Quantum Circuits and Logic Gates

André Chailloux

Contents

1 The formalism of quantum computing and quantum circuits 4
- 1.1 The Qubit . . . 6
  - 1.1.1 Vector spaces and definitions . . . 6
  - 1.1.2 Unitary operations . . . 8
  - 1.1.3 Measurements . . . 10
- 1.2 Two qubits . . . 12
  - 1.2.1 Definition, tensor product and entanglement . . . 12
  - 1.2.2 Measurements on 2 qubits . . . 14
- 1.3 n-qubit systems, unitaries and projective measurements . . . 15
- 1.4 Quantum circuits . . . 16
  - 1.4.1 The Solovay-Kitaev theorem and the gate model. . . . 17
  - 1.4.2 Simulating classical circuits with quantum circuits . . . 18

2 First examples of interesting quantum circuits 21
- 2.1 The Deutsch-Jozsa algorithm . . . 21
- 2.2 Simon's problem . . . 23

3 Grover's algorithm 25
- 3.1 First description . . . 25
- 3.2 More formally . . . 27
- 3.3 Amplitude amplification . . . 29
  - 3.3.1 What about randomized classical algorithms? . . . 31

4 The quantum Fourier transform 32
- 4.1 The classical Fourier transform . . . 32
  - 4.1.1 Definition . . . 32
  - 4.1.2 Computing the classical Fourier transform . . . 33
- 4.2 The quantum Fourier transform . . . 33
  - 4.2.1 Definition of the problem . . . 33
  - 4.2.2 Efficient quantum circuit for QFT . . . 34
- 4.3 Phase estimation . . . 36
- 4.4 Application: Fourier transform F_N for any $N \in \mathbb{N}$ . . . 37

5 Shor's quantum factoring algorithm **39**
5.1 From factoring to period finding 39
5.1.1 Classical algorithm for factoring a number N using period finding . 39
5.1.2 Proof that the algorithm works 40
5.2 Shor's period finding algorithm 40
5.2.1 Algorithm for period finding 40
5.2.2 Complexity of Shor's algorithm 42

Foreword

These lectures notes are intended for Masters students of Sorbonne Université attending the course Quantum Circuits and Logic Gates. They contain a mostly self-contained introduction to quantum computing with the mathematical and conceptual tools required for understanding the power of quantum computing and why it has gained so much interest in the last decades. There are no prerequisites in computer science or in quantum physics. Since we use linear algebra to formalize the model of quantum computing, familiarity with basic notions of linear algebra will be helpful.

If you have difficulties understanding some material in these lecture notes, a good thing to do is to read some other lecture notes where things will be explained in a different manner and maybe you will get a key information that wasn't here. I strongly recommend Ronald de Wolf's lecture notes[1] which cover most of the topics we will present here and are very well written. You can also check the book *Quantum Information and Quantum Computation* by Nielsen and Chuang which is still the reference textbook for quantum computing.

These lecture notes were written on the fly for the course of autumn 2021, and adjusted every year. There will probably be some typos and mistakes (hopefully not too many) in the first iterations of these lecture notes. Remarks, comments on these lecture notes are very welcome, particularly if you find some typos or mistakes. You can contact me at `andre.chailloux@inria.fr`.

[1] `https://homepages.cwi.nl/~rdewolf/qcnotes.pdf`

Chapter 1

The formalism of quantum computing and quantum circuits

Quantum mechanics is one of the most important discoveries of the last century in theoretical physics. Thanks to quantum mechanics, we know that at a very small scale, particles behave very differently than what we thought before. At this scale, particles are at several states at the same time and they are modified when observed. Even though these concepts have been developed in the late 1930's, there are still many mysteries related to this theory because of its counterintuitive nature. Still, many experiments have confirmed the quantum nature of the world.

In the mid-80's, the physicist Richard Feynman had a remarkable idea: If we can control some quantum particles, we are able to simulate physical systems in a more efficient way. From his article [Fey82], quantum computing was born. The basic idea is that instead of working on bits that take the value 0 or 1, we work on qubits that are superpositions of bits. A qubit takes the value 0 and 1 with some related coefficients.

There are two main advantages of quantum computing. By manipulating qubits in superposition, we could be able to make some computations in parallel and solve some problems much more quickly than in the classical case. In 1994, Peter Shor discovered that factoring (see Figure 1.1) can be done in polynomial time by a quantum computer [Sho94]. This means that every cryptographic application based on the hardness of factoring (including RSA) can be broken using a quantum computer. This result raised much interest in quantum computing which has now become a very wide and fruitful research topic. Another witness of quantum superiority : Grover showed that one can find an item in database of size n in time $O(\sqrt{n})$ [Gro97] using a quantum computer instead of $O(n)$ for a classical computer. However, such quantum algorithms are still very difficult to implement since it is hard to control many qubits simultaneously.

Another important feature of quantum states is that they lose their quantum behavior when observed. As long as a quantum state is not observed, it is in a superposition of states. However, when it is observed, it chooses probabilistically in which state it is. This

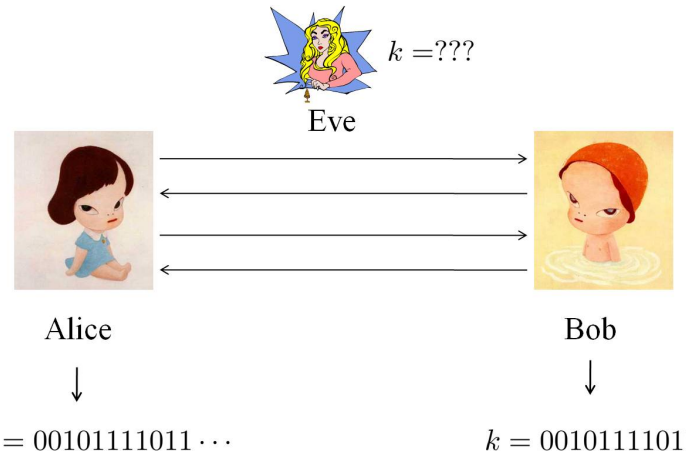
Figure 1.1: Factoring

- Input: any number $n = p \cdot q$ where p, q are prime numbers and $p, q \neq 1$
- Goal: find p and q

For example, if $n = 657713791279$, the goal is to find out that $657713791279 = 660661 \cdot 995539$. Typically, when n has 100 digits (when p, q can have each around 50 digits), the problem is hard for a classical computer but could be easily solved by a quantum computer.

means in particular that a quantum state changes when observed. In 1984, Bennett and Brassard [BB84] showed how to use this fact to perform quantumly a cryptographic task: Key Distribution (Figure 1.2). Their protocol doesn't use any computational assumption *i.e.* they don't need to assume that a computational problem is hard. Instead, the security is unconditional and relies on the laws of quantum computing. This kind of unconditional security is impossible to achieve in the classical computation model. Since then, Quantum Cryptography has also been developed in many directions. Note also that it is already possible to implement such protocols in practice. The cost and efficiency of quantum cryptography is still worse than its classical counterpart but it becomes more and more a viable solution and several companies sell such quantum devices.

Figure 1.2: Key distribution



Alice and Bob communicate quantumly. At the end, they want to share a common string k. Eve should not be able to gather information about the key k without Alice and Bob noticing.

So quantum computing is very promising. But how did we discover these results? What can we do using quantum computing. The first thing of course is to look at the laws of quantum mechanics such as Schrödinger's equation

$$i\hbar\frac{d}{dt}\left|\Psi(t)\right\rangle = \hat{H}\left|\Psi(t)\right\rangle.$$

It's not clear how to use these laws of quantum mechanics for factoring numbers for instance. Fortunately, several people, notably David Deutsch translated the laws of quantum mechanics into a model of quantum computing which can be used without any knowledge of the underlying physics [Deu85]. Actually, there are different computing models for quantum computing and all of them give us all the power of quantum computing. The most standard model of quantum computing: the Discrete Variable model. This model, which is the textbook model for most applications in quantum computing, is what we will present in these lecture notes.

1.1 The Qubit

In classical computing, bits are the basic units of information and each bit can take the value 0 or 1. In this section, we will discuss the quantum equivalent of a bit, called *qubit*.

1.1.1 Vector spaces and definitions

Definition of a qubit

We consider the vector space $\mathbb{C}^2$ over the field $\mathbb{C}$. Elements of this vector space will be represented as column vectors with 2 elements. This means any element $\vec{v}$ of $\mathbb{C}^2$ can be written $\vec{v} = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}$ with $\alpha, \beta \in \mathbb{C}$. Let also $\vec{0} = \begin{pmatrix}0\\ 0\end{pmatrix}$ be the zero vector. We consider the canonical addition and multiplication defined as follows:

$$\begin{pmatrix}\alpha_1\\ \beta_1\end{pmatrix} + \begin{pmatrix}\alpha_2\\ \beta_2\end{pmatrix} = \begin{pmatrix}\alpha_1+\alpha_2\\ \beta_1+\beta_2\end{pmatrix} \quad ; \quad \gamma\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\gamma\alpha\\ \gamma\beta\end{pmatrix} \text{ for any } \gamma \in \mathbb{C}.$$

The canonical basis of $\mathbb{C}^2$ is $(\vec{e}_0, \vec{e}_1)$ with $\vec{e}_0 = \begin{pmatrix}1\\ 0\end{pmatrix}$ and $\vec{e}_1 = \begin{pmatrix}0\\ 1\end{pmatrix}$ and we have $\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \alpha\vec{e}_0 + \beta\vec{e}_1$ for any $\alpha, \beta \in \mathbb{C}$.

On this vector space $\mathbb{C}^2$, we add the canonical inner product $\langle\cdot,\cdot\rangle$ defined as follows:

$$\text{For } \vec{v}_1 = \begin{pmatrix}\alpha_1\\ \beta_1\end{pmatrix} \text{ and } \vec{v}_2 = \begin{pmatrix}\alpha_2\\ \beta_2\end{pmatrix}, \text{ we define } \langle\vec{v}_1, \vec{v}_2\rangle = \alpha_1{\alpha_2}^* + \beta_1{\beta_2}^*.$$

where z^* is the complex conjugate of z for $z \in \mathbb{C}$. The vector space $\mathbb{C}^2$ with this inner product is a *Hilbert space*. This inner product has the following properties:

1. The inner product is conjugate symmetric: $\langle\vec{v}_1, \vec{v}_2\rangle = (\langle\vec{v_2}, \vec{v_1}\rangle)^*$ for any $\vec{v}_1, \vec{v}_2 \in \mathbb{C}^2$.
2. The inner product is linear in its first argument: $\langle\gamma_1\vec{v}_1 + \gamma_2\vec{v}_2, \vec{w}\rangle = \gamma_1\langle\vec{v}_1, \vec{w}\rangle + \gamma_2\langle\vec{v}_2, \vec{w}\rangle$ for any $\vec{v}_1, \vec{v}_2, \vec{w} \in \mathbb{C}^2$ and $\gamma_1, \gamma_2 \in \mathbb{C}$.
3. The inner product of an element with itself is positive definite: $\langle\vec{0}, \vec{0}\rangle = 0$ and $\langle\vec{v}, \vec{v}\rangle > 0$ for any $\vec{v} \in \mathbb{C}^2$ different than $\vec{0}$.

This inner product induces the norm $\|\vec{v}\| = \sqrt{\langle v, v\rangle}$. For $\vec{v} = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}$, we have $\|\vec{v}\| = \sqrt{\alpha\alpha^* + \beta\beta^*} = \sqrt{|\alpha|^2 + |\beta|^2}$ which is the *Euclidian norm.* This is indeed a norm since it satisfies the following properties:

1. Positive definiteness and nonnegativity: $\|\vec{v}\| \geq 0$ for any $\vec{v} \in \mathbb{C}^2$ and $\|\vec{v}\| = 0 \Leftrightarrow \vec{v} = \vec{0}$.
2. Absolute homogeneity: $\|\gamma\vec{v}\| = |\gamma| \, \|\vec{v}\|$ for any $\vec{v} \in \mathbb{C}^2$ and $\gamma \in \mathbb{C}$.
3. Triangle inequality: $\|\vec{v}_1 + \vec{v}_2\| \leq \|\vec{v}_1\| + \|\vec{v}_2\|$ for any $\vec{v}_1, \vec{v}_2 \in \mathbb{C}^2$.

Two vectors $\vec{v}, \vec{w}$ are said to be orthogonal if their inner product is 0, and we write $\vec{v} \perp \vec{w}$ in this case. We have now all the tools to define a quantum bit

Definition 1.1. *The set of quantum bits, usually called qubits, is the set of vectors $\vec{v}$ of $\mathbb{C}^2$ of norm 1. Each qubit can be written as $\begin{pmatrix}\alpha\\ \beta\end{pmatrix} \in \mathbb{C}^2$ with $|\alpha|^2 + |\beta|^2 = 1$ and reciprocally, any element $\begin{pmatrix}\alpha\\ \beta\end{pmatrix} \in \mathbb{C}^2$ satisfying $|\alpha|^2 + |\beta|^2 = 1$ is a qubit.*

The Dirac notation

A qubit will be represented by a *'ket'* which is the following symbol $|\cdot\rangle$. This notation was introduced by Dirac in order to represent quantum states. We define $|0\rangle = \begin{pmatrix}1\\ 0\end{pmatrix}$ and $|1\rangle = \begin{pmatrix}0\\ 1\end{pmatrix}$ so we write the canonical basis as $\{|0\rangle, |1\rangle\}$. We can write for example

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \text{ with } |\alpha|^2 + |\beta|^2 = 1$$

meaning that the qubit $|\psi\rangle$ is the vector $\begin{pmatrix}\alpha\\ \beta\end{pmatrix}$. We say that $|\psi\rangle$ is in *superposition* of $|0\rangle$ and $|1\rangle$ and α, β are the *amplitudes* of $|\psi\rangle$.

The 'bra' notation: for a vector "ket psi" $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}$, we define "bra psi" as follows

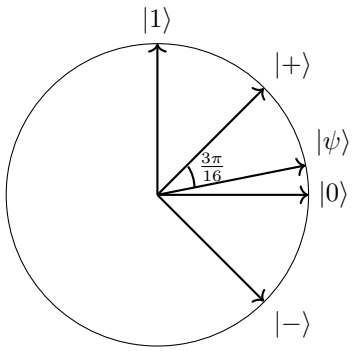
$$\langle\psi| = (\alpha^* \ \beta^*)$$

$\langle\psi|$ is a line vector of $\mathbb{C}^2$. In particular $\langle 0| = (1\ 0)$ and $\langle 1| = (0\ 1)$. This notation is useful for example as it allows us to interpret the inner product $\langle\vec{v}_1|\vec{v}_2\rangle$ as $\langle\vec{v}_1| \cdot |\vec{v}_2\rangle$, where the symbol $\cdot$ is a multiplication between a line vector and a column vector.

Example of qubits and different basis

Definition 1.2. *A basis of $\mathbb{C}^2$ is a pair of vectors $\{\vec{v_0}, \vec{v_1}\}$ such that $\alpha\vec{v_0} + \beta\vec{v_1} = \vec{0}$ iff. $\alpha = \beta = 0$. Such a basis is said to be orthogonal it additionally satisfies $\vec{v_0} \perp \vec{v_1}$. It is said to be orthonormal if it is orthogonal and $\|\vec{v_0}\| = \|\vec{v_1}\| = 1$. This means an orthonormal basis is a pair of orthogonal qubits.*

Figure 1.3: Graphical representation of a qubit. Each point of the circle is a qubit with real amplitudes. Conversely, any qubit with real amplitudes can be represented by a point on the circle.



Proposition 1.3. *For any quantum state $|\psi\rangle$ and for any orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$ there exists $\alpha, \beta \in \mathbb{C}$ such that*

$$|\psi\rangle = \alpha |e_0\rangle + \beta |e_1\rangle \qquad \text{and} \qquad |\alpha|^2 + |\beta|^2 = 1.$$

We say that we write or decompose $|\psi\rangle$ in the basis $\{|e_0\rangle, |e_1\rangle\}$. We also have $\alpha = \langle e_0|\psi\rangle$ and $\beta = \langle e_1|\psi\rangle$. Also, since $|\alpha|^2 + |\beta|^2 = 1$, there exists $\gamma \in [0, \pi/2]$ and $\theta \in [0, 2\pi]$ such that $|\psi\rangle = \cos(\gamma) |e_0\rangle + e^{i\theta} \sin(\gamma) |e_1\rangle$.

Definition 1.4. *The basis $\{|0\rangle, |1\rangle\}$ is called the computational basis ot the standard basis. The basis $\{|+\rangle, |-\rangle\}$ is called the Hadamard basis where*

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

This notation $|+\rangle, |-\rangle$ will be extensively used in these lectures notes.

As an example of decomposition, we can write for example $|0\rangle = \frac{1}{\sqrt{2}} (|+\rangle + |-\rangle)$ and $|1\rangle = \frac{1}{\sqrt{2}} (|+\rangle - |-\rangle)$. Here is also an example of a qubit $|\psi\rangle$ decomposed in the computational basis and in the Hadamard basis.

$$|\psi\rangle = \cos(\pi/16) |0\rangle + \sin(\pi/16) |1\rangle = \cos(3\pi/16) |+\rangle + \sin(3\pi/16) |-\rangle.$$

In the case the amplitudes are real, we have the following graphical representation of a qubit (Figure **??**). We saw the definition of a qubit. We will now show how to manipulate these qubits. There are 2 types of operations that we can perform on a qubit: unitary operations and measurements.

1.1.2 Unitary operations

Definition 1.5. *For a matrix $U = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$, let $U^\dagger = (U^*)^\intercal$ be the conjugate transpose of U i.e. $U^\dagger = \begin{pmatrix} a^* & b^* \\ c^* & d^* \end{pmatrix}$.*

Definition 1.6. *A unitary operation, (also called quantum unitary, unitary matrix or just unitary) is a matrix* $U = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$ *such that* $U^\dagger U = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. *This also implies that* $UU^\dagger = I$ *hence* $U^\dagger$ *is the inverse of* U.

Unitaries on qubits have the following property

Proposition 1.7. *A* 2×2 *matrix* U *is unitary iff. it satisfies the following properties.*

1. $|a|^2 + |b|^2 = |c|^2 + |d|^2 = 1$.
2. $\begin{pmatrix} a \\ b \end{pmatrix} \perp \begin{pmatrix} c \\ d \end{pmatrix}$ *meaning that* $ac^* + bd^* = a^*c + b^*d = 0$.

Proof. For any matrix $U = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$, we write

$$U^\dagger U = \begin{pmatrix} a^* & b^* \\ c^* & d^* \end{pmatrix} \cdot \begin{pmatrix} a & c \\ b & d \end{pmatrix} = \begin{pmatrix} |a|^2 + |b|^2 & a^*c + b^*d \\ ac^* + bd^* & |c|^2 + |d|^2 \end{pmatrix}.$$

One can immediately see that $U^\dagger U$ is the identity matrix iff. the above 2 properties are satisfied, which completes the proof. □

We can now state our first rule of quantum computing on qubits.

Rule 1.8 (Unitary operations on single qubits)**.** *It is possible to apply any unitary* $U = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$ *to a qubit* $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. *The output is the qubit* $U \cdot |\psi\rangle$ *where we perform a standard matrix/vector multiplication. If we apply* U *to* $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, *we therefore have*

$$U \cdot |\psi\rangle = (\alpha a + \beta c)|0\rangle + (\alpha b + \beta d)|1\rangle.$$

We will often omit the $\cdot$ and just write $U|\psi\rangle$ or sometimes $U(|\psi\rangle)$. One can check that the output is indeed a qubit *i.e.* that it has norm 1. Indeed, we have

$$\begin{aligned} \|U|\psi\rangle\|^2 &= (\alpha a + \beta c)(\alpha a + \beta c)^* + (\alpha b + \beta d)(\alpha b + \beta d)^* \\ &= |\alpha|^2(|a|^2 + |b|^2) + |\beta|^2(|c|^2 + |d|^2) + \alpha\beta^*(ac^* + bd^*) + \alpha^*\beta(a^*c + b^*d) \\ &= |\alpha|^2 + |\beta|^2 = 1 \end{aligned}$$

By definition, a unitary U is a linear operation meaning that for $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, we have $U|\psi\rangle = \alpha U|0\rangle + \beta U|1\rangle$.

Example of unitary operations.

Consider a state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. Here are some examples of unitaries.

- Bit flip $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$: $X|0\rangle = |1\rangle; X|1\rangle = |0\rangle \Rightarrow X(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle$.

- Phase flip $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$: $Z|0\rangle = |0\rangle\,; Z|1\rangle = -|1\rangle$.
- θ-Phase flip $Z_\theta = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$: $Z_\theta|0\rangle = |0\rangle\,; Z_\theta|1\rangle = e^{i\theta}|1\rangle$.
- Hadamard $H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$: $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$.

Proposition 1.9. *For any orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$, there exists a unitary U such that $U|0\rangle = |e_0\rangle$ and $U|1\rangle = |e_1\rangle$. This implies that $U^\dagger|e_0\rangle = |0\rangle$ and $U^\dagger|e_1\rangle = |1\rangle$. This unitary can be written*

$$U = |e_0\rangle\langle 0| + |e_1\rangle\langle 1| \quad ; \quad U^\dagger = |0\rangle\langle e_0| + |1\rangle\langle e_1|.$$

If we write $|e_0\rangle = \alpha_0|0\rangle + \beta_0|1\rangle$ and $|e_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$. The unitary U is the matrix $\begin{pmatrix} \alpha_0 & \alpha_1 \\ \beta_0 & \beta_1 \end{pmatrix}$.

Proposition 1.10. *For any two orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$ and $\{|f_0\rangle, |f_1\rangle\}$, there exists a unitary U such that $U|e_0\rangle = |f_0\rangle$ and $U|e_1\rangle = |f_1\rangle$.*

Proposition 1.11. *Let U and V be two unitaries. Then UV and VU are also unitaries. Also, $(UV)^\dagger = V^\dagger U^\dagger$.*

1.1.3 Measurements

Measurements are the second kind of admissible quantum operations on a qubit.

Rule 1.12. *[Measurements on single qubits] We are allowed to perform measurements on qubits. A measurement on a qubit $|\psi\rangle$ is the following operation*

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{Measurement}} \begin{cases} \textit{outcome } 0 & \textit{wp. } |\alpha|^2. & |\psi\rangle \textit{ collapses to } |0\rangle \\ \textit{outcome } 1 & \textit{wp. } |\beta|^2. & |\psi\rangle \textit{ collapses to } |1\rangle \end{cases}$$

So a measurement on a state $|\psi\rangle$ outputs a bit b (the outcome) probabilistically depending on the amplitudes of $|\psi\rangle$. The state then collapses to a computational basis state that depends on the outcome of the measurement.

A quantum measurement is the only way to extract some information about a qubit. Typically, if we are given $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, we have no way of determining the amplitudes α and β. The only thing we can do is to apply some unitary operations and in the end perform a measurement, which will give us only 1 bit of information about $|\psi\rangle$ while a qubit is characterized by 2 complex numbers! Notice that since the state $|\psi\rangle$ collapses to a computational basis state, we cannot perform another measurement to get some more information about $|\psi\rangle$. This is a serious limitation of quantum computing. On the one hand, we can manipulate qubits which are in a superposition of $|0\rangle$ and $|1\rangle$ but we can only extract a small amount of information from these qubits.

There are actually different formulations of what happens to the qubit $|\psi\rangle$ after the measurement. In some formulations, the qubit is destroyed, in others there is a collapse as we chose for the measurement rule. The two formulations are equivalent since if the state $|\psi\rangle$ is destroyed, we can always recreate the state $|b\rangle$ from the output b of the measurement. Our formulation will be however more consistent with the mathematical formulation of measurements that we will use later. This measurement we presented is usually called measurement in the computational basis.

Measurement in another basis

We can generalize the above measurement to other basis.

Definition 1.13. *For any orthonormal basis $\mathscr{B} = \{|e_0\rangle, |e_1\rangle\}$, we can measure any state $|\psi\rangle$ in the basis $\mathscr{B}$ which corresponds to the following:*

$$|\psi\rangle = \alpha|e_0\rangle + \beta|e_1\rangle \xrightarrow{\textit{Measurement in basis } \mathscr{B}} \begin{cases} \textit{outcome } 0 \textit{ wp. } |\alpha|^2 = |\langle e_0|\psi\rangle|^2. & |\psi\rangle \textit{ collapses to } |e_0\rangle \\ \textit{outcome } 1 \textit{ wp. } |\beta|^2 = |\langle e_1|\psi\rangle|^2. & |\psi\rangle \textit{ collapses to } |e_1\rangle \end{cases}$$

This definition generalizes measurements in the computational basis. We show that we can perform such measurements for any basis only using Rule 1.8 and Rule 1.12.

Proposition 1.14. *One can perform a measurement in any basis $\{|e_0\rangle, |e_1\rangle\}$ by performing unitary operations and a measurement in the computational basis.*

Proof. Fix a basis $\mathscr{B} = \{|e_0\rangle, |e_1\rangle\}$ and let U be the unitary st. $U|e_0\rangle = |0\rangle$ and $U|e_1\rangle = |1\rangle$. Such a unitary U exists from Proposition 1.10. Consider any state $|\psi\rangle = \alpha|e_0\rangle + \beta|e_1\rangle$. In order to measure $|\psi\rangle$ in basis $\mathscr{B}$, we can equivalently first apply U, perform a measurement in the computational basis and then apply $U^\dagger$. Indeed

$$|\psi\rangle \xrightarrow{\text{Measurement in basis } \mathscr{B}} \begin{cases} \text{outcome 0 wp. } |\alpha|^2. & |\psi\rangle \text{ collapses to } |e_0\rangle \\ \text{outcome 1 wp. } |\beta|^2. & |\psi\rangle \text{ collapses to } |e_1\rangle \end{cases}$$

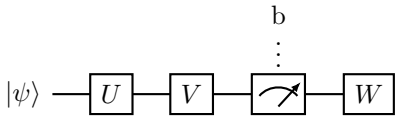
and

$$|\psi\rangle \xrightarrow{U} \alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{Measurement in basis } \{|0\rangle,|1\rangle\}} \begin{cases} \text{outcome 0 wp. } |\alpha|^2. & |\psi\rangle \text{ collapses to } |0\rangle \\ \text{outcome 1 wp. } |\beta|^2. & |\psi\rangle \text{ collapses to } |1\rangle \end{cases}$$
$$\xrightarrow{U^\dagger} \begin{cases} \text{outcome 0 wp. } |\alpha|^2. & |\psi\rangle \text{ collapses to } |e_0\rangle \\ \text{outcome 1 wp. } |\beta|^2. & |\psi\rangle \text{ collapses to } |e_1\rangle \end{cases}$$

□

Circuit representation

The 2 rules are the 2 only admissible operations that we can perform on single qubits. We can alternatively apply any of these operations. For example, if we start from a qubit $|\psi\rangle$, the circuit depicted below corresponds to: starting from $|\psi\rangle$, applying U, applying V, measuring in the computational basis and getting outcome b and then performing W on the resulting state.



While there are applications, such as Quantum Key Distribution, that require only single qubits and operations, the theory is much more interesting by manipulating several qubits at the same time. We now study the case of 2 qubits, which will already exhibit a key aspect of quantum computing and quantum mechanics: entanglement.

1.2 Two qubits

1.2.1 Definition, tensor product and entanglement

A qubit represents a quantum system that can be in a superposition of 2 different physical states called 0 and 1. Two qubits represent two of these systems, or equivalently a quantum system that can be in a superposition of 4 states $00, 01, 10$ and 11.

Definition 1.15. *A 2-qubit state is an element of the vector space $\mathbb{C}^4$ of norm 1. It can therefore be written*

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \quad \textit{with} \quad |\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1.$$

with the convention:

$$|00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \;;\; |01\rangle = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \;;\; |10\rangle = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \;;\; |11\rangle = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}.$$

Definition 1.16. *A unitary on 2 qubits is a 4×4 complex matrix U satisfying $UU^\dagger = U^\dagger U = I$ where $U^\dagger$ is the conjugate transpose of U.*

Similarly as for the 1-qubit case, for any orthonormal basis $\{|e_0\rangle, |e_1\rangle, |e_2\rangle, |e_3\rangle\}$, the matrix $U = \sum_{i=0}^{3} |e_i\rangle\langle i|$ is a unitary. More precisely, it is the unitary such that $U|i\rangle = |e_i\rangle$, and can also be written

$$U = \begin{pmatrix} \begin{pmatrix} e_0 \end{pmatrix} \begin{pmatrix} e_1 \end{pmatrix} \begin{pmatrix} e_2 \end{pmatrix} \begin{pmatrix} e_3 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} \alpha_{00}^0 & \alpha_{00}^1 & \alpha_{00}^2 & \alpha_{00}^3 \\ \alpha_{01}^0 & \alpha_{01}^1 & \alpha_{01}^2 & \alpha_{01}^3 \\ \alpha_{10}^0 & \alpha_{10}^1 & \alpha_{10}^2 & \alpha_{10}^3 \\ \alpha_{11}^0 & \alpha_{11}^1 & \alpha_{11}^2 & \alpha_{11}^3 \end{pmatrix}$$

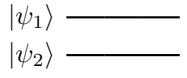
where we define α_j^i as follows: for each $i \in 0, 1, 2, 3$, $|e_i\rangle = \sum_{j \in \{0,1\}^2} \alpha_j^i |j\rangle$. Reciprocally, any unitary U is written can be written like this which implies that for any unitary U, each column of U has norm 1 and the columns are pairwise orthogonal.

We now define the tensor product of 2 qubits and of 2 unitaries.

Definition 1.17. *Consider a qubit $\alpha|0\rangle + \beta|1\rangle$ and another qubit $\alpha'|0\rangle + \beta'|1\rangle$. The joint state of these 2 qubits is described using the tensor product $\otimes$, and is*

$$(\alpha|0\rangle + \beta|1\rangle) \otimes (\alpha'|0\rangle + \beta'|1\rangle) = \alpha\alpha'|00\rangle + \alpha\beta'|01\rangle + \beta\alpha'|10\rangle + \beta\beta'|11\rangle.$$

The tensor product between 2 qubits really corresponds to the concatenation of the physical systems. This is why we often drop the $\otimes$ symbol and just write $|x\rangle|y\rangle, |x,y\rangle$ or even $|xy\rangle$ instead of $|x\rangle \otimes |y\rangle$. In particular, the 2 qubit computational basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ can be seen as the tensor products $|0\rangle\otimes|0\rangle, |0\rangle\otimes|1\rangle, |1\rangle\otimes|0\rangle, |1\rangle\otimes|1\rangle$. In terms of circuit, we represent $|\psi_1\rangle \otimes |\psi_2\rangle$ as:

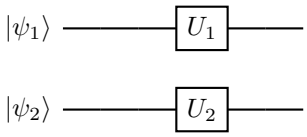


We now define the tensor product for unitaries.

Definition 1.18. *Consider* 2 *unitaries* U_1 *and* U_2 *each acting on* 1 *qubit.* $U_1 \otimes U_2$ *is a unitary acting on* 2 *qubits st. for any qubits* $|\psi_1\rangle, |\psi_2\rangle$*, we have*

$$(U_1 \otimes U_2)(|\psi_1\rangle \otimes |\psi_2\rangle) = U_1|\psi_1\rangle \otimes U_2|\psi_2\rangle.$$

So the unitary $(U_1\otimes U_2)$ acting on 2 qubits corresponds to applying U_1 to the first qubit and U_2 to the second qubit. In the circuit representation, we write $(U_1\otimes U_2)(|\psi_1\rangle\otimes|\psi_2\rangle) = U_1|\psi_1\rangle \otimes U_2|\psi_2\rangle$ as follows:



An important property of 2-qubit states is the notion of entanglement, which is motivated by the following claim.

Proposition 1.19. *Not every* 2 *qubit state* $|\phi\rangle$ *is of the form* $|v\rangle \otimes |w\rangle$ *for some qubits* $|v\rangle, |w\rangle$.

Proof. Consider the state $|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. $|\phi\rangle$ is indeed a valid 2-qubit state as it is an element of $\mathbb{C}^4$ of norm 1. Assume by contradiction that $|\phi\rangle = |v\rangle \otimes |w\rangle$ for some qubits $|v\rangle, |w\rangle$. We write $|v\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|w\rangle = \alpha'|0\rangle + \beta'|1\rangle$. This gives

$$|v\rangle \otimes |w\rangle = \alpha\alpha'|00\rangle + \alpha\beta'|01\rangle + \beta\alpha'|10\rangle + \beta\beta'|11\rangle.$$

Since $|\phi\rangle = |v\rangle \otimes |w\rangle$, we necessarily have (1) $\alpha\alpha' = \beta\beta' = \frac{1}{\sqrt{2}}$ and (2) $\alpha\beta' = \beta\alpha' = 0$. The second condition implies $\alpha = 0$ or $\beta' = 0$ which implies $\alpha\alpha' = 0$ or $\beta\beta' = 0$ which contradicts the first condition. □

The existence of entangled states is very important in quantum computing (and in quantum physics in general). For a 2-qubit entangled state, this means we cannot say in which state each qubit is, rather we have to consider the 2-qubits as a whole[1].

Definition 1.20. *A* 2*-qubit state* $|\psi\rangle$ *is can be written as* $|v\rangle\otimes|w\rangle$ *for some qubits* $|v\rangle, |w\rangle$ *is called a product state. A* 2*-qubit state which is not a product state is an* entangled state.

[1]Actually, we will still be able to say something about each individual qubit when we learn about density matrices and mixed states but still, the information about any 2-qubit entangled state will be more than the information about each individual qubit.

Definition 1.21. *A 2 qubit state $|\psi\rangle$ is called* maximally entangled *iff. there exists an orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$, $\theta \in [0, 2\pi]$ and $\gamma \in \mathbb{C}$ with $|\gamma| = 1$ st. $|\psi\rangle = \gamma\left(\frac{1}{\sqrt{2}}|0\rangle|e_0\rangle + e^{i\theta}|1\rangle|e_1\rangle\right)$.*

There are 4 useful 2-qubit maximally entangled states, which are called the Bell states

$$\begin{aligned}
\left|\Phi^+\right\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\
\left|\Phi^-\right\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \\
\left|\Psi^+\right\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \\
\left|\Psi^-\right\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)
\end{aligned}$$

1.2.2 Measurements on 2 qubits

Let $\mathscr{B} = \{|e_1\rangle, |e_2\rangle, |e_3\rangle, |e_4\rangle\}$ be an orthonormal basis of the vector space $\mathbb{C}^4$. We can perform a measurement w.r.t. this basis. For any state $|\psi\rangle = \alpha_1|e_1\rangle + \alpha_2|e_2\rangle + \alpha_3|e_3\rangle + \alpha_4|e_4\rangle$, we have

$$|\psi\rangle \xrightarrow{\text{Measurement in basis } \mathscr{B}} \begin{cases} \text{outcome 1 wp. } |\langle e_1|\psi\rangle|^2. & |\psi\rangle \text{ collapses to } |e_1\rangle \\ \text{outcome 2 wp. } |\langle e_2|\psi\rangle|^2. & |\psi\rangle \text{ collapses to } |e_2\rangle \\ \text{outcome 3 wp. } |\langle e_3|\psi\rangle|^2. & |\psi\rangle \text{ collapses to } |e_3\rangle \\ \text{outcome 4 wp. } |\langle e_4|\psi\rangle|^2. & |\psi\rangle \text{ collapses to } |e_4\rangle \end{cases}$$

Similarly as for the 1 qubit case, we can do measurements in any basis by performing unitaries and a measurement in the computational basis.

Proposition 1.22. *For any 2 qubit state $|\psi\rangle$. For any orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$ of the vector space $\mathbb{C}^2$. There exists $\alpha, \beta \in \mathbb{C}$ as well as qubits $|f_0\rangle, |f_1\rangle$ (not necessarily orthogonal) such that*

$$|\psi\rangle = \alpha|e_0\rangle|f_0\rangle + \beta|e_1\rangle|f_1\rangle.$$

Proof. You will prove this proposition in the exercise session. □

Similarly as above, we also have the following

Proposition 1.23. *For any 2 qubit state $|\psi\rangle$. For any orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$ of the vector space $\mathbb{C}^2$. There exists $\alpha, \beta \in \mathbb{C}$ as well as qubits $|f_0\rangle, |f_1\rangle$ (not necessarily orthogonal) such that*

$$|\psi\rangle = \alpha|f_0\rangle|e_0\rangle + \beta|f_1\rangle|e_1\rangle.$$

Partial measurements

When we have a 2-qubit state, it is possible to measure only 1 qubit. What is the resulting state? Consider any 2-qubit state $|\psi\rangle$ and an orthonormal basis $\mathscr{B} = \{|e_0\rangle, |e_1\rangle\}$ of $\mathbb{C}^2$. Using the previous propositions, we can write

$$|\psi\rangle = \alpha|e_0\rangle|f_0\rangle + \beta|e_1\rangle|f_1\rangle = \alpha'|f_0'\rangle|e_0\rangle + \beta'|f_1'\rangle|e_1\rangle$$

for some $\alpha, \beta, \alpha'\beta' \in \mathbb{C}$ and quantum states $|f_0\rangle, |f_1\rangle, |f'_0\rangle, |f'_1\rangle$. We can perform a partial measurements on $|\psi\rangle$ as follows:

$$|\psi\rangle \xrightarrow{\text{Measure } 1^{\text{st}} \text{ qubit in basis } \mathscr{B}} \begin{cases} \text{outcome } 0 & \text{wp. } |\alpha|^2 & \text{collapses to } |e_0\rangle |f_0\rangle \\ \text{outcome } 1 & \text{wp. } |\beta|^2 & \text{collapses to } |e_1\rangle |f_1\rangle \end{cases}$$

$$|\psi\rangle \xrightarrow{\text{Measure } 2^{\text{nd}} \text{ qubit in basis } \mathscr{B}} \begin{cases} \text{outcome } 0 & \text{wp. } |\alpha'|^2 & \text{collapses to } |f'_0\rangle |e_0\rangle \\ \text{outcome } 1 & \text{wp. } |\beta'|^2 & \text{collapses to } |f'_1\rangle |e_1\rangle \end{cases}$$

1.3 n-qubit systems, unitaries and projective measurements

We consider the vector space $\mathbb{C}^d$ with the canonical basis $\{|0\rangle, \ldots, |d-1\rangle\}$ with

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{pmatrix}, \ |1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \quad \ldots \quad |j-1\rangle = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

⚠ $|0\rangle$ usually refers to a single qubit but it can also represent a d-dimensional qudit as it is the case here. It is usually clear from context in which case we are.

For two d-dimensional vectors of $\mathbb{C}^d$, $\vec{v} = \sum_{j=0}^{d-1} \alpha_j |j\rangle$ and $\vec{v'} = \sum_{j=0}^{d-1} \alpha'_j |j\rangle$, we have the canonical inner product

$$\langle \vec{v}, \vec{v'} \rangle = \sum_{i=0}^{d-1} \alpha_i^* \alpha'_i$$

and the associated norm

$$\|\vec{v}\| = \sqrt{\langle \vec{v}, \vec{v} \rangle} = \sqrt{\sum_{i=0}^{d-1} |\alpha_i|^2}.$$

Definition 1.24. *A d-dimensional qudit $|\psi\rangle$ is a vector $|\psi\rangle = \sum_{j=0}^{d-1} \alpha_j |j\rangle$ with $\sum_{j=0}^{d-1} |\alpha_j|^2 = 1$. We also define $\langle\psi| = (|\psi\rangle^*)^\intercal = \begin{pmatrix} \alpha_0^* & \alpha_1^* & \ldots & \alpha_{d-1}^* \end{pmatrix}$. An n-qubit state is a d-dimensional qudit with $d = 2^n$. A qutrit is a 3-dimensional qudit.*

The computational basis of $\mathbb{C}^{2^n}$ is sometimes written $\{|0\rangle, \ldots, |2^n - 1\rangle\}$ and sometimes $\{|0^n\rangle, \ldots, |1^n\rangle\} = \{|s\rangle\}_{s \in \{0,1\}^n}$, where each basis element is described using an n-bit string. In the latter, we have $|s_1 s_2 \ldots, s_n\rangle = |s_1\rangle \otimes |s_2\rangle \otimes \cdots \otimes |s_n\rangle = \bigotimes_{i=1}^n |s_i\rangle$ for all $s_1, \ldots, s_n \in \{0, 1\}$.

Definition 1.25. *A d-dimensional unitary is a complex $d \times d$ matrix U such that $UU^\dagger = U^\dagger U = I$.*

Definition 1.26. *A k-outcome projective measurement acting on d-dimensional qudits is an ensemble of projectors $M = \{P_1, \ldots, P_k\}$ where $\sum_{i=1}^k P_i = I$ and the projectors are pairwise orthogonal[2]. When measuring a state $|\psi\rangle$ using measurement M, we get outcome i wp. $\|P_i |\psi\rangle\|^2$ and the state collapses to $\frac{P_i |\psi\rangle}{\|P_i |\psi\rangle\|}$.*

[2]This means $P_i P_j = P_j P_i = \delta_{ij} P_i$ where δ_{ij} is the Kronecker delta symbol: $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

Rule 1.27 (General rule of quantum computing)**.** *There are* 2 *admissible operations on d-dimensional qudits: d-dimensional unitaries as well as projective measurement on a d-dimensional space.*

On projectors and projective measurements

Projectors have the following property

Proposition 1.28. *For any projector P acting on d-dimensional qudits. There exist l pairwise orthogonal d-dimensional qudits $|e_1\rangle, \ldots, |e_l\rangle$ such that $P = \sum_{i=1}^{l} |e_i\rangle\langle e_i|$. P satisfies $P|e_i\rangle = |e_i\rangle$ for $i \in [l]$ and for any $|v\rangle$ orthogonal to all the $|e_i\rangle$, $P|v\rangle = \vec{0}$. l is the dimension of the projector.*

Proposition 1.29. *In dimension d, A projector P of rank $r \leq d$ is a matrix of the form $\sum_{i\in[r]} |e_i\rangle\langle e_i|$ for some pairwise orthogonal quantum states $\{|e_i\rangle\}_{i\in[r]}$. Consider an orthonormal basis $\mathscr{B} = \{|e_1\rangle, \ldots, |e_r\rangle, |f_1\rangle, \ldots, |f_{d-r}\rangle\}$ of the vector space $\mathbb{C}^d$. For any state $|\psi\rangle = \sum_{i=1}^{r} \alpha_i |e_i\rangle + \sum_{j=1}^{n-d} \beta_j |f_j\rangle$, we have*

$$P|\psi\rangle = \sum_{i=1}^{r} \alpha_i |e_i\rangle \text{ and } \|P|\psi\rangle\| = \sqrt{\sum_{i=1}^{r} |\alpha_i|^2} \leq 1.$$

Projective measurements generalize measurements in a given basis. Indeed, measuring in a basis $\mathscr{B} = \{|e_1\rangle, \ldots, |e_d\rangle\}$ is equivalent to performing the projective measurement $M = \{P_1, \ldots, P_d\}$ where for each $i \in [d], P_i = |e_i\rangle\langle e_i|$ is the 1-dimensional projector acting on d-dimensional qudits. Again we can construct any projective measurement with unitary operations and a measurement in the computational basis. Can you see how to do this?

Quantum registers

A k-qubit quantum register is simply the set of k-qubit quantum states, and correspond to a specific physical system described by k qubits. The idea of using registers is the following: if we have a state $|\psi\rangle$ on $n+m$ qubits and we will often apply operations on the first n qubits or the last m qubits, we say that $|\psi\rangle$ is in some registers A and B where A is an n-qubit register and B an m-qubit register, and we write $|\psi\rangle_{AB}$. Then we can say we apply a unitary or a measurement on register A for example instead of always saying on the n first qubits.

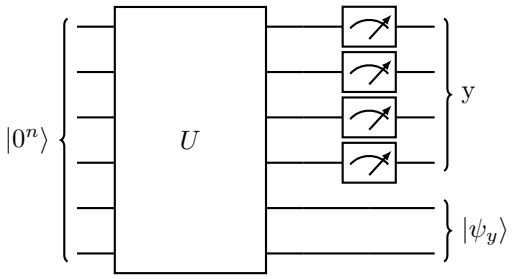
1.4 Quantum circuits

A natural formulation of a quantum algorithm is to start from n qubits initialized at $|0^n\rangle$ (or a d-dimensional qudit), and then successively apply the 2 admissible operations: unitary operations and measurements.

Applying 2 unitaries U_1 and U_2 one after the other is equivalent to applying the unitary (U_2U_1) so we can assume the algorithm performs a unitary, then perform a measurement, then applies a unitary, performs another measurement and so on.

Here, we will consider only algorithms where we first perform all the unitary operations, which corresponds to a single big unitary, and then perform measurements in the computational basis. These quantum algorithms are actually as powerful as general algorithm, and we defer the proof of this statement when we will be more used with the model of quantum circuits.

So quantum algorithms we consider are of the following form:



Auxiliary qubits

Instead of constructing $U : |\psi\rangle \to U|\psi\rangle$, it is often easier to construct the unitary $U' : |\psi\rangle |0\rangle_{Aux} \to U(|\psi\rangle)|0\rangle_{Aux}$. These extra qubits used are called *auxiliary qubits* or *ancilliary qubits*. Having these extra qubits is totally admissible and counts as an implementation of U. What is important however is that they start at $|0\rangle$ and end at $|0\rangle$ (even though they can be in other states during the computation)

1.4.1 The Solovay-Kitaev theorem and the gate model.

We said that any unitary on n qubits is an admissible quantum operation, meaning that it is in theory possible to implement but we didn't say how hard it is to perform such a unitary. A unitary U on n qubits is represented by a $2^n \times 2^n$ matrix so if we take $n = 100$, we need 2^{200} complex numbers to specify U, which is the number of particles in the universe.

In the classical setting, we decompose large function on n bits into elementary gates. The most common set of gates is the set $\mathcal{G} = \{\text{NOT}, \text{AND}, \text{OR}\}$ which is universal meaning that any classical function can be computed using only elements of this gate set, and the running time of the algorithm is the number of gates used. We will use a similar model in the quantum setting.

In the quantum setting, we need a set of elementary gates that will allow us to construct all unitaries. It's not possible to achieve this perfectly but there are set of elementary gates that can perform this approximately. This is the Solovay-Kitaev theorem

Theorem 1.30 (Solovay-Kitaev Theorem)**.** *Take the family of gates* $\mathcal{G} = \{\text{CNOT}, H, Z_{\pi/4}\}$. *Any unitary* U *of dimension* d *can be approximated by applying* $O(d^2 \log^4(\frac{1}{\varepsilon}))$ *gates from* $\mathcal{G}$ *with accuracy* ε. *In other words, from the description of* U, *one can construct a sequence* $G_1, \ldots, G_N \in \mathcal{G}$ *with* $N = O(d^2 \log^4(\frac{1}{\varepsilon}))$ *and*

$$\|G_N \ldots G_1 - U\| \leq \varepsilon,$$

where $\|G_N \ldots G_1 - U\| = \max_{|\psi\rangle} \|G_N \ldots G_1 |\psi\rangle - U|\psi\rangle\|$ *is the* operator norm.

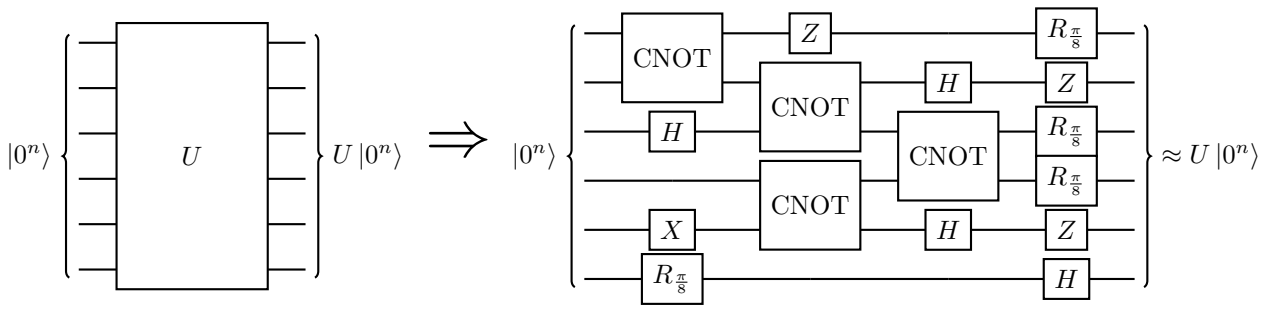


Figure 1.4: Any quantum unitary on n qubits can be approximately computed using a circuit that consists of 1 and 2 qubit gates. However, this number of gates often grows exponentially in n.

Corollary 1. *Any* 1 *or* 2 *qubit gates can be approximated by applying* $O(\log^4(\varepsilon))$ *gates from* $\mathcal{G}$ *with accuracy* ε*.*

If we know how to implement the gates in $\mathcal{G}$, then the running time of computing a unitary U will be the number of 1 and 2 qubit gates in $\mathcal{G}$ needed to properly approximate U. This definition depends on the gate set $\mathcal{G}$ which is to some extent unsatisfying. From the above corollary, we know that any 1 and 2 qubit gate can be efficiently approximated with a few gates from $\mathcal{G}$. Therefore, we make the simplifying assumption that any 1 and 2-qubit gate takes time 1, which is the quantum gate model.

Definition 1.31 (The quantum gate model)**.** *In* the quantum gate model*, the quantum running time of a unitary* U *is the amount of* 1 *and* 2*-qubit gates needed to apply* U*. The running time of a single-qubit measurement is* 1*.*

A first useful property is that if one can implement U using N one or two qubit gates then one can implement $U^\dagger$ with the same number of gates as follows: write $U = U_N \dots U_1$ where $U_1, \dots, U_N$ are one or two qubit gates. This means $U_1^\dagger, \dots, U_N^\dagger$ are all one or two qubit gates and we have $U^* = U_N^* \dots U_1^*$.

1.4.2 Simulating classical circuits with quantum circuits

Reversible classical circuits

Classical circuits use $\mathrm{NOT}, \mathrm{AND}, \mathrm{OR}$ gates while quantum unitaries are reversible so it is not clear whether quantum circuits are more powerful than classical circuits. We will show that this is indeed the case. In order to do this, we define the Toffoli gate:

Definition 1.32. *The Toffoli gate takes* 3 *input bits and outputs* 3 *bits, and does the following:*

$$\mathit{Toffoli}(x, y, z) = (x, y, z \oplus (x \wedge y)).$$

The Toffoli gate is a reversible gate meaning that $\mathrm{Toffoli}^{-1}$ is well defined.

Proposition 1.33. *Any classical circuit computing a function* $f(x)$ *consisting of* N *gates in the set* $\{\mathrm{NOT}, \mathrm{AND}, \mathrm{OR}\}$ *can be computed using* $O(N)$ *Toffoli gates only.*

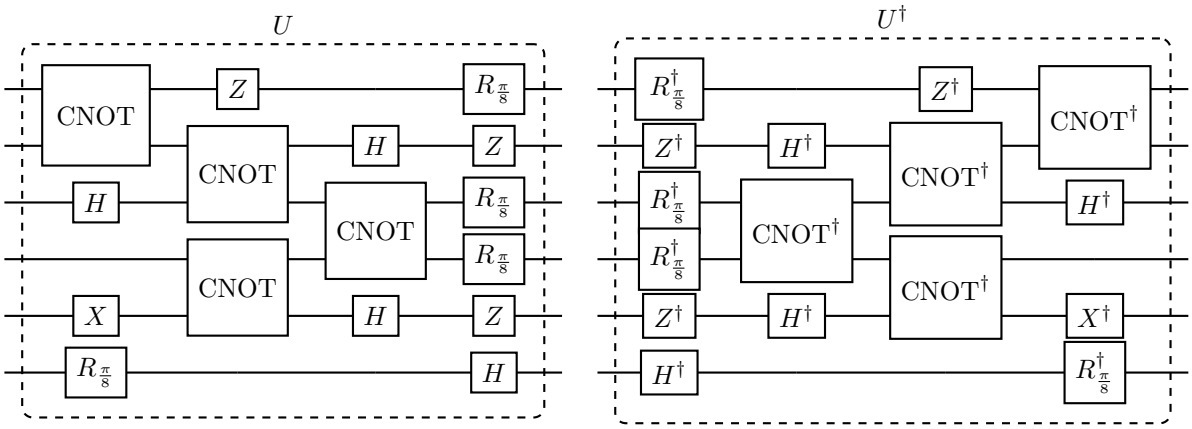


Figure 1.5: Constructing $U^\dagger$ from U. Just start from the end and reverse all the gates one by one. $U^\dagger$ has the same number of gates as U.

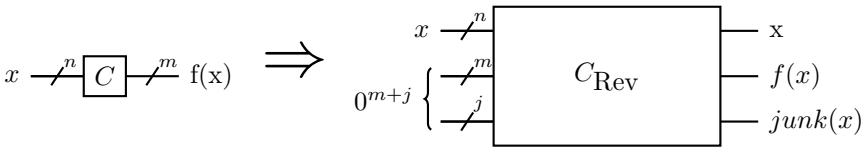


Figure 1.6: Any circuit C computing a function f with T gates can be transformed into a reversible circuit C_{Rev} that consists only of $O(T)$ Toffoli gates that computes $f(x)$ by preserving the input x, possibly with some junk state $junk(x)$.

Simulating classical circuits with quantum circuits

The Toffoli gate can also be interpreted as a quantum unitary on 3 qubits, satisfying $\text{Toffoli}(|x,y,z\rangle) = |x,y,z \oplus (x \wedge y)\rangle$ for any $x,y,z \in \{0,1\}$. Since the circuit C_{Rev} consists only of Toffoli, and because Toffoli gates can also be interpreted as quantum unitaries on 3 qubits, we can directly transform C_{Rev} into a quantum unitary U such that

$$U(|x\rangle \left|0^{m+j}\right\rangle) = |x\rangle \, |f(x)\rangle \, |junk(x)\rangle \, .$$

This already shows that any function f that can be computed classically in time T can be computed with a quantum computer in time $O(T)$ so quantum computers are at least as powerful as classical computers (up to the $O(\cdot)$ factor).

This is already interesting but we will improve this construction by removing the junk.

Proposition 1.34. *For any function $f : \{0,1\}^n \to \{0,1\}^m$ that can be computed classically with a circuit that runs in time T, there exists a quantum circuit on $n+m$ qubits that runs in time $O(T)$ that can perform the unitary*

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

where $\oplus$ here is the bitwise xor[3]*. This construction will use auxiliary qubits.*

[3]This means if $y = y_1 \dots y_m$ and $f(x) = z_1 \dots, z_m$, $y \oplus f(x) = (y_1 \oplus z_1) \dots (y_m \oplus z_m)$.

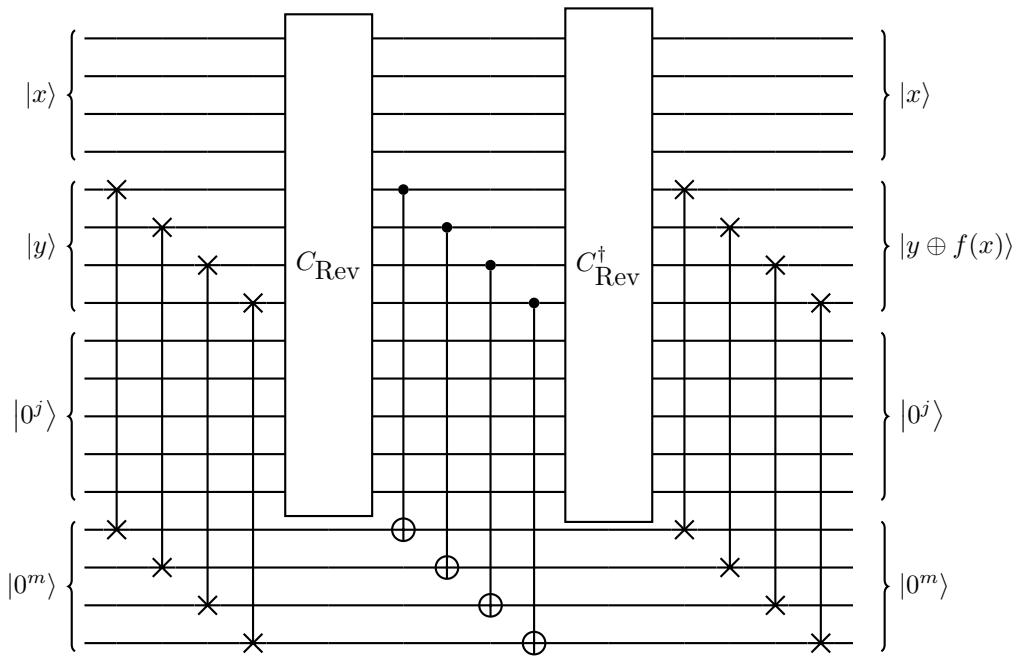


Figure 1.7: Construction O_f from C_{Rev}.

Proof. We fix a function $f : \{0,1\}^n \to \{0,1\}^m$ for which we have a circuit C with T gates computing f. We construct the reversible circuit C_{rev} from C using Proposition 1.33 that has $O(T)$ Toffoli gates and uses $n+m+j$ wires. We now give a construction U_f which is a unitary on $n+m$ qubits with $j+m$ additional auxiliary qubits. We consider the following procedure

1. On input $|x\rangle_n |y\rangle_m |0\rangle_j |0\rangle_m$, first swap the second and forth registers to get $|x\rangle_n |0\rangle_m |0\rangle_j |y\rangle_m$.
2. Apply C_{Rev} on the 3 first registers to get the state $|x\rangle |f(x)\rangle |junk(x)\rangle |y\rangle$.
3. For i from 1 to m, apply a CNOT gate between the i^{th} wire of the second register and the i^{th} wire of the forth register. We then have the state $|x\rangle |f(x)\rangle |junk(x)\rangle |y \oplus f(x)\rangle$.
4. Apply $C^\dagger_{\text{Rev}}$ on the three first registers to get the state $|x\rangle |0\rangle |0\rangle |y \oplus f(x)\rangle$.
5. Swap the second and forth register to get the state $|x\rangle |y \oplus f(x)\rangle |0\rangle |0\rangle$.

□

Chapter 2

First examples of interesting quantum circuits

2.1 The Deutsch-Jozsa algorithm

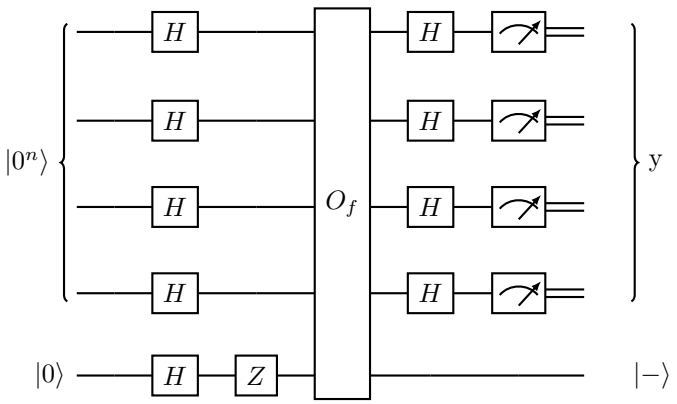
We present here our first example of a quantum algorithm that performs better than classical algorithms. It is extremely hard to prove in the usual model of computation that a quantum algorithm performs better than all classical algorithms. However, such statements are much easier to do in more specific models. We consider the following problem

Input: A function $f : \{0,1\}^n \to \{0,1\}$.
Promise: f is either a constant function or a balanced function, meaning that $\|\{x : f(x) = 0\}\| = \|\{x : f(x) = 1\}\| = 2^{n-1}$.
Goal: Find in which of these 2 cases we are.

As we said, it is extremely hard to prove in the usual model of computation that a quantum algorithm performs better than all classical algorithms. However, such statements are much easier to do in more specific models. Here, we will work in the query complexity model. Algorithms we consider will only have a black box access to f - quantum algorithm will have access to O_f.

Proposition 2.1. *There exists a quantum algorithm that gives the correct answer with probability* 1 *with a single quantum query.*

Proof. We first give the circuit description of our algorithm and then analyze each step of the computation.



1. Start with the $n+1$-qubit state

$$|\phi_0\rangle = |0^n\rangle\,|0\rangle\,.$$

2. Apply $H^{\otimes n}$ on the first n qubits and the unitary (ZH) on the last qubit.to obtain

$$|\phi_1\rangle = |+^n\rangle\,|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\,|-\rangle\,.$$

3. Apply O_f on the whole state to get

$$|\phi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\,\frac{1}{\sqrt{2}}\left(|f(x)\rangle - \left|\overline{f(x)}\right\rangle\right) = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}\,|x\rangle\,|-\rangle\,.$$

4. Apply $H^{\otimes n}$ on the first n qubits. We get

$$\begin{aligned}|\psi_3\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left(\frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}\,|y\rangle\right)|-\rangle \\ &= \frac{1}{2^n} \sum_{y\in\{0,1\}^n} \left(\sum_{x\in\{0,1\}^n} (-1)^{f(x)+x\cdot y}\right)|y\rangle\,|-\rangle\end{aligned}$$

5. Measure the n first qubits in the computational basis and get some outcome y. If $y = 0^n$, output "f is constant". Otherwise, output "f is balanced".

Let's now analyze the correctness of our algorithm. Let P_0 the probability that $y = 0^n$. Notice that for any $x \in \{0,1\}^n, x\cdot 0^n = 0$ hence we can write

$$P_0 = \left|\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}\right|^2 = \frac{1}{2^{2n}} \left|\sum_{x\in\{0,1\}^n} (-1)^{f(x)}\right|^2.$$

We distinguish 2 cases:

1. If f is a constant function, there is a $b \in \{0,1\}$ st. $\forall x \in \{0,1\}^n, f(x) = b$. In this case, we have $P_0 = \frac{1}{2^{2n}} \left|2^n(-1)^b\right|^2 = 1$ and our algorithm gives the correct answer wp. 1.

2. If f is balanced, we can write

$$P_0 = \frac{1}{2^{2n}} \left| \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \right|^2 = \frac{1}{2^{2n}} \left| \sum_{x\in\{0,1\}^n : f(x)=0} 1 - \sum_{x\in\{0,1\}^n : f(x)=1} 1 \right|^2 = 0.$$

   and our algorithm gives the correct answer wp. 1.

□

2.2 Simon's problem

Simon's problem is defined as follows

Input: A function $f : \{0,1\}^n \to \{0,1\}^n$.
Promise: $\exists s = (s_1, \ldots, s_n) \in \{0,1\}^n, (f(x) = f(y) \Leftrightarrow (x = y) \vee (x = y \oplus s))$.
Goal: Find s.

From the promise, we have that f is 2-to-1 meaning that for any $y \in Im(f)$, there exists exactly 2 values $x_0, x_1 \in \{0,1\}^n$, st. $f(x_0) = f(x_1) = y$. Also, this implies that half of the strings $y \in \{0,1\}^n$ are not in $Im(f)$. For each $y \in Im(f)$, let x_y be one of the preimages of y, so $f(x_y) = f(x_y \oplus s) = y$.

We perform the following algorithm:

1. Start from the $2n$ qubit state, with 2 registers of n qubits.

$$|\psi_0\rangle = |0^n\rangle |0^n\rangle .$$

2. Apply $H^{\otimes n}$ on the first n qubits to get

$$|\psi_1\rangle = |+^n\rangle |0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle |0^n\rangle .$$

3. Apply O_f on the state to get

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle |f(x)\rangle = \frac{1}{\sqrt{|Im(f)|}} \sum_{y\in Im(f)} \frac{1}{\sqrt{2}} \left(|x_y\rangle + |x_y \oplus s\rangle\right) |y\rangle .$$

4. Measure the second register and obtain some value $y \in Im(f)$. The resulting state on the first register is

$$|\psi_4(y)\rangle = \frac{1}{\sqrt{2}} \left(|x_y\rangle + |x_y \oplus s\rangle\right) .$$

5. Apply $H^{\otimes n}$ on the first register to get

$$|\psi_5(y)\rangle = \frac{1}{\sqrt{2^n}} \sum_{J\in\{0,1\}^n} \left(\frac{1}{\sqrt{2}}(-1)^{x_y \cdot J} + \frac{1}{\sqrt{2}}(-1)^{(x_y \oplus s)\cdot J} \right) |J\rangle .$$

Now, if $s \cdot J = 0$, we have $\left(\frac{1}{\sqrt{2}}(-1)^{x_y \cdot J} + \frac{1}{\sqrt{2}}(-1)^{(x_y \oplus s)\cdot J} \right) = \sqrt{2}(-1)^{x_y \cdot J}$ and if $s \cdot J = 1$, we have $\left(\frac{1}{\sqrt{2}}(-1)^{x_y \cdot J} + \frac{1}{\sqrt{2}}(-1)^{(x_y \oplus s)\cdot J} \right) = 0$. Therefore, we can write

$$|\psi_5(y)\rangle = \sqrt{\frac{2}{2^n}} \sum_{\substack{J\in\{0,1\}^n \\ s\cdot J=0}} (-1)^{x_y \cdot J} |J\rangle .$$

6. Measure this state in the computational basis. You get a random J satisfying $J \cdot s = 0$.

This algorithm gives us $J = (j_1, \ldots, j_n)$ satisfying $J \cdot s = 0$ meaning $\sum_{i=1}^{n} j_i s_i = 0$. We repeat the above algorithm M times (where M will be determined later), to get M random values $J^1, \ldots, J^M$ satisfying $J^k \cdot s = 0$ for $k \in [M]$. If we write $J_i^k = (j_1^k, \ldots, j_n^k)$ for $k \in [M]$ we have the following system:

$$\begin{bmatrix} j_1^1 s_1 & \oplus & j_2^1 s_2 & \oplus & \ldots & \oplus & j_n^1 s_n & = 0 \\ j_1^2 s_1 & \oplus & j_2^2 s_2 & \oplus & \ldots & \oplus & j_n^2 s_n & = 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ j_1^M s_1 & \oplus & j_2^M s_2 & \oplus & \ldots & \oplus & j_n^M s_n & = 0 \end{bmatrix}$$

Now, the idea is that by taking M large enough, we have enough equations to find s using Gaussian elimination. We have to take M such that the only solutions to the system are 0^n and s. This happens if we can extract $n-1$ linearly independent equations, which happens with overwhelming probability if we take $M = 3n$. Then, Gaussian elimination can be done in time $poly(M)$.

Chapter 3

Grover's algorithm

3.1 First description

Our goal is to solve the following problem.

Search problem
Input: a function $f : \{0,1\}^n \to \{0,1\}$. Goal: find x such that $f(x) = 1$.

We have a quantum access to f *i.e.* we have access to the quantum unitary O_f satisfying:

$$\forall x \in \{0,1\}^n,\ O_f(|x\rangle\,|0\rangle) = |x\rangle\,|f(x)\rangle\,.$$

Grover's algorithm: if there are T solutions, finds one with $O(\sqrt{\frac{2^n}{T}})$ calls to O_f and in time
$\widetilde{O}(\sqrt{\frac{2^n}{T}} \cdot Time(f))$ where we use the notation $\widetilde{O}(b(n)) = O\Big(b(n) \cdot polylog(b(n))\Big)$ for any $b(n)$.

Grover's algorithm

1. Create the state $|\psi_1\rangle = O_f\left(\frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} |x\rangle\,|0\rangle\right) = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\,|f(x)\rangle$.

2. Decompose into the good x and bad x (recall there are T solutions).

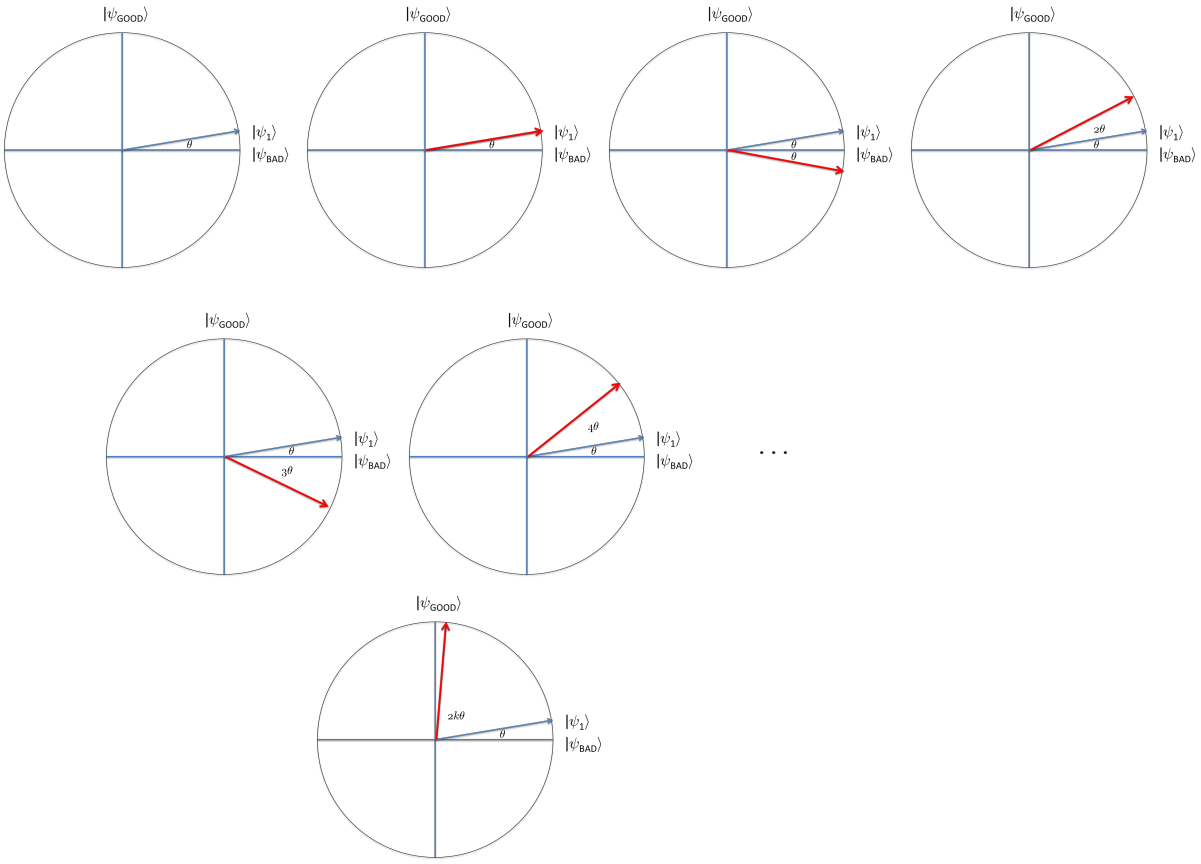
$$\begin{aligned}|\psi_1\rangle &= \sqrt{\frac{T}{2^n}} \underbrace{\sum_{x:f(x)=1} \frac{1}{\sqrt{T}} |x\rangle\,|f(x)\rangle}_{|\psi_{\text{Good}}\rangle} + \sqrt{1-\frac{T}{2^n}} \underbrace{\sum_{x:f(x)=0} \sqrt{\frac{1}{2^n-T}}\,|x\rangle\,|f(x)\rangle}_{|\psi_{Bad}\rangle} \\ &= \sqrt{\frac{T}{2^n}}\,|\psi_{\text{Good}}\rangle + \sqrt{1-\frac{T}{2^n}}|\psi_{Bad}\rangle\end{aligned}$$

3. Alternate the following 2 operations to transform $|\psi_1\rangle$ into a state close to $|\psi_{\text{Good}}\rangle$

- Phase flip over $f(x)$ register: $O_Z(|x\rangle\,|f(x)\rangle = (-1)^{f(x)}\,|x\rangle\,|f(x)\rangle$.
- Reflexion over $|\psi_1\rangle$: $\mathrm{Ref}_{|\psi_1\rangle}(|\psi_1\rangle) = |\psi_1\rangle$; $\mathrm{Ref}_{|\psi_1\rangle}(|\psi_1^{\perp}\rangle) = -\,|\psi_1\rangle^{\perp}$ for any $|\psi_1\rangle^{\perp}$ orthogonal to $|\psi_1\rangle$.

4. Geometrical interpretation in the subspace $\{|\psi_{\mathrm{Good}}\rangle\,, |\psi_{\mathrm{Bad}}\rangle\}$.
   - O_Z is a reflexion over $|\psi_{\mathrm{Bad}}\rangle$.
   - $\mathrm{Ref}_{|\psi_1\rangle}$ is a reflexion over $|\psi_1\rangle$.
   - $\left(O_Z \circ \mathrm{Ref}_{|\psi_1\rangle}\right)$ is a 2θ counter clockwise rotation where $\cos(\theta) := \langle\psi_1|\psi_{\mathrm{Bad}}\rangle$
   - $\langle\psi_1|\psi_{\mathrm{Bad}}\rangle = \sqrt{1-\frac{T}{2^n}}$ so $\theta \approx \sqrt{\frac{T}{2^n}}$ when $T \ll 2^n$.
5. Perform this rotation $\frac{1}{2\theta} - 1$ times to be close to $|\psi_{\mathrm{Good}}\rangle$.

Picturing the algorithm



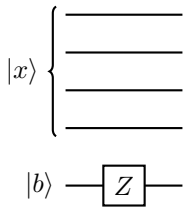


Figure 3.1: Circuit for B

3.2 More formally

We start from the state

$$|\psi_1\rangle = \sqrt{\frac{T}{2^n}}\,|\psi_{\text{Good}}\rangle + \sqrt{1-\frac{T}{2^n}}\,|\psi_{\text{Bad}}\rangle = \cos(\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\theta)\,|\psi_{\text{Good}}\rangle\,.$$

fpr some $\theta \in [0, \pi/2]$, where $|\psi_{\text{Good}}\rangle$ and $|\psi_{\text{Bad}}\rangle$ are defined in the previous section. We first 2 unitaries B and U that satisfy the following

$$\begin{aligned} B\,|\psi_{\text{Good}}\rangle = -\,|\psi_{\text{Good}}\rangle \;\;;\;\; B\,|\psi_{\text{Bad}}\rangle = |\psi_{\text{Bad}}\rangle \\ U\,|\psi_1\rangle = |\psi_1\rangle \;\;;\;\; U(|\phi\rangle) = -\,|\phi\rangle \;\text{ for }\; |\phi\rangle \perp |\psi_1\rangle \end{aligned}$$

For B, we consider the following $n+1$ qubit unitary:

$$B\,|x\rangle\,|b\rangle = (-1)^b\,|x\rangle\,|b\rangle\,.$$

One can easily check that

$$B\,|\psi_{\text{Good}}\rangle = -\,|\psi_{\text{Good}}\rangle \;\;;\;\; B\,|\psi_{\text{Bad}}\rangle = |\psi_{\text{Bad}}\rangle\,.$$

Now, for constructing U, we use the circuit U_1 st. $U_1\left|0^{n+1}\right\rangle = |\psi_1\rangle$ and show how to construct the reflexion over $|\psi_1\rangle$. You have seen this in exercise so I will just present here the final circuit that achieves this. We will also use the quantum unitary W (also seen in exercise) st.

$$W\,|x\rangle\,|0\rangle = (-1)^{g(x)}\,|x\rangle\,|0\rangle\,.$$

with $g : \{0,1\}^n \to \{0,1\}$ and $g(x) = 0 \Leftrightarrow x = 0^n$. From there, the circuit for constructing U is the following

Proposition 3.1. *Applying B then U is equivalent to performing a 2θ rotation unitary in the 2-dimensional subspace $\{|\psi_{Good}\rangle, |\psi_{Bad}\rangle\}$.*

Proof. We just need to prove this statement on a basis of $|\psi_{\text{Good}}\rangle, |\psi_{\text{Bad}}\rangle$. Recall that $|\psi_1\rangle = \cos(\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\theta)\,|\psi_{\text{Good}}\rangle$. Let us define

$$\left|\psi_1^{\perp}\right\rangle = \sin(\theta)\,|\psi_{\text{Bad}}\rangle - \cos(\theta)\,|\psi_{\text{Good}}\rangle\,.$$

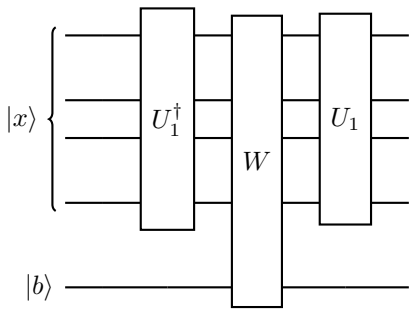


Figure 3.2: Circuit for U

Notice that $\left|\psi_1^\perp\right\rangle \perp |\psi_1\rangle$. From there, we write

$$\begin{aligned}
|\psi_{\text{Bad}}\rangle &= \cos(\theta)\,|\psi_1\rangle + \sin(\theta)\,\left|\psi_1^\perp\right\rangle \\
|\psi_{\text{Good}}\rangle &= \sin(\theta)\,|\psi_1\rangle - \cos(\theta)\,\left|\psi_1^\perp\right\rangle .
\end{aligned}$$

We write

$$\begin{aligned}
UB(\cos(\alpha)\,|\psi_{\text{Bad}}\rangle + \sin(\alpha)\,|\psi_{\text{Good}}\rangle) &= U(\cos(\alpha)\,|\psi_{\text{Bad}}\rangle - \sin(\alpha)\,|\psi_{\text{Good}}\rangle) \\
&= U(\cos(\alpha)\left(\cos(\theta)\,|\psi_1\rangle + \sin(\theta)\,\left|\psi_1^\perp\right\rangle\right) - \sin(\alpha)\left(\sin(\theta)\,|\psi_1\rangle - \cos(\theta)\,\left|\psi_1^\perp\right\rangle\right) \\
&= U\left((\cos(\alpha)\cos(\theta) - \sin(\theta)\sin(\alpha))\,|\psi_1\rangle + (\cos(\alpha)\sin(\theta) + \sin(\alpha)\cos(\theta))\,\left|\psi_1^\perp\right\rangle\right) \\
&= U(\cos(\alpha+\theta)\,|\psi_1\rangle + \sin(\alpha+\theta)\,\left|\psi_1^\perp\right\rangle \\
&= \cos(\alpha+\theta)\,|\psi_1\rangle - \sin(\alpha+\theta)\,\left|\psi_1^\perp\right\rangle
\end{aligned}$$

In order to conclude, we write

$$\begin{aligned}
|\psi_F\rangle &= \cos(\alpha+\theta)\,|\psi_1\rangle - \sin(\alpha+\theta)\,\left|\psi_1^\perp\right\rangle \\
&= \cos(\alpha+\theta)\,(\cos(\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\theta)\,|\psi_{\text{Good}}\rangle) - \sin(\alpha+\theta)\,(\sin(\theta)\,|\psi_{\text{Bad}}\rangle - \cos(\theta)\,|\psi_{\text{Good}}\rangle) \\
&= (\cos(\alpha+\theta)\cos(\theta) - \sin(\alpha+\theta)\sin(\theta))\,|\psi_{\text{Bad}}\rangle + \cos(\alpha+\theta)\sin(\theta) + \sin(\alpha+\theta)\cos(\theta)\,|\psi_{\text{Good}}\rangle \\
&= \cos(\alpha+2\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\alpha+2\theta)\,|\psi_{\text{Good}}\rangle .
\end{aligned}$$

□

We can now present Grover's algorithm. Recall the search problem we want to solve

Search problem
Input: a function $f : \{0,1\}^n \to \{0,1\}$. Goal: find x such that $f(x) = 1$. Let $T = \lvert\{x : f(x) = 1\}\rvert$.

Grover's algorithm

1. We construct $|\psi_1\rangle = \sqrt{\frac{T}{2^n}}\,|\psi_{\text{Good}}\rangle + \sqrt{1-\frac{T}{2^n}}\,|\psi_{\text{Bad}}\rangle = \cos(\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\theta)\,|\psi_{\text{Good}}\rangle$.
2. Apply $R = \lceil(\frac{\pi}{2}-\theta)\cdot\frac{1}{2\theta}\rceil$ times the unitary (UB) on the state $|\psi_1\rangle$.
3. Measure the $n+1$ qubits. If the last qubit is 1, the first n qubits contain x st. $f(x)=1$. Otherwise, repeat from step 1.

Let $|\psi_{\text{Final}}\rangle$ be the final quantum state before the measurement. From our previous proposition, we have that

$$|\psi_{\text{Final}}\rangle = \cos(\theta+2R\theta)\,|\psi_{\text{Bad}}\rangle + \sin(\theta+2R\theta)\,|\psi_{\text{Good}}\rangle\,.$$

The probability P that the final measurement gives a solution is

$$P = \sin^2(\theta+2R\theta).$$

We have

$$\pi/2-\theta \le 2\theta R \le 2\theta\left((\pi/2-\theta)\frac{1}{2\theta}+1\right) = \frac{\pi}{2}-\theta+2\theta.$$

From which we have $P \ge \sin^2(\pi/2+2\theta) = \cos^2(-2\theta) = \cos^2(2\theta)$.

Final analysis. We start from the case where the fraction of solution is less than $\frac{1}{4}$ meaning we take $T \le \frac{2^n}{4} = 2^{n-2}$. If this is not the case then we can find a solution in time $O(1)$ by computing $f(x)$ for random values of x until we find a solution.

Now we have $T \le 2^{n-2}$, one can check that this implies $\theta \le \pi/6$. From there, we have $P \ge \cos^2(\pi/3) = \frac{1}{4}$. The algorithm finds a solution with constant probability.

Now let's compute R. We have $R \le \frac{\pi}{4\theta}+1$ (since $0<\theta<\pi/2$). We have $\theta = \arcsin(\sqrt{\frac{T}{2^n}})$ which gives $\theta \ge \sqrt{\frac{T}{2^n}}$ (one can check that $\arcsin(x) \ge x$ for $x \in [0,1]$). From there, we have

$$R \le \frac{\pi}{4\theta}+1 \le \frac{\pi}{4}\cdot\sqrt{\frac{2^n}{T}} = O(\sqrt{\frac{2^n}{T}}).$$

From which we conclude that Grover's total running time is $O(\sqrt{\frac{2^n}{T}})$.

3.3 Amplitude amplification

We now present a generalization of Grover's algorithm which will be very useful. Recall the search problem that we solve using Grover's algorithm:

Search problem

Input: An efficiently computable function $f:\{0,1\}^n \to \{0,1\}$.
Goal: find a solution x such that $f(x)=1$. Goal: find x such that $f(x)=1$. Let $T = |\{x : f(x)=1\}|$.

Assume you have a classical or quantum algorithm $\mathscr{A}$ that can find a solution x in time T wp. p. One can repeat $\mathscr{A}$ $O(\frac{1}{p})$ times in order to find a solution in time $O(\frac{T}{p})$ with constant probability. Amplitude amplification improves this in the following way:

Theorem 3.2 (Amplitude amplification)**.** *Assume you have a classical or quantum algorithm $\mathscr{A}$ that can find a solution x to the search problem in time τ with probability p. If f is efficiently computable then you can find a solution in time $O(\frac{\tau}{\sqrt{p}})$ with constant probability to the same search problem.*

Before proving this theorem, notice that this is a generalization of Grover's algorithm. Indeed, consider for $\mathscr{A}$ the following algorithm: pick a random $x \in \{0,1\}^n$ and output x. This algorithm runs in time $\tau = O(n)$ (just the time to construct a random x) and succeeds wp. $p = \frac{T}{2^n}$. Using the above, you can find a solution in time $\widetilde{O}\left(\sqrt{\frac{2^n}{T}}\right)$. Amplitude amplification is more useful since sometimes, we will know algorithms better than random search and amplitude amplification shows we also have a quadratic speedup in this setting.

Proof. In order to prove this theorem, we first need to characterize what does it mean to have an algorithm $\mathscr{A}$ that finds a solution x wp. p. We assume $\mathscr{A}$ is an algorithm that performs only measurements at the end of the computation so $\mathscr{A}$ starts from $|0^m\rangle$ for some $m \in \mathbb{N}$ and then, before the final measurement, has a state $|\psi\rangle$ and measuring the output register gives a solution x wp. p. This means we can write $|\psi\rangle$ as

$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x |x\rangle |\phi_x\rangle, \quad \text{where} \sum_{x:f(x)=1} |\alpha_x|^2 = p.$$

where the $|\phi_x\rangle$ are some quantum states. We define again

$$|\psi_{Good}\rangle = \frac{1}{\sqrt{p}} \sum_{x:f(x)=1} \alpha_x |x\rangle |\phi_x\rangle$$
$$|\psi_{Bad}\rangle = \frac{1}{\sqrt{1-p}} \sum_{x:f(x)=0} \alpha_x |x\rangle |\phi_x\rangle$$

From which we can write $|\psi\rangle = \sqrt{p}\,|\psi_{Good}\rangle + \sqrt{1-p}\,|\psi_{Bad}\rangle$. Since f is efficiently computable, we can efficiently construct

$$O_f(|x\rangle |y\rangle) = |x\rangle |y + f(x)\rangle \quad \text{and} \quad O'_f(|x\rangle) = (-1)^{f(x)} |x\rangle .$$

Now, we perform the same algorithm as for Grover. We showed how from algorithm $\mathscr{A}$ to compute the unitary $Ref_{|\psi\rangle}$ which is the reflexion over $|\psi\rangle$. Moreover, again as in Grover's algorithm we have

$$(O'_f \otimes I)(|\psi_{Good}\rangle) = -|\psi_{Good}\rangle \ ; \ (O'_f \otimes I)(|\psi_{Bad}\rangle) = |\psi_{Bad}\rangle .$$

With the reflexion over $|\psi\rangle$ and the reflexion over $|\psi_{Bad}\rangle$ (in the 2 dimensional subspace $\{|\psi_{Good}\rangle, |\psi\rangle_{Bad}\}$), we can apply all the steps of Grover's algorithm that finds a solution in time $O(\frac{1}{\sqrt{p}})$. □

3.3.1 What about randomized classical algorithms?

How do we use amplitude amplification for classical randomized algorithms? We showed in a previous lecture how from a description of an efficient classical circuit computing f, one compute O_f efficiently. We will use this to transform a randomized algorithm into a quantum algorithm. A classical randomized algorithm $\mathscr{A}$ can be described as follows

$$\mathscr{A} : \text{pick a random } R \in \{0,1\}^r, \text{ compute } \mathscr{A}(R) \text{ to get some outcome } x_R.$$

Here, we just changed the algorithm so that all the randomness needed is chosen at the beginning of the algorithm. We can therefore interpret $\mathscr{A}$ as a deterministic function that takes as input R and outputs x_R, and we can therefore compute the efficiently unitary

$$O_{\mathscr{A}}(|R\rangle |y\rangle) = |R\rangle |y + x_R\rangle .$$

In order to conclude, consider the following quantum algorithm:

$$|0^r\rangle |0^n\rangle \xrightarrow{H^{\otimes r}\otimes I} \frac{1}{\sqrt{2^r}} \sum_{R\in\{0,1\}^r} |R\rangle |0\rangle \xrightarrow{O_{\mathscr{A}}} \frac{1}{\sqrt{2^r}} \sum_{R\in\{0,1\}^r} |x_R\rangle .$$

which by definition outputs a solution wp. p. We can therefore use amplitude amplification on this algorithm.

Chapter 4

The quantum Fourier transform

The quantum Fourier will be one of our main tools when constructing quantum algorithms. It will be at the heart of Shor's factoring algorithm.

4.1 The classical Fourier transform

Widely used: data compression, signal processing, complexity theory. Here, we will consider only the discrete Fourier transform.

4.1.1 Definition

Fourier transform F_N: $N \times N$ unitary matrix, with elements of the same magnitude.

$$F_2 := H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

$N = 3$ (for example): impossible to achieve with real numbers. Use complex numbers. We will use roots of unity $\omega_N = e^{\frac{2i\pi}{N}}$. The discrete Fourier transform F_N is defined as

$F_N := \frac{1}{\sqrt{N}} \begin{pmatrix} \ddots & \vdots & \iddots \\ \dots & \omega_N^{jk} & \dots \\ \iddots & \vdots & \ddots \end{pmatrix}$ meaning that for any line $j \in \{0, \dots, N-1\}$ and any column $k \in \{0, \dots, N-1\}$, we have $(F_N)_{jk} := \omega_N^{jk}$.

Properties

- Each column $C_k = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 \\ \omega_N^k \\ \vdots \\ \omega_N^{(N-1)k} \end{pmatrix}$ has norm 1 and any two columns are orthog-

onal. Indeed

$$\forall k, k' \in \{0, \dots, N-1\},\ \frac{1}{N} \sum_{j=0}^{N-1} \omega_N^{jk} \omega_N^{-jk'} = \frac{1}{N} \sum_{j=0}^{N-1} \omega_N^{j(k-k')} = \delta_{kk'}.$$

4.1.2 Computing the classical Fourier transform

Computing the classical Fourier transform problem
Input: a column vector $v = \begin{pmatrix} v_0 \\ \vdots \\ v_{N-1} \end{pmatrix}$.
Goal: compute $\hat{v} = F_N \cdot v$.

Naive way: Perform the whole multiplication entry-wise

- $O(N)$ operations ($+$ and $\times$) per entry.
- $O(N^2)$ operations in total.

Fast Fourier transform

- $O(N \log(N))$ operations in total.
- recursive algorithm.

4.2 The quantum Fourier transform

4.2.1 Definition of the problem

Take $N = 2^n$. Since F_N is a $N \times N$ unitary matrix, we can interpret it as a quantum unitary operation acting on n qubits.

Computing the quantum Fourier transform problem
Input: a quantum state $\|\psi\rangle$ of n qubits.
Goal: output $F_N(\|\psi\rangle)$.

How efficiently can we implement this quantum Fourier transform?

- QFT can be implemented with a quantum circuit of size $O(n^2)$. The rest of the chapter will be devoted to the construction of this algorithm.
- Exponentially faster than classical FFT which runs in $O(N \log(N))$.

⚠ In the classical setting, we are given an explicit (written) description of a vector $v = \begin{pmatrix} v_0 \\ \vdots \\ v_{N-1} \end{pmatrix}$ as an input and ask to have a similar description of the output. In the quantum setting, we are given a quantum state $|\psi\rangle = \sum_{i=0}^{N-1} v_i |i\rangle$ and ask to output the state

$F_N(|\psi\rangle) = \sum_{i=0}^{N-1} \hat{v}_i |i\rangle$. Notice that we cannot fully recover the vector $\hat{v} = \begin{pmatrix} \hat{v}_0 \\ \vdots \\ \hat{v}_{N-1} \end{pmatrix}$ from $F_N(|\psi\rangle)$.

Even though the speedup is exponential, it doesn't allow to recover $\hat{v}$, which makes it incomparable with the FFT. The quantum algorithm in particular doesn't help in computing the classical Fourier transform. It will still have other important uses.

4.2.2 Efficient quantum circuit for QFT

Elementary gates

- Hadamard gate $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$.
- Phase rotation $R_s = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{2i\pi}{2^s}} \end{pmatrix}$.
- Controlled R_s gate written $C\text{-}R_s$ and acting on 2 qubits.

  $C\text{-}R_s(|0\rangle |x\rangle) := |0\rangle |x\rangle \quad ; \quad C\text{-}R_s(|1\rangle |x\rangle) := |1\rangle R_s(|x\rangle) \quad$ which gives $\quad C\text{-}R_s(|b\rangle |x\rangle) = |b\rangle e^{\frac{2i\pi bx}{2^s}} |x\rangle$.

For each k, we have $F_N(|k\rangle) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} |j\rangle$. For any integer j, we write its binary decomposition $j = j_1, \ldots, j_n$ where j_1 is the bit of highest weight. This means we can write $j = \sum_{l=1}^{n} 2^{n-l} j_l$. We have

$$\begin{aligned} F_N(|k\rangle) &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2i\pi jk}{2^n}} |j\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2i\pi (\sum_{l=1}^{n} j_l 2^{-l})k} |j_1, \ldots, j_n\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \Pi_{l=1}^{n} e^{2i\pi j_l 2^{-l} k} |j_1, \ldots, j_n\rangle \\ &= \bigotimes_{l=1}^{n} \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2i\pi k 2^{-l}} |1\rangle \right) \end{aligned}$$

To prove the last equality, recall that the tensor product satisfies

$$(\alpha |0\rangle + \beta |1\rangle) \otimes (\alpha' |0\rangle + \beta' |1\rangle) = \alpha\alpha' |00\rangle + \alpha\beta' |01\rangle + \beta\alpha' |10\rangle + \beta\beta' |11\rangle .$$

For any integer k, with binary decomposition $k = k_1, \ldots, k_n$, we define $0.k := \frac{k}{2^n} = \sum_{l=1}^{n} k_l 2^{-l}$. For example, $0.010 = \frac{1}{4}$ and $0.101 = \frac{5}{8}$. Notice that

$$\begin{aligned} e^{2i\pi k 2^{-l}} &= e^{2i\pi (\sum_{m=1}^{n} 2^{n-m} k_m) 2^{-l}} \\ &= e^{2i\pi \left(\sum_{m=n-l+1}^{n} k_m 2^{m-(n-l+1)} \right) 2^{-l}} \\ &= e^{2i\pi \left(\sum_{m'=1}^{l} k_{n-m'+1} 2^{-m'} \right)} \\ &= 0.k_n \ldots k_{n-l+1}. \end{aligned}$$

The second equality uses the fact $e^{2i\pi C} = 1$ for any $C \in \mathbb{N}$ which implies that only the $l-1$ bits of k of least weight matter in the term $e^{2\mathrm{i}\pi k 2^{-l}}$. We can therefore rewrite

$$F_N(|k\rangle) = \bigotimes_{l=1}^{n} \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\mathrm{i}\pi k 2^{-l}} |1\rangle\right) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_n} |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_{n-1}k_n} |1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1 \ldots k_n} |1\rangle$$

The $n = 3$ ($N = 8$) case

From the above, we have

$$F_8(|k_1k_2k_3\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_3} |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_2k_3} |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1k_2k_3} |1\rangle).$$

Because of this product structure, we can easily construct each qubit separately. Notice that with the $0.k$ notation, we can write $C\text{-}R_s(|b\rangle |x\rangle) = |b\rangle e^{\frac{2\mathrm{i}\pi bx}{2^s}} |x\rangle = |b\rangle e^{2\mathrm{i}x\pi \cdot 0.0 \ldots 0b} |x\rangle$. where $0.\underbrace{0 \ldots 0}_{s \text{ bits}}b$.

1st qubit:

$$|k_3\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_3} |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_3} |1\rangle).$$

2nd qubit:

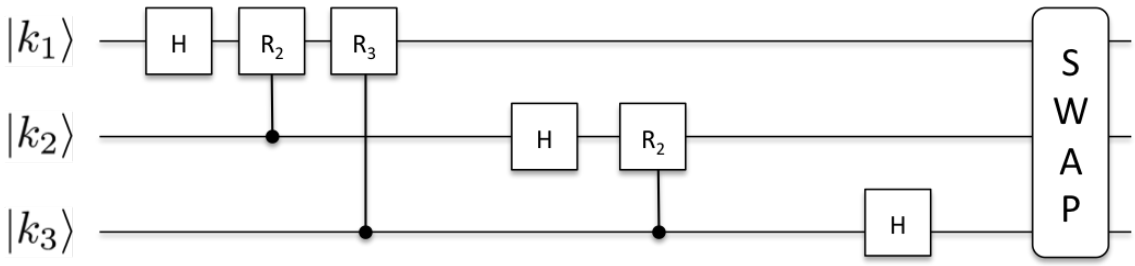
$$|k_2\rangle \xrightarrow{\mathrm{H}} \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_2} |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_2} |1\rangle)$$
$$|k_3\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_2} |1\rangle) \xrightarrow{C\text{-}R_2} |k_3\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_2k_3} |1\rangle)$$

3rd qubit:

$$|k_1\rangle \xrightarrow{\mathrm{H}} \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_1} |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1} |1\rangle)$$
$$|k_2\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1} |1\rangle) \xrightarrow{C\text{-}R_2} |k_2\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1k_2} |1\rangle)$$
$$|k_3\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1k_2} |1\rangle) \xrightarrow{C\text{-}R_3} |k_3\rangle \frac{1}{\sqrt{2}}(|0\rangle + e^{2\mathrm{i}\pi 0.k_1k_2k_3} |1\rangle)$$

The 3rd qubit is stored in the quantum register where $|k_1\rangle$ is and uses $|k_2\rangle$ and $|k_3\rangle$ as control qubits. The 2nd qubit is be stored in the quantum register where $|k_2\rangle$ is and uses $|k_3\rangle$ as a control qubit. The 1st qubit is stored where $|k_3\rangle$ is. In order to do this, we must first construct the 3rd qubit, then the 2nd and finally the 1st qubit. In order to have the good ordering of qubits, we end up inverting the order of all the qubits.

The circuit of F_8 is the following



The number of gates used here is $3+2+1$ gates + the gates in the SWAP.

General case

The construction for $n = 3$ can be extended to any n following the same pattern. The first qubit will consist only of an H gate while the last qubit will require applying H, $C\text{-}R_2, \ldots, C\text{-}R_n$. Similarly as in the $n = 3$ case, we finish by inverting the order of all the qubits. The total number of gates used is therefore $n+n-1+\cdots+1+\text{SWAP} = O(n^2)$gates.

Improvements: we can reduce the number of gates if we allow for some small errors:

- As s grows, $C\text{-}R_s$ fastly converges to the identity gate.
- We remove all those gates for $s \geq \Omega(\log(n))$. By a careful error analysis, we can show that the result will be $O(\frac{1}{poly(n)})$ close to the desired state.
- The total amount of needed gates becomes $O(n\log(n))$ (SWAP is $O(n)$).

4.3 Phase estimation

Our first protocol is a direct application of the quantum Fourier transform.

Phase estimation
Input: a quantum unitary U acting on n qubits. An eigenvector $\|\psi\rangle$ of U with eigenvalue λ given as a quantum state. Goal: output λ.

Recall that an eigenvector $|\psi\rangle$ of U with eigenvalue λ means that $U(|\psi\rangle) = \lambda\,|\psi\rangle$. Because U is a unitary, $|\lambda| = 1$ so we can write $\lambda = e^{2i\pi\phi}$ for some real number $\phi \in [0, 1)$ ($[0, 1[$ in French notation). We assume first that ϕ can be fully described with l bits of precision, *i.e.* there exists a natural number $C \in \mathbb{N}$ such that $\phi = \frac{C}{2^l}$.

We consider a quantum unitary Q satisfying

$$Q(|k\rangle\,|\psi\rangle) = |k\rangle\,U^k(|\psi\rangle).$$

for any $k \in \{0, \ldots, 2^l - 1\}$ and any state $|\psi\rangle$. We perform the following algorithm:

1. Start from $|0\rangle|\psi\rangle$ and apply F_{2^l} on the first register. The resulting state is

$$\frac{1}{\sqrt{2^l}}\sum_{k=0}^{2^l-1}|k\rangle|\psi\rangle.$$

2. Apply Q on both registers.

$$\begin{aligned}\frac{1}{\sqrt{2^l}}\sum_{k=0}^{2^l-1}U^k|k\rangle|\psi\rangle &= \frac{1}{\sqrt{2^l}}\sum_{k=0}^{2^l-1}\lambda^k|k\rangle|\psi\rangle \\ &= \frac{1}{\sqrt{2^l}}\sum_{k=0}^{2^l-1}e^{\frac{2i\pi kC}{2^l}}|k\rangle|\psi\rangle.\end{aligned}$$

3. Apply the inverse Fourier transform $F_{2^l}^{-1}$ on the first register. The resulting state is $|C\rangle|\psi\rangle$. It is then easy to recover ϕ from C.

General case If ϕ cannot be written with l bits of precision, we consider the closest approximation of ϕ of the form $\frac{C}{2^l}$. An error analysis (not detailed here) shows that the above procedure will find this C with probability at least $\frac{4}{\pi^2}$. By performing several iterations of this procedure, we can find the correct C, *i.e.* a good approximation of ϕ with a probability that exponentially converges to 1 in the number of iterations.

Efficiency of the algorithm If U can be computed efficiently and if, for any $k \in \{0,\dots,2^l-1\}$, U^k can be computed efficiently then Q can be computed efficiently and the whole algorithm is efficient. This means that if we want our algorithm to run in time $poly(n)$ (assuming U can be computed in time poly(n)), we have to take $l = O(\log(n))$.leq

4.4 Application: Fourier transform F_N for any $N \in \mathbb{N}$

In Chapter 4, we showed how to perform the Fourier transform F_N when $N = 2^n$ for some $n \in \mathbb{N}$. Here, we show how to perform the Fourier transform for any N. F_N will act on a quantum register that can take N values from 0 to $N-1$ and

$$\forall k \in \{0,\dots,N-1\},\ F_N(|k\rangle) = \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1}\omega^{jk}|j\rangle.$$

where $\omega := e^{\frac{2i\pi}{N}}$. Let U_1 and U_2 two unitaries that do the following, $\forall k \in \{0,\dots,N-1\}$.

$$U_1(|k\rangle|0\rangle) = |k\rangle F_N(|k\rangle) \quad ; \quad U_2(F_N|k\rangle|0\rangle) = F_N(|k\rangle)|k\rangle.$$

From those two unitaries, we can perform F_N as follows

$$|k\rangle|0\rangle \xrightarrow{U_1} |k\rangle F_N(|k\rangle) \xrightarrow{SWAP} F_N(|k\rangle)|k\rangle \xrightarrow{U_2} F_N(|k\rangle)|0\rangle.$$

what is left to show is how to perform U_1 and U_2. U_1 can be performed quite easily.

Let S_N a quantum unitary such that $S_N(|0\rangle) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle$. Since N is not a power of 2, S_N cannot be expressed as Hadamards but we can still easily construct such a unitary. Let also O_{mult} satisfying $O_{mult}(|k\rangle |j\rangle |0\rangle) = |k\rangle |j\rangle |kj \mod N\rangle$.

Let's now construct U_1.

1. Start from $|k\rangle |0\rangle |0\rangle$ and apply S_N on the second register. The resulting state is
$$|k\rangle \frac{1}{\sqrt{N}} \sum_j |j\rangle |0\rangle .$$
2. Apply O_{mult} on the three registers.
$$|k\rangle \frac{1}{\sqrt{N}} \sum_j |j\rangle |kj \mod N\rangle .$$
3. Apply the unitary $|x\rangle \to \omega^x |x\rangle$ on the third register.
$$|k\rangle \frac{1}{\sqrt{N}} \sum_j \omega^{kj} |j\rangle |kj \mod N\rangle .$$
4. Apply O_{mult}^{-1} on the three registers. We obtain
$$|k\rangle \frac{1}{\sqrt{N}} \sum_j \omega^{kj} |j\rangle |0\rangle = |k\rangle F_N(|k\rangle) |0\rangle .$$

Swapping registers is then easy. What is left to do is to perform unitary U_2. We consider the unitary O_{add} such that $O_{add}(|k\rangle) = |k+1 \mod N\rangle$. The idea is that $F_N(|k\rangle)$ is a eigenvector of O_{add} with eigenvalue λ_k and that it will be easy to recover k from λ_k. Doing this in a coherent way will give U_2.

First see that
$$O_{add} F_N(|k\rangle) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega^{jk} |j+1 \mod N\rangle = \omega^{-k} \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega^{jk} |j\rangle$$
which means that $F_N(|k\rangle)$ is an eigenvector of O_{add} with eigenvalue $\omega^{-k} = \omega^{N-1-k}$. If we apply the phase estimation from the previous section, with $l = \lceil \log(N) \rceil$, we obtain
$$F_N(|k\rangle) |0\rangle \xrightarrow{\text{Phase Estimation with } O_{add}} F_N(|k\rangle) |N-1-k\rangle .$$
Which by applying $|N-1-x\rangle \to |x\rangle$ gives the correct result. Actually, the phase estimation will only give $N-k$ with some (high) probability so this whole process will construct an approximation of U_2.

Putting everything together, we managed to construct $F_N(|k\rangle)$. This unitary has many applications, for example for the Discrete Log quantum algorithm.

Chapter 5

Shor's quantum factoring algorithm

Shor's idea:

- There exists an efficient quantum algorithm for finding the period of a function.
- Factoring can be reduced to period finding *i.e.* an efficient algorithm for period finding $\Rightarrow$ an efficient algorithm for factoring.

Period finding problem
Input: a function $f : \mathbb{N} \to \{0, \ldots, N-1\}$ such that $\exists r \in \{0, \ldots, N-1\}$ (unknown) such that $f(a) = f(b) \Leftrightarrow a = b \mod r$. Goal: output r.

5.1 From factoring to period finding

5.1.1 Classical algorithm for factoring a number N using period finding

Equivalent to finding a non trivial factor of N.

1. Pick a random $x \in \{2, \ldots, N-1\}$.
2. Calculate $x \wedge N$ (efficient, use Euclid's algorithm).
   - if $x \wedge N = c \neq 1 \to c$ divides N.
   - if $x \wedge N = 1 \to$ continue.
3. Consider the smallest $r \in \{0, \ldots, N-1\}$ such that $x^r = 1 \mod N$. Since $x \wedge N = 1$, such an r exists.
4. r is the period of the function $f(k) = x^k \mod N$. Use the period finding algorithm to find r. If r is odd, go back to step 1.

5. Calculate $(x^{r/2}+1) \wedge N$ and $(x^{r/2}-1) \wedge N$. If one of those values is different than 1 or N then this value is a non trivial factor of n. If both of those values are equal to 1 or N, start again from step 1.

5.1.2 Proof that the algorithm works

The main part of the proof will be the following lemma from number theory. The proof will be omitted.

Lemma 1. *For any odd N, for a randomly chosen x such that $x \wedge N = 1$ and r begin the smallest element in $\{0, \ldots, n-1\}$ satisfying $x^r = 1 \mod N$, the event*

$$E: \quad r \text{ is even} \quad \wedge \quad (x^{r/2}+1) \neq 0 \mod N \quad \wedge \quad (x^{r/2}+1) \neq 0 \mod N$$

occurs with probability $\geq \frac{1}{2}$.

If r is even, we have

$$\begin{aligned} x^r = 1 \mod N &\Leftrightarrow (x^{r/2}+1)(x^{r/2}-1) = 0 \mod N \\ &\Leftrightarrow \exists k \in \mathbb{N}^*,\ (x^{r/2}+1)(x^{r/2}-1) = kN. \end{aligned}$$

Notice first that $(x^{r/2}+1) > 0$ and we also have $(x^{r/2}-1) > 0$ because $x \geq 2$ and $r \geq 2$. If E holds, both $(x^{r/2}+1)$ and $(x^{r/2}-1)$ are not multiples of N. Therefore, they will both have a non trivial factor of N and we actually have $(x^{r/2}-1) \wedge N \neq 1$ and $(x^{r/2}+1) \wedge N \neq 1$. Conclusion: if E holds then step 5 outputs a non trivial factor of N. Since $\Pr[E] \geq \frac{1}{2}$, we require $O(1)$ calls to the period finding algorithm for the algorithm to succeed with a high (constant) probability.

5.2 Shor's period finding algorithm

Our goal here is to present Shor's quantum algorithm for period finding. Let $n := \lceil \log(N) \rceil$, $q := \lceil \log(N^2) \rceil$ and $Q := 2^q \in [N^2, 2N^2]$. We have a quantum access to $f : \mathbb{N} \to \{0, \ldots, N-1\}$. We restrict the input space to q input bits and consider the quantum unitary

$$O_f : |x\rangle_q |0\rangle_n \to |x\rangle_q |f(x)\rangle_n .$$

The subscripts represent the number of qubits in each register. This means for example that register $|x\rangle_q$ contains q qubits and register $|0\rangle_n$ contains n qubits.

5.2.1 Algorithm for period finding

1. Initialize the protocol at the state

$$|0\rangle_q |0\rangle_n .$$

2. Apply QFT_Q on the first register. We get

$$\frac{1}{\sqrt{Q}} \sum_{a=0}^{Q-1} |a\rangle_q |0\rangle_n .$$

3. Apply O_f on the whole state to obtain

$$\frac{1}{\sqrt{Q}} \sum_{a=0}^{Q-1} |a\rangle_q |f(a)\rangle_n .$$

4. Measure the second register: it gives some value $f(s)$ for some $s < r$. Let $m := \#\{a \in \{0, \ldots, Q-1\} : f(a) = f(s)\}$. We have

$$\begin{aligned} \{a \in \{0, \ldots, Q-1\} : f(a) = f(s)\} &= \{s, s+r, \ldots, s+(m-1)r\} \\ &= \{jr+s\}_{0 \le j < m} \end{aligned}$$

When measuring $f(s)$ in the second register, the first register collapses to

$$\frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} |jr+s\rangle .$$

5. Apply QFT_Q on this first register.

$$\begin{aligned} &\frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} \frac{1}{\sqrt{Q}} \sum_{b=0}^{Q-1} e^{\frac{2i\pi(jr+s)b}{Q}} |b\rangle \\ &= \frac{1}{\sqrt{mQ}} \sum_{b=0}^{Q-1} e^{\frac{2i\pi sb}{Q}} \left(\sum_{j=0}^{m-1} e^{\frac{2i\pi jrb}{Q}} \right) |b\rangle \end{aligned}$$

6. Measure the first register. What is the probability of outputting each specific b?

Special case developed here : r divides Q.

In this case, we have $m = \frac{Q}{r}$. We have

$$b \text{ is a multiple of } \frac{Q}{r} \Leftrightarrow e^{\frac{2i\pi rb}{Q}} = 1.$$

Any such b will therefore have squared amplitude

$$\begin{aligned} &\left| \frac{1}{\sqrt{mQ}} e^{\frac{2i\pi sb}{Q}} \left(\sum_{j=0}^{m-1} e^{\frac{2i\pi jrb}{Q}} \right) \right|^2 \\ &= \left| \frac{1}{\sqrt{mQ}} e^{\frac{2i\pi sb}{Q}} \left(\sum_{j=0}^{m-1} 1 \right) \right|^2 \\ &= \frac{m}{Q} = \frac{1}{r} \end{aligned}$$

Each $b \in \{0, \ldots, Q-1\}$ which is a multiple of $\frac{Q}{r}$ will be measured with probability exactly $\frac{1}{r}$. Notice also that there are exactly r such multiples, which are the elements

of $\{0, \frac{Q}{r}, \ldots, \frac{(r-1)Q}{r}\}$. Therefore, the measurement will always output a multiple of $\frac{Q}{r}$.

$$\text{Output } b : \text{a uniformly random multiple of } \frac{Q}{r}.$$

This means there exists a random (unknown) $c \in \{0, \ldots, r-1\}$ such that $b = \frac{cQ}{r}$. or equivalently $\frac{b}{Q} = \frac{c}{r}$.

7. Find r from the above equality. How?

   - b, Q are known, c, r are unknown. We can rewrite $\frac{b}{Q} = \frac{b'}{Q'}$ with $b' \wedge Q' = 1$.
   - c is a random number in $\{0, \ldots, r-1\}$. This implies that $c \wedge r = 1$ with probability greater than $\Omega(\frac{1}{\log(\log(r))})$. When this happens, we necessarily have $c = b'$ and $r = Q'$.
   - Check that Q' is a period of f. If yes: done. If no, go back to step 1.

General case (Sketch) : r does not divide Q.

If we measure the first register, we obtain b such that $|\frac{b}{Q} - \frac{c}{r}| \leq \frac{1}{2Q}$ with high probability for some random c. If this is the case, $\frac{c}{r}$ is the only fraction with $c \wedge r = 1$ and $r \leq N$ such that $|\frac{b}{Q} - \frac{c}{r}| \leq \frac{1}{2Q}$ (proof omitted). This is because we chose Q such that $Q \geq N^2$.

If we indeed have $c \wedge r = 1$, which still happens with probability greater than $\Omega(\frac{1}{\log(\log(r))})$, we can use the continuous fraction method to find the unique fraction $\frac{c}{r}$ satisfying $|\frac{b}{Q} - \frac{c}{r}| \leq \frac{1}{2Q}$ from which we can get r.

DONE :)

Complexity of the period finding algorithm

We apply the above procedure until we find c such that $c \wedge r = 1$. This means we perform $O(\log(\log(r)))$ loops. Each loop makes 2 calls to QFT_Q and 1 call to O_f.

The running time is therefore $PFA_f \leq O\left(\log(\log(r))(QFT_Q + O_f)\right)$.

5.2.2 Complexity of Shor's algorithm

Shor's algorithm: $O(1)$ calls to PFA_f with $f(k) = x^k \mod N$ for some random x.

One can show that O_f can be calculated in $O(\log^2(N)\log(\log(N))\log(\log(\log(N))))$ using efficient squaring. (Perfect) QFT_Q is made in $O(Q^2) = O(\log^2(N))$.

From there, we conclude that the total running time of Shor's algorithm is

$$\begin{aligned} O\left(\log(\log(r)) * \left(\log^2(N) + \log^2(N)\log(\log(N))\log(\log(\log(N)))\right)\right) &\leq O\left(\log(n) * \left(n^2 + n^2\log(n)\log(\log(n))\right)\right) \\ &= O(n^2 polylog(n)). \end{aligned}$$

Bibliography

[BB84] Bennett and Brassard. Quantum cryptography: Public key distribution and coin tossing. in Proc. Of IEEE Inter. Conf. on Computer Systems and Signal Processing, Bangalore, Kartarna, (Institute of Electrical and Electronics Engineers, New York, 1984.

[Deu85] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London Ser. A*, A400:97–117, 1985.

[Fey82] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6&7):467–488, 1982.

[Gro97] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack, 1997.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *IEEE Symposium on Foundations of Computer Science*, pages 124–134, 1994.