University of Ljubljana
Faculty of Computer and Information Science

Kokan Malenko

The consequences of quantum computing

bachelor's thesis

undergraduate university study program
computer science and mathematics

Advisor: Prof. Borut Robič, PhD
Co-advisor: Assist. Prof. Jurij Mihelič, PhD

Ljubljana 2017

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Kokan Malenko

Posledice kvantnega računalništva

diplomsko delo

univerzitetni interdisciplinarni študijski program
prve stopnje računalništvo in matematika

Mentor: prof. dr. Borut Robič
Somentor: doc. dr. Jurij Mihelič

Ljubljana 2017

The Faculty of Computer and Information Science issues the following thesis:

Quantum computers promise to solve certain problems faster than any currently most advanced classical computer. The realization of such a computer would have a significant impact on the current technology. Describe the areas on which quantum computing would have the biggest impact and how it would affect them. Some quantum algorithms promise exponential speedup over the classical algorithms. Explain the main ideas of the most important quantum algorithms and explore their potential applications. To do this, outline the basics of quantum computing and the ways we use quantum phenomena to manipulate data. Estimate how close is science to the implementation of a real universal quantum computer and what are the currently most promising models of quantum computing.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kvantni računalniki obetajo hitrejše reševanje nekaterih računskih problemov kot trenutno najnaprednejši klasični računalniki. Uresničitev takega računalnika bi močno vplivala na današnjo tehnologijo. Opišite področja, na katera bi kvantno računalništvo najbolj vplivalo in kako bi vplivalo nanja. Poznamo že kvantne algoritme, ki obetajo eksponentno pohitritev klasičnih algoritmov. Razložite bistvene zamisli najpomembnejših kvantnih algoritmov in preučite njihovo morebitno uporabo. S tem namenom predstavite tudi osnove kvantnega računanja in kako tako računanje izkorišča kvantnomehanske pojave za delo s podatki. Ocenite, kako blizu je znanost implementaciji stvarnega univerzalnega kvantnega računalnika in kateri modeli kvantnega računalništva so trenutno najbolj obetavni.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Kokan Malenko, z vpisno številko **63100363**, sem avtor diplomskega dela z naslovom:

Posledice kvantnega računalništva

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Boruta Robiča in somentorstvom doc. dr. Jurija Miheliča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 11. septembra 2017 Podpis avtorja:

Contents

Abstract

Quantum computing is a new promising field that might bring great improvements to present day technology. But it might also break some currently used cryptography algorithms.

Usable and stable quantum computers do not exist yet, but their potential power and usefulness has spurred a great interest.

In this work, we explain the basic properties of a quantum computer, which uses the following quantum properties: superposition, interference and entanglement. We talk about qubits, the units of quantum information, and the ways we can manipulate them.

We briefly explain the main idea behind three of the most important quantum algorithms, namely: Shor's algorithm, Grover's algorithm and the Quantum Algorithm for Linear Systems of Equations, also known as HHL.

The biggest emphasis in this thesis is put on quantum reality. In other words what would happen if quantum computers were to become real. We raise questions about the latest achievements in the field of quantum computing and the challenges that it faces, such as, how quantum computers might impact cryptography and should we be worried about the changes that they will bring.

We also discuss the improvements that quantum information might bring to the field of machine learning. Lastly, we introduce one of the most important application of quantum computers, that is, quantum simulation.

Keywords: quantum computing, quantum algorithms, quantum cryptography, quantum machine learning, quantum simulation.

Povzetek

Kvantno računalništvo je novo obetavno področje, ki bi lahko prineslo velike izboljšave današnje tehnologije. Vendar pa bi lahko obenem tudi razorožilo nekatere priljubljene kriptografske algoritme.

Čeprav uporabnih in stabilnih kvantnih računalnikov še ni, sta njihova potencialna moč in uporabnost spodbudili veliko zanimanje.

V tem delu bomo pojasnili osnovne lastnosti kvantnega računalnika, tj. računalnika, ki izkorišča kvantnomehanske pojave, kot so superpozicija, inteferenca in prepletanje. Spregovorili bomo o kubitih, tj. enotah kvantne informacije, ki ustrezajo klasičnim bitom, in o tem, kaj lahko z njimi počnemo.

Na kratko bomo pojasnili glavne zamisli treh najpomembnejših kvantnih algoritmov: Shorovega algoritma za faktorizacijo, Groverjevega algoritma za iskanje in algoritma HHL za reševanje linearnih sistemov enačb.

Največjo pozornost v zvezi z naštetim bomo posvetili t.i. kvantni resničnosti, to je vprašanju, kaj bi se zgodilo, če bi kvantni računalniki postali realnost. Zato si bomo ogledali tudi najnovejše dosežke na področju kvantnega računanja pa tudi izzive, s katerimi se sooča. Na primer, razmislili bomo, kako bi kvantni računalniki vplivali na kriptografijo ter posledično na varnost v računalništvu in, ali bi morali biti za to zaskrbljeni.

Razpravljali bomo tudi o izboljšavah, ki bi kvantno informacijo vpeljale v področje strojnega učenja. Na koncu našega dela se bomo posvetili še eni izmed najpomembnejših uporab kvantnih računalnikov, kvantni simulaciji.

Ključne besede: kvantno računanje, kvantni algoritmi, kvantna kriptografija, kvantno strojno učenje, kvantna simulacija.

Razširjeni povzetek

Kvantno računalništvo je relativno novo področje, ki se hitro razvija. V tej tezi pokrivamo osnovna načela kvantnega računalništva. Razpravljamo o najpomembnejših kvantnih algoritmih in njihovih aplikacijah. Toda glavni del teze raziskuje, kaj se bo zgodilo, če bodo kvantni računalniki postali realnost.

Kvantni biti ali kubiti so osnovne enote informacij v kvantnem računalniku. V primerjavi s klasičnimi biti, ki so lahko samo v enem stanju, torej $|0\rangle$ ali $|1\rangle$, je kubit lahko v superpoziciji obeh stanj. Skupina n klasičnih bitov je lahko le v enem od 2^n možnih stanj, medtem ko je n-kubitni register mogoče najti v katerem koli od 2^n stanj. Da bi dobili kakšno koristno informacijo o n-kubitnem registru, ga moramo najprej izmeriti. Merjenje bo povzročilo, da kubitni register zavzame samo eno od 2^n stanj, kar pomeni, da nimamo dostopa do vseh možnih stanj.

Ena izmed najpomembnejših kvantnih lastnosti je prepletenost. Če prepletemo dva ali več kubitov in izmerimo le enega izmed njih, potem se bodo vsi drugi kubiti takoj postavili na isto vrednost. Ta lastnost je bistvenega pomena za vse kvantne algoritme, ki jih omenjamo v tej tezi.

Za manipulacijo stanja kubitov uporabljamo kvantna vrata, ki morajo biti reverzibilna. V tej diplomski nalogi najpogosteje uporabljamo Hadamardova vrata, ki postavljajo kubitne registre v superpozicijo.

Morda je najpomembnejši kvantni algoritem Shorov algoritem za faktorizacijo celih števil. Njegov pomen je, da ponuja eksponentno pohitritev nad najboljšim znanim klasičnim algoritmom za faktorizacijo praštevil. Varnost kriptosistema RSA, enega izmed najpogosteje uporabljenih kriptosistemov, temelji na dejstvu, da je faktorizacija praštevil v praksi zelo težak problem. Zato bi imela implementacija

Shorovega algoritma močen vpliv na kriptografijo.

Eden od osnovnih problemov v računalništvu je nestrukturirano iskanje. Če bi hoteli poiskati element v tabeli velikosti N, bi morali v najslabšem primeru opraviti N primerjanj. Kvantni računalnik, ki uporablja Groverjev iskalni algoritem, pa lahko reši ta problem s $\sqrt{N}$ primerjanji, kar pomeni, da imamo kvadratično pohitritev. Groverjev algoritem se lahko uporabi za izboljšanje katerega koli klasičnega algoritma, ki kot podproblem vsebuje nestrukturirano iskanje. Na primer: iskanje najkrajše poti, minimalno vpeto drevo, ugotavljanje povezljivosti grafov, ujemanje vzorcev.

Sisteme linearnih enačb je mogoče najti na skoraj vseh področjih znanosti. Klasični algoritem lahko reši sistem linearnih enačb v polinomskem času, medtem ko kvantni računalnik z uporabo kvantnega algoritma za linearne sisteme enačb (HHL) lahko reši isti problem v logaritemskem času. Vendar pa nam HHL ne vrne rešitvenega vektorja eksplicitno, ampak v superpoziciji, iz katere lahko izločimo različne lastnosti rešitvenega vektorja. Zato se HHL ne uporablja za neposredno reševanje sistema linearnih enačb, temveč kot podprogram v drugih kvantnih algoritmih, še posebej pri kvantnem strojnem učenju.

V zadnjih letih se vse večje število velikih podjetij in znanstvenih raziskovalcev ukvarja z uresničevanjem kvantnega računalnika. Trenutno obstaja nekaj obetavnih eksperimentalnih izvedb, od katerih jih nekaj raziščemo v tej tezi. Prva naprava, o kateri razpravljamo, je kvantni računalnik D-Wave, ki ni univerzalni kvantni računalnik, temveč naprava za reševanje specifičnih optimizacijskih problemov. Druga težava z D-Waveom je, da ponuja le konstantno pohitritev nad klasičnimi algoritmi. Kljub vsem svojim težavam je komercialno najuspešnejši kvantni računalnik.

Najbolj obetavni kandidati za univerzalni kvantni računalnik so ionski (angl. trapped-ion) ter superprevodni kvantni računalniki. Trenutno obstaja nekaj eksperimentalnih implementacij ionskih in superprevodnih kvantnih računalnikov, na katerih lahko izvajamo enostavnejše kvantne algoritme. Vendar imajo te izvedbe le omejeno število kubitov (trenutno največ 20), zato ostaja vprašanje ali bodo tudi skalabilni in s tem uporabni, odprto.

Kriptografija igra pomembno vlogo v sodobni tehnologiji. Uporablja se za zagotavljanje varnosti na spletnih straneh, elektronskih transakcijah, e-poštnih sporočilih, telefonih itd. Najpogosteje uporabljeni algoritem za šifriranje z javnim ključem je RSA. Dejanska implementacija Shorovega algoritma na pravem in skalabilnem kvantnem računalniku bi resno ogrozila varnost kriptosistema RSA. Vendar to ne pomeni, da je tudi celotna kriptografija v nevarnosti, saj trenutno obstaja nekaj znanih, tako imenovanih postkvantnih kriptosistemov, ki bi lahko nadomestili RSA. Preden začnemo uporabljati te kriptosisteme, bi potrebovali dovolj časa za vzpostavitev zaupanja vanje ter izboljšanje njihove učinkovitosti in stabilnosti.

Z uporabo klasičnega protokola za zamenjavo ključev ni možno zamenjati ključa na popolnoma varen način. Po drugi strani pa kvantna distribucija ključev (KDK) – ki uporablja nekaj načel kvantne mehanike – omogoča, da dosežemo popolnoma varno izmenjavo. V prisotnosti prisluškovalca je KDK edini znani varni način za izmenjavo ključev med dvema komunikacijskima stranema. Vendar pa se praktična implementacija sistema KDK sooča s številnimi izzivi, kot so potreba po dragi strojni opremi in nezmožnost obravnave nekaterih pomembnejših varnostnih vidikov, kot sta avtentifikacija in integriteta. Poleg tega je KDK nova tehnologija, zato se v praksi lahko pojavijo neodkrite ranljivosti. Dejansko že obstaja nekaj realnih implementacij KDK, ki so ranljive na napade.

Z vsakim letom shranjena količina podatkov raste vse hitreje. Sedanji računalniki in algoritmi za strojno učenje so dosegli svojo mejo. Realizacija kvantnega računalnika in uporaba kvantnih algoritmov bi potencialno izboljšala učinkovitost algoritmov za strojno učenje. Na primer, Groverjev algoritem in HHL nam lahko pomagata doseči znatno in celo eksponentno pohitritev pri nekaterih algoritmih strojnega učenja. Trenutno je na voljo nekaj kvantnih algoritmov za strojno učenje, kot so kvantno prileganje podatkov (angl. data-fitting), kvantni algoritem najbližjega soseda, kvantni algoritem gručenja (angl. clustering), ki so hitrejši od ustreznih klasičnih algoritmov.

Zelo pomembna uporaba kvantnih računalnikov je kvantna simulacija. Razvoj natančnejših in hitrejših simulacij je zelo pomemben na področjih, kot so kvantna

kemija, metamateriali, fizika visokih energij in superprevodnost. Hitrejše simulacije nam lahko pomagajo pri boljšem razumevanju kvantno mehanskih sistemov ali celo pomagajo odkriti kak naravni pojav, ki ga ni mogoče simulirati na kvantnem računalniku. Takšno odkritje bi lahko prineslo dodatno motivacijo za izboljšanje znanih računskih modelov, ki bi morda presegali celo model kvantnega računanja.

Chapter 1

Introduction

Quantum computing is a relatively new field that is developing quickly. In recent years, the attention this fields gets is getting bigger and bigger. And there are really good reasons for that. Quantum computation is interdisciplinary, a mixture of mathematics, physics, computer science and engineering, so it gathers the attention of a wide range of researchers. But the biggest reason for its quick development is that quantum computers might be very powerful and have huge impact in today's technology.

In the second chapter, we will explain the basics of quantum computer architecture. The discussion will mainly revolve around qubits, their properties and ways we can manipulate them. We will present examples of quantum gates and describe how they form quantum circuits. The third chapter will be focused on some of the more important quantum algorithms: Shor's factoring algorithm, Grover's search algorithm and the Quantum Algorithm for Linear Systems of Equations, also known as HHL. We will briefly describe the main idea behind them, give a few simplified examples of how these algorithms work and discuss their importance. In the beginning of the fourth chapter, we will first take a look at the difficulties in creating a universal quantum computer and the newest developments in that field. In particular, we will focus on the two most promising current implementations, namely: the Trapped-Ion quantum computer and D-Wave's quantum annealer. In the rest of the chapter we will describe the consequences that would happen

if a universal quantum computer ever becomes real. Specifically, we will discuss the following questions: How will cryptography change? Are we ready for such a change? What are the current developments in quantum cryptography? Next, we touch upon quantum machine learning and what improvements it can bring. Finally, we will give a brief introduction to quantum simulation and its importance. The last chapter will summarize the whole thesis and give the final conclusion.

Chapter 2

Quantum Computation

In this chapter, we present the necessary concepts for understanding the thesis. Section 2.1 introduces the quantum bits and their most important features. In Section 2.2, we take a look at some of the operations on qubits, which we use in this thesis. In Section 2.3, we take a briefly inspect quantum circuits. [1, 2, 3, 4]

2.1 Quantum bits

The fundamental unit of information in classical computer science is the bit. The classical bit can only be in one of the two states which we will denote by $|0\rangle$ or $|1\rangle$. The quantum counterpart of the classical bit is the qubit (quantum bit), which can be in a state $|0\rangle$, $|1\rangle$ or any superposition $|\psi\rangle$ of them:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. The real number $|\alpha|^2$ is interpreted as the probability that after measuring the qubit, it will be found in the state $|0\rangle$. Similarly, $|\beta|^2$ is interpreted as the probability that after measuring the qubit, it will be found in state $|1\rangle$.

To describe a quantum state in the equation above, we used the Dirac notation, also known as bra-ket notation. In the following table, we summarize the notation we will be using throughout the thesis.

Table 2.1: The summary of the bra-ket notation

Notation	Description
$\|\psi\rangle$	Vector. Also called a ket.
$\langle\psi\|$	Dual vector to $\|\psi\rangle$. Also called a bra.
$\langle\phi\|\psi\rangle$	The inner product of the vectors $\|\phi\rangle$ and $\|\psi\rangle$. Also called a bra-ket.
$\|\phi\rangle \otimes \|\psi\rangle$	Tensor product of the vectors $\|\phi\rangle$ and $\|\psi\rangle$. (It can also be presented as $\|\phi\rangle\,\|\psi\rangle$ or $\|\phi\psi\rangle$.)

2.1.1 N-qubit states

To use a quantum computer we will need more than one qubit of storage. We can express the state of n qubits with the following equation:

$$\sum_{x=0}^{2^n-1} \alpha_x \left|x\right\rangle_n \text{ where } \sum_{x=0}^{2^n-1} |\alpha_x|^2 = 1$$

and the notation $\left|x\right\rangle_n$ means that there are n qubits representing the state x.

In the classical world, each subset of an n-bit group represents a state. This means that a group of n classical bits can be found in only one of the 2^n possible states. In the quantum world, however, we need all of the subsets to represent a state. Therefore, for example, to represent a state of a byte, we need to write only 8 bits, while the state of a qubyte (eight qubits) is represented by writing 2^8 complex numbers. Specifically, if we wanted to emulate a quantum computer having a 64-qubit register, we would need to store 2^{64} complex numbers.

A group of two or more qubits can be written using the tensor product. For example, given a pair of qubits, we can write them down as $|0\rangle \otimes |1\rangle$ or $|0 \otimes 1\rangle$, which means that the first qubit is in the state $|0\rangle$ and the second qubit is in the state $|1\rangle$. In this thesis, we will use a simplified notation for tensor products, where the above qubit pair will be represented by $|01\rangle$.

2.1.2 Measurement

The state of a qubit (the values of α and β) cannot be determined by only examining it. We have to measure the qubit if we want to obtain any useful information about it. And after the measurement, the qubit is left either in the state $|0\rangle$ or $|1\rangle$, which is also the result of the measurement.

Measuring a quantum register with n qubits, results in one of the 2^n basis states $|x\rangle_n$ with probability $|\alpha_x|^2$. For example, if we have a 2-qubit register,

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

and if after measuring the first qubit, we get the value 1, then the state of the quantum register has become

$$|\psi\rangle = \frac{\alpha_{10}|10\rangle + \alpha_{11}|11\rangle}{\sqrt{|\alpha_{10}|^2 + |\alpha_{11}|^2}}$$

2.1.3 Entanglement

Given a qubit in a state $\alpha_0|0\rangle + \alpha_1|1\rangle$ and a second qubit in a state $\beta_0|0\rangle + \beta_1|1\rangle$, then we can also define their joint state $\alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle$. But we cannot decompose every joint state into two qubits. For example, the 2-qubit register in the state:

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Measuring both qubits we get $|00\rangle$ or $|11\rangle$ with probabilities $\frac{1}{2}$. If we measure only one of the qubits, the state of the second qubit is set to the same value, since there is no other possibility. When the state of one qubit depends on the other, the qubits are entangled.

2.1.4 Bell states

The entangled state presented above is part of the so-called Bell basis. The Bell basis consists of four Bell states:

$$|\psi^+\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}}$$

$$
\begin{aligned}
|\psi^-\rangle &= \frac{|01\rangle - |10\rangle}{\sqrt{2}} \\
|\phi^+\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\
|\phi^-\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}}
\end{aligned}
$$

2.1.5 No-cloning theorem

The no-cloning theorem states that it is impossible to clone a quantum state. In other words, it is impossible to make a copy of an arbitrary quantum state without first destroying the original.

The no-cloning theorem is a consequence of linearity. Suppose we had a quantum operator U that could clone arbitrary inputs:

$$U((\alpha\,|\psi\rangle + \beta\,|\phi\rangle) \otimes |0\rangle) = (\alpha\,|\psi\rangle + \beta\,|\phi\rangle)\,(\alpha\,|\psi\rangle + \beta\,|\phi\rangle) \tag{2.1}$$

since U is a quantum operator, it must be linear. Consequently,

$$U((\alpha\,|\psi\rangle + \beta\,|\phi\rangle) \otimes |0\rangle) = \alpha U\,|\psi\rangle\,|0\rangle + \beta U\,|\phi\rangle\,|0\rangle = \alpha\,|\psi\rangle\,|\psi\rangle + \beta\,|\phi\rangle\,|\phi\rangle$$

which cannot be equal for any α and β to equation (2.1). This means that there is no unitary transformation that would take the state $|\psi\rangle\,|0\rangle$ into $|\psi\rangle\,|\psi\rangle$ for an arbitrary $|\psi\rangle$.

This theorem is of critical importance in the field of quantum cryptography.

2.2 Operations on qubits

In order to manipulate the state of qubit, we use quantum gates. Unlike classical gates, all of the quantum gates have to be reversible and represented by unitary matrices. Applying a quantum gate to a qubit simply means multiplying the state (vector) of the qubit by the gate's matrix.

We can represent the state of the qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in vector notation as

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

Suppose that we have a gate represented by the unitary matrix U:

$$U = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Then applying the gate U to the qubit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is the multiplication

$$U \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} a\alpha & b\beta \\ c\alpha & d\beta \end{bmatrix}$$

The state of the qubit, after applying the gate U to it, is $|\psi'\rangle = (a\alpha + b\beta)\,|0\rangle + (c\alpha + d\beta)\,|1\rangle$.

A quantum gate can manipulate more than one qubit. In that case, the matrix of the gate is larger. For example, we need an 8×8 matrix for a 3-qubit register.

In the following subsections, we will take a look at the most important quantum gates that we will need in the rest of this thesis.

2.2.1 NOT Gate

The NOT gate takes as input one qubit and outputs the negated version of that qubit. In other words, it changes the state of the qubit to the opposite. For example, NOT of $|0\rangle$ equals $|1\rangle$ and NOT of $|1\rangle$ equals $|0\rangle$. The gate matrix is

$$NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The symbol we will be using for the NOT gate is depicted in Figure 2.1.

Figure 2.1: NOT gate symbol

2.2.2 Controlled-NOT gate

A controlled-NOT gate is a two qubit gate. It has two inputs and two outputs. The top input is the control qubit and the bottom is the target qubit. The target qubit's state is negated if the control qubit is $|1\rangle$, and it remains the same if the control qubit is $|0\rangle$. The state of the control qubit is left unchanged. The matrix of CNOT gate is

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The symbol we will be using for the controlled-NOT gate is depicted in Figure 2.2.

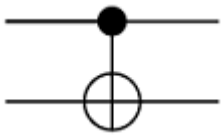


Figure 2.2: Controlled-NOT gate symbol

2.2.3 Toffoli gate

The Toffoli gate, also known as the "controlled-controlled-NOT" gate, is a universal reversible logic gate. This means that using only Toffoli gates, we can construct any reversible circuit. It is similar to the controlled-NOT gate, but it has two controlling qubits. The state of the target qubit changes depending on the states of both control qubits, while the states of the control qubits do not change. The

matrix of Toffoli gate is

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The symbol of the Toffoli gate is in Figure 2.3.

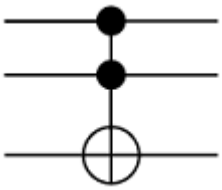


Figure 2.3: Toffoli gate symbol

2.2.4 Hadamard gate

The Hadamard Gate is one of the most important quantum gates. It puts an input qubit into superposition of equally probable states. For example, it maps $|0\rangle$ to $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ and $|1\rangle$ to $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$. It is actually a π-rotation about the $\frac{\pi}{8}$ axis in the complex plain. The matrix of Hadamard gate is

$$H = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

and the corresponding symbol is in Figure 2.4.

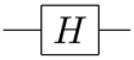


Figure 2.4: Hadamard gate symbol

2.2.5 Measurement gate

The measurement gate is not unitary and is non-reversible. It carries out the measurement of the qubit's state, usually at the end of a computation.
The symbol for the measurement gate is in Figure 2.5.

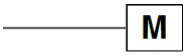


Figure 2.5: Measurement gate

2.3 Quantum circuits

We use quantum circuits to describe quantum computations. Unlike classical circuits, quantum circuits are built of qubits and reversible quantum gates.

An example of a quantum circuit is the following circuit for generating the Bell basis:

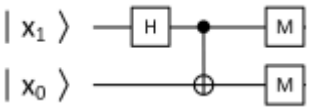


Figure 2.6: A Bell state quantum circuit

We input two qubits $|x_1\rangle$ and $|x_2\rangle$. The first qubit, $|x_1\rangle$ first passes through the Hadamard gate, which puts it in superposition of states. Next, the controlled-NOT gate entangles both qubits. Finally, the entangled qubit passes through the

measurement gate and gives us the result of the measurement.

We can easily see that this quantum circuit gives us arbitrary Bell state, depending on the input:

$$|00\rangle \longmapsto |\phi^+\rangle\,, \qquad |01\rangle \longmapsto |\psi^+\rangle\,, \qquad |10\rangle \longmapsto |\phi^-\rangle\,, \qquad |11\rangle \longmapsto |\psi^-\rangle\,.$$

Chapter 3

Quantum Algorithms

Classical algorithms existed long before the classical computer made an appearance. Similarly, quantum algorithms are developed without knowing when or if a large-scale quantum computer will be ever made. With some clever manipulation of qubits, they can achieve better efficiency than classical algorithms. In this chapter, we will take a look at some of the most important quantum algorithms [5, 6]. In Section 3.1, we are going to describe the main idea of Shor's Factoring Algorithm [7]. In Section 3.2, we are going to discuss Grover's Search Algorithm [8, 12]. Section 3.3 briefly summarizes the Quantum Algorithm for Linear Systems, otherwise known as HHL [9].

3.1 Shor's factoring algorithm

Perhaps the most important algorithm in quantum computing is Shor's factoring algorithm. On a classical computer, the time required to factor an n-digit number grows exponentially with n. The security of RSA [10], one of the most widely used cryptography algorithms, as well as the security of other cryptography algorithms relies on this difficulty of factoring large numbers. However, using a quantum computer, Shor's algorithm can factor an n-digit number in polynomial time, which can have a big impact on cryptography. In this section, we discuss the main idea behind Shor's algorithm.

3.1.1 An overview of the algorithm

The goal of the algorithm is to find the prime factors p and q of a positive integer N, if such primes exist. We can divide the algorithm into three parts, as shown in the figure below.

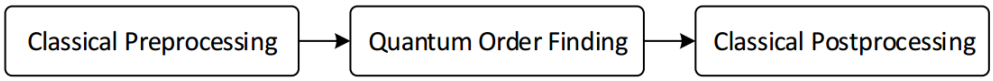


Figure 3.1: The three parts of Shor's factoring algorithm

The reason for this division on a classical and a quantum part is that the classical parts of the algorithm can be executed on a classical computer much more efficiently.

Classical pre-processing

The algorithm starts by preparing values for the quantum order finding:
1. We randomly choose a number a, where $1 < a < N$.
2. Using the Euclidean algorithm we check whether $\gcd(a, N) = 1$. If that is not the case, then we were lucky and we found a factor of N; otherwise, we continue with the algorithm.
3. We compute the number L of bits necessary to store N, that is, $L = log_2 N$.

Quantum order finding

This is the part that enables us to factor N into primes in polynomial time. To be able to find the factors, we need the order of a, which is by definition the smallest integer r such that $a^r + 1 \equiv 0 \mod N$. Rather than the value of r itself, quantum

order finding yields a value from which the order r and the factors p and q can be computed in the post-processing part of the algorithm. The easiest way to understand the quantum part is with the help of a simple example.

Let $N = 15$ and suppose that we choose $a = 2$. Let us make a 3-qubit register in a superposition of states. We can see from the table below that performing a measurement on this register will give us any one of the values with uniform probability $\frac{1}{8}$.

Decimal	Binary	Probability
0	000	$\frac{1}{8}$
1	001	$\frac{1}{8}$
2	010	$\frac{1}{8}$
3	011	$\frac{1}{8}$
4	100	$\frac{1}{8}$
5	101	$\frac{1}{8}$
6	110	$\frac{1}{8}$
7	111	$\frac{1}{8}$

Table 3.1: The superposition of states in a 3-qubit quantum register.

The next step is called Quantum Modular Exponentiation (QME). We entangle the exponent e, from the expression $a^e \mod N$, in a superposition of states. By doing that, we instantly get all the values for e, in our case $e \in 0, ..., 7$. We then raise a to the power of e for each value and calculate the remainders of division with N, which gives an output of equally probably states. The values for our example are depicted in the table below. We can clearly see that the order $r = 4$, because $1 = 2^4 \pmod{15} = 2^{kr} \pmod{15}$ for $k \geq 0$.

e	a^e	$f(a) = a^e \pmod N$	Probability
0	1	1	$\frac{1}{8}$
1	2	2	$\frac{1}{8}$
2	4	4	$\frac{1}{8}$
3	8	8	$\frac{1}{8}$
4	16	1	$\frac{1}{8}$
5	32	2	$\frac{1}{8}$
6	64	4	$\frac{1}{8}$
7	128	8	$\frac{1}{8}$

Table 3.2: Modular exponentiation in quantum parallelism.

The simplicity of the example enabled us to find the order with ease, but in a real-world example with huge numbers, finding the order r is a difficult task. Measuring the register gives us no information, because we can get each state with equal probability. So how do we find the order in real quantum computation?

The first idea would be to measure $f(a)$ without measuring a and then make copies of the output. If we then measure a on the resulting states, we will get different values of a for the same $f(a)$, which will lead us to the order r. But the no-cloning theorem prevents the realization of this idea, because making exact copies of a quantum state is impossible.

This is where the Quantum Fourier Transform (QFT) [11] comes in to help. QFT is a unitary transformation that operates on qubits and, as the name suggests, it is the quantum analogue of the discrete Fourier transform. By applying QFT to our superposition, we transform it to another state which returns, with high probability, the correct order r.

This is how the quantum part is implemented:

Step 1. We start with an $2L$-qubit input register and an L-qubit output register, $|0\rangle_{2L} |0\rangle_L$. We then apply QFT gates to all input qubits and thus put the input register into a superposition of states, $\frac{1}{2^L} \sum_{x=0}^{2^2L-1} |x\rangle |0\rangle$. With the registers

prepared like this, we can apply QME.

Step 2. Applying QME entangles the input and output qubits and sets the output register to $f(x) = a^x \mod N$. After performing QME, the state of the registers is $\frac{1}{2^L}\sum_{x=0}^{2^L-1}|x\rangle|f(x)\rangle$. This step is the runtime bottleneck of the algorithm and is far slower than QFT.

Step 3. We perform a measurement on the output register, which will collapse into state $|f_0\rangle$. While the input register will collapse into a superposition of values x for which $f(x) = f_0$ is true. Let x_0 be the least value such that $f(x_0) = f_0$ and r the period of the function f, then the we can write the input register as $|x_0 + kr\rangle$. Thus, the measurement gives us the state $\frac{1}{\sqrt{m}}\sum_{k=0}^{m-1}|x + kr\rangle|f_0\rangle$, where $m = \lfloor\frac{2^{2L}}{r}\rfloor$.

Step 4. Finally, we apply QFT to the input register which puts it in a state

$$\frac{1}{\sqrt{m}}\sum_{k=0}^{m-1}\frac{1}{2^L}\sum_{y=0}^{2^{2L}-1}e^{2\pi i(x_0+kr)y2^{2L}}|y\rangle = \frac{1}{2^{2L}m}\sum_{y=0}^{2^{2L}-1}e^{2\pi i x_0 y 2^{2L}}\sum_{k=0}^{m-1}e^{2\pi i k r y 2^{2L}}|y\rangle$$

From the last step we can see that the probability of observing the state $|y\rangle$ is $\frac{1}{\sqrt{2^{2L}m}}|\sum_{k=0}^{m-1}e^{2\pi i k r y 2^{2L}}|^2$. The probability function reaches its peak when $y \approx \frac{j2^{2L}}{r}$, where j is an integer. From which follows that, if we perform a measurement, we would most likely get the value $y = \frac{j2^{2L}}{r}$. We can see that $\frac{y}{2^{2L}} \approx \frac{j}{r}$. And if the values j and r have no common factors, we can easily infer the value of r, otherwise we must repeat the algorithm until we get a good measurement.

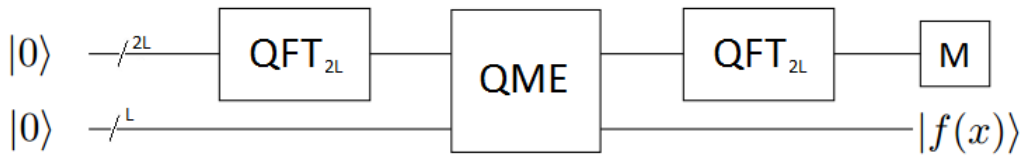


Figure 3.2: Shor's factoring algorithm circuit.

Classical post-processing

On a classical computer, we check whether the returned value is useful or not:
1. If the order r is odd or $a^r + 1 \equiv 0 \mod N$, we must restart the algorithm.
2. Using the Euclidean algorithm, we calculate $p := \gcd(a^{\frac{r}{2}} + 1, N)$ and $q := \gcd(a^{\frac{r}{2}} - 1, N)$.

3.2 Grover's search algorithm

One of the most important problems in computer science is unstructured search. The unstructured search problem can be formulated as follows: Given the ability to evaluate a function $f : \{0,1\}^n \to \{0,1\}$, find a such that $f(a) = 1$, if such an a exists.

Without prior information about the function, all classical algorithms, take in the worst case, $N = 2^n$ evaluations to find an item in an unsorted table. Grover's algorithm can solve this problems using $\sqrt{N} = 2^{\frac{n}{2}}$ evaluations. Even though it gives us only a quadratic speedup, it is one of the most important quantum algorithms, because of the wide range of problems where we can apply it.

3.2.1 The idea of the algorithm

The best way to describe Grover's search algorithm is geometrically. We will define two important states: $|a\rangle$, the state we are looking for; and $|e\rangle = \sum_{x \neq a} \frac{1}{\sqrt{N-1}} |x\rangle$, every other state. The vectors $|a\rangle$ and $|e\rangle$ are orthogonal and span a two dimensional subspace in which the uniform superposition $|\psi_0\rangle = \sum_x \frac{1}{\sqrt{N}} |x\rangle$ is contained. The starting point of Grover's algorithm is the state $|\psi_0\rangle$. As we can see in the geometric representation, $|\psi_0\rangle$ lies very close to $|e\rangle$, because only one part of $|\psi_0\rangle$ is $|a\rangle$ and the rest of it is $|e\rangle$. The goal is to get to a state close to $|a\rangle$, so that when we perform the measurement, we get a with a good probability. The way we achieve this is by using reflections. First, we reflect $|\psi_0\rangle$ about the $|e\rangle$ vector, thus getting the state $R_{|e\rangle} |\psi_0\rangle$. Then, we reflect $R_{|e\rangle} |\psi_0\rangle$ about $|\psi_0\rangle$, which gives us the state $R_{|\psi_0\rangle} R_{|e\rangle} |\psi_0\rangle$. By repeating this process, the angle between our state

and $|a\rangle$ decreases, which is exactly what we wanted. Once the angle between our state and $|a\rangle$ is sufficiently small, we can perform the measurement.

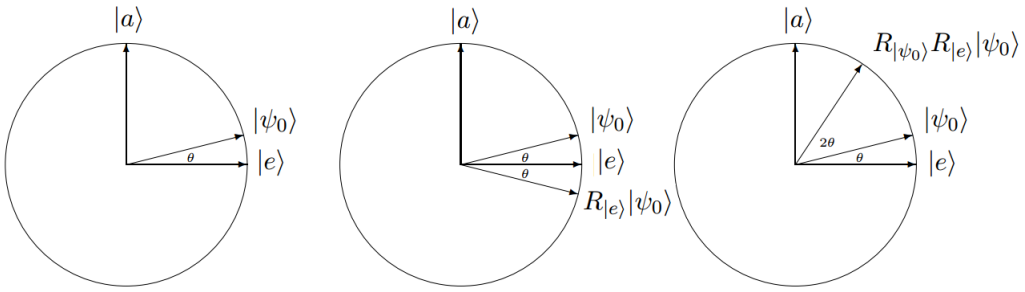


Figure 3.3: One iteration of Grover: (left) starting position, (middle) reflecting about $|e\rangle$, (right) reflecting about $|\psi_0\rangle$

As we can see from this geometrical representation of the algorithm, on each iteration, we rotate the vector $|\psi_0\rangle$ by 2θ, which means that to complete the algorithm, we would need at most $\frac{\pi/2}{2\theta}$ iterations. To see what the value of θ is, we need to take a look at the following equations:

$$\langle\psi_0|a\rangle = \cos(\pi/2 - \theta) = \sin(\theta),$$
$$\langle\psi_0|a\rangle = \sum_x \frac{1}{\sqrt{N}} \langle x|a\rangle = \frac{1}{\sqrt{N}}$$
$$\implies \sin(\theta) = \frac{1}{\sqrt{N}}$$

Since $\frac{1}{\sqrt{N}}$ is very small $\sin(\theta) \approx \theta$, and $\theta \approx \frac{1}{\sqrt{N}}$. We can conclude that we need $O(\frac{\pi/2}{2\theta}) = O(\frac{\pi/2}{2/\sqrt{N}}) = O(\sqrt{N})$ iterations to complete the algorithm. We should also notice that if we iterate further, we will rotate ψ_0 away from a, thus lowering the probability of getting the correct solution.

3.2.2 The quantum oracle

As we can see in the geometrical representation of the algorithm, the first operation that is necessary to reach the solution with high probability is reflection over the $|e\rangle$

vector. To reflect over $|e\rangle$, which is orthogonal to $|a\rangle$, we just need to flip the sign of any component $|a\rangle$ to $-|a\rangle$ and leave all other components as they are. Since we do not have any information about $|a\rangle$, we would have to use a quantum oracle. To put it simply, an oracle is a device which, given an input, can recognize the solution. In our case, we can construct a quantum circuit $U_f : \sum_x \alpha_x |x\rangle \rightarrow \sum_x (-1)^{f(x)} \alpha_x |x\rangle$, which has the property to set $f(x) = 1$ when $x = a$ and $f(x) = 0$ otherwise. We can see that this implementation of the quantum circuit flips the sign of any component $|a\rangle$, which is exactly what we needed.

3.2.3 The diffusion operator

The second operation is reflection over the $|\psi_0\rangle$ vector. To reflect over the $|\psi_0\rangle$ vector, we first map $|\psi_0\rangle \rightarrow |00...0\rangle$, which is achieved using the Hadamard gate [14]. Then, we simply reflect over the zero vector, which is given by $-I + |0\rangle \langle 0|$. Finally, using the Hadamard gate, we return to our original basis. Combining these three operations will give us the diffusion operator $D = H[-I + |0\rangle \langle 0|]H$.

3.2.4 The summary of the algorithm

Here are the simplified steps of Grover's algorithm:

Step 1. Start with the state $|\psi_0\rangle = \sum_x \frac{1}{\sqrt{N}} |x\rangle$.

We get to the starting state by applying $H^{\otimes n}$ to $|0\rangle$.

Step 2. Repeat $\sqrt{N}$ times:

Step 2a. Apply U_f.

Step 2b. Apply the diffusion operator: D.

Step 3. Measure the qubits.

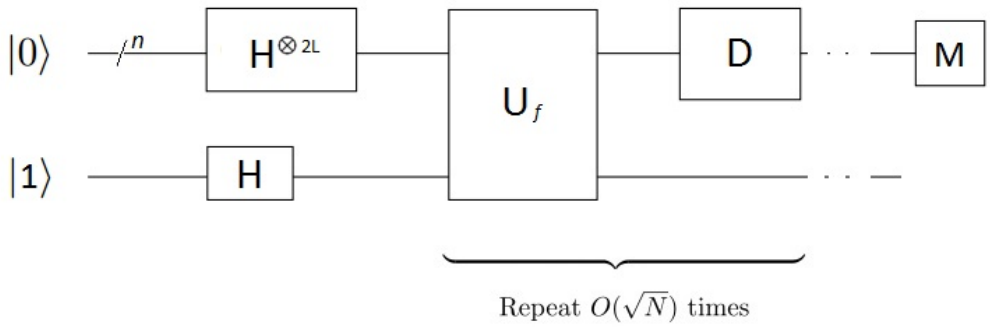


Figure 3.4: Grover's algorithm circuit

3.2.5 Applications

Grover's algorithm is one of the most applicable quantum algorithms [13]. It can be used to improve any classical algorithm that uses an unstructured search, such as finding shortest path, minimum spanning tree, various graph algorithms, etc. Below, we list just a few of the speedups.

1. Determining graph connectivity. If we have a graph with N vertices, using a classical algorithm, we can determine if it is connected or not in time $O(N^2)$, in the worst case. A quantum algorithm based on Grover's search can solve this problem in time $O(N^3/2)$.

2. Pattern matching. If we have a text T of the length N and we want to find a pattern P of the length M in it, we can achieve this in time $O(N + M)$, using a classical algorithm. Using a quantum algorithm, we can solve the problem in time $O(\sqrt{N}+\sqrt{M})$. This speedup is especially important in bioinformatics and text processing, where pattern matching is a fundamental problem.

3. It can be used to quadratically speed up the computation of NP-complete problems. For example, the satisfiability problem: Given a Boolean formula $\phi(i_1, ..., i_n)$, is there a setting of the bits $i_1, ..., i_n$ such that $\phi(i_1, ..., i_n) = 1$. Using brute-force,

we need to search through 2^n possible settings, which takes time $O(2^n)$. With a quantum search algorithm, we can speed up the search to $O(\sqrt{2^n})$

3.3 Quantum algorithm for linear systems of equations

One of the most fundamental tasks in engineering, mathematics and almost all of the other areas in science is solving systems of linear equations. A solution to a system of linear equations is a vector $\vec{x} \in \mathbb{R}^n$, such that $A\vec{x} = \vec{b}$, where $A \in \mathbb{R}^{n \times n}$ and $\vec{b} \in \mathbb{R}^n$. Using the fastest classical algorithm, this problem can be solved in polynomial time $O(nk)$, where k is the condition number, a parameter that measures how much the output value of a function can change for a small change in the input argument. The quantum algorithm for linear systems of equations, also known as HHL (A. Harrow, A. Hassidim, and S. Lloyd), can solve $A\vec{x} = \vec{b}$ in logarithmic time $O(\log(n)k^2)$. It has to be noted that it does not output the solution $\vec{x}$ explicitly, but the superposition $|x\rangle$, which encodes the elements of $\vec{x}$ in its amplitudes.

3.3.1 An overview of the algorithm

The goal of the algorithm is to find $\vec{x}$ that satisfies $A\vec{x} = \vec{b}$, where A is $n \times n$ Hermitian matrix [1] and $\vec{b}$ is a unit vector. The unit vector $\vec{b}$ can be represented as the quantum state $|b\rangle = \sum_{i=1}^{n} b_i |i\rangle$. What we want to do is apply e^{iAt} to $|b\rangle$, where t is a superposition of different times. This can be achieved using Hamiltonian simulation [15, 16]. Next, we need to decompose $|b\rangle$ into the eigenvectors of A, which will be denoted with u_j, and find their corresponding eigenvalues λ_j. As mentioned earlier the matrix A can be exponentiated, which allows us to decompose the state

[1] In case A is not a Hermitian matrix, we can define $H = \left(\begin{smallmatrix} 0 & A \\ A^\dagger & 0 \end{smallmatrix}\right)$, which is a Hermitian matrix. Then from $H\vec{y} = \left(\begin{smallmatrix} \vec{b} \\ 0 \end{smallmatrix}\right)$ we can obtain $y = \left(\begin{smallmatrix} 0 \\ \vec{x} \end{smallmatrix}\right)$. For the sake of simplicity, we will assume A is Hermitian in the rest of the thesis.

$|b\rangle$ using phase estimation [17]. After these actions are completed, the state in which the system will be is $\sum_{j=1}^{n} \beta_j |u_j\rangle |\lambda_j\rangle$, where $|u_j\rangle$ are vectors of the eigenbasis of A and $|b\rangle = \sum_{j=1}^{n} \beta_j |\lambda_j\rangle$. We then need to map $|\lambda_j\rangle$ to $C\lambda^{-1} |\lambda_j\rangle$, where C is a normalizing constant. Finally, we undo the phase estimation to uncompute $|\lambda_j\rangle$ and the system is left in the state $\sum_{j=1}^{n} \beta_j \lambda_j^{-1} |u_j\rangle = A^{-1} |b\rangle = |x\rangle$.

3.3.2 Algorithm caveats

When using the algorithm, there are four very important caveats to watch out for [18]. Each of these caveats can be detrimental in practice.

1. It is crucial to have the ability to load the vector b quickly, so that we can create the quantum state $|b\rangle = \sum_{i=1}^{n} b_i |i\rangle$.

2. The matrix A needs to be sparse; in other words, it should contain at most d nonzero elements in a row, for some $d \ll n$.

3. Then the amount of time required by HHL grows nearly linearly with k. This is why the condition number k needs to be small.

4. As mentioned earlier, the output is $|x\rangle$ and not the vector x itself. If we wanted to output any specific element x_i, we would need to repeat the algorithm n times, which would ruin the exponential speedup.

This last caveat is the reason why we use this algorithm only if we are interested in an approximation of the expectation value of some operator M associated with x. We can then map M to a quantum operator and perform a measurement on it, which will give us the expectation value[2] $\langle x| M |x\rangle = \vec{x}^T M \vec{x}$. Using different operators M, we can extract different features from the vector $\vec{x}$, such as normalization, moments, etc.

However, the algorithm is only useful if we can address all the caveats. This is why it is most often used as a subroutine in other quantum algorithms, especially in quantum machine learning.

[2]In quantum mechanics, the expectation value is the probabilistic expected value of the result (measurement) of an experiment.

Chapter 4

Quantum Reality

In this chapter, we are going to see what would happen if quantum computers became real. How would they impact our reality? What positive or negative changes would they bring? In Section 4.1, we will see what the current state of quantum computers is [19]. Section 4.2 will discuss the impact of quantum computing on cryptography [20]. In Section 4.3, we will take a look at what kind of changes will quantum computing bring to the field of machine learning [21, 22]. And in Section 4.4, we will briefly summarize quantum simulation [23, 24].

4.1 The current state of quantum computers

The reality of having a universal quantum computer may have a huge impact on a wide variety of scientific fields and on today's modern life as a whole. Since the first introduction of the idea of a quantum computer by Richard Feynman [25], there has been a lot of research and improvement in this field. With the rapid growth of big data and the need for more secure communication, the interest in developing a quantum computer rises. And in the last few years an increasingly large number of big companies and scientific researchers have been working toward the realization of such a device.

There are many challenges regarding the construction of a quantum computer. One of the main problems is decoherence. To initialize a classical computer, we

need to put it in a well-defined state and, in the quantum computer case, in a pure state. However, the quantum nature of a quantum computer makes it really difficult for us to do this. The qubits interact with the environment and they might become entangled with it, resulting in impure or mixed state. This loss of purity of the state of a quantum system as the result of entanglement with the environment is known as decoherence. The challenges that decoherence presents are tough to tackle. On the one hand, it is difficult to manage a quantum system, because it is prone to entangling with some usual particles in the environment, such as electrons. On the other hand, we are part of the environment which makes it even more difficult to interact with the quantum device.

In 2000, a theoretical physicist, David P. DiVincenzo, published a set of conditions that must be met in order to construct a quantum computer [26]. The set consists of five conditions necessary for quantum computation and additional two that are needed for quantum communication. Without going into details, we list the seven conditions, namely:

1. A scalable physical system with well-characterized qubits.

2. The ability to initialise the state of the qubits to a simple fiducial state, such as $|000...\rangle$.

3. Long relevant decoherence times, much longer than the gate operation time.

4. A "universal" set of quantum gates.

5. A qubit-specific measurement capability.

6. The ability to interconvert stationary and flying qubits [31].

7. The ability to faithfully transmit flying qubits between specified locations.

Currently, there are some promising experimental implementations that might lead us to a universal quantum computer. In the following subsections, we will take a look at some of the more important implementations. The first one is quantum annealer which is not a universal quantum computer and, for now, it is not trying to be one [27]. It is a device that rather than using quantum gates, uses the quantum annealing method to reach a solution. This is the reason why it cannot efficiently run some of the known quantum algorithms, such as Shor's algorithm. However,

it might prove to be useful in some specific tasks. In the second subsection, we discuss the implementation, such as Trapped-Ion quantum computing [28] and Superconducting quantum computing [29, 30], which are scientifically accepted and are more likely to develop into a real universal quantum computer.

4.1.1 Quantum annealer

To better understand what quantum annealer and quantum annaeling is, let us first take a look at simulated annealing. Simulated annealing is a probabilistic optimization method for approximating the optimum of a given function. It is an analogy for annealing in metallurgy, which is a technique that uses heating and cooling to reduce the defects of a material. In the case of a real algorithm, when we rise the "temperature", bits begin flipping between 1 and 0 regardless of the solution they bring. But when we start the "cooling", some of the bits tend to flip only to states that will make the solution better. When we finally reach "zero temperature", the bits will only flip towards values that bring better solution. If we imagine the possible solution as an energy landscape with hills and valleys, cooling the system will make the bits only go downhill towards the lowest state of energy.

The biggest problem with simulated annealing or any other method that searches only locally is that they can get stuck in a local optima. The goal is to find the lowest point in the energy landscape, but because it searches locally, the algorithm can get stuck in the lowest valley that is closest to its starting point, without knowing that there is a lower valley in the landscape. Simulated annealing tries to combat this with rising the temperatures, which might enable us to go over hill and find better solutions than the initial one. However, in the case of really high hills, we would need exponential amount of time to go over it. This is where quantum annealing has the advantage. Quantum annealing uses a quantum mechanical property, the so-called quantum tunneling, that allows for particles to "go through barriers". That means that in cases where we have high and narrow hills, it can bring exponential speedup over simulated annealing and find the minimum in a polynomial time.

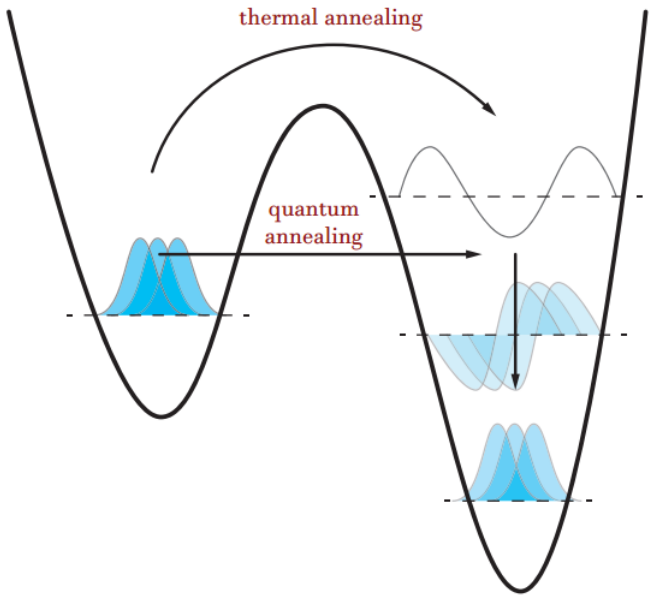


Figure 4.1: Depiction of thermal/simulated annealing and quantum annealing

Implementations

The first quantum annealer, called D-Wave One, was introduced in 2011 by D-Wave Systems. It has 128-qubit processor chipset. The same company introduced three more models, called D-Wave Two (512-qubits in 2013), D-Wave 2X (1152-qubits in 2015) and D-Wave 2000Q (2048-qubits in 2017). These systems have spurred a great interest because of the potential applications in optimization and machine learning. Some big name companies, like *Lockheed Martin*, *Volkswagen Group*, *Virginia Tech* and *Google*, in collaboration with *NASA*, have already purchased a D-Wave system.

However, because of the lack of proof that the system really offers quantum speedup, D-Wave Systems have been largely criticized by the scientific community [32]. The other thing they have been criticized for is that they only increased the number of qubits (as quickly as possible), neglecting their coherence and lifetime, and, consequently, error correction. They do not even try to go for something in

which we are confident that, in theory, will bring quantum speedup.

There are research papers that suggest that D-Wave systems use quantum tunneling to achieve speedup. In 2015, Google published a paper [33] that claims a 100 million times better performance than the approximation algorithm called Quantum Monte Carlo [34]. The problem with this is that, in computer science, we are interested in asymptotic and not constant speedup. Another problem is that D-Wave is a specific-purpose, hundred million dollar machine that offers a constant speedup over a classical computer. In fact, there is already an algorithm running on a classical computer called Selby's algorithm [35, 36] that according to Google's paper, outperforms D-Wave in every task.

Despite its quick and "unscientific" approach, the D-Wave computer is somewhat commercially successful and commercial success often leads to improvements. And even though it is not a universal quantum computer, it might find itself useful in specific-purpose tasks, especially in machine learning and optimization, or even serve as a stepping stone in the development of a real universal quantum computer.

Applications

Any quadratic unconstrained binary optimization (QUBO) problem [37] can be solved on a quantum annealer, such as the D-Wave. Embedding a QUBO problem onto the D-Wave processor is generally a NP-hard problem. However, there are strategies like using simulated annealing or structured-based methods that can help in overcoming this difficulty [38]. There are a number of problems that can be mapped to a QUBO, such as support vector machines, Hopfield neural network, correlation clustering etc. NASA researchers from the Google-NASA-USRA quantum artificial intelligence lab, published a paper [39] that shows how certain problems related to space exploration can be mapped onto a QUBO. These are some of the problems described in the paper:

- classification for planetary feature identification,
- clustering for pattern recognition,
- anomaly detection for space systems,

- structured learning for multiple label classification and
- data fusion and image matching for remote sensing.

There are a lot of problems that are not yet, but can potentially be, formulated as a QUBO. To name a few examples: Feed-forward neural networks, finding parameters of a hidden Markov model, graph coloring, finding the "shortest vector" in a lattice, etc.

Any algorithm that can be converted to simulated annealing can be run on the D-Wave. The process of converting algorithms to simulated annealing is shown in D-Wave's paper [40]. In 2016, Raouf Dridi and Hedayat Alghassi, published a paper [41] which showed how prime factorization can be carried out on a quantum annealer. They illustrated the process on the D-Wave 2X by factoring bi-primes[1] up to 200099. As we can see, the largest factored bi-prime is only a 17-bit number, which is not even nearly enough for factoring even the smallest RSA key of 512 bits. The biggest constraint is the number of qubits; with increasing the number of qubits, the D-Wave will be able to factor larger bi-primes.

As we can see, there are a lot problems potentially solvable on a D-Wave system. However, we must note that these are only ideas at this moment and there are no implementations on most of the above mentioned problems. In addition, even if a problem can be formulated as a QUBO, the QUBO might not be as efficient as the original problem. The other big problem that arises from the limited amount of qubits that D-Wave currently offers is space complexity. Moreover, at the current stage, it is not very clear how well the D-Wave system will scale with a large amount of qubits and whether certain problems will scale well enough.

4.1.2 Candidates for universal quantum computer

A universal quantum computer should be able to run every possible quantum algorithm as well as simulate quantum phenomena. As we will discuss in the following chapters, it can be applied and bring improvements to a lot of fields, such as cryptography, machine learning, physics, medicine, chemistry, etc. Currently, there are

[1] A bi-prime (semiprime) is the product of two prime number.

a lot of proposed implementations for a quantum computer, such as: trapped-ion quantum computers, Superconducting quantum computers, topological quantum computer, nuclear magnetic resonance (NMR), quantum dot computer, etc. The two most promising implementations are trapped-ion computing and superconducting quantum computing, which we discuss in the following subsections. In the last subsection, we also briefly mention topological quantum computing, which, at the current stage, is only theoretical.

Trapped-ion quantum computers

The most promising way to the realization of a universal quantum computer might be trapped-ion quantum computers. As the name suggest, atomic ions are used as qubits in this implementation. To perform quantum gate function, the qubits are manipulated using lasers and microwave pulses, which are then read out through the fluorescence the ions emit. Some of the most important quantum algorithms including Deutsch-Josza, QFT, quantum error correction and Grover's search, have been implemented on an experimental implementation of a trapped-ion computer [42]. And some quantum protocols, like quantum teleportation, quantum simulation and state mapping from an ion to a photon, have already been realized on it.

Perhaps the biggest improvement in trapped-ion computing was published in August 2016 in a paper that demonstrates a five-qubit re-programmable computer [44]. It works by compiling the algorithms into a set of universal quantum gates and it then executes that sequence of gates to perform the algorithm. One of the advantages of this computer is its flexibility. We can reprogram it by changing the gate sequence, without making any changes to the hardware. So far, a few quantum algorithms, including Deutsch-Josza, Bernstain-Vazirani, QFT and even Shor's algorithm [43], have been implemented on it with great success. But the biggest advantage and the most beautiful thing about this computer is that is modular. This means that by increasing the number of qubits in it and connecting it to other modules, we can create a large scale re-programmable quantum computer.

The advantages that trapped-ion computers offer over other implementations

are stability, high gate fidelity and the long lifetime of their qubits. Furthermore, using a small number of qubits, most of the conditions in DiVincenzo's criteria have been met. The biggest disadvantage of this approach is the number of stabilized and individually controlled lasers needed to manipulate the ions, which is a hard engineering challenge. Luckily, in March 2017, a blueprint for microwave trapped-ion computer was published, which introduces an implementation that will use long-wavelength radiation and locally applied magnetic fields [45]. A large scale trapped-ion quantum computer will need to have thousands of perfectly aligned and individually controlled lasers, while in the microwave-based implementation, this is not the case. Which suggests that one of the biggest problems might be resolved and a universal quantum computer is within our grasp.

Superconducting quantum computing

As the name suggests, superconducting quantum computers (SQC) are implemented using superconducting [2] electronic circuits. Implementations of algorithms such as Deutsch-Josza and Grover's algorithm have already been demonstrated on a SQC with a great success [47]. In February 2017, a group of researchers, published a paper in which they compared a 5-qubit SQC with a 5-qubit trapped-ion computer [48]. The paper showed that the SQC is faster, but the trapped-ion computer is more stable and might scale better.

The current state-of-the-art SQC runs on only 20 qubits and it is capable of entangling at most 10 qubits [49], which is barely enough for simple quantum computations. To be able to run more complex algorithms and call itself useful the SQC will need to run on at least 50 qubits.

However, SQC's future looks very bright as it is one of the most widely supported candidates for a universal quantum computer. Tech giants such as IBM and Google already have superconducting-qubit labs. In May 2016, IBM launched a 5-qubit cloud SQC, called IBM Q, which can be freely used by researchers around the world. Despite its low qubit number, the IBM Q spurred a lot of interest and

[2] Superconductors are materials in which, when cooled below a critical temperature occurs a phenomenon called superconductivity [46].

helped programmers tackle important challenges in programming a real quantum computer, which is very different from programming a simulated machine. Google also invested a lot in SQC and promised to launch a similar cloud SQC once they are able to get a machine to run on 50 qubits.

Topological quantum computing

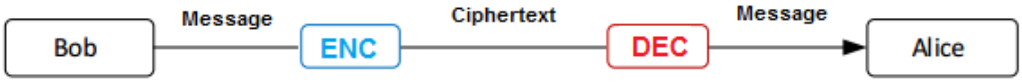
Topological quantum computing (TQC) [50] is an implementation of a quantum computer that offers a better stability than other implementations. Decoherence makes quantum computers prone to errors and they must rely on error correction to be able to work correctly. In TQC the protection from errors occurs on a hardware level, rather than relying on active error correction [51]. Since Microsoft's announcement that they are working on TQC, a few very important papers in this field have been published, and the interest in this technology has grown. One of the more important papers is the roadmap for scalable topological quantum computer based on topological superconductors [52], which was published in June 2017, by a team of experimentalists and theorists. However, currently TQC is just theoretical, as there are no real-life implementations. The reason for this is the difficulty to implement a TQC, and some physicists even think that it is physically impossible to implement it.

4.2 Quantum cryptography

Cryptography plays a really important role in modern technology. It is used to provide security on web sites, electronic transactions, e-mails, phones, etc. One of the most important changes that quantum reality might bring will be in today's cryptography. In this section, we are going to take a look at how quantum reality will affect classical cryptography and briefly introduce quantum cryptography.

4.2.1 Classical cryptography

Before diving into quantum cryptography, we need to familiarize ourselves with some basic ideas of classical cryptography. Cryptography is the study of techniques for secure communication through insecure channels. Or, to put it simply, cryptography is the art of concealing messages. Before sending a message, the sender encrypts the message. After receiving the encrypted message, the receiver decrypts the message. We refer to the original message as plaintext, and to the encrypted message as ciphertext.



A simple example of a cryptosystem is the one-time-pad protocol, also known as Vernam cipher. In the following figures, we can see how the Vernam cipher works. (For the sake of simplicity, we will use a binary string message.)

Step 1. Alice generates a random encryption key.

Step 2. Alice encrypts her message using the randomly generated key.

Alice								
Original message M		0	0	1	1	0	1	0
Encryption key K	⊕	1	0	0	1	0	0	1
Encrypted message E		1	0	1	0	0	1	1

Step 3. Alice sends the encrypted message to Bob.

Step 4. Bob receives the message and decrypts it using the same key.

Bob								
Original message M		1	0	1	0	0	1	1
Decryption key K	⊕	1	0	0	1	0	0	1
Decrypted message M		0	0	1	1	0	1	0

There are two bigger issues with this protocol:

1. For every new message that we want send, we have to generate a new key. If we use the same key twice, then someone can discover the text through statistical analysis. This is why the protocol is called one-time-pad.

2. The second issue with this protocol is that eavesdropper Eve can intercept the key and easily decipher the message. (Both Alice and Bob use the same key to communicate.)

In the next section, we will take a look at more complex cryptosystem, in which these issues have been resolved.

4.2.2 Breaking the RSA

One of the most widely used cryptosystems for secure data transmission is the RSA cryptosystem. The RSA is a part of the public-key cryptosystems which are based on algorithms that use two different keys, one private (secret) and one

public key. For encryption, we use the public key (which, as the name suggests, is available to everyone). For decryption, we use the private key, which is available only to the receiver of the encrypted message.

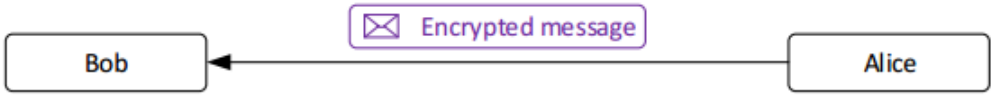
In the following figures, we will see, step by step, how an RSA encrypted message exchange works.

Step 1. To be able to send a message to Bob, Alice has to acquire his public key. He can generate a new key or use the old one.

Step 2. Bob sends his public key to Alice, but keeps his private key.

Step 3. Alice encrypts her message with Bob's public key.

Step 4. Alice sends her encrypted message to Bob.



Step 5. Bob decrypts the encrypted message using his private key.

Alice can reuse Bob's public key to send him other messages.

The public-private key generation starts by choosing two different prime numbers p and q and computing their product $N = pq$. Then, we compute $(p-1)(q-1)$ and find an integer e such that $1 < e < (p-1)(q-1)$ and $\gcd(e, (p-1)(q-1)) = 1$ (i.e., e and $(p-1)(q-1)$ are coprime). Finally, we calculate an integer d, such that $d = e^{-1} \pmod{(p-1)(q-1)}$ (i.e., d is the modular multiplicative inverse of $e \pmod{(p-1)(q-1)}$).

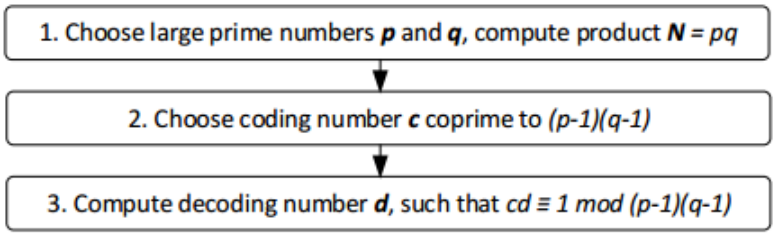


Figure 4.2: Private-public key pair generation

The public key consists of (N, e) and the private key consists of (N, d).

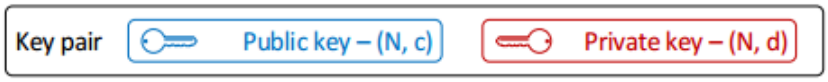


Figure 4.3: Private and public key content

We encrypt the message using the public key in the following equation:

$$c = m^e \pmod{N}$$

where $m < N$. If $m > N$, then we have to split the message into parts that are smaller than N and encrypt them separately. The decryption is done by using the private key:

$$m = c^e \pmod{N}$$

To ensure security, it is obvious that the decoding integer d must be retained secret. But the prime numbers p and q must also be kept secret, because by knowing them, we can compute d and easily decrypt the cipher.

Secret values	Non-secret values
• Prime numbers p and q • Decoding number d	• $N = pq$ • Coding number c

Figure 4.4: Values in RSA cryptosystem

The core of the RSA cryptosystem and the reason why $N = pq$ is public is that there is no efficient classical algorithm for factorization. However, when we take quantum computers into consideration, we might be able to break the RSA cryptosystem using Shor's factoring algorithm.

Let us say eavesdropper Eve has a quantum computer. She can easily take a hold of the public key and, using the Shor's algorithm, she can factor N into p and q. Now Eve knows p, q and c. This is all she needs to compute d in the same way Bob did. By knowing d and N, she already has the private key (N, d) and can decrypt the message that Alice sent to Bob.

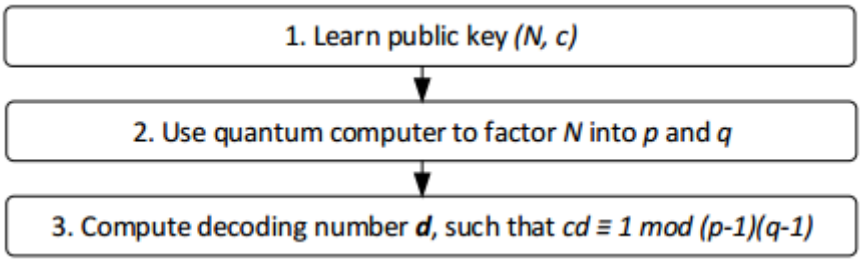


Figure 4.5: Finding d using a quantum computer

What would breaking the RSA mean for us? For starters, the certificates on the websites are signed with RSA key of a certificate authority. If someone breaks the RSA, then he or she can forge RSA signatures and we would not be able to

verify any sites that we visit. RSA is one of the algorithms that is used in the Transport Layer Security protocol (TLS). Our passwords are encrypted with this protocol, every time we log in to a website. And if RSA was broken, than someone could steal our passwords.

4.2.3 Post-quantum cryptography

However, breaking RSA does not mean that cryptography itself is broken. There are some cryptographic systems that we can use as a replacement for the RSA: hash-based cryptography, lattice-based cryptography, secret-key cryptography, etc. So far, there are not any known quantum algorithms that can break these cryptosystems [55]. Grover's search might cause some trouble, but it only offers a quadratic speedup, so using larger keys will be enough to compensate for that.

The question then arises: If RSA is under threat of being broken and we already have cryptosystems that will be able to withstand quantum attacks, why not use them? There are three important reasons for this, namely: we need time to build confidence, improve the efficiency and improve the stability of post-quantum cryptography. In other words, we are still not ready for the post quantum world. We will now take a look at the importance of these reasons.

Confidence

We are usually confident in systems that have survived for a long time. To build confidence in a post-quantum cryptosystem we must give cryptanalysts enough time to gain experience and familiarity with post-quantum cryptography and cryptanalysis, and enough time to review and try to break the cryptosystem.

Efficiency

Current state of the art signing and verification algorithms take time $O(n^2)$ in a signature system that has n-bit signatures and n-bit keys. So far, no post-quantum signature system achieves this feat. Heavy traffic web sites like *Google* cannot afford to use cryptographic protection even with today's fastest systems,

with post-quantum systems it would be even worse. However, there have already been promising advancements in post-quantum cryptography and, given enough time and research, it can hopefully achieve the efficiency of today's cryptographic systems.

Usability

Other than being well-defined and standardized, secure encryptions need to have software and perhaps even hardware implementations, so that they can be integrated into various applications. Their implementations must be correct, fast and avoid leaks. So far, no post-quantum cryptographic system has all of the above features, and if we want them to be usable, that must be achieved.

Hash-based cryptography

As an example of post-quantum cryptography, we take a look at hash-based public-key signature systems [56]. The way hash-based systems work is by generating a public key from the private key using a hash function. A simple example of a hash-based signature system is Lamport's one-time signature system. To generate a key pair, we first choose two random strings of length m, which will use as a private key $X = (x_{10}, x_{11}, x_{20}, x_{21}, ..., x_{m0}, x_{m1})$. The public key will consist of hashed values from the private key $Y = (h(x_{10}), h(x_{11}), h(x_{20}), h(x_{21}), ..., h(x_{m0}), h(x_{m1}))$. With each usage of the same key to sign a different message, the security of the system rapidly declines, which is why we need to use a different key for each new message. For signing multiple messages, we can "chain" multiple one-time signatures or use more advanced signature systems, such as Merkle's hash-tree signature system [57].

The security of these hash-based systems is based on the assumption that finding the input value x from its hashed value $h(x)$ is computationally hard. To recreate the input from its hashed output, we would have to use brute-force search on all possible inputs and find the one that matches the hashed output. That means that, currently, the only quantum algorithm that can pose somewhat of a threat to hash-based systems is Grover's algorithm. However, Grover's algorithm

offers only quadratic speedup, which means that the only thing we would need to change so that we can be safe from a potential "Grover" attack is double the size of the input. This is the reason why hash-based cryptography could be considered safe in the post-quantum world.

4.2.4 Quantum key distribution

As we said before, it is impossible to exchange a key over a public channel in a completely secure way. But by using some of the quantum mechanics principles, we can establish a secure key exchange channel. In the presence of an eavesdropper, the only known secure method for exchanging a key between two communicating parties is quantum key distribution (QKD) [53]. Another advantage of QKD over classical methods is that if we use it in combination with perfectly accurate quantum computer, we can achieve mathematically proven security against all attacks, classical or quantum. And, most importantly, QKD might serve as a vehicle that will help us reach higher goals in quantum computation and communication.

In theory, QKD looks really promising, but real-life QKD security is not perfect and there are known attacks already [54]. Fortunately, all discovered threats can be fixed and, in principle, any QKD system can be made perfectly secure against all attacks. This is the reason for the increased interest in QKD by the big industries, governments and security agencies.

The BB84 protocol

The BB84 is the first quantum key distribution protocol introduced by Charles Bennet and Gilles Brassard in 1984 [58]. To ensure that the key transmission has not been altered or eavesdropped, it uses the uncertainty principle and the no-cloning theorem.

Let us say the eavesdropper Eve is listening for some information, somewhere along the insecure channel. In the classical case she can:

1. Make copy of portions of the encrypted bit stream, store them and analyze them later.

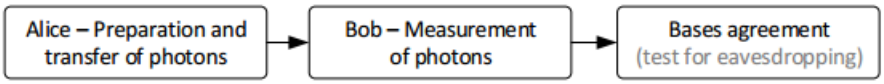
2. Listen without disrupting the bit stream, in other words, she does not leave traces of her eavesdropping.

But if Alice sends qubits instead of bits, through some channel:

1. Because of the no-cloning theorem, Eve cannot make copies of the qubit stream.

2. Eve has to measure the qubit stream to obtain information, but, by doing that, she alters it and can be easily detected.

The way BB84 works is similar to the one-time-pad protocol. Alice generates random key, but instead of normal bits, this time we encode the bits into qubits, using random bases, and send them over a quantum channel. Bob receives the qubits, measures them and stores the result. And, finally, Alice and Bob compare bases to detect if the eavesdropper was listening in.



The way Alice chooses what qubits to send over the quantum channel is really simple. She generates n random binary values (she can do this by flipping a coin) and n random bases (rectilinear or diagonal). And then she sends a qubit according to the randomly generated value and base. For example, she can use the table below to encode the bits into qubits.

		Base	
		Rectilinear	Diagonal
Value	0	0°	45°
	1	90°	135°

Using an example, we can see how BB84 works in more detail.

Step 1. Alice flips the coin 8 times to generate 8 classical bits, and another 8 times to generate 8 bases. Then she encodes the bits into qubits using the table above.

Alice sends n random bits in random bases								
Bit number	0	1	2	3	4	5	6	7
Alice's random bases	⤡	↕	⤡	⤡	↕	⤡	↕	↕
Alice's random bits	0	1	0	1	1	0	1	0
Alice sends	⤡	↕	⤡	⤢	↕	⤡	↕	↔

Step 2. Alice sends the qubits to Bob over quantum channel.

Step 3. Bob receives the qubits, but he does not know the basis in which Alice sent them. So he flips a coin 8 times to generate 8 random bases. Then he measures the qubits in those random bases.

Bob receives n random bits in random measurements								
Bit number	0	1	2	3	4	5	6	7
Bob's random bases	⤡	⤡	↕	⤡	↕	↕	↕	⤡
Bob observes	⤡	⤡	↔	⤢	↕	↕	↕	⤡
Bob's bits	0	0	0	1	1	1	1	0

We can see that the qubits that Alice sent are not equal to the qubits that Bob received. In our example, the first bit that Alice generated was 0 and she measured it in the  basis, so she sent . Bob received the qubit and measured it in the same  basis, which gave him the correct bit, 0. The second bit that Alice generated was 1, which she measured in  basis and sent to Bob. When Bob received the qubit and measured it in the randomly generated  basis, he got the result 0, which is not what Alice wanted to send. The reason why he got the wrong result was that the qubits were measured in different bases. In our case, the qubit was measured in , which puts it in superposition:

$$\oslash = \frac{1}{\sqrt{2}} \updownarrow + \frac{1}{\sqrt{2}} \ominus \tag{4.1}$$

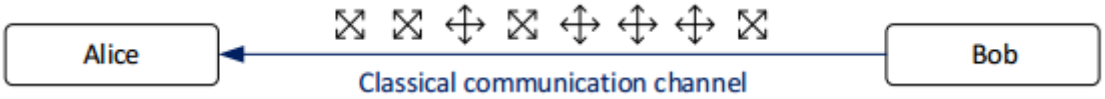
In other words, Bob has a 50-50 chance of getting the right bit after the measurement. There are four possible superpositions we can get by measuring in the wrong basis:

-  measured in  gives us $\frac{1}{\sqrt{2}}$ $-$ $\frac{1}{\sqrt{2}}$
- measured in gives us $\frac{1}{\sqrt{2}}$ $+$ $\frac{1}{\sqrt{2}}$
- measured in gives us $\frac{1}{\sqrt{2}}$ $-$ $\frac{1}{\sqrt{2}}$
-  measured in  gives us $\frac{1}{\sqrt{2}}$ $-$ $\frac{1}{\sqrt{2}}$

Half of the time Bob will generate the same bases as Alice, and that means that he will certainly get the correct bits after the measurement. And the other half of the time he will measure in a different basis. In that case, Bob has a 50% chance of getting the correct bit.

In the next step, Bob and Alice publicly compare their chosen bases. Then they remove the values that were not measured in the same basis and keep the others. Now they are each have the same subsequence of bits, which forms the candidate for the key.

Step 4. Bob reveals measurement bases to Alice over classical channel.



Alice responds stating which bases were CORRECT and which were WRONG.

Alice and Bob remove values for which Bob's measurement bases were wrong.

Bases agreement procedure								
Compliant bases	⤡	⤡	✥	⤡	✥	✥	✥	⤡
Agreed key	0			1	1		1	

The agreed upon key in our example is 0111.

The final step of the protocol is to check if Eve was eavesdropping. To do that, Bob chooses random key bits (usually half of them) and shows them to Alice. Alice then checks if she has the same values. If some of the bits are not equal, the transmission might have been altered and we need to repeat the protocol. If the bits were equal, then we can use the bits as the key.

Even if someone eavesdropped our communication, the protocol is still secure. As we mentioned earlier, because of the no-cloning theorem, Eve cannot make a copy of the qubits and leave the originals intact. In order to extract useful information, Eve has to measure the original qubits. If she measures them in the correct basis, then Alice and Bob will not notice. But if she measures them in an incorrect basis, Alice's and Bob's bits will not be equal and they will know that the communication was eavesdropped.

Challenges

Even though it has its advantages, QKD faces many challenges. First of all, QKD protocols are not packet-based like the ones that the Internet uses today, but they are point-to-point. Secondly, QKD does not handle some of the more important security aspects, like authentication and integrity, so we must implement them classically. Furthermore, QKD requires expensive hardware that is difficult to upgrade and maintain. And, lastly, QKD is a new technology and there might appear undiscovered vulnerabilities in practice. In fact, there are already known attacks to practical implementations of QKD, such as: intercept-resend, photon number splitting, timing attacks, Trojan attacks and other side-channel attacks. These vulnerabilities will never be fatal, but they will require modifications and re-tuning.

Real world implementation

There are already some real-life implementations of QKD. In the last decade, a number of large scale QKD networks have been constructed. To name a few examples: the DARPA Quantum Network which is supported by the US military, an EU project called the SECOQC Quantum Network, which encouraged the European Telecommunications Standards Institute to create a universally accepted QKD standards, the Swiss Quantum Network, the Tokyo QKD network, etc.

A lot of university centers are interested in QKD as well. Some of the more active are: the Group of Applied Physics at Geneva University, who hold the world record for the longest distance QKD through fiber, the Center for Quantum Technologies in Singapore and the Institute of Quantum Computing in Waterloo, which developed successful attacks against QKD, and, finally, one of the most important quantum information centers, The Key Laboratory of Quantum Information in China.

A number of commercial companies already sell QKD devices and systems. MagiQ Technologies from the US sells a QKD system that uses BB84 QKD in combination with a classical encryption, ID Quantique from Europe have a pure QKD product that implements BB84, Quintessence Labs from Australia use QKD

to provide a true random number generator.

With the interest in QKD rising, we can realistically expect large scale quantum networks to be widely used within 10-20 years. And personal hand-held QKD systems even sooner than that.

4.3 Quantum machine learning

Machine learning is a method of data analysis in which the goal is to construct algorithms that can make predictions and learn, given enough data. Machine learning today is all-present. It is used in search engines, image and speech recognition, spam mail filters, self-driving cars, assessing financial risks, pattern identification, etc. In fact, it is used in almost all instances in which we need to interpret data based on experience.

To be able to make good predictions, machine learning has to deal with the use of huge datasets. With the amount of data stored growing faster every year, current computers and machine learning algorithms are pushed to the limit. With the realization of quantum computers, we can potentially make machine learning algorithms a lot more efficient. As mentioned earlier, quantum mechanics restricts us from accessing all of the information in a quantum system, which makes coming up with an efficient quantum machine learning algorithm a difficult problem. However, powerful quantum tools, such as quantum annealing, Grover's search, HHL and other quantum algorithms based on them, already offer improvements over the best known classical machine learning methods. They achieve exponential speed-ups over all classical algorithms that include estimating distances and inner products between vectors. And they might achieve significant speedup for some of the machine learning tasks that include unstructured search or can be solved by quantum annealing. In the image below, we can see that quantum machine learning is an intersection between classical machine learning and quantum information processing.

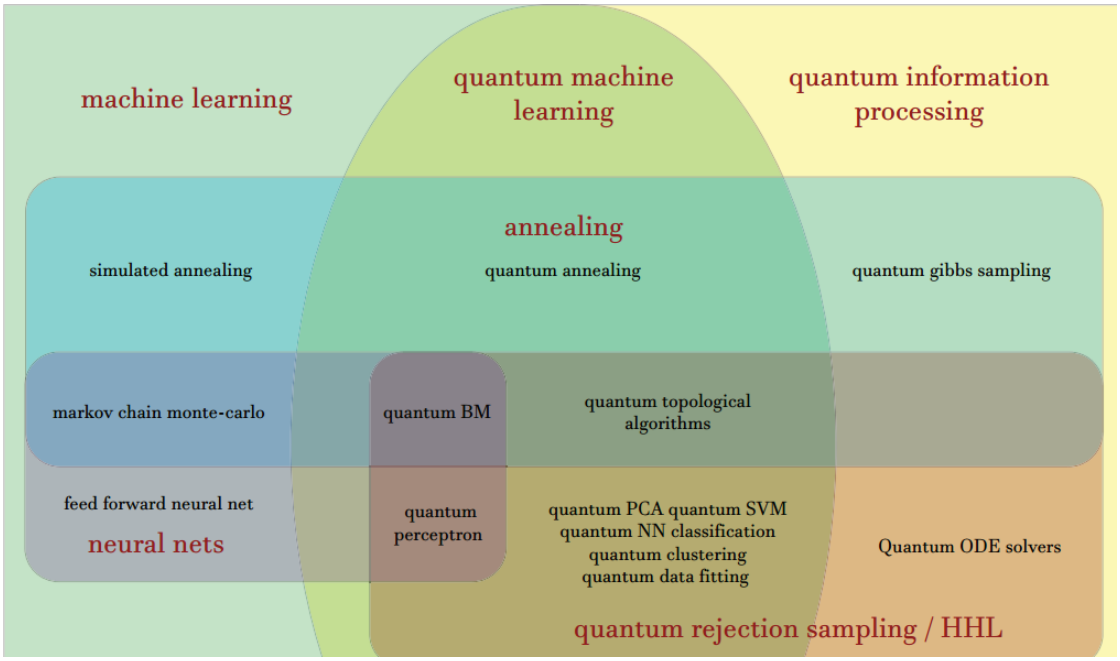


Figure 4.6: Crossover between classical and quantum machine learning.

In the next few subsections, we will see the improvement that quantum computing can bring in some of the most widely used machine learning methods.

4.3.1 Quantum data-fitting

One of the most important tools in quantitative science is fitting data to theoretical models. A theoretical model leads us to functional relations between parameters and the data that depends on them. Fitting a large amount of data allows us to obtain reliable estimates of the parameters. But with large quantities of data, the fitting becomes very costly. For example, experiments at the LHC produce gigabytes of data per second, and to fit that amount of data is a very difficult problem. Advances in quantum information theory provide us with algorithms that might solve this problem efficiently. In 2012, N. Wiebe, D, Braun and S. Lloyd published a paper [59] with a new quantum algorithm that efficiently determines

the quality of a least-squares fit over an exponentially large data set. The algorithm is built upon the previously mentioned HHL algorithm. And in cases where we have data that is efficiently computed by a quantum computer and we address all HHL caveats, the algorithm can achieve exponential speedup.

4.3.2 Quantum nearest-neighbor methods

Nearest-neighbor methods are popular and simple methods for pattern classification. Given a training set T of already classified feature vectors and an unclassified input vector x, the idea is to assign x to a class that appears the most amongst its nearest neighbours. We assume that the nearest feature vectors are the most similar to the input vector.

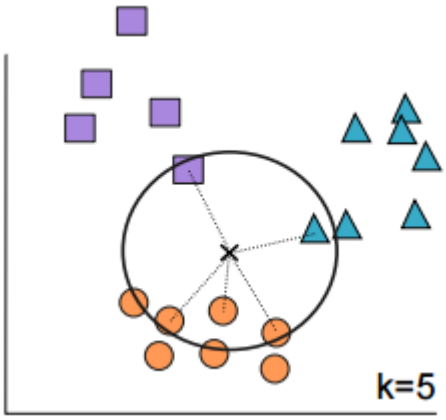


Figure 4.7: An illustration of the k-nearest-neighbor classification, which is a generalization of nearest–neighbor classification. The input vector x to a class that appears the most amongst its k nearest neighbours. In our case the orange circle shape.

Because of their high performance accuracy, they are most commonly used in handwriting recognition, detecting quasars and other problems that involve massive data sets. However, their biggest drawback is their computational expense. This drawback can be mitigated with the help of a quantum computer. Using a

quantum algorithm that uses Grover's search and HHL as a subroutine for computing distances between vectors, we can achieve almost quadratic speedup over the fastest classical algorithms [60].

4.3.3 Quantum algorithms for clustering

Clustering is the task of grouping a set of unclassified feature vectors in such a way that feature vectors in the same cluster are more similar to each other than those in other clusters. It extracts information from the structure of a data set, instead of using training sets for generalization.

The simplest example of clustering algorithm is k-means. It is an algorithm in which we assign feature vectors to their closest current centroid vectors, and on each step, the centroids are recalculated based on the clusters from the previous step. K-means is usually applied to problems where we need to reduce many datapoints into a small number of groups, for example, in data compression.

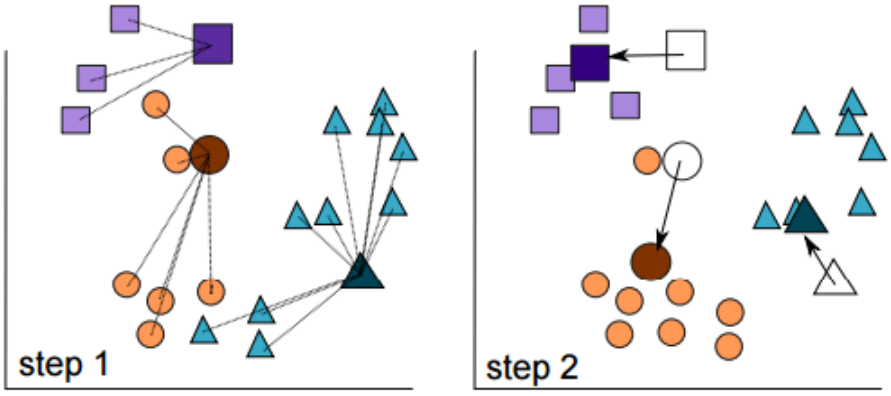


Figure 4.8: Step 1: The clusters are defined by attributing each vector to the closest centroid vector. Step 2: The centroids of each cluster defined in the previous step are recalculated and define a new clustering.

Each step of this algorithm takes time $O(M^2N)$ where M is the number of feature vectors we need to assign to k clusters, and N is the dimension of the

vectors. A quantum k-means algorithm that uses Grover's search and HHL as subroutine will take $O(M \log(MN))$ on each step [61]. By rephrasing the algorithm as a quadratic programming problem, we can find a solution using the adiabatic algorithm, which may be regarded as a subclass of quantum annealing. With this improvement, the algorithm will take time at most $O(k \log(MN))$. However, finding an optimal k-means is an NP-complete problem and we should not except to solve it in polynomial time on a quantum or classical computer. Luckily, we are not interested only in the optimal solution, but also in finding various sets of clustering vectors. This means that the quantum k-means algorithm, which gives us an approximate solution to this hard problem, will suffice in constructing good enough sets and clusters.

4.4 Quantum Simulation

One of the main uses of the earliest classical computers was the simulation of physical systems. Naturally, one of the most important applications of quantum computers might be quantum simulation [62], which classical computers have trouble with. The most obvious problem in simulating quantum systems is the memory needed to store their states. In classical systems, the number of complex numbers used to describe them grows linearly, while in quantum systems it grows exponentially with the size of the system. To describe a quantum system with n distinct components on a classical computer, we would need c^n bits.[3] To describe the same system on a quantum computer, we would need only cn qubits.

There is one big limitation to quantum simulation: even though we can simulate the quantum system efficiently, we cannot obtain all information about the quantum system. In other words, if we measure a quantum simulation with cn qubits, we can only obtain cn bits of information and the remainder of the bits stay "hidden". Despite this problem, quantum simulation may have a lot of important applications. Obtaining a more accurate and faster simulation is of great importance in fields such as quantum chemistry, metamaterials, high-energy physics

[3] c is a constant which depends upon the accuracy and the details of the simulation.

and superconductivity. It will help us understand all systems that have quantum mechanics involved in them. It might even help us discover a new natural phenomenon that cannot be simulated on a quantum computer, which will motivate us to further improve our computational models, beyond the quantum computing model.

While a quantum computer can act like a universal quantum simulator, it is not required to perform a quantum simulation. Many simpler problem-specific quantum devices can be used to imitate the evolution of a quantum system. This fact, combined with the growing interest in quantum simulation and advances in manipulation of quantum systems, might lead us to practical quantum simulators in the very near future, well before the realization of a universal quantum computer. In fact, there are already research groups experimenting with 10-qubit quantum simulators, which could be the first practical applications in which classical simulators are outperformed.

Chapter 5

Summary and Conclusion

In the first section of this final chapter, we will summarize the key points of the three main chapters. In the second section, we will give the conclusion and some final words of this thesis.

5.1 Summary

We began the second chapter by introducing the quantum information bits - qubits. Then we saw how we can use qubits to our advantage by entangling and measuring them. We mentioned what are Bell states and what limitations does the no-cloning theorem bring. We ended up with introducing some of the most used quantum gates and quantum circuits.

The third chapter was an introduction to three of the most important quantum algorithms, namely: Shor's algorithm, Grover's algorithm and HHL. We described the potential of Shor's algorithm, how it works and mentioned its most important building block, QFT (Quantum Fourier Transformation). Then we explained the main idea behind Grover's search algorithm and where can it be applied. And finally, we saw what HHL can do under proper conditions and what its limitations are.

The fourth chapter on Quantum Reality was the main focus of the thesis. In the first section, we saw what a universal quantum computer is, what the challenges

in constructing one are as well as some state of the art implementations, mainly D-Wave's quantum annealer and a trapped-ion quantum computer. The second section showed us what quantum reality would mean to current day cryptography and how Shor's algorithm can be used to break the RSA cryptosystem. Then we introduced post-quantum cryptography, on which we might need to transfer, if a quantum computer becomes reality. And, in the end of the section, we introduced quantum key distribution with an example of such a protocol (BB84), and showed QKD's capabilities and challenges. The next section was dedicated to quantum machine learning, how it can be used to improve machine learning and showed some examples. The final section of this chapter was a brief summary of quantum simulation.

5.2 Conclusion

Throughout this thesis, we described the positive and negative impacts of quantum computing. We also saw the problems which it will bring and how we can counter them. We identified a point where our current classical machine learning barriers can be broken and, additionally, the potential to reach new barriers that even quantum computing might not be able to breach. However, we also recognized their limitations, the challenges they are facing and the reasons why we should not expect magic from them. But, in the end, whatever changes they may bring, we must be prepared for them, because it is getting more and more likely that quantum computers will become a reality.

Bibliography

[1] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010

[2] Noson S. Yanofsky and Mirco A. Mannucci. *Quantum Computing for Computer Scientist*. Cambridge University Press, 2008

[3] N. David Mermin. *Quantum Computer Science an Introduction*. Cambridge University Press, 2007

[4] Scott Aaronson. *Quantum Computing since Democritus*. Cambridge University Press, 2013

[5] Jordan, S. [Online] The quantum algorithm zoo. Available at http://math.nist.gov/quantum/zoo/

[6] Ashley Montanaro. "Quantum Algorithms: an overview", arXiv:1511.04206, 2015

[7] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", arXiv:quant-ph/9508027v2, 1996

[8] L. K. Grover. "A fast quantum mechanical algorithm for database search", arxiv:quant-ph/9605043, 1996

[9] Aram W. Harrow, Avinatan Hassidim and Seth Lloyd. "Quantum algorithm for linear systems of equations", arXiv:0811.3171, 2009

[10] R. L. Rivest, A. Shamir and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM 21.2, pp. 120-126, 1978

[11] L. Hales and S. Hallgren. "An improved quantum Fourier transform algorithm and applications", Proc. 41st Ann. Symp. on Foundations of Computer Science, pp. 515–525, 2000

[12] Andris Ambainis. "Quantum Search Algorithms", arXiv:quant-ph/0504012, 2005

[13] George F. Viamontes, Igor L. Markov and John P. Hayes. "Is Quantum Search Practical?", arXiv:quant-ph/0405001, 2004

[14] L. K. Grover. "Quantum mechanics helps in searching for a needle in a haystack", Phys. Rev. Lett., 79:325, 1997

[15] D.W. Berry, G. Ahokas, R. Cleve, and B.C. Sanders. "Efficient Quantum Algorithms for Simulating Sparse Hamiltonian", arXiv:quant-ph/0508139, 2007

[16] A.M. Childs. "On the relationship between continuousand discretetime quantum walk", arXiv:0810.0312, 2008

[17] A. Luis and J. Peřina. "Optimum phase-shift estimation and the quantum description of the phase difference", Phys. Rev. A, 54(5):4564–4570, 1996

[18] Scott Aaronson. "Quantum machine learning algorithms: Read the fine print", Nat. Phys. **11**, pp. 291–293, 2015

[19] M. I. Dyakonov. "State of the art and prospects for quantum computing", arXiv:1212.3562, 2012

[20] N. Gisin, G.Ribordy, W. Tittel, and H. Zbinden. "Quantum cryptography", Reviews of modern physics, 74(1), p. 145, 2002

[21] Maria Schulda, Ilya Sinayskiy, and Francesco Petruccione. "An introduction to quantum machine learning", arXiv:1409.3097, 2014

[22] Vedran Dunjko, Jacob M. Taylor and Hans J. Briegel. "Quantum-Enhanced Machine Learning", Phys. Rev. Lett. **117**, 130501, 2016

[23] I. M. Georgescu, S. Ashhab and Franco Nori. "Quantum Simulation", arXiv:1308.6253, 2014

[24] Tomi H Johnson, Stephen R Clark and Dieter Jaksch. "What is a quantum simulator?", EPJ Quantum Technology **1**:10, 2014

[25] R. P. Feynman. "Simulating physics with computers", International journal of theoretical physics 21.6, pp. 467-488, 1982

[26] D. P. DiVincenzo. "The physical implementation of quantum computation" quant-ph/0002077, 2000

[27] B. Tamir, E Cohen. "Notes on Adiabatic Quantum Computers", arXiv:1512.07617v4, 2006

[28] Amira M. Eltony, Dorian Gangloff, Molu Shi, Alexei Bylinskii, Vladan Vuletić and Isaac L. Chuang. "Technologies for trappedion quantum information systems", arXiv:1502.05739, 2016

[29] M. H. Devoret, A. Wallraff and J. M. Martinis. "Superconducting Qubits: A Short Review", arXiv:cond-mat/0411174, 2004

[30] J. M. Gambetta, J. M. Chow and M. Steffen. "Building logical qubits in a superconducting quantum computing system", arXiv:1510.04375, 2015

[31] Q. A. Turchette, C. J. Hood, W. Lange, H. Mabuchi, and H. J. Kimble. "Measurement of conditional phase shifts for quantum logic", Phys. Rev. Lett. **75**, 4710, 1996

[32] Tameem Albash, Itay Hen, Federico M. Spedalieri and Daniel A. Lidar. "Reexamination of the evidence for entanglement in a quantum annealer", arXiv:1506.03539, 2015

[33] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis and H. Neven. "What is the Computational Value of Finite Range Tunneling?", arXiv:1512.02206v4, 2016

[34] D. Ceperley and B. Alder. "Quantum monte carlo", Science 231, 1986

[35] Alex Selby. [Online] QUBO-Chimera. Available at https://github.com/alex1770/QUBOChimera

[36] J. King, S. Yarkoni, M. M. Nevisi, J. P. Hilton and C. C. McGeoch. "Benchmarking a quantum annealing processor with the timetotarget metric", arXiv:1508.05087, 2015

[37] E. Boros, P. L. Hammer and G. Tavares. "Local search heuristics for Quadratic Unconstrained Binary Optimization (QUBO)", Journal of Heuristics 13, 2, pp. 99-132, 2007

[38] Vicky Choi. "Minorembedding in adiabatic quantum computation: I. The parameter setting problem", Quantum Information Processing 7.5, pp. 193-209, 2008

[39] V. N. Smelyanskiy, E. G. Rieffel, S. I. Knysh, C. P. Williams, M. W. Johnson, M. C. Thom and K. L. Pudenz. "A near-term quantum computing approach for hard computational problems in space exploration", arXiv:1204.2821, 2012

[40] E. D. Dahl. [Online] Programming with DWave: Map Coloring Problem. Available at https://www.dwavesys.com/sites/default/files/Map%20Coloring%20WP2.pdf

[41] R. Dridi, H. Alghassi. "Prime factorization using quantum annealing and computational algebraic geometry", arXiv:1604.05796, 2016

[42] S. Gulde, M. Riebe, G. P. Lancaster and C. Becher. "Implementation of the DeutschJozsa algorithm on an ion-trap quantum computer", Nature, 421(6918), p. 48, 2003

[43] T. Monz, D. Nigg, E. A. Martinez, M. F. Brandl, P. Schindler, R. Rines, S. X. Wang, I. L. Chuang and R. Blatt. "Realization of a scalable Shor algorithm", arXiv:1507.08852, 2015

[44] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright and C. Monroe. "Demonstration of a small programmable quantum computer with atomic qubits", Nature **536**, pp. 63-66, 2016

[45] B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich and W. K. Hensinger. "Blueprint for a microwave trapped ion quantum computer" Sci. Adv. 3, e1601540, 2017

[46] J. Bardeen, L. N. Cooper and J. R. Schrieffer. "Theory of superconductivity", Physical Review, 108(5), p. 1175, 1957

[47] L. DiCarlo, J. M. Chow, L. S. Bishop, et al. "Demonstration of twoqubit algorithms with a superconducting quantum processor", arXiv:0903.2030, 2009

[48] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright and C. Monroe. "Experimental Comparison of Two Quantum Computing Architectures", arXiv:1702.01852, 2017

[49] C. Song, K. Xu, W. Liu, et al. "10-qubit entanglement and parallel logic operations with a superconducting circuit", arXiv:1703.10302, 2017

[50] V. Lahtinen and J. K. Pachos. "A Short Introduction to Topological Quantum Computation", arXiv:1705.04103, 2017

[51] A. Roy and D. P. DiVincenzo. "Topological Quantum Computing", arXiv:1701.05052, 2017

[52] J. Sau. "A Roadmap for a Scalable Topological Quantum Computer.", Physics 10 (2017): 68, 2017

[53] Daniel D. Moskovich. "An overview of the state of the art for practical quantum key distribution", arXiv:1504.05471, 2015

[54] N. Jain, B. Stiller, I. Khan, D. C. Marquardt and G. Leuchs. "Attacks on practical quantum key distribution systems (and how to prevent them)", arXiv:1512.07990v2, 2016

[55] Daniel J. Bernstein, Johannes Buchmann and Erik Dahmen. *Post-Quantum Cryptography*. Springer, 2009

[56] Daniel J. Bernstein and Tanja Lange. "Post-quantum cryptography-dealing with the fallout of physics success", IACR Cryptology ePrint Archive 2017, p. 314, 2017

[57] M. Szydlo. "Merkle tree traversal in log space and time" In Eurocrypt Vol. 3027, pp. 541-554, 2004

[58] C. H. Bennett and G. Brassard.'Quantum cryptography: Public key distribution and coin tossing". In Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, volume 175, p. 8, 1984

[59] Nathan Wiebe, Daniel Braun and Seth Lloyd. "Quantum Data-Fitting", arXiv:1204.5242, 2012

[60] Nathan Wiebe, Ashish Kapoor and Krysta M. Svore. "Quantum Algorithms for Nearest-Neighbor Methods for Supervised and Unsupervised Learning", arXiv:1401.2142, 2014

[61] Seth Lloyd, Masoud Mohseni, Patrick Rebentrost. "Quantum algorithms for supervised and unsupervised machine learning", arXiv:1307.0411, 2013

[62] J. I. Cirac and P. Zoller. "Goals and opportunities in quantum simulation", Nature Physics, 8(4), pp. 264-266, 2012