Quantum Computing

10112 — Advanced Quantum Mechanics

Authors:
Ask H. Larsen, s021864
Martin F. Laursen, s021767
Kasper Reck, s021817

October 12, 2006

Technical University of Denmark
Lyngby

Contents

1 Introduction

Classical computing is evolving ever closer to a number of fundamental limits that will eventually halt further progress. These limits, generally of thermodynamic and quantum mechanical nature[2]: as components grow smaller, the inevitable heat generation will become more significant, and quantum mechanical effects will become observable, introducing random effects in the calculations. Quantum computers, while presently still in their infancy, may offer the solution to several of these problems.

This report will provide an overview of the basic concepts of quantum computing, specifically the concepts of *qubits, gates* and *entanglement*. Further we will discuss some advantages and challenges in the field, notably algorithms which are particularly well-suited for quantum computers, including a description of a physical implementation that performs Shor's prime factorization algorithm.

2 Fundamentals of quantum computing

The *bit*, holding a value of either 0 or 1, is the basic unit of information in classical computing. While it is not difficult to envision a similar scheme in the context of quantum mechanics, there are notable properties of quantum mechanical systems that vastly change the behaviour compared to classical computing.

2.1 The qubit

Consider a quantum mechanical system having two eigenstates which we shall label $|0\rangle$ and $|1\rangle$. It is a basic property of quantum mechanics that such a system can be in any of the following states

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \quad \alpha^2 + \beta^2 = 1 \tag{2.1}$$

where α and β are complex numbers. Thus there are not only two distinct states, but infinitely many, occupying a two-dimensional Hilbert space. We shall refer to such a system as a *qubit*, and Section 5.1 will discuss ways to realize such systems physically. The constants α and β can serve as coefficients in a matrix representation of the qubit

$$\boldsymbol{\psi} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \tag{2.2}$$

Let us expand this system to include two qubits. Classically this would yield four possible states, namely 00, 01, 10 and 11. Quantum mechanically we can obtain the eigenstates of the composite system by evaluating the Kronecker products of the matrices representing each of the single-qubit systems' eigenvectors, for example

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{2.3}$$

This system, where each single qubit is in the state $|0\rangle$, shall be denoted $|00\rangle$. Evaluating the remaining combinations and applying analogous notation we see that this system is described by the *four eigenstates* $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$, defined compactly as

$$|q_1 q_2\rangle = |q_1\rangle \otimes |q_2\rangle\,, \quad \text{where} \quad \{q_1, q_2\} \in \{0,1\} \times \{0,1\}. \tag{2.4}$$

Thus, the eigenstates of a two-qubit system span a four-dimensional Hilbert space. Indeed for n qubits $q_1 \dots q_n$ we have a Hilbert space of dimension 2^n, the basis of which consists of all vectors of the form

$$|q_1 q_2 \dots q_n\rangle = |q_1\rangle \otimes |q_2\rangle \otimes \dots \otimes |q_n\rangle\,, \quad \text{where} \quad \{q_i\}_{i=1}^n \in \{0,1\}^n. \tag{2.5}$$

This finally allows any state of the system to be expressed as a linear combination

$$|\psi\rangle \;=\; \alpha_1 |00\dots0\rangle + \alpha_2 |0\dots01\rangle + \dots + \alpha_{2^n} |11\dots1\rangle\,, \tag{2.6}$$

where the absolute value of the sum of the squares of the coefficients must be unity. As in the former cases, the constants α_i serve as the coefficients in the matrix representation

$$\boldsymbol{\psi} = [\alpha_1, \dots, \alpha_{2^n}]^T \tag{2.7}$$

which will be used throughout the following sections. The property that the dimension increases exponentially with n is quite important: it means that the state of an n-qubit register actually encompasses 2^n complex numbers, which means that quantum registers can potentially hold extremely large amounts of classical data, supposing that a method exists whereby this data can be reliably extracted.

2.2 Quantum mechanical entanglement

A state is said to be *entangled* if it cannot be expressed as a Kronecker product of single-qubit states. Entanglement thus has no meaning for single qubits, but pertains only to *collections* of qubits.

Consider therefore a system of two qubits a and b. As noted above, the state of this system can be any superposition of the four eigenstates $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. The basis vectors themselves do *not* – by their very definition as Kronecker products – represent entangled states. Note that the concept of entanglement has no equivalent in classical physics.

There are four particularly simple entangled states of two-qubit systems that are called the *Bell* states:

$$\left|\Phi^{\pm}\right\rangle \;=\; \frac{1}{\sqrt{2}}\left(|00\rangle \pm |11\rangle\right), \tag{2.8}$$

$$\left|\Psi^{\pm}\right\rangle \;=\; \frac{1}{\sqrt{2}}\left(|01\rangle \pm |10\rangle\right). \tag{2.9}$$

The significance of entanglement is easily illustrated: suppose a measurement is performed on $|\Psi^+\rangle$. Since the system is either found in state $|01\rangle$ or $|10\rangle$, qubits a and b *cannot* have the same value. The result of a measurement of a alone will therefore immediately force qubit b into the opposite state, even if the physical systems comprising a and b are separated by a large distance!

Quantum entanglement is for this reason said to be a *non-local* property, and has thus been a subject of immense controversy although the phenomenon has later been observed experimentally.[3]

3 Quantum mechanical gates

Classical computers represent data by means of *bits* and manipulate data by means of logical *gates*. The last sections have dealt with the properties of the qubit; now it is therefore time to treat the concept of quantum mechanical gates.

A logical gate is generally a device which implements some logical operation on some number of bits. For example an AND gate takes two bits as input and produces an single bit of output, namely 1 if both inputs are 1, and otherwise 0. Quantum mechanical gates, corresponding simply to unitary operators, similarly perform operations on qubits. Because the operators are unitary, the number of output qubits must equal the number of input qubits. Indeed, the output system is actually a modification of the input system (and therefore belongs to the same space), unlike in classical computing where the output is actually a *separate copy* of the input system.

The simplest operators are the *unary* ones which operate on a single input. Since a single qubit belongs to a two-dimensional space, a unary operator must evidently have a representation in the form of a 2×2 matrix of complex numbers. All unary operators are therefore of the form

$$\mathbf{U} = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \in \mathbb{C}^{2\times 2}, \tag{3.1}$$

and in matrix notation the output state $|\Phi\rangle = \mathbf{\Phi}$ corresponding to a given input state $|\Psi\rangle = \mathbf{\Psi}$ is calculated by simple matrix multiplication. In the general case of n qubits we have

$$\mathbf{\Phi} = \mathbf{U}\mathbf{\Psi}, \quad \text{where} \quad \mathbf{\Psi}, \mathbf{\Phi} \in \mathbb{C}^{2^n}, \quad \mathbf{U} \in \mathbb{C}^{2^n \times 2^n}. \tag{3.2}$$

We shall now turn our attention to two-qubit systems, which are described by four-dimensional spaces. We shall use the usual basis

$$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\} \tag{3.3}$$

which will prove convenient since the lack of explicit superpositions ensures that these basis vectors have direct analogues in classical computing.

3.1 The XOR gate

As an example let us propose a quantum mechanical XOR gate. While XOR ordinarily has only one output (and thus cannot be described by a unitary operator), we can simply let one of the input qubits stay unmodified and put the "return value" into the other.

It follows from basic linear algebra that any operation (such as the XOR gate) can be uniquely defined by specifying its action on each of the basis vectors (this means that specifying its actions on classical bits is sufficient). Let therefore each input qubit be either $|0\rangle$ or $|1\rangle$. If the first qubit is $|0\rangle$ then both qubits are unmodified. If the first qubit is $|1\rangle$ the second qubit is negated.

The XOR operation is also known as the *controlled*-NOT, or CNOT, since the action on the second qubit is either nothing or NOT, depending on the value of the first qubit.

The resulting matrix representation is, as can easily be derived by applying the above mentioned rules on each basis vector,

$$\mathbf{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{3.4}$$

Surprisingly no such thing as a quantum mechanical AND gate can exist. From the result of an AND operation it is not always possible to replicate the input, since the output 0 is

yielded by several different inputs. The requirement of unitarity (specifically reversability) thus prevents the strict implementation of the `AND` and `OR` operations. However, by acting on one qubit it is possible to obtain an operation on the other qubit which is consistent with either `AND` or `OR`.

A particularly useful property of the `XOR` gate is that it can be applied to entangle qubits. Take for example the unentangled state

$$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |10\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \otimes |0\rangle\,. \tag{3.5}$$

Applying the `CNOT` operation yields

$$\begin{aligned} \texttt{CNOT}\,|\psi\rangle &= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \\ &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) = \left|\Phi^+\right\rangle, \end{aligned} \tag{3.6}$$

which is one of the Bell states.

3.2 The no-cloning theorem

In the article by [1] it is mentioned that the `XOR` operation cannot be used for copying a given state of a qubit in general. This is a result of what is known as the *no-cloning theorem*, which states that it is not possible to create identical copies of an arbitrary unknown quantum state. A short proof [4, p. 531] of the no-cloning theorem using a `XOR` operation can be given, assuming we have two pure quantum states $|\psi\rangle$ and $|s\rangle$, and want to copy $|\psi\rangle$ into $|s\rangle$. The initial state, $|C\rangle_1$, in the copying process is

$$|C\rangle_1 = |\psi\rangle \otimes |s\rangle\,. \tag{3.7}$$

If a unitary evolution U now acts on the copying, e.g. a `XOR` operation, we get

$$|C\rangle_2 = |\psi\rangle \otimes |s\rangle \rightarrow U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle\,. \tag{3.8}$$

Assuming this copying procedure works for two different pure states, $|\psi\rangle$ and $|\phi\rangle$, we have

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle \tag{3.9}$$

$$U(|\phi\rangle \otimes |s\rangle) = |\phi\rangle \otimes |\phi\rangle\,. \tag{3.10}$$

The inner product of these two equations yields

$$\langle\psi|\phi\rangle = (\langle\psi|\phi\rangle)^2 \tag{3.11}$$

Since the equation $x = x^2$ only has two solutions, $x = 0$ and $x = 1$, either $|\psi\rangle = |\phi\rangle$ or $|\psi\rangle$ and $|\phi\rangle$ are orthogonal. This is of course not the case in general, hence is general cloning of pure states using unitary evolution not possible. In the above, two assumptions were made: The state to be copied is a pure state and the operator is unitary. Fortunately these two assumptions cause no loss of generality, since the state can be purified if it is initially a mixed state, and an arbitrary quantum operation can be implemented if we introduce an ancilla and perform a suitable unitary evolution.

4 Potential quantum computer superiority

There are several computational tasks where a quantum computer would outrun todays classical computers. In this section an overview of these will be presented.

4.1 Prime factorization

One of the potentially most important applications for the quantum computer is its ability to factor large number much faster than classical computers. When a classical computer factorizes numbers it basically tries the numbers from end to end. If a computer thus were able to test 10^{10} different numbers per second, which is actually more than any computer built today can handle, the average time to find the factors of a 60-digit long number would exceed the age of the universe with billions of years.

Using quantum computers, on the other hand, factorization problems might be solved much faster than with classical computers. The quantum computer's ability to factor large numbers quickly derives from the quantum computer's ability to vastly parallelize the performance of a fast Fourier transform, using destructive interference among a number of parallel computation paths, that increases exponentially with the number of physical qubits involved in the computation [1].

One algorithm which can be used for factorization of large numbers using quantum computers is the so-called *Shor's* algorithm. Like many quantum computer algorithms, Shor's algorithm is probabilistic, meaning that it gives the correct answer with a high probability, and that the probability of failure can be decreased by repeating the algorithm. The fastest algorithm used for factorizing of a large number, n, whose representation has $\log_2(n)$ bits, on a classical computer runs in exponential time as

$$\exp\left(c(\log n)^{\frac{1}{3}} \cdot \log\log n\right)^{2/3}, \tag{4.1}$$

where c is a constant. In contrast, Shor's algorithm running on a quantum computer with $2n$ qubits would be able to do the same computation in a time that is polynomial in the logarithm

$$(\log n)^2 \cdot \log\log n. \tag{4.2}$$

Apart from factorization of large numbers the quantum computer is also capable of solving a related problem called the *discrete logarithm* problem. These abilities would enable the quantum computer to break many of the cryptographic systems in use today. The creation of a quantum computer could there have significant ramifications for electronic privacy and security.[1][9][8]

4.2 Quantum data-base search

Another area where quantum computers may be more effective than conventional computers is database search. A quantum computer using the so-called *Grover's algorithm* for a database search may be much faster than a classical computer using todays conventional search algorithms. When searching a database you need the search criterion, $p(x)$, that can be evaluated on any record x of the database. A search algorithm then searches for an x where $p(x) = 1$. In this context x can be adressed by a k-bit string, and the database can contain up to $N = 2^k$ records. A classical algorithm evaluates $p(x)$ one input at a time. In the quantum domain, however, the query p can be evaluated on records x or y

or a superposition of the two $(x+y)/\sqrt{2}$, with the result $[p(x)+p(y)]/\sqrt{2}$. This quantum mechanical property enables search with only $\sqrt{N}$ queries. In the classical case of serial computation, many more queries are required because an unsuccesful evaluation of $p(x)$ does not yield any new information about records other than x. Therefore anywhere from 1 to N or an average of $N/2$ queries are needed.[5]

4.3 Other applications

In 1982 Richard Feynman proposed that it would be possible to do simulations of quantum mechanical problems on quantum computers instead of classical computers. The speedup achieved by the quantum computer could be as great as for prime factorization. This could have great significance for medicine, biology and nanotechnology, because todays conventional computers take a very long time solving the computationally heavy quantum mechanical calculations.

5 Difficulties in quantum computation

A search on google scholar of the keywords "quantum computation" results in 324,000 hits. The amount of research being done in the field is staggering, yet an operational quantum computer comparable with todays computers is still many years into to the future. Although several advances has been made in the field, there are still a number of problems that need to be overcome in order to make an efficient quantum computer.

The system should be scalable so as to hold a large number of qubits. To create a quantum computer a large number of qubits must be created and controlled. One of the best quantum computers today has only 7 qubits - enough to factor the number 15[1], see Section 6. If an n-bit number is to be factorized, a $2n$-qubit quantum computer is needed. The problem is that the increase in difficulty of implementation rises greatly with the number of qubits.

Qubits can be initialized to arbitrary values. This problem is harder than it sounds. If a quantum computer should work, it must be able to place the qubits in arbitrary values. This could e.g. be a particular spin or polarization of a particle.

The decoherence time of the qubit should be long enough to make operations on them. The spontaneous collapse of the particle wavefunctions into a mixture of states (possibly entangled with the environment), called decoherence, is a very important factor when designing a quantum computer. If the decoherence time of the qubit particles is shorter than the gate operation time, the computer will not be able to do computations. This is because measurements of the states of the particles will be corrupted when they become entangled with the environment. This interaction is generally irreversible. It is therefore not all particles that can be used in a quantum computer, which could pose a problem when designing the computer. Furthermore qubits have a tendency to decohere when a measurement is performed on them. This is not very appropriate since it causes the invertibility of the quantum computational steps to be broken.

The gate set of a quantum computers must be *Turing-complete*. The Turing-completeness criterion basically means that the computer must be programmable and must be able to perform a computational task, i.e. emulating a universal Turing machine.

[1] Only 4 of the qubits are used for factorization

If a quantum computer does not fulfill the Turing-completeness criterion, it can never be used as a conventional computer.

It must be possible to perform single quantum sensitivity measurements. If the quantum computer only uses one copy of a qubit to perform the calculations it must be possible to do single-particle measurements. Although possible, this can be a problem since the particles can be electrons or photons, and because the measurements should be very precise.

Many of these problems have been solved individually, but the real challenge is to realize all of these criteria simultanously. There are however several groups who have shown remarkable progress in doing so.[9][1]

5.1 Physical quantum computer implementations

One of the fields where many of the criteria have been realized is the field of ion trap quantum computing. In the ion trap quantum computer the qubits are represented by the internal spin of individual ions held in an electromagnetic trap, see Figure 1. The more atoms in the trap, the more qubits in the system. The qubits can be set in a zero-state with a technique called laser cooling. Since the coupling to the environment in the ion trap is very low, the decoherence time of the particles is long enough to use them for computational operations. A technique called quantum-jump spectroscopy makes it possible to perform single quantum measurements with almost 100% efficiency. By coupling the spin of the ions with the quantum state of the vibration of the ions in the trap, it is furthermore possible to entangle the spin of the ions, thus making it possible to use all the entanglement-related features of the quantum computer. Unfortunately it has proved very hard to cool the trap to the ground state of motion using the laser cooling technique. This has only been done by one group for one or two ions[1]. In the next section another way of realizing a quantum computer will be discussed.

6 Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance

In the article [6] by L. M. K. Vandersypen et. al. an experimental quantum computing setup for computation of the prime factors of a given number using Shor's algorithm is presented. In a classical computer, the number of steps for finding the factors of an l-digit integer N increases exponentially with l. However, a quantum computer using Shor's algorithm, can make the same factorization in polynomial time, thereby decreasing the computation time from years to minutes. In [6] $N = 15$, hence the prime factors are 3 and 5. The setup is based on measuring the spin of seven spin-1/2 nuclei in a molecule using room temperature liquid-state nuclear magnetic resonance (or NMR).

6.1 Shor's algorithm

The problem of factoring a given integer into primes can be formulated as finding an integer p that divides N, where $1 < p < N$. Shor's Algorithm consists of two parts. The first part is to turn the factoring problem into a period finding problem, while the second part is to find the period r of a function $f(x)$, using an inverse quantum Fourier transformation (QFT). When the period is found, it is inserted into f and the prime factors are found immediately.

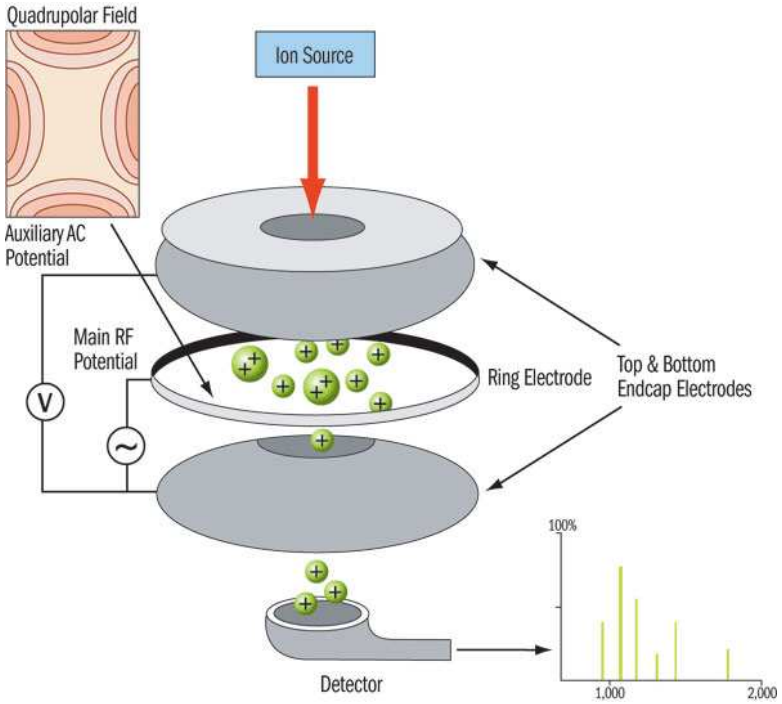

Figure 1: *The ion trap keeps the ions suspended in an electromagnetic field.*

We start by considering the function

$$f(x) = a^x \bmod N, \tag{6.1}$$

i.e. the remainder of a^x divided by N, and then try to find the smallest integer r for which $f(x+r) = f(x)$. a is randomly chosen with the restriction that $a < N$. The problem now is to find values of r that satisfy Equation 6.1. We start out with n qubits, in the initial state [8]

$$\frac{1}{\sqrt{n}} \sum_{x=0}^{n-1} |x\rangle |0\rangle . \tag{6.2}$$

Here $|x\rangle$ and $|0\rangle$ denote the states of two different registers called the *input* and *output* registers. Applying the function $f(x)$ on the input register we get, storing the result in the output register,

$$\frac{1}{\sqrt{n}} \sum_{x=0}^{n-1} |x\rangle |f(x)\rangle . \tag{6.3}$$

When using quantum computing, applying $f(x)$ once will apply it to all coefficients of the state, thus making the process parallel, and hence decreasing the computation time from exponential to polynomial compared to classical methods. A side effect of applying f(x) is that x collapses into an equal superposition, x', of each value of x, between 0 and $n-1$ that satisfy Equation 6.1. If one tried to measure the resulting superposition the state would collapse and information would be lost. We therefore apply the inverse QTF

$$U_{QFT} |x'\rangle = \frac{1}{\sqrt{n}} \sum_{y=0}^{n-1} \exp(-2\pi i x y/n) |y\rangle , \tag{6.4}$$

thereby getting the state

$$\frac{1}{\sqrt{n}} \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} \exp(-2\pi i x y/n) |y\rangle |f(x)\rangle . \tag{6.5}$$

If a measurement is made now, so that the input and output registers will hold the outcome y and $f(x)$, respectively, it can be shown that y will take values that are a multiple of $2^n/r$, with a very high probability. If the obtained result, which is only an informed guess and therefore denoted r', does not satisfy $f(x) = f(x+r')$, the correct r, is found by trying multiples of r'. Finally the factors of N are found by calculating the greatest common denominator (gcd) of $a^{r/2} \pm 1$ and N[2].

6.2 Quantum computing molecule

In the setup used in [6], the quantum computations are performed by a molecule consisting of five ^{19}F and two ^{13}C spin-1/2 nuclei, i.e. 7 qubits, see Figure 2. If the molecule is placed in a static magnetic field, each spin, i, has only two separate eigenstates, $|0\rangle$ or $|1\rangle$ (spin up or spin down). Since $N = 15$, four of the qubits are to be used for storing $f(x)$, i.e. the output register[3], while the remaining three are used for storing the result of the period

[2] See [8] for the mathematical proof

[3] The required number of qubits is found as $\log_2 N$, which in this case equals 4.

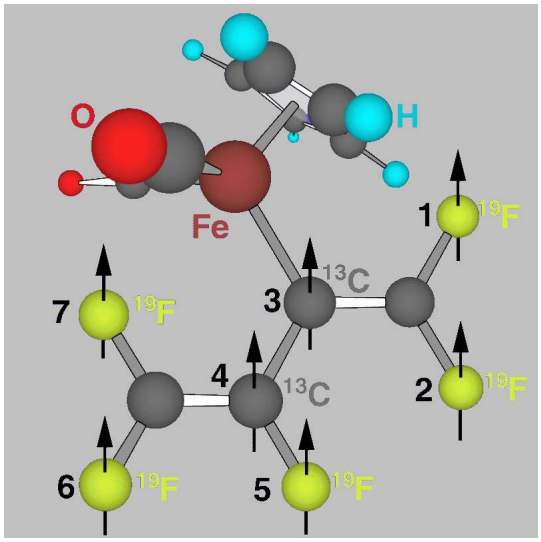

Figure 2: *The molecule,a perfluorobutadienyl iron complex, used for quantum computations consists of five* ^{19}F *and two* ^{13}C *spin-1/2 nuclei. The molecule is placed in a static magnetic field, hence the spins has only two discrete eigenstates.*

finding (that is the QFT), which is in the input register. In fact only two qubits are needed to find the period, but using three gives the possibility of detecting more periods and thus testing the QTF more thoroughly. The Hamiltonian for the molecule in the magnetic field is

$$H_0 = -\sum_i \hbar\omega_i I_{zi}, \tag{6.6}$$

where ω_i is the angular frequency and I_z is the $\hat{z}$ component of the angular momentum of the i^{th} spin. The density matrix for the molecule is initially determined by the thermal equilibrium, that is

$$\rho_{th} = \frac{\exp -H_0/k_bT}{2^7}. \tag{6.7}$$

The pairwise interaction of the 7 spins through the chemical bonds, that is the J-coupling, is given by

$$H_J = -\sum_{i<j} 2\pi\hbar J_{ij} I_{zi} I_{zj}, \tag{6.8}$$

Using spin-selective radio frequency pulses separated by time intervals of free evolution under the hamiltonian, the system can be manipulated into another computational state. The actual state of the three first qubits is estimated using NMR spectroscopy, using that

$$\rho \sim \sum_c w_c \left| c\frac{2^3}{r} \right\rangle \left\langle c\frac{2^3}{r} \right| \tag{6.9}$$

Before starting the Shor algorithm, the system is brought into a 7-spin effective pure state, using temporal averaging, with a NMR signal equivalent to $|\psi\rangle = |0000001\rangle$, see a)

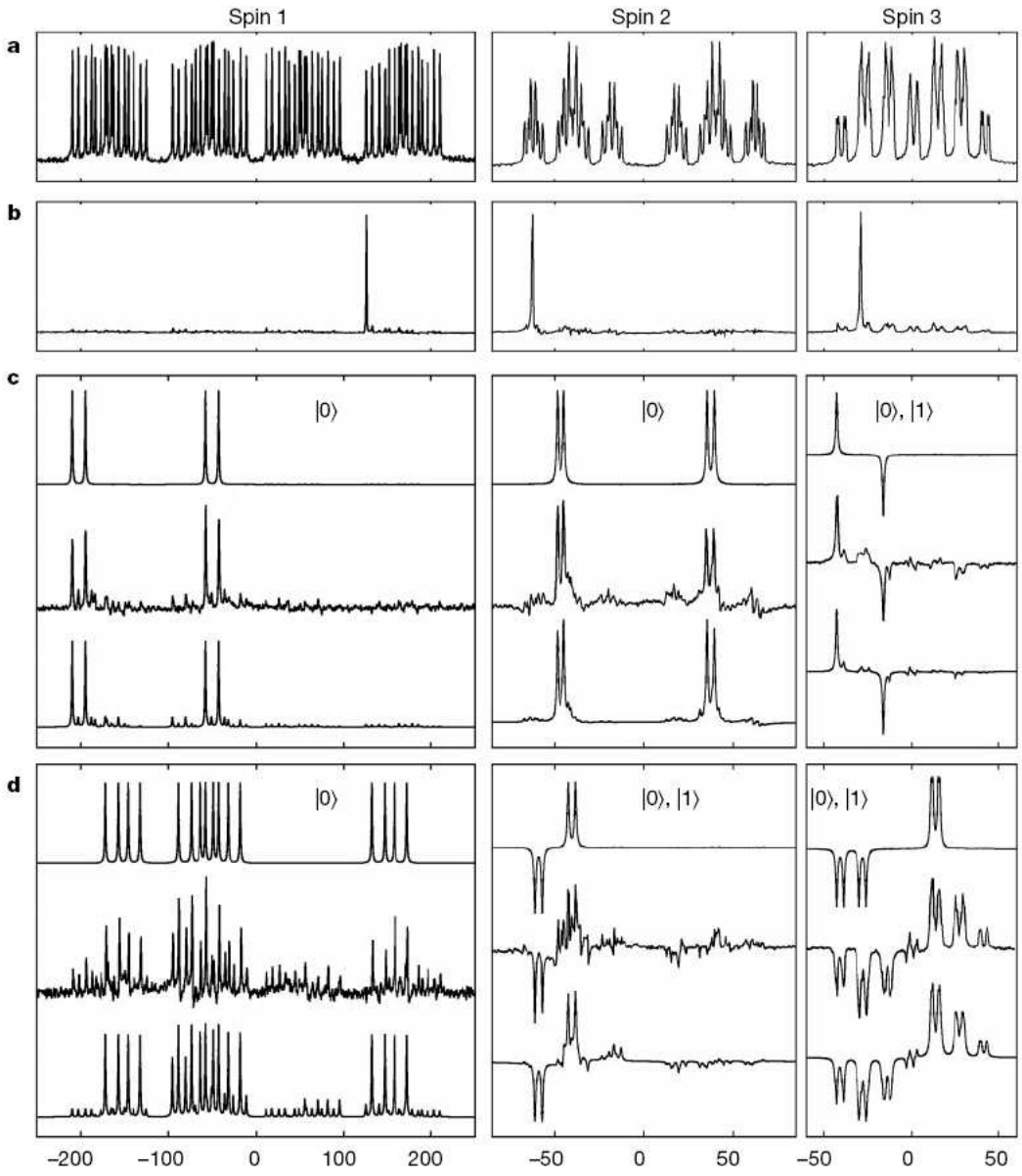

Figure 3: *Nuclear magnetic resonance (NMR) measurements of the first three qubits. a) is when the system is in thermal equilibrium. b) is when the system has been prepared for Shor's algorithm using temporal averaging. c) and d) is the output for* $a = 11$ *and* $a = 7$*, respectively. Both c) and d) has three traces where the top ones are the ideally expected results, the middle traces are the measured data and the bottom traces are simulated data with the effect of decoherence taken into account*

and b) on Figure 3. The computer is tested for two different choices of a, namely $a = 11$ and $a = 7$. The results are shown on Figure 3 c) and d). For $a = 11$ it is seen that qubit 1 and 2 has two spectral peaks pointing upwards, hence they are in the $|0\rangle$ state, while qubit 3 has a peak up and a peak down, hence it is in an equal mixture of $|0\rangle$ and $|1\rangle$. The input register is therefore in a mixture of $|000\rangle$ and $|100\rangle$ (qubit 3 is the first bit), or $|0\rangle$ and $|4\rangle$ in decimal notation. r can now be found as $r = 2^n/4 = 2$, and $\gcd(11^{2/2} \pm 1, N)$ the yields the two prime factors 5 and 3. Similarly for the $a = 7$ case, where qubit 1 is in the $|0\rangle$ state, and qubit 2 and 3 are both in a mixed state. The input register is therefore $|100\rangle$, $|110\rangle$, $|010\rangle$ and $|000\rangle$ or $|6\rangle$, $|4\rangle$, $|2\rangle$ and $|0\rangle$. This gives $r = 2^n/2 = 4$ and the prime factors are again $\gcd(7^{4/2} \pm 1, 15)$=$\{3,5\}$. Comparing the top and bottom simulated results in c) and d) on Figure 3 it is also seen that decoherence has too be taken into account when trying to describe the system, although there still are some deviations from the theoretically expected results. These deviations is partly due approximations in the model, partly due to imperfections in the experimental setup.

7 Conclusion

We have now completed our treatment of several aspects of quantum computing. First we have described formally how data can be represented in the form of qubits, their mathematical properties and significance.

Further we have introduced the notion of gates, along with a mathematical description of the interaction of qubits through these, the properties of entangled states and the significance of the no-cloning theorem.

This has allowed us to describe how quantum computers have fundamental advantages within certain problems such as Shor's algorithm for prime factorization in polynomial time, which can revolutionize the field of data encryption, along with Grover's algorithm which can provide substantial performance improvements to database searches.

Last we have considered a specific implementation of a quantum computer which has been used to actually perform prime factorization, albeit only of the number 15. Still the successful implementation even on such a small scale suggests that the ideas behind quantum computers are indeed feasible and may become efficient tools some time in the future.

References

[1] *Quantum Information and Computation*, Charles H. Benneth and David P. DiVincerro, Macmillan Magazine Ltd. Nature vol. 404, 2000

[2] *Limits on Silicon Nanoelectronics for Terascale Integration*, James D. Meindl et al., Science, September 2001

[3] *Experimental quantum teleportation*, Dik Bouwmeester, Jian-Wei Pan, Klaus Mattle, Manfred Eibl, Harald Weinfurter & Anton Zeilinger. Nature vol. 390, 1997.

[4] *Quantum Computation and Quantum Information*, Michael A. Nielsen and Isaac L. Chuang, Cambridge University Press, 2004

[5] *Is Quantum Search Practical?*, George F. Viamontes, Igor L. Markov, and John P. Hayes. arXiv:quant-ph/0405001 v1 30 Apr 2004.

[6] *Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance*, Lieven M. K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Mark H. Sherwood & Isaac L. Chuang. Nature vol. 414, 2001

[7] Quantiki article on quantum gates,
`http://www.quantiki.org/wiki/index.php/Quantum_gates`

[8] Wikipedia article on Shor's algorithm,
`http://en.wikipedia.org/wiki/Shors_algorithm`

[9] Wikipedia article on quantum computers,
`http://en.wikipedia.org/wiki/Quantum_computers`