Quantum Computing

Kevin Chen

December 25, 2021

Contents

Foreword

The purpose of these notes is mainly for personal reference, so the overall presentation will be very terse. I hope this may also serve as a useful resource for others. The primary reference is John Watrous's lecture notes, which can be found at https://cs.uwaterloo.ca/ watrous/QC-notes/. This is supplemented in some places by John Preskill's lecture notes, which can be found at https://theory.caltech.edu/ preskill/ph229/.

1 Quantum mechanics

We first review some basic facts about quantum mechanics that will be important in discussing quantum computing. Knowledge of quantum mechanics and notation is assumed.

1. A **qubit** is a state (i.e. a vector) of a two-dimensional complex vector space. A general qubit $|\psi\rangle$ can be written as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \ , \tag{1.1}$$

where $|0\rangle$, $|1\rangle$ are the basis vectors and α, β are complex numbers. Note that the qubit is normalized:

$$|\alpha|^2 + |\beta|^2 = 1 \ . \tag{1.2}$$

2. A system of two qubits is represented by taking the *tensor product* of the qubits,

$$(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (\gamma\,|0\rangle + \delta\,|1\rangle) = \begin{matrix} \alpha\gamma\,|00\rangle \\ +\alpha\delta\,|01\rangle \\ +\beta\gamma\,|10\rangle \\ +\beta\delta\,|11\rangle \end{matrix} \ . \tag{1.3}$$

This implies that a system of n qubits forms a state of a 2^n-dimensional complex vector space. In what follows, we will often omit $\otimes$ and use the shorthand $|0\rangle \otimes |1\rangle \equiv |0\rangle\,|1\rangle \equiv |01\rangle$.

3. The evolution of a system of qubits is represented by the application of a *unitary operator*,

$$|\Psi\rangle \to U\,|\Psi\rangle \ , \tag{1.4}$$

where U is a complex-valued matrix that satisfies

$$UU^\dagger = U^\dagger U = I \ . \tag{1.5}$$

4. The measurement of a qubit yields one of two outcomes: 0 or 1. For the qubit described in (1.1), a measurement of 0 occurs with probability $|\alpha|^2$ and a measurement of 1 occurs with probability $|\beta|^2$. After the measurement, the qubit is altered and takes on the state corresponding to the measured outcome, either $|0\rangle$ or $|1\rangle$.

Intuitively, a quantum computation is carried out by (i) preparing some input qubits, (ii) applying unitary operators to those qubits, and then (iii) measuring the result at the end.

2 Circuits

In this section, we establish some common language to describe quantum algorithms, as well as explore some important consequences of quantum mechanics, such as quantum teleportation and no-cloning.

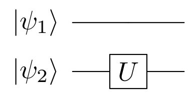
Let us define some useful unitary matrices:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} , \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} , \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} , \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} . \tag{2.1}$$

Note that X is the NOT operator, i.e. $|0\rangle$ is mapped to $|1\rangle$, and vice versa. H is called the **Hadamard matrix**. All four matrices square to the identity matrix, I.

The circuit below shows the application of a unitary operator U on the second qubit of a two-qubit system. This is equivalent to the two-qubit unitary operator $I \otimes U$, which can be represented as a matrix on the ordered basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$,

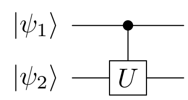
$$I \otimes U = \begin{pmatrix} u_{00} & u_{01} & 0 & 0 \\ u_{10} & u_{11} & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix} . \tag{2.2}$$



(2.3)

A qubit can control the application of a unitary operator on another qubit. The circuit below shows the application of U on the second qubit only when the first qubit is $|1\rangle$. This two-qubit unitary operator is represented by the matrix,

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix} . \tag{2.4}$$

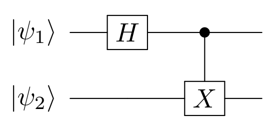


(2.5)

The **Bell states** are four mutually-orthogonal maximally entangled[1] states,

$$\left|\Phi^{\pm}\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle) ,$$
$$\left|\Psi^{\pm}\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle) . \tag{2.6}$$

These states can be obtained from the basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ using the circuit,



(2.7)

$$|00\rangle \to \left|\Phi^+\right\rangle , \qquad |01\rangle \to \left|\Psi^+\right\rangle , \qquad |10\rangle \to \left|\Phi^-\right\rangle , \qquad |11\rangle \to \left|\Psi^-\right\rangle . \tag{2.8}$$

We will now describe three phenomena related to entanglement.

2.1 Dense coding

Holveo's theorem states that the amount of classical information that can be retrieved from n qubits is n bits. However, using entanglement, one can send two bits of classical information by sending only *one* qubit.

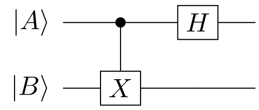
- Alice wants to send two bits of information to Bob.
- The state $|\Phi^+\rangle$ is created from two qubits $|A\rangle$ and $|B\rangle$. $|A\rangle$ is sent to Alice and $|B\rangle$ is sent to Bob.
- Alice prepares one of the four Bell states by applying single-qubit operators on $|A\rangle$:

$$Z : \begin{matrix} |\Phi^+\rangle \leftrightarrow |\Phi^-\rangle , \\ |\Psi^+\rangle \leftrightarrow |\Psi^-\rangle , \end{matrix} \qquad X : \begin{matrix} |\Phi^+\rangle \leftrightarrow |\Psi^+\rangle , \\ |\Phi^-\rangle \leftrightarrow -|\Psi^-\rangle . \end{matrix} \tag{2.9}$$

Then Alice sends qubit $|A\rangle$ to Bob.

[1] A two-qubit state $|\psi_{AB}\rangle$ is **maximally entangled** when the partial trace of $|\psi_{AB}\rangle \langle\psi_{AB}|$ over any one of the qubits is proportional to the identity matrix. Intuitively, a measurement on only one qubit acquires no information.

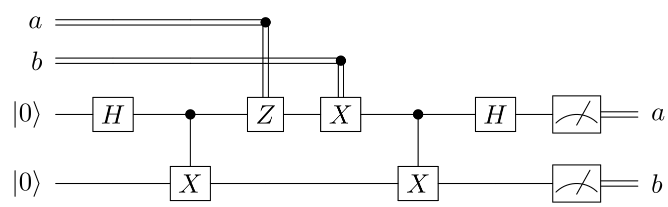
- Bob applies the circuit,



(2.10)

to undo the circuit in (2.7) and map the four Bell states back to the basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. Bob can then perform measurements to determine which state he has and read off Alice's message.

This entire operation can be described by a circuit. Suppose the two bits Alice wants to send are ab. Double lines in the circuit represent classical bits, and the meter represents a measurement.



(2.11)

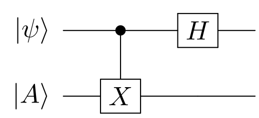
2.2 Teleportation

Using entanglement, a qubit can be teleported by sending two bits of classical information.

- Alice has a qubit $|\psi\rangle$ that she wants to give to Bob.
- The state $|\Phi^+\rangle$ is created from two qubits $|A\rangle$ and $|B\rangle$. $|A\rangle$ is sent to Alice and $|B\rangle$ is sent to Bob.
- If we let $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the state describing all three qubits is

$$(\alpha|0\rangle + \beta|1\rangle)\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle)\ . \tag{2.12}$$

  Alice applies the circuit below on the two qubits $|\psi\rangle$, $|A\rangle$ in her possession:



(2.13)

  The three-qubit state becomes

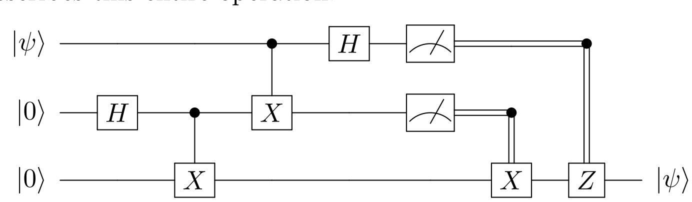
$$\frac{1}{2}\Big[|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\beta|0\rangle + \alpha|1\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(-\beta|0\rangle + \alpha|1\rangle)\Big]\ . \tag{2.14}$$

  Alice makes a measurement on her two qubits to obtain two classical bits ab, which she sends to Bob. This measurement causes qubit $|B\rangle$ to be one of four possibilities, depending on ab. The possibilities are tabulated below.

ab	$\|B\rangle$
00	$\alpha\|0\rangle + \beta\|1\rangle$
01	$\beta\|0\rangle + \alpha\|1\rangle = X(\alpha\|0\rangle + \beta\|1\rangle)$
10	$\alpha\|0\rangle - \beta\|1\rangle = Z(\alpha\|0\rangle + \beta\|1\rangle)$
11	$-\beta\|0\rangle + \alpha\|1\rangle = XZ(\alpha\|0\rangle + \beta\|1\rangle)$

- Given the bits ab, Bob knows exactly what state his qubit $|B\rangle$ is in. He can apply single-qubit operators on $|B\rangle$ to obtain the original state $\alpha|0\rangle + \beta|1\rangle$ that Alice had.

The circuit below describes this entire operation.



(2.15)

2.3 No-cloning

Note that in teleportation, the original $|\psi\rangle$ qubit held by Alice is destroyed. It is not possible to clone a qubit. More precisely, there is no unitary operator that maps $|\psi\rangle \otimes |0\rangle$ to $|\psi\rangle \otimes |\psi\rangle$ for all qubits $|\psi\rangle$.

To prove this, let $|\psi\rangle$, $|\phi\rangle$ be two different qubits where $\langle\psi|\phi\rangle \neq 0$.

$$\begin{aligned}\langle\psi|\phi\rangle &= (\langle\psi| \otimes \langle 0|)(|\phi\rangle \otimes |0\rangle) \\ &= (\langle\psi| \otimes \langle 0|)U^\dagger U(|\phi\rangle \otimes |0\rangle) \\ &= (\langle\psi| \otimes \langle\psi|)(|\phi\rangle \otimes |\phi\rangle) \\ &= \langle\psi|\phi\rangle^2 \ .\end{aligned} \tag{2.16}$$

This implies that $\langle\psi|\phi\rangle = 1$, i.e. they are the same qubit, which is a contradiction.

3 Algorithms

We will now describe some quantum algorithms that scale better than their classical counterparts.

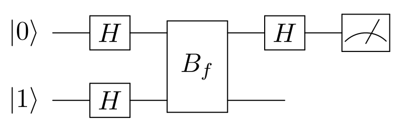
3.1 Deutsch's algorithm

Given a function $f : \{0,1\} \to \{0,1\}$ on one bit, determine whether it is **constant** (i.e. same output for all inputs) or **balanced** (i.e. the number of inputs for which f equals 0 and 1 are equal).

Classically, two queries of f are needed to solve this problem. Quantum mechanically, if we have access to the two-qubit operator,[2]

$$B_f\,|a\rangle\,|b\rangle = |a\rangle\,|b \oplus f(a)\rangle \ , \tag{3.1}$$

where $\oplus$ is the XOR operator, then this problem can be solved in one query using the circuit,



(3.2)

Let us track the two-qubit state through this circuit. The initial state is $|0\rangle\,|1\rangle$. The state after the initial Hadamard gates is

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}\,|0\rangle\,(|0\rangle - |1\rangle) + \frac{1}{2}\,|1\rangle\,(|0\rangle - |1\rangle) \ . \tag{3.3}$$

B_f maps this state to

$$\begin{aligned}&\frac{1}{2}\,|0\rangle\,(|0 \oplus f(0)\rangle - |1 \oplus f(0)\rangle) + \frac{1}{2}\,|1\rangle\,(|0 \oplus f(1)\rangle - |1 \oplus f(1)\rangle) \\ &\quad = \frac{1}{2}(-1)^{f(0)}\,|0\rangle\,(|0\rangle - |1\rangle) + \frac{1}{2}(-1)^{f(1)}\,|1\rangle\,(|0\rangle - |1\rangle) \\ &\quad = \frac{1}{\sqrt{2}}\left((-1)^{f(0)}\,|0\rangle + (-1)^{f(1)}\,|1\rangle\right)\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \ .\end{aligned} \tag{3.4}$$

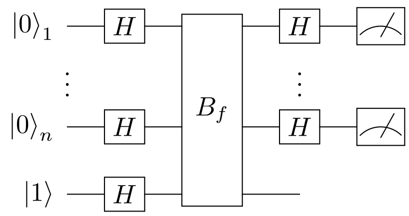
Note that the second qubit is independent of f and always equals $H\,|1\rangle$. This qubit is discarded in the computation. Applying a final Hadamard gate to the first qubit gives us the state $(-1)^{f(0)}\,|f(0) \oplus f(1)\rangle$. A measurement of this state determines whether $f(0) \oplus f(1) = 0$ (constant) or $= 1$ (balanced).

[2] B_f is unitary because it is a permutation operator; the states $|a\rangle\,|b\rangle$ and $|a\rangle\,|b \oplus f(a)\rangle$ are mapped to each other under B_f. This will be so for any function f.

3.2 Deutsch-Jozsa algorithm

Given a function $f : \{0,1\}^n \to \{0,1\}$ on n bits, determine whether it is constant or balanced. f is guaranteed to be one of these two cases.

This is a generalization of Deutsch's problem. Classically, up to $2^{n-1}+1$ queries are needed to solve this problem in the worst-case scenario. Probabilistically, with k queries we can solve this problem with a probability of error $\leq 2^{1-k}$ by guessing "constant" when all k outputs are equal. Quantum mechanically, if we have access to the $(n+1)$-qubit operator B_f, which is defined in the same way as in (3.1) where a represents an n-bit vector, then this problem can be solved in one query using the circuit,



(3.5)

Let us track the state through this circuit. The state after the initial Hadamard gates is

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle \, \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \ . \tag{3.6}$$

B_f maps this to

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle \, \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \ . \tag{3.7}$$

Discarding the final qubit and operating the final Hadamard gates, we have

$$\begin{aligned} \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} H^{\otimes n} |x\rangle &= \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} \left(\frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle \right) \\ &= \sum_{y \in \{0,1\}^n} \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x) + x \cdot y} \right) |y\rangle \ , \end{aligned} \tag{3.8}$$

where $x \cdot y$ is interpreted as the dot product of two n-bit vectors x and y, taken modulo 2. Note that the amplitude for the $|y\rangle = |0^n\rangle$ state simplifies when f is constant or balanced:

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} = \begin{cases} (-1)^{f(0)} & \text{if } f \text{ is constant,} \\ 0 & \text{if } f \text{ is balanced.} \end{cases} \tag{3.9}$$

Therefore a measurement of the n output qubits produces an n-bit vector, which are all 0 only when f is constant. Otherwise, f is balanced.

3.3 Bernstein-Vazirani algorithm

Given a function $f : \{0,1\}^n \to \{0,1\}$ defined by

$$f(x) = x \cdot s \ ,$$

where s is a secret n-bit vector, determine s.

This is a variation on the Deutsch-Jozsa problem, since when $s \neq 0$ iff the function f is balanced. Classically, n queries are needed to solve this problem. Quantum mechanically, this problem can be solved in one query using the same circuit as for the Deutsch-Jozsa algorithm. In the last step, the final state is

$$\sum_{y \in \{0,1\}^n} \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot (s+y)} \right) |y\rangle \tag{3.10}$$

The amplitude is non-zero only for $|y\rangle = |s\rangle$. Therefore, a measurement of the n output qubits produces the n-bit vector s.

This is the first problem that we have discussed so far for which the quantum algorithm, which requires $\mathcal{O}(1)$ queries, is faster than the probabilistic/classical algorithm, which requires $\mathcal{O}(n)$ queries.

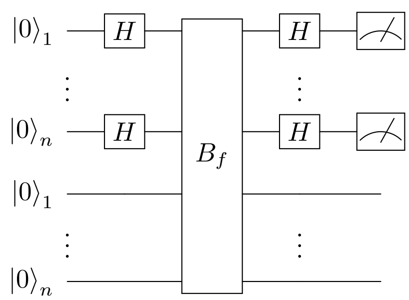
3.4 Simon's algorithm

Given a function $f : \{0,1\}^n \to \{0,1\}^n$ that has a **period** s, i.e.

$$y = x \oplus s \qquad \Longleftrightarrow \qquad f(y) = f(x) \ ,$$

determine s. Here x, y, and s are all n-bit vectors and the operation $\oplus$ is taken element-wise.

Classically, this problem requires exponentially many queries to solve, even for a probabilistic solution—intuitively, the reason why is that the chance that two random inputs create the same output is 2^{-n}. Quantum mechanically, this problem can be solved in $\mathcal{O}(n)$ queries of the $2n$-qubit operator B_f given in (3.1), where a and b both represent n-bit vectors. Consider the following circuit:



(3.11)

Let us track the state through this circuit. The state after the initial Hadamard gates is

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle \, |0^n\rangle \ . \tag{3.12}$$

B_f maps this to

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle \, |f(x)\rangle \ . \tag{3.13}$$

The state after the final Hadamard gates is

$$\begin{aligned} \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} H^{\otimes n} |x\rangle \, |f(x)\rangle &= \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} \left(\frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle \right) |f(x)\rangle \\ &= \sum_{y \in \{0,1\}^n} |y\rangle \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot y} |f(x)\rangle \right) \ . \end{aligned} \tag{3.14}$$

If $s = 0$, then f is a bijective function and the term inside the parentheses is a sum over 2^n distinct states with coefficients that are either $\pm 2^{-n}$. Therefore, the measurement of y gives a random n-bit vector with uniform probability.

If $s \neq 0$, then f is a two-to-one function. Let $X \subset \{0,1\}^n$ be a maximal set on which f is bijective. In other words, for each pair x and $x \oplus s$ that map to the same output $f(x)$, X contains exactly one element of the pair. Then,

$$\begin{aligned}\frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot y}\left|f(x)\right\rangle &= \frac{1}{2^n}\sum_{x\in X}\Big[(-1)^{x\cdot y}\left|f(x)\right\rangle + (-1)^{(x+s)\cdot y}\left|f(x+s)\right\rangle\Big] \\ &= \frac{1}{2^{n-1}}\sum_{x\in X}(-1)^{x\cdot y}\left[\frac{1+(-1)^{s\cdot y}}{2}\right]\left|f(x)\right\rangle \end{aligned} \tag{3.15}$$

For y that satisfy $s \cdot y = 0$, this is a sum over 2^{n-1} distinct states with coefficients that are either $\pm 2^{-(n-1)}$. Otherwise, the coefficient is zero when $s \cdot y \neq 0$. Therefore, the measurement of y gives a random n-bit vector that satisfies $s \cdot y = 0$ with uniform probability.

The strategy is to run this circuit until we collect $n-1$ linearly independent measurements $y_1, \ldots, y_{n-1}$. This will be very quick, since if we already have k linearly independent vectors, the probability of obtaining a new linearly independent vector (if one exists) is $1 - 2^{k-n} \geq 50\%$. The solution to the linear system of equations $s \cdot y_i = 0$ for $i = 1, \ldots, n-1$ gives a unique non-zero vector $\hat{s}$. Next, we run the circuit a couple more times to make sure there is not an nth linearly independent vector y_n. If we do find y_n, then we conclude $s = 0$. Otherwise, we conclude $s = \hat{s}$. Since the probability of finding y_n, if it exists, is 50%, with k more runs the probability of an incorrect conclusion is 2^{-k}. Thus we can find the period in $\mathcal{O}(n)$ queries with an arbitrarily low probability of error.

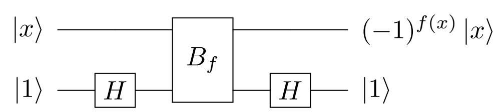
3.5 Grover's algorithm

Given a function $f : \{0,1\}^n \to \{0,1\}$, find an x such that $f(x) = 1$, if such a x exists.

Classically, 2^n queries are needed to solve this problem in the worst-case scenario. Probabilistically, we still need $\mathcal{O}(2^n)$ queries for any finite probability of success. Quantum mechanically, this problem can be solved in $\mathcal{O}\big(2^{n/2}\big)$ queries of the $(n+1)$-qubit operator B_f, which is defined in the same way as in (3.1) where a represents an n-bit vector. Using B_f, we can implement the n-qubit operator,

$$Z_f : |x\rangle \mapsto (-1)^{f(x)}\,|x\rangle \ , \tag{3.16}$$

using a circuit similar to the one used for the Deutch-Josza algorithm:



(3.17)

Note that Z_f can be viewed as a reflection that flips any "good vector" $|a\rangle \mapsto -|a\rangle$, where $f(a) = 1$, and leaves "bad vectors" $|b\rangle$ unchanged, where $f(b) = 0$. Let A be the set of "good vectors" and B be the set of "bad vectors". We can alternatively write

$$Z_f = I - \sum_{a\in A} 2\,|a\rangle\langle a| = \sum_{b\in B} 2\,|b\rangle\langle b| - I \ . \tag{3.18}$$

Next, we define another n-qubit operator,

$$Z_h = 2\,|h\rangle\langle h| - I \ , \qquad |h\rangle \equiv \frac{1}{2^{n/2}}\sum_{x\in\{0,1\}^n}|x\rangle \ , \tag{3.19}$$

which represents a reflection preserving the component parallel to $|h\rangle$. This operator can be efficiently implemented by $Z_h = H^{\otimes n} Z_0 H^{\otimes n}$ where $Z_0 = 2\,|0\rangle^n\langle 0|^n - I$ is negative the n-qubit operator that sends $|0^n\rangle \mapsto -|0^n\rangle$ but leaves all other vectors unchanged.

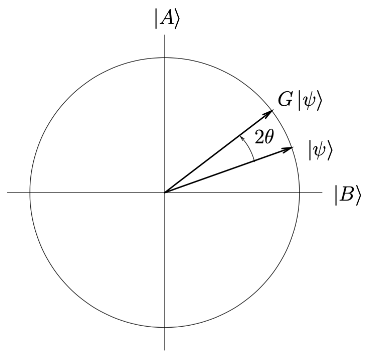
To summarize, Z_f is a reflection that preserves the component parallel to the hyperplane spanned by B, and Z_h is a reflection that preserves the component parallel to $|h\rangle$. Let us define the normalized vectors,

$$|A\rangle = \frac{1}{\sqrt{|A|}} \sum_{a \in A} |a\rangle \ , \qquad |B\rangle = \frac{1}{\sqrt{|B|}} \sum_{b \in B} |b\rangle \ . \tag{3.20}$$

$|h\rangle$ is a (real) linear combination of $|A\rangle$ and $|B\rangle$. On the (real) two-dimensional subspace spanned by $\{|A\rangle, |B\rangle\}$, the composition of two reflections $G = Z_h Z_f$ is a rotation through an angle 2θ, where θ is the acute angle between $|h\rangle$ and $|B\rangle$:

$$\cos\theta = \langle h|B\rangle = \frac{\sqrt{|B|}}{2^{n/2}} \qquad \Longrightarrow \qquad \sin\theta = \frac{\sqrt{|A|}}{2^{n/2}} \ . \tag{3.21}$$

This is sketched below. By repeatedly applying G on the vector $|\psi\rangle = |h\rangle$, the resulting vector will eventually be almost parallel to $|A\rangle$. A measurement at that time will have a high probability of yielding an element of A, that is, an n-bit vector that solves the problem.



To be concrete, let us consider the case where $|A| = 1$. After k iterations of G on $|h\rangle = \cos\theta\,|A\rangle + \sin\theta\,|B\rangle$, the state is

$$G^k\,|h\rangle = \cos[(2k+1)\theta]\,|A\rangle + \sin[(2k+1)\theta]\,|B\rangle \ . \tag{3.22}$$

Thus, we should pick $k \approx \pi/4\theta - 1/2$ to maximize the amplitude of $|A\rangle$. We can always pick a k such that the amplitude of $|B\rangle$ is less than $\sin\theta$. Taking a measurement, the probability of erroneously obtaining an element of B is $\leq 2^{-n}$. To summarize Grover's algorithm,

- Start with the n-qubit state $|h\rangle = H^{\otimes n}\,|0^n\rangle$.
- Apply the operator $G = Z_h Z_f$ a total of $k \approx 2^{n/2}\pi/4 - 1/2$ times.
- Measure the state to obtain an n-bit vector x. With high probability, $f(x) = 1$.
- If $f(x) = 0$, repeat the steps above as needed. If no x such that $f(x) = 1$ is found after a couple iterations, then we can conclude that $f(x) = 0$ for all inputs x.

3.6 Quantum Fourier transform

In preparation of discussing Shor's algorithm, we will introduce the **quantum Fourier transform** (QFT).

Given a function f on 2^n values $x = 0, 1, \ldots, 2^n - 1$, compute the discrete Fourier transform,

$$\mathcal{F}(y) = \sum_{x=0}^{2^n - 1} e^{2\pi i x y / 2^n} f(x) \ ,$$

for $y = 0, 1, \ldots, 2^n - 1$.

Classically, the $\mathcal{F}(y)$ coeffecients can be computed in $\mathcal{O}(n2^n)$ time using the *fast Fourier transform.* Quantum mechanically, we can do this in $\mathcal{O}\big(n^2\big)$ time, in the sense that the n-qubit operator,

$$\mathrm{QFT}\,|x\rangle = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} e^{2\pi i x y/2^n}\,|y\rangle\ , \tag{3.23}$$

can be implemented with $\mathcal{O}\big(n^2\big)$ gates. Note that here, x and y are integers where $0 \leq x, y < 2^n$, not n-bit vectors. This notation will be used for the remainder of this section, unless stated otherwise. We can also express the QFT as an $2^n \times 2^n$ matrix,

$$\mathrm{QFT} = \frac{1}{2^{n/2}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{2^n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(2^n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{2^n-1} & \omega^{2(2^n-1)} & \cdots & \omega^{(2^n-1)^2} \end{pmatrix}\ , \tag{3.24}$$

where $\omega \equiv e^{2\pi i/2^n}$. In retrospect, we can observe that all the algorithms we have discussed so far, except Grover's, actually rely on the QFT since the Hadamard gate can be viewed as the QFT on a single qubit. Although Grover's algorithm technically uses Hadamard gates, the general technique is better classified as "amplitude amplification".

Let us now implement the QFT operator. We first decompose the integers $0 \leq x, y < 2^n$ into their constituent bits,

$$\begin{aligned} x &= 2^{n-1}x_{n-1} + \cdots + 2x_1 + x_0\ , \\ y &= 2^{n-1}y_{n-1} + \cdots + 2y_1 + y_0\ . \end{aligned} \tag{3.25}$$

$$\tag{3.26}$$

Note that in computing $e^{2\pi i x y/2^n}$, when taking the product xy we can throw away any multiples of 2^n. Thus,

$$\frac{xy}{2^n} \equiv y_{n-1}(.x_0) + y_{n-2}(.x_1x_0) + \cdots + y_0(.x_{n-1}\ldots x_1x_0) \pmod 1\ , \tag{3.27}$$

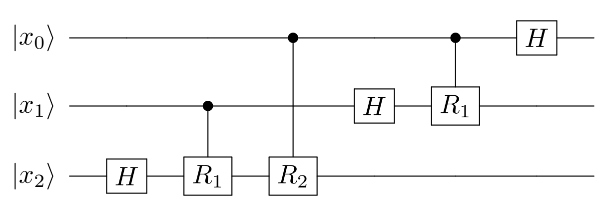
where the notation $(.abc\ldots) \equiv 2^{-1}a + 2^{-2}b + 2^{-3}c + \cdots$ represents the decimal expansion in base 2. With this observation, the QFT can be expanded as a tensor product of n qubits,

$$\sum_{y=0}^{2^n-1} e^{2\pi i x y/2^n}\,|y\rangle = \Big(|0\rangle + e^{2\pi i(.x_0)}\,|1\rangle\Big)\Big(|0\rangle + e^{2\pi i(.x_1x_0)}\,|1\rangle\Big)\cdots\Big(|0\rangle + e^{2\pi i(.x_{n-1}\ldots x_1x_0)}\,|1\rangle\Big)\ . \tag{3.28}$$

If we define the single-qubit operator,

$$R_d = \begin{pmatrix} 1 & 0 \\ 0 & e^{\pi i/2^d} \end{pmatrix}\ , \tag{3.29}$$

then the QFT can be constructed out of $n(n-1)/2$ controlled applications of these operators and n Hadamard gates; shown below is the $n = 3$ case.



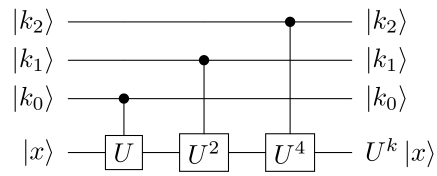
(3.30)

3.7 Phase estimation

This algorithm is used as a subroutine for many other algorithms, including Shor's algorithm, and relies on the QFT. It solves the following problem: *given an n-qubit unitary operator U and an eigenvector* $U|\Psi\rangle = e^{2\pi i\theta}|\Psi\rangle$, *estimate* $\theta \in [0,1)$. We will show that it is possible to evaluate θ to m bits of precision, where m can be arbitrarily large. However, the algorithm requires the implementation of an $(m+n)$-qubit operator,

$$\Lambda_m(U) : |k\rangle \otimes |x\rangle \mapsto |k\rangle \otimes U^k |x\rangle \quad , \tag{3.31}$$

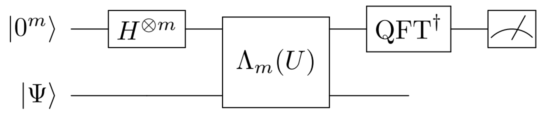
where $k = 0, 1, \ldots, 2^m - 1$. This can be implemented for general U with $\mathcal{O}(2^m)$ controlled applications of U; shown below is the $m = 3$ case.



(3.32)

More efficient implementations for $\Lambda_m(U)$ may exist for certain operators U.

Then, consider the following circuit:



(3.33)

Let us track the state through this circuit. The state after the initial Hadamard gates is

$$\frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} |k\rangle |\Psi\rangle \quad . \tag{3.34}$$

$\Lambda_m(U)$ maps this to

$$\frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} |k\rangle U^k |\Psi\rangle = \frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} e^{2\pi i k\theta} |k\rangle |\Psi\rangle \quad . \tag{3.35}$$

We disregard the $|\Psi\rangle$ part of the state. Applying the inverse of QFT, we have

$$\frac{1}{2^m} \sum_{j=0}^{2^m-1} \sum_{k=0}^{2^m-1} e^{2\pi i k(\theta - j/2^m)} |j\rangle \quad . \tag{3.36}$$

The measurement of j occurs with probability,

$$\begin{aligned} p_j &= \frac{1}{2^{2m}} \left| \sum_{k=0}^{2^m-1} e^{2\pi i k(\theta - j/2^m)} \right|^2 \\ &= \frac{1}{2^{2m}} \left| \frac{e^{2\pi i 2^m(\theta - j/2^m)} - 1}{e^{2\pi i(\theta - j/2^m)} - 1} \right|^2 \\ &= \frac{\sin^2(2^m\delta)}{2^{2m}\sin^2\delta} \ , \end{aligned} \tag{3.37}$$

where we have defined $\delta \equiv \pi(\theta - j/2^m)$. This probability is maximized for the value of j closest to $2^m\theta$. We can derive a lower bound for this probability:

$$p_j \overset{j=2^m\theta+1/2}{\longrightarrow} \frac{\sin^2(\pi/2)}{2^{2m}\sin^2(\pi/2^{m+1})} \geq \frac{1}{2^{2m}(\pi/2^{m+1})^2} = \frac{4}{\pi^2} \approx 0.405 \ . \tag{3.38}$$

Therefore, with probability $> 40.5\%$, this circuit returns the integer $0 \leq j < 2^m$ that is the closest m-bit integer to $2^m\theta$, so that $\theta \approx j/2^m$. This circuit can be run multiple times or with a slightly higher precision before rounding down to ensure that the correct approximation for θ is obtained.

3.8 Shor's algorithm

Given an integer N, return its prime factorization,

$$N = p_1^{k_1} p_2^{k_2} \cdots p_m^{k_m} \ .$$

The quantum part of Shor's algorithm solves a related problem:

(*Order finding*) Given coprime integers a and N, find the **order** of a modulo N, i.e. the smallest integer r such that

$$a^r \equiv 1 \pmod{N} \ .$$

Let n be the number of bits needed to express the integer N, i.e. $N \leq 2^n < 2N$. We define an n-qubit operator that implements multiplication by a modulo N,

$$M_a : |x\rangle \mapsto |ax \ (\mathrm{mod}\ N)\rangle \ , \tag{3.39}$$

for $x = 0, 1, \ldots, N-1$. The mapping of the values $N \leq x < 2^n$ can be arbitrary (while keeping M_a unitary), since they are not used here. The order finding algorithm makes use of the phase approximation algorithm on this M_a operator. We note in passing that $\Lambda_m(M_a)$ can be efficiently implemented using $\mathcal{O}\big(mn^2\big)$ gates. While we will not prove this fact, it is ultimately due to the fact that any classical circuit that uses k operations (such as AND, OR, and NOT) can be implemented by a quantum circuit that uses $\mathcal{O}(k)$ gates. For instance, the multiplication of two n-bit integers x, y may be implemented by a quantum circuit that uses $\mathcal{O}\big(n^2\big)$ gates, following the usual rules of multiplication. In order to maintain unitarity, we need n ancilliary qubits to carry the output, denoted in the circuit below by w:

$$\begin{array}{lcl} |x\rangle & \multirow{3}{*}{$\boxed{\times}$} & |x\rangle \\ |y\rangle & & |y\rangle \\ |w\rangle & & |w + xy\rangle \end{array} \tag{3.40}$$

To implement the classical map $x \mapsto a^k x \pmod{N}$, we can write a^k for $k = 2^{m-1}k_{m-1} + \cdots + 2k_1 + k_0$ as the controlled multiplication of repeated squares of a:

$$a^k = (a)^{k_0}(a^2)^{k_1}(a^4)^{k_2} \cdots (a^{2^{m-1}})^{k_{m-1}} \ . \tag{3.41}$$

There are $\mathcal{O}(m)$ factors in this product and each successive squaring requires the $\mathcal{O}\big(n^2\big)$ multiplication circuit, so in total there are $\mathcal{O}\big(mn^2\big)$ gates to calculate a^k for any $0 \leq k < 2^m$. For additional reference, the gates needed to implement Shor's algorithm are explicitly constructed in https://arxiv.org/abs/quant-ph/9511018.

Now that we have $\Lambda_m(M_a)$, let us turn to the eigenvectors of M_a. We note that if r is the order of a modulo N, then we have r eigenvectors $|\Psi_\ell\rangle$ for $\ell = 0, 1, \ldots, r-1$ with eigenvalues ω_r^ℓ where $\omega_r \equiv e^{2\pi i/r}$:

$$|\Psi_\ell\rangle = \frac{1}{\sqrt{r}}\Big(|1\rangle + \omega_r^{-\ell}\,|a\rangle + \omega_r^{-2\ell}\,\big|a^2\big\rangle + \cdots + \omega_r^{\ell}\,\big|a^{r-1}\big\rangle\Big) \ . \tag{3.42}$$

It is not possible to construct these eigenvectors without knowing r beforehand. However, we can note that the easily prepared state $|1\rangle$ can be written as a sum of all these eigenvectors,

$$|1\rangle = \frac{1}{\sqrt{r}}\sum_{\ell=0}^{r-1} |\Psi_\ell\rangle \ . \tag{3.43}$$

So if we pass the state $|1\rangle$ through the phase estimation algorithm, the measurement $j/2^m$ will approximate the phase $\theta = \ell/r$ of an $|\Psi_\ell\rangle$ eigenvector chosen uniformly at random. Moreover, if we pick a high enough precision m, then we can know the exact value of the fraction ℓ/r, as a consequence of the following simple fact: the difference between any two reduced fractions x_1/y_1 and x_2/y_2 is $> 1/N^2$ if all integers x_1, x_2, y_1, y_2 are $\leq N$. Letting $m = 2n$ is sufficient. Note that a single measurement may not determine r since ℓ/r may not be a reduced fraction. But by making multiple measurements and obtaining many different reduced fractions, taking the least common multiple of all denominators will yield r with high probability.

In summary, the order finding algorithm is a quantum algorithm that can compute the order of a modulo N in $\mathcal{O}(\log^3 N)$ time.[3] This is exponentially faster than any classical computation—the brute force algorithm runs in $\mathcal{O}(N)$ time. The remainder of the prime factorization algorithm is classical and runs in the same time complexity. The general strategy is to find a square-root b of 1 modulo N that is not ± 1. The existence of such square-roots are guaranteed by the Chinese Remainder Theorem, as explained below. Since

$$b^2 - 1 = (b-1)(b+1) \equiv 0 \pmod{N} , \tag{3.44}$$

and N cannot divide $b-1$ or $b+1$, N must share non-trivial divisors with both factors. Then $d = \gcd(b-1, N)$ will compute one of those non-trivial divisors.

We start by preprocessing N: (i) eliminate all factors of 2 from N, (ii) check that N is not a prime (using a quick method such as Miller-Rabin), and (iii) check that N is not a power of a prime. Shor's algorithm then finds a non-trivial divisor for N:

- Pick a random integer $1 < a < N$.
- Compute $d = \gcd(a, N)$ using the Euclidean algorithm. If $d \neq 1$, then we are very lucky and have found a non-trivial divisor.
- Run the order finding algorithm to find the order r of a modulo N.
- If r is even and $a^{r/2} \not\equiv -1 \pmod{N}$, then $d = \gcd(a^{r/2} - 1, N)$ is a non-trivial divisor.
- Repeat the above steps until a divisor is found.

In order for this algorithm to work, we have to show that each iteration has a high probability of finding a divisor. This requires a bit of number theory, but the conclusion is that each iteration of Shor's algorithm has a $\geq 50\%$ chance of finding a non-trivial divisor. We note that after the preprocessing steps we can write $N = p_1^{k_1} p_2^{k_2} \cdots p_m^{k_m}$ where p_i are odd primes and $m \geq 2$. By the Chinese Remainder Theorem, for any $0 \leq a < N$ there exist unique integers a_i for $i = 1, 2, \ldots, m$ such that

$$a \equiv a_i \pmod{p_i^{k_i}} , \qquad 0 \leq a_i < p_i^{k_i} . \tag{3.45}$$

This implies that there are 2^{m-1} unique square-roots of 1 modulo N apart from $a = \pm 1$, since any combination of $a_i = \pm 1$ corresponds to an a that squares to 1. Note that all $a_i = 1$ corresponds to 1, and all $a_i = -1$ corresponds to $a = -1$.

If a is coprime to N, then all a_i have an order r_i modulo $p_i^{k_i}$. r is the least common multiple of these orders, i.e. $r = \operatorname{lcm}(r_1, r_2, \ldots, r_m)$. Now if r is even and $a^{r/2} \not\equiv -1 \pmod{N}$, as required in Shor's algorithm, then $b = a^{r/2}$ is the candidate square-root of 1 modulo N, and $a^{r/2} \equiv \pm 1 \pmod{p_i^{k_i}}$ for $i = 1, 2, \ldots, m$. Note that we cannot have all $a^{r/2} \equiv 1 \pmod{p_i^{k_i}}$ else $a^{r/2} \equiv 1 \pmod{N}$ which contradicts r being the order of a modulo N, and we cannot have all $a^{r/2} \equiv -1 \pmod{p_i^{k_i}}$ else $a^{r/2} \equiv -1 \pmod{N}$ which is not a square-root of 1 that we want.

Let $c(n)$ be the number of times 2 can divide into n. We claim that statements "r is even and $a^{r/2} \not\equiv -1 \pmod{N}$" and "$c(r_i) \neq c(r_j)$ for some i and j" are equivalent. For the forward direction, r even implies there exists an r_i even. Assuming for a contradiction that all $c(r_i)$ are equal, this implies that all r_i are even. So $a^{r_i/2} \equiv -1 \pmod{p_i^{k_i}}$ for all i, since r_i is the order and this is the only remaining square-root of 1 modulo $p_i^{k_i}$. But since $r = r_i \cdot (\text{odd number})$, this implies $a^{r/2} \equiv -1 \pmod{p_i^{k_i}}$ and so $a^{r/2} \equiv -1 \pmod{N}$ which is a contradiction. For the backward direction, WLOG suppose $c(r_1) < c(r_2)$. Then $r = 2r_1 \cdot (\text{integer})$ and is even. Moreover, $a^{r/2} = (a^{r_1})^{(\text{integer})} \equiv 1 \pmod{p_1^{k_1}}$ which implies $a^{r/2} \not\equiv -1 \pmod{N}$.

[3] Other sources cite a slightly faster time complexity of $\mathcal{O}\big((\log N)^2 (\log\log N)(\log\log\log N)\big)$, but I do not know how this is justified.

Thus, an iteration of Shor's algorithm fails if we pick an a whose a_i all have identical $c(r_i)$. How likely is this to happen? Since the multiplicative group of integers modulo $p_i^{k_i}$ is cyclic, there exists a primitive element u_i that generates the group.[4] The order of u_i is the size of the group, which we denote as $\varphi_i = p_i^{k_i-1}(p_i - 1)$. As we can write $a_i \equiv u^{q_i} \pmod{p_i^{k_i}}$ for some integer q_i, we can note that $r_i = \varphi_i / \gcd(q_i, \varphi_i)$. Picking a random a_i in this multiplicative group is equivalent to picking a random integer $0 \leq q_i < \varphi_i$ uniformly. So if we pick a random q_1 that gives us a $c(r_1)$, we will need to pick a q_2 such that

$$c(r_1) = c(\varphi_2) - \min(c(q_2), c(\varphi_2)) \ , \qquad \implies c(q_2) \begin{cases} \geq c(\varphi_2) & \text{if } c(r_1) = 0, \\ = c(\varphi_2) - c(r_1) & \text{if } c(r_1) > 0. \end{cases} \tag{3.46}$$

In either case, the probability of picking such a q_2 from a uniform distribution is $\leq 50\%$. In the first case, since φ_2 is necessarily even we just need to pick an even number. In the second case, we need to pick a specific power of 2. Therefore, an iteration of Shor's algorithm has a $\leq 50\%$ chance of failing to find a non-trivial root. To be specific, if N has $m \geq 2$ distinct prime factors, then the chance to fail is $\leq 2^{m-1}$.

[4] This is why we needed N to be odd. The multiplicative group of integers modulo n is cyclic iff $n = 2, 4, p^k, 2p^k$ where p is an odd prime. See https://mathworld.wolfram.com/ModuloMultiplicationGroup.html for more details.