Turing in Quantumland

Harry Buhrman*
Centrum Wiskunde & Informatica
University of Amsterdam
buhrman@cwi.nl

Abstract

We revisit the notion of a *quantum Turing-machine*, whose design is based on the laws of quantum mechanics. It turns out that such a machine is not more powerful, in the sense of computability, than the machine originally constructed by Turing. Quantum Turing-machines do not violate the Church-Turing thesis. The benefit of quantum computing lies in *efficiency*. Quantum computers appear to be more efficient, in time, than classical Turing-machines, however its exact additional computational power is unclear, as this question ties in with deep open problems in complexity theory. We will sketch where BQP, the quantum analogue of the complexity class P, resides in the realm of complexity classes.

1 Introduction

A decade before Turing developed his theory of computing, physicist struggled with the advent of quantum mechanics. During the famous 5th Solvay Conference in 1927 it was clear that a new era of physics had surfaced. Its strange features like superposition and entanglement still lead to heated discussions and much confusion. However strange and counter-intuitive, the theory has never been refuted by experiments that are performed daily and in great numbers throughout laboratories around the world. Time after time the predictions of quantum mechanics are in full agreement with experiment.

Shortly after the advent of quantum mechanics, Church, Turing and Post developed the notion of computability [Chu36, Tur36, Pos36]. Less than 10 years later these formal ideas would be put to practice resulting in the ENIAC, the first general purpose machine. During that time Turing also specified an electromechanical machine that helped break Enigma-ciphers during the Second World War. It was not the first time that mechanical computing and cryptanalysis were developed side by side. Around 1820 Babbage worked on the blueprints of a mechanical

*Science Park 123, 1098 XG, Amsterdam The Netherlands. Partially supported by the EU 7th framework project QCS.

computing device, which he called Difference Engine #1 and #2. The first prototype was never finished and funding was cut for the second prototype. Babbage was also a gifted cryptanalyst, in 1854 he broke the, until then unbreakable, Vigenère cypher [Sin00].

Today Computers occupy an indispensable place in our every day life, leading to an ever-lasting demand on faster and bigger computing power. The mathematical notions developed in the the 1930's are now firmly rooted in our physical world. As Landauer [Lan61] put it: "computing is physical". Indeed, trying to satisfy Moores law [Moo65], which states that the number of transistors on integrated circuits doubles approximately every two years, the physical limitations of computing have become apparent. In particular quantum mechanical effects are starting to pop up. Incorporating quantum mechanics into our computational paradigm is therefore the next logical step.

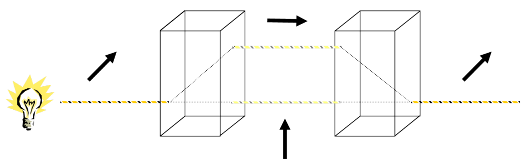
In this chapter we will review quantum mechanics and show how this lead Deutsch [Deu85] to the Definition of a quantum Turing-Machine. The new field of quantum information processing gained a lot of momentum after Shor [Sho94] constructed an efficient quantum algorithm for the factorization problem. Since the security of most of modern public-key cryptography is based on our *inability* to factorize numbers efficiently, a quantum computer, once built, will severely compromise the security of these protocols. It is interesting to see that again a new computing paradigm is developed alongside advances in cryptanalysis. Quantum cryptography is the subfield that studies cryptography in a quantum world, with as cornerstone the quantum key distribution protocol by Bennet and Brassard [BB84]. The field has also developed new areas like quantum communication complexity [BCMdW10], quantum information theory [BS98], and quantum inspired proofs [DdW11]. We won't discuss these developments here, but work out the basic definitions and show how this new paradigm fits into the classical framework of complexity theory.

2 Quantum Mechanics

Quantum mechanics is the most complete description of nature to date, that governs the atomic and subatomic world. Some of its features are very counter intuitive, but have been corroborated by experiments time after time. The best way to highlight some of the properties of quantum mechanics is by means of an experiment, which we will discuss next.

2.1 An experiment with photons

In the first part of the experiment we have a light source that emits light that is polarized at an angle of 45°. It is not necessary to understand in detail what polarized light exactly is. The only important issue at this point is that polarization is a *property* of light, which has a *direction*, that can be measured. The outcome of such a measurement can be expressed by an angle. In our experiment we shine light with a certain polarization through a calcite crystal. A calcite crystal is transparent and has the property of birefringence, a light beam polarized in a certain direction gets split into two beams (see Figure 1). The birefringence of calcite crystals is known for a long time and a recent article [RGF$^+$12] argues that the Vikings used the calcite crystal, the mysterious “sunstone” according to the Norse sagas, to determine the position of the sun on cloudy days. Perhaps the first quantum computer avant la lettre!


(a) two beams interfere

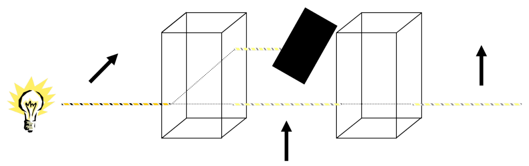
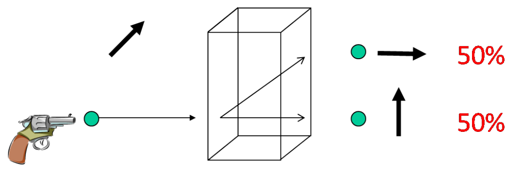
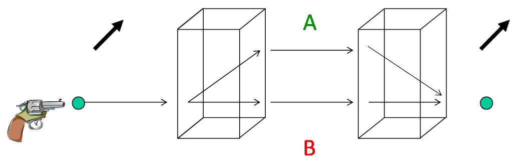

(b) no interference

Figure 1: Polarized light going through two calcite crystals

The 45° polarized light is sent through the first calcite crystal and is split into two beams (Figure 1a). When measured right after exiting the first crystal, the light in the upper beam turns out to have horizontal polarization whereas the lower beam is vertical. The light in these two beams has only half the *intensity* of the original light coming directly out of the source. If we don’t measure the polarization but let the two beams enter a second crystal, they merge and exit the second crystal as one beam, which has full intensity. The polarization after the second crystal is again 45°. This can be explained classically as an interference effect. By blocking the upper beam with a piece of black material (Figure 1b), interference is no longer possible and the light emitted by the second crystal is vertically polarized.



(a) single photons



(b) path A or B

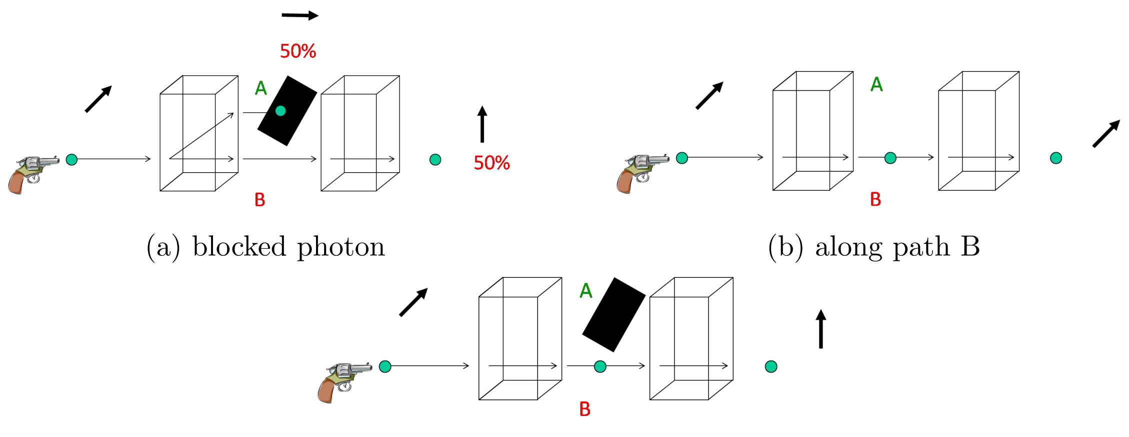
Figure 2: Single photon experiment

Lets now do the same experiment with a very very dim light source. We dim the light so much so that only *single* photons are produced each time we do the experiment. This second experiment is much harder than the first, but very good single photon sources have been accomplished [EFMP11]. Single photons still have a polarization and the photons that we use are again polarized at an angle of 45°.

The first observation (Figure 2a) to make is that in every run of the experiment the photons end up at two different spots, precisely where in the first experiment the two light beams were. Moreover, after doing the experiment many times, 50% of the time we find a photon at the top position and 50% of the time at the bottom. This, in itself is already a bit strange. What determines which path a photon takes? The ones coming from the source are all created in the same way, as much as possible.

When we measure the polarization of the photons that have gone through the calcite crystal it turns out that the top ones are always horizontally polarized and the bottom ones vertically. This is all in perfect agreement with the first experiment using a bright light source.

Next we place another calcite crystal and measure the photon when it exits the second one (Figure 2b). The photons always come out in the same place and when their polarization is measured it is again 45°. From all we observed so far it seams reasonable to assert that each photon either took the path labelled A in Figure 2b or path B. Lets examine this assertion by



(a) blocked photon

(b) along path B

(c) along path B with path A blocked

Figure 3: examining the paths

blocking path A (Figure 3a) and measuring the photons that end up there. We find that half of the time the photons travel along path A and their polarization is always horizontal, as it should be in order to be consistent with Figure 2a. Moreover whenever the photons don't end up at the blockage along path A, they end up behind the second crystal and have polarization vertical.

Our assertion says that the photons took either path A or path B. We see that half of the time the photons go along path A. Lets focus on the ones that take path B (Figure 3b). These photons exit the second crystal and have a polarization of 45° in total agreement with the situation in Figure 2b. But now we have a very strange situation when we consider these photons, going along path B, when we also block path A as in Figure 3c. In these cases the photon exits the second crystal with vertical polarization. It appears as if the photon that takes path B *knows* whether path A is blocked or not and changes polarization accordingly. The way out of this conundrum is given by quantum mechanics. The photon in Figure 2b does not travel along path A or B, it is in a *superposition* of going along A and B at the *same* time. Moreover it interferes with itself at the second calcite crystal. In the next Section we will formalize this behavior.

2.2 Qubits, superposition and measurement

Superposition is one of the main and counter intuitive ingredients of quantum mechanics. As we saw in the experiment above, a photon can be in a superposition of being in two different locations at the same time. But the a superposition principle also applies to larger systems. Famous is the example of Schroedinger's cat [Sch35] who is in a superposition of being dead and alive. In what follows we will apply these ideas to more familiar objects: bits. Classically a bit can be in any of two states: 0 or 1. Quantum mechanically a quantum bit or *qubit* may be in a superposition of both 0 and 1. It is useful to describe such systems as vectors in a finite dimensional Hilbert space, in this case a two dimensional one. We will identify the (basis) vector $\begin{pmatrix}1\\0\end{pmatrix}$ with $|0\rangle$ to denote the classical bit 0 and (basis) vector $\begin{pmatrix}0\\1\end{pmatrix}$ with $|1\rangle$ denoting the classical bit 1. This notation is called Dirac or bra-ket notation. Define for any vector $|a\rangle$ the complex conjugate transpose $\langle a|$, the expression $\langle a| \cdot |b\rangle = \langle a|b\rangle$ then boils down to the inner product between $|a\rangle$ and $|b\rangle$. *Superposition* of two classical (basis) states/vectors, in this case the $|0\rangle$ and $|1\rangle$, is modelled as follows:

$$\alpha|0\rangle + \beta|1\rangle \tag{1}$$

Where α and β, called *amplitudes*, are complex numbers with the property that:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2}$$

When one *observes* or *measures* a qubit $\alpha|0\rangle + \beta|1\rangle$ the outcome 0 is obtained with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$. After the measurement has been performed and one of the outcomes has been observed, the state *collapses* to the outcome that has been observed. For example if outcome 0 was observered, the state has collapsed to the state $|0\rangle$. Note that Equation 2 guarantees that measurement of a qubit induces a probability distribution over the outcomes 0 and 1.

Let's try to plug in some values for α and β:

$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \tag{3}$$

Measuring this qubit will yield with probability $\frac{1}{2}$ in observing a 0 and with probability $\frac{1}{2}$ a 1.

In general our system will consist of more than just one qubit. If we have two qubits $|x\rangle$ and $|y\rangle$ then $|x\rangle \otimes |y\rangle$ describes the two qubits together in a 4 dimensional Hilbert space. This construction is called the tensor or Kronecker product. If $|x\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and $|y\rangle = \beta_0|0\rangle + \beta_1|1\rangle$ then

$$\begin{aligned} |x\rangle \otimes |y\rangle &= (\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle) \\ &= \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle. \end{aligned}$$

By convention $|0\rangle\otimes|0\rangle$, $|0\rangle|0\rangle$, and $|00\rangle$ will denote the same state. The 4 dimensional Hilbert space obtained in this way has a natural basis, called the computational basis: $|00\rangle, |01\rangle, |10\rangle$, and $|11\rangle$, and any two qubit state can be expressed as a linear combination/superposition of these basis states. Any state on n qubits becomes:

$$\sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \tag{4}$$

with the additional requirement that:

$$\sum_{i \in \{0,1\}^n} |\alpha_i|^2 = 1 \tag{5}$$

When measuring these n qubits we will observe i with probability $|\alpha_i|^2$.

In general not all the 2 qubit states that satisfy Equations 2 and 4 are obtained as the tensor of two single qubits. Such states are called *entangled.* For example the well known EPR-pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is entangled.

2.3 Partial measurement

In the previous Section we saw how to measure a quantum system

$$\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle \tag{6}$$

in the computational basis. Such a measurement induces a probability distribution over the outcomes x.

$$\Pr[\text{outcome} = x] = |\alpha_x|^2 \tag{7}$$

The state after measuring outcome x will collapse to $|x\rangle$

For the Definition of the quantum Turing machine it will be important to also be able to *partially* measure a system. This is a measurement that only "looks" at part of the state. For example take the following two register state:

$$\sum_{x,y \in \{0,1\}^n} \alpha_{x,y} |x\rangle \otimes |y\rangle \tag{8}$$

on which we measure the first register in the computational basis. This will result in a probability distribution over outcomes of the first register:

$$P_x = \Pr[\text{outcome} = x] = \sum_y |\alpha_{x,y}|^2 \tag{9}$$

Again the state will collapse to x in the first register, but the second register stays in a superposition that is consistent with outcome x:

$$\frac{1}{\sqrt{P_x}} \sum_y \alpha_{x,y} |x\rangle \otimes |y\rangle \tag{10}$$

The $1/\sqrt{P_x}$ factor renormalizes the partially collapsed state, so that it has norm 1.

2.4 Unitary Operations

Next we need to model operations on qubits, such evolution of the system, according to quantum mechanics, is a *linear* map with the additional constraint that it preserves the probability interpretation, that is the squares of the amplitudes sum up to 1 (see Equations 2 and 5). Such norm-preserving linear transformations are called *unitary* and can be defined in mathematical terms:

$$UU^* = I \tag{11}$$

Where U^* is the complex conjugate transpose of U and I is the identity matrix. In terms of computation the unitary constraint implies that the computation is *reversible.*

The following transformation on a single qubit is important and very useful. It is called the Hadamard transform.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{12}$$

It is a unitary operation since:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \times \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Let's do a Hadamard operation on a qubit that is in the classical state $|0\rangle$:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{13}$$

This state in ket notation, $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, is the random qubit from Equation 3. Also the Hadamard transform models the behavior of the calcite crystal from Section 2.1, it puts the photon in an equal superposition of the two paths, and the probabilistic nature of where we found the photon (figure 2a) is now explained by the measurement axiom!

When we apply the Hadamard transform again on this qubit:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \times \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} + \frac{1}{2} \\ \frac{1}{2} - \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{14}$$

We get the $|0\rangle$ again. Note the minus sign in the Hadamard transform. Its effect is illustrated in Equation 14. The minus sign caused the $\frac{1}{2} - \frac{1}{2}$ in the lower half of the vector to cancel out, or destructively interfere, while both terms in the upper half constructively interfered. It is both the superposition principle together with this interference behavior that gives quantum computing its power.

The tensor product is also defined on linear operations. In general if we have an $m \times n$ matrix A and an $n' \times m'$ matrix B then $A \otimes B$ is a $(m \cdot m') \times (n \cdot n')$ matrix defined as:

$$\begin{pmatrix} a_{1,1} \cdot B & a_{1,2} \cdot B & \dots & a_{1,n} \cdot B \\ a_{2,1} \cdot B & a_{2,2} \cdot B & \dots & a_{2,n} \cdot B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} \cdot B & a_{m,2} \cdot B & \dots & a_{m,n} \cdot B \end{pmatrix}$$

For example applying the Hadamard transform to n qubits ($H^{\otimes n}$) in the state $|0\rangle$ will generate a uniform superposition of all the basis states.

$$\overbrace{H|0\rangle \otimes \ldots \otimes H|0\rangle}^{n} = H^{\otimes n}|0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} |y\rangle \tag{15}$$

3 Quantum Turing Machines

In order to appreciate the Definition of a quantum Turing-machine we will first remind the reader of the Definition of a classical Turing Machine, see for example [AB09]. This is a device that has a two-way infinite tape of cells, which are ordered and labelled by an integer, and a read/write head that can move left and right ($\{L, R\} = D$) on the tape and write a symbol from a fixed finite alphabet Σ in the cell that it is currently reading. There is also a finite control that governs the behavior of the Turing Machine. This is modelled by a finite set of states Q, a dedicated starting state q_0, and a final halting state q_f. The transition function δ describes a single computation step.

$$\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times D \tag{16}$$

When the TM is in state $q \in Q$ and reading symbol $\sigma \in \Sigma$ at the current location of the head, $\delta(q, \sigma) = (\sigma', q', d)$ indicates to write σ' in the current tape cell, move the tape-head one cell in direction $d \in D$, and go to state q'. The computation starts with the input $x \in \Sigma^n$ written in the first n cells, the tape-head is at the cell with label 0 and the TM is in state q_0. All the cells, except for the first n have a special symbol[1] # written in them indicating that this cell has no content. During each step of the computation, the transition function δ is applied as described above. The computation halts when it enters the final state q_f, and the non-empty contents of the tape indicate the result of the computation.

A quantum Turing Machine [Deu85] is a generalization of the classical one, extended with quantum mechanical properties. As before we describe a QTM by means of its transition function.

$$\delta : Q \times \Sigma \rightarrow \mathbb{C}^{Q \times \Sigma \times D} \tag{17}$$

The way to interpret this is as follows. In the classical case δ changed the configuration (a, q, m) of a TM, where a is a description of the tape contents, q is the state of the TM, and m is the place where the head is located, to an new configuration (a', q', m'). In the quantum case δ defines a similar operation, but it now maps a state $|a, q, m\rangle$ to a superposition of states $\sum_{a',q',m'} \alpha_{a',q',m'} |a', q', m'\rangle$, where the a' are the tape contents that differ in at most one location from a, and $m' = m \pm 1$. Moreover the resulting state has to be norm one: $\sum_{a',q',m'} |\alpha_{a',q',m'}|^2 = 1$, which implies that δ is restricted to implement a unitary transformation on the configuration space of the QTM. Since δ specifies the behavior on basis states $|a, q, m\rangle$,

[1]We assume that # is an element of Σ.

by linearity it is also defined on super positions of basis states. One step of the QTM maps a superposition of basis configurations to a new superposition:

$$\sum_{a,q,m} \alpha_{a,q,m}|a,q,m\rangle \xrightarrow{\delta} \sum_{a,q,m} \alpha'_{a,q,m}|a,q,m\rangle \tag{18}$$

Observe that at any point in time the sums in Equation 18 contain a *finite* number of terms with non-zero amplitude. The measurement axioms of quantum mechanics tell us what happens when we measure the QTM in configuration: $\sum_{a,q,m} \alpha_{a,q,m}|a,q,m\rangle$. We will observe the classical configuration $|a,q,m\rangle$ with probability $|\alpha_{a,q,m}|^2$. However after making this measurement, and observing, for example configuration $|a,q,m\rangle$, the original superposition has collapsed to the state $|a,q,m\rangle$.

As with classical TM's a QTM starts in the starting state q_0, the input $x \in \Sigma^n$ is written in the first n cells, and the head is at the cell with label 0, reading the first symbol of x. Each computation step is now applied according to Equation 17. With a classical Turing Machine it is clear when the computation is over, namely when the machine is in the final configuration q_f. How to define this with a QTM? Since the configuration of a QTM can be a superposition of states that are the final sate q_f and other states, this may not be well defined. Moreover how can one determine when a QTM has halted, since observing a machine that has not yet halted may collapse its superposition and thus disturb the computation. One way out is to add an additional register to the quantum state, that indicates whether the QTM has halted: $|a,q,f,m\rangle$, where $f = 1$ indicates that the QTM has halted and $f = 0$ means that it is still running. The transition function from Equation 17 needs to be extended to deal with this extra register:

$$\delta : Q \times F \times \Sigma \to \mathbb{C}^{Q\times F\times\Sigma\times D} \tag{19}$$

where $F = \{0,1\}$. In terms of configuration space, an application of δ translates to:

$$\sum_{a,q,f,m} \alpha_{a,q,f,m}|a,q,f,m\rangle \xrightarrow{\delta} \sum_{a,q,f,m} \alpha'_{a,q,f,m}|a,q,f,m\rangle \tag{20}$$

The computation starts in state q_0 and $f = 0$. After each computation step only register f is measured, leaving the remaining registers unmeasured. Such a partial measurement (see Section 2.3 can affect the total state, but when this flag register is not entangled with the other registers, such a measurement will not effect the remaining state. If this partial measurement yields $f = 0$ then δ is applied again to the state, over and over again, until $f = 1$ is measured in which case the computation has ended, and the remaining register is measured[2] in the computational basis and the output of the computation can be read off. Note that halting of such a computation is a probabilistic event. Just like for a classical TM it has to halt after a finite amount of steps, we require that the *expected* number of steps that the QTM takes is

[2]It is not necessary to measure the remaining register. The output of a QTM could also be a quantum state, something that a classical TM simply can not output. Here we will only be concerned however with classical outputs.

finite. Since the computation is probabilistic in nature we only demand that with probability $\frac{1}{2}+\epsilon$ the function value is computed, for $\epsilon > 0$.

A few more words about the description of the transition function δ are in place. In order to yield a finite description of δ it is necessary that the complex numbers in 19 come from a finite subset and have an efficient description. It turns out that this is not a problem since it can be proven that just a finite number of transformations are sufficient to approximate *any* unitary function arbitrary well by composing them using tensor products and products. This also forms the main ingredient for proving that there exists a universal quantum Turing machine [Deu85, BV97]. For example the Hadamard transform (H) and the rotation over $\pi/4$ (R), and the phase flip (S) form a universal set of operations:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad R = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{21}$$

As mentioned before the transition function δ corresponds to a unitary matrix. We can also see how for example the Hadamard matrix H corresponds to some transition function. For example it could be implemented as follows:

$$(q, 0, 0) \to \frac{1}{\sqrt{2}}(|q, 0, 0, L\rangle + |q, 0, 1, L\rangle) \tag{22}$$

$$(q, 0, 1) \to \frac{1}{\sqrt{2}}(|q, 0, 0, L\rangle - |q, 0, 1, L\rangle) \tag{23}$$

The right-hand side of Equations 22 and 23 are a vector of complex numbers, in this example with entries $\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}$, and 0, as is the right-hand side of Equation 19.

Another important point is that, due to the unitarity of the transition function, the computation of any QTM is *reversible*. Since the transition function δ is a unitary transformation U_δ, it follows that U_δ^*, its complex conjugate transpose, also models a transition function. Since $U_\delta U_\delta^* = I$, it follows that U_δ^* reverses a computation step. This seems to be a severe restriction since classical computations do not have to be reversible. However it possible to transform any non-reversible computation into a reversible one with only a little overhead in time and space [Ben89].

Summarizing, we have the following Definition.

Definition 1. *A QTM is specified by its transition function*

$$\delta : Q \times F \times \Sigma \to \mathbb{C}^{Q \times F \times \Sigma \times D}$$

which represents a unitary transformation on the configuration space. One step of the machine is the application of δ to each of the basis states, followed by a partial measurement of the F register. A QTM computes a function f if the expected number of computation steps is finite for every input x and it outputs $f(x)$ with probability greather than $\frac{1}{2}+\epsilon$, for some $\epsilon > 0$

4 The Church-Turing thesis

We will now examine the computational power of a QTM and prove that it does not exceed that of ordinary TM's.

Theorem 2. *[Deu85] The class of languages accepted by QTM's is equal to the computable sets.*

Proof. Since every TM can be transformed in one that is reversible it is clear that the class of languages accepted by QTM's contains the computable sets. We only need to show that it is not larger. We do this by simulating a QTM by means of a classical algorithm. Given a QTM M and its δ function, we need to establish what it outputs with probability $\frac{1}{2}+\epsilon$ conditioned on halting. We have already remarked that after t steps the number of terms with non-zero amplitude in 20 is finite. This is because in each step the number of non-zero configurations, can go in a super-position of at most a finite number of new ones, specified by δ. The idea now is for t to compute the configuration vector $|v_t\rangle$ after t steps.

$$|v_t\rangle = \sum_{a,q,f,m} \alpha_{a,q,f,m} |a, q, f, m\rangle \tag{24}$$

This involves doing the partial measurements also $t-1$ times, and collapsing $|v_{t'}\rangle$ ($t' < t$) to the state corresponding to $f = 0$ as outcome of the partial. This way we can compute the probability of halting after exactly t steps, which we call p_t. We know that the expected running time of M is finite:

$$\sum_t p_t \cdot t \leq c \tag{25}$$

There could be runs of the machine that never halt, but they occur with vanishing probability. In particular, using Markov's inequality, there is time $t_0 \leq c/\epsilon$ such that the probability that the machine runs for more than t_0 steps is bounded by ϵ, which implies that

$$\sum_{t \leq t_0} p_t \geq 1 - \epsilon \tag{26}$$

Since we can compute p_t we can find t_0. Next we compute the probability of accepting conditioned on halting within t_0 steps. We accept the input if and only if this conditional accepting probability is $> \frac{1}{2}$. □

It is easy to see that the above argument also works for functions instead of languages. Theorem 2 shows that the Church-Turing thesis, which states that effective computability is captured by Turing machines, is not violated when we introduce quantum mechanical Turing-machines.

4.1 Efficient Church-Turing thesis

Although one can not compute more functions on a quantum computer, the proof sketched above shows that an exponential overhead in time is used in order to *simulate* a QTM by a classical one. Therefore Deutsch [Deu85] asked the question of whether it is possible that a quantum computer is more *efficient* than its classical counterpart. This question was related to Feynman [Fey82], who asked whether physics could be efficiently simulated on a Turing-machine. This touches upon an extension of the Church-Turing thesis that does not only ask about simulation of an algorithmic process by a Turing machine but also requires that the simulation is efficient, with only polynomial overhead in time and space [vEB90].

Deutsch gave some indication that indeed QMT's can be more efficient, by exhibiting a quantum algorithm that determines for any $f : \{0,1\}^n \to \{0,1\}$, what the value of $f(x) \oplus f(y)$ is, querying f only one time. In particular Deutsch argues, if $f(x)$ and $f(y)$ are computed on a classical TM and each computation costs a day, then it would cost two days to compute $f(x) \oplus f(y)$, but on a quantum computer it would only cost one day. Deutsch's Definition of a quantum Turing machine and his quantum algorithm mark the start of the field of quantum information processing. See also the excellent book [NC00]. These algorithms are best described in the black-box setting. See [BBC$^+$01, BdW02] for precise Definitions.

The mother of all quantum algorithms was generalized by Deutsch and Jozsa [DJ92] showing that there exists a problem that can be computed with a single query, but classically requires n queries when the solution has to be computed exactly. The drawback of this (super)exponential separation is that there does exist an efficient randomized algorithm that solves this problem also in a constant number of queries. Building upon this, Simon [Sim97] managed to construct a quantum algorithm, that truly establishes an exponential separation between quantum and randomized computations. All these separations are proven in the black-box or oracle setting, which essentially only counts queries to the input f.

A major advance was made by Shor [Sho94] who constructed, extending Simon's ideas, an efficient quantum algorithm to factorize numbers in their prime factors in time $O(n^3)$. Shor's quantum algorithm is special in that it is not a black-box algorithm, although at the heart of it resides a black-box procedure, called period-finding. The problem of factorization is well studied and it is believed that no fast (randomized) classical algorithms exists. The best known classical algorithm has expected running time $O(2^{n^{1/3}\log(n)^{2/3}})$.

The relevance of an efficient quantum algorithm for factorization is not just academic. The security of most public-key cryptography techniques, like RSA [RSA78], that are used frequently by numerous applications, rely on the absence of fast factorization algorithms. These cryptographic protocols become *insecure* when an efficient factorization method can be implemented. In particular they will be rendered useless when Shor's algorithm can be executed on a quantum computer that operates on a few thousand qubits.

After Shor's algorithm the field of quantum information processing got a tremendous boost and evolved into a flourishing community with active researchers from experimental and theoretical physics, computer science, and mathematics.

We need to mention one more important algorithm, which is due to Grover [Gro96]. This algorithm is able to search a marked item in an unordered list with n items, using only $O(\sqrt{n})$

look-ups or queries. Classically $\Omega(n)$ queries are necessary. Though not as impressive as Shor's speed-up, Grover's is only a quadratic, search is a prominent primitive in many algorithms, and it is not surprising that variants of Grover's algorithm yield a quantum advantage in many computational settings.

Where does this leave us with respect to the extended Church-Turing thesis? In order to answer that question we need to have a better understanding of efficient (classical) computation, which is the realm of complexity theory.

4.2 Complexity theory

Complexity theory is the area of mathematics and theoretical computer science, that studies the question of the efficiency of optimal algorithms for a broad class of computational problems. Central is the class of problems that admit an efficient algorithm. An algorithm is efficient if its running time is upper-bounded by a polynomial in the size of the input.

Definition 3. *A language or set $A \subseteq \Sigma^*$ is in the complexity class P if there is a polynomial $p(n)$ and a Turing machine M, such that $M(x) = A(x)$ for all x and the running time of $M(x)$ is bounded by $p(|x|)$-many steps.*

Many problems that arise in practice are not known to be in P nor do we have a proof that no efficient algorithm exists. In order to study the complexity of these problems the complexity class NP was introduced by Cook and Levin [Coo71, Lev73].

Definition 4. *A is in NP if there exists a TM M and a polynomial p such that:*

$$x \in A \Leftrightarrow \exists y : M(x, y) = 1$$

with $|y| \leq p(|x|)$ and M runs in time $p(|x| + |y|)$.

The name NP is somewhat esoteric, it is an abbreviation for Non-deterministic Polynomial-time. NP is the class of problems for which there is an efficient procedure to *verify* that a given solution to a problem instance is correct. Such correct solutions are also called witnesses. An example of an NP problem is the satisfiability problem SAT. Given a formula ϕ in conjunctive normal-form on n variables, $\phi \in$ SAT iff there exists an assignment $\alpha = \alpha_1 \ldots \alpha_n$ to the n variables such that $\phi(\alpha)$ evaluates to true. It is easy to see that SAT is in NP, since it is easy to check whether an assignment α satisfies ϕ, but the best known algorithm requires 2^n time steps, which tries all 2^n possible assignments.

Cook and Levin showed that SAT and a handful of other computational problems are in NP and in fact *characterize* NP in the sense that each of them is in P if and only if all of NP is in P. Such problems are called NP-complete. The handful of problems has grown to a long list of problems that come from many areas of science and operations research [GJ79], with new ones being added every year. The question of whether P=NP is one of the central open problems in mathematics and computer science. It is also one of the seven millennium prize problems.[3]

In order to study the power of efficient QTM's we need the quantum analogue of P, but before we do that we first define randomized polynomial-time.

[3]See: `http://www.claymath.org/millennium/`

Definition 5. *A is in BPP if there exists a TM M, $\epsilon > 0$, and polynomial p such that*

- $x \in A \Rightarrow \Pr[M(x, y) = 1] > \frac{1}{2} + \epsilon$
- $x \notin A \Rightarrow \Pr[M(x, y) = 0] > \frac{1}{2} + \epsilon$

where the probability is taken uniformly over the $y \in \{0,1\}^{p(|x|)}$ and the running time of M is bounded by $p(|x| + |y|)$.

The complexity class BPP could be a first contender for violating the extend Church-Turing thesis. The best known deterministic simulation of a BPP computation is to loop over all the $y \in \{0,1\}^{p(|x|)}$, simulate $M(x, y)$ and accept iff the majority of these y lead to an accepting computation.[4] There is evidence, starting from the beautiful work of Nisan and Wigderson [NW94] that efficient simulations exist, that only need to examine a polynomial number of *pseudo random* strings y. Under a fairly natural complexity-theoretical assumptions about the computational hardness of functions in exponential time, BPP can be simulated in polynomial time [AB09]. Indeed computational problems for which initially an efficient randomized algorithm was discovered get later on *derandomized.* The polynomial-time algorithm for primality testing [AKS04] being a prime example of this.

We next define, the quantum variant of BPP.

Definition 6. *A is in BQP if there exists a QTM $M, \epsilon > 0$, and a polynomial $p(n)$ such that:*

- $x \in A \Rightarrow \Pr[M(x) = 1] > \frac{1}{2} + \epsilon$
- $x \notin A \Rightarrow \Pr[M(x) = 0] > \frac{1}{2} + \epsilon$

and the running time of $M(x)$ is bounded by $p(|x|)$.

Using the same idea as in the proof of Theorem 2 one can show that if the expected running time of $M(x)$ is bounded by $p(|x|)$ then there exists another machine whose *total* running time is bounded by $p'(|x|)$ for some other polynomial p'. That is why we dropped the expectation in the running time in Definitions 5 and 6.

It is not hard to see that BPP $\subseteq$ BQP. Applying the Hadamard operation, like in Equation 2.4, to each of the qubits of the state $|0^{p(|x|)}\rangle$ gives a uniform superposition over all the strings $y \in \{0,1\}^{p(|x|)}$, next running $M(x, y)$ in superposition and measuring the state:

$$\frac{1}{\sqrt{2^{p(|x|)}}} \sum_{y \in \{0,1\}^{p(|x|)}} |y\rangle \otimes |M(x, y)\rangle \tag{27}$$

will give the same probability distribution of accepting and rejecting computations as the original BPP-machine.

Unlike for BPP, there is no evidence that BQP can be efficiently simulated. If this is not the case (BQP $\not\subseteq$ P) then the extended Church-Turing thesis is false. It turns out that it is quite hard to establish this because of the following Theorem.

[4]In fact a $\frac{1}{2} + \epsilon$ fraction of the y will give the correct answer.

Theorem 7. *[ADH97] BQP $\subseteq$ PP*

PP is defined as follows.

Definition 8. *A is in PP if there exists a TM M, polynomial p, such that:*

$$x \in A \Leftrightarrow \Pr[M(x, y) = 1] > \frac{1}{2}$$

where the probability is taken uniformly over the $y \in \{0,1\}^{p(|x|)}$ and M runs in time $p(|x|)$.

PP is the class sets recognized by probabilistic Turing machines with success probability larger than $\frac{1}{2}$. Compare this to BPP which has correctness probability $\frac{1}{2}+\epsilon$. The derandomization techniques that are amenable for BPP do not work for PP, and in fact it is believed that PP is a much more powerful class that contains NP. The complexity class PP itself is included in PSPACE, the class of languages accepted by Turing machines that use at most a polynomial amount of space.

Definition 9. *A is in PSPACE if there exists a TM M and polynomial p such that M accepts A and for every input $x, M(x)$ writes in at most $p(|x|)$ many different tape cells.*

We now have the following inclusions:

$$\mathrm{P} \subseteq \mathrm{BPP} \subseteq \mathrm{BQP} \subseteq \mathrm{PP} \subseteq \mathrm{PSPACE} \tag{28}$$

and

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PP} \tag{29}$$

From Equation 28 we see that if P=PSPACE it follows that BQP = P. Hence showing that P $\neq$ BQP entails separating P from PP and PSPACE, which would be a major advance in complexity theory.

4.3 NP and BQP

What is the precise power of BQP in this landscape of complexity classes? In particular does BQP contain NP? The answer to this question is one of the main open problems in quantum complexity theory.

First we need some more notation. The polynomial-time hierarchy is defined as the complexity classes one gets by giving NP access to an oracle in NP. A Turing machine equipped with an extra oracle tape, is called an oracle Turing-machine. It can write a string q, called the query, on this extra tape and enter the special query state. Then in one step it enters the YES or NO state, depending on whether q was in the oracle set O. It is as if it had a subroutine that computes membership in O for free.

Definition 10. Σ_1^p *= NP and* $\Sigma_k^p = NP^{\Sigma_{k-1}^p}$. *PH* $= \bigcup_k \Sigma_k^p$

It is not hard to prove that P=NP iff P=PH, and if $\Sigma_k^p = \Sigma_{k-1}^p$ then PH $=\Sigma_{k-1}^p$. It is believed that the PH is infinite.

The most compelling evidence for P$\neq$ BQP is the fact that factoring[5] is in BQP. But factoring is in NP $\cap$ co-NP. It is therefore unlikely to be NP-complete since this would imply a collapse of the PH to NP $\cap$ co-NP $\subseteq$ NP.

Before we go on lets investigate this question for BPP, because here the situation is much clearer. Not only is there reasonable evidence that P=BPP, it can also be proven that if NP=BPP then the polynomial-time hierarchy collapses to its second level. This follows from results of Adleman [Adl78] (BPP in P/poly) and Karp and Lipton [KL80]. We give a different proof below.

The following Theorem, lists a few facts about BPP and its relation to PH and NP.

Theorem 11. *The following are a few results about BPP.*

1. *BPP $\subseteq \Sigma_2^p$ [Sip83]*
2. *$BPP^{BPP} = BPP$*
3. *$NP^{BPP} \subseteq BPP^{NP}$[Zac86]*

The following is a well known fact about BPP.

Theorem 12. *if $NP \subseteq BPP$ then PH=Σ_2^p*

Proof. Assume that NP $\subseteq$ BPP. This implies that $\mathrm{NP}^{\mathrm{NP}} \subseteq \mathrm{NP}^{\mathrm{BPP}} \subseteq \mathrm{BPP}^{\mathrm{NP}}$. The first inclusion follows from the assumption and the second from Theorem 11 item 3. Using the assumption again we have that $\mathrm{BPP}^{\mathrm{NP}} \subseteq \mathrm{BPP}^{\mathrm{BPP}} = \mathrm{BPP}$. The equality is due to item 2. This way we have shown that $\Sigma_2^P = \mathrm{NP}^{\mathrm{NP}} \subseteq \mathrm{BPP}$. Employing the same chain of idea's again we get, using item 1 that $\Sigma_3^p \subseteq \mathrm{BPP} \subseteq \Sigma_2^p$ which implies PH $= \Sigma_2^p$. □

It is an interesting open problem to prove a similar consequence from the assumption NP $\subseteq$ BQP. This would follow if one could show the same properties for BQP as in Theorem 11. Unfortunately only item 2 is known to hold: $\mathrm{BQP}^{\mathrm{BQP}} \subseteq \mathrm{BQP}$. With respect to item 1, the question of whether this is false with repect to an oracle is even a challenging open question. See the papers [Aar10, Aar11, FU10] for a possible approach to showing that relative to an oracle BQP $\not\subseteq$ PH.

5 Conclusions and open problems

We have shown how the laws of quantum mechanics can be incorporated in the computing paradigm of Turing. We have seen how the resulting quantum Turing-machine makes use of

[5]Strictly speaking factoring is a function and not a set. One can however define the set factoring $=\{\langle x, i, b\rangle \mid i^{th}$ bit of the prime factoriszation of x is $b\}$, which has the same complexity as factoring and is in BQP,

superposition and interference. These quantum mechanical add-ons do not enrich the computing power, quantum Turing-machines do not go beyond the computable sets, and the Church-Turing thesis is not violated.

The quantum model does appear to be more efficient for certain computational problems, and a general simulation by classical means seems to require exponentially more time. Notably there is an efficient quantum algorithm for the factorization of numbers into their prime factors. No such efficient algorithms are known to exist for the classical setting. This fundamental result is the driving force behind the field of quantum information processing.

In order to understand the actual power of quantum computers we defined the class BQP and studied its relation to classical complexity classes like P, BPP, NP, PH, PP, and PSPACE. If $\mathrm{P} \neq \mathrm{BQP}$ then the extended Church-Turing thesis, which asks for an efficient simulation on classical Turing-machines, is violated. However establishing such a feat will imply the separation of P from PP and PSPACE, which would be a major breakthrough.

Last we have tried to establish the relationship between BQP and NP. In particular we would like to see evidence that BQP can not efficiently simulate NP, showing the computational limits of quantum Turing-machines. Such evidence is currently missing. We showed how for BPP the picture is much clearer, $\mathrm{NP} \subseteq \mathrm{BPP}$ impplies PH collapses, and tried to adapt this reasoning to BQP. Specific open problems are the following:

- Is $\mathrm{P} \neq \mathrm{BQP}$?
- Show that $\mathrm{BQP} \subseteq \mathrm{PH}$ or construct an oracle for which this is false.
- Show that $\mathrm{NP}^{\mathrm{BQP}} \subseteq \mathrm{BQP}^{\mathrm{NP}}$ or construct an oracle for which this is false.
- Does $\mathrm{NP} \subseteq \mathrm{BQP}$ imply that PH collapses?

References

[Aar10] Scott Aaronson. BQP and the polynomial hierarchy. In *STOC*, pages 141–150, 2010.

[Aar11] Scott Aaronson. A counterexample to the generalized linial-nisan conjecture. *CoRR*, abs/1110.6126, 2011.

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach.* Cambridge University Press, 2009.

[ADH97] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. Quantum computability. *SIAM J. Comput.*, 26(5):1524–1540, 1997.

[Adl78] Leonard M. Adleman. Two theorems on random polynomial time. In *FOCS*, pages 75–83, 1978.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of Mathematics*, 160(2):pp. 781–793, 2004.

[BB84] C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179. IEEE, 1984.

[BBC$^+$01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001.

[BCMdW10] Harry Buhrman, Richard Cleve, Serge Massar, and Ronald de Wolf. Nonlocality and communication complexity. *Rev. Mod. Phys.*, 82:665–698, Mar 2010.

[BdW02] Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theor. Comput. Sci.*, 288(1):21–43, 2002.

[Ben89] Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM J. Comput.*, 18(4):766–776, 1989.

[BS98] Charles H. Bennett and Peter W. Shor. Quantum information theory. *IEEE Transactions on Information Theory*, 44(6):2724–2742, 1998.

[BV97] Ethan Bernstein and Umesh V. Vazirani. Quantum complexity theory. *SIAM J. Comput.*, 26(5):1411–1473, 1997.

[Chu36] Alonzo Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 1936.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *STOC*, pages 151–158, 1971.

[DdW11] Andrew Drucker and Ronald de Wolf. Quantum proofs for classical theorems. *Theory of Computing, Graduate Surveys*, 2:1–54, 2011.

[Deu85] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proc. Roy. Soc. London Ser. A*, 1985.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proc. Roy. Soc. London Ser. A*, 493(1907):553–558, December 1992.

[EFMP11] M. D. Eisaman, J. Fan, A. Migdall, and S. V. Polyakov. Invited review article: Single-photon sources and detectors. *Review of Scientific Instruments*, 82(7):25, 2011.

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

[FU10] Bill Fefferman and Chris Umans. Pseudorandom generators and the BQP vs. PH problem. Manuscript, 2010.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.

[Gro96] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of 28th ACM STOC*, pages 212–219, 1996. quant-ph/9605043.

[KL80] Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *STOC*, pages 302–309, 1980.

[Lan61] Rolf Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191, 1961.

[Lev73] Leonid Levin. Universal search problems. *Problems of Information Transmission*, 9(3):265–266, 1973.

[Moo65] Gordon Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8), 1965.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*, volume 70. Cambridge University Press, 2000.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[Pos36] Emil Post. Finite combinatory processes-formulation 1. *The Journal of Symbolic Logic*, 1(3):103–105, September 1936.

[RGF$^+$12] Guy Ropars, Gabriel Gorre, Albert Le Floch, Jay Enoch, and Vasudevan Lakshminarayanan. A depolarizer as a possible precise sunstone for viking navigation by polarized skylight. *Proc. R. Soc. A*, 468(2139):671–684, 2012.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[Sch35] E. Schrödinger. Die gegenwärtige situation in der quantenmechanik. *Naturwissenschaften*, 23(48):807–812, 1935.

[Sho94] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science*, pages 124–134, Los Alamitos, CA, 1994. IEEE.

[Sim97] Daniel R. Simon. On the power of quantum computation. *SIAM J. Comput.*, 26(5):1474–1483, 1997.

[Sin00] Simon Singh. *The Code Book. The Science of Secrecy from Ancient Egypt to Quantum Cryptography.* HarperCollins Publishers, 2000.

[Sip83] Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335, 1983.

[Tur36] Alan Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936. Adendum 1937.

[vEB90] Peter van Emde Boas. Machine models and simulation. In *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity (A)*, pages 1–66. 1990.

[Zac86] Stathis Zachos. Probabilistic quantifiers, adversaries, and complexity classes: An overview. In *Structure in Complexity Theory Conference*, pages 383–400, 1986.