Chapter 5

COMPLEXITY

INTRODUCTION

The study of complexity questions and of complexity classes, computational and communicational, has proved to be very enlightening and important for classical computation. It has developed a firm theoretical basis for our understanding of the potentials and limitations of computational resources, models, and modes. There is reason to expect the same for the complexity investigations in quantum computation and communication.

It is of utmost importance to determine whether quantum classification of inherent computational complexity is indeed different from the classical one. Would this prove to be the case the very basic foundations of computing would be shaken.

Quantum computational complexity theory is characterized, as its classical counterpart, by a number of fundamental open problems concerning the proper inclusions of complexity classes. In order to get a better insight into these problems, and to test potential methods to solve them, the relativized quantum complexity theory is of interest and importance.

It is also of importance to find out how much quantum features can speed up computations, shorten communications and achieve efficiency in size or space.

Investigations of the potential impacts on the power of computing of the existence of slightly non-linear evolutions in quantum physics are also of interest.

LEARNING OBJECTIVES

The aim of the chapter is to learn:

1. the way universal quantum Turing machines can be constructed;
2. the basic quantum complexity classes and their properties;
3. the basic relations between classical and quantum complexity classes;
4. the basic results concerning relativized quantum complexity;
5. the basic concepts of quantum communication complexity;
6. a reduction of quantum communication protocols to quantum computation problems;
7. the potential impacts of non-linearity on the power of quantum computing.

Every man takes the limits of his own
field of vision for the limits of the world.

Arthur Schopenhauer (1788-1860)

Quantum complexity theory is surely the most basic one to develop our complexity oriented understanding of the potentials and limitations of information processing and communication.

The classical computational complexity theory has been based on feasibility-oriented strengthening of the Church–Turing thesis that itself has formed foundations for the computability theory. The former thesis, called also the "Modern Church-Turing computational thesis", says that any "reasonable" (practically realizable) model of computation can be efficiently simulated (in polynomial time and linear space), on probabilistic Turing machines. In the development of the fundamental concepts and models of classical complexity theory an abstraction of human computational customs and means is used, and they actually correspond to the view of the physical world classical physics offers. Once the idea of quantum computing was born and shown to be successful, at least theoretically so far, it became clear that one has to base the computational complexity theory on the laws of quantum physics. The same is true for the communication complexity problems, and that is why quantum communication complexity is also of interest and importance.

5.1 Universal Quantum Turing Machines

We present the basic ideas how to construct a QTM U which is universal in the following sense. Given an arbitrary DBV-QTM $\mathcal{M}$, its input string x, time bound T, and accuracy $\varepsilon > 0$, U produces a state whose distance from the state produced by $\mathcal{M}$, after T steps, is at most ε. In addition, the time of the simulation of $\mathcal{M}$ on $\mathcal{U}$ is bounded by a polynomial in T, $|x|$ and $\frac{1}{\varepsilon}$.

The design of the universal QTM $\mathcal{U}$ presented in the following is much more complex than those designs known for the classical universal TM and therefore only the basic ideas will be given here—for details see Bernstein (1997) and Bernstein and Vazirani (1997). At the same time the structure of $\mathcal{U}$ is simple. In addition, $\mathcal{U}$ uses a single quantum operation, a one-qubit rotation. The rest of $\mathcal{U}$ can be seen as a classical reversible TM.

5.1.1 Efficient implementation of unitary transformations

The key component of the universal QTM is a QTM which can realize efficiently and with required accuracy, an arbitrary finite unitary transformation, even in the case that this unitary transformation is given only by a "sufficiently good approximation". The design of such a QTM is the subject of this section.

The main problem with having a single QTM performing an arbitrary unitary transformation is that a fixed QTM maps in a single step any configuration to a superposition of a fixed number of configurations and therefore it cannot simulate in one step a step of any QTM. The way out is first to decompose, efficiently, any given unitary transformation U (or its approximation) to a "small number" of elementary unitary transformations and then to carry out these elementary transformations, in order to realize U.

Decomposition of unitary transformations

Any unitary matrix of degree d can be decomposed by a polynomial time, with respect to d, deterministic algorithm into a product of the so-called near-trivial unitary matrices. In doing that we assume that all entries of matrices are given within accuracy bound 2^{-r} for some r.

Definition 5.1.1 *A unitary matrix M of degree d is* **near-trivial** *if M satisfies one of the following conditions.*

1. *M is like a unit matrix except for one diagonal element which has the form $e^{i\theta}$, $\theta \in [0, 2\pi]$. (Such a transformation is called a* **near-trivial shift** *and denoted by (j, j, θ) if the element $e^{i\theta}$ is in the jth row.)*
2. *M is like the unit matrix except for the elements in the intersections of a jth and a kth row and column which form the matrix*

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

for some $\theta \in [0, 2\pi]$. (Such a transformation is called a **near-trivial rotation** *and it is denoted by (j, k, θ).)*

In the proof of the decomposition theorem presented below we make use of the following result.

Lemma 5.1.2 *There is a deterministic algorithm $\mathcal{A}$ such that given a d-dimensional column vector v of complex numbers, and an $\varepsilon > 0$, $\mathcal{A}$ computes near-trivial matrices $U_1, \ldots, U_{2d-1}$ such that*

$$||U_1 \ldots U_{2d-1}v - ||v||\, e_1|| \leq \varepsilon,$$

where e_1 is the unit vector, with one in the first component, and $\mathcal{A}$ runs in polynomial time with respect to d, $\lg \frac{1}{\varepsilon}$ and length of the input.

Proof. The first step is to use, for $j = 1, \ldots, d$, phase shifts $P_j = (j, j, \phi_j)$, where $\phi_j = 2\pi - \arccos \frac{Re(v_j)}{|v_j|}$ or $\phi_j = \arccos \frac{Re(v_j)}{|v_j|}$ (depending on whether $Im(v_j)$ is positive or negative), if $v_j \neq 0$ and $\phi_j = 0$, otherwise—to replace all components v_j by $|v_j|$.

As the next step $d-1$ rotations R_j, $j = d-1, \ldots, 1$, are used to move all weights of the vector into its first component—all other components will have at the end of this procedure the value 0. This can be achieved with $R_j = (j, j+1, \psi_j)$ and $\psi_j = \arccos \frac{|v_j|}{\sqrt{\sum_{j=1}^{n} |v_j|^2}}$ if the sum of the elements is not 0 and $\psi_i = 0$ otherwise. As the result we have

$$R_1 \ldots R_{d-1} P_1 \ldots P_d v = ||v|| e_1. \tag{5.1}$$

The angles ϕ_j and ψ_j can be computed in polynomial time, with respect to d, $\lg \frac{1}{\varepsilon}$ and the length of input, with the precision $\delta = \frac{\varepsilon}{(2d-1)||v||}$. Let us denote by P_j', R_j' the near-trivial matrices corresponding to P_j, R_j but with elements determined with the precision δ. Since $|\phi_j - \phi_j'| \leq \delta$ we have $||P_j - P_j'|| \leq \delta$ and since $|\psi_j - \psi_j'| \leq \delta$ we get $|R_j - R_j'| \leq \delta$, and therefore also the inequality

$$||R_1' \ldots R_{d-1}' P_1' \ldots P_d' - R_1 \ldots R_{d-1} P_1 \ldots P_d|| \leq (2d-1)\delta,$$

and consequently,

$$||R'_1 \dots R'_{d-1} P'_1 \dots P'_d v - ||v|| e_1|| \leq (2d-1)\delta ||v|| = \varepsilon.$$

(The last inequality is due to the fact that if the distance between angles is at most δ, then so it is between the corresponding points on the unit circle in both real and complex plane.) □

Theorem 5.1.3 (Decomposition Theorem) *There is a deterministic algorithm $\mathcal{A}$, which runs in polynomial time, with respect to d, $\lg \frac{1}{\varepsilon}$ and the length of input, such that given as input an $\varepsilon > 0$, and a complex matrix U which is $\frac{\varepsilon}{2(10\sqrt{d})^d}$-close to a unitary matrix, $\mathcal{A}$ computes a sequence of near-trivial matrices of degree d, $U_1, \dots, U_n$, with n polynomial in d, such that $||U - U_n \dots U_1|| \leq \varepsilon$.*

Proof. Let us say that a matrix of the degree d is k-simple if its first k rows and columns are identical with those of the unit matrix of degree d. The basic idea of the proof is to show how to reduce the problem of the decomposition of an i-simple matrix of degree d, close to a unitary matrix, to the problem of decomposition of an $(i+1)$-simple matrix of degree d, for $i = 0, 1, \dots, d-1$. (Observe that the product of two k-simple matrices is again a k-simple matrix.)

Suppose that we have started to approximate a k-simple matrix U by near-trivial matrices whose product is V. In order to produce the desirable reduction, we need to create another sequence of near-trivial matrices whose product is $W = UV^*$. In order to reduce the problem, V should be such that UV^* is close to being $(k+1)$-simple. To achieve that Lemma 5.1.2 will be used.

Let U be k-simple and δ-close to a unitary matrix. Let Z be the lower right $(d-k)\times(d-k)$ submatrix of U and Z_1 its first row. By Lemma 5.1.2, one can construct a sequence of near-trivial matrices $V_1, \dots, V_{2(d-k)-1}$ of degree $d-k$ whose product $V = V_1 \dots V_{2(d-k)-1}$ is such that $||VZ_1^T - ||Z_1|| e_1|| \leq \delta$. Finally, let us extend the matrices V and all V_i into the matrices of degree d that are k-simple.

Denote $W = UV^*$. V is unitary and W is k-simple and δ-close to a unitary matrix (because U is unitary and V is δ-close to a unitary matrix). In addition, W is close to being $(k+1)$-simple in the following sense. For the $(k+1)$th row W_{k+1} of W it holds, as discussed below,

$$||W_{k+1} - e_{k+1}|| \leq 2\delta \quad \text{and all} \quad ||W_{j,k+1}|| \leq 6\delta \quad \text{for all } j \geq k+1. \tag{5.2}$$

Let X be a $(k+1)$-simple matrix of degree d such that $X_{i,j} = W_{i,j}$ if $i, j > k+1$. From (5.2) and Exercise 1.4.21 it follows that $||W - X|| \leq 2\delta + 6\sqrt{d}\delta$. Since W is δ-close to a unitary matrix, X is $(3\delta + 6\sqrt{d}\delta)$-close to a unitary matrix.

The problem with the above idea is that it may happen that not all entries of $W = UV^*$ can be computed exactly. However, there is a way out. By Exercise 1.4.21, it is sufficient to compute them with the accuracy $\frac{\delta}{d}$ to obtain a matrix W_1 such that $||W - W_1|| \leq \delta$. Let us now use W_1 to construct a new matrix X_1 in a similar way as above. On the base of the triangle inequality we can then derive that $||W - X_1|| \leq 3\delta + 6\sqrt{d}\delta$ and that X_1 is $(4\delta + 6\sqrt{d}\delta)$-close to a unitary matrix.

The problem of decomposing a k-simple matrix U was reduced in this way to the problem of the decomposition of a $(k+1)$-simple matrix X_1 provided we are willing to accept two errors:

1. An error $||W - X_1|| \leq 3\delta + 6\sqrt{d}\delta$, because we are going to decompose in the next step X_1 and not W.

2. X_1 is only $(3\delta + 6\sqrt{d}\delta)$-close to a unitary matrix.

If $\delta' = 10\sqrt{d}\delta$, then δ' is the upper bound on both of the above errors. The total error, after d steps, is therefore $\sum_{j=1}^{d}(10\sqrt{d})^j\delta \leq 2(10\sqrt{d})^d\delta$, since $10\sqrt{d} \leq 2$. Since U is δ-close to a unitary matrix with $\delta = \frac{\varepsilon}{2(10\sqrt{d})^d}$, the total error of the approximation is, as required, at most ε.

To finish the proof, it remains to show (5.2). This requires to make several careful estimations and to use the inequality from Exercise 2.3.29b. This is left to the reader (see also Bernstein, 1997, and Bernstein and Vazirani 1997). $\Box$

Implementation of unitary transformations

The existence of a single QTM capable to implement, efficiently and with a sufficient accuracy, any unitary transformation is based on the decomposition result presented above and on the existence of a single QTM that can carry out any rotation. This is in turn based on the following lemma, which implies that a fixed rotation can be used to approximate efficiently any other rotation.

Lemma 5.1.4 *If $R = 2\pi\sum_{i=1}^{\infty} 2^{-2^i}$, then there is a deterministic polynomial time algorithm, with respect to $\lg\frac{1}{\varepsilon}$ and the length of the input, which for given $\varepsilon > 0$, and $\theta \in [0, 2\pi]$, produces an integer k, bounded by a polynomial in $\frac{1}{\varepsilon}$ such that $|kR - \theta| \bmod 2\pi \leq \varepsilon$.*

Proof. The algorithm is simple: it first computes an n, a power of 2, such that $\varepsilon > \frac{\pi}{2^{n-1}}$, and then an $m \in [1, 2^n]$ such that $|\frac{\theta}{2\pi} - \frac{m}{2^n}| \leq \frac{1}{2^n}$. The outcome is $k = m2^n$.

Indeed,

$$
\begin{aligned}
(m2^nR) \bmod 2\pi &= (2\pi m \sum_{i=\lg n+1}^{\infty} 2^{n-2^i}) \bmod 2\pi \\
&= (\frac{2\pi m}{2^n} + 2\pi m \sum_{i=\lg n+2}^{\infty} 2^{n-2^i}) \bmod 2\pi
\end{aligned}
$$

and since

$$
m \sum_{i=\lg n+2}^{\infty} 2^{n-2^i} \leq m2^{n-4n+1} \leq 2^{-2n+1},
$$

we have

$$
\begin{aligned}
|m2^nR - \theta| \bmod 2\pi &\leq |m2^nR - \frac{2\pi m}{2^n}| \bmod 2\pi + |\frac{2\pi m}{2^n} - \theta| \\
&\leq \frac{2\pi}{2^{2n-1}} + \frac{2\pi}{2^n} < \frac{2\pi}{2^{n-1}} < \varepsilon.
\end{aligned}
$$

$\Box$

The following theorem yields the key result used in Section 5.1.2 to design a universal QTM.

Theorem 5.1.5 (Unitarity Transformation Theorem) *There is a stationary and normal form QTM $\mathcal{M}$, with the first track alphabet $\{0,1,\lambda\}$, that can perform, given an $\varepsilon > 0$, T steps of any unitary transformation U_0, of the degree d, on its first track, in polynomial time, with respect to $\frac{1}{\varepsilon}, d, T$ and the length of input, provided U_0 is given by an $\frac{\varepsilon}{4(10\sqrt{d})^d}$-close matrix.*

Sketch of the proof. Given an $\varepsilon > 0$ and a transformation U of degree $d = 2^k$ which is $\frac{\varepsilon}{2(10\sqrt{d})^d}$-close to a unitary transformation, the machine $\mathcal{U}$ to carry out U with accuracy ε, on the first k cells of the first track will be designed with the help of the following QTM $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$.

1. $\mathcal{M}_1$ decomposes U into a sequence $U_1, \ldots, U_n$ of near-trivial transformations such that $||U - U_n \ldots U_1|| \leq \frac{\varepsilon}{2}$, where n is polynomial in d, and writes on an empty track $\frac{\varepsilon}{2n}$. The existence of $\mathcal{M}_1$ follows from Theorem 5.1.3.

2. $\mathcal{M}_2$ applies, one after another, transformations $U_1, \ldots, U_n$, each with accuracy $\frac{\varepsilon}{2n}$. The stationary, normal-form QTM $\mathcal{M}_2$ to perform Step 2, can be designed by composition of the following quantum Turing machines $\mathcal{T}_1$ and $\mathcal{T}_2$:

$\mathcal{T}_1$ applies any near-trivial transformation specified by a triple (i, j, θ), which is a shift if $i = j$ and a rotation if $i \neq j$. Such a QTM can be designed using the branching lemma 4.2.37 and two QTM, $\mathcal{T}_s$ for shifts and $\mathcal{T}_r$ for rotations. To design $\mathcal{T}_2$ a DTM $\mathcal{T}_d$ is used that can permute a given sequence of near-trivial transformations, each given by a triple (i, j, θ)—by moving the rightmost one to the left end of the sequence. This can be clearly done by a DTM and therefore, by synchronization theorem, we can design a QTM $\mathcal{T}'$ to perform such a permutation.

3. $\mathcal{M}_3$ erases matrices $U_1, \ldots, U_n$ and $\frac{\varepsilon}{2n}$. (This can again be done by a DTM and therefore by a stationary, normal-form QTM.)

It remains to put together QTM for Steps 1 to 3. This can be done by combining methods presented in Exercises 4.2.31 and 4.2.32 and the composition lemma.

The running time of QTM $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$ is independent of the contents of the first track (which contains the description of the configuration), and therefore so is the running time of the resulting QTM $\mathcal{M}$.

If $\mathcal{M}$ is run on the first k cells of the first track, then a unitary transformation U' is applied such that

$$||U' - U|| \leq ||U' - U_n \ldots U_1|| + ||U_n \ldots U_1 - U|| \leq n\frac{\varepsilon}{2n} + \frac{\varepsilon}{2} = \varepsilon.$$

5.1.2 Design of a universal quantum Turing machine

It is in principle easy to design a multitape or multitrack classical universal Turing machine $\mathcal{U}$ (UTM) capable of simulating an arbitrary one-tape TM $\mathcal{M}$. Given a representation of $\mathcal{M}$ in a fixed-size alphabet, $\mathcal{U}$ simulates one step of $\mathcal{M}$ after another. To simulate one step of $\mathcal{M}, \mathcal{U}$ first determines, from the current configuration c of $\mathcal{M}$, the current state and the tape symbol under the head at c. Afterwards $\mathcal{U}$ searches, in the finite description of $\mathcal{M}$, for the transition $\mathcal{M}$ should perform and implements the required change of the state, of the tape symbol, and of the position of the head. In order to achieve polynomial time efficiency, $\mathcal{U}$ keeps shifting the description of $\mathcal{M}$ so it is always close to the position of the head (on another track) of $\mathcal{M}$.

There are surprisingly small classical universal TM (see Roghozin 1996, Gruska 1997). Currently the smallest universal TM has 5 states and 5 tape symbols. There is also a universal TM with 2 states (and 18 tape symbols) and 2 tape symbols (and 24 states). The design of a universal QTM is much more complex.

It is natural to try to design a UQTM on the basis of similar ideas as the classical UTM. However, there are some fundamental difficulties that need to be overcome.

1. Unitary transformations specifying evolutions of QTM have in principle infinite dimension. A way out is, given a QTM, first to design an equivalent unidirectional QTM, evolution of which can be specified by a finite unitary matrix.

2. A natural idea that a UQTM $\mathcal{U}$ simulates the evolution of any given QTM $\mathcal{M}$ step by step is not so easy to realize. The problem lies in the following. Each evolution step of $\mathcal{M}$ is an application of its unitary matrix. If this is to be simulated by $\mathcal{U}$ then it seems that $\mathcal{U}$ has to implement one step of $\mathcal{M}$ using several steps of $\mathcal{U}$. Therefore $\mathcal{U}$ will not map a state of $\mathcal{M}$ immediately to the proper superposition, but it has to create this superposition during several steps. This causes difficulties because it is necessary that $\mathcal{U}$ is reversible. The way out has been found in Section 5.1.1. A complex unitary transformation can be decomposed into a sequence of near-trivial unitary matrices.

Encodings of QTM

Any universal QTM $\mathcal{U}$ gets as input an encoding of a QTM $\mathcal{M}$ and of its input x. Both encodings have to be words in the alphabet of $\mathcal{U}$. The number of tape symbols and states can be encoded in a similar way as for classical TM (see, for example, page 224 in Gruska (1997) or Appendix on web). The transition function $\delta_{\mathcal{M}}$ is encoded by an algorithm that for given arguments $q, \sigma, \sigma', q', d$ computes the amplitude $\delta_{\mathcal{M}}(q, \sigma, \sigma', q', d)$ with precision 2^{-n} in time polynomial in n.

Design of UQTM

Theorem 5.1.6 *There is a normal form QTM $\mathcal{U}$ such that for any QTM $\mathcal{M}$, its input x, $\varepsilon > 0$, and $T \in \mathbf{N}$, $\mathcal{U}$ can simulate $\mathcal{M}$ with accuracy ε for T steps in time polynomial in T and $\frac{1}{\varepsilon}$.*

Proof. The universal QTM $\mathcal{U}$, the existence of which we are going to demonstrate, gets as input a description of $\mathcal{M}$, x, T, and the simulation accuracy ε. $\mathcal{U}$ works in two phases.

Preprocessing. Using the algorithm of the proof of Theorem 4.2.14, $\mathcal{M}$ is transformed into an equivalent unidirectional QTM $\mathcal{M}' = \langle \Sigma, Q, \delta \rangle$. During this preprocessing the following outputs are produced:

1. A representation of the initial configuration of $\mathcal{M}'$ for x of length $2T + 1$. The reason for such length is the fact that T steps of $\mathcal{M}'$ are to be simulated. Each configuration of $\mathcal{M}'$ will be represented on the working tape of $\mathcal{U}$ with each cell of $\mathcal{M}'$ represented by a "supercell" consisting of $|\Sigma|\,|Q|$ cells—enough to store representation of a symbol from Σ and of a state from Q, if the head of $\mathcal{M}'$ is to scan a particular cell, and 0, otherwise.

2. The description of the unitary transformation $U_{\mathcal{M}'}$ for the evolution of $\mathcal{M}'$, written with the accuracy $\frac{\varepsilon}{40T(10\sqrt{d})^{d+2}}$, where $d = |\Sigma|\,|Q|$.

3. The binary string s of length $|Q|$ of the directions of the head moves of $\mathcal{M}'$, for $|Q|$ states of $\mathcal{M}'$.

4. The desired number of the simulation steps, $5T$, and the desired accuracy ε.

All that can be computed by a DTM in time polynomial in T, $\frac{1}{\varepsilon}$ and the length of the input and therefore, by the synchronization theorem, with the asymptotically same efficiency also by a QTM.

Simulation of the computation steps of $\mathcal{M}'$. Let us first describe a QTM called STEP that simulates one step of $\mathcal{M}'$. STEP operates as follows:

1. STEP transfers the current state q and the tape symbol σ to an empty work space near the starting cell, leaving a special marker in their places.

2. STEP applies $U_{\mathcal{M}'}$ to (q, σ) to produce, with accuracy ε, a superposition of new states and symbols (q', σ').

3. STEP reverses Step 1, to replace the marker with (q', σ') and to empty the working space.

4. STEP transfers the state on the tape either one cell left or right according to the value of q'th bit of the string s.

Steps 1, 3 and 4 can be done easily by DTM and therefore, by the synchronization theorem, there exists a stationary, normal form QTM that realizes the above steps that works, for a fixed $\mathcal{M}$, in time polynomial in T. The fact that Step 2 can be done by a QTM in time polynomial in T, d and $\frac{1}{\varepsilon}$ follows from Theorem 5.1.5. Moreover, using the results of Exercises 4.2.31 and 4.2.32, QTM for Steps 1 to 4 can be composed to get the resulting QTM STEP to perform simulation of one step of $\mathcal{M}'$. By inserting STEP for the special state in the RTM constructed according to the looping lemma 4.2.38, together with an additional input T, the resulting QTM simulating $\mathcal{M}'$ will compute in time polynomial with respect to T and $\frac{1}{\varepsilon}$ and with the accuracy $T\varepsilon$.

The resulting universal QTM is obtained by composing QTM of both phases.

Analysis. If, in the preprocessing, the transformation $U_{\mathcal{M}'}$ is computed to the specified accuracy, then the transformation provided for the simulation is within $\frac{\varepsilon}{40T(10\sqrt{d})^d}$ of the desired unitary transformation $U_{\mathcal{M}'}$ and therefore it will be $\frac{\varepsilon}{40T(10\sqrt{d})^d}$-close to a unitary transformation as required for the output of STEP. This implies that if each time STEP works with accuracy $\frac{\varepsilon}{40T}$, then it applies a unitary transformation which is within $\frac{\varepsilon}{20T}$ of $U_{\mathcal{M}'}$. After $5T$ runs of STEP the transformation is applied which is within $\frac{\varepsilon}{4}$ of the transformation of $\mathcal{M}'$. This implies that an observation of the simulation track of $\mathcal{M}'$, after it has completed its run, yields a sample from a distribution which is within the total variance distance ε of the distribution sampled by observing $\mathcal{M}$ on input x at time T (see Exercise 4.2.110. □

Exercise 5.1.7 *(Bernstein and Vazirani, 1997) Let $\mathcal{M}$ be a QTM such that there is a polynomial time algorithm $\mathcal{A}$ such that, for any input x and any $t > 0$, all measurements of $\mathcal{M}$ at time t are determined by $\mathcal{A}$. Show that there is a QTM $\mathcal{M}'$ such that for any $t > 0$, any input x and any $\varepsilon > 0$, a measurement of $\mathcal{M}'$ at time t, determined by a fixed algorithm polynomial time in t and $\frac{1}{\varepsilon}$, allows sampling from a probability distribution of $\mathcal{M}'$, which is within the total variation distance ε of the distribution samples from $\mathcal{M}$, with respect to the outcomes of $\mathcal{A}$ over t steps.*

5.2 Quantum Computational Complexity

To ask the hard question is simple.

Wystan Hugh Auden (1907-1973)

Quantum variations of the main time and space computational complexity classes of classical computing are other important theoretical concepts to use in order to get a deeper insight into the power of quantum computing.

For quantum time complexity classes, and for their relations to classical computational complexity classes, several inclusion results are already known and will be discussed below. Unfortunately, as is usual in complexity theory, in many cases it is not known whether inclusions obtained are proper. However, it is known that many major new open problems cannot be resolved without making a breakthrough concerning the separation of the classical computational complexity classes.

In the case of quantum space complexity classes the situation is quite different, quantum computing has not brought an asymptotical decrease in the space resources needed.

5.2.1 Basic quantum versus classical complexity classes

Definitions of the complexity classes to consider for quantum complexity theory parallel that for randomized complexity classes. The main new parameter is the way measurements are made and, consequently, probabilities are calculated, as well as what types of amplitudes are allowed.

Types of QTM. In order to define the time complexity classes, one-tape multitrack QTM are considered. To define the space complexity classes, off-line multitape QTM are considered with one-way, read-only, input tape, a working tape, and one-way, write-only, output tape. Such a model is needed to investigate sublinear complexity classes. In both cases only such QTM are considered all amplitudes of which are rational. The case of more general amplitudes is discussed separately.

Measurements and probabilities. It has been shown by Bernstein (1997), (see also Exercise 5.1.7), that in order to study the time complexity classes, it is sufficient to consider only computations in which the measurement is done only after the machine comes into the halting state. On the other hand, to study the space complexity of multitape QTM (see Watrous, 1997a), a measurement is done each time a symbol is written on the output tape. More precisely, to study time complexity classes, it is sufficient to consider only such multitrack QTM $\mathcal{M}$ where the last track alphabet is $\{\lambda, 0, 1\}$ and if $\mathcal{M}$ runs with input string x on the first track and with all other track empty, then the last track is observed each time $\mathcal{M}$ gets into the halting state, which can be determined by a measurement that has no effect on the computation, and the probability p with which we see 1 on the starting cell of the last track is the overall probability x is accepted by $\mathcal{M}$.[1]

Acceptance. Table 5.1 summarizes the definitions of the main complexity classes. Each

[1] One can define observation/acceptance also for space-bounded computations the same way as for time bounded computation. In this case the resulting space-bounded classes correspond to the "halting" classes, i.e., classes for which QTM halts absolutely

	EQ	RQ	BQ	NQ	PrQ
$x \in L$	1	$\geq \frac{1}{2} + \varepsilon$	$\geq \frac{1}{2} + \varepsilon$	> 0	$> \frac{1}{2}$
$x \notin L$	0	0	$\leq \frac{1}{2} - \varepsilon$	0	$\leq \frac{1}{2}$

Table 5.1: Definitions of the main complexity classes. The occurrence of ε in a column of the table means that there is an $\varepsilon > 0$ such that the corresponding condition for acceptance and rejection holds for all x in the alphabet of L.

of the classes is defined as the class of languages L such that for $x \in L$ and for $x \notin L$ certain conditions for the acceptance probabilities are satisfied. E stands here for "error-free computation"; R stands for "one-sided bounded-error computation" (Monte Carlo); B stands for "bounded-error computation"; N for "nondeterministic computation" and Pr for unbounded-error computation. (In the Appendix, in Section 9.3.2, this class is denoted **PP** in the case of the acceptance in polynomial time.)

Resources. If $t : \mathbf{N} \to \mathbf{N}$ is a (time-constructible[2]) function and **CQ** a complexity class, then $CQTime(t(n))$ is the class of languages that belong to **CQ** and can be accepted by a CQ-Turing machine[3] in time $t(n)$. **CQP** stands for acceptance in polynomial time.

Similarly, if $s : \mathbf{N} \to \mathbf{N}$ is a (space-constructible) function and **CQ** a complexity class, then $CQSpace(s(n))$ is the class of languages that belong to **CQ** and can be accepted by a CQ-Turing machine within $s(n)$-space.

Quantum time complexity classes

Quantum versions of the classical time complexity classes **P** and **BPP** are classes **EQP** and **BQP**. The class **BQP** can also be defined as the class of languages $L \subseteq \Sigma^*$ such that a QTM can decide, with probability at least $\frac{2}{3}$ for each $x \in \Sigma^*$, whether $x \in L$.

In quantum computing the class **BQP** is regarded as the class of languages (problems) efficiently decidable (on quantum computers).

The following basic relations between the main classical and quantum time complexity classes hold:

$$\mathbf{P} \subseteq \mathbf{EQP} \subseteq \mathbf{BQP} \qquad \mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{PSPACE}.$$

However, in order to prove these inclusions we need to show more precisely how the acceptance of languages by QTM is defined and how the probability of acceptance is determined.

We consider only such multitrack QTM $\mathcal{M}$ where the last track alphabet is $\{\lambda, 0, 1\}$ and if $\mathcal{M}$ runs with input string x on the first track and with all other tracks empty, then the last track is observed when $\mathcal{M}$ gets into the halting state (which can be determined by a measurement that has no effect on the computation) and the probability p with which we see 1 on the starting cell of the last track is the overall probability x is accepted by $\mathcal{M}$. (Here the polynomial time restriction translates as follows: there exist a polynomial p, such that the halting state is observed after at most $p(|x|)$ steps.)

[2]A function $f : \mathbf{N} \to \mathbf{N}$ is $t(n)$-time-constructible and $s(n)$-space-constructible if the function $f' : \{1\}^* \to \{0,1\}^*$, defined by $f'(1^{(n)}) = bin^{-1}(f(n))$—binary representation of $f(n)$—is computable by a $t(n)$-time-bounded and $s(n)$-space-bounded 2-tape TM. f is called time-constructible (space-constructible) if f is f-time-constructible (f-space-constructible).

[3]A CQ-Turing machine is a QTM such that the language it accepts is accepted in a way satisfying the definition of the class CQ.

Relations between P, EQP, BPP, BQP and PSPACE

Basic inclusions between the main classical and quantum complexity classes are easy to show (see Bernstein and Vazirani (1997) for the first detailed proofs adapted here).

Theorem 5.2.1 $\mathbf{P} \subseteq \mathbf{EQP} \subseteq \mathbf{BQP}$

Proof. If $L \in \mathbf{P}$, then there is a DTM $\mathcal{M}_L$ such that $\mathcal{M}_L$ produces 1 on an input x if $x \in L$ and produces 0 otherwise. By synchronization theorem, there is a stationary, normal-form, two-track QTM which reads x from the first track and produces in polynomial time as the output, on the second track, the symbol 1 if $x \in L$ and 0 otherwise. Hence $L \in \mathbf{EQP}$. The inclusion $\mathbf{EQP} \subseteq \mathbf{BQP}$ is trivially true. ∎

Theorem 5.2.2 $\mathbf{BPP} \subseteq \mathbf{BQP}$

Proof. If $L \in \mathbf{BPP}$, then there exists a polynomial $p(n)$ and a PTM $\mathcal{M}$ with output 0 or 1 which has the following properties. For an input x, $\mathcal{M}$ computes for different $y \in \{0,1\}^{p(n)}$, as the content of a tape with random bits, a multiset S_x of $2^{p(n)}$ bits where the proportion of 1s in S_x is at least $\frac{2}{3}$ if $x \in L$ and at most $\frac{1}{3}$ otherwise. $\mathcal{M}$ will be used in the QTM $\mathcal{M}'$ accepting L as follows. $\mathcal{M}'$ first generates an equally weighted superposition of all basis states $|x,y\rangle$, $y \in \{0,1\}^{p(n)}$ and then $\mathcal{M}$ is applied to each basis state. This can be done as follows.

By composing a stationary normal form RTM which produces $0^{(p(n))}$ on the second track for an input x on the first track, with a stationary, normal-form QTM from Example 4.2.28, which applies Hadamard transformation to the contents of the second track, we get a stationary QTM $\mathcal{M}_0$ which produces on the input x the superposition

$$\frac{1}{\sqrt{2^{p(n)}}} \sum_{y \in \{0,1\}^{p(n)}} |x,y\rangle.$$

By composition of $\mathcal{M}_0$ with a stationary, normal form version of $\mathcal{M}$, designed according to synchronization theorem, we get a polynomial time QTM which on the input x produces the final superposition

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

where $\mathcal{M}(x,y)$ is the output of $\mathcal{M}$ on inputs x and y. By the assumption about $\mathcal{M}$ the proportion of 1s in $\{\mathcal{M}(x,y) \,|\, y \in \{0,1\}^{p(n)}\}$ is at least $\frac{2}{3}$ if $x \in L$ and at most $\frac{1}{3}$ otherwise. L is therefore in $\mathbf{BQP}$. ∎

Theorem 5.2.3 $\mathbf{BQP} \subseteq \mathbf{PSPACE}$

Proof. Let $L \in \mathbf{BQP}$ and $\mathcal{M} = \langle \Sigma, Q, q_0, \delta \rangle$ be a QTM that accepts L in polynomial time with the probability at least $\frac{2}{3}$. By Theorem 4.2.23, any QTM $\mathcal{M}'$ which is $(\varepsilon/(24|Q|\,|\Sigma|p(n)))$-close to $\mathcal{M}$ will simulate $\mathcal{M}$ for $p(n)$ steps with accuracy ε. Therefore, if $\mathcal{M}$ is simulated with accuracy $\frac{1}{12}$ the probability of success is still at least $\frac{2}{3} - \frac{1}{12} = \frac{7}{12}$. It is therefore sufficient to use such QTM $\mathcal{M}'$, amplitudes of which are given only to $\lg(288|\Sigma|\,|Q|p(n))$ bits.

It is now easy to see that $\mathcal{M}'$ can be simulated in polynomial space by a DTM. Indeed, in order to compute the amplitude of a configuration of $\mathcal{M}'$ after $p(n)$ computational steps, it is sufficient to make a depth-first search of the configuration tree to the depth $p(n)$. Since each amplitude has at most $\lg(288|\Sigma|\,|Q|p(n))$ bits, the overall amplitude of each configuration can be computed in polynomial space. In order to determine the overall probability of acceptance, it is sufficient to compute the sum of the squared magnitudes of those configurations at time $p(n)$ that are in the halting state and have blanks in all cells of the distance at most $p(n)$ from the starting cell. This can be done again within polynomial space. ∎

The above result has been improved (see Bernstein and Vazirani, 1997, credited to L. Valiant), and by Adleman et al. (1997), as follows.

Theorem 5.2.4 $\mathbf{BQP} \subseteq \mathbf{PP} \subseteq \mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{PSPACE}$

Open problem 5.2.5 *Determine which of the inclusions* $\mathbf{P} \subseteq \mathbf{EQP} \subseteq \mathbf{BQP}$ *and* $\mathbf{BPP} \subseteq \mathbf{BQP} \subseteq \mathbf{PP} \subseteq \mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{PSPACE}$ *are proper.*

It follows from the above inclusions that one cannot expect a proof that QTM are more time efficient than the classical TM, in the sense that some strict inclusions of the classical and the corresponding quantum polynomial time complexity classes would be shown, unless there is some breakthrough in the classical complexity theory.

Impacts of amplitudes on quantum time complexity classes

The type of amplitudes allowed in QTM can have an essential impact on particular complexity classes, as shown by Adleman et al. (1997). If A is a set of amplitudes and C a complexity class, then let C_A denote the variant of $\mathbf{C}$ obtained by considering only such QTM all transition amplitudes of which are from A.

If $A = \mathbf{C}$, the set of complex numbers, then the class $\mathbf{BQP}_{\mathbf{C}}$ has uncountable cardinality[4] and $\mathbf{BQP}_{\mathbf{Q}} \subsetneq \mathbf{BQP}_{\mathbf{C}}$, where $\mathbf{Q}$ is the set of rational numbers. On the other hand $\mathbf{EQP}_{\mathbf{C}} = \mathbf{EQP}_{\mathbf{Q}}$. Moreover, if $\mathbf{A}$ is the set of algebraic numbers[5], then $\mathbf{EQP}_{\mathbf{A}} \subseteq \mathbf{PP}$.

As follows from the results of Section 5.1, when considering the class $\mathbf{BQP}$ it is sufficient to use QTM with a single quantum operation, a rotation by the angle $2\pi\sum_{i=1}^{\infty} 2^{-2^i}$. In this spirit let $\mathbf{EQP}_\theta$ denote the sets in $\mathbf{EQP}$ accepted by QTM with a single quantum operation—rotation by the angle θ. It has been shown for θ such that $\cos\theta$ is transcendental that $\mathbf{EQP}_\theta = \mathbf{P}$.

Open problem 5.2.6 *Is it true that* $\mathbf{EQP}_{\mathbf{A}} = \mathbf{BQP}_{\mathbf{A}}$*?*

Position of the class NQP

The class $\mathbf{NQP}$, a quantum analogue of $\mathbf{NP}$,[6] was introduced by Adleman et al. (1998). A series of results (see Adleman et al. 1998, Fortnow and Rogers, 1998, and Fenner et al. 1998), about the power of the classes $\mathbf{NQP}$, culminated by the result that

$$\mathbf{NQP} = \mathbf{co\text{-}C_{=}P}, \tag{5.3}$$

[4]In addition, the class $\mathbf{BQP}$ contains non-recursive sets of all Turing degrees.

[5]Algebraic numbers are roots of polynomials with integer coefficients.

[6]Formally, in a sense, $\mathbf{NQP}$ can be seen as quantum analogue of the class $\mathbf{NP}$ even if such a definition does not fully correspond to the basic idea behind the class $\mathbf{NP}$: guess and verify. A different view of $\mathbf{NQP}$, represented as a complexity class called quantum $\mathbf{NP}$ was presented by Kitaev at AQIP'99. He defines this class in terms of quantum witnesses and verifiers—see Section 9.3 for the classical version of this approach.

where $\mathbf{C_{=}P}$ is the class of decision problems that determine whether the number of accepting computable paths (of a NTM) equals that of rejecting paths.[7] This is the first result showing that a quantum complexity class can be characterized by a counting complexity class. In addition, Yamakami and Yao (1998) have shown that $\mathbf{NQP}_K = \mathbf{co\text{-}C_{=}P}$ for any K such that $\mathbf{Q} \subseteq K \subseteq \mathbf{C}$. Since the class $\mathbf{BQP_C}$ has uncountable cardinality this implies $\mathbf{BQP_C} \neq \mathbf{NQP_C}$.[8]

There is an analogy between **NP** and **NQP** also with respect to a "complete problem" characterization, see Fenner et al. (1998). Namely, a language L is in **NP** (**NQP**) if and only if there is a PTM (QTM) $\mathcal{M}$ and a polynomial p such that

$$x \in L \Leftrightarrow Pr[\mathcal{M} \text{ accepts } x \text{ in } p(|x|) \text{ steps}] \neq 0.$$

The above characterization of the class **NQP** has also brought a deeper understanding how hard are some basic problems of quantum computing. Fenner et al. (1998) have shown that the following problems are hard for polynomial time hierarchy: to determine whether a quantum computation has non-zero probability of acceptance and to determine whether a basis state will appear with no-zero amplitude in a superposition.

Quantum space complexity classes

Concerning space efficiency there is no essential difference between the classical and quantum computation.

The first result along these lines, for reversible DTM, was obtained by Bennett (1989), who showed that RTM require at most quadratic increase of space (at the cost of only slight increase in time) comparing to DTM. This result has been improved by Lange et al. (1997). They proved that RTM can simulate DTM with a constant factor increase of space, at the expense of the exponential increase in time.

The space efficiency of QTM, in comparison to PTM, was investigated by Watrous (1997). The main result is that unbounded-error space $\mathcal{O}(s(n))$-bounded QTM and PTM are equivalent in power. Since PTM can be simulated by a DTM with only a quadratic increase of space, this implies that any space $s(n)$-bounded QTM can be simulated by a DTM in space $\mathcal{O}(s^2(n))$.

Formally, the main result is stated as follows.

Theorem 5.2.7 *If a language L is accepted by an $s(n)$ space-bounded QTM with unbounded error, then L can be accepted by an $\mathcal{O}(s(n))$ space-bounded PTM with unbounded error. (In short $PrQSpace(s(n)) \subseteq PrSpace(s(n))$.)*

Sketch of the proof. The following two technical results, for the first one see Allender and Ogihara (1996), the second to Watrous (1997a), are the base of the proof.

1. If $w_{A,B}$ denotes a binary encoding of integer matrices A, B, then $L = \{w_{A,B} \mid det(A) > det(B)\} \in PrSpace(\lg n)$.
2. Let $\mathcal{M}$ be an $s(n)$-space bounded multitape QTM. Then for each input x there are integer matrices A, B of degree $2^{\mathcal{O}(s(n))}$, elements of which are integers of length $2^{\mathcal{O}(s(n))}$, such that the following properties hold:

[7] This class, introduced by Wagner (1986), contains, for example, the graph nonisomorphism problem, which is not known to be in **NP**. Equality 5.3 therefore implies that there is a QTM that can accept in polynomial time with nonzero probability a description of two graphs if and only if they are nonisomorphic.

[8] A space-bound versionb of equality 5.3 was shown by Watrous (1997).

- $det(A) > det(B)$ if and only if $\mathcal{M}$ accepts x with probability $> \frac{1}{2}$.
- There exists a DTM $\mathcal{M}$ which on input x and an integer $k \in 2^{\mathcal{O}(s(n))}$, initially written on its working tape, computes the kth bit of $w_{A,B}$ in space $\mathcal{O}(s(n))$.

Let $\mathcal{M}$ be a QTM running in space $s(n)$, and let A and B be matrices the existence of which follows from Claim 2. Since both matrices are of degree $2^{\mathcal{O}(s(n))}$ and all their elements have $2^{\mathcal{O}(s(n))}$ size, the encoding $w_{A,B}$ can be assumed to have asymptotically at most the same length.

According to Claim 1, there is a lg-space bounded PTM $\mathcal{M}_1$ accepting the string $w_{A,B}$ such that $det(A) > det(B)$ with probability $> \frac{1}{2}$. On this base we can design a PTM $\mathcal{M}_2$ which works as follows. On an input x, $\mathcal{M}_2$ simulates $\mathcal{M}_1$ and keeps recording the position of the head of $\mathcal{M}_1$ on $w_{A,B}$. (For that $\mathcal{M}_2$ needs $\mathcal{O}(s(n))$ space.) On the base of this position, using the machine $\mathcal{M}_0$ the existence of which is assumed in Claim 2, second item, $\mathcal{M}_2$ computes in $\mathcal{O}(\lg 2^{\mathcal{O}(s(n))}) = \mathcal{O}(s(n))$ space the bit of $w_{A,B}$, which $\mathcal{M}_1$ needs to inspect. Since $det(A) > det(B)$, if $\mathcal{M}$ accepts x with probability $> \frac{1}{2}$, $\mathcal{M}_2$ also accepts x with probability $> \frac{1}{2}$. □

Open problem 5.2.8 *Are QTM with algebraic amplitudes equivalent to those with rational amplitudes with respect to space efficiency?*

As it follows from Theorem 5.2.1 and results of Watrous (1998), bounded-error probabilistic computations can be simulated by QTM either in time-efficient way or in space-efficient way. An open problem is whether they can be simulated by QTM in both time and space efficient way. The existence of such simulations for a class of important graph problems was shown by Watrous (1998).

Remark 5.2.9 Quantum version QNC of the classical parallel time complexity class NC is also of interest (see, for example, Moore and Nilsson, 1998a). Due to the enormous problems decoherence causes, it is of special interest to find out what can be computed in quantum parallel polylogarithmic time. (However, from a complexity-theoretic point of view (e.g. with a uniformity condition analogous to the classical case) a fully satisfactory definition of **QNC** seems to be still a challenge.)

5.2.2 Relativized quantum complexity

The most significant demonstration of the computational power of quantum evolution was in Chapter 3 on the black-box promise problems. The idea of the black-box computations is on the level of QTM captured, similarly as in the classical case, by the concept of **oracle QTM**.

Oracle QTM

In classical computing an oracle can be seen as a special subroutine each call of which costs only one time unit (see page 382 in the Appendix, Section 9.3, for classical oracle TM). In the context of QTM, subroutine calls have to satisfy a special requirement which has no classical parallel. It is necessary that subroutine calls do not leave around any garbage, only their outcomes, because computational paths with the same result, but different garbage behind do not interfere.

The simplest case of an oracle QTM is that with a Boolean oracle $f : \{0,1\}^* \to \{0,1\}$. An oracle QTM $\mathcal{M}$ with the oracle f has a special query (oracle) track, with the alphabet $\{\lambda, 0, 1\}$, on which the machine writes its question to the oracle in the form of a string xb with $x \in \{0,1\}^*, b \in \{0,1\}$ and it has two distinguished query states: a pre-query state $q?$ and a post-query state $q!$. If $\mathcal{M}$ enters the state $q?$ and the oracle track contains at that moment a string xb, where $x \in \{0,1\}^*$, $b \in \{0,1\}$, then $\mathcal{M}$ enters, in one step, the state $q!$ and the content of the oracle tape is changed to $x \cdot (b \oplus f(x))$, where $\cdot$ is here a symbol for concatenation. In other words, the XOR operation is performed with b and $f(x)$ as arguments.

In a more general case the oracle is represented by a unitary operator U and the oracle call changes the state $|x\rangle$ into $|Ux\rangle$.

As in the classical case, one can define such complexity classes as $\mathbf{C}^A$, where $\mathbf{C}$ is a quantum complexity class and A is an oracle and also classes $\mathbf{C}^{\mathcal{A}}$, where $\mathcal{A}$ is a set of oracles.

Separation of quantum complexity classes by oracles

As in the classical setting, various separation results have been obtained between relativized quantum complexity classes. The basic separation results are due to Berthiaume and Brassard (1992, 1992a, 1992b).

Theorem 5.2.10 *There are oracles A and B such that*

$$\mathbf{P}^A \subsetneq \mathbf{EQP}^A, \qquad \mathbf{EQP}^B \not\subseteq \mathbf{NP}^B \cup \textbf{co-NP}^B.$$

Proof. We prove in detail the existence of an oracle A with the above property. The existence of oracle B can be shown in a similar way and we only show the main new trick.

The construction of A is done recursively. In doing that we use two integer-to-integer functions, $p(n)$ and $s(n)$, where

$$p(1) = 1, \qquad p(n) = 2^{p(n-1)} \text{ if } n > 1$$

and $s(n)$ is any recursive function which maps $\mathbf{N}$ into $\mathbf{N}$ and takes each value infinitely many times.

Let $\{\mathcal{M}_i\}_{i=1}^{\infty}$ be a fixed enumeration of oracle DTM with input alphabet $\Sigma = \{0,1\}$. The oracle A is defined by $A = \bigcup_{i=1}^{\infty} A_i$, where $A_1 = \emptyset$ and, for $n > 1$, $A_{n+1} = A_n \cup R_n$, with R_n is chosen as follows.

Let the machine $\mathcal{M}_{s(n)}$ be simulated on the input $1^{p(n)}$ for $2^{p(n)-1}$ steps with A_n as the oracle. If $\mathcal{M}_{s(n)}$ does not stop within $2^{p(n)-1}$ steps, or rejects the input within $2^{p(n)-1}$ steps, then $R_n = \emptyset$. Otherwise, let Q_n be the set of queries asked during the computation of $\mathcal{M}_{s(n)}$ on the input $1^{p(n)}$. Clearly, $|Q_n| \leq 2^{p(n)-1}$. In this case R_n is chosen as any subset of $\Sigma^{p(n)}$ of size $2^{p(n)-1}$ such that $R_n \cap Q_n = \emptyset$. Such a set must exist because at most half of the binary strings of length $p(n)$ can be query strings in the computation of $\mathcal{M}_{s(n)}$ on $1^{p(n)}$ for $2^{p(n)-1}$ steps.

Let now

$$S_A = \{1^n \mid A \cap \Sigma^n = \emptyset\}.$$

We show first that there is an oracle QTM with the oracle A which accepts S_A in polynomial time.

Observe that $1^n \in S_A$ if and only if there are no words of length n in A. Let $f : \{0,1\}^* \to \{0,1\}$ be a recursive function such that $f(x) = 1$ if and only if $x \in A$. For any $n \in \mathbf{N}$ let f_n be the restriction of f to inputs of length n. From the definition of A it follows that f_n is either constant or balanced for each n. Hence $1^n \in S_A$ if and only if f_n is not balancedbalanced function. For a given n this can be decided by a quantum computer in one step using the Deutsch–Jozsa algorithm. It therefore requires maximally linear time (to write down 1^n) on a quantum computer to decide whether $1^n \in S_A$.

Suppose now that S_A can be accepted by an oracle DTM M_i with oracle A in polynomial time. Let n be such that $s(n) = i$. We claim that $\mathcal{M}_i^A$ accepts the input $1^{p(n)}$ and uses at least $2^{p(n)-1}$ steps to do so.

Suppose first that $\mathcal{M}_i^A$ rejects the input $1^{p(n)}$. Then $1^{p(n)} \notin S_A$ because $\mathcal{M}_i^A$ should recognize S_A. Therefore, $A \cap \Sigma^{p(n)} \neq \emptyset$. However, this means, by the definition of oracle A, that $\mathcal{M}_i^A$ accepts $1^{p(n)}$ within $2^{p(n)-1}$ steps using oracle A_n. By the definition of A_n, the set A_n does not contain any oracle queries asked during such a computation. In addition, none of these queries can be longer than $2^{p(n)-1}$—there would be no time to write down such a query. Therefore, the sets A and A_n can differ in strings smaller than $2^{p(n)}$ only in strings from R_n that are in A, but not in A_n. Therefore the same accepting path in $\mathcal{M}_i^{A_n}$ for input $1^{p(n)}$ exists if A_n is replaced by A. Consequently, M_i^A accepts S_A—a contradiction. Hence S_A cannot be accepted by an oracle DTM with the oracle A.

Suppose now that $\mathcal{M}_i^A$ accepts $1^{p(n)}$ with less than $2^{p(n)-1}$ steps. Then $1^{p(n)} \in S_A$, since $\mathcal{M}_i^A$ is supposed to accept S_A. Hence $A \cap \Sigma^{p(n)} = \emptyset$ and therefore A and A_n are identical when restricted to strings of length $2^{p(n)}$ or less. Consequently, it does not matter whether we use as oracle A or A_n for the first $2^{p(n)-1}$ steps of $\mathcal{M}_i$. Hence $\mathcal{M}_i^{A_n}$ also accepts $1^{p(n)}$, and therefore $A \cap \Sigma^{p(n)} \neq \emptyset$ by the construction of A—a contradiction.

In a similar way we can show the existence of an oracle B and a set $Y_B \in \{1\}^*$ such that $Y_B \subseteq \mathbf{EQP}^B$, $Y_B \notin \mathbf{NP}^B$ and the set $Y_B \cup \{0^n \mid 1^n \notin Y_B\}$ is neither in $\mathbf{NP}^B$ nor in $\mathbf{co-NP}^B$, but it is in $\mathbf{EQP}^B$. □

In addition, a variety of relativized results concerning **BQP** have been obtained by Fortnow and Rogers (1998) that seem even to suggest an hypothesis that **BQP** actually contains no interesting complexity class outside **BPP**. For example, there is a relativized setting where $\mathbf{P} = \mathbf{BQP}$ and the polynomial time hierarchy (see, for example, Gruska (1997)) is finite. And there is a relativized setting in which **BQP** does not have complete sets.

Simon (1994) showed the existence of an oracle relative to which **BQP** cannot be simulated by a PTM in $2^{n/2}$ steps.

Another key question is whether $\mathbf{NP} \subseteq \mathbf{BQP}$. Bennett et al.(1997) showed that relative to a random oracle, with probability 1, the class **NP** cannot be solved on a QTM in time $o(2^{n/2})$. This bound is tight due to Grover's result in Section 3.3.[9]

As already mentioned, oracles of QTM have to fullfil special conditions in order not to have undesirable effects. An important task is therefore to determine how powerful oracles can be used. Bennett et al. (1997) showed that each BQP-machine[10] $\mathcal{M}$ can be modified into another equivalent BQP-machine whose final superposition consists almost entirely of a tape configuration containing just the input and a single bit answer. On this base they

[9] These results do not rule out the possibility that $\mathbf{NP} \subseteq \mathbf{BQP}$. (It is not even clear whether $\mathbf{BQP} \subseteq \mathbf{BPP}^{\mathbf{NP}}$; i.e. whether nondeterminism+randomness is sufficient to simulate QTM.) These results only imply that there is no "black-box approach" to solving **NP**-complete problems utilizing some uniquely quantum mechanical features of QTM (see Bennett et al. 1997).

[10] A QTM $\mathcal{M}$ is said to be a BQP-machine, if it halts in polynomial time and if it accepts (rejects) an input x, then it accepts (rejects) it with probability at least $\frac{2}{3}$.

have shown that $\mathbf{BQP}^{\mathbf{BQP}} = \mathbf{BQP}$. In addition, Fortnow and Rogers (1998) have shown that $\mathbf{PP}^{\mathbf{BQP}} = \mathbf{PP}$.

In the classical complexity one considers, in addition to various classes $\mathbf{P}$, $\mathbf{NP}$,... of decision problems, also their F-versions—classes of functions computable with similar resources. The same can be done also in quantum computing. For example, Aharonov et al. (1998) have introduced the class **FQP** of functions computable by uniform quantum circuits with polynomial size and depth. For this class they have shown that $\mathbf{FQP}^{\mathbf{FQP}} = \mathbf{FQP}$.

One-way functions and the power of quantum computing

Another important concept of classical complexity theory and also of classical cryptography is that of one-way functions.

A polynomial time computable function $f : \Sigma^* \to \Sigma^*$, for an alphabet Σ, is one-way if it is one-to-one, honest and not invertible in polynomial time. (Honesty means that there is a polynomial p such that $p(|f(x)|) > |x|$ (that is honest functions do not map long input strings into short ones).)

It is well known that one-way functions exists if and only if $\mathbf{P} \neq \mathbf{UP}$, where $\mathbf{UP}$ is the class of languages L that can be accepted by NTM which have exactly one accepting path for each $x \in L$.

The following result (due to Fortnow and Rogers, 1998), shows that the existence of one-way functions does not necessarily imply that quantum computing is more powerful than the classical one.

Theorem 5.2.11 *There is an oracle A relative to which one-way functions exist and* $\mathbf{P}^{\mathbf{A}} = \mathbf{BQP}^{\mathbf{A}}$,

Remark 5.2.12 A variety of results concerning the power of quantum computation have been presented in Chapters 3 to 5. In spite of all these insights we have to realize the fact that we are still far away from understanding well the computational power of quantum systems.

Remark 5.2.13 The study of quantum computational complexity can be of importance beyond quantum computing. Indeed, let us assume that we can prove that quantum systems can be, from the computational point of view, exponentially more powerful. In such a case, perhaps, using complexity oriented tools we could get a better insight concerning a borderline between classical and quantum physics.

5.3 Quantum Communication Complexity

There are two essentially different ways of using quantum phenomena to make communications more efficient.

In the so-called **entanglement model** only classical bits are communicated but communication is facilitated by an a priori distribution of entangled qubits among the communicating parties. It has already been demonstrated in Section 2.2 that this can bring asymptotic improvement compared to the classical communication complexity. [11]

[11] Sometimes also a modification of the above entanglement communication model is considered in which qubits are used for communication. As shown in Section 6.4.4, with n entangled pairs of particles one can send n qubits by sending $2n$ bits. Entangled model therefore needs for communication at most twice as many bits as qubits, if enough of entangled pairs is available.

In the so-called **qubit communication model**, discussed in this section, communicating parties exchange qubits. This model is quite a straightforward generalization of the model of communication in classical communication complexity.

Classical communication complexity theory (see Hromkovič, Juraj (1997), Kushilevitz and Nisan (1997), Gruska (1997)), has already been much developed. Its importance stems from the experience that it is to a large extent the complexity of the communication that is behind the complexity of parallel and distributed computing. Lower-bound results on communication complexity are often used to derive lower-bound results for computation complexity. On the other hand, the development of the quantum communication complexity theory is only just beginning and seems to be essentially more difficult.

It is not immediately clear whether qubits can reduce communication costs because one of the fundamental results of the quantum information theory, due to Holevo (1973), says that by sending n qubits one cannot convey faithfully more than n bits of information.

5.3.1 Classical and quantum communication protocols and complexity

The main concepts and definitions of quantum communication complexity theory parallel those for the classical communication complexity theory.

Classical communication protocols and complexity

The basic (two-parties) **communication problem** is to determine how many bits parties A and B, usually called Alice and Bob, have to exchange in order to compute $f(x,y)$, provided Alice knows x, Bob knows y, and both of them know the function f. It is assumed that in any communication both parties exchange bits and in doing that they follow a communication algorithm, usually called **protocol**. $C_\pi(f)$ denotes the minimal number of bits they need to communicate in order to compute f provided input data are divided between Alice and Bob according to a given partition π.

Example 5.3.1 *Let $x, y \in \{0,1\}^n$, $\pi = \{x, y\}$. If $f(x,y)$ is the parity of the string xy, then clearly $C_\pi(f) = 1$. However, if $f(x,y) = 1$ if and only if $x = y$, then it can be shown that $C_\pi(f) = n$.*

In the above setting we have considered a fixed partition of inputs. Another model of the classical communication complexity is to take least number of communication bits needed to solve the problem with respect to all **balanced** (equal size) **partitions** of inputs between Alice and Bob—notation $C_b(f)$.

Example 5.3.2 *(Addition of binary numbers.) Assume that parties A and B are to compute the sum of two n-bit numbers $x = a_n \dots a_1$, $y = b_n \dots b_1$, where n is even, and each of them knows exactly half of the input bits. Assume also that B is to compute $\frac{n}{2}$ of the least significant bits of the sum and A the rest. How many bits do they need to exchange? The answer largely depends much on how the input bits are divided between the two parties. Let us consider two possible cases.*

1. If B knows $a_{\frac{n}{2}} \dots a_1, b_{\frac{n}{2}} \dots b_1$ and A the rest of the input bits, then it is clearly enough that B sends to A the single bit, namely 0, if

$$bin(a_{\frac{n}{2}} \dots a_1) + bin(b_{\frac{n}{2}} \dots b_1) < 2^{n/2}$$

and 1 *otherwise. A can then compute the remaining bits of the sum.*

2. However, if A knows $a_n \dots a_1$ *and B knows* $b_n \dots b_1$, *then it seems to be intuitively clear that B needs to get bits* $a_{\frac{n}{2}} \dots a_1$ *and A needs to get at least bits* $b_n \dots b_{\frac{n}{2}}$ *and an additional bit carrying information whether the sum of* $n/2$ *least significant parts of both numbers is or is not larger than* $2^{n/2}$.

Another communication model is to consider the minimum of communication bits needed for all **almost balanced partitions** of inputs—each party gets at least one-third and at most two thirds of the input bits.

In the case of **randomized communications** both parties are allowed to use their private random bits to decide in each communication round which bits they communicate. In such a case, the outcome of a communication is a random distribution on outputs and it is required that the outcome is correct with the probability $1-\varepsilon$, for some $\frac{1}{2} > \varepsilon \geq 0$.

There are several ways complexity of communication is defined in the case of randomized communications and a fixed distribution of inputs. We talk about **zero-error** or **Las Vegas** communication complexity of a communication problem P, notation $C_0(P)$, if the result is always correct. **Bounded-error communication complexity** $C(P)$ is in the case all results are correct with the probability at least $\frac{2}{3}$.

Example 5.3.3 *Let* $f, g : \{0,1\}^n \to \{0,1\}$ *and let Alice get as the input f and Bob g. The following communication problems belong to the basic ones.*

- $EQ(f,g) = \bigwedge_{x\in\{0,1\}^n}(f(x) = g(x))$,
- $IP(f,g) = \bigoplus_{x\in\{0,1\}^n}(f(x) \wedge g(x))$,
- $DISJ(f,g) = \bigvee_{x\in\{0,1\}^n}(f(x) \wedge g(x))$,

where these functionals represent equality, inner product and disjointness[12] *of two functions, respectively.*

The following results hold; see references in Gruska (1997) and Buhrman et al. (1998).

$$C_0(\mathrm{EQ}) = 2^n, C(\mathrm{EQ}) = \mathcal{O}(n), C(\mathrm{IP}) = \Omega(2^n), C(\mathrm{DISJ}) = \Omega(2^n).$$

Quantum communication protocols and complexity

Communication protocols are defined in the quantum case in a similar way to the classical case, the only difference being that complexity of communication is considered by the number of qubits exchanged. Similarly with randomized classical communication protocols, the result of quantum communications are random distributions on outputs.

A special feature of quantum communication is that due to the fact that parties cannot make copies of their qubits they have sometimes sent them forth and back.

Several quantum communication complexity measures are considered as analogous to the classical ones. For a communication problem P, let $QC_\varepsilon(P)$ denote the minimal number of qubits that need to be exchanged if the outcomes are to be correct with probability $1-\varepsilon$. $QC_0(P)$ then stands for Las Vegas, error-free, communication complexity and $QC(P) = QC_{1/3}(P)$ for bounded-error quantum communication complexity.

[12] As defined above DISJ(f,g) stands for the complement of the set of disjointness problems. However, for the model considered here the communication complexity of DISJ problem and its complement are equal (Buhrman et al. 1998).

5.3.2 Quantum communication versus computation complexity

A simple technique, presented below, due to Buhrman et al. (1998), allows the transfer of quantum algorithms for black-box computations to quantum communication protocols. This technique can be used in two ways.

1. To derive upper bounds for quantum communication complexity from the upper bounds for quantum computation complexity.

2. To derive lower bounds for quantum computational complexity from the lower bounds for quantum communication complexity .

Let $\mathcal{F}_n$ denote the set of Boolean functions $f : \{0,1\}^n \to \{0,1\}$.

Theorem 5.3.4 *Let $F : \mathcal{F}_n \to \{0,1\}$ and $L : \{0,1\} \times \{0,1\} \to \{0,1\}$. L induces a mapping $\mathcal{L} : \mathcal{F}_n \times \mathcal{F}_n \to \mathcal{F}_n$ such that $\mathcal{L}(g,h)(x) = L(g(x),h(x))$ for all $x \in \{0,1\}^n$. If there is a quantum algorithm $\mathcal{A}$ to compute $F(f)$ with t calls of f, then there is a $t(2n+4)$-qubit quantum communication protocol $\mathcal{P}$ for the following problem: Alice gets g, Bob gets h and the aim is for Alice to determine $F(\mathcal{L}(g,h))$ by communication with Bob. In addition, the probability that communication according to $\mathcal{P}$ produces the correct result is the same as for the algorithm $\mathcal{A}$.*

Proof. Let $\mathcal{A}$ be a quantum algorithm that computes $F(f)$ with t calls of f. In the communication protocol Alice makes use of the algorithm $\mathcal{A}$ with $f = \mathcal{L}(g,h)$. She needs to communicate with Bob only when an $\mathcal{L}(g,h)$-call is to be made because she does not know h. To get the result of an $\mathcal{L}(g,h)$-call Alice initializes the following communication protocol with Bob, where $x \in \{0,1\}^n$, $y \in \{0,1\}$, is the current argument for which $\mathcal{L}(g,h)$ is to be computed.

1. Alice performs the unitary operation $|x,y,0\rangle \to |x,y,g(x)\rangle$ and sends $n+2$ qubits to Bob;

2. Bob performs the unitary operation $|x,y,g(x)\rangle \to |x, L(g(x),h(x)) \oplus y, g(x)\rangle$ and sends $n+2$-qubits to Alice;

3. Alice uncomputes $g(x)$ to produce $|x, L(g(x),h(x)) \oplus y, 0\rangle$.

Since there are t calls for which such a communication is needed, the total amount of exchanged qubits is $t(2n+4)$. ∎

Exercise 5.3.5 *(a) Does the protocol in the proof of Theorem 5.3.4 make use of entanglement? If yes, then where? If not, could we do the whole protocol classically? (b) Does Bob need to send Alice $n+2$ qubits? Is it not sufficient for him to send back only one qubit with $L(g(x),h(x)) \oplus y$?.*

Example 5.3.6 *The upper bound $QC(DISJ) = \mathcal{O}(n\sqrt{2^n})$ follows from Theorem 5.3.4, by taking L to be binary AND function and F to be 2^n-ary OR function, because for computation of 2^n-ary OR-function we have the upper bound $\sqrt{2^n}$, see Section 3.3.*

Example 5.3.7 *An exponential gap between the exact classical and quantum communication complexity has been shown by Buhrman et al. (1998) for the following problem.*

Let $f, g : \{0,1\}^n \to \{0,1\}$ *and let* $\Delta(f,g)$ *be the Hamming distance between* f *and* g *which equals the Hamming distance of* 2^n*-bit strings* $f(0)f(1)\dots f(2^n-1)$ *and* $g(0)g(1)...g(2^n-1)$. *Let EQ' be the partial function defined by*

$$EQ'(f,g) = \begin{cases} 1, & \text{if } \Delta(f,g) = 0, \\ 0, & \text{if } \Delta(f,g) = 2^{n-1}; \end{cases}$$

and undefined for other arguments. (In the case of partial functions we require that communication yields the correct outcome only for arguments at which the partial function is defined.)

The upper bound $QC_0(EQ') = \mathcal{O}(n)$ *is the consequence of Theorem 5.3.4. Indeed, take* L *to be XOR function and* F *to be the* 2^n*-ary OR function restricted to balanced or zero functions. The upper bound now follows from the analysis of the Deutsch–Jozsa algorithm in Section 3.1. It has been shown, by Buhrman et al. (1998), that* $C_0(EQ') = \Omega(2^n)$.

Example 5.3.7 shows a communication task for which the number of bits needed to communicate in order to compute a given function with zero-error is exponentially larger then the number of qubits that need to be communicated. However, for this task there is a classical randomized communication protocol that achieves the same result with small error and requires to communicate the same number of bits as the number of qubits needed for the best quantum communication protocol.

The first fully exponential gap between classical bounded-error randomized communication and quantum communication has been shown by Ambainis et al. (1998a) for the following sampling task. Alice has a subset $A \subseteq \{1, 2, \dots, n\} = S$ of cardinality k and Bob task is to pick up another subset $B \subseteq S$ of cardinality k disjoint with A (if possible), The result was obtained as a byproduct of a method to deal with the following important communication primitive.

Definition 5.3.8 *(Sampling) Let* $f : X \times Y \to \{0,1\}$ *and let* D *be any probability distribution on* $X \times Y$. *A communication protocol* $\mathcal{P}$ *is said to sample* f *according to* $\mathcal{D}$ *with error* $\varepsilon > 0$, *if the distribution the protocol induces on* $\{(x,y,z)\}$ *is* ε*-close, in the total variation distance, to the distribution* $(\mathcal{D}, f(\mathcal{D}))$ *obtained by picking first* (x,y) *according to* $\mathcal{D}$ *and then computing* $z = f(x,y)$.

For the disjointness problem discussed above nad $k = \Theta(\sqrt{n})$ they give a quantum protocol in which Alice sends $\mathcal{O}(\lg n \lg \frac{1}{\varepsilon})$ qubits to Bob and they allow him to sample from a distribution ε-close to the desired uniform distribution on subsets of S disjoint to A. In addition, they show that each classical randomized protocol needs $\Omega(\sqrt{n})$ bits to be exchanged between Alice and Bob.

5.4 Computational Power of quantum non-linear mechanics

It is one of the basic axioms of quantum mechanics that quantum evolution is fully described by the linear Schrödinger equation. For many classical phenomena linear models are just a first approximation of the reality, and when a deeper insight has to be obtained nonlinear models have to be considered. It has therefore been natural to expect that the same could apply to quantum phenomena. However, so far all quantum experiments confirm linearity. In spite of that some believe that small nonlinearities in quantum evolution have not yet been completely ruled out.

There are various models of quantum nonlinear mechanics and it is of interest to explore how much computational power their validity would bring. Some of the nonlinear models have turned out to be "too weird"—within them effects are possible that are commonly considered as "unphysical". For example, faster than light effects or even an arbitrarily fast influence between non-interacting systems. However, there are also nonlinear models that are claimed to be free of such effects.

Abrams and Lloyd (1998) described two ways how to use nonlinearity of quantum mechanics to solve **NP**-complete problems in polynomial time. We present one of them and we show that there exists a polynomial time algorithm, which uses some nonlinear quantum operations, the existence of which is fully within a model of nonlinear quantum mechanics, to decide, given a function $f : \{0,1\}^n \to \{0,1\}$ whether there is an x such that $f(x) = 1$.

The algorithm makes use of the following nonlinear operator on two qubits .

$$|00\rangle + |11\rangle \to |01\rangle + |11\rangle, \quad |01\rangle + |10\rangle \to |01\rangle + |11\rangle, \quad |00\rangle + |10\rangle \to |00\rangle + |10\rangle \tag{5.4}$$

In order to simplify presentation we assume that there is at most one i such that $f(i) = 1$.

Algorithm 5.4.1 *Given* $f : \{0, \dots, 2^n - 1\} \to \{0,1\}$

1. In the superposition
$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i, f(i)\rangle$$
group all basis states $|i, f(i)\rangle$ into 2^{n-1} pairs depending on the values of the qubits number 2 to n. This way we create 2^{n-1} subspaces of dimension 4 for the first and $(n+1)$st qubit. For each subgroup the first and $(n+1)$st qubits will be in one of the following three states
$$|00\rangle + |11\rangle, \qquad |01\rangle + |10\rangle, \qquad |00\rangle + |10\rangle \tag{5.5}$$
(Observe that at most one of the 2^{n-1} superpositions will not be in the third state.) Apply now to these two qubits the nonlinear transformation (5.4).

2. Apply the previous construction $n - 1$ times in such a way that at the ith repetition the ith and $(n+1)$th qubit are chosen to form the 4-dimensional space. These two qubits will always be, for any combination of values of other qubits, in one of the states (5.5).

   The key point now is that at each such iteration, and at each application of the operator (5.4), the number of components in the superposition which have the value 1 in the last qubit doubles. Therefore, after n iterations the last qubit is no longer entangled with the others and it is in the same state for all components of the superposition.

3. Measure the last qubit.

Nonlinear transformation (5.4) used in the algorithm seems to be artificial, chosen just to get the desirable result. However, Abrams and Lloyd (1998) demonstrated that virtually any deterministic nonlinear quantum theory will include such a nonlinear operator. In addition, they have shown that virtually any deterministic nonlinear operation can be recast into this form and the operation (5.4) can be constructed from an arbitrary unitary operation and a simple nonlinear operation $|0\rangle \rightarrow |0\rangle$ and $|1\rangle \rightarrow |0\rangle$.

Abrams and Lloyd (1998), developed their algorithms within Weinberg's (1989) model of nonlinear dynamics and they used operators that do not preserve scalar product of states. In such a case Weinberg's model can exhibit unphysical effects. By Czachor (1998) slight modifications of both algorithms work also in the model that is known to be free of unphysical influences.

Using quite a different model of quantum computing, Černý (1993) has shown how to solve one **NP**-complete problem, namely the traveling salesman problem, in quantum polynomial time, but using an exponentially large amount of energy.