On quantum computation theory
van Dam, W.K.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Chapter 2
Quantum Information and Computation

In the previous chapter we described the foundations of quantum information and the quantum mechanical transformations that are possible with it. The central idea of computational complexity theory is to assign different 'costs' to different operations. Typically, a fixed set of elementary operations is used to construct all other transformations. The computational cost is then expressed as the minimal number of elementary operations that is necessary to establish the desired transformation.

2.1 Some Elementary Operations

In quantum computing and communication we look at the possibilities of transforming information as is allowed by the laws of quantum mechanics. We usually decompose such quantum algorithms in a series of small elementary steps that consist of one and two qubit operations. The following elementary unitary gates will be used throughout the rest of the thesis.

Definition 5 (Some elementary quantum gates) The Not gate: This is the gate that we know in classical computation with the additional characteristic that it respects the superposition of a qubit:

\[ \text{Not}(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle. \]

Phase Flip: The Flip gate changes the phase of a qubit conditional on its value:

\[ \text{Flip}(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - \beta|1\rangle. \]

Phase Rotation: A more general phase rotation is provided by the Phase operation, which has a free parameter \( \phi \) that determines the angle of the phase change:

\[ \text{Phase}_\phi(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + e^{i\phi}\beta|1\rangle. \]

(Note: Flip = Phase\( _\pi \).)
**Hadamard transform:** This transformation \( H \) maps the zero and one state to the following superpositions of the two basis states:

\[
H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \text{and} \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).
\]

The Hadamard is its own inverse \((H^2 = I)\).

**General Rotation:** The general rotation \( R \) with angles \( \alpha, \theta, \phi \) is the unitary one qubit transformation with eigenvectors 

\[
|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \exp(i\phi) \sin\left(\frac{\theta}{2}\right)|1\rangle \quad \text{and} \quad |\psi^\perp\rangle = \sin\left(\frac{\theta}{2}\right)|0\rangle - \exp(i\phi) \cos\left(\frac{\theta}{2}\right)|1\rangle.
\]

The corresponding eigenvalues are indicated by the equalities

\[
R_{\alpha,\theta,\phi}|\psi\rangle = |\psi\rangle \quad \text{and} \quad R_{\alpha,\theta,\phi}|\psi^\perp\rangle = \exp(i\alpha)|\psi^\perp\rangle,
\]

and are therefore 1 and \( \exp(i\alpha) \).

**Controlled-Not:** The controlled-not is a two-qubit operation that applies the Not gate to the target bit if the control bit equals “1”; otherwise it leaves the target unchanged:

\[
\text{CNot}|x, y\rangle = |x, y \oplus x\rangle,
\]

for all \( x, y \in \{0, 1\} \).

**Controlled-Flip:** The controlled-flip is, like the CNot, a two-qubit operation. It applies the Flip gate if both bits equals “1”; otherwise it leaves the state unchanged:

\[
\text{CFlip}|x, y\rangle = (-1)^{xy}|x, y\rangle,
\]

for all \( x, y \in \{0, 1\} \).

### 2.2 Fault Tolerant and Universal Quantum Gates

It has been shown that there exists finite sets of quantum gates that are universal in the following sense. Consider the networks that can be constructed from a countable set of gates \( \{G_1, G_2, \ldots \} \). Each network will implement a unitary transformation, and we want to consider if any finite-dimensional unitary transformation can be implemented in such a way. Clearly, because the set of networks is countable, we cannot hope that we can construct every element of \( U(n) \) exactly. Hence, we will have to aim for the approximation (within an arbitrary small error) of every such element. It has been proven that there are indeed universal sets of quantum gates with which this can be achieved, and these sets can be remarkable simple. The following collection was described in [26] and has the additional useful feature that the gates are ‘fault-tolerant’ [74].
2.3. Quantum versus Classical Query Complexity

Fact 1 (Universal, Fault Tolerant Sets of Quantum Gates [26]) With the Hadamard gate $H$, the controlled-not CNot and the $\frac{\pi}{4}$ phase gate $R_{\pi/4,0,0}$ any other unitary transformation can be approximated within an arbitrary small error (with respect to some distance measure on the set of operators). Also, these three gates can be implemented in a fault-tolerant way.

2.3 Quantum versus Classical Query Complexity

The theory of quantum computation investigates if, and if so, how, we can use quantum mechanical effects to solve computational problems more efficiently than we can do by classical means. So far, the strongest indication that there maybe such a difference in computational power between quantum and classical computing is provided by Peter Shor's factoring algorithm[91]. Unfortunately, the result by Shor does not prove that there is a superpolynomial separation between the two models of computation. This is because the classical time complexity of factoring and discrete logarithms is still unknown, despite more than two thousand years of effort, starting with Eratosthenes's sieve in 300 B.C.

A complexity measure for which we do have rigorous results is provided by the the black-box, or oracle, model of computation. The algorithms of Deutsch [38], Deutsch & Jozsa [39], Berthiaume & Brassard [23], Bernstein & Vazirani [22], Simon [92], Grover [48], and Buhrman & van Dam [28] give examples of problems for which we have a quantum reduction in the query complexity of a problem, whereas the lower bounds of Jozsa [59], Bennett et al. [19], and Beals et al. [10] show that there are also limits to the advantage that quantum computation can give us. The general picture that has emerged from these results is that we can only expect a superpolynomial difference between classical and quantum computation if we can use the specific structure of the problem that we try to solve. The promise on the function of Simon's problem is a typical example of such a structure that establishes an exponential quantum improvement over the classical complexity.[92] To find more structured problems that allow such a gain is one of the quests for researchers in quantum complexity theory.

Consider a problem that is defined in terms of $n$ (unknown) values $f(1), \ldots, f(n)$. The (probabilistic) query complexity of such a problem is the minimum number of times that an algorithm has to 'consult' the string $f(1), \ldots, f(n)$ to solve the problem (with high probability). A typical example of this setting is the calculation of the OR of $n$ bit values: the question whether there is an index $i$ with $f(i) = 1$. The classical query complexity of this task is $n$, whereas in the quantum setting we only need $O(\sqrt{n})$ calls to $f$ to solve the problem. We therefore say that we have a 'quadratic' separation between the classical and the quantum query complexity of the OR function. The question is which tasks allow a quantum reduction in the query complexity, and if so, how much.

The reason why quantum algorithms sometimes require less queries starts with the superposition principle of quantum mechanics. A single call "$i$" to the function $f$
establishes the evolution \( |i\rangle|b\rangle \rightarrow |i\rangle|f(i) \oplus b\rangle \) (where \( \oplus \) denotes addition modulo two), which in classical computation is the best we can expect from an \( f \)-query. But by the rules of quantum mechanics, we can also consult \( f \) in superposition. Hence, with a single call we can create a state that depends on several values \( f(i) \):

\[
\sum_i |i\rangle \otimes (\alpha_i|0\rangle + \beta_i|1\rangle) \quad \rightarrow \quad \text{one } f\text{-query} \quad \sum_i |i\rangle \otimes (\alpha_i f(i) + \beta_i (f(i) \oplus 1)).
\]

It is this 'parallelism' in combination with the quantum mechanical phenomenon of interference that allows us to solve some problems more efficiently than is possible with classical protocols.

### 2.4 Earlier Results in Quantum Computing

This thesis uses, and builds on, a combination of earlier results in quantum computation. We are especially concerned with the query complexity of procedures that prepare a state that depends on a black-box function. For example, how often do we have to read out the bit values \( f(i) \) if we want to create the state \( \sum_i (-1)^{f(i)} \alpha_i |i\rangle \)? The following fact shows us that this can be done with the minimum of a single query.

**Fact 2 (Phase-kick-back trick [31])** If we can query the function \( f \) in quantum mechanical fashion as follows:

\[
|i\rangle \otimes |b\rangle \quad \rightarrow \quad |i\rangle \otimes |b \oplus f(i)\rangle
\]

with \( f(i), b \in \{0, 1\} \), then the phase changing transition

\[
\sum_i \alpha_i |i\rangle \quad \rightarrow \quad \sum_i (-1)^{f(i)} \alpha_i |i\rangle
\]

can be established with only one call to the unknown bit values of \( f \).

**Proof:** First, we append to the superposition of \( |i\rangle \) states the qubit \( \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \). Then, in superposition, we add (modulo two) the function value \( f(i) \) to this bit. For a specific value of \( i \), this yields the evolution

\[
|i\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \quad \rightarrow \quad |i\rangle \otimes \frac{1}{\sqrt{2}}(|0 \oplus f(i)\rangle - |1 \oplus f(i)\rangle)
\]

(2.1)

\[
= \begin{cases} 
  +|i\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) & \text{if } f(i) = 0 \\
  -|i\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) & \text{if } f(i) = 1 
\end{cases}
\]

(2.2)

Hence, by the superposition principle, this gives the desired evolution with only one query to the function \( f \).

Using this fact, we can easily prove the following core result.
2.5. More Classic Quantum Results

**Fact 3 (Single Query Parity Trick)**[31, 38] Let \( f : \{0, 1\} \rightarrow \{0, 1\} \). There exists a deterministic quantum algorithm that computes the parity bit \( f(0) \oplus f(1) \) with one query to the function \( f \). This algorithm works in constant time.

**Proof:** Construct the following initial state:
\[
|\text{Initial} \rangle = \frac{1}{\sqrt{2}}(|0 \rangle + |1 \rangle) \otimes (|0 \rangle - |1 \rangle).
\]
(2.3)

Next, we add (modulo two) the bit values \( f(i) \) to the rightmost bit, where the index \( i \in \{0, 1\} \) is described by the first bit of the initial state. Note that by the superposition of this rightmost bit, both values \( f(0) \) and \( f(1) \) are also queried in superposition. Applying \( f \) in such a way establishes the following evolution on the two qubits:
\[
|i\rangle \otimes |b\rangle \quad \longrightarrow \quad |i\rangle \otimes |b \oplus f(i)\rangle,
\]
for \( b \in \{0, 1\} \). This results in the following outcome when applied to the initial state mentioned in the beginning of the proof:
\[
\frac{1}{\sqrt{2}}(|0 \rangle + |1 \rangle) \otimes (|0 \rangle - |1 \rangle) \quad \text{if} \quad f(0) = f(1) = 0
\]
\[
\frac{1}{\sqrt{2}}(|0 \rangle - |1 \rangle) \otimes (|0 \rangle - |1 \rangle) \quad \text{if} \quad f(0) = 0, f(1) = 1
\]
\[
-\frac{1}{\sqrt{2}}(|0 \rangle - |1 \rangle) \otimes (|0 \rangle - |1 \rangle) \quad \text{if} \quad f(0) = 1, f(1) = 0
\]
\[
-\frac{1}{\sqrt{2}}(|0 \rangle + |1 \rangle) \otimes (|0 \rangle - |1 \rangle) \quad \text{if} \quad f(0) = f(1) = 1.
\]

Hence, if we apply a Hadamard transformation to the first register, we obtain
\[
|\text{Final} \rangle = (-1)^{f(0)}|f(0) \oplus f(1)) \otimes (|0 \rangle - |1 \rangle).
\]

Observing the first bit of this final state yields the correct answer \( f(0) \oplus f(1) \) without error. \(\Box\)

### 2.5 More Classic Quantum Results

In 1993 Bernstein & Vazirani gave the following example of a family of functions that are more easily distinguished with quantum queries than with classical ones.

**Fact 4 (Bernstein & Vazirani's inner-product problem)**[22, 31] Let the black-box function \( g_s : \{0, 1\}^n \rightarrow \{0, 1\} \) be defined by
\[
g_s(x) = (x, s) = \sum_{i=1}^{n} s_i x_i \mod 2,
\]
(2.4)

where \( s = s_1 \ldots s_n \in \{0, 1\}^n \) is an unknown \( n \)-bit mask. A quantum computer can determine the value \( s \) with one call to the function \( g_s \), whereas any probabilistic, classical algorithm needs at least \( n \) queries to \( g_s \) to perform the same task.
Chapter 2. Quantum Information and Computation

Proof: (See [22] for the original proof, and [31] for the single query version of it.) First, initialize the \((n + 1)\)-qubit register

\[
|\text{start}\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).
\]

By XOR-ing the rightmost bit with the function value \(g_s(x)\) (cf. Fact 2), we obtain the state

\[
\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} (-1)^{(x,s)} |x\rangle \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle),
\]

with only one \(g_s\)-call. The bit string \(s\) is then easily obtained with an \(n\)-fold Hadamard transform on the first \(n\) bits:

\[
\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} (-1)^{(x,s)} |x\rangle \quad \overset{H^\otimes n}{\longrightarrow} \quad |s\rangle,
\]

which concludes the quantum algorithm.

For the classical lower bound we observe that every traditional query will only give (maximally) one bit of information about the \(n\) bits of \(s\).

The above result uses the unitarity of \(H^\otimes n\) and its connection with the inner-product function. In Chapter 5 we will derive a similar result for a different family of unitary matrices and the Legendre function that it uses.

Because the Hadamard is its own inverse we have, in fact, the following 'bi-directional statement' about this transform

\[
\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} (-1)^{(x,s)} |x\rangle \quad \overset{H^\otimes n}{\leftrightarrow} \quad |s_1s_2\cdots s_n\rangle.
\]

The above leads to the observation that if we want to know the string \(s_1\cdots s_n\), it is sufficient to have a superposition with phase values of the form \((-1)^{(x,s)}\), for every \(x \in \{0,1\}^n\). This is a well-known result in quantum computation and has been used several times to underline the differences between quantum and classical information processing.[22, 31, 49, 97]

Another key result in quantum computation is the square-root speed-up that one can obtain when querying a database for a specific element.

Fact 5 (Grover's search algorithm [48]) Let \(f(1), \ldots, f(n)\) be a string of \(n - 1\) zeros and one entry \(f(s) = 1\). With a quantum computer the unknown value \(s\) can be determined exactly with only \(\lceil \frac{n}{4} \sqrt{n} \rceil\) queries to the function \(f\).

Proof: See the original article by Lov Grover[48], or better yet, the excellent analysis of it by Boyer et al.[25]
2.6 Notation

We use $x, y, \ldots$ to denote finite, classical Boolean strings. When we write $|x\rangle$, we mean the quantum state vector in the standard basis that corresponds to the classical string $x$. In general we use $\phi, \psi, \ldots$ to denote pure quantum states. Mixed states are represented by the letters $\rho, \sigma$ et cetera. We also use uppercase letters $X, Y, \ldots$ for (mixed) quantum states that are strings of qubits. The terms quantum state, qubit string, and quantum register are used interchangeably (sometimes to emphasize the purpose of the quantum state at hand). Lower-case letters $i, j, k, l, m, n$ denote integer indices or string lengths.

For classical strings over the alphabet $\{0, 1\}$, $\ell(x)$ denotes the length of the string. For finite sets $A$, $|A|$ denotes the cardinality of the set. Concatenation of $x, y$ is written as the juxtaposition $xy$, and the $n$-fold concatenation of $x$ is written $x^n$.

For Hilbert spaces, we write $\mathcal{H}_d$ for the $d$-dimensional Hilbert space and $\mathcal{H}^m$ for the $m$-fold tensor product space $\mathcal{H} \otimes \cdots \otimes \mathcal{H}$. A pure quantum state $\phi$ represented as a vector in such a Hilbert space is denoted by the ket $|\phi\rangle$.

We slightly abuse notation by sometimes letting the state symbols $\phi, \rho, \ldots$ also stand for the corresponding density matrices. Hence, a pure state $\phi$ as a Hilbert space vector is denoted by $|\phi\rangle$, whereas its density matrix $|\phi\rangle\langle\phi|$ can also be indicated by $\phi$.

An ensemble $\mathcal{E}$ is a specific distribution $p_1, p_2, \ldots$ over a set of (mixed) states $\rho_1, \rho_2, \ldots$. We denote this by $\mathcal{E} = \{(\rho_i, p_i)\}$. The average state of such an ensemble $\mathcal{E}$ is $\rho = \sum_i p_i \rho_i$. An average state corresponds to several different ensembles. When an ensemble is used to produce a sequence of states $\rho_i$ according to the probabilities $p_i$, we speak of a source $\mathcal{E}$.

The length of a quantum state is denoted by $\ell(X)$, by which we mean the smallest $\ell$ for which $X$ sits in the $2^\ell$-dimensional Hilbert space (in the standard basis).

A transformation $S$ on the space of density matrices is allowed by the laws of quantum mechanics if and if only it is a completely positive, trace preserving mapping.

Throughout this thesis, results that were already known are indicated as 'facts'.