{2}} \gamma(b) \exp(\pi i b_k b_k) + \frac{1}{\sqrt{2}} \gamma(\bar{b}) \exp(\pi i \bar{b}_k b_k) \\ &= \frac{1}{\sqrt{2}} [\exp(\pi i b_k^2) \gamma(b) + \gamma(\bar{b})] \,. \end{aligned}$$

Now letting z^* denote the complex conjugate for any $z \in \mathbb{C}$ we have

$$\begin{aligned} |\delta(b)|^2 &= \delta(b)\delta^*(b) \\ &= \frac{1}{2} [\exp(\pi i b_k^2) \gamma(b) + \gamma(\bar{b})][\exp(\pi i b_k^2) \gamma^*(b) + \gamma^*(\bar{b})] \\ &= \frac{1}{2} [\gamma(b)\gamma^*(b) + \gamma(\bar{b})\gamma^*(\bar{b}) + \exp(\pi i b_k^2) \gamma(b)\gamma^*(\bar{b}) + \exp(\pi i b_k^2)(\gamma(b)\gamma^*(\bar{b}))^*] \\ &= \frac{1}{2} |\gamma(b)|^2 + \frac{1}{2} |\gamma(\bar{b})|^2 + \exp(\pi i b_k^2) \mathrm{Re}(\gamma(b)\gamma^*(\bar{b})) \,. \end{aligned}$$

Likewise

$$|\delta(\bar{b})|^2 = \frac{1}{2} |\gamma(b)|^2 + \frac{1}{2} |\gamma(\bar{b})|^2 + \exp(\pi i \bar{b}_k^2) \mathrm{Re}(\gamma(b)\gamma^*(\bar{b})) \,,$$

and hence

$$\begin{aligned} |\delta(b)|^2 + |\delta(\bar{b})|^2 &= |\gamma(b)|^2 + |\gamma(\bar{b})|^2 + [\exp(\pi i b_k^2) + \exp(\pi i \bar{b}_k^2)] \mathrm{Re}(\gamma(b)\gamma^*(\bar{b})) \\ &= |\gamma(b)|^2 + |\gamma(\bar{b})|^2 \,. \end{aligned}$$

Now we have

$$\sum_{b=0}^{2^l-1} |\delta(b)|^2 = \sum_{b=0}^{2^l-1} |\gamma(b)|^2 \,,$$

and so H_k is unitary.

Note that two Hadamard gates with different indecies may commute. To see this consider

$$\begin{aligned} H_j H_k|a\rangle &= H_j \frac{1}{\sqrt{2}} \sum_{b=0}^{2^l-1} \Phi_k(a,b) \exp(\pi i a_k b_k)|b\rangle \\ &= \frac{1}{\sqrt{2}} \sum_{b=0}^{2^l-1} \Phi_k(a,b) \exp(\pi i a_k b_k) \frac{1}{\sqrt{2}} \sum_{c=0}^{2^l-1} \Phi_j(b,c) \exp(\pi i b_j c_j)|c\rangle \\ &= \frac{1}{2} \sum_{b,c=0}^{2^l-1} \Phi_k(a,b)\Phi_j(b,c) \exp(\pi i [a_k b_k + b_j c_j])|c\rangle \,. \end{aligned} \tag{5}$$

The values of b and c that affect the sum must satisfy $\Phi_k(a,b)\Phi_j(b,c) = 1$, hence if we assume that $j \neq k$ then $a_j = b_j$ and $b_k = c_k$. Thus we have

$$H_j H_k|a\rangle = \frac{1}{2} \sum_{c=0}^{2^l-1} \exp(\pi i [a_k c_k + a_j c_j])|c\rangle = H_k H_j|a\rangle \,.$$

We will now show that H_k is its own inverse. Let $\overline{a}$ be such that $\Phi_k(a,\overline{a}) = 1$ and $\overline{a}_k = 1 - a_k$. Returning to (5) and taking $j = k$ we see that

$$\begin{aligned} H_k H_k|a\rangle &= \frac{1}{2} \sum_{b,c=0}^{2^l-1} \Phi_k(a,b)\Phi_k(b,c) \exp(\pi i [a_k b_k + b_k c_k])|c\rangle \\ &= \frac{1}{2} \sum_{b,c=0}^{2^l-1} \Phi_k(a,b)\Phi_k(b,c) \exp(\pi i b_k [a_k + c_k])|c\rangle \,. \end{aligned}$$

If $\Phi_k(a,b)\Phi_k(b,c) = 1$ then $a_t = b_t = c_t$ for $t \neq k$ and b_k and c_k are free. Enumerating all possibilities, i.e

$$\begin{aligned} b_k &= a_k & c_k &= a_k \\ b_k &= 1 - a_k & c_k &= a_k \\ b_k &= a_k & c_k &= 1 - a_k \\ b_k &= 1 - a_k & c_k &= 1 - a_k \,, \end{aligned}$$

we have

$$\begin{aligned} H_k H_k|a\rangle &= \frac{1}{2} \exp(\pi i a_k [a_k + a_k])|a\rangle + \frac{1}{2} \exp(\pi i (1 - a_k)[a_k + a_k])|a\rangle \\ &\quad + \frac{1}{2} \exp(\pi i a_k [a_k + (1 - a_k)])|\overline{a}\rangle + \frac{1}{2} \exp(\pi i (1 - a_k)[a_k + (1 - a_k)])|\overline{a}\rangle \\ &= \frac{1}{2} \exp(2\pi i a_k^2)|a\rangle + \frac{1}{2} \exp(2\pi i (1 - a_k) a_k)|a\rangle + \frac{1}{2} \exp(\pi i a_k)|\overline{a}\rangle + \frac{1}{2} \exp(\pi i (1 - a_k))|\overline{a}\rangle \\ &= |a\rangle \,. \end{aligned}$$

We may apply the Hadamard gate to each qubit of $|0\rangle$ to obtain a uniform superposition of all observable states. Define $H^{(l-1)}$ to do just this, i.e let

$$H^{(l-1)} := H_0 H_1 \cdots H_{l-1} \,. \tag{6}$$

Note that $(H^{(l-1)})^{-1} = H^{(l-1)}$. Defining $b^{(l)} = 0$ we have

$$
\begin{aligned}
H^{(l-1)}|0\rangle &= H_0 \cdots H_{l-3} H_{l-2} H_{l-1} |0\rangle \\
&= H_0 \cdots H_{l-3} H_{l-2} \frac{1}{\sqrt{2}} \sum_{b^{(l-1)}=0}^{2^l-1} \Phi_{l-1}(b^{(l)}, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l)} b_{l-1}^{(l-1)}\right) |b^{(l-1)}\rangle \\
&= H_0 \cdots H_{l-3} \frac{1}{\sqrt{2}} \sum_{b^{(l-1)}=0}^{2^l-1} \Phi_{l-1}(b^{(l)}, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l)} b_{l-1}^{(l-1)}\right) H_{l-2} |b^{(l-1)}\rangle \\
&= H_0 \cdots H_{l-3} \frac{1}{2^{1/2}} \sum_{b^{(l-1)}=0}^{2^l-1} \Phi_{l-1}(b^{(l)}, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l)} b_{l-1}^{(l-1)}\right) \\
&\qquad \frac{1}{\sqrt{2}} \sum_{b^{(l-2)}=0}^{2^l-1} \Phi_{l-2}\left(b^{(l-1)}, b^{(l-2)}\right) \exp\left(\pi i b_{l-2}^{(l-1)} b_{l-2}^{(l-2)}\right) |b^{(l-2)}\rangle \\
&= H_0 \cdots H_{l-3} \left(\frac{1}{\sqrt{2}}\right)^2 \sum_{b^{(l-1)}, b^{(l-2)}=0}^{2^l-1} \Phi_{l-1}(b^{(l)}, b^{(l-1)}) \Phi_{l-2}\left(b^{(l-1)}, b^{(l-2)}\right) \\
&\qquad \exp\left(\pi i b_{l-1}^{(l)} b_{l-1}^{(l-1)}\right) \exp\left(\pi i b_{l-2}^{(l-1)} b_{l-2}^{(l-2)}\right) |b^{(l-2)}\rangle
\end{aligned}
$$

After applying every Hadamard gate we get

$$
\frac{1}{\sqrt{2^l}} \sum_{b^{(l-1)}, \ldots, b^{(0)}=0}^{2^l-1} \left[\prod_{k=0}^{l-1} \Phi_k\left(b^{(k+1)}, b^{(k)}\right)\right] \exp\left(\sum_{k=0}^{l-1} \pi i b_k^{(k+1)} b_k^{(k)}\right) |b^{(0)}\rangle \tag{7}
$$

We will now consider a lemma that applies to the first set of factors in the above sum.

Lemma 2.1. *Suppose we have a set of integers $b^{(0)}, b^{(1)}, ..., b^{(l)}$ where $a := b^{(l)}$, $c := b^{(0)}$ and that*

$$
\begin{aligned}
b^{(l)} &= \sum_{k=0}^{l-1} a_k 2^k \\
b^{(0)} &= \sum_{k=0}^{l-1} c_k 2^k
\end{aligned}
$$

and

$$
\prod_{k=0}^{l-1} \Phi_k\left(b^{(k+1)}, b^{(k)}\right) = 1\,.
$$

Then we have

$$
b_t^{(k)} = \begin{cases} a_t & t < k \\ \\ c_t & t \geq k \end{cases} \tag{8}
$$

for all $0 \leq k \leq l$ and $0 \leq t \leq l-1$.

Proof. The proof can be simply carried out by two inductions. Below all values of t are assumed to lie between 0 and $l-1$. First note that by definition

$$b_t^{(l)} = a_t \qquad t < l\,.$$

Now suppose for some $0 \leq k \leq l-1$ that

$$b_t^{(k+1)} = a_t \qquad t < k+1\,.$$

Since $\Phi_k(b^{(k+1)}, b^{(k)}) = 1$ it follows that

$$b_t^{(k)} = b_t^{(k+1)} \qquad t \neq k\,.$$

Hence combining the two above equations we have

$$b_t^{(k)} = a_t \qquad t < k\,, \tag{9}$$

and so by induction this statement holds for all $0 \leq k \leq l$.

Another induction will demonstrate the second part of (8). Note that by definition

$$b_t^{(0)} = c_t \qquad t \geq 0\,.$$

Now suppose for some $0 < k \leq l-1$ that

$$b_t^{(k-1)} = c_t \qquad t \geq k-1\,.$$

Since $\Phi_{k-1}(b^{(k)}, b^{(k-1)}) = 1$ it follows that

$$b_t^{(k)} = b_t^{(k-1)} \qquad t \neq k-1\,.$$

Hence combining the two above equations we have

$$b_t^{(k)} = c_t \qquad t \geq k\,, \tag{10}$$

and so by induction this statement holds for all $0 \leq k \leq l$.

Now (8) follows from both (9) and (10). □

Returning to (7) and applying the above lemma we have

$$\frac{1}{\sqrt{2^l}} \sum_{b^{(l-1)},\ldots,b^{(0)}=0}^{2^l-1} \left[\prod_{k=0}^{l-1} \Phi_k\left(b^{(k+1)}, b^{(k)}\right)\right] \exp\left(\sum_{k=0}^{l-1} \pi i b_k^{(k+1)} b_k^{(k)}\right) |b^{(0)}\rangle$$
$$= \frac{1}{\sqrt{2^l}} \sum_{b^{(0)}=0}^{2^l-1} \exp\left(\sum_{k=0}^{l-1} \pi i a_k c_k\right) |b^{(0)}\rangle\,.$$

where a_i and c_i are defined in the above lemma. Recalling in this instance that $b^{(l)} = 0$, i.e $a_i = 0$ for all i we have

$$\frac{1}{\sqrt{2^l}} \sum_{b^{(0)}=0}^{2^l-1} |b^{(0)}\rangle\,.$$

This represents a uniform superposition of all observable states in an l-bit system.

2.2 The controlled phase shift gate

Another useful quantum gate is the controlled phase shift gate $S_{j,k}$, $j < k$. This gate specifies two qubits as control bits and adds a phase angle of $\pi/2^{k-j}$ if and only if both control bits are 1. For example on a two qubit system (in which both qubits must act as the control bits)

$$\begin{array}{ccc} & S_{0,1} & \\ |00\rangle & \mapsto & |00\rangle \\ |01\rangle & \mapsto & |01\rangle \\ |10\rangle & \mapsto & |10\rangle \\ |11\rangle & \mapsto & \exp(\pi i/2^1)|11\rangle\,. \end{array}$$

Note that this gate does not alter the probability of obtaining any state, but merely modifies the phase angle of the component of the last state.

For a general l qubit state $|a\rangle$ we may define $S_{j,k}$ as

$$S_{j,k}|a\rangle := \exp(\pi i a_j a_k/2^{k-j})|a\rangle\,, \tag{11}$$

where

$$a = \sum_{k=0}^{l-1} a_k 2^k\,.$$

2.3 The Toffoli gate

In Shor's algorithm it will be necessary to construct a quantum gate array that mimics a classical array. This can be done by noting that any classical gate can be decomposed into NAND gates, and the quantum Toffoli gate can be used to mimic a NAND gate. The classical NAND gate is simply the negation of the AND gate.

The Toffoli gate is a double controlled NOT gate, i.e the value of the third qubit is negated if and only if the first two are both 1:

$$\begin{array}{ccc} & \text{Toffoli} & \\ |000\rangle & \mapsto & |000\rangle \\ |001\rangle & \mapsto & |001\rangle \\ |010\rangle & \mapsto & |010\rangle \\ |011\rangle & \mapsto & |011\rangle \\ |100\rangle & \mapsto & |100\rangle \\ |101\rangle & \mapsto & |101\rangle \\ |110\rangle & \mapsto & |111\rangle \\ |111\rangle & \mapsto & |110\rangle\,. \end{array}$$

Note that when the third qubit of the input is 1 then the third qubit of the output is the NAND of the other two.

3 Grover's Algorithm

Suppose we are given a polynomial function p and a set of N numbers containing a single root of p. If we wish to identify this root on a classical computer the only general algorithm available to us is to test each number one by one until we find the root, a process which will be of $O(N)$. This can be generalized to the problem of searching among a list of elements for a particular element identified by a special function. This function is often called the oracle and is such that analysing the structure of the function lends no information about which particular element the function will select. On a quantum computer this problem can be solved in $O(\sqrt{N})$ using Grover's algorithm.

Let $\mathcal{B} := \{0, 1, ..., N-1\}$ for some integer N. Suppose we have an oracle operator Q for some $\omega \in \mathcal{B}$ such that $Q|\omega\rangle = -|\omega\rangle$, but $Q|\beta\rangle = |\beta\rangle$ for all integers $\beta \neq \omega$. Our goal is to discover the unique element $|\omega\rangle$ identified by Q. Note that Q may be represented as a Householder reflection

$$Q = I - 2|\omega\rangle\langle\omega|.$$

However it should be noted that the above representation can only facilitate our analysis and that Q should be interpreted as a black box, as in practice the algorithm operates from the standpoint that no information about ω can be efficiently extracted from examining the mere structure of the oracle.

On a classical computer we could test each element of $\mathcal{B}$ until we discover ω, an algorithm of $O(N)$. On a quantum computer we can essentially test each element of $\mathcal{B}$ simultaneously, maximizing the likelyhood of discovering ω after $O(\sqrt{N})$ steps.

Let $|\alpha\rangle$ be the uniform superposition of all quantum states, i.e

$$|\alpha\rangle = \frac{1}{\sqrt{N}} \sum_{\beta=0}^{N-1} |\beta\rangle\,, \tag{12}$$

Now let

$$P := 2|\alpha\rangle\langle\alpha| - I$$

and let

$$G := PQ.$$

Note that we may construct P in the following way:

$$\begin{aligned} P &= 2|\alpha\rangle\langle\alpha| - I \\ &= 2H^{(l-1)}|0\rangle\langle 0|H^{(l-1)} - H^{(l-1)}IH^{(l-1)} \\ &= H^{(l-1)}(2|0\rangle\langle 0| - I)H^{(l-1)}\,, \end{aligned}$$

where $H^{(l-1)}$ is defined in (6). Hence in order to implement this algorithm our machine must employ an oracle Q, the Hadamard gate, and inversion about $|0\rangle$.

Below we will demonstrate that the composition of P and Q applied successively to $|\alpha\rangle$ may be interpreted as a sequence of rotations by a fixed angle towards $|\omega\rangle$ in the plane defined by $|\alpha\rangle$ and $|\omega\rangle$.

Thus the algorithm will proceed by simply forming the iterates $G^k|\alpha\rangle$ and stopping once we are certain the current iterate is near $|\omega\rangle$. At this point when we observe the internal state it will collapse to the classical state $|\omega\rangle$ with high probability.

3.1 Why it works

3.1.1 An outline

The analysis of the algorithm can be broken into the following propositions:

- The first application of G to $|\alpha\rangle$ progresses towards $|\omega\rangle$.
- For $k \in \mathbb{N}$ the iterates $G^k|\alpha\rangle$ remain in the plane defined by $|\alpha\rangle$ and $|\omega\rangle$.
- The angle between any two successive iterates is constant, i.e for all $k \in \mathbb{N}$ and for some $\theta \in \mathbb{R}$ we have $\langle G^{k+1}|\alpha\rangle, G^k|\alpha\rangle\rangle = \cos\theta$.
- All iterates progress in the same direction. This will be demonstrated by noting that the iterates cannot backtrack, i.e $G^{k+2}|\alpha\rangle \neq G^k|\alpha\rangle$ for all $k \in \mathbb{Z}$.

These propositions taken together will show that the iterates $G^k|\alpha\rangle$ will rotate towards $|\omega\rangle$ by a specific angle until it is surpassed. In the last section we will show that terminating the algorithm when k is near $\pi\sqrt{N}/4$ will ensure a high likelyhood of success. We will now consider each proposition in detail.

3.1.2 The details

Note that $\langle\alpha|\alpha\rangle = 1$ and $\langle\alpha|\omega\rangle = 1/\sqrt{N}$. The first iteration of the algorithm results in $G|\alpha\rangle$, where

$$\begin{aligned} G|\alpha\rangle &= PQ|\alpha\rangle \\ &= (2|\alpha\rangle\langle\alpha| - I)(I - 2|\omega\rangle\langle\omega|)|\alpha\rangle \\ &= (2|\alpha\rangle\langle\alpha| - I)\left(|\alpha\rangle - \frac{2}{\sqrt{N}}|\omega\rangle\right) \\ &= |\alpha\rangle - \frac{4}{N}|\alpha\rangle + \frac{2}{\sqrt{N}}|\omega\rangle \\ &= \frac{N-4}{N}|\alpha\rangle + \frac{2}{\sqrt{N}}|\omega\rangle\,. \end{aligned} \tag{13}$$

From this we see for $N > 2$ that

$$\langle\omega|G|\alpha\rangle = \frac{N-4}{N}\frac{1}{\sqrt{N}} + \frac{2}{\sqrt{N}} > \frac{1}{\sqrt{N}}\,. \tag{14}$$

Recall that $\langle\omega|\alpha\rangle = 1/\sqrt{N}$. Thus it follows that if ϕ_0 is the angle between $|\alpha\rangle$ and $|\omega\rangle$ and ϕ_1 is the angle between $G|\alpha\rangle$ and $|\omega\rangle$ then

$$\langle\omega|G|\alpha\rangle = \cos\phi_1 > \cos\phi_0 = \langle\omega|\alpha\rangle$$

and so $G|\alpha\rangle$ has rotated towards $|\omega\rangle$.

Note also that we have

$$\begin{aligned} G|\omega\rangle &= (2|\alpha\rangle\langle\alpha| - I)(I - 2|\omega\rangle\langle\omega|)|\omega\rangle \\ &= (2|\alpha\rangle\langle\alpha| - I)(-|\omega\rangle) \\ &= -\frac{2}{\sqrt{N}}|\alpha\rangle + |\omega\rangle\,. \end{aligned} \tag{15}$$

Thus from (13) and (15) we see that $G|\alpha\rangle, G|\omega\rangle \in \text{span}(|\alpha\rangle, |\omega\rangle)$ and so if $|\beta\rangle \in \text{span}(|\alpha\rangle, |\omega\rangle)$ then $G|\beta\rangle \in \text{span}(|\alpha\rangle, |\omega\rangle)$, thus in particular it follows by induction that

$$G^k|\alpha\rangle \in \text{span}(|\alpha\rangle, |\omega\rangle) \tag{16}$$

for all $k \in \mathbb{N}$.

Now the algorithm will proceed by repeatedly applying G to $|\alpha\rangle$. Note that since G^k is unitary it follows from (1) that the application of G^k preserves angles, i.e

$$\langle G^{k+1}|\alpha\rangle, G^k|\alpha\rangle\rangle = \langle G|\alpha\rangle, |\alpha\rangle\rangle = \cos\theta\,,$$

where θ is the angle between $|\alpha\rangle$ and $G|\alpha\rangle$. Thus it follows that each iterate $G^k|\alpha\rangle$ is generated by rotating $G^{k-1}|\alpha\rangle$ in $\text{span}(|\alpha\rangle, |\omega\rangle)$ by θ.

In order to show that each rotation is in the same direction we must show that $G^{k+2}|\alpha\rangle \neq G^k|\alpha\rangle$ for all $k \in \mathbb{N}$. Now suppose

$$G^{k+2}|\alpha\rangle = G^k|\alpha\rangle$$

for some $k \in \mathbb{N}$. Since G^k is invertible and $G^{-1} = Q^{-1}P^{-1} = QP$ we have

$$\begin{aligned} & G^2|\alpha\rangle = |\alpha\rangle \\ \Rightarrow\ & G|\alpha\rangle = G^{-1}|\alpha\rangle \\ \Rightarrow\ & PQ|\alpha\rangle = QP|\alpha\rangle = Q|\alpha\rangle\,. \end{aligned} \tag{17}$$

Now since $|\alpha\rangle \neq |0\rangle$ and Q is a householder reflection it follows that $Q|\alpha\rangle \neq |0\rangle$. This implies from (17) that $Q|\alpha\rangle$ is an eigenvector of P of eigenvalue 1, which is a simple eigenvalue of P since P is the opposite of a householder reflection. This implies that $Q|\alpha\rangle = \lambda|\alpha\rangle$ for some $\lambda \in \mathbb{R}$ and $\lambda \neq 0$. Hence $|\alpha\rangle$ is an eigenvector of Q and so we have

$$\begin{aligned} & Q|\alpha\rangle = \lambda|\alpha\rangle \\ \Rightarrow\ & (I - 2|\omega\rangle\langle\omega|)|\alpha\rangle = \lambda|\alpha\rangle \\ \Rightarrow\ & |\alpha\rangle - 2\langle\omega|\alpha\rangle|\omega\rangle = \lambda|\alpha\rangle \\ \Rightarrow\ & -2\langle\omega|\alpha\rangle|\omega\rangle = (\lambda - 1)|\alpha\rangle\,. \end{aligned}$$

Since $\langle\omega|\alpha\rangle \neq 0$ this implies that $|\omega\rangle$ and $|\alpha\rangle$ are parallel, contradicting (12). Thus all applications of G rotate in the same direction. Since we have already established in (14) that the first application of G rotates $|\alpha\rangle$ towards $|\omega\rangle$ it follows that the iterates $G^k|\alpha\rangle$ will rotate towards $|\omega\rangle$ until it is surpassed. □

3.2 Computational cost

It remains to be shown how many iterations should be performed in order to maximize the probability of discovering $|\omega\rangle$. As we will see, the number of steps needed to maximize the probability of obtaining the desired result is of $O(\sqrt{N})$. Let

$$\begin{aligned} |\overline{\omega}\rangle &:= \frac{1}{\sqrt{N-1}} \sum_{\beta=0}^{N-1} |\beta\rangle - \frac{1}{\sqrt{N-1}} |\omega\rangle \\ &= \sqrt{\frac{N}{N-1}} |\alpha\rangle - \sqrt{\frac{1}{N-1}} |\omega\rangle \,, \end{aligned}$$

and from this we see that

$$|\alpha\rangle = \sqrt{1 - \frac{1}{N}} |\overline{\omega}\rangle + \sqrt{\frac{1}{N}} |\omega\rangle \,. \tag{18}$$

Note that $|\overline{\omega}\rangle$ is orthogonal to $|\omega\rangle$.

We wish to determine the angle between $|\overline{\omega}\rangle$ and $|\alpha\rangle$, which we denote φ. Now from (18) we have

$$\langle \alpha | \overline{\omega} \rangle = \sqrt{1 - \frac{1}{N}} = \cos \varphi$$

and

$$\sin \varphi = \sqrt{\frac{1}{N}} \,.$$

From (13) we have

$$\langle \alpha | G | \alpha \rangle = \langle \alpha | \left(\frac{N-4}{N} |\alpha\rangle + \frac{2}{\sqrt{N}} |\omega\rangle \right) = \frac{N-4}{N} + \frac{2}{N} = 1 - \frac{2}{N} \,.$$

This implies that

$$2 \cos^2 \varphi - 1 = \cos \theta \,,$$

and since $\cos \varphi > 0$ and $\cos \theta > 0$ we have $\varphi = \theta/2$. Moreover

$$\theta = 2 \arcsin \left(\frac{1}{\sqrt{N}} \right) \approx \frac{2}{\sqrt{N}} \,.$$

Note that from (18) we have

$$|\alpha\rangle = \cos(\theta/2) |\overline{\omega}\rangle + \sin(\theta/2) |\omega\rangle \,.$$

Now from (16) and recalling that the iterates $G^k |\alpha\rangle$ are rotations in $\mathrm{span}(|\alpha\rangle, |\omega\rangle) = \mathrm{span}(|\overline{\omega}\rangle, |\omega\rangle)$ by θ we see that each step of the algorithm can be represented as

$$G^k |\alpha\rangle = \cos \left(\frac{2k+1}{2} \theta \right) |\overline{\omega}\rangle + \sin \left(\frac{2k+1}{2} \theta \right) |\omega\rangle \,.$$

In order to maximize the likelyhood of observing $|\omega\rangle$ when the algorithm is finished we need

$$\sin^2 \left(\frac{2k+1}{2} \theta \right) \approx 1$$

or

$$\frac{2k+1}{2}\theta \approx \frac{\pi}{2}\,.$$

Thus we have

$$k \approx \frac{\pi-\theta}{2\theta} \approx \frac{\pi - 2/\sqrt{N}}{4/\sqrt{N}} \approx \frac{\pi}{4}\sqrt{N}\,,$$

and if we take $k := \text{round}(\pi\sqrt{N}/4)$ we see that the algorithm runs in $O(\sqrt{N})$.

4 Shor's algorithm

4.1 The Algorithm

Suppose we wish to identify a non-trivial factor of the odd number n. The algorithm proceeds as follows:

1. Choose a random integer x such that $2 < x < n$.
2. Compute $\gcd(x, n)$. If $\gcd(x, n) \neq 1$ we are done, for then we have a nontrivial factor of n.
3. Otherwise compute the order r of $x \bmod n$ as described in section 4.5. Recall that r is the smallest non-negative integer such that $x^r \equiv 1 \pmod n$.
4. If r is odd then return to step 1.
5. Now compute $\gcd(x^{r/2} + 1, n)$. If this value is not 1 or n we are done, otherwise return to step 1.

Note that step 4 is necessary to insure that $x^{r/2}$ may be formed for step 5. We will show in section 4.7 that repeating this algorithm $O(\log \log n)$ times will ensure a high probability of success. The total cost will then be $O[(\log n)^3]$.

The algorithm may be divided conceptually into two parts. The first is that under the proper assumptions we may compute a nontrivail factor of n if r is know, namely by computing $\gcd(x^{r/2} + 1, n)$. This will be explained in section 4.2.

This leaves us with the matter of determining r. It turns out that r may be recovered with high probability by employing the fourier transform and modular exponentiation. This will be described in section 4.5.

4.2 Determining a non-trivial factor of n from the order of $x \bmod n$

Given a number n that we wish to factor and a random number x suppose that r is the order of x in mod n and that r is even, i.e r is the smallest non-negative integer such that

$$x^r - 1 \equiv (x^{r/2} + 1)(x^{r/2} - 1) \equiv 0 \pmod n . \tag{19}$$

Also assume that $x^{r/2} + 1 \not\equiv 0 \pmod n$, i.e that $\gcd(x^{r/2} + 1, n) \neq n$.

If $\gcd(x^{r/2} + 1, n) = 1$ then

$$a(x^{r/2} + 1) + bn = 1$$

for some integers a and b. Hence

$$x^{r/2} - 1 = a(x^{r/2} + 1)(x^{r/2} - 1) + bn(x^{r/2} - 1) \equiv 0 \pmod n,$$

contradicting the definition of r. Thus $\gcd(x^{r/2}+1, n) \neq 1$ and so is a nontrivial factor on n.

For this process to succeed, i.e for $\gcd(x^{r/2} + 1, n)$ to be a nontrivial factor of n, we require that $x \in \mathcal{C}_1$ where

$$\mathcal{C}_1 = \{x \in \mathbb{N} \ : \ r \text{ is even and } x^{r/2} + 1 \not\equiv 0 \pmod n\} . \tag{20}$$

It turns out that this is highly likely as the following shows.

Let us write

$$n = \prod_{i=1}^{k} p_i^{\alpha_i}$$

where each p_i is prime. We may assume that $p_i \neq 2$ for all i and that $k \neq 1$, as either of these cases may be factored efficiently on a classical computer.

Let r_i be the order of $x \pmod{p_i^{\alpha_i}}$. Note that

$$r = \text{lcm}\{r_1, r_2, ..., r_k\}\,. \tag{21}$$

This may be established as follows. If s is a positive integer such that for each r_i we have $s = a_i r_i$ for some integer a_i, then it follows that

$$x^s = (x^{r_i})^{a_i} \equiv 1^{a_i} \equiv 1 \pmod{p_i^{\alpha_i}}\,.$$

This establishes that the common multiples of $r_1, ..., r_k$ are potential candidates for r. Now for each i there exists non-negative integers b_i and c_i such that $r = b_i r_i + c_i$ where $c_i < r_i$. Therefore for each i we have

$$x^{c_i} = x^{r - b_i r_i} = x^r (x^{r_i})^{-b_i} \equiv 1 \pmod{p_i^{\alpha_i}}\,.$$

This violates the definition of r_i and so each $c_i = 0$ and we have $r = b_i r_i$. This establishes that r is among the common multiples of $r_1, ..., r_k$, and hence must be the least of these.

Let even(A) be the largest power of 2 dividing A, i.e if o is an odd integer then even$(2^s o) = 2^s$ where $s \geq 0$. Note that (20) is true if and only if even$(r_i) \neq$ even(r_j) for some $i \neq j$. For if even$(r_i) \neq$ even(r_j) for some $i \neq j$ then either r_i or r_j is even and so by (21) r is even. Also from (21) there exists an even e such that either $r = er_i$ or $r = er_j$. WLOG assume $r = er_i$. Then

$$x^{r/2} = (x^{r_i})^{e/2} \equiv 1^{e/2} \equiv 1 \pmod{p_i^{\alpha_i}}\,.$$

Thus by the Chinese Remainder Theorem we have $x^{r/2} \not\equiv -1 \pmod{n}$ and so (20) holds.

If even$(r_i) =$ even(r_j) for all i, j then by (21) $r = o_i r_i$ where each o_i is odd. If even$(r_i) = 1$ for all i then by (21) r is odd and (20) does not hold. Otherwise even$(r_i) \neq 1$ for all i, i.e $r_i/2$ is an integer giving

$$x^{r/2} = (x^{r_i/2})^{o_i} \equiv (-1)^{o_i} \equiv -1 \pmod{p_i^{\alpha_i}}\,,$$

and so by the Chinese Remainder Theorem we have $x^{r/2} \equiv -1 \pmod{n}$ and still (20) does not hold.

We may now turn our attention to the probability that even$(r_i) \neq$ even(r_j) for some $i \neq j$. Note that $(\text{mod}\, p_i^{\alpha_i})$ is a cyclic multiplicative group, i.e it has a generating element g_i. Let $x = g_i^{k_i} \pmod{p_i^{\alpha_i}}$ and let r_{g_i} be the order of g_i in mod $p_i^{\alpha_i}$. Note that since x is random then k_i is random.

Let us write $k_i r_i = a_i r_{g_i} + b_i$, where $0 \leq b_i < r_{g_i}$. Then

$$1 \equiv x^{r_i} = (g_i^{k_i})^{r_i} = g_i^{k_i r_i} = (g_i^{r_{g_i}})^{a_i} g_i^{b_i} \equiv g_i^{b_i} \pmod{p_i^{\alpha_i}}\,.$$

Thus $b_i = 0$ and

$$r_i = \frac{a_i r_{g_i}}{k_i}\,.$$

This establishes that

$$r_i \in T := \{ar_{g_i}/k_i \in \mathbb{Z} \ : \ a \in \mathbb{Z}, a \geq 1\}\,.$$

Note that for any integer $t \in T$ there exist some integer $a \geq 1$ such that

$$x^t = x^{ar_{g_i}/k_i} = (g_i^{k_i})^{ar_{g_i}/k_i} = (g_i^{r_{g_i}})^a \equiv 1^a \equiv 1 \pmod{p_i^{\alpha_i}}\,.$$

Thus $r_i = \min(T)$. That is

$$r_i = \frac{k_i}{\gcd(r_{g_i}, k_i)}\frac{r_{g_i}}{k_i} = \frac{r_{g_i}}{\gcd(r_{g_i}, k_i)} = \frac{r_{g_i}\mathrm{lcm}(r_{g_i}, k_i)}{r_{g_i}k_i} = \frac{\mathrm{lcm}(r_{g_i}, k_i)}{k_i}\,.$$

Here we have used the well known property that

$$\gcd(A, B) = \frac{AB}{\mathrm{lcm}(A, B)}\,.$$

Let us determine the probability that $\mathrm{even}(r_i) = 2^s$ for some integer $s \geq 0$. Note that k_i is random and hence

$$\mathrm{prob}(\mathrm{even}(k_i) = t) \leq \frac{1}{2}$$

for any integer t. Now

$$\begin{aligned}
\mathrm{prob}(\mathrm{even}(r_i) = 2^s) &= \mathrm{prob}\left(\mathrm{even}\left(\frac{\mathrm{lcm}(r_{g_i}, k_i)}{k_i}\right) = 2^s\right) \\
&= \mathrm{prob}\left(\frac{\mathrm{even}(\mathrm{lcm}(r_{g_i}, k_i))}{\mathrm{even}(k_i)} = 2^s\right) \\
&= \mathrm{prob}\left(\mathrm{even}(k_i) = \frac{\mathrm{even}(\mathrm{lcm}(r_{g_i}, k_i))}{2^s}\right) \\
&\leq \frac{1}{2}
\end{aligned}$$

From this it follows that the probability that $\mathrm{even}(r_i) = \mathrm{even}(r_j)$ for all $1 \leq i, j \leq k$ is less than or equal to $1/2^{k-1}$. Hence the probability that $\mathrm{even}(r_i) \neq \mathrm{even}(r_j)$ for some i, j is at least $1 - 1/2^{k-1}$. Recall that $\mathrm{even}(r_i) \neq \mathrm{even}(r_j)$ for some i, j if and only if $x \in \mathcal{C}_1$ where again

$$\mathcal{C}_1 = \{x \in \mathbb{N} \ : \ r \text{ is even and } x^{r/2} + 1 \not\equiv 0 \pmod{n}\}\,.$$

Thus for a random $x \in \mathbb{N}$ we have

$$\mathrm{prob}(x \in \mathcal{C}_1) \geq 1 - \frac{1}{2^{k-1}}\,. \tag{22}$$

It is therefore highly likely that any choice for x will permit the computation of a factor of n when k is not too small.

4.3 Modular exponentiation

Suppose we have an l-bit machine. We wish to compute

$$x^a \pmod n \tag{23}$$

where a is some non-negative integer less than 2^l. To do this we write a in binary as

$$a = \sum_{k=0}^{l-1} 2^k a_k$$

where each $a_k \in \{0, 1\}$.

First we square x repeatedly to form $x^{2^k} \pmod n$ for $k = 0, 1, ..., l-1$. We may now compute

$$x^a \pmod n = \prod_{k=0}^{l-1} x^{a_k 2^k} \pmod n .$$

Thus computing (23) will require $O(l)$ modular multiplications.

The fastest classical algorithm for integer multiplication of large l bit numbers is the Schönhage-Strassen algorithm [2]. Classically this algorithm requires $O(l \log l \log\log l)$ NAND gates to multiply two l bit numbers. For a quantum gate array we must use the Toffoli gate in place of the NAND gate.

Since we must perform $O(l)$ multiplications to compute $x^a \pmod n$ it follows that this process may be carried out in

$$O(l^2 \log\log l \log\log\log l) \tag{24}$$

time.

4.4 The quantum fourier transform

The quantum fourier transform is at the heart of the algorithm. In particular it will be used in step 3 of section 4.1 to compute the order of x in mod n.

Let us suppose a is a non-negative integer and consider the basis state $|a\rangle$. Assuming we are operating on an l bit register there are $q = 2^l$ possible internal states $|a\rangle$. Let F denote the fourier transform. Then

$$F|a\rangle = \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp(2\pi i ac/q)|c\rangle . \tag{25}$$

This operation is implemented using the Hadamard gate and the controlled Phase gate. The implementation will in fact produce the fourier transform with the bits in reverse order, hence we denote it by F_r and define it as follows:

$$F_r|a\rangle := T_0 T_1 T_2 \cdots T_{l-1}|a\rangle , \tag{26}$$

where $T_k := S_{0,k} S_{1,k} \cdots S_{k-1,k} H_k$ for $k \geq 1$ and $T_0 = H_0$. To see why this produces the fourier transform we simply apply each gate as defined in (4) and

(11). Applying H_{l-1} to $|a\rangle$ yields

$$H_{l-1}|a\rangle = \frac{1}{2^{1/2}} \sum_{b^{(l-1)}=0}^{q-1} \Phi_{l-1}(a, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l-1)} a_{l-1}\right) |b^{(l-1)}\rangle \,.$$

We may now apply $S_{0,l-1} \cdots S_{l-2,l-1}$ within the sum to get

$$T_{l-1}|a\rangle = \frac{1}{2^{1/2}} \sum_{b^{(l-1)}=0}^{q-1} \Phi_{l-1}(a, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l-1)} a_{l-1}\right) \exp\left(\sum_{t=0}^{l-2} \pi i b_{l-1}^{(l-1)} b_t^{(l-1)} / 2^{l-1-t}\right) |b^{(l-1)}\rangle \,.$$

This accounts for the first block T_{l-1}. Applying the next block T_{l-2} yields

$$\begin{aligned}
&T_{l-2}T_{l-1}|a\rangle \\
&= \frac{1}{2^{1/2}} \sum_{b^{(l-1)}=0}^{q-1} \Phi_{l-1}(a, b^{(l-1)}) \exp\left(\pi i b_{l-1}^{(l-1)} a_{l-1}\right) \exp\left(\sum_{t=0}^{l-2} \pi i b_{l-1}^{(l-1)} b_t^{(l-1)} / 2^{l-1-t}\right) \\
&\quad \frac{1}{2^{1/2}} \sum_{b^{(l-2)}=0}^{q-1} \Phi_{l-2}(b^{(l-1)}, b^{(l-2)}) \exp\left(\pi i b_{l-2}^{(l-2)} b_{l-2}^{(l-1)}\right) \exp\left(\sum_{t=0}^{l-3} \pi i b_{l-2}^{(l-2)} b_t^{(l-2)} / 2^{l-2-t}\right) |b^{(l-2)}\rangle \\
&= \left(\frac{1}{2^{1/2}}\right)^2 \sum_{b^{(l-1)}, b^{(l-2)}=0}^{q-1} \Phi_{l-1}(a, b^{(l-1)}) \Phi_{l-2}(b^{(l-1)}, b^{(l-2)}) \exp\left(\pi i b_{l-1}^{(l-1)} a_{l-1}\right) \exp\left(\pi i b_{l-2}^{(l-2)} b_{l-2}^{(l-1)}\right) \\
&\quad \exp\left(\sum_{t=0}^{l-2} \pi i b_{l-1}^{(l-1)} b_t^{(l-1)} / 2^{l-1-t}\right) \exp\left(\sum_{t=0}^{l-3} \pi i b_{l-2}^{(l-2)} b_t^{(l-2)} / 2^{l-2-t}\right) |b^{(l-2)}\rangle
\end{aligned}$$

Now applying the remaining blocks $T_0 \cdots T_{l-3}$ we have

$$\begin{aligned}
F_r|a\rangle = \left(\frac{1}{2^{1/2}}\right)^l \sum_{b^{(l-1)}, \ldots, b^{(0)}=0}^{q-1} &\left[\prod_{k=0}^{l-1} \Phi_k\left(b^{(k+1)}, b^{(k)}\right)\right] \exp\left(\sum_{k=0}^{l-1} \pi i b_k^{(k+1)} b_k^{(k)}\right) \\
&\exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{k-1} \pi i b_k^{(k)} b_t^{(k)} / 2^{k-t}\right) |b^{(0)}\rangle \,.
\end{aligned}$$

Where $b^{(l)} := a$. We see that for the nonzero terms of this sum Lemma 2.1 implies that $b^{(l-1)}, ..., b^{(1)}$ are completely determined by $c := b^{(0)}$ and that

$$b_t^{(k)} = \begin{cases} a_t & t < k \\ \\ c_t & t \geq k \end{cases}$$

Hence

$$\begin{aligned}
F_r|a\rangle &= \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(\sum_{k=0}^{l-1} \pi i a_k c_k\right) \exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{k-1} \pi i c_k a_t / 2^{k-t}\right) |c\rangle \\
&= \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{k} \pi i c_k a_t / 2^{k-t}\right) |c\rangle \,.
\end{aligned}$$

Noting that each exponential is unaffected by adding multiples of 2π we get

$$\frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{l-1} \pi i c_k a_t / 2^{k-t} \right) |c\rangle \, .$$

Now this sum has produced the fourier transform of $|a\rangle$ where the result is in reverse order bitwise. To see this we simply reverse the bits of $|c\rangle$ to get

$$\begin{aligned} & \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{l-1} \pi i c_{l-1-k} a_t / 2^{k-t} \right) |c\rangle \\ = & \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(\sum_{k=0}^{l-1} \sum_{t=0}^{l-1} \pi i c_k a_t / 2^{l-1-k-t} \right) |c\rangle \\ = & \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(2\pi i \sum_{t=0}^{l-1} a_t 2^t \sum_{k=0}^{l-1} c_k 2^k / 2^l \right) |c\rangle \\ = & \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp\left(2\pi i a c / q \right) |c\rangle \, . \end{aligned}$$

Hence our algorithm does indeed produce the fourier transform so long as we reverse the bits of the computed result.

Referring back to (26) we see that this operation requires $l(l+1)/2 = O(l^2)$ quantum gates.

4.5 Determining the order of $x \pmod n$

Given $\gcd(x, n) = 1$ we wish to find the order of x in $\bmod n$, i.e. we wish to find the smallest positive integer r such that $x^r \equiv 1 \pmod n$.

The state of our machine will be described by two l-bit registers initialized to zero, i.e

$$|0\rangle|0\rangle \, .$$

Let l be such that $n^2 \leq q = 2^l < 2n^2$. We first apply Hadamard gates to obtain a uniform superposition of states in the first register.

$$H_0 H_1 \cdots H_{l-1} |0\rangle|0\rangle = \frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a\rangle|0\rangle \, . \tag{27}$$

We now compute $x^a \pmod n$ in the second register as described in section 4.3.

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a\rangle|x^a \pmod n\rangle \, .$$

Lastly we apply the quantum fourier transform to the first register as described in section 4.4.

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} F|a\rangle|x^a \pmod n\rangle = \frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} \exp(2\pi i a c / q) |c\rangle|x^a (\bmod n)\rangle \, .$$

The quantum aspect of the algorithm is now complete and we simply observe the value of the first register $|c\rangle$. The value of c will allow us to compute r as described below, where r is the order of $x \pmod n$.

Define $\{rc\}_q$ to be the unique integer such that $rc = dq + \{rc\}_q$ for some integer d where

$$-\frac{q}{2} < \{rc\}_q \leq \frac{q}{2}\,.$$

Note that d is a function of c. The successful computation of r from c will require the observed value of c to be in the following set:

$$\mathcal{C}_2 = \{c \in \mathbb{N} \,:\, \gcd(d, r) = 1 \text{ and } |\{rc\}_q| \leq r/2\}\,. \tag{28}$$

The assumption that d and r are relatively prime will be used later in this section. If $|\{rc\}_q| \leq r/2$ then

$$-\frac{r}{2} \leq rc - dq \leq \frac{r}{2}\,,$$

and rearranging gives

$$\left|\frac{c}{q} - \frac{d}{r}\right| \leq \frac{1}{2q}\,. \tag{29}$$

Note that since r is the order of x in mod n it follows that $1 \leq r \leq n$. let e be any integer and let s be an integer with $1 \leq s \leq n$. Also assume $d/r \neq e/s$, i.e $er - ds \neq 0$. Then

$$\left|\frac{e}{s} - \frac{d}{r}\right| = \left|\frac{er - ds}{sr}\right| \geq \frac{1}{n^2} \geq \frac{1}{q}\,.$$

Now

$$\left|\frac{c}{q} - \frac{e}{s}\right| \geq \left|\frac{e}{s} - \frac{d}{r}\right| - \left|\frac{c}{q} - \frac{d}{r}\right| \geq \frac{1}{q} - \frac{1}{2q} = \frac{1}{2q}\,, \tag{30}$$

thus from (29) and (30) we see that d/r is the nearest fraction to c/q with a denominator less than n. Let $[a_0; a_1, a_2, ..., a_k]$ be the continued fraction expansion of c/q, i.e

$$\frac{c}{q} = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots \cfrac{1}{a_k}}}} \tag{31}$$

where a_0 may be any integer and $a_1, a_2, ..., a_k$ are nonnegative integers. Since $c < q$ we have $a_0 = 0$. Note also that since $q \geq n^2 \geq r^2$ we have

$$\left|\frac{c}{q} - \frac{d}{r}\right| \leq \frac{1}{2q} \leq \frac{1}{2r^2}\,.$$

This implies that d/r is among the convergents of c/q (see [1, p 187, prop 4]), i.e d/r may be obtained by taking the continued fraction expansion $[0; a_1, a_2, ..., a_k]$

of c/q and truncating as in

$$\begin{aligned}
[0;a_1] &= \frac{c_1}{q_1} \\
[0;a_1,a_2] &= \frac{c_2}{q_2} \\
&\vdots \\
[0;a_1,a_2,...,a_k] &= \frac{c_k}{q_k} = \frac{c}{q}\,.
\end{aligned}$$

These truncations are the successive convergents of c/q. Thus to find d/r we will compute the nearest convergent to c/q with a denominator less then n.

The convergents may be computed in lowest terms using the recursion

$$\begin{aligned}
c_n &= a_n c_{n-1} + c_{n-2}\,, \\
q_n &= a_n q_{n-1} + q_{n-2}\,.
\end{aligned}$$

where $c_{-1} = 1$, $c_{-2} = 0$, $q_{-1} = 0$, $q_{-2} = 1$, and $n \geq 0$. From this we see that $q_i > q_j$ for $i > j \geq 0$. Also a simple result of convergents is that

$$\left|\frac{c}{q} - \frac{c_i}{q_i}\right| < \left|\frac{c}{q} - \frac{c_j}{q_j}\right|$$

for $i > j$, i.e each subsequent convergent is a better approximation of c/q (see [1, p 181]). Hence we find j such that $q_j \leq n < q_{j+1}$, then $d/r = c_j/q_j$. Since we assume d and r are relatively prime we may determine r from the value of $d/r = c_j/q_j$, i.e $r = q_j$.

The worst case scenario is that we must compute all q_i for $1 \leq i < k$. The a_i's in (31) are the quotients generated by applying the Euclidean algorithm to c and q, and k is thus the number of steps needed to finish the algorithm. The Euclidean algorithm costs $O(\log h)$ where h is the smaller of the two numbers. Hence k is of $O(\log c) \leq O(\log n)$. As we will see this will not alter the final asymptotic runtime.

4.6 A lower bound for observing a sufficient state to compute the order of x (mod n)

It now remains to show that the probability of observing a value for $|c\rangle$ such that $|\{rc\}_q| \leq r/2$ is sufficiently high as to make this algorithm useful. To that end suppose our algorithm results in the particular state $|c, x^k \mod n\rangle$ with $|\{rc\}_q| \leq r/2$ and $0 \leq k < r$. The probability of observing this state is

$$\left|\frac{1}{q}\sum_{a:\ x^a \equiv x^k} \exp(2\pi i a c/q)\right|^2 . \tag{32}$$

Since $x^a \equiv x^k$ if and only if $a = br + k$ for some integer b and a ranges from 0 to $q-1$ we may reformulate the above sum as

$$\left|\frac{1}{q}\sum_{b=0}^{\lfloor (q-k-1)/r \rfloor} \exp(2\pi i (br+k) c/q)\right|^2 .$$

Moreover $rc = pq + \{rc\}_q$ for some integer p. Hence we have

$$\left| \frac{1}{q} \sum_{b=0}^{\lfloor (q-k-1)/r \rfloor} \exp(2\pi i b \{rc\}_q/q) \exp(2\pi i b p) \exp(2\pi i k c/q) \right|^2 .$$

The last factor has a norm of 1 and does not depend on b and so may be removed from the sum. Also $\exp(2\pi i b p) = 1$. Thus we have

$$\left| \frac{1}{q} \sum_{b=0}^{\lfloor (q-k-1)/r \rfloor} \exp(2\pi i b \{rc\}_q/q) \right|^2 .$$

We condense our notation by writing

$$\left| \frac{1}{q} \sum_{b=0}^{Q} \exp(\beta i b) \right|^2 \tag{33}$$

where

$$Q := \lfloor (q-k-1)/r \rfloor$$

and

$$\beta := 2\pi \{rc\}_q/q \, .$$

We wish to find a lower bound for (33), and to do so we approximate it as an integral, namely

$$\frac{1}{q} \int_0^{Q+1} \exp(\beta i b) db \, .$$

We will show that the error from this approximation is small.

Lemma 4.1. *Let A and B be real constants, $f : \mathbb{R} \mapsto \mathbb{C}$ be a continuous differentiable function, N and m be nonnegative integers, and let $\Delta x = (B-A)/N$. Then*

$$\left| \int_A^B f(x) dx - \Delta x \sum_{m=0}^{N-1} f(A + m\Delta x) \right| \le \frac{1}{2} (B-A) \Delta x \sup_{A<x<B} |f'(x)| \, .$$

Proof. Note that for any continuously differentiable function we have

$$f(x) \quad = \quad f(A + m\Delta x) + [x - (A + m\Delta x)] f'(\xi_x)$$

where $A + m\Delta x \le \xi_x \le x$. Hence

$$\begin{aligned}
& \int_{A+m\Delta x}^{A+(m+1)\Delta x} f(x) dx \\
= & \int_{A+m\Delta x}^{A+(m+1)\Delta x} f(A + m\Delta x) dx + \int_{A+m\Delta x}^{A+(m+1)\Delta x} (x - (A + m\Delta x)) f'(\xi_x) dx \\
\le & \int_{A+m\Delta x}^{A+(m+1)\Delta x} f(A + m\Delta x) dx + \sup_{A<\xi<B} f'(\xi) \int_{A+m\Delta x}^{A+(m+1)\Delta x} (x - (A + m\Delta x)) dx \\
= & \Delta x f(A + m\Delta x) + \frac{1}{2} (\Delta x)^2 \sup_{A<\xi<B} f'(\xi) \, .
\end{aligned}$$

Now

$$\begin{aligned}\int_A^B f(x)dx &= \sum_{m=0}^{N-1}\int_{A+m\Delta x}^{A+(m+1)\Delta x} f(x)dx \\ &\leq \Delta x\sum_{m=0}^{N-1} f(A+m\Delta x) + \frac{1}{2}N(\Delta x)^2 \sup_{A<\xi<B} f'(\xi) \\ &= \Delta x\sum_{m=0}^{N-1} f(A+m\Delta x) + \frac{1}{2}(B-A)\Delta x \sup_{A<\xi<B} f'(\xi)\,.\end{aligned}$$

Hence

$$\left|\int_A^B f(x)dx - \Delta x\sum_{m=0}^{N-1} f(A+m\Delta x)\right| \leq \frac{1}{2}(B-A)\Delta x \sup_{A<x<B}|f'(x)|$$

□

In particular we may apply the above lemma to $\mathrm{Re}(\exp(\beta ib)) = \cos(\beta b)$ and $\mathrm{Im}(\exp(\beta ib)) = \sin(\beta b)$. Take $\Delta x = 1$, $N = Q+1$ and let

$$E := \frac{1}{q}\int_0^{Q+1}\exp(\beta ib)db - \frac{1}{q}\sum_{b=0}^{Q}\exp(\beta ib)\,.$$

Then for $|\mathrm{Re}(E)|$ we have

$$\begin{aligned}&\left|\mathrm{Re}\left(\frac{1}{q}\int_0^{Q+1}\exp(\beta ib)db - \frac{1}{q}\sum_{b=0}^{Q}\exp(\beta ib)\right)\right| \qquad (34)\\ =\ &\left|\frac{1}{q}\int_0^{Q+1}\cos(\beta b)db - \frac{1}{q}\sum_{b=0}^{Q}\cos(\beta b)\right| \\ \leq\ &\frac{(Q+1)}{2q}\sup_{0<b<Q+1}|\beta\sin(\beta b)| \\ \leq\ &\frac{(Q+1)}{2q}|\beta|\,.\end{aligned}$$

Clearly the same argument applies for $|\mathrm{Im}(E)|$. Now since $|\{rc\}_q| \leq r/2$, $r < q$, and $q \geq n^2$ we have

$$\begin{aligned}\frac{(Q+1)}{2q}|\beta| &\leq \frac{(q-k-1+r)}{2qr}\frac{2\pi r}{2q} \\ &\leq \frac{(q+q)\pi}{2q^2} \leq \frac{\pi}{q} \leq \frac{\pi}{n^2}\,.\end{aligned}$$

Thus we have

$$\left|\frac{1}{q}\int_0^{Q+1}\exp(\beta ib)db - \frac{1}{q}\sum_{b=0}^{Q}\exp(\beta ib)\right| = |E| \leq |\mathrm{Re}(E)| + |\mathrm{Im}(E)| \leq \frac{2\pi}{n^2}\,. \quad (35)$$

Thus the absolute error is of $O(1/n^2)$. Note that n will be large for any reasonable problem (recall that n is the number we wish to factor). Thus we focus now on the integral

$$\frac{1}{q}\int_0^{Q+1} \exp(\beta i b)db\,. \tag{36}$$

Performing the simple change of variable $u = rb/q$ and letting

$$\alpha := q\beta/r = 2\pi\{rc\}_q/r$$

we may rewrite the above integral as

$$\frac{1}{r}\int_0^{r(Q+1)/q} \exp(\alpha i u)du\,. \tag{37}$$

To simplify we will split the integral in two as

$$\frac{1}{r}\int_0^{1} \exp(\alpha i u)du + \frac{1}{r}\int_1^{r(Q+1)/q} \exp(\alpha i u)du\,. \tag{38}$$

We will show that the right hand integral is small and so may be neglected. Let $\epsilon := r(Q+1)/q - 1$ and note that

$$\begin{aligned}
|\exp(\alpha i\epsilon) - 1| &= \left|\int_0^{\epsilon} \alpha i \exp(\alpha i t)dt\right| \\
&\leq \sup_{0\leq t\leq\epsilon} |\alpha i \exp(\alpha i t)|\,\epsilon \\
&\leq \alpha\epsilon\,.
\end{aligned}$$

Hence the magnitude of the right hand term of (38) is

$$\begin{aligned}
\left|\frac{1}{r}\int_1^{r(Q+1)/q} \exp(\alpha i u)du\right| &= \left|\frac{1}{r\alpha i}(\exp((1+\epsilon)\alpha i) - \exp(\alpha i))\right| \\
&= \left|\frac{1}{r\alpha i}\exp(\alpha i)(\exp(\alpha i\epsilon) - 1)\right| \\
&= \left|\frac{1}{r\alpha i}\alpha\epsilon\right| \\
&= \frac{1}{r}\left[\frac{r(\lfloor (q-k-1)/r\rfloor + 1)}{q} - 1\right] \\
&\leq \frac{1}{r}\left[\frac{-k-1+r}{q}\right] \\
&\leq \frac{1}{n^2}\,.
\end{aligned}$$

The last statement above again uses the fact that $q \geq n^2$. Thus we see that the error from neglecting the right hand term of (38) is $O(1/n^2)$, i.e

$$\left|\frac{1}{r}\int_0^{r(Q+1)/q} \exp(\alpha i u)du - \frac{1}{r}\int_0^{1} \exp(\alpha i u)du\right| \leq \frac{1}{n^2}\,. \tag{39}$$

Now combining (35) and (39) we have

$$\left|\frac{1}{q}\sum_{b=0}^{Q}\exp(\beta ib)\right| = \left|\frac{1}{r}\int_0^1 \exp(\alpha iu)du\right| + O\left(\frac{1}{n^2}\right). \tag{40}$$

Hence we focus our attention on the left hand term, namely

$$\frac{1}{r}\int_0^1 \exp(\alpha iu)du\,. \tag{41}$$

We will now find an explicit lower bound for this integral. Note that if $\alpha = 0$ (i.e $\{rc\}_q = 0$) then (41) becomes $1/r$ and the probability of observing the state $|c, x^k \bmod n\rangle$ is $1/r^2$. Assuming $\alpha \neq 0$ we have

$$\begin{aligned}
\left|\frac{1}{r}\int_0^1 \exp(\alpha iu)du\right|^2 &= \left|\frac{1}{r\alpha i}(\exp(\alpha i) - 1)\right|^2 \\
&= \left|\frac{1}{r\alpha i}(\cos\alpha - 1 + i\sin\alpha)\right|^2 \\
&= \frac{1}{r^2\alpha^2}\left(\cos^2\alpha + \sin^2\alpha - 2\cos\alpha + 1\right) \\
&= \frac{2}{r^2\alpha^2}\left(1 - \cos\alpha\right) \qquad (42)
\end{aligned}$$

We wish to find the minimum value of (42) for $\alpha \in [-\pi, \pi]$, $\alpha \neq 0$. Noting that (42) is independent of the sign of α we take $\alpha \in (0, \pi]$. Taking the derivative of (42) with respect to α gives

$$\frac{2}{r^2}\left[\alpha^{-2}\sin\alpha - 2\alpha^{-3}(1 - \cos\alpha)\right].$$

This derivative is negative for all $\alpha \in (0, \pi)$. This can be seen by noting that

$$\tan(\alpha/2) = \frac{1 - \cos\alpha}{\sin\alpha},$$

and so

$$\begin{aligned}
&\alpha^{-2}\sin\alpha - 2\alpha^{-3}(1 - \cos\alpha) < 0 \\
\iff \qquad & 1 - 2\frac{1 - \cos\alpha}{\alpha\sin\alpha} < 0 \\
\iff \qquad & \tan(\alpha/2) > \alpha/2\,.
\end{aligned}$$

The last statement is true for all $\alpha \in (0, \pi)$. Thus we see that (42) is decreasing on $\alpha \in (0, \pi]$, and so by symmetry the minimum value over $\alpha \in [-\pi, \pi]$ of (42) occurs at $\alpha = \pm\pi$. This minimum value is $4/(\pi^2 r^2)$, i.e

$$\left|\frac{1}{r}\int_0^1 \exp(\alpha iu)du\right|^2 \geq \frac{4}{\pi^2 r^2},$$

and thus

$$\left|\frac{1}{r}\int_0^1 \exp(\alpha i u)du\right| \geq \frac{2}{\pi r}\,.$$

Now recalling that $r < n$ we have

$$\begin{aligned}\left|\frac{1}{q}\sum_{b=0}^{Q}\exp(\beta i b)\right| &= \left|\frac{1}{r}\int_0^1 \exp(\alpha i u)du\right| + O\left(\frac{1}{n^2}\right)\\ &\geq \frac{4}{\pi r} + O\left(\frac{1}{n^2}\right)\\ &\geq \frac{K}{r}\end{aligned}$$

for some $K > 0$ and sufficiently large n. Thus the probability of observing a state $|c, x^k \bmod n\rangle$ with $|\{rc\}_q| \leq r/2$ is

$$\left|\frac{1}{q}\sum_{b=0}^{Q}\exp(\beta i b)\right|^2 \geq \frac{K^2}{r^2}\,. \tag{43}$$

This lower bound will be used in the next section to determine the final computational cost of the algorithm.

4.7 Computational cost

Recall that $n^2 \leq q = 2^l < 2n^2$, hence

$$l = \log q < \log 2 + 2\log n < 3\log n$$

for $n > 2$. The cost to compute $\gcd(a, b)$ for $a \leq b$ is $O(\log a)$. The cost to run one iteration is laid out in the table below:

process	cost
$\gcd(x, n)$	$\mathrm{O}(\log n)$
$H^{(l-1)} = H_0 \cdots H_{l-1}$	$\mathrm{O(l)}$=$\mathrm{O}(\log n)$
mod exponentiation	$\mathrm{O(l^2 \log l \log\log l)} = O((\log n)^2 \log\log n \log\log\log n)$
fourier transform	$\mathrm{O(l^2)} = O((\log n)^2)$
$\gcd(c, q)$	$\mathrm{O}(\log q) = O(\log n)$
$\gcd(x^{r/2}+1, n)$	$\mathrm{O}(\log n)$

Adding these up we see the cost of a single iteration is

$$O((\log n)^2 \log\log n \log\log\log n)\,.$$

Given r we may compute a nontrivial factor of n as described in section (4.2) so long as $x \in \mathcal{C}_1$ where

$$\mathcal{C}_1 = \{x \in \mathbb{N} \,:\, r \text{ is even and } x^{r/2} + 1 \not\equiv 0 \pmod n\}\,.$$

As shown in (22) we have

$$\mathrm{prob}(x \in \mathcal{C}_1) \geq 1 - \frac{1}{2^{k-1}}$$

where k is the number of distinct prime factors of n.

We will now enumerate the number of states $|c, x^k \pmod n\rangle$ that allow us to compute r. Recall that $rc = dq + \{rc\}_q$ for some unique integer d. Since $0 \leq c \leq q-1$ we have

$$0 \leq d \leq r\frac{q-1}{q} - \frac{\{rc\}_q}{q} < r\,.$$

Recall from (28) that the state $|c, x^k \pmod n\rangle$ is sufficient to compute r if $r \in \mathcal{C}_2$ where

$$\mathcal{C}_2 = \{c \in \mathbb{N} \,:\, \gcd(d, r) = 1 \text{ and } |\{rc\}_q| \leq r/2\}\,.$$

Suppose that d is some integer and note that there exists some integer j such that

$$rj \leq dq \leq r(j+1)\,.$$

Then

$$|rc - dq| \leq \frac{r}{2} \leq \frac{q}{2}$$

for at least one of $c = j$ or $c = j+1$. Thus for each d there exists a c such that

$$|\{rc\}_q| = |rc - dq| \leq \frac{r}{2}\,.$$

Hence since there are $\phi(r)$ values of d such that $\gcd(d, r) = 1$ where ϕ is Euler's totient function. And so it follows there are $\phi(r)$ values of c such that $c \in \mathcal{C}_2$.

Note also that there are r possible values of $x^k \pmod n$. Thus altogether there are at least $r\phi(r)$ states $|c, x^k \pmod n\rangle$ such that $c \in \mathcal{C}_2$, and the probability of observing each of these states is at least K^2/r^2 for some constant $K > 0$ as expressed in (43).

Now assuming $x \in \mathcal{C}_1$ the probability of observing a state $|c, x^k \pmod n\rangle$ such that $c \in \mathcal{C}_2$ is

$$\text{prob}(c \in \mathcal{C}_2 | x \in \mathcal{C}_1) \geq \frac{K^2}{r^2} r\phi(r) = \frac{K^2\phi(r)}{r} \geq \frac{\delta}{\log\log r} \geq \frac{\delta}{\log\log n}$$

where $\delta > 0$ is a constant [4, Thm. 328].

The algorithm will succeed only if both $x \in \mathcal{C}_1$ and $c \in \mathcal{C}_2$, thus the probability of a successful run is

$$\text{prob}(x \in \mathcal{C}_1 \text{ and } c \in \mathcal{C}_2) = \text{prob}(x \in \mathcal{C}_1)\text{prob}(c \in \mathcal{C}_2 | x \in \mathcal{C}_1) \geq \frac{1}{2}\frac{\delta}{\log\log n}\,.$$

Hence if we repeat the algorithm an $O(\log\log n)$ number of times we stand a high probability of success. Now one run of the algorithm requires

$$O((\log n)^2(\log\log n)(\log\log\log n))$$

time, hence the final expected runtime is

$$O((\log n)^2(\log\log n)^2(\log\log\log n)) \leq O[(\log n)^3]\,.$$

5 Conclusion

Grover's algorithm and Shor's algorithm each represent a revolutionary paradigm shift in the way in which a computer may be employed to solve a problem. This shift entails not only a fundamental advancement in the physical hardware used to construct the machine, but also a fundamental change in the manner in which algorithms must be designed to utilize this new hardware.

The hardware revolution comes in the form of the physical implementation of a device capable of exhibiting quantum superposition, which lies at the heart of the power of quantum computation. Quantum superposition allows massive parallel processing with no additional resources. In order to exploit this power however algorithms must be probabilistic in nature. This gives rise to the almost paradoxical circumstance that two separate runs of an algorithm, given identical data, may produce two different outputs.

Thus superposition is the source of not only a quantum machines enhanced power but also potentially its only weakness. In order to achieve dramatically faster algorithms we must overcome an inherent uncertainty in our results. This uncertainty was successfully overcome in the case of the two algorithms discussed in this paper. However, it is not known what range of problems the benefits of quantum computation may be applicable.

In 2001 IBM successfully produced the first implementation of Shor's algorithm on a quantum computer. The algorithm was employed to factor the number 15 on a 7-qubit machine. Since that time there has been steady progress towards a commercially viable quantum machine. In the last few years the rate of this progress has increased dramatically. In the near future quantum algorithms will no doubt leave the theoretical realm to play an increasingly significant role in real world applications.

References

[1] W.A. Coppel. *Number theory: an introduction to mathematics*. Springer Verlag, 2009.

[2] Knuth D.E. *The art of computer programming*. Addison-Wesley, City, 2001.

[3] L.K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, page 219. ACM, 1996.

[4] G.H. Hardy, E.M. Wright, D.R. Heath-Brown, and J.H. Silverman. *An introduction to the theory of numbers*, volume 6. Clarendon press Oxford, 1979.

[5] P.W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.