Universiteit Leiden
Opleiding Informatica

Quantum Computing

Name: Liam Zwitser

Date: 24/1/2017

1st supervisor: J. M. De Graaf
2nd supervisor: A. H. Deutz

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Table of Contents

1 Introduction

Quantum Computing is a relatively young field of research, which started in the 1980s. I decided to pick this subject for my thesis because it is an interesting field with many potentially useful applications. It started with physicists such as Richard Feynmann wondering if it would maybe be possible to use some of the strange phenomena known from quantum mechanics to find faster methods of computation. It turned out this was the case for at least some traditionally difficult problems. Quantum Computing uses a phenomenon known as superposition to create *qubits*, which are a different sort of bit from the ones classical computers use.

In quantum mechanics, one can not know exactly both the location and velocity of a particle. This is known as the uncertainty principle. Until the moment one measures the state of the particle, there is a range of possible states one can find the particle in, each with their own probability. This property is exploited in the creation of qubits. Classical bits can only be in one of two defined states: 0 or 1, true or false. qubits on the other hand can, as long as they have not been measured, be in a whole range of states ranging from one extreme to another. This does not mean one can read more than one value when observing (a series of) qubits. When observed, a qubit collapses to one of its two base states. Because of this fact, it is not possible to read more than a single bit of information from each qubit. However, before observation, there are operations that one can do that do not collapse the qubit's state but have a known influence on the probability of each state to be measured when one observes the qubit later. So quantum algorithms can do operations on qubits without collapsing them. When one has applied all the desirable operations on the qubit, one measures it's state, and thus get the "output" of the quantum algorithm.

I will now give a brief discussion of some of the basic literature in the field. More recent developments will be discussed in chapter four.

1.1 Literature Discussion

For such a young research area it is surprising how much has been written on Quantum Computing during the last decades. There is a wide array of literature available on the subject, ranging from books to journal articles, for a variety of different audiences, from computer scientists to physicists. In this section I will give some examples of literature on the subject which the interested reader might want to look up to get more in-depth information about the (mathematical) background of quantum computing and the algorithms I describe in the next chapters.

There are a number of books available for people to learn the basic concepts of quantum computing. These include *Quantum Computing: a gentle introduction* by Eleanor G. Rieffel and Wolfgang H. Polak [1], *Quantum Computing for Computer Scientists* by Noson S. Yanofsky and Mirco A. Mannucci [2], *Quantum Computation and Quantum Information* by Michael A. Nielsen and Isaac L. Chuang [3], and *An Introduction to Quantum Computing Algorithms* by Arthur O. Pittenger [4]. All of these books are meant to explain the basics to the readers who want to start learning about quantum computing. They explain notation, necessary mathematical and physics background knowledge, (although most assume the reader is already familiar with basic linear algebra, as will I in this thesis), and a number of important quantum computing algorithms. Algorithms discussed in these books include those for factorizing large numbers,

finding database items, secure cryptography protocols, and more.

There is also a whole range of articles available. A long, thorough introduction mostly aimed at physicists can be found in *Quantum Computing* by Andrew Steane, a 1998 article published in the journal *Reports on Progress in Physics* [5]. Because it is aimed at physicists, who cannot be expected to have any background knowledge in computer science, it does not only explain quantum information theory and the physics underlying quantum computing, but also basic computer science concepts such as Turing machines. This, combined with the age of the article, might make it less interesting for people holding a degree in computer science, but readers with a physics degree might want to look it up. A much shorter (only 7 pages) paper meant for computer scientists is *Quantum Computing: an Introduction* written by T. Hey and published in *Computing Control Engineering Journal* [6]. It describes quantum complexity, basic structures such as gates and qubits, and some important algorithms.

Finally, there is also a large amount of papers written about various methods that scientists are trying in building quantum computers. Although I will not discuss how to build a physical quantum computer in this thesis because it is outside of the realm of computer science, I would like to point the interested reader to a few of these papers. There are many different ways in which quantum computers might be built in the future, and the wide variety of experiments and research papers reflects this. Some papers on the subject of building quantum computers that readers might be interested in include *Linear optical quantum computing with photonic qubits* by Pieter Kok et al [7], *A silicon-based nuclear spin quantum computer* by B. E. Kane (1997) [8] and *Quantum computing with realistically noisy devices* by E. Knill (2005). [9] As stated we will not be discussing these papers here because their subject is beyond the scope of this thesis.

The remainder of this thesis will focus on three things. Firstly, there will be an explanation of the basic concepts behind quantum computing: the formal notation, mathematical background and why quantum computers can be faster than "traditional" computers for certain applications. Secondly, I will dedicate two chapters to explaining, with examples, the workings of a few elementary quantum algorithms which have been influential in the development of this new field of study. I will first dedicate a chapter to the Deutsch and the Deutsch-Josza algorithms. Although they do not solve any important or even interesting problems, they were among the first to prove that quantum computing can sometimes be faster than any classical computer, and thereby helped to popularize the subject. The third chapter will be dedicated to Grover's algorithm, which promises to provide an actually useful speedup over classical computers by being able to search trough unstructured databases faster than any classical computer will ever be able to. Finally, and thirdly, the fourth and last chapter will focus on recent developments in quantum computing, and will feature a number of articles on a diverse range of subjects to give an overview of the promise contained within the field of quantum computing research. It will also serve as an overview of related literature which the interested reader might want to look up that is more modern or in-depth than the basic literature discussed above. After that chapter there will be a short conclusion to conclude the thesis.

We will now go to the first chapter, which will serve as an introduction to the basics of quantum computing.

2 Basic Concepts

In this chapter I will explain the basic building blocks of quantum computing the reader will need to be familiar with to understand the rest of the thesis. These include the notation of qubits and states and their superposition, how to calculate the probability of a superposition collapsing to a certain base state upon measurement, and more. After this section we will be ready to take a look at two elementary quantum computing algorithms, Deutsch's algorithm and the Deutsch-Jozsa algorithm, in the next chapter. I got (most of) the information in this chapter from a reader used in foothill college [22].

2.1 Notation of Qubits, States, and Superpositions

As in any academic field, it is important to know the notations used in quantum computing before one can make a honest effort to learn to understand it. We will start with qubits, since these are the most basic building blocks of a quantum system. Normal bits can have one of two states, usually denoted as 0 or 1. Qubits can be in any superposition of their base states, which means that each base state (eg. 0 and 1) has associated with it a probability that it will collapse into that state upon being measured. A single qubit with two base states 0 and 1 is typically written as $\alpha|0\rangle + \beta|1\rangle$ where α and β are complex numbers. These indicate the probabilities of finding the qubit in their respective base states upon measurement. This probability is $|\alpha|^2$ and $|\beta|^2$ for base states 0 and 1 respectively. Because of this, there is the obvious requirement that $|\alpha|^2 + |\beta|^2$ must be 1, because after measurement it is impossible for the system to be in any other state then one of it's base states. These complex numbers here denoted as $|\alpha$ and $|\beta$ are called the *amplitudes* or *probability amplitudes* of the qubit. Theoretically, the quantum systems used could also have more then two base states, such as 0, 1, and 2. Such a "unit" would be written down as $\alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$, and the probabilities for collapsing into a certain state after measurement would be calculated in the same way as for a qubit.

It should be noted that quantum states can actually be written down in four different ways. One of these is as x^n-dimensional complex vectors where n is the number of qubits and x the number of base states per qubit. When we do this for n = 1, $|0\rangle$ for example becomes $\begin{pmatrix}1\\0\end{pmatrix}$ and $|1\rangle$ becomes $\begin{pmatrix}0\\1\end{pmatrix}$. In general, $\alpha|0\rangle + \beta|1\rangle = \begin{pmatrix}\alpha\\\beta\end{pmatrix}$. As another example, suppose we have two qubits in our system, which means there are four possible base states. Then we write them as $\begin{pmatrix}1\\0\\0\\0\end{pmatrix}$, $\begin{pmatrix}0\\1\\0\\0\end{pmatrix}$, $\begin{pmatrix}0\\0\\1\\0\end{pmatrix}$ and $\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$. Quantum gates are then expressed as matrices which act on these vectors. We have now seen two ways to describe qubits, as a final example I will show all four ways to write down all four base states of a 2-qubit system. The following four notations are equivalent to each other:

$$|0\rangle|0\rangle = |00\rangle = |0\rangle^2 = \begin{pmatrix}1\\0\\0\\0\end{pmatrix},\ |0\rangle|1\rangle = |01\rangle = |1\rangle^2 = \begin{pmatrix}0\\1\\0\\0\end{pmatrix},$$

$$|1\rangle|0\rangle = |10\rangle = |2\rangle^2 = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \text{ and } |1\rangle|1\rangle = |11\rangle = |3\rangle^2 = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}.$$

$|0\rangle$, $|1\rangle$, $|2\rangle$, and $|3\rangle$ is the customary notation however, and we will mostly use it in this thesis.

A quantum computing system consisting of only one qubit would not be able to do much useful work. Because of this, systems are usually described and made with tens or hundreds of qubits. Obviously, the number of base states in a multi-qubit system is equal to x^n, where x is the number of base states each qubit can be in and n is the number of qubits making up the system. A superposition of such a system again consists of all the base states plus complex numbers describing the probability to be measured for each of them. To keep our example compact, for a 2 qubit system where each of the qubits can be $|0\rangle$ or $|1\rangle$, this gives us $\alpha \begin{pmatrix}1\\0\\0\\0\end{pmatrix} + \beta \begin{pmatrix}0\\1\\0\\0\end{pmatrix} + \gamma \begin{pmatrix}0\\0\\1\\0\end{pmatrix} + \delta \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$. It should be noted that current real-life quantum systems are still very experimental in nature and as such do not usually have more then about 10 qubits. This is only because of our limited current understanding of how to implement quantum computing, however, and not because such a low number of qubits is enough to do any useful work.

2.2 How to View Quantum Computations

I will now explain how to describe a sequence of quantum operations to be executed on a system with a certain number of qubits. One of the ways this can be done is with quantum circuits. A quantum circuit is nothing more then any number of individual quantum operations applied to a system in sequence. Measuring the state of the system can also be a part of a quantum circuit. These circuits make it easier for people to understand quantum computations and quantum algorithms. A quantum algorithm can be implemented as a quantum circuit which is implemented as a number of gates all of which do a single part of the algorithm. The gates are analogous to the classical logic gates from which classic circuits are build.

There are a lot of different gates from which a quantum circuit can be build. Some basic gates acting on single qubits include the Hadamard gate and the Pauli gates. The Hadamard gate transforms single qubits which are in one their two base states in such a way that afterward there is an equal chance of measuring both base states when measuring the qubit. Incidentally, applying the Hadamard gate two times in a row to a qubit in one of it's base states is equal to applying the identity matrix, in other words at output the qubit will be in the exact same state it was in at the moment it went into the first of the two Hadamard gates. In quantum circuits, a Hadamard gate is denoted by —$\boxed{H}$—. It transforms a base state $|0\rangle$ into $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ and $|1\rangle$ into $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$. The former state is often denoted as $|+\rangle$ and the latter as $|-\rangle$. The matrix associated with a Hadamard gate is $\begin{pmatrix}1 & 1\\1 & -1\end{pmatrix}$. When one applies a Hadamard gate to a system with 2 qubits ($H^{\otimes 2}$), the following matrix expresses the result: $\begin{pmatrix}1 & 1 & 1 & 1\\1 & -1 & 1 & -1\\1 & 1 & -1 & -1\\1 & -1 & -1 & 1\end{pmatrix}$

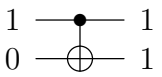
Using Hadamard gates, it is possible, with only n gates, to bring n qubits into a superposition in which each of the 2^n base states the qubits can be in has the same probability of being found upon measurement. One can use this state as an input to the quantum evaluation of f which is akin to applying f to all possible values of x at the same time. This is called quantum parallelism and it differs from the classical variant as only one circuit is used to do all this work. In classical parallelism, if we want to run n operations in parallel, we need n identical circuits running at the same time.

Now we will see what the Pauli gates —$\boxed{X}$—, —$\boxed{Y}$—, and —$\boxed{Z}$— have as output when given either $|0\rangle$

or $|1\rangle$ as input. I will also give the matrices which represent these gates. The Pauli-X gate transforms $|0\rangle$ into $|1\rangle$ and $|0\rangle$ into $|1\rangle$. It's matrix, called the Pauli matrix, is $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. The Pauli-Y gate maps $|0\rangle$ to $i|1\rangle$ and $|1\rangle$ to $-i|0\rangle$. The Pauli-Y matrix which represents it is $\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$. Finally, the Pauli-Z gate maps $|0\rangle$ to itself (ie it is unaffected) and $|1\rangle$ to $-|1\rangle$. This leads to it being known as the phase-flip gate. The Pauli-Z matrix is $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. We have now seen how the three Pauli gates affect the base states of a qubit and what their matrices are.

The last one-qubit state we have to look at is the phase gate, R_ϕ. This is actually a collection of gates, which differ in the value of ϕ. In general, the matrix describing the transformation they do is $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$.

Continuing our review of basic quantum gates, there are also some fundamental gates which act on more than one qubit. One such gate is the controlled not or CNOT gate, written in a circuit as —$\boxed{CNOT}$— or as



It takes two qubits as input one of which serves as a "switch" of sorts. If this switch qubit is in state $|1\rangle$ the other qubit is flipped as by the Pauli x gate. This means that the CNOT gate transforms $|00\rangle$ into $|00\rangle$, $|01\rangle$ into $|01\rangle$, $|10\rangle$ into $|11\rangle$ and finally $|11\rangle$ into $|10\rangle$. It's matrix representation is $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

As with classical logic gates, there is also a universal set of gates for quantum logic. A universal set of gates is any set/group of gates from which all possible circuits can be built. In the case of quantum computing, this set consists of all possible 1-qubit quantum gates plus the CNOT gate which we have seen earlier. This means that the universal set, unlike the one for classical digital logic, is infinite in size; after all, there is an infinite number of possible 1-qubit quantum circuits which one can make. Obviously, it is impossible to construct any computer implementing this universal set, but quantum computing is saved by the fact that there are finite sets of gates which can approximate any quantum circuit, also called universal approximation. The theorem which tells us this is possible is called the Solovay - Kitaev Theorem.

It has been proven that with the combination of CNOT gate(s), Hadamard gates, and the $R_{\frac{\pi}{4}}$ gate it is possible to approximate the functionality of *any* quantum gate that can be constructed to an arbitrary degree. This is not a universal set however because not all possible quantum circuits can be made exactly. "To an arbitrary degree" here means that while we can never *perfectly* approximate the functionality of any possible quantum gate with this set (much like one can not make a mathematically perfect sphere in reality), we can construct ever more accurate approximations until the deviation from the theoretical, perfect gate approaches zero. In mathematical terms, the deviation is called "0 + epsilon". There are other sets which can also be used for the same purpose, but I think it redundant to list all of them here.

Before we are ready to move on to our first quantum algorithms, we still have to define some mathematical operations and symbols which will be used extensively in the upcoming chapters. It is important to keep in mind that quantum states are often represented as vectors and the effect a quantum circuit has on these vectors is usually expressed using a matrix. This is why a lot of the basic math we will see in this thesis is meant to deal with matrices and vectors, including the operators and definitions I will explain here. During my research I found out that there are two operators which are frequently used which I did

not understand at first. I will now elaborate on these two operators. They are the $\bigotimes$ operator and the combination of $\rangle$ and $\langle$ directly next to each other. $\bigotimes$ is the Kronecker product of two matrices/vectors. The Kronecker product of two arbitrary sized matrices is a block matrix, as in the following example:

$$\begin{pmatrix} 1 & 3 \\ 5 & -2 \end{pmatrix} \bigotimes \begin{pmatrix} 1 & -1 \\ 2 & 3 \end{pmatrix} = \begin{pmatrix} 1*1 & 1*-1 & 3*1 & 3*-1 \\ 1*2 & 1*3 & 3*2 & 3*3 \\ 5*1 & 5*-1 & -2*1 & -2*-1 \\ 5*2 & 5*3 & -2*2 & -2*3 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 3 & -3 \\ 2 & 3 & 6 & 9 \\ 5 & -5 & -2 & 2 \\ 10 & 15 & -4 & -6 \end{pmatrix}$$

$\rangle\langle$ operator can be best defined with an example on two small vectors from a system with two qubits:

$$|01\rangle\langle 11| = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

In the next chapter I will explain some very basic quantum algorithms which demonstrate that at least in some cases quantum computers have the potential to be faster than any classical computing system which can be made to solve the same problem.

3 Some Basic Algorithms

In this chapter we will look at a number of important quantum algorithms. These algorithms are considered so important because they were used to show that for some problems the best possible quantum algorithm will *always* be faster than the best possible classical algorithm. As such, they demonstrated the usefulness of quantum algorithms. We will begin by explaining Deutsch's algorithm and then move on to Deutsch-Jozsa.

3.1 Deutsch's Algorithm

When quantum computation was first introduced, it was unclear whether or not there really were computational problems for which a quantum system would *always* be faster then the fastest possible classical solution. It was soon clear that a quantum computer can efficiently simulate a classical computer, which was to be expected because physics tells us that at the base level everything in the world around us follows the rules of quantum mechanics. On it's own however, this does not provide any good reasons for investing a lot of resources into developing quantum systems. For this to be worth the effort, a quantum computer should be able to work faster than a classical one, not only just as fast.

Deutsch's algorithm was the first algorithm to prove that this is possible. It uses a simple quantum circuit employing a combination of several hadamard gates to determine a property of a function which requires two evaluations of that function to determine using classical computations with only one quantum evaluation of that function. The simple function f mentioned is a function whose domain and co-domain both consist of only a single bit. The value, as always, of these bits is either 0 or 1. The question Deutsch's algorithm answers is whether the function f is constant or balanced. If f is constant, $f(0)$ gives the same output as $f(1)$. Otherwise, $f(0) = 0$ and $f(1) = 1$ or vice-versa. To solve this problem efficiently, the algorithm uses the quantum parallelism explained in the introduction.

The problem with quantum parallelism is the collapse of the superposition upon measurement. As discussed before, this property of quantum mechanics ensures that only a single base state will eventually be seen, thus the information embedded in the superposition about all the other base states is "lost". The power of Deutsch's algorithm lies in the fact that it combines the simultaneous computation of $f(x)$ for all legal values of x with a way to obtain information about the aforementioned global property of the function f (namely if it is balanced or constant) from the final measurement in spite of the unavoidable collapse of the superposition.

Deutschs algorithm consists of three basic parts: The two Hadamard gates which are used to bring the qubits on both input lines into what is called a maximally mixed superposition (1/2 chance of measuring $|0\rangle$ and 1/2 chance of measuring $|1\rangle$), an *oracle*, which is a black box which calculates $f(x)$ for us for an input x, and finally a trick which is used to get information from one of the registers into the other so we can use just one measurement to obtain all the data from the processing of both basis states by the oracle. Ultimately, the oracle manipulates our two maximally mixed superpositions in such a way that if we put one of the two output lines through a final Hadamard gate, and then measure only that one output line, we know that $f(x)$ is balanced if we measure $|1\rangle$ and constant if we measure $|0\rangle$.

The Circuit and Mathematics of Deutsch's Algorithm
Before continuing with the next algorithm I first want to show the quantum circuit of Deutschs algorithm and step through the calculations which show that we can indeed see whether $f(x)$ is constant or balanced with just one evaluation of the quantum oracle U_f. The quantum circuit implementing Deutschs algorithm

is the following one:

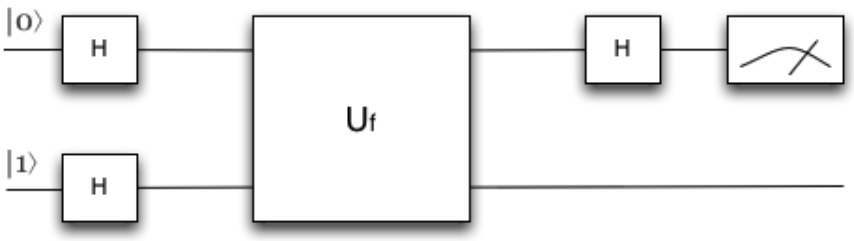


Figure 1: Circuit Which Implements Deutschs Algorithm

In this, the boxes with a H in them stand for Hadamard gates, the measuring device represents a measurement, and the big U_f box in the middle is the oracle. As we can see, the circuit takes $|0\rangle$ and $|1\rangle$ as input as expected. The measurement at the end of the circuit will tell us that $f(x)$ is constant if we measure $|0\rangle$ or balanced if we measure $|1\rangle$. Lets walk through the calculations which prove that the circuit works in this way. I found the explanation found in [22] Loceff (2015: 362-367) so useful that I will use it here. To understand the algorithm, we need to work out what the output of the oracle will be for different possible inputs. First we define an operator $\oplus$: x $\oplus$ y = (x + y) Now, we start by feeding a base state $x = 0$ and base state $y = 1$ into the oracle and we find that by definition the output of U_f for this input is

$$U_f(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle.$$

We will now see what happens when we put a superposition as one of the inputs for U_f, then what happens when we do so for both inputs of U_f as in the circuit, and then finally how we can use one last Hadamard gate and one measurement to tell whether $f(x)$ is constant or balanced.

As the next step in our calculations, we still put the base state x into the top input of U_f, but we now let $|-\rangle$ be the input of the bottom line going into U_f in the circuit. For the following calculations we can use the fact that U_f is a linear operator. The output of U_f in this situation is:

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

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

$$= \frac{|x\rangle|f(x)\rangle - |x\rangle|\overline{f(x)}\rangle}{\sqrt{2}} = |x\rangle(\frac{|f(x)\rangle - |\overline{f(x)}\rangle}{\sqrt{2}})$$

Since there are only two possible outcomes when evaluating $f(x)$, namely 0 or 1, we can see that

$$U_f(|x\rangle|-\rangle) = |x\rangle \begin{cases} \frac{|0\rangle-|1\rangle}{\sqrt{2}}, & \text{when } f(x) = 0 \text{ or} \\ \frac{|1\rangle-|0\rangle}{\sqrt{2}}, & \text{when } f(x) = 1 \end{cases} \tag{1}$$

which equals

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

Is it at this time that the kickback trick is used. Because the term $(-1)^{f(x)}$ is a scalar, it can be moved to the left side of the equation without changing it's algebraic meaning. The equation is transformed as follows:

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

The information about $f(x)$ which is encoded in the bottom output line of U_f after this input is thus transferred entirely to the top output line of U_f as a part of the term $(-1)^{f(x)}$. This is also why at the end of the circuit when we want to draw our conclusion on whether $f(x)$ is balanced or constant we need only measure the top data line and ignore the bottom one. If we measure both lines, we obtain the same information about $f(x)$ but no additional, extra information is obtained, so this is considered to be useless extra work.

We can now proceed with the third part of our calculations, which entails calculating the output of the oracle U_f when both of it's input lines have superpositions on them. The top input gets superposition $|+\rangle$ and the bottom input line is set to the superposition $|-\rangle$ like in the last part. The $|+\rangle$ in the top input line is what enables U_f to evaluate both $f(0)$ and $f(1)$ simultaneously. Because of the kickback from the bottom to the top line which I explained earlier, after this query to U_f information about the outcome of *both* $f(0)$ and $f(1)$ will be in the top register, ready for us to measure after the final Hadamard gate. I will now show the calculations which demonstrate how the oracle processes this input:

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

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

After these calculations are complete we are ready to use the final Hadamard gate, pictured in the top right part of the circuit, to prepare the top output line for measurement. After measuring this line, we will have the knowledge we were seeking about $f(x)$. When we measure a base state $|0\rangle$, f is constant and when we measure a base state $|1\rangle$, f is balanced. This is because when $f(0)$ equals $f(1)$, the two enumerators in the last result above have the same sign, whereas they have different signs when f is balanced. As promised, we needed only one query to U_f. We have thus proven there is a real speedup compared to the best possible classical algorithm (which is to simply check both inputs and see if the output is the same, if one wants the algorithm to give the correct answer with certainty).

3.2 Deutsch-Jozsa Algorithm

The algorithm described in the last section is a special case of a more general algorithm known as Deutsch-Jozsa. This algorithm also evaluates whether a function $f(x)$ is constant or balanced, but the difference is that the domain of $f(x)$ can now be bigger than one qubit, in other words it can take more than one qubit as input. Being limited to using classical algorithms, it is easy to see this problem requires at least $2^n/2+1$ evaluations of $f(x)$. After all, it is possible to get back exactly $2^n/2$ zeroes (or ones) before getting

the decisive result. This last result would tell us whether the function is balanced; if after $2^n/2+1$ queries we still have only seen all zeroes or all ones, we know for sure that $f(x)$ is constant.

Much like Deutschs algorithm described above, the Deutsch-Jozsa algorithm can determine this property of $f(x)$ with certainty with only one query to the oracle. First, we prepare a maximally mixed superposition of n qubits by using a n-dimensional Hadamard gate. This will be the query register input for the oracle. We also prepare a 1 qubit register maximally mixed for U_f to store the answer in. U_f applies $f(x)$ to the superposition, resulting in all of the possible inputs being evaluated at the same time as in the previous algorithm. We can then use a final n-dimensional Hadamard gate on our query register to find out whether $f(x)$ is constant or balanced. If it is constant we will obtain only zeroes from measurement, otherwise we will measure at least one qubit with value one. Since this took only one evaluation of $f(x)$, it is substantially faster than the classical method described above.

3.3 The Circuit and Mathematics of Deutsch-Josza Algorithm

As said before, Deutsch-Josza is a generalization of Deutsch's algorithm to n qubits. This can be clearly seen in the circuit implementing Deutsch-Josza, pictured below:

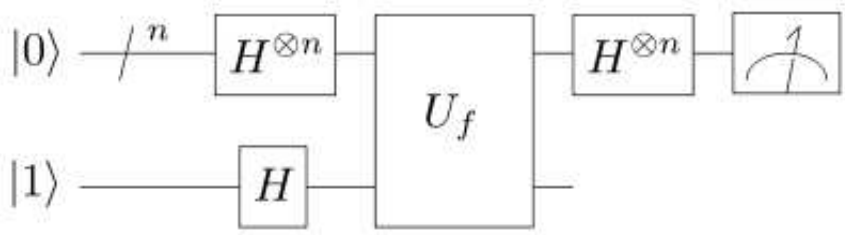


Figure 2: Circuit Which Implements Deutsch-Josza Algorithm

As we can see, the only difference is that the Hadamard gates on the top input line now works on n qubits instead of just one. This does however have some implications for the mathematics which make the algorithm tick, which we will take a look at now. As we did for Deutsch's algorithm, we will again split the discussion into a number of steps. First of all, the output of the U_f in this circuit on two base state inputs is (by definition of U_f) given by the equation

$$U_f(|x\rangle^n|y\rangle) = |x\rangle^n|y \oplus f(x)\rangle).$$

The next step would be to show what happens when we put a $|x\rangle^n$ in the top input and superposition state into the bottom input of U_f, but since both the result and the calculations are exactly identical to Deutsch's algorithm for this step, I skip it here. Suffices to say that in this situation the output of U_f is described by

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

.

The next step where we put superpositions into both inputs of U_f is where things start to differ substantially from the calculations we did for Deutsch's algorithm. The Hadamard gate on the top input line of U_f takes the n qubits it takes as input to $|0\rangle^n$, which is a superposition in which all 2^n base states have the same chance of being found upon measurement. In formula form: $\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}|y\rangle^n$. This will be on the

top input of U_f with the bottom input consisting of the superposition $|-\rangle$. In the following calculations, y indicates all the 2^n different base states which the n qubits can be in, eg $|0\rangle^n$, $|1\rangle^n$, etc. So the sum from 0 to $2^n - 1$ represents all values which the n qubits can represent. As in the previous algorithm, the crux is that U_f evaluates $f(x)$ for all possible x in one query. The output for U_f for these inputs can be calculated as follows:

$$U_f(|0\rangle^n|-\rangle) = U_f((\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}|y\rangle^n)|-\rangle)$$

$$= \frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}U_f(|y\rangle^n|-\rangle)$$

$$= \frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}((-1)^{f(y)}|y\rangle^n)|-\rangle$$

$$= (\frac{1}{\sqrt{2^n}}\sum_{y=0}^{2^n-1}(-1)^{f(y)}|y\rangle^n)|-\rangle.$$

In other words, we have applied U_f to all of the possible base states at once and obtained our expected result:

$$U_f(|x\rangle^n|-\rangle) = |x\rangle^n(-1)^{f(x)}(\frac{|0\rangle - |1\rangle}{\sqrt{2}}).$$

Now the only thing left to do is to show that the Hadamard gate on the right side of U_f together with the measurement will tell us if $f(x)$ is balanced or constant. To provide insight into this, I will do this final step assuming we are running the algorithm with $n = 2$ qubits, which makes for 4 states ranging from $|0\rangle$ to $|3\rangle$. We now get the following calculation:

$$\sum_{y=0}^{3}(-1)^{f(y)}H^{\otimes 2}|y\rangle$$

$$= (-1)^{f(0)}H^{\otimes 2}|0\rangle|0\rangle + (-1)^{f(1)}H^{\otimes 2}|0\rangle|1\rangle + (-1)^{f(2)}H^{\otimes 2}|1\rangle|0\rangle + (-1)^{f(3)}H^{\otimes 2}|1\rangle|1\rangle$$

$$= (-1)^{f(0)}H|0\rangle H|0\rangle + (-1)^{f(1)}H|0\rangle H|1\rangle + (-1)^{f(2)}H|1\rangle H|0\rangle + (-1)^{f(3)}H|1\rangle H|1\rangle$$

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

$$+(-1)^{f(2)}(\frac{|0\rangle - |1\rangle}{\sqrt{2}})(\frac{|0\rangle + |1\rangle}{\sqrt{2}}) + (-1)^{f(3)}(\frac{|0\rangle - |1\rangle}{\sqrt{2}})(\frac{|0\rangle - |1\rangle}{\sqrt{2}})$$

$$= (-1)^{f(0)}(\frac{|0\rangle|0\rangle + |0\rangle|1\rangle + |1\rangle|0\rangle + |1\rangle|1\rangle}{2}) + (-1)^{f(1)}(\frac{|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle}{2})$$

$$+(-1)^{f(2)}(\frac{|0\rangle|0\rangle + |0\rangle|1\rangle - |1\rangle|0\rangle - |1\rangle|1\rangle}{2}) + (-1)^{f(3)}(\frac{|0\rangle|0\rangle - |0\rangle|1\rangle - |1\rangle|0\rangle + |1\rangle|1\rangle}{2})$$

$$= \frac{1}{2}((-1)^{f(0)} + (-1)^{f(1)} + (-1)^{f(2)} + (-1)^{f(3)})|0\rangle|0\rangle$$

$$+\frac{1}{2}((-1)^{f(0)} - (-1)^{f(1)} + (-1)^{f(2)} - (-1)^{f(3)})|0\rangle|1\rangle$$

$$+\frac{1}{2}((-1)^{f(0)} + (-1)^{f(1)} - (-1)^{f(2)} - (-1)^{f(3)})|1\rangle|0\rangle$$

$$+\frac{1}{2}((-1)^{f(0)} - (-1)^{f(1)} - (-1)^{f(2)} + (-1)^{f(3)})|1\rangle|1\rangle$$

We only look at one base state, the one where the qubits are both $|0\rangle$. As seen from the last calculations, if f is constant, signs in the sum $\sum_{y=0}^{3}(-1)^{f(y)}H^{\otimes 2}|y\rangle$ are either $+$ or $-$ meaning we end up with a probability of $1^2 = 1$ or $-1^2 = 1$ to measure this base state. This automatically makes the probability to measure any other base state 0, because chances add up to 1. If f is balanced on the other hand, this means that the chance of measuring $|0\rangle|0\rangle$ in the final measurement is exactly 0, because the coefficient of $|0\rangle|0\rangle$ will now always be 0. With this observation we have proven that the Deutsch-Josza algorithm does indeed provide the answer if $f(x)$ is balanced or constant with a single query to U_f. In our 2 qubit example, if we measure $|0\rangle|0\rangle$ we know for sure f is constant, otherwise we are sure it is balanced. The calculations and results are similar if there are more then 2 qubits, but I will not show these calculations here. When there are more then 2 qubits in the system, the result can still be obtained with just one final measurement.

We now understand the mathematics behind both Deutsch's algorithm and the Deutsch-Josza algorithm. Unfortunately, these two algorithms do not solve any particularly important or interesting problems. They are useful however, because the concepts like quantum parallelism which make them work are the same ones which allow more important algorithms to function. In the following chapter I will discuss an algorithm called "Grover's algorithm" which is used to quickly find elements in an unstructured database of information.

4 Grover's Algorithm

As stated before, there would be little incentive to develop quantum computing further if all it could do was simulate classical computing. If this were the case there would be no decrease in the number of calculations needed to solve a problem, and actually building a quantum computer requires a lot of work; even after decades of academic research and projects, the biggest experimental quantum computer built so far has about 10 qubits.

Luckily quantum computing solves some problems faster than any classical method, as we have already proven in the last chapter. Those two algorithms, however, did not have much practical significance. The problem they solve, to see if a function $f(x)$ is balanced or constant, does not have any known practical applications. In this chapter, we will take a look at a more useful quantum algorithm, *Grover's algorithm*. The speedup provided by Grover's algorithm is not that big in terms of complexity, but its application, searching through unstructured data sets, is so common and important that it is still very interesting. It should be noted that, like some other quantum algorithms, Grover's algorithm is not deterministic, that is to say there is no guarantee that the solution we obtain by running the algorithm is correct. We can only say it is correct with a certain probability. However, this probability is generally high enough to make the algorithm and the outcome useful, and to obtain a virtually guaranteed-to-be-correct result with an acceptable number of repetitions.

Like the algorithms in the last chapter, this one also uses a quantum oracle. (Nielsen- Chuang 2010: 248) Again, we do not know how this oracle works internally, but we only care about its input and output, which we do know by definition. Similarly, we also don't care about any qubits or quantum calculations it might use internally when analyzing the overall algorithm. I will now explain the information about the oracle that is relevant for understanding the algorithm. As before, I will first explain the algorithm in words, before showing the math to prove everything works.

If our table contains N elements, and we want to find a certain element then searching indexes instead of the elements allows us to define a function $f(x)$ which corresponds to the search problem. If the search space has N elements, the index numbers go from 0 to $N-1$. The function takes an index number x and returns 1 if the element associated with that index number is (one of) the element(s) we are searching for, and 0 otherwise. Calculating $f(x)$ for a given x is the job of the oracle. In other words, the oracle can recognize solutions to the search problem (Nielsen-Chuang 2010: 249). In formula form, $f(x)$ is defined as

$$f(x) = \begin{cases} 1 & \text{when } x \text{ is the index of a desired element, or} \\ 0 & \text{otherwise} \end{cases}$$

I will now explain how Grover's algorithm works mathematically. First, n qubits, with n chosen such that 2^n equals the number of elements in the search space, are created and, using Hadamard gates, are brought into a superposition where all the base states have the same chance of being measured, just like the first step in the Deutsch and Deutsch-Josza algorithms. Before the first iteration we take the average of the amplitudes of all base states, which at this point is simply the amplitude of any of them as the Hadamard transform has made all of them equal. We then apply a certain number of so-called *Grover iterations*. Before we look at what happens exactly in each iteration, we should first note that there is an optimal number of iterations depending on the number of elements we are searching through.

It is known that the optimal number of Grover iterations for a database with N elements is approximately $\frac{\pi}{4}\sqrt{N}$. After this number of iterations, we have a very high chance of finding the right answer. It will never reach 100 percent, because Grover's algorithm is not deterministic, but it will usually be over 95

percent at the moment of measure. However, we should not do too many iterations. Until approximately $\sqrt{N}$ iterations the chance of finding the right element increases. Afterwards however, this chance starts to decrease again with more iterations. The same optimal number of iterations later, it starts to increase again, and this goes on in a never- ending cycle until one measures. This is because the amplitude of the state you want to measure, which governs its choice to be measured, moves over the unit circle with every iteration. It should also be noted there can be multiple valid elements in the database, in which case all of their amplitudes increase and decrease equally with every iteration.

Every iteration starts with a call to the oracle. The oracle leaves all amplitudes the same, except for the amplitude belonging to the state associated with the element we are looking for. It multiplies this amplitude with -1. After the oracle has done it's job, the amplitudes of all base states are updated in the following way: the new amplitudes become the previously calculated average plus the difference between that average and the current amplitude. It is obvious that this means that the amplitude of the base state associated with the sought-after element is now (much) larger than that of all the states we are not looking for. This means that the probability that this base state will be measured right now is also higher than for any of the other base states. The following picture shows what happens :

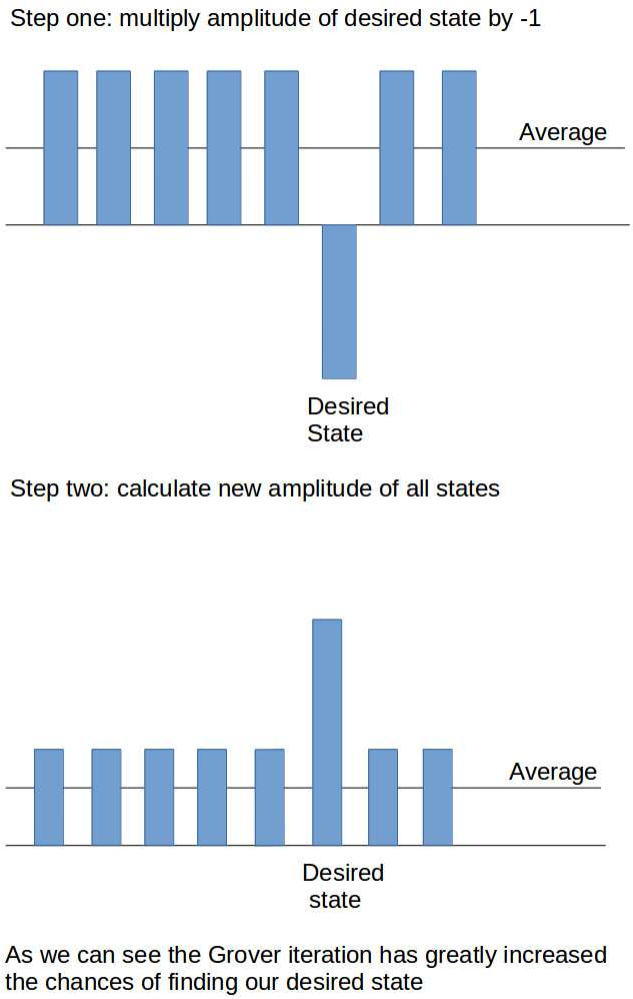


Figure 2: Inversion

After each iteration, we again calculate the new average amplitude to use for the calculations in the next iteration as described above. Calculating the current average amplitude of the states is in itself NOT a part of the Grover iterations. We now repeat the process: calculate the average, call the oracle, update the amplitudes etc. In formula form, the Grover Iteration G is $(H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n})O$ where O is the oracle. The following picture gives a visual summary of what we have just discussed:

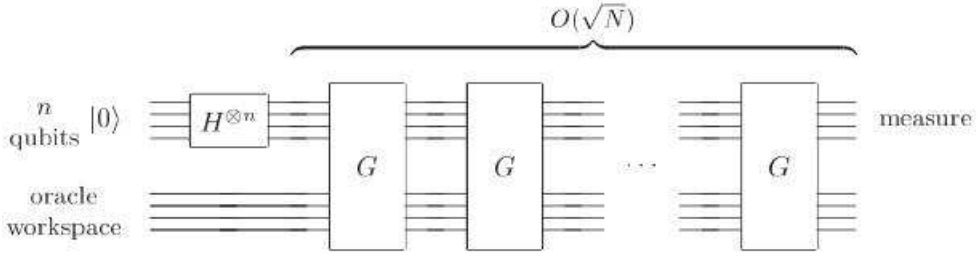


Figure 3: Circuit which Implements Grover's Algorithm

Instead of demonstrating the math for the whole algorithm, I will do a calculation to demonstrate a small but crucial part of it: I will show that two statements about the algorithm are correct for $n = 3$ qubits (which gives us a search space of $2^3 = 8$ elements). Hopefully, this will give some insight into the mathematical workings of the algorithm. The interested reader can find more detailed and complete calculations in the *quantum computing and quantum information* book [3]. Our 3 qubits have already been sent through a Hadamard gate in an operation denoted by ϕ: $H^{\otimes 3}|0\rangle = \sqrt{\frac{1}{8}}\sum_{k=0}^{2^3-1}|k\rangle^3$. The first thing I want to prove for $n = 3$, which I will number as 1, is this:

$$H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n} = 2|\psi\rangle\langle\psi| - I \qquad (1)$$

Let us start with filling in what we know for $n = 3$:

$$H^{\otimes 3}(2|0\rangle\langle 0| - I)H^{\otimes 3}$$

$$= \frac{1}{2^{\frac{3}{2}}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}\left(\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}\right.$$

$$\left. - \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}\right)\frac{1}{2^{\frac{3}{2}}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}$$

$$= \frac{1}{4}\begin{pmatrix} -3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -3 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -3 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -3 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & -3 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & -3 \end{pmatrix}$$

Now we fill in $n = 3$ for the right side of the equation, in which

$2|\psi\rangle\langle\psi| - I = 2(\frac{1}{2\sqrt{(2)}}\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}\frac{1}{2\sqrt{(2)}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}) - I$, which should be the same:

$$2|\psi\rangle\langle\psi| - I = 2\left(\frac{1}{8}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}\right) - I$$

$$= \left(\frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}\right)$$

$$= \frac{1}{4}\begin{pmatrix} -3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -3 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -3 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -3 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & -3 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & -3 \end{pmatrix}$$

As we can see, the equation holds true: the terms on the left hand side and the terms on the right hand side are one and the same. With this we have demonstrated that statement number one is correct.

There is a second thing that I want to prove, again for the case that $n = 3$. This is that the following holds true: $2(|\psi\rangle\langle\psi| - I)\sum_{k=0}^{2^n-1}\alpha_k|k\rangle^n = \sum_{k=0}^{2^n-1}(-\alpha_k + 2\langle\alpha\rangle)|k\rangle^n$ (2).
$\sum_{k=0}^{7}\alpha_k|k\rangle^3$ here expands as follows:

$$\alpha_0\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} + \alpha_1\begin{pmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{pmatrix} + \alpha_2\begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{pmatrix} + \alpha_3\begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{pmatrix} + \alpha_4\begin{pmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{pmatrix} + \alpha_5\begin{pmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{pmatrix} + \alpha_6\begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix} + \alpha_7\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}\alpha_0\\\alpha_1\\\alpha_2\\\alpha_3\\\alpha_4\\\alpha_5\\\alpha_6\\\alpha_7\end{pmatrix}$$

Let us now see what happens when we apply $2(|\psi\rangle\langle\psi| - I)$ (the very first matrix in the following calculations) to $\sum_{k=0}^{7}\alpha_k|k\rangle^n$:

$$\frac{1}{4}\begin{pmatrix}-3&1&1&1&1&1&1&1\\1&-3&1&1&1&1&1&1\\1&1&-3&1&1&1&1&1\\1&1&1&-3&1&1&1&1\\1&1&1&1&-3&1&1&1\\1&1&1&1&1&-3&1&1\\1&1&1&1&1&1&-3&1\\1&1&1&1&1&1&1&-3\end{pmatrix}\begin{pmatrix}\alpha_0\\\alpha_1\\\alpha_2\\\alpha_3\\\alpha_4\\\alpha_5\\\alpha_6\\\alpha_7\end{pmatrix}$$

$$= \frac{1}{4}\begin{pmatrix}-3\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7\\\alpha_0-3\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7\\\alpha_0+\alpha_1-3\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7\\\alpha_0+\alpha_1+\alpha_2-\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7\\\alpha_0+\alpha_1+\alpha_2+\alpha_3-3\alpha_4+\alpha_5+\alpha_6+\alpha_7\\\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4-3\alpha_5+\alpha_6+\alpha_7\\\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5-3\alpha_6+\alpha_7\\\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6-3\alpha_7\end{pmatrix}$$

Now we must show this is equal to the right-hand side of equation number one (α here is defined as the current average amplitude of all base states):

$$\sum_{k=0}^{7}[-\alpha_k + 2\langle\alpha\rangle]|k\rangle^3$$

=

$$(-\alpha_0 + 2\langle\alpha\rangle)\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} + (-\alpha_1 + 2\langle\alpha\rangle)\begin{pmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{pmatrix} + (-\alpha_2 + 2\langle\alpha\rangle)\begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{pmatrix} + (-\alpha_3 + 2\langle\alpha\rangle)\begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{pmatrix}$$

$$+(-\alpha_4+2\langle\alpha\rangle)\begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{pmatrix}+(-\alpha_5+2\langle\alpha\rangle)\begin{pmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{pmatrix}+(-\alpha_6+2\langle\alpha\rangle)\begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix}+(-\alpha_7+2\langle\alpha\rangle)\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix}$$

$$=\begin{pmatrix}-\alpha_0+2\langle\alpha\rangle\\-\alpha_1+2\langle\alpha\rangle\\-\alpha_2+2\langle\alpha\rangle\\-\alpha_3+2\langle\alpha\rangle\\-\alpha_4+2\langle\alpha\rangle\\-\alpha_5+2\langle\alpha\rangle\\-\alpha_6+2\langle\alpha\rangle\\-\alpha_7+2\langle\alpha\rangle\end{pmatrix}=\begin{pmatrix}-\alpha_0+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_1+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_2+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_3+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_4+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_5+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_6+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\\-\alpha_7+\frac{1}{4}(\alpha_0+\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6+\alpha_7)\end{pmatrix}$$

The end results of both of these calculations are equivalent to each other, meaning that we have shown that both statements we wanted to prove are valid. To conclude the mathematical examples related to Grover, I will now show what happens when you apply the oracle and then a Grover iteration G, again assuming that n = 3. This will show how inversion about the mean works mathematically. In doing so, it will demonstrate how the probability of being measured increases for the desired element during the initial iterations, and also how it decreases again afterwards.

If we have 3 qubits, and thus 8 base states, the optimal number of iterations is equal to $\frac{\pi}{4}\sqrt{8}\approx 2,221$, so we would expect our chances of finding the right element to be best after 2 iterations. Let us see now if this is indeed the case. The probability of measurement for each of the base states after the initial Hadamard operation is $\frac{1}{8}$ and the amplitude for each of them is $\sqrt{\frac{1}{8}}$. Since all of them are the same, the average amplitude is also $\sqrt{\frac{1}{8}}$ at this time. We then apply the oracle for the first time. As discussed before, this leaves all amplitudes unchanged except for the one(s) belonging to the base states associated with the element(s) we are looking to find. The amplitude(s) of these element(s) are multiplied by -1. Let us assume that out of the 8 elements we have in our 3-qubit example database, only element number 5 is desired. In this case, after we have applied the oracle, all elements except number 5 still have an amplitude of $\sqrt{\frac{1}{8}}$, with element 5 now having an amplitude of $-\sqrt{\frac{1}{8}}$. We now use this to perform the operation earlier described as "inversion about the mean" to make the probability of our desired base state to be measured increase.

The new amplitude (the 1 above α indicates that this is after the first application of the oracle) of all base states other than 5 becomes:

$$\alpha_i^{(1)}=2(\frac{N-2}{N\sqrt{N}})-(\frac{1}{\sqrt{N}})=2(\frac{N-2}{N\sqrt{N}})-\frac{N}{N\sqrt{N}}=\frac{2N-4}{N\sqrt{N}}-\frac{N}{N\sqrt{N}}=\frac{N-4}{N\sqrt{N}}=\frac{4}{8\sqrt{8}}$$

This means that all of these base states now have a probability of $(\frac{4}{8\sqrt{8}})^2=0,03125$ to be measured. The new amplitude for base state number 5 however, becomes:

$$\alpha_5^{(1)}=2(\frac{N-2}{N\sqrt{N}})-(-\frac{1}{\sqrt{N}})=2(\frac{N-2}{N\sqrt{N}})+\frac{N}{N\sqrt{N}}=\frac{2N-4}{N\sqrt{N}}+\frac{N}{N\sqrt{N}}=\frac{3N-4}{N\sqrt{N}}=\frac{20}{8\sqrt{8}}$$

This means that this desired base state now has a probability of $(\frac{20}{8\sqrt{8}})^2 = 0,883883476$ to be measured. To show that there is an optimal number of Grover iterations after which the probability of finding the desired element starts to decrease again, I will show the result of two more applications of G. First the oracle again multiplies the amplitude of base state 5 by -1 resulting in $-\frac{3N-4}{N\sqrt{N}}$. All other amplitudes are kept unchanged from what was calculated above. After the second inversion about the mean, the new average amplitude $\langle\alpha\rangle$ becomes:

$$\frac{\frac{(N-1)(N-4)}{N\sqrt{N}} - \frac{3N-4}{N\sqrt{N}}}{N} = \frac{N^2-4N-N+4-3N+4}{N^2\sqrt{N}} = \frac{N^2-8N+8}{N^2\sqrt{N}}$$

.
The new amplitude for our desired base state, number 5, becomes:

$$\alpha_5^{(2)} = \frac{2(N^2-8N+8)}{N^2\sqrt{N}} + \frac{3N-4}{N\sqrt{N}} * \frac{N}{N} = \frac{2N^2-16N+16+3N^2-4N}{N^2\sqrt{N}} = \frac{5N^2-20N+16}{N^2\sqrt{N}}$$

.
This means that after the second inversion about the mean is complete, we now have a probability of $(\frac{5*8^2-20*8+16}{8^2\sqrt{8}})^2 \approx 0,9723$ to measure the desired base state. As we can see, we are almost sure of measuring the right element even after only two Grover iterations. The new amplitude for all the other states after two Grover iterations is

$$\alpha_i^{(2)} = \frac{2(N^2-8N+8)}{N^2\sqrt{N})} - \frac{N-4}{N\sqrt{N}} * \frac{N}{N} = \frac{2(N^2-8N+8)-N(N-4)}{N^2\sqrt{N}}$$

$$= \frac{2N^2-16N+16-N^2+4N}{N^2\sqrt{N}} = \frac{N^2-12N+16}{N^2\sqrt{N}}$$

So the chance of measuring any of these base states after two iterations is equal to $\frac{(8^2-12*8+16)^2}{N^5} \approx 0,0078$. Now I will show what happens when we do a third Grover iteration, to demonstrate that this will only serve to decrease our chances of finding our right element, because we will have done more then the optimal number of iterations. First the oracle once more multiplies amplitude of base state 5 with -1, so that state's amplitude becomes $-\frac{5N^2-20N+16}{N^2\sqrt{N}}$ while the rest remains unchanged. The new average amplitude $\langle\alpha\rangle$ of all states together now becomes:

$$\frac{\frac{(N-1)(N^2-12N+16)}{N^2\sqrt{N}} - \frac{5N^2-20N+16}{N^2\sqrt{N}}}{N} = \frac{N^3-12N^2+16N-N^2+12N-16-5N^2+20N-16}{N3\sqrt{N}}$$

$$= \frac{N^3-18N^2+48N-32}{N^3\sqrt{N}}$$

.
The new amplitude of state 5 becomes:

$$\alpha_5^{(3)} = \frac{2(N^3-18N^2+48N-32)}{N^3\sqrt{N}} + \frac{5N^2-20N+16}{N^2\sqrt{N}} * \frac{N}{N} = \frac{2N^3-36N^2+96N-64+5N^3-20N^2+16N}{N^3\sqrt{N}}$$

$$= \frac{7N^3-56N^2+112N-64}{N^3\sqrt{N}}$$

So, after three iterations, the chance of finding the sought-for element has *decreased* to only $\frac{(7*8^3-56*8^2+112*8-64)^2}{N^7} \approx 0,3301$. This is the result of applying too many Grover iterations. We have shown that our calculation at the start, that two iterations would give us the best chance, was correct. If we keep applying more, the chance of finding our sought-for base state will eventually start going up towards 1 again. For good measure, I will also calculate the amplitude for the other base states after three iterations:

$$\alpha_i^{(3)} = 2(\frac{N^3 - 18N^2 + 48N - 32}{N^3\sqrt{N}}) - \frac{N^2 - 12N + 16}{N^2\sqrt{N}} * \frac{N}{N} = \frac{2N^3 - 36N^2 + 96N - 64 - N^3 + 12N^2 - 16N}{N^3\sqrt{N}}$$

$$= \frac{N^3 - 24N^2 + 80N - 64}{N^3\sqrt{N}}$$

We have now finished our discussion of the mathematics behind Grover's algorithm. In the next chapter, some recent developments in quantum computing are discussed, both in terms of algorithms that have been invented and progress that has been made in actually building a quantum computer.

5 Recent Developments

In this chapter, I would like to discuss some recent developments in the field of quantum computing. Given the mathematical complexity of most of these developments, I will not go deeply into the math behind them. I will however discuss what the developments are about and why they could be useful for practical applications. There exists a large number of papers describing both possible novel applications for quantum computing as well as advances in building a real quantum computer. There are too many of these to discuss all of them here, but here is a selection of some of the most interesting papers.

Although quantum communication and encryption was not discussed before in this thesis, it is an important area of research. In my opinion, by *far* the most significant development in quantum computing hardware to date is the Chinese quantum satellite *Micius* launched August 2016. It is the most relevant because this is the first actual piece of quantum computer hardware used in practice outside of small lab experiments. According to an internet article [10] all of the satellites' systems are either working as expected or better, and the satellite will not only be used for quantum communication and encryption experiments but also for physics experiments. This satellite is the clear proof that was needed that at least certain aspects of quantum computing have a practical use outside of academics. It will be used as part of a network being built by China to provide hacking-proof cryptography to Chinese government institutions, banks, etc. It will also be interesting to see what kind of scientific experiments will be done with it in the coming years.

Another potentially important development in the last years are the D-wave quantum computers being sold commercially. According to the company itself, their latest quantum computers have numbers of qubits in the triple digits, and are successfully running certain quantum algorithms much faster then classical computers. Their quantum processors are said to use a technique called The problem with D-wave is that a lot of academics do not trust the company's claims that the processors inside the computers they sell actually use quantum mechanics to execute their calculations. The company has attempted to prove this by showing that their computers execute calculations faster then would be possible if they were classical - reasoning that this means there must be quantum effects going on. This is clearly a difficult way to prove definitively that the chips "are quantum". The reason this difficult proving method has to be used is that the way the quantum effects are implemented on the d-wave CPU makes it difficult to prove the "qubits" are actually in a quantum state. On the other hand, however, companies like Google and Lockheed-Martin are clients of D-wave and have bought these computers, which would appear to lend them some credibility. Some papers on the subject of D-wave and their maybe-quantum-maybe-not computers include:

On the topic of building quantum computers, a study [11] from 2014 gives hope that it might be much easier to prove that quantum computers can outperform classical ones in at least one significant way then we thought so far. If this can be proven, it will serve as a clear justification of the (high) costs and difficulty of developing quantum algorithms and quantum computers. The algorithm which the researchers in question was easier to run then we thought before is called Boson Sampling. In this algorithm, single photons of light are used together with optical circuits to sample exponentially large probability distributions, something that has been proven to be hard for classical computers. The researchers proved that running this algorithm with a single photon to sample a small distribution is still enough to prove a definitive advantage over a classical computer solving that same problem.

One of the first algorithms developed for quantum computers was Shor's algorithm for factoring numbers. The quick and efficient factoring of numbers has important applications, chief among which is the possibility of breaking the currently-secure RSA encryption. This popular form of encryption secures, among other things, a large part of online commerce. The security of these commerce operations, as well as others, depend on the assumption that RSA is a safe system to use for highly sensitive data, which in itself relies on the fact that factorizing (extremely) large numbers is hard on a classical computers.

Although the size in bits of the number RSA uses increases every few years because of the ever-increasing power of classical computers, this basic assumption has hold up until now. Shor's algorithm challenges this status quo, as it proves to factorize large numbers much quicker than a classical algorithm can. Even more importantly, however, researchers have now discovered there might be an easier way to factorize (certain) large numbers on a quantum computers [12]. Whereas the biggest number ever factorized using Shor's algorithm was 21, and that required a quantum computer with 8 qubits, these scientists used a trick to factorize 56153 = 233 x 241 with only 4 qubits. It should be noted that the trick the researchers used only works for some numbers and is thus not universal, and that classical computers can still feasibly factorize much larger numbers including RSA-756, a 756 bit long number. The fact not all numbers can be factorized might make this "trick" less useful when trying to eg break RSA encryption.s Still, this study may prove to have important consequences for which numbers are seasonably factorization when we start to be able to build quantum computers with more qubits in the future, if it so happens that (some of) the numbers people want to factorize do support the use of this method.

One paper [13] about earthquake detection suggests a new method based on quantum computing which could help detect these deadly events before they take place so that people can be evacuated from the area in time. Despite decades of research, there is currently no fully reliable method for predicting earthquakes more then a few seconds in advance. The proposed system uses a pair of entangled photons. One of these is placed in earth orbit, the other on the planet surface in an earthquake-prone region. The subtle gravitational changes which take place before a major earthquake are big enough to disturb the state of the system. Analysis of the system can then be used to analyze the gravity distortion itself which in turn can be used to predict the epicenter and severity of the earthquake. Although the authors were unable to test the system in practice because this would require a ground station, a satellite, functional communications between the two, and an earthquake to take place, they did test their method using simulation and found it to be effective. However, the authors also mentioned two downsides to their system, namely that it can only detect gravitational changes at the exact place on Earth where the photon is positioned, and as a result of this it will be extremely expensive to build enough ground stations to create decent coverage. Personally, I wonder about a related point which the authors do not mention: how many photons can one satellite carry? How many very expensive satellites will one need to build to support an acceptable number of ground stations? Also, how long will the entangled pair remain entangled without collapsing? Keeping qubit in a superposition for longer periods is currently an extremely difficult problem. Nevertheless, this research obviously has the potential to prevent major loss of life in earthquake prone regions, and I believe it is imperative that it is continued further.

Another interesting paper was about representing graphics on a quantum system. [14] If a quantum system does not have a way to accomplish this, it will never be able to display or work with images. The proposed algorithm still only supports grey-scale images (black/white) but does so in a more efficient manner than other known possible quantum algorithms. Also, the authors claim it would be possible to use this method as a basis for other operations, such as color operations. They list four problems with the existing Flexible Representation of Quantum Images (FRQI) model, being that it barely compresses image data, operations applied to one pixel have to be applied to all, it takes eons to convert an image to the format, and images can not be read from the format accurately. The way the authors suggest to improve on these drawbacks is by using two qubits, one to store the grey value of every pixel and another to store its location. The current algorithm uses only one qubit for this, which is why individuals pixels can not be modified without modifying the rest. In the rest of the paper they demonstrate that their proposed model fixes all mentioned drawbacks, albeit at the cost of having to use almost twice as many qubits to be able to store all the data. Still, I agree with them that this is a worthwhile trade off for such significant improvements. The authors conclude the article by suggesting future research might include such questions as whether their algorithm can be used to support more complex operations such as image recognition or image comparison on a quantum computing system.

In [15] the same authors recommend a method for representing audio as quantum data. Together with the previous paper, this would give us a method of creating multimedia systems on quantum computers. The paper explains how audio signals are by nature analog, and how they are normally converted into a discrete signal for processing on a classical computer. Like all waves, sound waves have an amplitude which can be used to describe the sound at any point in time. The authors use a single qubit to store an audio signal in a quantum way: the amplitude of one base state corresponds to the amplitude of the wave on a certain moment and the amplitude of the other base state to store the point in time associated with this wave. This way, they can store all information contained in a discrete audio signal (a series of intensities and time stamps) in a single qubit. Because only one qubit is used to store an entire audio signal, this method can be used to compress audio data much further than is currently possible on a regular computer. After explaining how they would represent audio date on a quantum computer, the authors mention a number of basic sound operations which can be done using their system, such as mixing audio tracks, appending one track to another track, etc.

One of the more purely-mathematical papers deals with using quantum algorithms to work with linear systems of equations. [16] I say "work with" because the proposed algorithm does not find a solution to the equation. It allows one to calculate an approximation of a certain value associated with the equation, which according to the authors is often all that is needed. The authors prove that it allows one to calculate these values exponentially faster than the best theoretically possible classical algorithm which tries to do the same thing. However, a number of quite restrictive conditions had to hold for this to be the case. Still, as the authors point out, linear equations are important in many scientific fields as well as more practical fields, and the set of equations which need to be worked with are often extremely large. A quantum algorithm which can execute at least some common tasks related to these equations exponentially faster than the best classical algorithm is still extremely valuable. The authors of the previous paper listed [17] as an extension of their work because it deals with non linear equations. Like the previous paper, the authors of this work also state their algorithm provides an exponentional speedup over the best possible classical method. Although, also like in the last paper, not all equations of the type the paper deals with can be solved by the algorithm, these equations are also very important in a number of different scientific and industry fields and the algorithm is thus just as important if not more so than the previous one.

There was also a paper about applying quantum computing techniques to future models of robots. [18] These quantum techniques, according to the authors, would enable robots based on this software to learn quicker and work more efficiently. The quantum algorithms suggested by the authors included quantum machine learning, and also Grover's algorithm for more quickly finding the correct action for the robot to take in a particular situation out of all possible actions. It was not made clear however how the oracle used in Grover's algorithm would be able to recognize correct solutions. Off course, it would be impossible for the authors to build an actual robot with actual quantum software inside of it as none exists yet, but they did include results of simulations they wrote to prove their points.

I will now discuss a pape [19] about supervised and unsupervised machine learning on a quantum computer. These are the kind of algorithms the paper mentioned in the previous paragraph would suggest using to build a quantum robot. Robots use machine learning to learn how to perform certain tasks, such as how to lift something without breaking it or what is the shortest route possible to get to a certain location. The authors write that not only would their method be much more efficient than existing classical algorithms in terms of performance, it would also require much less data from the learning database. This means that in addition to enabling the quantum system to learn much faster then it's classical counterpart, it is also much better for privacy, since a much smaller part of the database has to be searched. The authors do not explain, however, why one can not just make up a similar database with fake values to train on and solve the privacy problem that way.

On a more philosophical note, there was a paper [20] which discussed the question if the universe we live in may be one huge quantum computer. It starts with an overview of the development from analog, to digital, and then to quantum computing. After this, the paper gets around to answering its titular question. First, the authors explain that it is not possible for a traditional digital computer to simulate some of the more complicated phenomena in our universe efficiently. They also point out the fact that since classical rules of physics as we know them are fairly simple, and the universe before the big bang was to the best of our knowledge also a pretty simple system, one would not expect the many complex phenomena we observe in our universe based on non-quantum physics. These include everything from intricate biological lifeforms to giant gass-planets to supernovas. In the last section of the paper, the authors use the example of the monkeys and the typewriter to show that it is plausible that the universe is, in fact, one giant quantum computer. A million monkeys typing random texts on a typewrite for a year have a small but non-zero chance of creating all the examples of great literature from all of human history. It then explains that a universe behaving as a quantum computer would not only support more complex phenomena than a classical one, but that it would also necessarily spawn them. The universe itself is the typewriter and the monkeys are replaced by inherent random quantum occurrences which randomize the universe and (part of) its contents every now and then. The article concludes that based on observations of complex phenomena in our universe and the fact that the universe can not be efficiently simulated on a classical computer, it is in fact likely that our universe is equivalent to one giant quantum computer. (There are a lot of projects underway looking to use quantum computing to simulate astrophysical phenomena, which quantum computers are much better at then classical ones because they can directly work with the qubits representing these phenomena then a classical computer which has to simulate them. While I understand this paper is different from the others discussed here in that it does not have a direct practical application, I thought it was interesting to include a paper like this to show what influence quantum computing and concepts related to it might have in other fields such as philosophy.

We have seen that quantum computing provides ways to solve certain problems which are very hard for our current computers in a very quick and efficient way. However, there are also a lot of problems which can (probably) not be solved efficiently by a quantum computer. One paper [21] explored some of the likely limits of quantum computers. It shows that not all problems that are hard on a classical computer will magically be better on a quantum one. Realistic expectations mean there will be less disappointment when it turns out that quantum computers are not the holy grail, able to solve every problem with a snap of their magical fingers, so I can only encourage this kind of theoretical research into the limits of this new but promising technology.

6 Conclusion

In this thesis, we have covered quiet a lot of ground in a relatively small amount of space. In this conclusion, I would like to summarize the most important points from the rest of the text. After this conclusion, I will give a literature list in which I will list all of the referenced books and papers.

We started with a small introduction to quantum computing. A quantum computer is a computing system which uses phenomena known from quantum physics to execute certain calculations faster then is theoretically possible on a classical computing system. There are a lot of methods being studied by physicists and engineers on how to actually implement a quantum computing system, but for the sake of studying algorithms and their applications, the exact underlying implementation of quantum mechanics which the system on which the algorithms are executed uses doesn't matter. Quantum computing researchers have also standardized a lot of the terms and notations used so that it is already possible to research and write about new quantum algorithms despite the fact that no quantum computers have as of yet been build, or at least not with more then about five qubits.

I discussed a number of important quantum algorithms in this thesis, starting out with Deutsch and Deutsch-josza. Although the problem of whether a function f is balanced or unbalanced, which these algorithms solve, has no real practical use, there are an important stepping stone for understanding basic quantum computing concepts such as quantum parallelism and the fact that most quantum algorithms are probabilistic and not deterministic. After these algorithms, I moved on to Grover's search algorithm for quantum computers, which is already a lot more useful. It promises to find a searched-for element from an unstructured database in exponentially less time then the fastest possible classical algorithm. The potential applications of Grover's algorithm range from robots having to choose from a list of possible actions in a certain situation to big data applications were a huge amount of data has to be searched for certain useful information.

In the previous chapter, we have discussed a rather large number of recent developments. Some of the most significant included the launch of a Chinese quantum satellite for communication and also experimental uses, which finally moves at least the cryptography and part of the research applications of quantum computing out of the lab and into actual daily life, and a paper about building quantum robots, which could obviously have a huge impact on any industry which relies on robots for a (large) part of it's work. Meanwhile, the papers on working with audio and imaging data are important for making quantum computers more useful for everyday tasks such as displaying images and playing music, and I also included some more theoretical and philosophical papers which examine such questions as what the limits of quantum computers are and whether our entire universe is in fact a quantum computer.

Concluding, I think that quantum computers and quantum algorithms are an interesting and important field of research which will continue to flourish and grow both in theory and hopefully within the foreseeable future also in practice, and I am curious to see what kind of advances it will bring to the field of computer science. I hope my thesis was interesting to read and has served it's purpose of giving the reader a short overview of the most important aspects of quantum computing.

7 Bibliography

[1] Rieffel, E. and Polak, W., *Quantum Computing: A Gentle Introduction*, The MIT Press, Cambridge, Massuchusetts and London, England, 2011.

[2] Noson S. Yanofsky, Mirco A. Mannucci, *Quantum Computing for Computer Scientists 1st Edition*, Cambridge University Press, New York, United States, 2008.

[3] Nielsen, M. A. and Chuang, I. L., *Quantum Computing and Quantum Information 10th Anniversary Edition*, Cambridge University Press, Cambridge, United Kingdom, 2010.

[4] Arthur O. Pittenger, *An Introduction to Quantum Computing Algorithms*, Birkhuser Basel, 2000.

[5] Andrew Steane, *Quantum Computing*, Reports on Progress in Physics (61), 1998.

[6] Tony Hey, *Quantum Computing: an Introduction*, Computing Control Engineering Journal 10(3), 1999.

[7] , *Linear optical quantum computing with photonic qubits*, Reviews of Modern Physics 79(135), 2007.

[8] B. E. Kane, *A silicon-based nuclear spin quantum computer*, Nature 393, 1998.

[9] E. Knill, *Quantum computing with realistically noisy devices*, Nature 434, 2005.

[10] Stephen Chen, *Handshake shows Chinas quantum satellite performing even better than expected, says scientist*, South China Morning Post, 2016. on internet: http://www.scmp.com/news/china/article/2010667/chinas-quantum-satellite-performing-even-better-expected-says-team

[11] A. P. Lund, A. Laing, S. Rahimi-Keshari, T. Rudolph, J. L O'Brien, T. C. Ralph, *Boson Sampling from Gaussian States*, avaliable on arxiv via https://arxiv.org/abs/1305.4346 2013.

[12] Nikesh S. Dattani, Nathaniel Bryans, *Quantum factorization of 56153 with only 4 qubits*, avaliable on arxiv via https://arxiv.org/abs/1411.6758 , 2014.

[13] Akhter Al Amin, Mahmudul Hasan, Kazi Sinthia Kabir, Tanzila Choudhury, and A. B. M. Alim Al Islam, *Early Detection of Earthquake Using Satellite Based Quantum Computing*, 4th International Conference on Computer Science and Network Technology, 2015.

[14] Yi Zhang, Kai Lu, Yinghui Gao, Mo Wang, *NEQR: A novel enhanced quantum representation of digital images*, Quantum Information Processing 12(8), 2013.

[15] Jian Wang, *QRDA: Quantum Representation of Digital Audio* International Journal of Theoretical Physics 55(3), 2016.

[16] Aram W. Harrow, Avinatan Hassidim, Seth Lloyd, *Quantum algorithm for solving linear systems of equations*, Physical Review Letters 15(103), avaliable on arxiv via https://arxiv.org/abs/0811.3171 , 2009.

[17] Sarah K. Leyton, Tobias J. Osborne, *A quantum algorithm to solve nonlinear differential equations*, avaliable on arxiv via https://arxiv.org/abs/0812.4423 , 2008.

[18] Dao-Yi Dong, Chun-Lin Chen, Chen-Bin Zhang, Zong-Hai Chen, *Quantum Robot: Structure, Algorithms and Applications*, Department of Automation, University of Science and Technology of China, 2005.

[19] Seth Lloyd, Masoud Mohseni, Patrick Rebentrost, *Quantum algorithms for supervised and unsupervised machine learning*, avaliable on arxiv via https://arxiv.org/abs/1307.0411 , 2013.

[20] Seth Lloyd, *The universe as quantum computer*, avaliable on arxiv via https://arxiv.org/abs/1312.4455 , 2013.

[21] Scott Aaronson, *The Limits of Qauntum Computers*, Scientific American 298, 62 - 69, 2008.

[22] Michael Loceff, *A Course in Quantum Computing (for the Community College) Volume 1*, Foothill College, 2015.