HAAGA-HELIA ammattikorkeakoulu

Quantum computing

Markus Kiili

Opinnäytetyö

Tietojenkäsittelyn koulutusohjelma

2014

HAAGA-HELIA
ammattikorkeakoulu

Tiivistelmä

25.5.2014

Tietojenkäsittelyn koulutusohjelma

Tekijä tai tekijät Markus Kiili	**Ryhmätunnus tai aloitusvuosi** 2013
Raportin nimi Kvanttilaskenta	**Sivu- ja liitesivumäärä** 47 + 3
Opettajat tai ohjaajat Ahti Kare	

Tämän opinnäytetyön tavoite oli selittää, mitä on kvanttilaskenta.

Informaatio työtä varten kerättiin kirjoista, tieteellisistä julkaisuista ja uutisartikkeleista.

Informaation analysointi osoitti, että kvanttilaskenta voidaan jakaa kolmeen osa-alueeseen: kvanttitietokoneen rakenteen selittäviin teorioihin, tunnettuihin kvanttialgoritmeihin ja varsinaisiin kvanttitietokoneen fysikaalisiin toteutuksiin.

Lopputyössä selviää, että siirtyminen klassisista muistibiteistä kvanttimekaniikkaa noudattaviin bitteihin, mahdollistaa monimutkaisempien operaatioiden suorittamisen. Niiden avulla voidaan muodostaa uusia algoritmeja, jotka näyttävät ratkaisevan eräitä tärkeitä ongelmia klassista laskentaa nopeammin.

Kvanttitietokoneet osaavat etsiä tietoa ja ratkaista matemaattisia Fourier muunnoksia selvästi nopeammin kun klassiset tietokoneet, minkä takia ne pystyvät murtamaan moderneja salausjärjestelmiä.

Kvanttibittien kvanttimekaaninen luonne johtaa myös uusiin ongelmiin. Kvanttibitit ovat hyvin epävakaita ja kvanttilaskennan monimutkainen luonne tekee uusien kvanttialgoritmien kehittämisen vaikeaksi.

Vain hyvin pieniä muutaman kvanttibitin kokoisia kvanttitietokoneita on rakennettu. Teorioita suurista kvanttitietokoneista ja kvanttiohjelmoinnista on kehitetty, mutta ei tiedetä, onko suurten kvanttitietokoneen rakentaminen mahdollista. Kvantti-ilmiöt tapaavat pysytellä mikroskooppisilla etäisyyksillä ja saattaa olla fysikaalisia esteitä, jotka estävät suurten kvanttitietokoneiden rakentamisen.

Asiasanat
Laskenta, tietokoneet, kvanttimekaniikka, kvanttiteoria, salaus.

HAAGA-HELIA
University of Applied Sciences

Abstract

25 May 2014

Degree Programme in Information Technology

Authors Markus Kiili	**Group or year of entry** 2013
The title of thesis Quantum computing	**Number of report pages and attachment pages** 47+3
Advisor(s) Ahti Kare	

The aim of this thesis was to explain what quantum computing is.

The information for the thesis was gathered from books, scientific publications, and news articles.

The analysis of the information revealed that quantum computing can be broken down to three areas: theories behind quantum computing explaining the structure of a quantum computer, known quantum algorithms, and the actual physical realizations of a quantum computer.

The thesis reveals that moving from classical memory bits to quantum bits obeying the laws of quantum mechanics allows more complicated operations to be performed. This leads to new algorithms that seem to outperform classical computing on some important problems.

Quantum computers can search for information and calculate mathematical Fourier transforms much faster than classical computers, which allows them to break modern encryption techniques.

The quantum mechanical nature of quantum bits also leads to new challenges. The quantum bits are very unstable and the complicated nature of quantum computing makes it very hard to figure out new quantum algorithms.

Only very small quantum computers, consisting of a few quantum bits, have been built. Theories about large quantum computers and quantum programming languages are being studied, but it is not known whether or not it is possible to build a large quantum computer. Quantum phenomena tend to stay on microscopic distances and there might be physical barriers that prevent large quantum computers from being built.

Key words
Quantum computer, computation, quantum mechanics, encryption.

Contents

1 Preface

Computers are computing devices that manipulate data stored in bits that obey the laws of classical mechanics. However, the fundamental theories of physics that govern the structure of matter, are theory of relativity and quantum mechanics. Relativity explains gravity, the structure of the universe, and phenomena that occurs when objects are moving with velocities close to the speed of light. Quantum mechanics explains the dynamics of low energy objects of subatomic scales. Relativity is not likely to offer too much to computing because interstellar computing or high energy computing are in conflict with the usual idea of stable and local computer memory. In the other hand, it is quite reasonable to ask whether it is possible to build a computer manipulating data stored in medium governed by the laws of quantum mechanics. The progress of making smaller and smaller classical electric circuits is already reaching the limit where quantum effects start to disturb the circuits.

Quantum computers are interesting because they seem to expand the boundaries of computing. Some redoubts of important unsolvable problems crumble when a quantum computer sets its eye on them. At the moment, many uses have been figured out for a quantum computer, but only very small quantum computers have been built. What will change if a large quantum computer is one day built? Has it already been done?

This thesis aims to explain quantum computing to a reader with no knowledge on quantum mechanics. Theoretical section is limited to subjects that are needed in order to understand how a quantum computer works. The influence of quantum computing on other fields, like the theory of computation, information theory, and quantum communication are mostly left out.

Basic terms of theory of computation and quantum mechanics are defined and explained in the theoretical section, so the reader can, in principle, understand everything written in the thesis. The red line is always to show how quantum computing differs from classical computing and how the more complicated structure of quantum bits leads to more complicated basic logical processes and new algorithms.

Theoretical history of quantum computing is reviewed, giving the credit to those who have developed the theories behind quantum computing. Thesis also reviews the actual physical experiments where quantum computation has been performed.

Terminology in this thesis used in describing quantum computing and quantum computers is so, that quantum computing and quantum computation are synonymous, meaning the area of research studying computer technology based on quantum mechanical properties of matter. The actual device doing the computation is a quantum computer. If a quantum computer can run any quantum algorithm, it is called an universal quantum computer. If a quantum computer can be programmed with a programming language, it is called a programmable quantum computer.

The process of writing the thesis started in the fall of 2013, when I, amongst other new students, was starting my studies at Haaga-Helia. We were told that we should start to think about the thesis so it would not slow our graduation at the end of our studies. I took it literarily, and at the end of 2013, I had my thesis subject chosen. I didn't know anything about the subject then, but had studied theoretical physics in the past. I started to seek out everything written about quantum computing. I gathered books, scientific publications, and news articles and tried to understand the subject. At the spring of 2014, I thought that I understood the subject and wrote the thesis.

2 Classical computation

This chapter covers the basics of classical theory of computation. In order to understand quantum computation in comparison to classical computation, the reader must be familiar with computational terminology, different computational models, and the basic logical processes of computation. The circuit model is explained in more detail, because it is the model used in generalizing classical computation to quantum computation.

2.1 Theory of computation and complexity of algorithms

A computational problem is a problem that a computer might try to solve. A solution to a computational problem is called an algorithm. It is a set of instructions that tells how to solve the problem. A measure for the difficulty of a computational problem is how much resources are needed to perform the calculation with the best known algorithm. Although energy and memory needed are also resources, the most important resource used in classifying computational problems is how much time it takes to reach the solution. Time taken is proportional to the number of calculational steps which is usually proportional to size of the input data. Because we are especially interested in how the time taken behaves as the input data size grows, the difficulty of a problem is usually given as an expression, which tells how the time taken behaves asymptotically, as the size of the input data size is large. A major distinction is made by finding out whether the dependence is polynomial or more than polynomial (usually exponential). Problems taking polynomial time are called easy problems and problems taking exponential time are called hard problems. Easy problems are also called computable or efficiently computable problems. In many cases hard problems have no useful algorithm and must be solved by a brute force search. A central principle of classical computing is that solving a problem is hardware independent. Easy problems are easy and hard problems hard on all computers. (Pathak 2013, 33-44.)

To show the difference between easy and hard problems, an example of two problems is reviewed. First one is easy taking time $T = 0.5N^3 + 3N$ to run the algorithm, where N is the size of the input data in bits. For large N, $N^3 \gg N$ and thus asymptotically

$T = 0.5N^3$. This asymptotic behavior is denoted $T = O(N^3)$. It shows the meaningful behavior of the relationship as the input data size increases. In this case, if the input data size is changed from 100 to 10000, the time taken would roughly be million (10^6) times longer. The second problem is hard taking $T = 2^N$. If the input data size is changed from 100 to 10000, the time taken is changed from $2^{100} \sim 10^{30}$ to $2^{10000} \sim 10^{3010}$. The time taken is roughly 10^{2980} times longer. This shows how easily resources can run out solving hard problems.

In chapter four it is shown that some important problems that are hard on classical computers turn out to be easy on quantum computers.

2.2 Models of computation

There are different mathematical models that describe computation. They differ by the basic operations that can be performed. In respect of computability, they are all equivalent formulations. Few of the most important ones are:

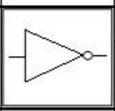
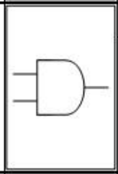
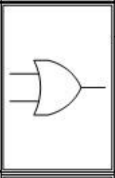
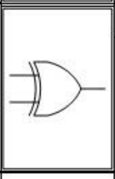
- Turning machine. Used mainly as the main model of complexity theory.
- Lambda calculus. Forms the basis of many functional programming languages.
- Boolean circuits. Uses logical gates to perform operations on memory bits.
- Random-access machine (RAM). Consists of memory registers holding integer numbers and a program which is a list of simple commands for arithmetic, logical, comparing and jump operations.
- Universal programming languages. Artificial languages which are used for creating programs that express algorithms to a computer.

Logical circuits capture the essential structure of electrical components. It is a good model to express small algorithms. RAM-model resembles the structure of modern computers and, in contrast to the Turing model, has the benefit that any memory register can be accessed directly. For expressing large complicated algorithms and programs, programming languages are needed. (Miszczak 2011, 2-3.)

2.3 Circuit model

The classical circuit model is a simple and realistic model of computation and it shows the basic processes of classical computation. Also, the simplest way to explain quantum computing, is to generalize circuit model into quantum circuit model. Circuits are made of bit registers, wires and logical gates. Gates transform input bits into output bits and the functionality of the gate can be expressed showing how the bits transform. Some of the most important gates are shown in the table (Table 1).

Table 1. Logical gates

Gate	Picture	INPUT			OUTPUT		
NOT		a			NOT a		
		0			1		
		1			0		
AND		a	b		a AND b		
		0	0		0		
		0	1		0		
		1	0		0		
		1	1		1		
OR		a	b		a OR b		
		0	0		0		
		0	1		1		
		1	0		1		
		1	1		1		
XOR		a	b		a XOR b		
		0	0		0		
		0	1		1		
		1	0		1		
		1	1		0		
TOFFOLI		a	b	c	a'	b'	c'
		0	0	0	0	0	0
		0	0	1	0	0	1
		0	1	0	0	1	0
		0	1	1	0	1	1
		1	0	0	1	0	0
		1	0	1	1	0	1
		1	1	0	1	1	1
		1	1	1	1	1	0

Because circuit input and output data are in bits, a circuit defines a function (Equation 1).

$$f: \{0,1\}^n \rightarrow \{0,1\}^m$$

Equation 1. Function defined by a circuit

It can be shown that using only the first three gates from the table (Table 1), any such function and so any circuit can be constructed. Therefore the group $\{$NOT, AND, OR$\}$ is called an universal gate set. The TOFFOLI gate, which flips the third bit if the first two bits are 1, is an example of a gate that is universal by itself. (Katzgraber & Renner 2008, 5-13.)

Usually, when gates operate on bits, some information is lost. If the input state of the bits can be retrieved from the output state, it is said that the gate is reversible. Heat production in electric circuits is due to the information loss and therefore reversible circuits produce no heat (Landauer 1961, 183-191.) Of the five gates in the table (Table 1), only NOT and TOFFOLI are reversible. Therefore, a typical circuit made out of the universal gate set $\{$NOT, AND, OR$\}$, will produce heat. There are many reversible universal gate sets like $\{$TOFFOLI$\}$ and the study of heat loss free circuits is ongoing. (Drechsler & Wille 2012.)

3 Quantum computing

A quantum computer stores data in qubits instead of classical bits. This chapter covers the basics of quantum mechanics, so the reader can understand what qubits are and what operations can be performed on qubits. Different computational models are reviewed and the basic logical processes are explained using the quantum circuit model. Data stored in qubits is unstable, because interaction with the environment disturbs the states of the qubits. Error correction is therefore an essential part of quantum computation leading to fault tolerant quantum computing.

3.1 Quantum mechanics

Quantum mechanics describes dynamics of microscopic particles with low energies. This means that the energies are not so large that particle production from kinetic energy is meaningful. Different physical states of a quantum mechanical system are described by a vector in a complex linear space called a Hilbert space.

3.1.1 Complex numbers $\mathbb{C}$

Complex numbers have the following properties (Equation 2).

$$
\begin{aligned}
&Complex\ number\ \alpha = a + ib, where\ a\ and\ b\ are\ real\ numbers\\
&i\ is\ the\ imaginary\ unit\ satisfying\ i^2 = -1\\
&Conjugate\ \alpha^* = a - ib\\
&Modulus\ |\alpha| = \sqrt{a^2 + b^2}\\
&Euler's\ formula\ e^{ia} = \cos(a) + i\sin(a)
\end{aligned}
$$

Equation 2. Complex numbers

3.1.2 State space

State of a quantum mechanical system means all the information about the system. States form a complex linear space called a Hilbert space. It means, that states can be multiplied by complex numbers and added together forming new states. There is also an inner product between the states, which transforms two states into a complex number (Equation 3).

$$\langle \psi | \vartheta \rangle \rightarrow \mathbb{C}$$

Equation 3. Inner product

The inner product defines the notions of orthogonality and distance in the state space. Later it is shown that inner product is also used in the probabilistic interpretation of quantum mechanics. States are said to be orthogonal, if the inner product is zero. Usually it is easiest to work with normalized states satisfying the normalization condition (Equation 4).

$$\langle \psi | \psi \rangle = 1$$

Equation 4. Normalized state

If a state is multiplied with a complex number, the result represents the same physical state (Equation 5).

$$| \psi \rangle \sim \alpha | \psi \rangle$$

Equation 5. Physically meaningless coefficient

Complex coefficients of states become meaningful when states are added together. (Sakurai & Napolitano 2011, 10-14.)

3.1.3 Eigenvalues

Usually it is needed to measure something about the system. Every observable quantity has a set of possible real values for the measurement, called eigenvalues (Equation 6).

$$A = a_1, a_2, \dots, a_n$$

Equation 6. Eigenvalues

.

The spectrum of the eigenvalues can be discrete or continuous. For example, the energy spectrum of a system of particles in a finite size space is discrete, but the spectrum of possible locations of particles is continuous. The allowed energy values of a discrete energy spectrum are separated by small intervals, or quantums, hence the name quantum mechanics. For every eigenvalue, there exist a state where it is certain that we will measure exactly the specific eigenvalue. These states are called eigenstates and denoted as (Equation 7) (Sakurai & Napolitano 2011, 12.)

$$|a_1\rangle, |a_2\rangle, \dots, |a_n\rangle$$

Equation 7. Eigenstates

3.1.4 Basis and superposition

Eigenstates of an observable form a basis for the state space. It means that in addition to the eigenstates, where we are certain to measure the corresponding eigenvalue, we can also have states that are superpositions of those eigenstates. A general state can then be expressed as a complex linear combination of the eigenstates (Equation 8).

$$|\psi\rangle = \alpha_1|a_1\rangle + \alpha_2|a_2\rangle + \cdots + \alpha_n|a_n\rangle$$

Equation 8. General state

When there is a basis, an inner product of two states can be calculated (Equation 9) (Sakurai & Napolitano 2011, 17-18.)

$$\begin{aligned}\langle\psi|\vartheta\rangle &= \langle(\alpha_1|a_1\rangle + \alpha_2|a_2\rangle + \cdots + \alpha_n|a_n\rangle)|(\beta_1|a_1\rangle + \beta_2|a_2\rangle + \cdots + \beta_n|a_n\rangle)\rangle \\ &= \alpha_1^*\beta_1 + \alpha_2^*\beta_2 + \cdots + \alpha_n^*\beta_n \,.\end{aligned}$$

Equation 9. Inner product expansion

3.1.5 Measurement

When a measurement is made, it will give a result a_i with a probability (Equation 10),

$$P(a_i) = |\langle a_i|\psi\rangle|^2 = |\alpha_i|^2$$

Equation 10. Measurement probabilities

showing that the probability to measure each eigenvalue is calculated by squaring the modulus of the eigenstate coefficient. These probabilities must add up to value one (Equation 11),

$$|\alpha_1|^2 + |\alpha_2|^2 + \cdots + |\alpha_n|^2 = 1$$

Equation 11. Total probability

because when a measurement is made, it will always give exactly one result. When the measurement is made, and the result is a_i, the state changes (Equation 12).

$$|\psi\rangle \rightarrow |a_i\rangle$$

Equation 12. Collapse of the state

Measurement in quantum mechanics usually changes the state of the system. (Sakurai & Napolitano 2011, 23-24.)

As an example, a state shown in the equation (Equation 13) can be considered.

$$|\psi\rangle = \frac{2}{3}|0\rangle + \frac{2+i}{3}|1\rangle$$

Equation 13. Example state

The probabilities for measuring 0 and 1 are calculated by squaring the modulus of the corresponding coefficient (Equation 14) using the properties of complex numbers (Equation 2).

$$P(0) = \left|\frac{2}{3}\right|^2 = \left(\frac{2}{3}\right)^2 = \frac{4}{9}$$

$$P(1) = \left|\frac{2+i}{3}\right|^2 = \left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2 = \frac{4}{9} + \frac{1}{9} = \frac{5}{9}$$

Equation 14. Calculation of the example

This also shows that the state is normalized because the probabilities add up to one.

The state of a system is an abstract object, which is not directly observable. The system can be prepared to a state and its time development follows deterministic equations. The state of the system can be found out by preparing the system many times and making many different measurements. Then the coefficients of the eigenstates can be found out and therefore the state is known with some margin of error. The situation is different if the calculation shows that the state is certainly in one of the eigenstates and not in superposition of states. Then the state can be found out by making only one measurement. (Sakurai & Napolitano 2011, 10-24.) System in an eigenstate resembles classical mechanics where observables have fixed values.

3.1.6 Time development

Time development of an eigenstate corresponding to energy e_i is (Equation 15),

$$|e_i\rangle \rightarrow e^{\left(\frac{-ie_it}{\hbar}\right)}|e_i\rangle$$

Equation 15. Time development of energy eigenstates

where time is denoted with t and ħ is the planck constant divided by 2π. Time development is unitary, which means that it is essentially a rotation in the state space. No information is lost and it is always possible to find out en earlier state by rotating the state back. (Sakurai & Napolitano 2011, 66-73.) In quantum computing, physical systems used as qubits are usually chosen so, that the states corresponding to 0 and 1 have the same energy, and thus there is no real time development. Time development only happens when the states are operated on in a controlled way, for example with quantum gates.

Measurement is a special process, because it doesn't follow the usual rules of motion and conservation of probabilities. Measurement loses information and therefore is not an unitary process. (Pathak 2013, 74-75.)

3.1.7 Entanglement

In systems with several particles, states can be entangled so that measurements of one particle affect measurements of another particle. For example, if there are two particles that have two possible values 0 and 1 when some observable is measured, and the system is prepared in a state (Equation 16),

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Equation 16. Bell state

where the first number in the state refers to the eigenstate of the first particle and the second number to the second particle. Something strange happens if the particles are separated far away from each other after preparing them. Measurement of either of the particles will give a random result of 0 or 1 with equal ½ probabilities, because both eigenstates have the same $\frac{1}{\sqrt{2}}$ coefficient. However, measuring of one of the particles will instantly fix the future measurement of the other particle to the same value. The curious thing about this experiment is that the measurement of one particle instantly affects the other particle far away. (Pathak 2013, 81-88.) This famous experiment was proposed in 1964 by John Stewart Bell. It shows that quantum mechanics is in conflict with quite natural assumptions of reality and locality. Reality means that physical observables have some predefined values waiting to be measured. Locality means that changes affect only nearby things. (Bell 1964.) The Bell experiment was based on a thought experiment made in 1935 by Albert Einstein, Boris Podolsky and Nathan Rosen (Einstein, Podolsky & Rosen 1935). The Bell experiment was done experimentally in 1982 (Aspect, Dalibard & Roger 1982).

3.2 Qubits

If an observable in a quantum mechanical system has only two possible values, such a system is called a qubit. It is the simplest of all quantum systems. These two eigenstates can be named $|0\rangle$ and $|1\rangle$ and a general state can be given as a complex linear combination (Equation 17) (Valiron 2012a, 2-3.)

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

Equation 17. Qubit

Using the equation for measurement probabilities (Equation 10), the probabilities for measuring 0 and 1 become $|\alpha|^2$ and $|\beta|^2$, provided that the state is normalized. This means that the state of a system is not directly observable. By preparing and measuring the superpositional state many times, a better approximation of the coefficients α and β is gotten. If it is somehow known that the system is not in superposition, but in one of the eigenstates, then the state can be found out with just one measurement.

One example of a physical system that can be used as a qubit is angular momentum of an electron. If the angular momentum of the electron is measured along some axis, the result is always either $+\hbar/2$ or $-\hbar/2$. (Sakurai & Napolitano 2011, 3.)

A qubit is a vector in 2-dimensional Hilbert space. If the basis vectors are denoted as (Equation 18),

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$$
$$|1\rangle = \begin{bmatrix}0\\1\end{bmatrix}$$

Equation 18. Basis vectors

a general state can be written as (Equation 19).

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$$

Equation 19. General state as a vector in Hilbert space

States of a qubit are parametrized by two complex parameters which means four real parameters. Because states should be normalized (Equation 4) and a constant factor multiplying a state is meaningless (Equation 5), all these parameters are not really physical. Different physical states of a qubit can be parametrized by two continuous real parameters. These parameters turn out to be angles that specify points on an unit sphere, called the Bloch sphere (Figure 1). (Valiron 2012a, 3-6.)

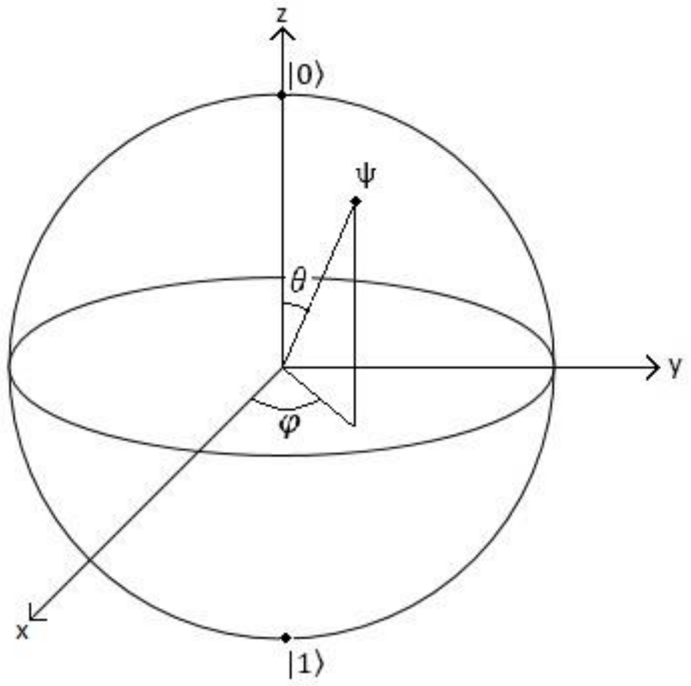


Figure 1. Bloch sphere

The Bloch sphere shows the difference in complexity between a classical bit and a qubit. A classical bit is represented by just the north and south poles, while the states of a qubit cover the whole unit sphere.

For two qubits, a 4-dimensional basis can be created (Equation 20),

$$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$$

Equation 20. Two-qubit basis

where the first number in a state vector refers to the eigenvalue of the first particle and the second number to the second particle. For example, if the system is in the state $|01\rangle$, the measurement will always give 0 for the first particle and 1 for the second particle. A general 2-particle state has four complex parameters for the four basis vertors (Equation 21).

$$|\psi\rangle = \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \xi \end{bmatrix}$$

Equation 21. General vector in 4-dimensional Hilbert space

For three qubits 8 basis states are needed (Equation 22),

$$\{|000\rangle, |001\rangle, |010\rangle, |011\rangle|100\rangle, |101\rangle, |110\rangle, |111\rangle\}$$

Equation 22. Three-qubit basis

and the general state is a 8-dimensional vector. For n qubits, the state is parametrized by 2^n complex parameters forming a vector in a 2^n -dimensional Hilbert space. (Valiron 2012a, 7-11.)

Operators in linear spaces are represented by square matrices. An operator in n-dimensional Hilbert space transforms vectors to vectors (Equation 23),

$$|\psi\rangle_{(n\times 1)} \rightarrow A_{(n\times n)}|\psi\rangle_{(n\times 1)}$$

Equation 23. Hilbert space operator

where dimensions are marked in the subscript. (Valiron 2012a, 2-10.) This means that while classical operations transform input bits into output bits, quantum operations change the states of the qubits by modifying the coefficients of the basis states according to the operation matrix.

Transpose matrix and adjoint matrix are defined by (Equation 24).

$$A_{ij}^{T} = A_{ji}$$

$$A^{+} = (A^{*})^{T}.$$

Equation 24. Transpose and adjoint matrices

Adjoint matrix is made by conjugating each element and then transposing the matrix. A matrix A is unitary if fulfills the unitarity condition (Equation 25),

$$A^{+}A = I,$$

Equation 25. Unitary matrix

where I denotes a unit matrix that does nothing to the operated vector. The fact, that unitary operations are reversible, is clear by the definition of unitarity (Equation 25), because there always exists an adjoint operation that reverses the operation. (Pathak 2013, 66-67.)

3.3 Quantum information

The state of a classical bit can be specified with one boolean variable, and the state of n bits with n boolean variables or one integer variable. The state of one qubit is specified with two continuous real variables or infinite number of boolean variables. Therefore, one might argue that the information content of a qubit is infinitely larger than that of a classical bit. The situation in not so clear, because the quantum state is not directly observable, and when a measurement is made, only one bit of information is gotten out from the measurement. (Girvin 2013, 10.)

It is later shown that although qubits are much more complicated than bits and there are more basic processes that can be performed on qubits which leads to more possible algorithms, and quantum calculations take advantage of entanglement and superposition, there exist only a few cases where major speedup over classical calculation is obtained by a quantum algorithm. The problem is that quantum calculations usually start with qubits on eigenstates and the states are then mixed up with operations. In order to get the results out, the state must somehow be transformed to a state where the qubits that contain the result are again in eigenstates.

The advantage of using superposition can be seen by considering a quantum process similar to classical calculation of boolean function $f\colon \{0,1\} \to \{0,1\}$. Because this process is not reversible, an unitary process must be constructed, which describes this classical function. Such process is called a quantum oracle. The unitary process corresponding to calculating the function f is $U(f)$ defined for the basis states $x, y \in \{0,1\}$ by (Equation 26),

$$|x\rangle \to |x\rangle$$
$$|y\rangle \to |y +_2 f(x)\rangle$$

Equation 26. Quantum oracle

where $+_2$ means modulo 2 addition, where the results are always Boolean. Odd results are converted to 1 and even results to 0. The first qubit is left unchanged and the second qubit is flipped if $f(x) = 1$. This way the value of the function is gotten out and the initial values can be determined from the output values. Extra bits added to the circuit this way, because of reversibility, are called ancilla bits and output bits that are not used later in the calculation, are called garbage bits. Classically, finding out what the function is, requires two runs of the oracle in order to get results $f(0)$ and $f(1)$. In the quantum case, an entangled state can be used as an input giving the result (Equation 27).

$$U\colon (|0\rangle + |1\rangle)|0\rangle \to |0\rangle|f(0)\rangle + |1\rangle|f(1)\rangle$$

Equation 27. Oracle result

With just one run of the process $U(f)$, information about both values of the function are obtained. The resulting state as an entangled state is not directly observable and serves only as an intermediate step during calculation. This phenomena of making simultaneous operations is called quantum parallelism. (Katzgraber & Renner 2008, 69.)

3.4 Models of quantum computation

There are different computational models that describe quantum computing. They have different basic operations that can be performed on the qubits.

- Quantum Turing machine. Generalization of the classical Turing machine to quantum calculation. It has qubits intead of memory tape and the state of the read and write head operates on the qubit states (Deutsch 1985, 97-117.) It is hard to describe algorithms using quantum Turing machine, because processes are described as operations between states of the read and write head and the qubits (Miszczak 2011, 5).

- Quantum circuits. The most popular model for expressing quantum algorithms. Circuits are made of qubits, wires, and quantum gates. Quantum circuit model is explained in chapter 3.5.

- 1-way quantum computer. Starts with a highly entangled cluster state. Computation is performed by only doing 1-qubit measurements. Because of the measurements, calculation is not reversible. (Katzgraber & Renner 2008, 163.)

- Adiabatic quantum computer. Starts by finding out a complicated qubit system, where the lowest energy state (ground state) contains the solution. Qubits are then cooled into the ground state of a simpler system. This simpler system is them slowly transformed into the more complicated system. Adiabatic quantum calculation is well suited for solving minimization and optimization problems. (Martikainen 2007. Lecture 10, 60-63.)

- Topological quantum computer. Uses global topological properties of a quantum system to make calculation more stable. Model uses two-dimensional quasiparticles known as anyons, which can form braids, which function as logic gates. Topological properties protect braids against small errors. (Colbert 2011, 8-9.)

- Quantum RAM (QRAM). Extension of classical RAM model. Quantum calculation is controlled by a classical computer. The quantum device holds the qubits and can change the number of qubits, measure them, and operate the basic unitary operations. The quantum device takes instructions from the classical computer and returns the results of the measurements. (Valiron 2012b, 3.)

- Quantum programming languages. Similarly to classical computing, in order to express complicated algorithms on large quantum computers, we will eventually need to use some kind of programming language. At minimum, it should have commands for allocating quantum registers, applying logical gate operations, measuring the registers and using subroutines. Several such languages have been proposed in the literature. (Valiron 2012b, 7-9.)

If programmable universal quantum computer is taken as the ultimate goal of quantum computing research, the models that are most suited to link hardware and programming lanquages are QRAM, quantum circuits and 1-way quantum computing. Quantum Turing machine and adiabatic quantum computer don't seem to be suitable. (Valiron 2012b, 2.)

3.5 Quantum circuits

Quantum circuit model is the most used model for expressing simple quantum algorithms and understanding the basic processes of quantum computing (Miszczak 2011, 8).

Time development of quantum mechanics is unitary. Simple unitary operations that operate on qubits are called quantum gates. It turns out that similarly to classical computing, where all circuits can be built from an universal gate set {AND, OR, NOT}, it is possible to find universal quantum gate sets. Possibilities of quantum computing can then be analyzed using chosen gate set. Unitarity of quantum computing means that all operations are reversible and this implies that quantum calculation produces no heat and therefore requires no energy. Only operations that are not unitary are measurements. Measurements are usually performed at the end of the calculation. (Pathak 2013, 75,137,151.)

Because quantum mechanical operations are unitary, the operation can be specified by giving the unitary matrix that operates on the Hilbert space vectors. Because unitary matrices have equal number of files and ranks, quantum gates always have equal number of input and output qubits. (Pathak 2013, 137.)

3.5.1 Quantum gates

In classical circuit model, the only possible one bit gates are the identity gate which does nothing, and NOT gate that flips the bit. In the quantum circuit model, there are infinite number of different one qubit gates. Because quantum operations are linear, it is enough to specify how the operation acts on the basis states. Attachment 1 shows the most important quantum gates and how they operate on state vectors. Pauli X gate flips the qubit. Pauli Z gate flips the relative phase of a superpositional state. Pauli Y is combination of qubit and phase flip. Hadamard gate transforms an eigenstate into superpositional state. Phase gate makes an arbitrary phase transformation. Phase sift and $\pi/8$ are especially important phase transformations. CNOT gate flips the second qubit if the first qubit is in the state $|1\rangle$. SWAP gate swaps the qubit states. Any 1-qubit gate U can be made into a 2-qubit gate called Controlled-U, where the first qubit if the control qubit, and the 1-gubit gate U acts on the second qubit if the first qubit is in the state $|1\rangle$. CNOT gate is in fact Controlled-NOT gate, as the name implies. Three qubit gates are usually controlled 2-qubit gates or controlled controlled 1-qubit gates. TOFFOLI gate is Controlled-Controlled-NOT and FREDKIN gate is Controlled-SWAP. (Pathak 2013, 137-147.)

Because classical NOT and TOFFOLI gates are reversible, they can easily be generalized into quantum gates. Irreversible gates like AND and OR have no quantum analogies. Therefore it is not possible to straightforwardly simulate classical circuits made out of {AND,OR,NOT} gate library using a quantum computer. However, any classical circuit can be constructed from reversible TOFFOLI gates, which has a quantum counterpart. (Pathak 2013, 46-47.) This is an easy way to show that any classical computation can be run on a quantum computer.

3.5.2 Circuit examples

As the basic quantum gates are known, it is possible to express quantum circuits and simple quantum algorithms using them. As an example, a random number generator and the entangled state used in the EPR experiment are generated using quantum gates.

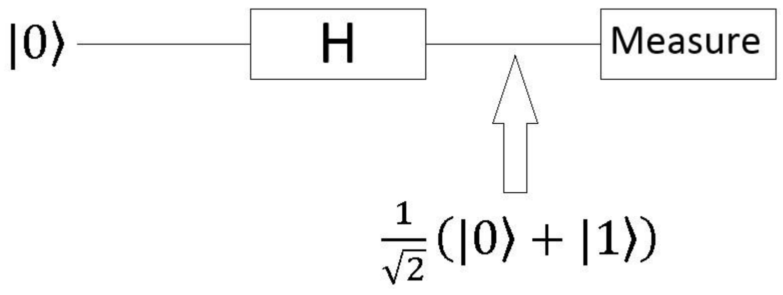


Figure 2. Random number generator

With a single Hadamard gate, a random number generator (Figure 2) can be generated. The measurement gives 0 and 1 with equal ½ probabilities. (Valiron 2012a, 5.)

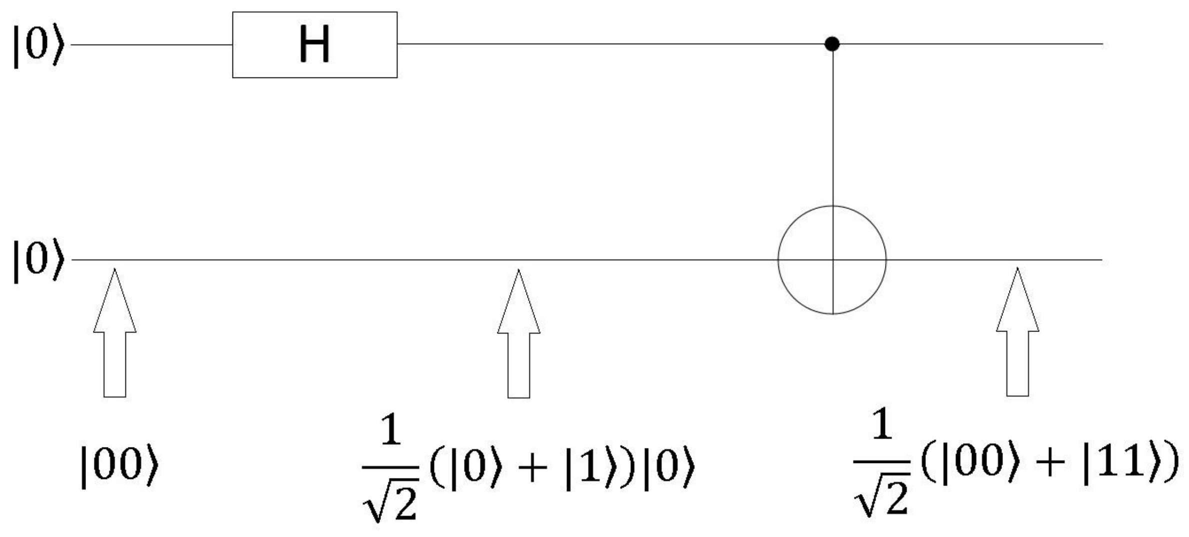

$|0\rangle$
H
$|0\rangle$
$|00\rangle$
$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle$
$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

Figure 3. Bell state generator

Combination of Hadamard gate and CNOT gates (Figure 3) can produce exactly the state used in the EPR experiment, which was reviewed in chapter 3.1.7 (Pathak 2013, 149-150).

3.5.3 Universal quantum gate sets

It can be shown, that any unitary quantum operation can be approximated to arbitrary accuracy by a set of universal quantum gates. One such set is {All 1-qubit gates, CNOT}. This set is not convenient, because the number of gates is infinite, and therefore error correction becomes problematic. However, any 1-qubit operation can be approximated by Hadamard and $\pi/8$ gates. Therefore $\{H, \pi/8, CNOT\}$ is universal. The phase sift gate is usually included in this universal gate set because it is used in error correction, making another universal gate set $\{H, S, \pi/8, CNOT\}$. (Martikainen 2008. Lectures 5-6.)

3.6 Decoherence and error correction

Information stored in qubits is unstable, because interaction with the environment affects qubits and causes errors, eventually destroying the quantum superposition. Depending on the qubit technology, the lifetimes of the qubits range from milliseconds to tens of seconds. (Valiron 2012a, 14-15.) Because of this unstability, error correction is an essential part of quantum computation.

There are two basic reasons why quantum error correction is problematic. Firstly, quantum states cannot be cloned. Measurement of a state destroys the state sending it to one of the eigenstates. Secondly, the continuous structure of a qubit (Equation 28)

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

Equation 28. Qubit

means that there can be an infinite number of different errors. Classically the only possible error is that the bit flips $0 \leftrightarrow 1$. Until 1995, when Peter Shor developed techniques for quantum error correction, it was believed that quantum error correction was impossible. (Pathak 2013, 207-211.)

Classically, the simplest error correction code is a repetition code. Every bit is cloned into an odd number of bits and from time to time bits are aligned by majority voting. In three bit repetition, this means that finding the bits to be 001 would conclude that the last bit had flipped and it should be corrected back to 0. (Pathak 2013, 209-210.)

Repetition code cannot be generalized to qubits, because quantum states cannot be cloned. However, it can be shown that a general 1-qubit error is a combination of a qubit flip (Equation 29)

$$\alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|1\rangle + \beta|0\rangle$$

Equation 29. Qubit flip

and a phase flip (Equation 30).

$$|0\rangle + |1\rangle \rightarrow |0\rangle - |1\rangle$$

Equation 30. Phase flip

In 1995 Peter Shor developed a 9-qubit code that can correct an arbitrary 1-qubit error. The circuit has the following structure (Figure 4).

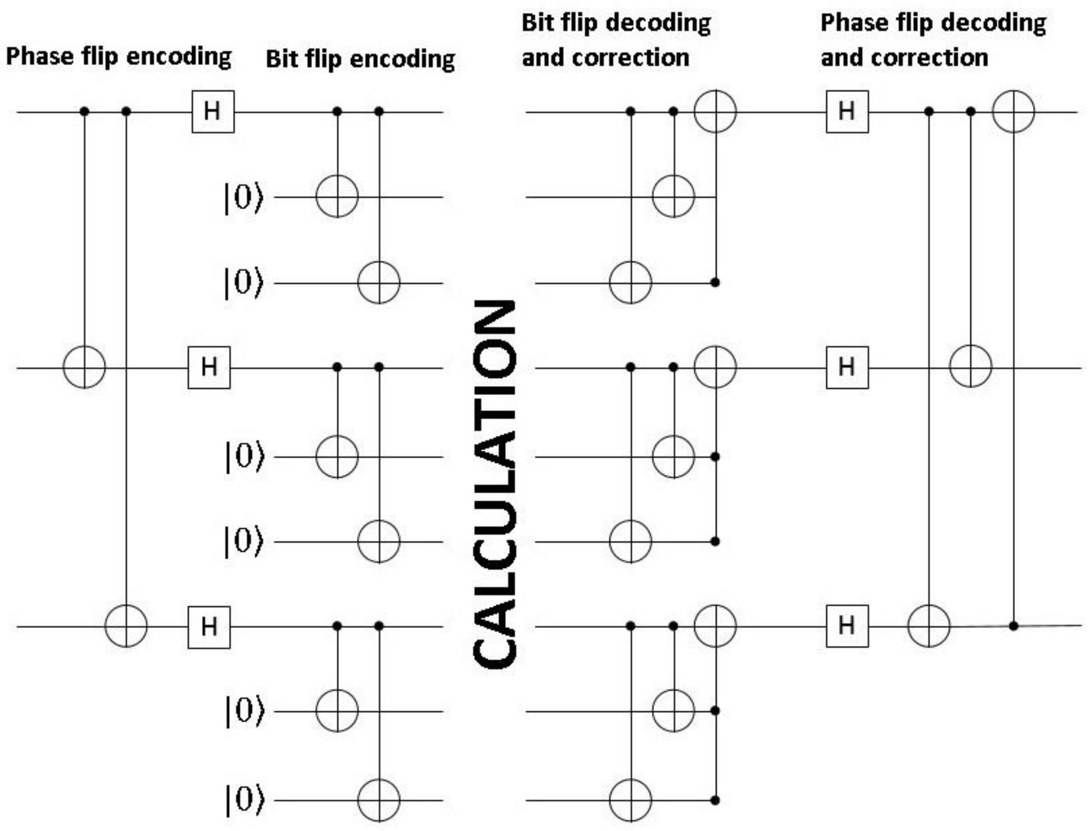

Phase flip encoding
Bit flip encoding
Bit flip decoding and correction
Phase flip decoding and correction
CALCULATION
H
|0⟩

Figure 4. Shor code circuit

Each qubit is encoded into nine qubit system and after some operations decoded back. (Shor 1995.)

The Shor code is not optimal, because the minimum number of qubits needed for correcting an arbitrary 1-qubit error is five. Such a code was published in 1996. (Laflamme, Miquel, Paz & Zurek 1996, 198-201.)

Just using error correcting codes with quantum gates is problematic because encoded information must be decoded before each gate operation. This leads to massive amount of extra operations. Error correcting codes are themselves made out of quantum gates which also produces errors. Better approach is to integrate error correction into the quantum gates producing fault tolerant quantum gates. Study of fault tolerant quantum circuits is one of the main areas of quantum computing research. (Pathak 2013, 221-223.)

4 Quantum algorithms

Knowledge of universal quantum gate sets means that the basic building blocks of quantum computing are known. This allows the analysis of possible quantum algorithms. The goal is to find quantum algorithms for important problems giving significant speedup over classical algorithms.

4.1 Simple quantum algorithm

In 1985, the first quantum algorithm was developed by David Deutsch. It proved that there are some problems were quantum computers outperform classical computers. It is also a good example of the basic structure of quantum algorithms. The problem considers a boolean function (Equation 31).

$$f: \{0,1\} \rightarrow \{0,1\}$$

Equation 31. Boolean function

There are four such functions, because each input value has two possible input values. Quantum mechanically, an unitary process called an oracle, must be constructed (Equation 32).

$$U_f:$$
$$|x\rangle \rightarrow |x\rangle$$
$$|y\rangle \rightarrow |y +_2 f(x)\rangle$$

Equation 32. Quantum oracle

This oracle can perform classical calculation, if we use only eigenstates as input values, but can also take superpositional states as an input and perform quantum calculation. Classically the function is found out using input values $|00\rangle$ and $|10\rangle$, and measuring the function values from the second qubit (Equation 33).

$$|0\rangle \rightarrow |0\rangle$$
$$|0\rangle \rightarrow |0 +_2 f(0)\rangle = |f(0)\rangle$$
$$|1\rangle \rightarrow |1\rangle$$
$$|0\rangle \rightarrow |0 +_2 f(1)\rangle = |f(1)\rangle$$

Equation 33. Measuring the function classically

Quantum mechanically, the problem of finding out the function is equally hard, because using a superpositional state as an input produces (Equation 34),

$$(|0\rangle + |1\rangle)|0\rangle \rightarrow |0\rangle|f(0)\rangle + |1\rangle|f(1)\rangle$$

Equation 34. Output of a superpositional state

which doesn’t help, because the resulting state is entangled superpositional state. Function values cannot be red even with two measurements. However, If another problem of finding out whether the function f is constant or balanced, is considered, it turns out that quantum mechanical properties can be taken advantage of. Function f is constant if the function values are the same (Equation 35),

$$f(0) = f(1) = 0 \; or \; f(0) = f(1) = 1$$

Equation 35. Constant boolean function

and balanced, if the function values are different (Equation 36).

$$f(0) = 0, f(1) = 1 \; or \; f(0) = 1, f(1) = 0$$

Equation 36. Balanced boolean function

Classically, the function must again be run twice to get the function values, which can be used to determine whether the function is constant or balanced. Quantum mechanically, only one run of the process is needed. The following circuit (Figure 5) shows how it is done.

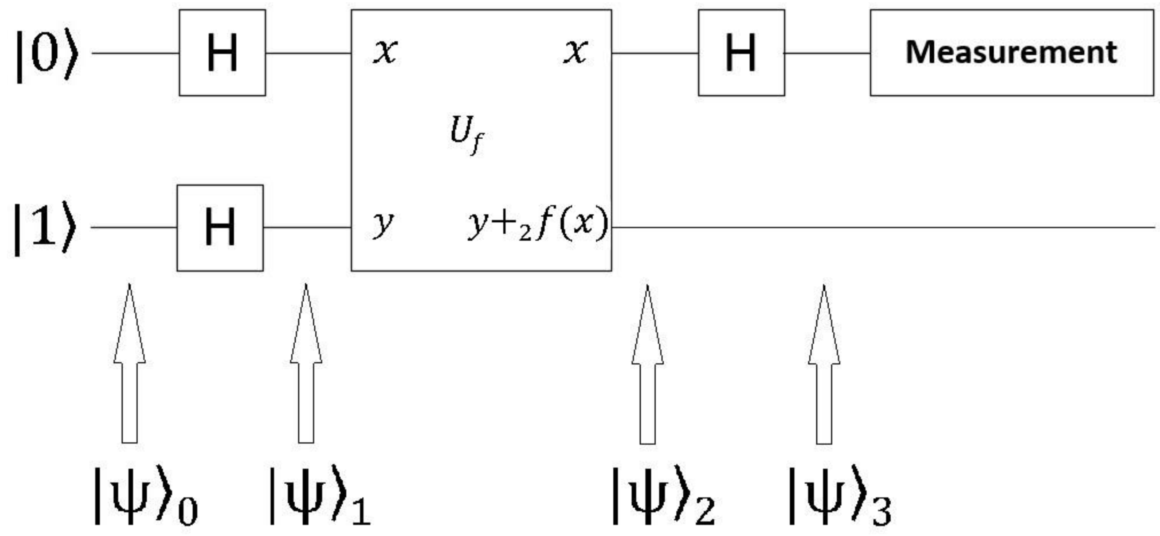


Figure 5. Deutsch algorithm circuit

The calculation starts with qubits in eigenstates (Equation 37).

$$|\psi\rangle_0 = |01\rangle$$

Equation 37. Initial state

The states are then mixed up into entangled superpositional states with Hadamard gates (Equation 38).

$$|\psi\rangle_1 = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

Equation 38.

After the oracle operation the state is (Equation 39).

$$|\psi\rangle_2 = \begin{cases} \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = f(1) = 0 \\ -\left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = f(1) = 1 \\ \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = 0, f(1) = 1 \\ -\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = 1, f(1) = 0. \end{cases}$$

Equation 39.

This can be rewritten as (Equation 40).

$$|\psi\rangle_2 = \begin{cases} \pm\left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = \ f(1) \ \rightarrow constant\ f \\ \pm\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) \neq \ f(1) = 0 \ \rightarrow balanced\ f. \end{cases}$$

Equation 40.

Applying Hadamard gate to the first qubit yields (Equation 41).

$$|\psi\rangle_3 = \begin{cases} \pm|0\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) = \ f(1) \ \rightarrow constant\ f \\ \pm|1\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], if\ f(0) \neq \ f(1) = 0 \ \rightarrow balanced\ f \end{cases}$$

Equation 41. Deutch algorithm result

Now the first qubit is always in an eigenstate and with one measurement on the first qubit it can be found out whether the function is constant or balanced. (Pathak 2013, 174-176.) The Deutsch algorithm was originally presented in (Deutsch 1985, 97-117) in a more complicated probabilistic form and later converted to this simpler form presented here.

The Deutch algorithm is not very useful but rather shows that some things can be calculated faster using a quantum calculation. It also shows the basic structure of quantum computation. The input qubits are initialized into eigenstates and then mixed up into entangled superpositions. Quantum operations are then performed and finally the qubits that contain the result are transformed back into eigenstates. Only this way the results can be obtained with single measurements from the output qubits. One could say that quantum calculation in the circuit model starts and ends in the classical world, and visits the quantum world during the calculation.

The Deutch algorithm can be generalized from the one qubit case to a n-qubit case. There we have a function (Equation 42),

$$f: \{0,1\}^n \rightarrow \{0,1\}$$

Equation 42.

which has n bits input and one bit output. The promise is that the function is either constant with all function values either 0 or 1, or balanced with exactly half of the values 0 and half 1. There are 2^n different inputs and classically the function must be calculated for $2^{n-1} + 1$ input values to find out the answer. Quantum mechanically, only one run of the oracle is needed. The procedure is similar to the Deutsch algorithm discussed earlier. This generalized case shows an exponential speedup over the classical calculation. (Pathak 2013, 176-178.) This generalized Deutsch-Jozsa algorithm was first published in (Deutsch & Jozsa 1992, 553).

4.2 Central quantum algorithms

Although moving from bits to qubits leads to more basic logical operations and therefore to new possibilities in creating algorithms, the number of known algorithms that outperform classical algorithms is very small. There are many possible reasons for this. Quantum calculation involves entangled superpositional states and it is hard to produce the result into qubits in eigenstates. The logic and structure of a quantum computer is hard to imagine and very fast becomes impossible to simulate with a classical computer. Therefore designing new quantum algorithms is not easy. It is also possible, that the number of quantum algorithms that outperform classical algorithms is in fact very small. (Shor 2003.)

A list of known quantum algorithms can be found at http://math.nist.gov/quantum/zoo/. On 3 Apr 2014 it lists 214 quantum algorithms that offer speedup over classical computation. Most of the algorithms are however applications of two powerful quantum algorithms, namely Grover's search algorithm and quantum Fourier transform. There are also some applications involving simulation of quantum mechanical systems on quantum computers and quantum walk.

4.2.1 Simulation of quantum systems

In 1982, it was proven by Richard Feynman, that simulating quantum mechanical systems is a hard problem for classical computers. He also had an idea that it could be possible to build a quantum computer that could efficiently simulate quantum mechanical systems. (Feynman 1982.) Since then, many applications of this idea have been developed. Amongst them, an algorithm for simulating systems of elementary particles (Abrams & Lloyd 1997), and an algorithm for simulating chemical reactions in (Kassal, Jordan, Love, Mohseni & Aspuru-Guzik 2008).

4.2.2 Quantum walk

Many classical algorithms are formed using random walk. Quantum mechanical generalization is called quantum walk. The walker is in superposition of positions and can offer speedup when going through graph- or tree-like information structures. (Farhi & Gutmann 1998.)

4.2.3 Amplitude amplification and quantum search

Grover's search algorithm is a quantum algorithm for searching in unsorted data. Classically, finding the wanted entry in a database of N entries, requires N queries in the worst case and $N/2$ queries in average. Therefore the complexity of the problem is classically $O(N)$. If the data is stored in quantum bits, it takes only $\frac{\pi}{4}\sqrt{N} = O(\sqrt{N})$

queries to find the solution. The algorithm uses a method called amplitude amplification, which uses certain quantum operations to iterate the solution from a superposition of all entries. (Grover 1996.) Grover's search algorithm is important, because searching is a basic process of computing. Many mathematical and computational algorithms involve searching and Grover's search algorithm can offer a quadratic speedup moving from classical to corresponding quantum algorithms. Although the speedup is not exponential, even a quadratic speedup can be tremendous when dealing with large data sizes.

4.2.4 Quantum Fourier transform

Fourier transform is a mathematical process that is used in mathematics and all over natural sciences. An example of a Fourier transform is dividing a sound wave into component waves of different wavelengths. Moving from the values of the function to the coefficients of different component waves, is the Fourier transform.

Fourier transform is a hard process for classical computers and thus many problems involving Fourier transform are not efficiently solvable. Quantum computers however solve Fourier transforms easily. (Girvin 2013, 7-8.)

For real functions, a Fourier transform means expressing the function as an integral or a sum over sine and cosine functions of different wavelengths. For complex valued functions, the sum runs over different exponential functions. If the data set is not continuous but discrete, the transformation is called a discrete Fourier transform.

For a discrete data set (Equation 43)

$$\alpha(j) = \{\alpha_0, \alpha_1, \dots, \alpha_{N-1}\}, \qquad j = 0,1,\dots,N-1$$

Equation 43. Discrete data set

of N complex numbers, the data set can be expressed as a sum (Equation 44).

$$\alpha(j) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \beta(k) e^{-2\pi ijk/N}, \qquad j = 0,1,\dots,N-1$$

Equation 44. Fourier expansion

Moving from the function values α to the coefficients of the exponential functions β is the Fourier transform. The function β can be calculated by (Equation 45).

$$\beta(j) = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \alpha(j) e^{2\pi ijk/N}, \qquad k = 0,1,\dots,N-1$$

Equation 45. Fourier coefficients

Quantum Fourier transform operates on a quantum state (Equation 46).

$$|\psi\rangle = \sum_{j=0}^{N-1} \alpha(\mathrm{j})|j\rangle$$

Equation 46. Quantum state

Quantum Fourier transform is just a regular discrete Fourier transform on the discrete data set $\alpha(\mathrm{j})$. For n-qubit system, the dimensionality is $N = 2^n$. The quantum fourier transform can be expressed as a NxN matrix (Equation 47).

$$F = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & & \omega^{2(N-1)} \\ & \vdots & & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix}, \qquad where\ \omega = e^{2\pi i/N}$$

Equation 47. Quantum Fourier transform matrix

For a single qubit system (Equation 48),

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

Equation 48. Qubit

the Fourier transform is (Equation 49),

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

Equation 49. One-qubit Fourier transform

which is just the Hadamard gate. This shows that, at least for a single qubit, quantum gates quite naturally solve Fourier transforms. The quantum circuit for the 1-qubit Fourier transform is then simply (Figure 6).

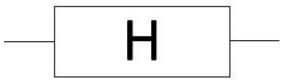


Figure 6. One-qubit Fourier transform circuit

For three qubits, the circuit is more complicated (Figure 7).

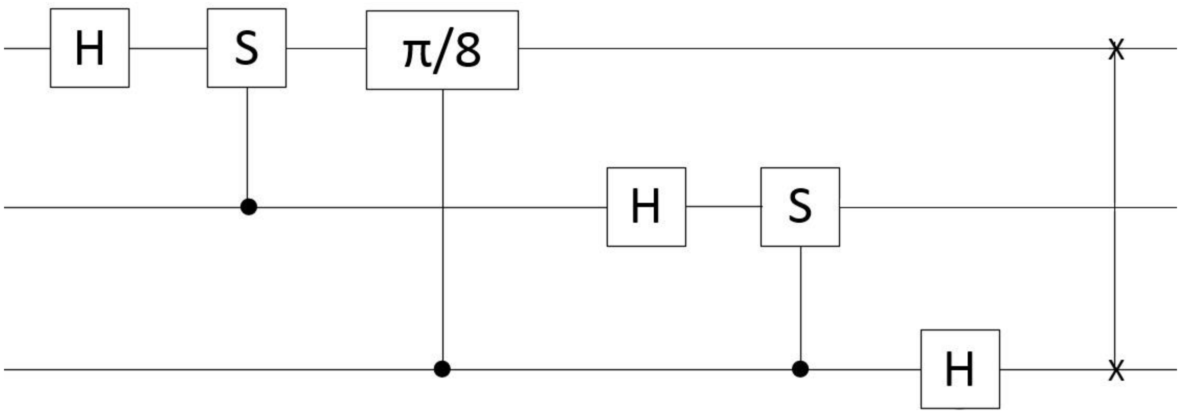


Figure 7. Three-qubit Fourier transform circuit

For n qubits, the Fourier circuit is also constructed from Hadamard and rotation gates (Pathak 2013, 191-194.) The number of quantum operations is $O(n^2)$. Classically the best known algorithm takes $O(n2^n)$ steps, so the quantum speedup for performing a Fourier transform is exponential. (Girvin 2013, 7.)

Because the output of a quantum Fourier transform is given as coefficients of a super-positional quantum state, it cannot be used to actually calculate a Fourier transform

(Martikainen, Lecture 7, 3). It can however be used to gain speedup on many important mathematical problems. All known exponential quantum speedups are found on a group of problems called hidden subgroup problem. Examples of such problems gaining an exponential speedup are order finding, discrete logarithm, and factoring. (Martikainen 2007, Lecture 8, 24.)

An efficient quantum algorithm for factoring was developed in 1994 by Peter Shor. It is of special interest at the moment, because the assumption that factoring is a hard problem is an integral part of RSA cryptographic protocol used in most of modern secure communication protocols. If a large universal quantum computer is one day built, or at least a quantum computer capable of running Shor's algorithm, it would make modern RSA encryption unsecure and out of date. (Pathak 2013, 22.)

5 History of theories behind quantum computing

The progress of quantum computing has been mainly theoretical progress over the last 35 years. There has been many developments on the fields of quantum information and quantum communication, but here the most important theoretical developments relating directly to quantum computing are listed:

- In 1980-82 Paul Benioff developed the idea that a quantum device could simulate classical computation with no energy expense (Benioff 1982).
- In 1982, Richard Feynman proved that simulating quantum systems is hard for classical computers. He also proposed that perhaps quantum systems could be easily simulated using a computer that took advantage of quantum mechanics. (Feynman 1982.)
- In 1985, David Deutsch formulated an universal quantum computer and a quantum Turing machine. He also proved that a quantum computer could solve some problems faster than classical computers by introducing the Deutsch algorithm. (Deutsch 1985.)
- In 1994, the first quantum algorithms for computing something meaningful were developed. Peter Shor developed algorithms for factoring and discrete logarithm problems. (Shor 1994.)
- In 1995, Peter Shor published an error correcting algorithm for correcting an arbitrary 1-qubit error (Shor 1995).
- In 1996, Peter Shor developed the concept of fault tolerant quantum computing (Shor 1996).
- In 1996, Lov Grover designed an algorithm for searching an unordered database (Grover 1996).
- In 2000, David DiVincenzo introduced the criteria for a scalable quantum computer (DiVincenzo 2000).

6 Physical realizations and results

Roughly 20 years after quantum computers were theoretically modelled, actual physical realizations started to emerge at the start of the current millennium. There are still only a couple of scientific experiments where quantum computing has been performed. This chapter reviews the criteria for successfully building a quantum computer and then reviews the history of actual constructions. Recent findings and ongoing projects are also reviewed.

6.1 Requirements for a quantum computer

In order to perform quantum computation, a physical system must fulfill certain criteria. Qubits must be scalable, well characterized, and with low enough error rate. Decoherence times (lifetimes) of the qubits must be much lower than gate operation times. There must also be a way to initialize and measure the qubits. In order to perform any algorithm, there must be operations that correspond to some universal set of quantum gates. These criteria can be used to test the adequacy of a realizations of a quantum computer. (DiVincenzo 2000.)

Using the terminology of this thesis, these criteria correspond to building an universal quantum computer. For building a quantum computer that runs some specific algorithm, one can do with limited scalability and no universal set of operations.

6.2 Qubit realizations

A large number of different physical realizations of qubits are being studied. Some examples are reviewed here concentrating on ones that have provided proven results. Every realization has its pros and cons. Most of the systems can be put into following categories:

- Photons are fine candidates for qubits because they interact weakly with the environment and are therefore naturally quite free from decoherence. The main problem is how to make photons interact. There are several ways to make qubits from photons. Photon polarization states and position of the photon are two natural possibilities.

- Ions, which are electrically charged atoms, can be trapped in free space by electric fields. Ions are arranged into an array and lowest energy state is obtained by laser cooling. Interaction with one ion is done with laser and interaction between ions by local interaction or photonic interaction. At small scale the model seems to be working but the problem is how to scale up the system.
- Atomic nuclei are naturally isolated from the environment behind the electron cloud. Nuclear spin is therefore very stable. It can be operated on either by an electric field or by nuclear magnetic resonance (NMR). Problems with nuclei are related to initialization of the system and scaling up the system.
- Quantum particles can be stored into a solid host such as a semiconductor, a piece of silicon, a crystal like for example a diamond. The fact that the quantum particle is trapped into a solid host makes the qubit realization more stable. There is also possibility to manage without cooling the system to very cold temperatures.
- Electron spins are natural candidates for qubit realizations, but in typical electric hardware, decoherence due to resistivity is extremely high. Superconductors however minimize resistivity, and quantum computing has be experimented on superconducting systems. (Valiron 2012a, 16-19.)

6.3 Constructions and results

This chapter reviews experiments were an algorithm has been run on a quantum computer. The list is not complete, but tries to show how the area has evolved.

- In 1998, the first 2-qubit quantum computer was built. It used NRM technique on chloroform nuclei. The computer was able to run Grover's search algorithm on 4 states. (Chuang, Gershenfeld & Kubinec 1998.)
- The NRM technique was later used to produce several results. Grover's search algorithm was run on a 3-qubit quantum computer (Vandersypen et al. 2000a). Order finding algorithm was run on a 5-qubit quantum computer (Vandersypen et al. 2000b). Shor's factoring algorithm was run on a 7-qubit quantum computer, successfully factoring $15 = 3x5$ (Vandersypen et al. 2001). A 4-qubit quantum computer factored $143 = 11x13$ (Nanyang et al. 2011).

- The first 2-qubit solid state quantum processor was created, that was able to run basic algorithms. Superconducting chip, cooled near absolute zero temperature, used two qubits, each made up out of a billion aluminium atoms. This was the first quantum computer that resembled typical electronic device. Earlier, all quantum computing had been performed using atoms, nuclei, or photons. (Yale news 2009.)
- A silicon based quantum computing chip using quantum optics was able to run Shor's algorithm (NewScientist 2009).

Several interesting techniques have also been developed. In 2012 Australian engineers created qubits from single silicon atoms. This is interesting because qubits were generated using the same material used in typical classical electronics. (Griffith 2012.) In the same year, a 2-qubit quantum computer was created on a crystal of a diamond. It worked on room temperature and might be scaled up in size. (Perkins 2012.)

The first commercial quantum computer D-Wave One, was put to sale in 2011 by D-Wave Systems (D-Wave 2011a). The company claims that the chip, costing $10 million, is a 128-qubit adiabatic quantum computer that can solve optimization problems (Anthony 2011). On May 25 2011, the first D-Wave One system was sold to Lockheed Martin (D-Wave 2011b). On May 16 2013, D-wave announced that their new 512-qubit D-Wave Two quantum computer was going to be installed at the new Quantum Artificial Intelligence Lab founded by NASA, Google and the Universities Space Research Association (USRA) (D-Wave 2013). Because D-Wave chips are commercial devices, it becomes very hard to verify that the chip is actually doing quantum computing. Scientists do not know the exact structure of the chip and are not allowed to look inside the chip. Determining the quantumness of a device from the input and the output is an ongoing research problem of its own. (Zagoskin, Ilichev, Grajcar, Betouras & Nori 2014.) At the moment, it is not clear whether the D-Wave chips are classical computers, quantum devices doing classical computing, or real quantum computers (Shin, Smith, Smolin & Vazirani 2014). Adiabatic quantum computing limits the use of D-Wave chips to some optimization problems. As stated in chapter 3.4, adiabatic quantum computing is not likely platform for building an universal quantum computer, or a programmable universal quantum computer.

7 Summary and conclusions

This thesis has shown that theoretically it is quite well known what is required in order to build a large quantum computer. The basic logical processes of quantum computation are known and some powerful quantum algorithms have been developed. These algorithms offer quadratic speedup on processes involving searching and turn some important unsolvable mathematical problems into solvable ones. One such a problem is factoring of large integer numbers into a product of prime numbers, which allows quantum computers to break all the modern public key encryption algorithms used in network communication. Another advantage of quantum computation is that it produces no heat and therefore requires no energy source. Only measurement of the qubits at the end of the computation and error correction produce heat, which makes quantum computation much more energy efficient than classical computation.

Only very small quantum computers, made out of a couple of qubits, have been built. A large quantum computer would stretch quantum phenomena into macroscopic distances, and it is not known whether or not it is possible to build one. Qubits are also very unstable because interaction with the environment disturbs the states of the qubits. The qubits must therefore be isolated but somehow made to interact when wanted. The continuous structure of qubits also makes error correction hard.

Personal learning experience during the thesis process was quite straightforward. I started to think about the thesis right after my studies had started at Haaga-Helia in the fall of 2013. In the spring of 2014 I had my subject chosen, but didn't know anything about it. My goal was to write the thesis during the first school year, so I could concentrate of finishing the rest of the courses during the second year. The spring courses went well and I had enough time to learn the basics of the thesis subject. I devoted the April of 2014 to writing the thesis. Writing the thesis was very instructive, because I hadn't written any long texts in many years. It was also a delightful practice on my English language skills.

References

Abrams, D. & Lloyd, S. 1997. Simulation of many-body Fermi systems on a universal quantum computer. Physical review letters, vol 79 (13), 2586-2589.

Anthony, S. 2011. First Ever Commercial Quantum Computer Now Available for $10 Million. ExtremeTech. URL: http://www.extremetech.com/computing/84228-first-ever-commercial-quantum-computer-now-available-for-10-million. Accessed: 15 Apr 2014.

Aspect, A., Dalibard, J. & Roger, G. 1982. Experimental test of Bell's inequalities using time-varying analyzers. Physical review letters, vol 49, 1804-1807.

Bell, J. 1964. On the Einstein Podolsky Rosen paradox. Physics, vol 1, 195-200.

Benioff, P. 1982. Quantum mechanical Hamiltonian models of Turing machines that dissipate no energy. Physical review letters, vol 48, 1581-1585.

Chuang, I., Gershenfeld, N. & Kubinec, M. 1998. Experimental implementation of fast quantum searching. Physical review letters, vol 80, 3408-3411.

Church, A. 1936. An unsolvable problem of elementary number theory. American Journal of Mathematics 58.

Colbert, J. 2011. Topological quantum computing. URL: http://hep.uchicago.edu/~rosner/p342/projs/colbert.pdf. Accessed 27 Mar 2014.

Deutsch, D. 1985. Quantum theory, the Church-Turing principle and the universal quantum computer. Proceedings of the Royal Society of London A, vol 400, 97-117.

Deutsch, D. & Jozsa, R. Rapid solutions of problems by quantum computation. Proceedings of the Royal Society of London A, vol 439.

DiVincenzo, D. 2000. The physical implementation of quantum computation. URL: http://arxiv.org/abs/quant-ph/0002077v3. Accessed: 16 Apr 2014.

Drechsler, R. & Wille, R. 2012. Reversible circuits: Recent accomplishments and future challenges for an emerging technology. URL: http:// informatik.uni-bremen.de /agra/doc/konf/2012_vdat_reversible_circuits_accompl_chall.pdf. Accessed: 5 Jan 2014.

D-Wave. 2011a. Learning to program the D-Wave One. URL: http://dwave.word-press.com/2011/05/11/learning-to-program-the-d-wave-one. Accessed: 15 Apr 2014.

D-Wave. 2011b. D-Wave Systems sells its first Quantum Computing System to Lock-heed Martin Corporation. URL: http://www.dwavesys.com/news/d-wave-systems-sells-its-first-quantum-computing-system-lockheed-martin-corporation. Accessed: 15 Apr 2014.

D-Wave. 2013. D-Wave TwoTM Quantum Computer Selected for New Quantum Ar-tificial Intelligence Initiative, System to be Installed at NASA's Ames Research Center, and Operational in Q3. URL: http://www.dwavesys.com/updates/d-wave-twotm-quantum-computer-selected-new-quantum-artificial-intelligence-initiative-system. Ac-cessed: 15 Apr 2014.

Einstein, A., Podolsky, B. & Rosen, N. 1935. Can quantum-mechanical description of physical reality be considered complete? Physical review, vol 47 (10), 777-780.

Feynman, R. 1982. Simulating physics with computers. International journal of theoret-ical physics, vol 21, 467-488.

Girvin, S. 2013. Basic concepts in quantum information. URL: http://arxiv.org/abs/1302.5842v1. Accessed 5 Jan 2014.

Griffith, C. 2012. Australian engineers write quantum computer 'qubit' in global breakthrough. The Australian. URL: http://www.theaustralian.com.au/technology/australian-engineers-write-quantum-computer-qubit-in-global-breakthrough/story-fn4htb9o-1226477592578. Accessed: 15 Apr 2014.

Grover, L. 1996. A fast quantum mechanical algorithm for database search. URL: http://arxiv.org/abs/quant-ph/9605043. Accessed: 4 Apr 2014.

Kassal, I., Jordan, S., Love, P., Mohseni, M. & Aspuru-Guzik, A. 2008. Polynomial-time quantum algorithm for the simulation of chemical dynamics. URL: http://arxiv.org/abs/0801.2986. Accessed 3 Apr 2014.

Katzgraber, H. & Renner, R. 2008. Quantum computing. URL: http://katzgraber.org/teaching/fs08/files/proseminar.pdf. Accessed 17 Dec 2013.

Laflamme, R., Miquel, C., Paz, J. & Zurek, W. Perfect quantum error correcting code. Physical review letters, vol 77.

Landauer, R. 1961. Irreversibility and heat generation in the computing process. IBM journal of research and development, vol. 5.

Martikainen, J. 2007. Quantum information and computing. URL: http://theory.physics.helsinki.fi/~kvanttilaskenta. Accessed: 7 Dec 2013.

Miszczak, J. 2011. Models of quantum computation and quantum programming languages. URL: http://arxiv.org/abs/1012.6035v2. Accessed: 5 Feb 2014.

Nanyang, X., Jing, Z., Dawei, L., Xianyi, Z., Xinhua, P. & Jiangfeng, D. 2011. Quantum Factorization of 143 on a Dipolar-Coupling NMR system. URL: http://arxiv.org/abs/1111.3726. Accessed: 15 Apr 2014.

NewScientist. 2009. Code-breaking quantum algorithm run on a silicon chip. URL: http://www.newscientist.com/article/dn17736-codebreaking-quantum-algorithm-run-on-a-silicon-chip.html#.U00EdIX4J8E. Accessed: 15 Apr 2014.

Pathak, A. 2013. Elements of quantum computation and quantum communication. Taylor & Francis. Boca Raton.

Perkins, R. 2012. Quantum computer built inside diamond. Futurity. URL: http://www.futurity.org/quantum-computer-built-inside-diamond. Accessed: 15 Apr 2014.

Sakurai, J. J. & Napolitano, J. 2011. Modern quantum mechanics. Addison-Wesley. San Francisco.

Shin, S., Smith, G., Smolin, J. & Vazirani, U. 2014. How "Quantum" is the D-Wave Machine? URL: http://arxiv.org/abs/1401.7087. Accessed: 15 Apr 2014.

Shor, P. 1994. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. URL: http://arxiv.org/abs/quant-ph/9508027. Accessed 16 Apr 2014.

Shor, P. 1995. Scheme for reducing decoherence in quantum computer memory. Physical review A, vol 52, 2493-2496.

Shor, P. 1996. Fault-tolerant quantum computation. URL: http://arxiv.org/abs/quant-ph/9605011. Accessed 16 Apr 2014.

Shor, P. 2003. Why haven't more quantum algorithms been found? Journal of the ACM, vol 50, 87-90.

Turing, A. 1937. On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, ser. 2, vol 42. Corrections, Ibid, vol 43, 544-546.

Valiron, B. 2012a. Quantum computation: a tutorial. URL: http://www.monoidal.net/papers/tutorialqpl-1.pdf. Accessed: 7 Jan 2014.

Valiron, B. 2012b. Quantum computation: from a Programmer's perspective. URL: http://www.monoidal.net/papers/tutorialqpl-2.pdf. Accessed: 7 Jan 2014.

Vandersypen, L., Steffen, M., Sherwood, M., Yannoni, C., Breyta, G. & Chuang, I. 2000a. Implementation of a three-quantum-bit search algorithm. Applied physics letters, vol 76, 646-648.

Vandersypen, L., Steffen, M., Breyta, G., Yannoni, C., Cleve, R. & Chuang, I. 2000b. Experimental realization of an order-finding algorithm with an NMR quantum computer. Physical review letters, vol 85, 5452-5455.

Vandersypen, L., Steffen, M., Breyta, G., Yannoni, C., Sherwood, M. & Chuang, I. 2001. Experimental realization of Shor's quantum factoring algorithm using magnetic resonance. URL: http://arxiv.org/abs/quant-ph/0112176. Accessed: 15 Apr 2014.

Yale news. 2009. Scientists Create First Electronic Quantum Processor. URL: http://news.yale.edu/2009/06/28/scientists-create-first-electronic-quantum-processor. Accessed: 15 Apr 2014.

Zagoskin, A., Ilichev, E., Grajcar, M., Betouras, J. & Nori, F. 2014. How to test the "quantumness" of a quantum computer? URL: http://arxiv.org/abs/1401.2870. Accessed: 15 Apr 2014.

Käsiteet

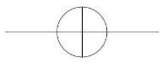
Complex linear space	Avaruus, jossa alkioita voidaan kertoa kompleksiluvuilla, laskea yhteen ja ottaa alkioiden pistetuloja.
Easy problem	Ongelma, jonka ratkaisuun kuluva aika riippuu lähtödatan määrästä lineaarisesti.
Eigenstate	Ominaistila, jossa mitattava suure saa aina tietyn arvon.
Eigenvalue	Mitattavan suureen mahdollinen arvo.
Entanglement	Usean hiukkasen tila, jossa hiukkasen mittaus vaikuttaa toisen hiukkasen mittaukseen.
Exponential	Lauseke jossa muuttuja on eksponentissa.
Hard problem	Ongelma, jonka ratkaisuun kuluva aika riippuu lähtödatan määrästä eksponentiaalisesti.
Hilbert space	Kvanttimekaanisten tilojen muodostama kompleksinen lineaariavaruus.
Heat	Elektroniset komponentit tuottavat lämpöä ja energiaa kuluu.
Linear	Lauseke jossa muuttuja ei esiinny eksponentissa.
Normalized	Todennäköisyyksien laskemiseksi tilat täytyy normalisoida eli kertoa sopivalla vakiolla.
Observable	Mitattava suure.
Oracle	Klassista bittioperaatiota vastaava kvanttioperaatio.
Orthogonal	Suorakulmainen.
Prepare	Systeemin saattaminen haluttuun alkutilaan.
Qubit	Kvanttibitti. Yksinkertaisin kvanttimekaaninen systeemi.
Reversible	Käännettävissä oleva. Muutos, joka voidaan perua.
State	Systeemin tila. Sisältää kaiken tiedon systeemistä. Kvanttimekaniikassa tilat muodostavat Hilbertin avaruuden.
Superposition	Systeemin mielivaltainen tila on ominaistilojen lineaarikombinaatio.
Unitary	Kvanttimekaaniset operaatiot ovat unitaarisia eli kiertoja Hilbertin avaruudessa.

Attachments

Attachment 1. Quantum gates

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

or



$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}$$

$Pauli\ Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ i \end{bmatrix} = i|1\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -i \\ 0 \end{bmatrix} = -i|0\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} -i\beta \\ i\alpha \end{bmatrix} = i \begin{bmatrix} -\beta \\ \alpha \end{bmatrix}$$

$Pauli\ Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} = -|1\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ -\beta \end{bmatrix}$$

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

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} \alpha + \beta \\ \alpha - \beta \end{bmatrix}$$

$Phase = P(\emptyset) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\emptyset} \end{bmatrix}$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & e^{i\emptyset} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & e^{i\emptyset} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ e^{i\emptyset} \end{bmatrix} = e^{i\emptyset}|1\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & e^{i\emptyset} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ e^{i\emptyset}\beta \end{bmatrix}$$

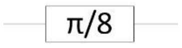
$Phase\ sift = P\left(\frac{\pi}{2}\right) = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ i \end{bmatrix} = i|1\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ i\beta \end{bmatrix}$$

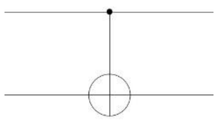
$\pi/8 = P\left(\frac{\pi}{4}\right) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$



$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ e^{i\pi/4} \end{bmatrix} = e^{i\pi/4}|1\rangle$$

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ i\beta \end{bmatrix}$$



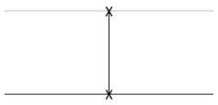
$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = |00\rangle$$

$|01\rangle \rightarrow |01\rangle$

$|10\rangle \rightarrow |11\rangle$

$|11\rangle \rightarrow |10\rangle$



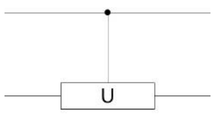
$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = |00\rangle$$

$|01\rangle \rightarrow |10\rangle$

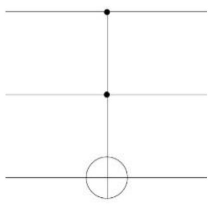
$|10\rangle \rightarrow |01\rangle$

$|11\rangle \rightarrow |11\rangle$



$Controlled\ U = \begin{bmatrix} I_2 & 0_2 \\ 0_2 & U \end{bmatrix}, where\ I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\ and\ 0_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$$TOFFOLI = CCNOT = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$



$|000\rangle \rightarrow |000\rangle$

$|001\rangle \rightarrow |001\rangle$

$|010\rangle \rightarrow |010\rangle$

$|011\rangle \rightarrow |011\rangle$

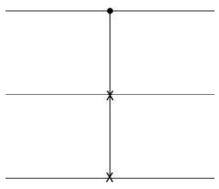
$|100\rangle \rightarrow |100\rangle$

$|101\rangle \rightarrow |101\rangle$

$|110\rangle \rightarrow |111\rangle$

$|111\rangle \rightarrow |110\rangle$

$$FREDKIN = CSWAP = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



$|000\rangle \rightarrow |000\rangle$

$|001\rangle \rightarrow |001\rangle$

$|010\rangle \rightarrow |010\rangle$

$|011\rangle \rightarrow |011\rangle$

$|100\rangle \rightarrow |100\rangle$

$|101\rangle \rightarrow |110\rangle$

$|110\rangle \rightarrow |101\rangle$

$|111\rangle \rightarrow |111\rangle$