Der Quantencomputer:

Inhaltsverzeichnis

1 Der Quantencomputer **5**

1.1 Vergleich klassischer Computer und Quantencomputer 5

1.2 Reversible Operationen und Gatter 6

1.3 Quanteninformationsverarbeitung 9

1.4 Quantenfehlerkorrektur 11

2 Quantenalgorithmen **13**

2.1 Der Deutsch–Algorithmus 13

3 Experimentelle Realisierung **17**

4 Der Weg zum Quantencomputer **23**

5 Quantenmechanik und Computer **27**

6 Der Shor–Algorithmus **31**

6.1 Der klassische Shor–Algorithmus 32

6.2 Die diskrete Fouriertransformation 34

6.3 Der Quantenalgorithmus von Shor 36

6.3.1 Theoretische Betrachtung des Quanten–Shor–Algorithmus 36

6.3.2 Zahlenbeispiele für den Quanten–Shor–Algorithmus 41

A Zahlentheoretische Grundlagen **49**

A.1 Kongruenzen und Restklassen 49

A.2 Euklidischer Algorithmus 50

A.3 Chinesischer Restesatz 51

A.4 Eulersche φ-Funktion, Ordnung $\mathrm{mod}\ N$, großer Primzahlsatz 51

A.4.1 Die Eulersche φ-Funktion 51

A.4.2 Der große Primzahlsatz 52
A.5 Kettenbrüche 52

B Zahlenbeispiel für DFT_q 55

C Programm zur Berechnung der Periodizität und der Wahrscheinlichkeitsverteilung 59

Kapitel 1

Der Quantencomputer

1.1 Vergleich klassischer Computer und Quantencomputer

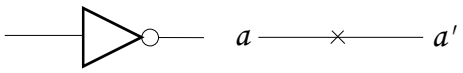
Ein universeller klassischer Rechner besteht aus einem Register, in dem beliebig (aber endlich) viele Bits gespeichert sind. Ein Bit an Information entspricht einer Entscheidung zwischen zwei Möglichkeiten wie ja oder nein, 1 oder 0, wahr oder falsch. In einem Digitalrechner ist ein Bit an Information beispielsweise die Ladung eines Kondensators: Ein geladener Kondensator bezeichnet eine 1, ein entladener Kondensator eine 0. Ein weiteres Merkmal des klassischen Computers ist das Programm, das aus einer frei wählbaren und teilweise auch wiederholten Abfolge logischer Verknüpfungen (auch Gatter genannt) besteht, die auf die Bits des Registers wirken. Üblicherweise wählt man als elementare Operationen die aus der mathematischen Logik bekannten Verknüpfungen AND, OR und NOT. Die NOT-Verknüpfung ist eine einwertige Grundfunktion, d.h. dass sie auf ein Eingangsbit wirkt. Das Eingangsbit wird invertiert: ist es 1, dann wird das Ausgangsbit 0 und umgekehrt. Das AND und das OR sind zweiwertige Grundfunktionen, also Verknüpfungen, die auf zwei Eingangsbits wirken. Das AND bewirkt, dass das Ausgangsbit nur dann 1 wird, wenn beide Eingangsbits 1 sind und in den sonstigen Fällen zu 0 wird. Bei der OR-Verknüpfung ist das Ausgangsbit 1, wenn mindestens ein Eingangsbit 1 ist. Aus diesen drei elementaren Operationen können alle logischen Verknüpfungen erzeugt werden. Ein klassischer Rechner kann jede arithmetische Aufgabe bewältigen, wenn er über eine geeignete Auswahl von Gattern verfügt.
Der Quantencomputer ist bisher lediglich ein theoretisches Konstrukt mit der Aufgabe, den Quanteninformationsprozess formal zu analysieren. In der experimentellen Realisierung werden zwar stetig Fortschritte erzielt, aber bis ein lauffähiger Quantencomputer, der einen in Kapitel 6 dargestellten Algorithmus durchführen kann, existiert, bedarf es noch erheblicher Forschungs- und Entwicklungsarbeit. Ein Quantencomputer funktioniert nun, indem er die gewohnten quantenmechanischen Niveauzustände und Superpositionszustände nutzt. Eine Aufreihung von Wasserstoffatomen kann im Prinzip ebenso gut Bits speichern wie eine Reihe von Kondensatoren. Ein Wasserstoffatom im energetischen Grundzustand würde z.B. einer 0 (im folgenden $|0\rangle$) entsprechen und eines im ersten angeregten Zustand einer 1 ($|1\rangle$).[1] Ein Spin-$\frac{1}{2}$-System, z.B. der Spin eines Elektrons, kann ebenfalls zur Speicherung von Bits benutzt werden: *Spin up* repräsentiert den Zustand $|1\rangle$ und *spin down* den Zustand $|0\rangle$. Zusätzlich können

[1] Die anderen Anregungszustände sollen hierbei keine Rolle spielen.

Elektronen aber auch jeden Zustand annehmen, der einer Überlagerung (auch Superposition) $\alpha|0\rangle + \beta|1\rangle$ mit $|\alpha|^2 + |\beta|^2 = 1$ entspricht. Im Kontext der Quanteninformationsverarbeitung nennt man die Zustände eines Zweiniveausystems Qubits. Da aber Speichern nicht ausreichend ist, muss es möglich sein, Information in das System einzubringen, sie zu verarbeiten und wieder auszulesen. Das bedeutet, aus den vorhandenen Bits durch logische Verknüpfungen neue zu gewinnen und wieder aus dem System herauszuholen, also lesen, rechnen und schreiben. Für das Problem 'Information in ein System einzubringen', fand der amerikanische Physiker Isidor Isaac Rabi (1898-1988) als erster eine Lösung: Ein Wasserstoffatom befinde sich in seinem Grundzustand mit der Energie E_0. Um eine Null einzuschreiben, tue man gar nichts. Um eine Eins einzuschreiben, muss man das Atom auf einen Zustand höherer Energie E_1 anregen. Dazu bestrahlt man es mit Laserlicht aus Photonen, deren Energie genau gleich der Differenz von E_1 und E_0 ist. Ein Laserpuls wird dann das Atom vom Grundzustand in den angeregten Zustand versetzen (siehe auch [1]). Das Auslesen der Bits aus dem Quantensystem funktioniert ähnlich wie das Schreiben: Dazu wird das System ebenfalls mit einem Laserpuls bestrahlt. War es im angeregten Zustand, wird ein Photon abgestrahlt, war es im Grundzustand, wird kein Photon registriert. Vom Schreiben und Lesen ist es dann noch ein Schritt zum Rechnen. Welche Ansätze, Ideen und Voraussetzungen dafür nötig sind, soll im Folgenden dargelegt werden.

1.2 Reversible Operationen und Gatter

Wie in Abschnitt (1.1) schon erläutert, besteht ein Computer aus einem komplexen Netzwerk von untereinander verbundenen Operationen, den Gattern. Die klassischen Operationen sind das NOT, AND und OR, wobei die letzten zwei Elemente irreversibel sind [2]. Da die Gesetze der Quantenmechanik in der Zeit reversibel sind, müssen die Operationen der Quanteninformationsverarbeitung ebenfalls reversibel sein[3], d.h. Verknüpfungen wie das AND und OR eignen sich nicht für den quantenmechanischen Informationsverarbeitungsprozess. Richard Feynman [2, 3] verfolgte wie schon Benett [4] Fredkin und Toffoli [5] den Ansatz, Computermaschinen zu betrachten, die den reversiblen Gesetzen der Quantenmechanik gehorchen. An dieser Stelle soll zunächst dargelegt werden, dass sich reversible Operationen ebenfalls eignen, logische Verknüpfungen zu erzeugen. Wie auf diesen Grundlagen unter Benutzung der Gesetze der Quantenmechanik ein Computer konstruiert werden kann, ist in Kapitel 5 erläutert.
Um eine universelle Maschine zu erstellen, können drei reversible Operationen benutzt werden. Das erste ist das NOT, das aus einer Leitung besteht. Es verliert keine Information und ist durch nochmalige Anwendung reversibel. In der folgenden Abbildung ist das NOT symbolisch dargestellt mit der zugehörigen Tabelle für die sich ergebenen Werte des Inputs a und Outputs a':



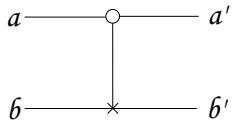
a	a'
0	1
1	0

Abbildung 1.1: NOT

[2] Das AND und das OR sind irreversibel, da durch Anwenden irgendeiner Operation der Ausgangszustand nicht reproduziert werden kann; z.B. im Fall der Verknüpfung OR kann man nicht sicher sagen, aus welchen Werten der Eingangsbits der Wert 1 im Ausgangsbit resultiert.

[3] Dieser Zusammenhang wird in Kapitel 5 verdeutlicht.

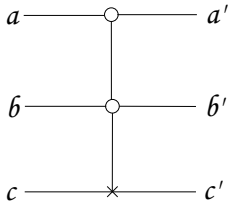
Das zweite reversible Element ist das CONTROLLED NOT. Es besteht aus zwei Eingangsleitungen a und b und zwei Ausgangsleitungen a' und b'. Die erste Leitung ist die Kontrollleitung, für die $a = a'$ gilt. Diese Kontrollleitung ist notwendig, um Zweideutigkeiten im Output zu beheben. Das CONTROLLED NOT bewirkt, dass ein NOT auf b ausgeführt wird, wenn $a = 1$. Ansonsten gilt $b = b'$ (siehe Abbildung 1.2). Da man für den Output $b' = 0$ nicht sagen kann, ob er aus $(a, b) = (0, 0)$ oder $(a, b) = (1, 1)$ entstanden ist, wird diese Zweideutigkeit im CONTROLLED NOT durch die Kontrollleitung $a = a'$ behoben. Damit ist auch das CONTROLLED NOT durch einfache Wiederholung reversibel.



a	b	a'	b'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Abbildung 1.2: CONTROLLED NOT

Die Kombination nur dieser beiden Elemente reicht noch nicht aus, um beliebige logische Funktionen auszuführen. Man benötigt noch eine weitere Operation mit drei Leitungen, das CONTROLLED CONTROLLED NOT (siehe Abbildung 1.3). Dieses besteht aus zwei Kontrollleitungen, die unverändert bleiben $(a = a', b = b')$, sowie einer dritten Leitung c, auf die NOT angewandt wird, wenn $a = b = 1$ gilt, und die ansonsten unverändert bleibt. Ist die dritte Eingangsleitung z.B. auf $c = 0$ gesetzt, dann erhält man $c' = 1$ nur dann, wenn $a = b = 1$ gilt. Die drei Kombinationen für (a, b), nämlich $(0, 0)$, $(0, 1)$ und (1,0) ergeben alle den Wert 0, so dass zwei Bits benötigt werden, um die Mehrdeutigkeit aufzuheben. Diese sind in den Kontrollleitungen a und b realisiert, so dass das CONTROLLED CONTROLLED NOT reversibel ist.

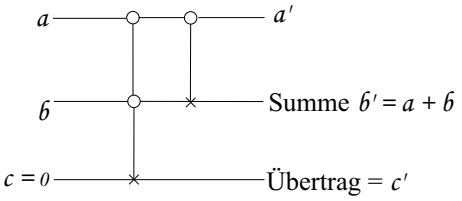


a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Abbildung 1.3: CONTROLLED CONTROLLED NOT

In der Informatik wurde gezeigt, dass man mit diesen drei Elementen und deren Kombinationen jeden logischen “Kreislauf” und somit auch einen universellen Computer erzeugen kann. Die folgenden Beispiele sollen dies verdeutlichen. In den in diesem Zusammenhang erstellten Gattern repräsentiert jede Leitung ein Bit; die Gatter werden in ihren Operationen von links nach rechts gelesen.

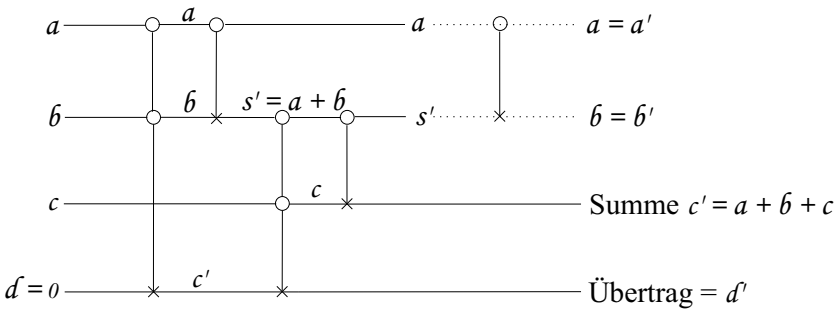
Man erhält einen ADDER (Halbaddierer), indem man zuerst das CONTROLLED CONTROLLED NOT und dann das CONTROLLED NOT in Folge benutzt (siehe Abbildung 1.4). Dieses Gatter erzeugt von den Eingangsleitungen a, b und 0 den ursprünglichen Wert von a in der zugehörigen Ausgangsleitung a', die Summe von a und b in b' und den Übertrag des CONTROLLED CONTROLLED NOT angewendet auf a und b in c'.



a	b	c	a'	b'	c'
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	1	0
1	1	0	1	0	1

Abbildung 1.4: ADDER

Ein etwas komplizierterer Schaltkreis ist der FULLADDER (Volladdierer), welcher einen Übertrag c von einer vorherigen Addition zu a und b addiert und eine zusätzliche Eingangsleitung $d = 0$ besitzt (siehe Abbildung 1.5).



a	b	c	d	a	s'	c'	d'
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	1	0
0	1	1	0	0	1	0	1
1	0	0	0	1	1	1	0
1	0	1	0	1	1	0	1
1	1	0	0	1	0	0	1
1	1	1	0	1	0	1	1

a	b	c	d	a'	b'	c'	d'
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	1	0
0	1	1	0	0	1	0	1
1	0	0	0	1	0	1	0
1	0	1	0	1	0	0	1
1	1	0	0	1	1	0	1
1	1	1	0	1	1	1	1

Abbildung 1.5: FULLADDER

Dieser Schaltkreis besteht aus vier Operationen, die sich aus den primitiven Elementen CONTROLLED CONTROLLED NOT und CONTROLLED NOT zusammensetzen. Neben der totalen Summe $c' = a + b + c$ und dem Übertrag d' erhält man auf den zwei anderen Ausgangsleitungen weitere Informationen: Den Startwert a und einen Zwischenwert s', der sich während der Routine ergeben hat. Neben den gewünschten Outputwerten erhält man also auch “Abfall”, aus dem sich jedoch die Werte der Eingangsleitungen rekonstruieren lassen. Dies ist eine für reversible Systeme typische Vorgehensweise. Im Fall eines FULLADDERs geschieht dies durch

eine zusätzliche Anwendung eines CONTROLLED NOTs auf die ersten zwei Leitungen a und s'. In dem Schaltbild des FULLADDERs ist dies durch die gestrichelten Linien dargestellt.

Auf diese Art und Weise kann man also durch verschiedene Kombinationen einen beliebigen logischen "Kreislauf" erzeugen, der n Bits reversibel in n Bits umformt. Wenn das zu berechnende Problem selbst reversibel ist, hat man keinen zusätzlichen Abfall. Allgemein benötigt man aber zusätzliche Leitungen, die die für die Reversibilität notwendigen Informationen speichern.

1.3 Quanteninformationsverarbeitung

Im idealen Fall besteht die Quanteninformationsverarbeitung aus einer Sequenz von unitären Operationen auf Registerzuständen [4], die aber möglicherweise durch Mess- und Ausleseprozesse unterbrochen und gesteuert werden können [6, 7, 8]. Ein Quantencomputer besteht letztlich aus einem endlich großen Quantenregister, also einer endlichen Anzahl N von Qubits. Wesentlich ist, dass das Quantenregister nicht nur in einem seiner 2^N Basiszustände vorliegen kann, wie z.B. $|0\rangle|1\rangle|0\rangle$ oder $|1\rangle|0\rangle|1\rangle$ für $N = 3$, sondern auch in einer beliebigen Linearkombination dieser Zustände, d.h. in verschränkten Zuständen wie $\alpha|0\rangle|1\rangle|0\rangle + \beta|1\rangle|0\rangle|1\rangle$ mit $|\alpha|^2 + |\beta|^2 = 1$. Da das Quantenregister in gewissem Sinne gleichzeitig in allen seiner möglichen Zustände sein kann, können bestimmte Informationsverarbeitungen "in hohem Maße" parallel stattfinden. Dies ist das besondere Potential der Quanteninformationsverarbeitung, die in sogenannten Quantengattern stattfindet, deren Operationen reversibel sind, wie schon in Kapitel 1.2 erläutert. Diese Gatter haben die gleiche Anzahl von Inputs und Outputs, wobei jede Leitung hier ein Qubit repräsentiert. Die zur Quanteninformationsverarbeitung benötigten Operationen sind im einfachsten Fall sogenannte Ein-Qubit- und Zwei-Qubitoperationen, wie z.B. das CONTROLLED NOT (siehe auch Kapitel 1.2). Man kann zeigen, dass sich jede Rechenoperation in eine Folge von Operationen und zwar eines CONTROLLED NOTs zusammen mit allgemeinen unitären Transformationen eines einzelnen Qubits zerlegen lässt [9].
Beispiele für Ein-Qubitoperationen sind die Hadamartransformation und die Phasenverschiebung. Die Hadamartransformation H ist von der Gestalt:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.1}$$

Betrachtet man die Zustände

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

so bewirkt die Hadamartransformation folgendes:

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \end{aligned}$$

[4] In der Quantenmechanik ist die Operation, die einen Eingangszustand z.B. $|a\rangle, |b\rangle, |c\rangle, |d\rangle$ reversibel in einen Ausgangszustand $|a'\rangle, |b'\rangle, |c'\rangle, |d'\rangle$ überführt, ein unitärer Operator, d.h. ein Operator U für den $U^\dagger U = \mathbb{1}$ gilt.

$$
\begin{aligned}
&= \frac{1}{\sqrt{2}}\left[\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\right] \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)
\end{aligned}
$$

und

$$
\begin{aligned}
H|1\rangle &= \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right).
\end{aligned}
$$

Eine weitere wichtige Ein-Qubittransformation, die Phasenverschiebung, hat die Form:

$$
P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}. \tag{1.2}
$$

Betrachtet man wiederum die Zustände $|0\rangle, |1\rangle$ dann bewirkt die Phasenverschiebung folgendes:

$$
\begin{aligned}
P|0\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= |0\rangle
\end{aligned}
$$

und

$$
\begin{aligned}
P|1\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} \\
&= i|1\rangle = \mathrm{e}^{i\frac{\pi}{2}}|1\rangle.
\end{aligned}
$$

Als elementare Zwei-Qubitoperation ist das CONTROLLED NOT (siehe Abbildung 1.2)

$$
\text{CONTROLLED NOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.3}
$$

zu nennen.
Eine weitere, für einen später angesprochenen Quantenalgorithmus (siehe Kapitel 6) grundlegende Zwei-Qubitoperation ist die Phasenveränderung, wenn der Zustand $|1\rangle|1\rangle$ vorliegt:

$$
S_{jk} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{i\varphi} \end{pmatrix}. \tag{1.4}
$$

Alle für die quantenmechanische Informationsverarbeitung relevanten Transformationen lassen sich aus der Hadamartransformation, der Phasenverschiebung und dem CONTROLLED NOT zusammensetzen [10].

1.4 Quantenfehlerkorrektur

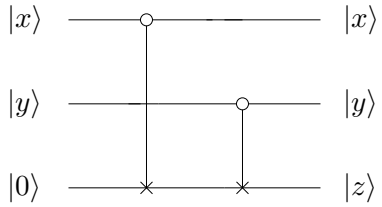
In den bisherigen Ausführungen wurde davon ausgegangen, dass man Quantengatteroperationen völlig fehlerfrei ausführen kann. Dies ist in der Realität allerdings nicht der Fall. Da die Informationsträger den Gesetzen der Quantenmechanik unterliegen, ist das Problem der Quanteninformationsverarbeitung eng mit dem Problem der Dekohärenz verbunden. Die unitäre Zeitentwicklung ist nur in vollständig isolierten Systemen gewährleistet. Jedes reale System ist jedoch, wenn auch nur schwach, an Freiheitsgrade seiner Umgebung gekoppelt, und die unkontrollierte Wechselwirkung der Informationsträger mit der Umgebung verursacht eine Dekohärenz im Zustandsraum der Qubits. Quantenmechanische Verschränkung und Superpositionszustände werden dadurch empfindlich gestört, was die Anzahl der ausführbaren Operationen begrenzt und die Realisierbarkeit von Quantenrechnern sehr schwierig gestaltet. Die Quantenfehlerkorrektur wird daher zu einem wesentlichen Bestandteil eines realen Quantencomputers.
Klassische Rechner sind recht tolerant gegenüber ungewollten äußeren Einflüssen, da jedes Bit z.B. mit Hilfe einer makroskopischen Anzahl von Elektronen gespeichert wird. Im Prinzip kann ein solches Bit mit Hilfe einer ziemlich starken – und deshalb sehr unwahrscheinlichen – Wechselwirkung mit der Umgebung spontan umspringen. Quantencomputer speichern dagegen die Information in quantenmechanischen Zweizustandssystemen (siehe Kapitel 1.1). In diesem Fall reichen Stöße oder schon eine viel schwächere Wechselwirkung mit der Umgebung aus, um ein Qubit umspringen zu lassen. Wenn man etwa zur Informationsspeicherung zwei Zustände eines Atoms benutzt, so kann der angeregte Zustand ungewollt mit einer bestimmten Wahrscheinlichkeit durch die spontane Emission eines Photons zerfallen. Außerdem können in Quantencomputern nicht nur Bitaustausche als Fehler auftreten, sondern auch Phasenfehler. Auf den ersten Blick scheinen daher reale Quantencomputer selbst bei einer sehr optimistischen Abschätzung der Fehlerquellen um viele Zehnerpotenzen von den Anforderungen entfernt zu sein, die man brauchen würde, um etwa eine Zahl mittels des Shor–Algorithmus (siehe Kapitel 6) zu faktorisieren. Es wurden jedoch Fehlerkorrekturmethoden gefunden, die Quantenrechner gegenüber ungewollten äußeren Einflüssen sehr viel unanfälliger machen, was zur Hoffnung Anlass gibt, dass die prinzipielle Realisierung des Quantencomputers durchaus möglich ist. Insbesondere wurde gezeigt, dass man im Prinzip beliebig lange Rechnungen verlässlich auf einem Quantencomputer ausführen kann, solange die einzelnen Quantengatteroperationen mit einer gewissen Mindestgenauigkeit ausgeführt werden. Dies ist insofern bemerkenswert, als dass jede Fehlerkorrekturmethode die Anzahl der Operationen, die Zahl der benötigten Qubits und somit auch die Fehleranfälligkeit erheblich erhöht. Die Lösung dieses Problems liegt in der wiederholten Anwendung der Fehlerkorrektur. Die momentanen Abschätzungen der erforderlichen Mindestgenauigkeit zeigen, dass sie bei **einem** Fehler in 10^4 Operationen liegt [11].
Im Folgenden soll ein kleiner Einblick in die Funktionsweise solcher Fehlerkorrekturmethoden vermittelt werden. Diese Methoden bauen auf den klassischen Konzepten der Fehlerkorrektur auf. Die zugrundeliegende Idee ist, in die Speicherung der Information in den Qubits eine gewisse Redundanz einzubauen. Zum Beispiel kann man ein Qubit an Information in drei realen Qubits kodieren, indem man festlegt, dass die $|0\rangle$ durch $|0\rangle|0\rangle|0\rangle$ dargestellt wird und die $|1\rangle$ durch $|1\rangle|1\rangle|1\rangle$. Nimmt man an, dass in einem der drei Qubits ein Bitflip [5] als Fehler auftritt, hat man trotzdem noch genug Information, um den Fehler zu finden und zu beheben. Jeder mög-

[5] Eine Bitposition wechselt von 0 auf 1 oder umgekehrt.

liche Bitflipfehler hat ein ihm eigenes Fehlersyndrom, was am Beispiel des Produktzustands $|0\rangle|0\rangle|0\rangle$ illustriert werden soll:

1. & 2. Qubit	2. & 3. Qubit	Qubits	Fehler
gleich	gleich	$\|0\rangle\|0\rangle\|0\rangle$	keiner
verschieden	gleich	$\|1\rangle\|0\rangle\|0\rangle$	Flip im ersten Qubit
verschieden	verschieden	$\|0\rangle\|1\rangle\|0\rangle$	Flip im zweiten Qubit
gleich	verschieden	$\|0\rangle\|0\rangle\|1\rangle$	Flip im dritten Qubit

Zum Beispiel liegt kein Fehler vor, wenn sowohl die ersten zwei als auch die letzten zwei Qubits gleich sind. Wenn die ersten zwei Qubits in unterschiedlichen Zuständen sind, die letzten zwei aber in demselben Zustand, so ist ein Flipfehler im ersten Qubit aufgetreten, usw. Da diese Information in den drei Qubits gespeichert ist, kann man durch einen geeigneten Satz von Quantengattern den Fehler detektieren und rückgängig machen. Wichtig ist, nur die (Un)gleichheit von Qubits zu messen, nicht die Qubits selbst, da dann Information zerstört würde. Im Folgenden sollen allgemeine Qubits $|x\rangle$ und $|y\rangle$ ($|x\rangle$ kann z.B. $|0\rangle$ oder $|1\rangle$ sein) betrachtet werden, um die Operation, die das Gewünschte leistet, zu erläutern (siehe Abbildung 1.6): Sind die Qubits $|x\rangle$ und $|y\rangle$ gleich, wechselt das Hilfsqubit, welches zunächst auf $|0\rangle$ gesetzt ist, kein- oder zweimal den Zustand und bleibt so am Ende im Zustand $|0\rangle$. Sind $|x\rangle$ und $|y\rangle$ verschieden, so liegt das Hilfsqubit am Ende im Zustand $|1\rangle$ vor.



x	y	0	x	y	z
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	1	0	1
1	1	0	1	1	0

Abbildung 1.6: Messung von Qubits auf Fehler

Mit einer ähnlichen Methode können auch Phasenfehler behandelt werden. Dazu nutzt man die Tatsache aus, dass ein Phasenfehler durch Anwendung der Hadamartransformation zu einem Bitflipfehler wird (siehe dazu [12, 13, 14]).

Kapitel 2

Quantenalgorithmen

Die Erweiterung des Informationsbegriffes erlaubt es, neue Algorithmen, sogenannte Quantenalgorithmen, zu entwickeln. Die Möglichkeit, durch quantenmechanische Überlagerung verschiedene Registerzustände quasi gleichzeitig zu verarbeiten, wird Quantenparallelismus genannt und verspricht, gewisse mathematische Probleme auf einem Quantencomputer effizienter zu lösen, als dies mit einem klassischen Computer möglich ist. Die Anforderungen an die Präzision, mit der die Gatteroperationen realisiert werden müssen, um beliebige Quantenalgorithmen implementieren zu können, sind allerdings sehr hoch. Die Theorie des fehlertoleranten Quantenrechnens liefert einen Wert in der Größenordnung von 10^{-4} als relative Ungenauigkeit pro Rechenschritt bzw. Gatteroperation [11]. Mit anderen Worten, von 10000 Operationen darf höchstens eine Operation fehlerhaft sein, damit der Quantencomputer noch funktioniert.
Die neuen Kapazitäten des Quantencomputers sollen in diesem Kapitel an Hand eines einfachen Beispiels, dem Deutsch–Algorithmus verdeutlicht werden. Dieses Beispiel zeigt, dass Quantencomputer Probleme lösen können, vor denen klassische Computer kapitulieren. Außerdem kann man an diesem Beispiel schon ein wesentliches Charakteristikum komplizierter Quantenalgorithmen, den Quantenparallelismus, ablesen. Ein zweites Beispiel ist der Shor–Algorithmus (siehe Kapitel 6), der der Faktorisierung von großen Zahlen dient. Dieser kann als einer der wichtigsten bisher ausgearbeiteten Algorithmen angesehen werden.

2.1 Der Deutsch–Algorithmus

Der Deutsch–Algorithmus dient der effizienten Berechnung einer binären Funktion $f : \{0,1\} \to \{0,1\}$. Dazu stelle man sich vor, man habe eine “Blackbox”, die diese binäre Funktion berechnet, die also ein Bit x in ein einzelnes Bit $y := f(x)$ verwandelt. Nimmt man an, dass eine Berechnung 24 Stunden dauert, dann müssen die Vorgänge in der Box dementsprechend recht kompliziert sein. Weil jeder der beiden Funktionswerte $f(0)$ und $f(1)$ einen der möglichen Werte 0 oder 1 annehmen kann, gibt es insgesamt vier Möglichkeiten für die Berechnung der Funktion und man möchte wissen, was die Box berechnet. Nun benötigt man das Resultat aber in 24 Stunden. Es sei angenommen, dass es reichen würde zu wissen, ob $f(x)$ konstant ist, also $f(0) = f(1)$, oder nicht konstant ist, also $f(0) \neq f(1)$. Ein klassischer Computer müsste $f(0)$ und $f(1)$ berechnen und dann die Ergebnisse vergleichen. Dies würde für beide Funktionswerte $f(0)$ und $f(1)$ insgesamt 48 Stunden dauern. Der Quantencomputer

hingegen benötigt nur eine Messung, was nun dargestellt werden soll:
Im Folgenden bezeichnet $\oplus$ die binäre Addition (mod 2)[1], d.h.

$\oplus$	0	1
0	0	1
1	1	0

.

Angenommen die "Blackbox" berechnet $f(x)$ und der Computer ist so programmiert, dass er auf den Produktbasiszuständen $|x\rangle|y\rangle$ mit $x, y \in \{0, 1\}$ die Operation

$$U(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle \qquad \text{mit} \quad x, y \in \{0, 1\} \tag{2.1}$$

ausführt (siehe Abbildung 2.1).

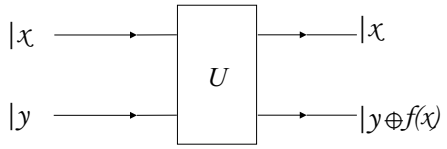


Abbildung 2.1: Operation $U(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle$

Diese Operation bedeutet in Worten, dass die Maschine das zweite Qubit verändert, wenn f auf das erste Qubit 1 ergibt und dass die Maschine gar nichts macht, wenn f auf das erste Qubit 0 ergibt. Um die Reversibilität sicherzustellen, muss die Blackbox zwei Eingänge haben, die durch die Qubits $|x\rangle$ und $|y\rangle$ realisiert sind.
Die Quantenmechanik erlaubt es, als Eingangszustand einen Überlagerungszustand der beiden Qubits $|x\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ und $|y\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$ zu wählen, was der Schlüssel zur Lösung von Deutschs Problem ist. Diesen Eingangszustand präpariert man, indem man ausgehend vom Zustand $|0\rangle|0\rangle$ zunächst ein NOT (siehe Kapitel 1.2) auf den zweiten Eingang anwendet, was den Zustand $|0\rangle|1\rangle$ liefert (siehe Abbildung 2.2). Im Anschluss daran wird die Hadamartransformation (siehe Kapitel 1.3) auf beiden Leitungen ausgeführt und man erhält den gewünschten Anfangszustand

$$|\psi_{in}\rangle := |x\rangle|y\rangle = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right). \tag{2.2}$$

Nach Anwendung von U auf $|\psi_{in}\rangle$, ergibt sich

$$\begin{aligned}
|\psi_{out}\rangle &:= U|\psi_{in}\rangle \\
&= U\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\
&= \frac{1}{2}\, U\left(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle\right) \\
&= \frac{1}{2}\left(|0\rangle|0 + f(0)\rangle - |0\rangle|1 + f(0)\rangle\right. \\
&\qquad \left. + |1\rangle|0 + f(1)\rangle - |1\rangle|1 + f(1)\rangle\right).
\end{aligned}$$

[1]Eine Einführung in die zahlentheoretischen Grundlagen der Kongruenzen und Restklassen befindet sich im Anhang A.1

Nun sei für $z \in \{0,1\}$, $1+z=\bar{z}$, wobei $\bar{1}=0$ und $\bar{0}=1$. Mit folgender Tabelle

$\oplus$	$f(0)$	$f(1)$
0	$f(0)$	$f(1)$
1	$\overline{f(0)}$	$\overline{f(1)}$

erhält man

$$\begin{aligned} |\psi_{out}\rangle &= \frac{1}{2}\left(|0\rangle|f(0)\rangle - |0\rangle|\overline{f0)}\rangle + |1\rangle|f(1)\rangle - |1\rangle|\overline{f(1)}\rangle\right) \\ &= \frac{1}{\sqrt{2}}\left[|0\rangle\left(\frac{|f(0)\rangle - |\overline{f(0)}\rangle}{\sqrt{2}}\right) + |1\rangle\left(\frac{|f(1)\rangle - |\overline{f(1)}\rangle}{\sqrt{2}}\right)\right]. \end{aligned} \tag{2.3}$$

Wie erwartet kommen in diesem Zustand Zustände für die Funktionswerte sowohl für $x=0$ als auch für $x=1$ vor. Diese Eigenschaft wird als Quantenparallelismus bezeichnet.
Durch eine Messung kann man allerdings nicht beide Funktionswerte explizit bestimmen: Sobald man das erste Qubit misst und dies z.B. den Wert 1 ergibt, enthält das zweite Qubit nur noch Information über den Funktionswert $f(1)$. Die Frage nach der Konstanz der Funktion kann dennoch in einem Schritt beantwortet werden; dazu betrachte man folgende Fallunterscheidung:

(1) f konstant $\Leftrightarrow f(0)=f(1)$

$$\Rightarrow |\psi_{out}\rangle = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|f(0)\rangle - |\overline{f(0)}\rangle}{\sqrt{2}}\right). \tag{2.4}$$

(2) f nicht konstant $\Leftrightarrow f(0) \neq f(1)$

$$\begin{aligned} &\Leftrightarrow \quad f(1) = \overline{f(0)}, \qquad \overline{f(1)} = f(0) \\ &\Rightarrow \quad |\psi_{out}\rangle = \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\left(\frac{|f(0)\rangle - |\overline{f(0)}\rangle}{\sqrt{2}}\right). \end{aligned} \tag{2.5}$$

Also gilt:

$$|\psi_{out}\rangle = \left(\frac{|0\rangle \pm |1\rangle}{\sqrt{2}}\right)\left(\frac{|f(0)\rangle - |\overline{f(0)}\rangle}{\sqrt{2}}\right). \tag{2.6}$$

Daran erkennt man, dass der Zustand des ersten Qubits davon abhängt, ob die Funktion konstant ist ("+") oder nicht ("$-$"). Folglich muss man nur noch messen, in welchem dieser beiden Zustände sich das erste Qubit befindet, um die Frage nach der Konstanz der Funktion beantworten zu können, d.h. die Blackbox wird nur einmal durchlaufen.
Konkret kann man durch Anwenden der Hadamartransformation (siehe Kapitel 1.3) die beiden in $|\psi_{out}\rangle$ enthaltenen orthogonalen Zustände unterscheiden:

$$|0\rangle \quad \rightarrow \quad \frac{|0\rangle + |1\rangle}{\sqrt{2}} \tag{2.7}$$

$$|1\rangle \quad \rightarrow \quad \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \tag{2.8}$$

Man erhält also den Zustand $|0\rangle$ für eine konstante Funktion und $|1\rangle$ für nicht konstantes f. Alle Operationen, die für die Durchführung des Deutsch–Algorithmus' notwendig sind, sind in der folgenden Abbildung nochmals zusammenfassend dargestellt:

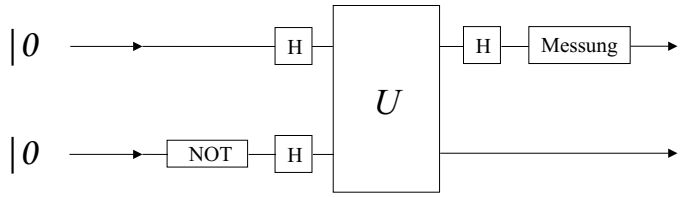


Abbildung 2.2: Schematische Darstelllung des Deutsch–Algorithmus

Verallgemeinert lässt sich folgendes festhalten: Mit Hilfe des Quantenparallelismus ist es zwar nicht möglich alle Funktionswerte gleichzeitig auszuwerten, gewisse globale Eigenschaften der Funktion sind aber doch effektiver bestimmbar als mit klassischen Algorithmen. Dies ist ein Merkmal für Quantenalgorithmen. So beruht der wesentliche Schritt im später (Kapitel 6) behandelten Quanten–Shor–Algorithmus darauf, die Periode einer Funktion zu finden. Dafür wird wiederum der Quantenparallelismus verwendet, indem man eine diskrete Fouriertransformierte berechnet, von der die Periode mehr oder weniger direkt abgelesen werden kann. Die diskrete Fouriertransformierte wird als Überlagerung aller möglichen Verschiebungen der Funktion mit verschiedenen Phasen quantenparallel berechnet. Der Deutsch–Algorithmus berechnet für einen besonders einfachen Fall ebenfalls die Periode einer Funktion.

Kapitel 3

Experimentelle Realisierung

Es stellt sich die Frage, warum man heute noch keine Quantencomputer hat, die die schnellen Rechnungen, wie beispielsweise die Faktorisierung großer Zahlen mittels des Shor–Algorithmus (siehe Kapitel 6), durchführen können.
Für die Implementierung eines Quantencomputers ist es notwendig, ein physikalisches System zu präparieren, das es erlaubt, Quantenzustände verlässlich zu speichern sowie mit Quantenoperationen gezielt und präzise zu manipulieren. In der Praxis gibt es bisher nur wenige Kandidaten, die diese Voraussetzungen erfüllen. Zur Realisierung eines Quantencomputers müssen eine Reihe von Bedingungen gewährleistet sein, die nun weiter beleuchtet werden sollen:

- Identifikation einzelner Qubits,
- Adressierbarkeit und Auslesen der Bits,
- Implementierung von Quantengattern,
- schwache Dekohärenz,
- effiziente Implementierung von Fehlerkorrekturen,
- Skalierbarkeit von wenigen auf viele Qubits.

Das Hauptproblem der experimentellen Realisierung ist die Forderung nach guter Manipulierbarkeit der Information einerseits und möglichst guter Abschirmung störender Einflüsse andererseits. Zum einen müssen die Qubits für die Informationsverarbeitung selektiv transformiert oder ausgelesen werden (Ein-Qubitoperationen) und idealerweise auch zwei beliebige Qubits miteinander verknüpft werden können (Zwei-Qubitoperationen); zum anderen soll der Kohärenzverlust minimiert werden, so dass die Qubitzustände möglichst lange ihre quantenmechanische Verschränkung beibehalten. Jede unkontrollierte Wechselwirkung erzeugt eine Verschränkung des Quantencomputers mit der Umgebung, welche die Zustände so beeinflusst, dass die Voraussetzungen zur Ausführung von Quantenrechnungen nicht mehr erfüllt sind. Diese ungewollten Effekte fasst man unter den Begriff der Dekohärenz zusammen. Kurz gesagt erfordert die Manipulierbarkeit der Information einen gewissen Einfluss der Umgebung, die Speicherung der Information jedoch ein möglichst von der Umwelt isoliertes System. Benötigt wird infolgedessen ein gezielt an- und ausschaltbarer Einfluss der Umgebung. Außerdem sollte es möglich

sein, das System ohne allzu großen Aufwand zu vergrößern; das System sollte skalierbar sein, d.h. die Hinzunahme eines weiteren Qubits sollte die Adressierbarkeit und Dekohärenz nicht vervielfachen, sondern nur anteilig erhöhen.
Eine Vorreiterrolle beim Bau von Quantencomputern spielt die Quantenoptik. Als Träger der Quanteninformation kommen in der Quantenoptik einzelne Atome und Photonen in Frage. Ziel ist die Speicherung einzelner Atome in Fallen und die Präparation eines Atoms im Bewegungsgrundzustand [15, 16, 17, 18, 19]. Mittels Laserpulsen können die internen Zustände der Atome (die Qubits) gezielt manipuliert werden. Somit lassen sich Ein-Qubitoperationen durch Wechselwirkung von Laserlicht mit Atomen realisieren. Die Implementierung von Zwei-Qubitoperationen lässt sich entweder durch Ankopplung an Hilfsfreiheitsgrade, wie z.B. kollektive Schwingungsmoden von gespeicherten Atomen, Ionen oder Photonen in einem Resonator, erreichen, oder auch durch gezielte Zweiteilchenwechselwirkungen zwischen zwei Atomen. Zur gezielten Verschränkung der internen Zustände zweier Atome kann eine Reihe von kontrollierbaren Zweiteilchenwechselwirkungen verwendet werden, deren Stärke vom Zustand der Qubits abhängt. Um Ionen zu verschränken, eignet sich die Coulomb-Wechselwirkung, deren Stärke proportional zum inversen Abstand der Ionen ist. Zur Verschränkung von neutralen Atomen werden isotrope Stoßprozesse bei tiefen Temperaturen (so genannte kalte Stöße) verwendet. Das Wechselwirkungspotential hat allerdings nur eine sehr begrenzte Reichweite. Eine weitere Möglichkeit ist die Wechselwirkung zwischen permanenten Dipolmomenten von Rydberg-Zuständen in wasserstoffartigen Atomen in statischen elektrischen Feldern. Dort wird die Wellenfunktion des Valenzelektrons so verzerrt, dass es zu einer Verschiebung des Ladungsschwerpunktes der Elektronenhülle im Vergleich zum Ladungsschwerpunkt des Kerns kommt. Es entsteht ein Dipolmoment, dessen Stärke mit dem Quadrat der Hauptquantenzahl des Zustandes anwächst. Die Wechselwirkungsstärke zwischen den Dipolmomenten zweier benachbarter Atome ist proportional zur vierten Potenz der Hauptquantenzahl und umgekehrt proportional zur dritten Potenz des Abstandes der Atome. Selbst für eine Hauptquantenzahl von weniger als 15 liefert dies wesentlich größere Wirkungsquerschnitte als kalte Stöße zwischen neutralen Atomen.
Ein Beispiel ist der in [20] vorgestellte Quantencomputer, der auf der Wechselwirkung von Lasern mit gekühlten Ionenketten basiert. Dies ist vermutlich der erste Vorschlag zur Verwirklichung eines Quantenrechners oder zumindest eines kleineren Quanteninformationssystems. An der experimentellen Realisierung wird zur Zeit weltweit in mehreren Labors, darunter am NIST Boulder, an der Universität Innsbruck, dem Max-Planck-Institut für Quantenoptik in Garching und in Los Alamos gearbeitet. Dieses Modell nutzt als Qubits ultrakalte Ionen, die zwischen zwei inneren Zuständen wechseln können und die eine Kette bilden, die in einer linearen Ionenfalle, der Paul-Falle, gespeichert wird. Diese Falle besteht aus vier parallelen Metallstäben. Ein zeitabhängiges Radiofrequenzpotential wird an jeweils zwei gegenüberliegenden Stäben angelegt. Damit können sich die Ionen nur noch auf einer Linie parallel zu den Stäben anordnen. Zwei zusätzliche zirkulare Elektroden dienen als Abschlüsse an den beiden Enden (siehe Abbildungen 3.1, 3.2).

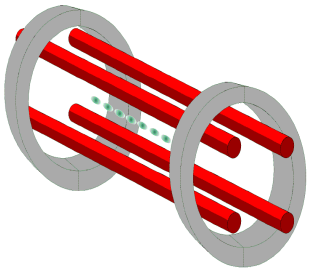


Abbildung 3.1: Paul–Falle (entnommen aus [22]).

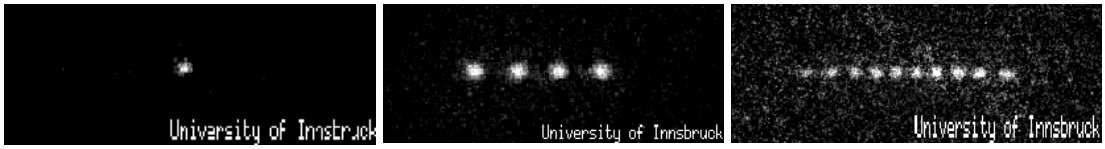


Abbildung 3.2: Ionen in der Paul–Falle (entnommen aus [22]).

Um die störenden spontanen Zerfälle des energetisch höheren Qubitniveaus zu minimieren, benutzt man einen metastabilen Zustand mit der verglichen mit typischen Lebensdauern im Nanosekundenbereich großen Lebensdauer von etwa einer Sekunde. Die Gleichgewichtslage der Ionen im Querschnitt der Falle ergibt sich aus dem Fallenpotential. Schwingungen der Ionenkette entsprechen kleinen Auslenkungen der Ionen entlang der Fallenachse aufgrund der Coulombabstoßung. Durch Laserkühlen kann der Schwingungszustand der Ionen ausgefroren werden, und die Ionenkette wird im Schwingungsgrundzustand präpariert. Der Gesamtzustand der Ionenkette ist somit durch einen Zustandsvektor zu beschreiben, der ein Produktzustand der internen Ionenzustände, des Quantenregisterzustands und der quantisierten Schwerpunktsbewegung im Grundzustand ist. Die phononischen Freiheitsgrade dienen nun als Datenbus zur Verschränkung der internen Ionenzustände. Insbesondere kann ein einzelnes Ion derart mit einem rotverstimmten Laserpuls bestrahlt werden, dass es vom Zustand $|1\rangle$ in den Zustand $|0\rangle$ übergeht und dabei die Ionenkette durch die Abgabe eines Phonons in Schwingung versetzt. Falls das Ion im Zustand $|0\rangle$ ist, wird der Zustand nicht verändert. Im Fall einer Ionenkette kann in einem beliebigen zweiten Ion der interne Zustand gemäß $|0\rangle \leftrightarrow |1\rangle$ umgeschaltet werden. Insgesamt erhält man so einen Prozess, bei dem der interne Zustand des zweiten Ions verändert wird, sofern das erste Ion sich im Zustand $|1\rangle$ befindet. Wird abschließend der Anfangszustand des ersten Ions wieder hergestellt, lässt sich damit die schon bekannte CONTROLLED NOT-Operation (siehe Abbildung 1.2) realisieren [21, 23]. Jede Rechenoperation kann als eine Folge von solchen Operationen zusammen mit einfachen Zustandsänderungen der einzelnen Ionen dargestellt werden. Am Beginn einer Rechnung setzt man dabei durch optisches Pumpen den Zustand aller Ionen auf einen gewünschten Anfangswert, z.B. auf $|0\rangle|0\rangle \ldots |0\rangle$. Um den Messprozess ausführen, d.h. um die Bitzustände der Ionen am Ende einer Rechnung auslesen zu können, verwendet man die Methode der Quantensprünge. Dabei wird die Ionenkette mit Laserlicht einer geeigneten Frequenz bestrahlt, so dass ein Ion im Zustand $|1\rangle$ Fluoreszenzlicht ausstrahlt, während es im Zustand $|0\rangle$ dunkel bleibt. Experimentell wurden bisher quantenlogische Verschränkungsoperationen mit einem und bis zu vier Ionen nachgewiesen [18, 19].
An Erweiterungen wird intensiv gearbeitet, und man hofft, in näherer Zukunft bis auf etwa zehn

Qubits zu kommen [24]. Die Ionenfallen bieten sicherlich sehr gut abgeschirmte Qubits. Die Manipulierbarkeit von Qubitpaaren erscheint eher als Schwachpunkt, insbesondere bezüglich größerer Systeme. Ein weiteres Problem ist das Laserkühlen in den Grundzustand des Fallenpotentials, das eine der Voraussetzungen, zugleich aber eine Schwierigkeit der experimentellen Realisierung ist.

Bei einem anderen Realisierungsvorschlag nimmt man an, dass Ionen in periodisch angeordneten Mikrofallen gefangen werden [25], wie sie beispielsweise mit lithographischen Methoden erzeugt werden können. Auch hier sind die Träger der Qubits langlebige elektronische Zustände der Ionen. Die Ionen sind ebenfalls mit Lasern einzeln adressierbar, so dass Ein-Qubitoperationen realisiert werden können. Zur Verwirklichung von Zwei-Qubitoperationen müssen die Ionen in kontrollierter Weise miteinander und mit dem äußeren Coulombfeld wechselwirken. Das Grundprinzip der Realisierung der Zwei-Qubitoperationen in diesem Modell ist die Anwendung eines äußeren Laserfeldes auf ein Ion, so dass seine Wellenfunktion in Abhängigkeit von seinem internen Zustand räumlich verschoben wird (Abbildung 3.3). Indem zwei benachbarte Ionen in $|1\rangle$ mit dem Laser verschoben werden, ergibt sich eine Energieverschiebung im äußeren Feld, und somit tritt – über die Zeit integriert – eine Phasenänderung auf. Das Ergebnis ist ein Phasengatter (siehe Gleichung 1.4) mit einer Phase, die sich aus der zeitlichen Integration der Änderung der Coulomb-Wechselwirkungsenergie der beiden benachbarten Ionen ermitteln lässt. Einen Quantencomputer, der auf diesen Ideen basiert, zeigt Abbildung (3.4). Dabei nimmt man an, dass ein Ion als Kopf und ein zweites Ion nahe aneinander gebracht werden können; es ist nicht notwendig, die beiden Ionen einzeln zu adressieren.

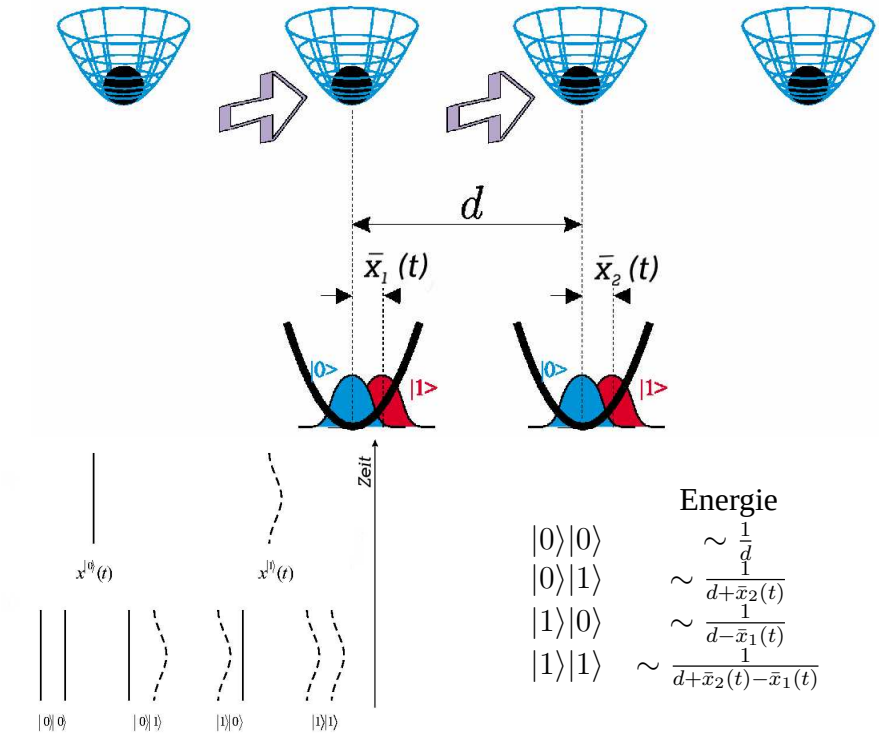


Abbildung 3.3: Schema einer Zwei-Qubitoperation mit in unabhängigen Mikrofallen gespeicherten Ionen. Um eine Operation auszuführen, werden die Ionen räumlich verschoben, falls sie im Zustand $|1\rangle$ sind (entnommen aus [26, 25, 27]).

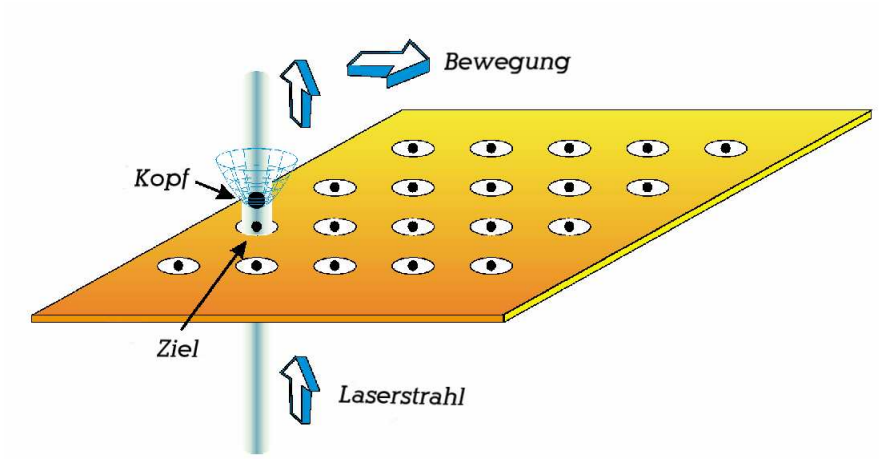


Abbildung 3.4: Ein skalierbarer Quantencomputer. Die Qubits (Ionen) sind in unabhängigen Fallen in der Ebene gefangen. Ein weiteres Ion, als Kopf bezeichnet, wird über der Ebene bewegt und kann mit einem beliebigen anderen Ion eine Zwei-Qubitoperation ausführen. Damit lassen sich Verschränkungsoperationen zwischen zwei beliebigen Ionen ausführen (entnommen aus [25]).

Der Abstand zwischen den Ionen in der Ebene kann hingegen groß sein, da keine direkte Wechselwirkung zwischen ihnen erforderlich ist. Eine solche zweidimensionale Anordnung hat offensichtlich die Eigenschaft, zu einer großen Zahl von Quantenbits skalierbar zu sein. Ein weiterer Vorteil ist, dass die Forderung nach niedrigsten Temperaturen in diesem Modell nicht besteht.

Ein Vorschlag, der besonderes Interesse ausgelöst hat, betrifft die Entwicklung eines Quantencomputers mittels Kernspinresonanz [28, 29, 30]. Einige der ersten Realisierungen von Quantenalgorithmen wie des Deutsch-Algorithmus' [31, 32] und einer sehr einfachen Grover-Suche [33] basieren auf Kernspinresonanzexperimenten, wie sie aus Anwendungen in der physikalischen Chemie und auch in der Medizin sehr gut bekannt sind. Dabei sind die Qubits durch den Spin der Kerne einzelner Atome in den betrachteten Molekülen, etwa Chloroform, gegeben. Ein-Qubitoperationen werden durch gezielte Pulse magnetischer Felder bei abgestimmten Frequenzen erreicht. Bei Zwei-Qubitoperationen nutzt man hingegen die sehr schwache Wechselwirkung der Kernspins miteinander. Der wesentliche Unterschied zu den Vorschlägen aus der Quantenoptik besteht darin, dass anstelle von Einzelsystemen ein Ensemble von Molekülen verwendet und das System außerdem bei endlicher Temperatur betrachtet wird. Obwohl dieser Ansatz einige schöne Demonstrationen ermöglicht hat, wird sein Nutzen durch zwei Nachteile eingeschränkt: Die Wechselwirkung zwischen den Kernspins ist nicht gut manipulierbar. Außerdem lassen sich die Systeme kaum wesentlich vergrößern, da die verwertbaren Signale bei größeren Systemen schnell sehr klein werden und so nicht mehr skalierbar sind. Hinzu kommt noch der Nachweis, dass die mit NMR-Methoden bislang erzeugten Quantenzustände bei endlicher Temperatur separabel sind, d.h. keine Verschränkung aufweisen [34].

Eine Reihe von Vorschlägen beruht auf Systemen der Festkörperphysik in Anlehnung an die hochentwickelte Halbleitertechnologie. Hier besteht ein wissenschaftliches Umfeld mit Erfahrungen in der Erzeugung von immer kleineren geordneten Strukturen (Mikrochips, Nanotechnologie), auf die man aufbauen kann. Daraus kann man schon vermuten, dass diese Vorschläge Stärken in der Manipulierbarkeit und der Skalierbarkeit haben werden. Ihre Schwäche liegt in

der Abschirmung der Umgebung. Zwei Modelle beruhen auf dem Spin. Bei dem einen Modell besteht das Qubit aus den Niveaus eines Kernspins gezielt eingebrachter Fremdatome, etwa Phosphor $^{31}\mathrm{P}$ in einen Siliziumhalbleiter [35]. Ein konstantes Magnetfeld B_0 von ungefähr 2 Tesla trennt die beiden Niveaus. Ein-Qubitoperationen werden über ein zeitlich oszillierendes sehr schwaches Magnetfeld B_{osz} von 10^{-3} Tesla in Resonanz realisiert. Die Resonanzfrequenz kann dabei über die Hyperfeinwechselwirkung mit der Elektronenwolke, die das Donatoratom Phosphor umgibt, beeinflusst werden. Zieht man mittels einer positiven Gatespannung V_1 die Elektronen weg, vermindert sich diese Wechselwirkung und die Resonanzfrequenz sinkt ebenfalls. Zwei-Qubitoperationen können kontrolliert aus- oder eingeschaltet werden, indem die Elektronenwolken benachbarter Donatoratome durch eine negative Gatespannung V_2 auseinander gehalten werden oder durch eine positive Gatespannung V_2 verschmolzen werden. Kernspins haben den Vorteil, dass sie sehr gut von der Umgebung isoliert sind.
Bei dem zweiten Vorschlag übernimmt der Spin eines überzähligen Elektrons auf einem Quantenpunkt die Rolle des Qubits [36]. Die Ein-Qubitoperationen werden wiederum über Magnetfelder realisiert. Die Zwei-Qubitoperationen werden über verschiedene Gatespannungen, die die Elektronenverteilungen in den Quantenpunkten verformen, erzeugt. Als Beispiele für die Realisierung von Quantenoperationen sei hier auf Vorschläge mit Cooper-Paaren in Josephson-Kontakten [37] und mit Spinzuständen von Elektronen in Quantenpunkten [36] als Träger von Qubits verwiesen. Außerdem gibt es einen Vorschlag, einen NMR-Quantencomputer mit Methoden der Festkörperphysik zu realisieren [35].
Eine andere Möglichkeit nutzt von vornherein ein Quantenphänomen, das makroskopisch auftritt: die Supraleitung [38]. Für diesen Vorschlag gibt es die Realisierung eines einzelnen Qubits [39]. Dies ist zwar noch wenig, aber man hofft, dass die Entwicklung in diese Richtung noch Erfolge erzielen wird.
Die Erwartung für die Zukunft ist, dass während der nächsten 5 Jahre kleine Quantencomputer mit etwa 10 Qubits im Labor zur Verfügung stehen werden. Dies wird sicher die experimentelle Grundlage für eine Reihe von fundamentalen Experimenten zur Teilchenverschränkung, zum Messprozess und für Dekohärenzstudien in der Quantenmechanik sein. Ferner werden auch “Proof-of-Principle”-Experimente in der Quanteninformationsverarbeitung erfolgen können. Wann die tatsächliche Anwendung als Quantencomputer im Sinne von Shor und Grover möglich sein wird, wann also effizientere Implementierungen skalierbarer Konzepte, die sich auf eine größere Anzahl von Qubits anwenden lassen, erfolgen können, bleibt abzuwarten.

Kapitel 4

Der Weg zum Quantencomputer

Das Phänomen Quantencomputer vereinigt die Ideen der klassischen Informationstheorie, der Informatik und der Quantenphysik. Die Informationstheorie ist eine 1948 von dem amerikanischen Mathematiker Claude Elwood Shannon begründete mathematische Theorie, die sich mit der strukturellen und quantitativen Erfassung und mit den (statistischen) Gesetzmäßigkeiten der Übermittlung und Verarbeitung von Nachrichten sowie den in ihnen enthaltenen Informationen beschäftigt. Die Informationstheorie war die erste Theorie, die es erlaubte, den Informationsbegriff mathematisch zu fassen und so eine quantitative Untersuchung von Informationsübertragung und -verarbeitung zu ermöglichen.
Die Grundlagen der Informatik wurden ungefähr zur gleichen Zeit formuliert wie Shannons Informationstheorie. Als Väter der Informatik sind Charles Babbage (1791-1871) und Alan Turing (1912-1954) zu nennen. Charles Babbage ist wegen seiner Beiträge zum grundlegenden Design des Computers durch seine analytische Maschine auch als “Vater des automatischen Rechnens” bekannt. Seine Differenzmaschine war speziell für das Erstellen von astronomischen und mathematischen Tabellen (z.B. Logarithmentafeln) sowie Versicherungstabellen bestimmt.
Die in der Mitte der 1930er Jahre von Alan Turing entwickelte universelle Maschine wird nach ihm “Turing-Maschine” genannt. Die Turing-Maschine ist ein idealisiertes mathematisches Modell eines Computers, das benutzt werden kann, um die Grenzen der Anwendung eines Computers zu fassen. Es erhebt keinen Anspruch darauf, ein praktisches Design für jede aktuelle Maschine zu sein, sondern dient eher dazu, die essentiellen Merkmale eines jeden Computers darzulegen. Turing wollte mit seiner Maschine zeigen, wie ein Rechenvorgang in eine Folge kleinster und einfachster Schritte zerlegt werden kann.
Der dritte Aspekt, der im Quantencomputer Anwendung findet, ist das Einbeziehen der quantenmechanischen Sichtweise. Die ersten Ideen dazu befassen sich mit der Umwandlung der Arbeitsschritte einer Turing-Maschine in einen äquivalenten reversiblen Prozess und dem Aufstellen eines Hamiltonoperators für das dazugehörige Quantensystem. Diese Ideen basieren auf einer Arbeit von Bennett [4], die zeigte, dass ein universeller klassischer Computer, wie die Turing-Maschine, unter Beibehaltung seiner einfachen Prinzipien reversibel gemacht werden kann (siehe auch [40]).
In den frühen 1980er Jahren zeigte Benioff vom Argonne National Laboratory (Illinois), dass ein Computer, der ausschließlich nach den Gesetzen der Quantenmechanik arbeitet, theoretisch funktionieren kann [41, 42]. Er stellte ein Modell einer reversiblen Turing-Maschine vor, die

lesen und schreiben konnte, sowie Operationen vollständig ausführte und dafür quantenmechanische Wechselwirkungen benutzte. Obwohl sich die daraus resultierende Maschine bezüglich des Rechenaufwands noch wie ein klassischer Computer verhielt, ist dies doch als der erste Versuch anzusehen, die Quantenmechanik in das mathematische Modell eines Computers einzubringen. In diesem Zusammenhang machte Benioff verschiedene Vorschläge für Turing-artige Hamiltonoperatoren. Allerdings waren seine Ideen keine vollständige Untersuchung der Quantenrechner, da sie lediglich klassische Rechner reproduzierten.

Darüber hinausgehende Überlegungen stellte der Physik-Nobelpreisträger Richard Feynman [2, 3] an. Er beschäftigte sich mit der kontroversen Frage, wie gut klassische Computer Quantensysteme simulieren können. Er schloss aus seinen Betrachtungen, dass klassische Computer Quantensysteme nicht effizient simulieren können, Quantensysteme sich hingegen im Prinzip immer für die Simulation irgendeines anderen Systems eignen. Dies war der erste Hinweis darauf, dass die rechnerische Effektivität einer Quantenvorrichtung, speziell eines Quantensimulators, die Kapazität einer klassischen Maschine übertreffen könnte. Seinen Ansatz kann man allerdings nicht als ausgereiftes Computersystem bezeichnen, da er zwar voraussetzte, dass jede Wechselwirkung zwischen angrenzenden Zweizustandssystemen “geordnet” werden kann, jedoch nicht beschrieb, wie dies geschehen soll.

Im Jahr 1985 gelang David Deutsch vom Mathematischen Institut der Universität Oxford (England) bezüglich des Quantencomputers ein entscheidener Schritt vorwärts: Deutsch beschrieb die erste Quanten-Turing-Maschine [43]. Dies war eine Maschine, die lesen, schreiben und Operationen durchführen konnte, die alle durch quantenmechanische Wechselwirkungen zustande kamen, und deren Register jetzt zusätzlich in nicht klassischen Zuständen existieren konnte. Während eine herkömmliche klassische Turing-Maschine nur 0, 1 oder Leerzeichen an jeder Stelle des Registers lesen konnte, konnte die Quanten-Turing-Maschine gleichzeitig auch eine Superposition von 0 und 1 entschlüsseln. Daher hat die Quanten-Turing-Maschine das Potential, mehrere Inputs eines Problems simultan im gleichen Register zu entschlüsseln und eine Berechnung mit allen Inputs in der gleichen Zeit durchzuführen, die benötigt wird, um eine Berechnung klassisch auszuführen. Dieser Effekt wird Quantenparallelismus genannt. Die Gesetze der Quantenmechanik erlauben es allerdings nicht, mehr als einen dieser Werte explizit zu extrahieren. Das Problem liegt darin, dass das Erhalten eines Wertes eine Messung notwendig macht, durch die die restlichen Ergebnisse unwiderruflich verloren gehen. Somit ist das endgültige Resultat nicht besser als das mit einer klassischen Turing-Maschine erzielte. Trotzdem erhält man in effizienter Weise verschiedene globale Eigenschaften der Outputs (siehe dazu Kapitel 2.1).

Essentiell ist das System von Deutsch eine Reihe von Zweizustandssystemen und sieht eher aus wie eine Registermaschine als wie eine Turing-Maschine; beides sind jedoch universelle klassische Rechenmaschinen. Deutsch bewies, dass für die Simulation eines jeden physikalischen Systems eine unitäre Entwicklung aufgestellt werden kann, wenn eine Entwicklung des Zweizustandssystems durch eine bestimmte kleine Anzahl von einfachen Operationen möglich ist. Die gleichen Ideen legte er auch der Betrachtung zugrunde, wie man eine Turing-ähnliche Verhaltensweise erzielen kann.

Diese einfachen Operationen von Deutsch werden Quantengatter genannt, da sie eine analoge Rolle zu den binären logischen Gattern des klassischen Computers einnehmen. Verschiedene Wissenschaftler untersuchten im Anschluss daran die kleinste Klasse von Gattern, die für den Quantencomputer notwendig ist. Es zeigte sich, dass Deutschs Simulator im strikten Sinne nicht universell ist. Trotzdem ist seine Idee insofern effektiv, als dass man auf diese Weise

eine große Klasse von Quantensystemen simulieren kann [44]. Auch in anderer Hinsicht ist die Arbeit von Deutsch hervorzuheben: Sie führte Konzepte für Quantennetzwerke [45] und logische Gatter ein, welche für den Quantencomputer sehr wichtig sind und es überhaupt erst ermöglichen, sich weitere Gedanken über die Quantenrechner zu machen.
Auf der Grundlage des Modells der Quanten-Turing-Maschine war man nun in der Lage, die Fähigkeiten des Quantencomputers zu untersuchen. Diese Untersuchungen beziehen sich in erster Linie auf die Berechenbarkeit, d.h. auf die Art der mit dieser Maschine zu lösenden Probleme, den damit verbundenen Aufwand, d.h. dem Verhältnis von Arbeitsspeicher und Zeitaufwand zur Größe des Problems, sowie auf die Allgemeingültigkeit, d.h. die Frage, ob eine Maschine alle anderen effizient simulieren kann. Im Folgenden werden diese einzelnen Aspekte genauer dargelegt.
Für die Betrachtung der Allgemeingültigkeit sei zunächst die Church-Turing-These vorangestellt: “Jede (im intuitiven Sinn) berechenbare Funktion ist auch Turing-berechenbar.” Feynman stellte 1982 jedoch fest, dass eine klassische Turing-Maschine die Quantenmechanik nicht effizient simulieren kann.
Die Diskrepanz zwischen den Aussagen von Church-Turing und Feynman führte zu einer Umformulierung der Church-Turing These durch David Deutsch [43]: “Jedes endlich realisierbare physikalische System kann im endlichen Sinn perfekt durch eine universelle Computermaschine simuliert werden.” Diese Aussage kann nur dann mit Feynmans Beobachtung über die Effektivität von Quantensystemsimulationen in Einklang gebracht werden, wenn man das Modell einer Computermaschine auf die Quantenmechanik stützt.
Den Aspekt der Berechenbarkeit muss man unter folgendem Blickwinkel betrachten: Die Computertheorie befasst sich damit, welche Probleme in endlicher Zeit auf einem Computer gelöst werden können. Wenn es im Hinblick auf das betrachtete Computermodell keinen Algorithmus gibt, der garantiert, in endlicher Zeit eine Antwort für das gegebene Problem zu finden, gilt das Problem für dieses Computermodell als unberechenbar. In Bezug auf den Quantencomputer bewies wiederum Deutsch [43], dass der Quantencomputer verschiedene Outputs berechnen kann, deren Berechnung durch deterministische Turing-Maschinen nicht möglich ist, da klassische deterministische Turing-Maschinen darauf beschränkt sind Funktionen, d.h. mathematische Prozeduren, zu berechnen, die ein einzelnes reproduzierbares Ergebnis besitzen. Es gibt aber durchaus Probleme, die nicht durch die Anwendung einer Funktion lösbar sind, wie z.B. das Erzeugen von echten Zufallsvariablen. Daher kann eine Turing-Maschine das Erzeugen von Zufallsvariablen nur vortäuschen.
Während sich die Berechenbarkeit damit befasst, welche Probleme mit einem Computer lösbar bzw. unlösbar sind, geht es bei der Betrachtung des Aufwands darum, wie effizient der Computer die Probleme lösen kann. Die Effizienz ist eine wichtige Frage in der Informatik. Die Tatsache, dass ein Computer ein Problem theoretisch lösen kann, garantiert nicht dessen Lösbarkeit mit der heutigen Technologie in der Praxis. Die Lösung von sehr komplexen Problemen kann durch zu hohen Bedarf an Laufzeit oder Speicherplatz auf klassischen Computern nicht praktikabel sein. Informatiker haben ein Klassifikationsschema für die Beschreibung des Aufwandes von verschiedenen realisierbaren Algorithmen auf unterschiedlichen Computertypen entwickelt. Das größte gemeinsame Maß der Effizienz zeichnet sich durch das Verhältnis der Komplexität des Problems zur Größe der Zeit oder des Speicherplatzes, die zur Lösung eines Problems benötigt werden, aus. Vereinfacht gesagt ist die Größe des Speicherplatzes die Anzahl der benötigten Bits, um in dem Computer das Problem anzugeben. Die Klassifizierung wurde entwickelt, um die Schwierigkeit eines Problems quantitativ zu bestimmen. Diese Einteilung

basiert auf der mathematischen Form einer Funktion, die die Vergrößerung des rechnerischen Aufwands in Abhängigkeit von der Größe des Problems beschreibt. Der größte quantitative Unterschied liegt zwischen einem polynomial wachsenden Aufwand (diese Probleme werden als lösbar erachtet) und einem exponentiell wachsenden Aufwand (diese Probleme werden als nicht lösbar erachtet).
Zur Verdeutlichung betrachte man die Multiplikation zweier großer Zahlen $p \cdot q = N$ und das Faktorisieren der Zahl N. Es ist relativ leicht, zwei große Zahlen p und q miteinander zu multiplizieren, wogegen die Umkehroperation, das Finden der Faktoren von N, sich weitaus schwieriger gestaltet:

10433	$\cdot$	16453	$=$	$?$	leicht	(polynomial)
$?$	$\cdot$	$?$	$=$	171654149	schwer	(exponentiell)

Wenn in binärer Notation die zu multiplizierenden Zahlen eine Größe von L Bits haben, dann kann die Multiplikation in einer Zeit proportional zu L^2, also polynomial in L, durchgeführt werden. Für die Faktorisierung sind die besten bekannten klassischen Methoden für Zahlen mit ungefähr 100-150 Dezimalstellen das mehrfache polynomiale quadratische Sieb [46], sowie das Zahlfeldsieb [47, 48] für Zahlen mit mehr als 110 Dezimalstellen. Die Laufzeit dieser Algorithmen wächst stärker als polynomial mit L, der Anzahl von Bits, die benötigt werden, um die zu faktorisierende Zahl N anzugeben ($L \approx \log_2 N$): Der beste Faktorisierungsalgorithmus von Zahlen erfordert eine Zeit der Ordnung $\exp(L^{\frac{1}{3}} \log(L)^{\frac{2}{3}})$ (dazu mehr in Kapitel 6).
Mitte der 1980er Jahre ließ das Interesse am Quantencomputer zunächst nach. Das lag daran, dass sich kein mathematisches Problem fand, das mit einem Quantenrechner besser oder schneller zu lösen wäre, als mit einem klassischen Computer.
In den frühen 1990er Jahren wurde die Suche nach solchen Problemen von verschiedenen Wissenschaftlern wie Deutsch und Jozsa [49, 50], Berthiaume und Brassard [51, 52], Bernstein und Vazirani [53] erneut aufgenommen. In dieser Hinsicht gelang Simon mit dem Simonalgorithmus [54] und Peter W. Shor von den AT & T-Bell-Laboratorien in Murray Hill (New Jersey) mit dem Shoralgorithmus [49, 55, 56, 57, 58, 14, 59, 60, 61] der entscheidende Durchbruch. Shor diskutierte u.a. die Faktorisierung großer Zahlen unter Benutzung von Quantenfouriertransformationen. Die Methode der Anwendung von Quantenfouriertransformationen ging von Coppersmith [62] und Deutsch aus (dazu auch [63]). Weitere wichtige Quantenalgorithmen wurden von Grover [64, 65, 66, 14, 67], Durr und Hoyer [68] und Kitaev [69] vorgestellt. 1991 beschrieb Jozsa zusätzlich noch eine Klasse von Funktionen, die nicht mit Hilfe des Quantenparallelismus berechnet werden können [70].
Theoretisch sind in den letzten Jahren durch die Entwicklung von Quantenalgorithmen und der Quantenfehlerkorrektur große Fortschritte erzielt worden. Dagegen stehen die experimentelle Realisierung und der Einbezug der theoretischen Konzepte in die Praxis erst am Anfang, wenn auch mit Quantengattern und der Teleportation schon erfolgreiche Laborergebnisse erzielt wurden. Der entscheidende Schritt zum skalierbaren Quantencomputer mit einem größeren Rechenumfang ist nach heutigem Stand noch nicht absehbar.

Kapitel 5

Quantenmechanik und Computer

Wie kann nun unter Benutzung der Gesetze der Quantenmechanik ein Computer gebildet werden? Man stellt einen Hamiltonoperator für ein System wechselwirkender Teilchen auf, das sich wie ein Gesamtsystem verhält, welches einem universellen Computer dient. Der Hamiltonoperator soll alle internen Rechenoperationen im Detail beschreiben, nicht aber die Wechselwirkungen mit den äußeren Einflüssen beinhalten, wie das Einlesen des Inputs und Lesen des Outputs.
Man betrachtet als Prototyp ein Zweizustandssystem, also z.B. das Spin-$\frac{1}{2}$-System eines Elektrons. Jedes Bit wird durch einen der beiden Zustände $|0\rangle$ oder $|1\rangle$ dargestellt. [1] Der Anschaulichkeit halber soll das Beispiel des FULLADDERs (siehe Kapitel 1.2), in dem die Eingangsleitungen durch $|a\rangle, |b\rangle, |c\rangle, |d\rangle$ und die Ausgangsleitungen durch $|a'\rangle, |b'\rangle, |c'\rangle, |d'\rangle$ gegeben sind, nochmals aufgegriffen werden. In der Quantenmechanik ist die Operation, die $|a\rangle, |b\rangle, |c\rangle, |d\rangle$ reversibel in $|a'\rangle, |b'\rangle, |c'\rangle, |d'\rangle$ überführt, ein unitärer Operator, d.h. ein Operator U für den $U^\dagger U = \mathbb{1}$ gilt. Für den FULLADDER bedeutet dies, dass der Operator M, der aus einzelnen Operationen besteht, den Anfangszustand $|\psi_{in}\rangle$ in den Ausgangszustand $|\psi_{out}\rangle = M|\psi_{in}\rangle$ überführt. Die Matrix M besteht aus den Elementen NOT, CONTROLLED NOT und CONTROLLED CONTROLLED NOT. Für die einfachste Operation, das NOT, ist der zugehörige Operator auf den Zustand $|a\rangle$ in der Ein-Teilchen-Basis von der Gestalt

$$A_a = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Diesen Operator kann man mit Hilfe des Aufsteige- und Absteigeoperators darstellen. [2] Um zu verdeutlichen, dass dieser Operator im vorliegenden Fall auf die einzelne Leitung $|a\rangle$ wirkt,

[1] z.B. $|0\rangle = |\downarrow\rangle, |1\rangle = |\uparrow\rangle$,
oder bei jedem anderen Zweizustandssystem: $|0\rangle = |\text{Grundzustand}\rangle, |1\rangle = |\text{angeregter Zustand}\rangle$

[2] Die Aufsteige- und Absteigeoperatoren sind von der allgemeinen Gestalt: $S_+ := \hbar|\uparrow\rangle\langle\downarrow|, S_- := \hbar|\downarrow\rangle\langle\uparrow|$. Auf die Zustände $|\uparrow\rangle, |\downarrow\rangle$ angewendet erhält man: $S_+|\uparrow\rangle = 0, S_-|\uparrow\rangle = \hbar|\downarrow\rangle, S_+|\downarrow\rangle = \hbar|\uparrow\rangle$ und $S_-|\downarrow\rangle = 0$. Die physikalische Interpretation von S_+ (Aufsteigeoperator) besagt folglich, dass S_+ die Spinkomponente um eine Einheit $\hbar$ erhöht. Wenn die Spinkomponente nicht mehr erhöht werden kann, erhält man den Nullzustand. Für den Absteigeoperator S_- gilt das entsprechende. In der Matrixdarstellung sind S_+ und S_- von der Form:

$$S_+ = \hbar \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad S_- = \hbar \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}.$$

Für das Aufstellen des Operators M wird im Folgenden der Faktor $\hbar$ vernachlässigt.

werden S_+ und S_- (siehe Fußnote 2) in a_+ und a_- umbenannt. In dieser Darstellung ergibt sich die Matrix A_a für das NOT zu

$$A_a = a_+ + a_- = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Da der Operator A_a unitär ist, d.h. $A_a^\dagger A_a = \mathbb{1}$ gilt, liegt mit NOT ein reversibles Element vor. In der gleichen Weise erhält man die Matrix $A_{a,b}$ in der Zwei-Teilchen-Basis für das CONTROLLED NOT:

$$A_{a,b} = a_- a_+ (b_+ + b_-) + a_+ a_- = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Im ersten Term $a_- a_+ (b_+ + b_-)$ wird durch $a_- a_+$ überprüft ob $|a\rangle = |1\rangle$ gilt. Wenn dies der Fall ist, führt $b_+ + b_-$ das NOT auf $|b\rangle$ aus. Der zweite Term $a_+ a_-$ liest aus, ob $|a\rangle = |0\rangle$ gilt und wendet dann auf $|b\rangle$ die Identität an. Die Matrix $A_{a,b}$ kann auch über die Vorschrift

$$A_{a,b} = \mathbb{1} + a_- a_+ (b_+ + b_- - \mathbb{1})$$

erhalten werden. Dabei bewirkt die Einheitsmatrix keine Veränderung der Werte der Eingangsleitungen. Für den Fall, dass $|a\rangle = |1\rangle$ gilt, wird dies korrigiert, indem ein NOT ausgeführt wird, anstatt $|b\rangle$ unverändert zu lassen. Dieser Operator ist ebenfalls unitär, so dass das CONTROLLED NOT auch ein reversibles Element ist.

Der Operator $A_{ab,c}$ für das CONTROLLED CONTROLLED NOT ist von der Gestalt

$$\begin{aligned} A_{ab,c} &= a_- a_+ b_- b_+ (c_+ + c_-) + a_- a_+ b_- b_+ \\ &= \mathbb{1} + a_- a_+ b_- b_+ (c_+ + c_- - \mathbb{1}) \\ &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \end{aligned}$$

Auch hier gilt die Unitarität von $A_{ab,c}$ und die daraus folgende Reversibilität des CONTROLLED CONTROLLED NOTs.
Das Gatter des FULLADDERs besteht aus fünf Operationen und somit erhält man die Matrix M aus dem Produkt der einzelnen Operatoren der notwendigen Operationen [3]:

$$M = A_{a,b} A_{b,c} A_{bc,d} A_{a,b} A_{ab,d}.$$

Da diese Matrix aus einem Produkt einzelner unitärer Matrizen besteht, gilt die Unitaritätsbedingung. M ist somit eine reversible Operation, und $M^\dagger$ führt $|\psi_{out}\rangle$ in $|\psi_{in}\rangle$ über. Zu beachten

[3] Mit $A_{a,b} \equiv A_{a,b} \otimes \mathbb{1}_c : \mathbb{C}^8 \to \mathbb{C}^8$

ist hier, dass man die Anwendung der einzelnen Operationen in der Matrixschreibweise von rechts nach links ausliest, im Gegensatz zu den schematischen Darstellungen wie in Abbildung (1.5).
Verallgemeinert stellt sich folgendes Problem: Sei $A_1, A_2, A_3, \ldots, A_k$ eine Folge von Operatoren, die auf n Leitungen wirken. Die notwendige $2^n \times 2^n$-Matrix M, für die $|\psi_{out}\rangle = M|\psi_{in}\rangle$ gilt, ist das Produkt $M = A_k \cdots A_3 A_2 A_1$. Wie implementiert man M experimentell?
In der Quantenmechanik ist der Zustand zum Zeitpunkt t in einem System mit Hamiltonoperator H gegeben durch $\mathrm{e}^{-\frac{i}{\hbar}Ht}\psi_{in}$, mit dem Anfangszustand ψ_{in}. Die Schwierigkeit besteht darin, für eine gegebene Zeit t den Hamiltonoperator H zu finden, der $M = \mathrm{e}^{-\frac{i}{\hbar}Ht}$ erzeugt, wenn M ein solches Produkt von nichtvertauschbaren Matrizen ist. Die Entwicklung von $\mathrm{e}^{-\frac{i}{\hbar}Ht}$ lautet

$$\mathrm{e}^{-\frac{i}{\hbar}Ht} = \sum_{l=0}^{n} \left(-\frac{i}{\hbar}Ht\right)^l \cdot \frac{1}{l!} = 1 - i\frac{Ht}{\hbar} - \frac{H^2t^2}{2\hbar^2} + i\frac{H^3t^3}{6\hbar^3} + \frac{H^4t^4}{24\hbar^4} - \ldots.$$

Es wird deutlich, dass H beliebig oft wirkt und so der Gesamtzustand als Superposition dieser Möglichkeiten gegeben ist. Dies führt zu dem Ansatz, die Matrix M der einzelnen Operatoren A zu finden. Man addiert zu dem Rechenregister, das aus n Zweizustandssystemen besteht, eine völlig neue Folge von $k+1$ Zweizustandssystemen, das Zählregister. Seien S_{+_i} und S_{-_i} der Aufsteige- und Absteigeoperator für die Stelle i im Zählregister mit $0 \leq i \leq k$. Wenn die Stelle i besetzt ist, lautet der zugehörige Zustand $|1\rangle$, wenn die Stelle unbesetzt ist, $|0\rangle$. Der Hamiltonoperator H ergibt sich zu

$$\begin{aligned}
H &= \sum_{i=0}^{k-1} S_{+_{i+1}} S_{-_i} A_{i+1} + \text{hermitesch konjugiert} \\
&= \sum_{i=0}^{k-1} S_{+_{i+1}} S_{-_i} A_{i+1} + \left(S_{+_{i+1}} S_{-_i} A_{i+1}\right)^\dagger \\
&= S_{+_1} S_{-_0} A_1 + S_{+_2} S_{-_1} A_2 + S_{+_3} S_{-_2} A_3 + \ldots + S_{+_k} S_{-_{k-1}} A_k \\
&+ S_{+_0} S_{-_1} A_1^\dagger + S_{+_1} S_{-_2} A_2^\dagger + S_{+_2} S_{-_3} A_3^\dagger + \ldots + S_{+_{k-1}} S_{-_k} A_k^\dagger.
\end{aligned}$$

Unter Berücksichtigung dieser Entwicklung betrachtet man den Fall, dass sich ein Zweizustandssystem im Zustand $|1\rangle$, die restlichen Zweizustandssysteme im Zustand $|0\rangle$ befinden. Dann bleibt auch nach der Anwendung des Hamiltonoperators H nur ein Zustand besetzt. Die Anzahl der besetzten Zustände ist eine erhaltene Größe, da der Besetzungszahloperator N mit dem Hamiltonoperator kommutiert. Da sich während einer herkömmlichen Operation nie zwei oder mehr Zweizustandssysteme im Zustand $|1\rangle$ befinden, nimmt man im Folgenden an, dass in der Operation des Computers immer nur eine Zählregisterstelle besetzt ist. Der Anfangszustand des Systems aus Rechen- und Zählregister sei von der Gestalt, dass sich das Rechenregister in $|\psi_{in}\rangle$ befindet und die Zählregisterstelle $i = 0$ besetzt ist, alle anderen Stellen aber unbesetzt sind, also $|1\rangle|0\rangle \ldots |0\rangle$. Wenn sich nach einiger Zeit die Stelle k des Zählregisters im Zustand $|1\rangle$ befindet und damit alle anderen im Zustand $|0\rangle$, liegt die Vermutung nahe, dass auf das Rechenregister der Größe n die Matrix $M = A_k \cdots A_2 A_1$ angewandt wurde. Dies bestätigt sich folgendermaßen: Angenommen man startet mit einem beliebigen Anfangszustand $|\psi_{in}\rangle$ des Rechenregisters, und die Zählregisterstelle $i = 0$ ist besetzt. Dann ist der einzige Term des Hamiltonoperators, der auf dieses System zunächst wirken kann, der erste Term $S_{+_1} S_{-_0} A_1$. Der Operator S_{-_0} verwandelt die Zählregisterstelle $i = 0$ von einem besetzten Zustand in

einen unbesetzten. Der Operator S_{+_1} hingegen verwandelt die Zählregisterstelle $i = 1$ von einem unbesetzten Zustand in einen besetzten. Somit verschiebt die Operation $S_{+_1}S_{-_0}A_1$ den Zustand $|1\rangle$ von der Position $i = 0$ auf die Position $i = 1$ des Zählregisters mit zusätzlicher Anwendung der Matrix A_1: Hierbei wirkt der Operator A_1 nur auf das Rechenregister, so dass der Anfangszustand $|\psi_{in}\rangle$ mit A_1 multipliziert wird. Wendet man den Hamiltonoperator wiederum auf das so erhaltene System an, wirkt nur der zweite Term $S_{+_2}S_{-_1}A_2$, der den besetzten Zustand von $i = 1$ nach $i = 2$ verschiebt und die zusätzliche Multiplikation von A_2 auf das Rechenregister ausführt. Somit wurde nun insgesamt die Operation A_2A_1 auf den Anfangszustand $|\psi_{in}\rangle$ dieses Registers angewandt. Auf diese Art verschiebt sich nach Durchlaufen der ersten Zeile des Hamiltonoperators einerseits der besetzte Zustand $|1\rangle$ von der Stelle $i = 0$ zu $i = k$ im Zählregister, und andererseits werden auf das Rechenregister der Größe n nacheinander die Operationen $A_k \dots A_2A_1$ angewandt, die insgesamt die Operation M ergeben, wobei es gleichgültig ist wie dieser Zustand erreicht wird:
Der Hamiltonoperator muss hermitesch sein und somit die transponiert komplex konjugierten aller Operationen enthalten. Mit diesen Operationen kann man die bisher ausgeübten rückgängig machen. Angenommen man befindet sich im Programm an der Stelle $i = 2$ des Zählregisters, an der auf das Rechenregister die Operation A_2A_1 ausgeführt wurde. Durch den Term $S_{+_1}S_{-_2}A_2^\dagger$ wird der besetzte Zustand von $i = 2$ nach $i = 1$ verschoben, und auf das Rechenregister wird die Operation $A_2^\dagger A_2A_1$ ausgeübt. Da $A_2^\dagger A_2 = \mathbb{1}$ gilt, wirkt also nur noch A_1 auf $|\psi_{in}\rangle$. Somit kann man sich durch Anwendung der einzelnen Terme des Hamiltonoperators sowohl vorwärts als auch rückwärts durch das Programm bewegen. Befindet man sich z.B. an der Stelle $i = j$, haben die Matrizen A_1 bis A_j auf das Rechenregister gewirkt. Dabei macht es keinen Unterschied, ob man auf direktem Wege bis zur Stelle $i = j$ gelangt ist, oder ob man bis dorthin sowohl vorwärts als auch rückwärts operiert hat.
Auf diese Weise wurde von Feynman argumentiert, dass die Gesetze der Quantenmechanik zur Realisierung eines Computers benutzt werden können.

Kapitel 6

Der Shor–Algorithmus

Bereits vor Euklid (um 300 v. Chr.) war die Existenz einer Zerlegung jeder Zahl $N \in \mathbb{N}$ in ein Produkt von Primzahlen bekannt, aber die erste klare Formulierung mit Beweis scheint von C. F. Gauß (1777-1855) in den “Disquistiones Arithmeticae” (Art. 16) gegeben worden zu sein. Jedoch mangelt es bis in die heutige Zeit an einem effizienten Verfahren, diese Zerlegung zu finden. Erst um 1970, als die Anwendung von Paradigmen der theoretischen Informatik auf die Zahlentheorie vollzogen wurden, erfolgte der entscheidende Durchbruch, da diese Verknüpfung erstmalig zu einer höheren Effizienz bei den Faktorisierungsalgorithmen führte. Der effizienteste klassische Algorithmus, der heute bekannt ist, ist der von Lenstra und Lenstra [47, 48], der eine zum Input exponentielle Laufzeit benötigt ($\sim \exp(c(\log(N))^{\frac{1}{3}}(\log(\log(N))^{\frac{2}{3}}))$ für $c =$ konstant). Der sehr hohe zeitliche Aufwand für die Faktorisierung von Zahlen und eine nicht erkennbare Aussicht auf effizientere Algorithmen führten schließlich zur Anwendung bei Verschlüsselungssystemen (z.B. der RSA-Algorithmus [71]).
1994 fand Peter Shor einen Algorithmus [55, 56, 57, 58, 14, 59, 60, 61], der es mit Hilfe des Quantencomputers möglich macht, eine Zahl N in polynomialer Zeit zu faktorisieren. Diese Methode hatte so große Auswirkungen auf die wissenschaftlichen Bereiche der Mathematik, Physik und Informatik, dass ein breites Interesse an dem Quantencomputer entstand.
Den meisten Faktorisierungsalgorithmen, einschließlich dem von Shor, liegt eine Standardreduktion des Faktorisierungsproblems auf das Problem, die Periode einer Funktion zu finden, zugrunde. Shor benutzt dazu den Quantenparallelismus, um eine Superposition aller Werte einer Funktion in einem Schritt zu erhalten, was bedeutet, dass er die diskrete Fouriertransformation einer Funktion berechnet, die wie klassische Fouriertransformationen alle Amplituden der Funktion in Vielfache der Reziproken der Periode umwandelt. Bei Wiederholung liefert die Messung des Zustands mit gewünschter Wahrscheinlichkeit die Periode, was es ermöglicht, eine Zahl N zu faktorisieren. Das Finden der Periode ist allerdings nicht ganz so einfach, da die diskrete Fouriertransformation in den meisten Fällen nur approximative Werte liefert. Die Techniken für das Erhalten der Periode sind jedoch klassisch.
Im Folgenden soll der Shor–Algorithmus erläutert werden; um das Wesen des Algorithmus zu beleuchten, wird mit dem Algorithmus, der sich rein auf zahlentheoretische Grundlagen stützt, begonnen. Im Anschluss daran wird zunächst die Diskrete Fouriertransformation erläutert um den Quanten–Shor–Algorithmus zuerst theoretisch betrachten zu können und dann mit Zahlenbeispielen zu schließen.

6.1 Der klassische Shor–Algorithmus

Mit Hilfe des Shor–Algorithmus kann man eine beliebige Zahl N in ihre Primfaktoren zerlegen.

Zu einem gegebenen N wähle man eine dazu beliebige teilerfremde Zahl y, also $\mathsf{ggT}(y, N) = 1$ ($\mathsf{ggT}(y, N)$ steht für den größten gemeinsamen Teiler von y und N). Nun muss die Periode oder auch Ordnung r der Funktion $F_N(a) = y^a \pmod{N}$ gefunden werden. Dies bedeutet, dass man sich die Reste von y^a bezüglich der Division durch N anschauen muss. Beträgt das Ergebnis $y^r \equiv 1 \pmod{N}$ (siehe auch Anhang A.1), so hat man die Periode r gefunden. Dies ist der Hauptbestandteil des Algorithmus, da dieser zu einer nichttrivialen Lösung $x = y^{r/2}$ der quadratischen Gleichung $x^2 \equiv 1 \pmod{N}$ mit den gesuchten Faktoren von N führt [1] (zur Erläuterung siehe Exkurs 1).

Exkurs 1: Zusammenhang zwischen der quadratischen Gleichung $x^2 \equiv 1 \pmod{N}$ und der Primfaktorzerlegung von N

Betrachtet wird die quadratische Gleichung $x^2 \equiv 1 \pmod{N}$. Diese besitzt stets die trivialen Lösungen $x = \pm 1$, die die einzigen Lösungen sind, sofern N eine ungerade Primzahl ist. Dies wird wie folgt deutlich: Sei $(\mathbb{Z}/_{p\mathbb{Z}}, +, \cdot)$ ein Körper für alle Primzahlen p, also der Restklassenkörper $\pmod{p}$. In diesem Körper ist das multiplikativ inverse Element eindeutig bestimmt und 1 das neutrale Element der Multiplikation. Das bedeutet, dass mit der quadratischen Gleichung nach jenen x gesucht wird, die zu sich selbst invers sind.

$$\begin{aligned} x^2 &\equiv 1 \pmod{p} \\ (x^2 - 1) &\equiv 0 \pmod{p} \\ (x-1)(x+1) &\equiv 0 \pmod{p} \end{aligned} \tag{6.1}$$

In den Restklassenkörpern $\pmod{p}$ sind $[1]_p$ und $[p-1]_p = [-1]_p$ die einzigen selbstinversen Elemente (dabei bedeutet $[a]_p$ Restklasse $\pmod{p}$). Damit sind die trivialen Lösungen $x = 1$ oder $x = -1$ die einzigen Lösungen der Gleichung (6.1). Dies veranschaulicht auch das folgende Zahlenbeispiel: $R_5 = \{[0], [1], [2], [3], [4]\}$ (siehe auch Anhang A.1)

$+$	$[0]$	$[1]$	$[2]$	$[3]$	$[4]$
$[0]$	$[0]$	$[1]$	$[2]$	$[3]$	$[4]$
$[1]$	$[1]$	$[2]$	$[3]$	$[4]$	$[0]$
$[2]$	$[2]$	$[3]$	$[4]$	$[0]$	$[1]$
$[3]$	$[3]$	$[4]$	$[0]$	$[1]$	$[2]$
$[4]$	$[4]$	$[0]$	$[1]$	$[2]$	$[3]$

$\cdot$	$[0]$	$[1]$	$[2]$	$[3]$	$[4]$
$[0]$	$[0]$	$[0]$	$[0]$	$[0]$	$[0]$
$[1]$	$[0]$	$[1]$	$[2]$	$[3]$	$[4]$
$[2]$	$[0]$	$[2]$	$[4]$	$[1]$	$[3]$
$[3]$	$[0]$	$[3]$	$[1]$	$[4]$	$[2]$
$[4]$	$[0]$	$[4]$	$[3]$	$[2]$	$[1]$

Wenn N aber zusammengesetzt ist, d.h. in Faktoren $p, q \in \mathbb{N}$ zerlegbar ist, existieren noch zusätzlich sogenannte nichttriviale Lösungen $x \equiv \pm a \pmod{N}$. Sei $N = n_1 \cdot n_2$ mit $\mathsf{ggT}(n_1, n_2) = 1$, dann gibt es folgende vier Sätze von Lösungen:

$$\begin{aligned} &\text{a)} \quad \begin{cases} x_1 \equiv +1 \pmod{n_1} \\ x_1 \equiv +1 \pmod{n_2} \end{cases} && \text{b)} \quad \begin{cases} x_2 \equiv -1 \pmod{n_1} \\ x_2 \equiv -1 \pmod{n_2} \end{cases} \\ &\text{c)} \quad \begin{cases} x_3 \equiv +1 \pmod{n_1} \\ x_3 \equiv -1 \pmod{n_2} \end{cases} && \text{d)} \quad \begin{cases} x_4 \equiv -1 \pmod{n_1} \\ x_4 \equiv +1 \pmod{n_2} \end{cases} . \end{aligned} \tag{6.2}$$

[1] Im Folgenden werden der Vollständigkeit halber in den Exkursen Zusatzinformationen angeboten. Diese Zusatzinformationen sind für das Durcharbeiten des Inhalts nicht zwingend notwendig, sondern dienen dem interessierten Leser zur Vertiefung.

In allen Fällen gilt $x_i^2 \equiv 1 \pmod{n_1}$ und $\pmod{n_2}$ und damit erfüllt jedes x_i auch die Gleichung $x^2 \equiv 1 \pmod{N}$. Laut dem Chinesischen Restesatz (Anhang A.3) hat jeder Satz eine eindeutige Lösung $\pmod{N}$. Für die Fälle a) und b) erhält man $x_1 \equiv 1$ und $x_2 \equiv -1 \pmod{N}$, die trivialen Lösungen der Gleichung $x^2 \equiv 1 \pmod{N}$. Für die Fälle c) und d) ergibt sich $x_3 \equiv a$ und $x_4 \equiv -a \pmod{N}$, ein Paar nichttrivialer Lösungen. Also gilt $(a+1)(a-1) \equiv 0 \pmod{N}$ mit $a \pm 1$ ungleich Null. Damit ist N ein Teiler von $(a+1)(a-1)$, aber kein Teiler von $a \pm 1$ (mit $a \pm 1 \leq N+1$). Somit ist der $\mathsf{ggT}(N, a \pm 1)$ ein nichttrivialer Faktor von N (für $a \neq \pm 1$), der über den Euklidischen Algorithmus (Anhang A.2) gefunden wird. Dieser letzte Schritt für die Bestimmung der Faktoren von N ist in polynomialer Zeit durchführbar.
Das Erhalten von Lösungen mit Hilfe der vier Sätze von Gleichungen soll nun anhand eines Zahlenbeispiels vorgeführt werden: Es sei $x^2 \equiv 1 \pmod{35}$. Dann ergeben sich in Anlehnung an Gleichung (6.2) folgende Sätze:

$$\text{a)} \quad \begin{cases} x_1 \equiv +1 \pmod 5 \\ x_1 \equiv +1 \pmod 7 \end{cases} \qquad \text{b)} \quad \begin{cases} x_2 \equiv -1 \pmod 5 \\ x_2 \equiv -1 \pmod 7 \end{cases}$$

$$\text{c)} \quad \begin{cases} x_3 \equiv +1 \pmod 5 \\ x_3 \equiv -1 \pmod 7 \end{cases} \qquad \text{d)} \quad \begin{cases} x_4 \equiv -1 \pmod 5 \\ x_4 \equiv +1 \pmod 7 \end{cases} .$$

Für die Fälle a) und b) ergeben sich wie erwartet die Lösungen $x_1 \equiv 1$ und $x_2 \equiv -1 \pmod{35}$. Hingegen erhält man für c) $x_3 \equiv 6 \pmod{35}$ bzw. für d) $x_4 \equiv 29 \equiv -6 \pmod{35}$. Damit gilt $(6+1)(6-1) \equiv 35 \equiv 0 \pmod{35}$, bzw. $(-6+1)(-6-1) \equiv 35 \equiv 0 \pmod{35}$, aber 35 ist weder Teiler von 6 noch von -6. Damit hat man ein Paar nichttrivialer Lösungen der quadratischen Gleichung $x^2 \equiv 1 \pmod{35}$ gefunden.

Für den Shor–Algorithmus berechnet man also zu dem gewählten y mit $\mathsf{ggT}(N, y) = 1$ zunächst die Ordnung r. Wenn r eine gerade Zahl ist, setze man $x = y^{r/2}$, eine nichttriviale Lösung der quadratischen Gleichung $x^2 \equiv 1 \pmod{N}$. Zum Schluss bleiben der $\mathsf{ggT}(x-1, N) = p$ und der $\mathsf{ggT}(x+1, N) = q$, die die jeweiligen Faktoren p und q von $N = p \cdot q$ ergeben, zu berechnen.
Der beschriebene Algorithmus muss nicht zum Erfolg führen. Dies ist der Fall, wenn das gewählte y eine ungerade Ordnung r besitzt, da dann nicht notwendig $y^{r/2} \in \mathbb{N}$ gilt. Außerdem liefert das Verfahren keine verwendbare Lösung, wenn r gerade ist, aber $y^{r/2}$ eine triviale Lösung von $x^2 \equiv 1 \pmod{N}$ ist.

Nun soll der Shor–Algorithmus an einem konkreten Zahlenbeispiel vorgeführt werden. Man betrachte die zu faktorisierende Zahl $N = 21$. Wähle y so, dass der $\mathsf{ggT}(N, y) = 1$ ist, d.h. $y \in \{2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$. Hier sei $y = 5$ und die zugehörige Ordnung von $5 \pmod{21}$ erhält man aus

$$\begin{aligned}
5^1 &\equiv 5 \pmod{21} \\
5^2 &\equiv 4 \pmod{21} \\
5^3 &\equiv -1 \pmod{21} \\
5^4 &\equiv 16 \pmod{21} \\
5^5 &\equiv -4 \pmod{21} \\
5^6 &\equiv 1 \pmod{21}
\end{aligned}$$

Damit ergibt sich $r = 6$. Da r gerade ist, gilt $x = 5^{6/2} = 5^3 \equiv -1 \pmod{21}$. Dies ist jedoch eine triviale Lösung der quadratischen Gleichung $x^2 \equiv 1 \pmod{21}$. Somit scheitert der Algorithmus.
Wähle nun $y = 2$. Die zugehörende Ordnung ist $r = 6$ und man erhält $x = 8$. Die Faktoren von $N = 21$ ergeben sich aus

$$\begin{array}{lrll} \text{ggT}(7,21): & 21 & = 3 \cdot 7 + 0, & \text{also p=7} \\ \text{ggT}(9,21): & 21 & = 2 \cdot 9 + 3, & \\ & 9 & = 3 \cdot 3 + 0, & \text{also q=3.} \end{array}$$

Die Zahl $N = 21$ lässt sich daher in $21 = 3 \cdot 7$ faktorisieren.

6.2 Die diskrete Fouriertransformation

Um den Quantenalgorithmus von Shor zur Faktorisierung einer Zahl N zu erläutern, sei hier zunächst ein kleiner Überblick über die diskreten Fouriertransformationen im Zusammenhang mit unitären Transformationen gegeben. Ein Zahlenbeispiel, das diesen Zusammenhang illustriert befindet sich im Anhang B
Die diskrete Fouriertransformation $\pmod{q}$, im Folgenden durch DFT_q abgekürzt, ist eine unitäre Transformation in q Dimensionen. Sie hat bezüglich einer gewählten Basis $|0\rangle, |1\rangle, \ldots,$ $|q-1\rangle$ die Form

$$DFT_q|a\rangle = \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \mathrm{e}^{\frac{i2\pi ac}{q}} |c\rangle. \tag{6.3}$$

Insbesondere bedeutet dies, dass der Zustand $|0\rangle$ in eine gleichförmige Superposition aller $c \pmod{q}$ transformiert wird oder allgemeiner ausgedrückt, dass die DFT_q eine diskrete Fouriertransformation der Inputamplituden durchführt.
Für den Algorithmus muss man die DFT_q für $q \approx N^2$ anwenden, wobei N die zu faktorisierende Zahl ist und q die Form $q = 2^L$ mit $L \in \mathbb{N}$ hat. In diesem Fall kann der effiziente Quantenalgorithmus in Anlehnung an die Fast-Fouriertransformation konstruiert und in Termen von unitären Operationen ausgedrückt werden, was nun gezeigt werden soll. Im Folgenden sei $q = 2^L$ und a eine Zahl, die in binärer Darstellung von der Form $|a_{L-1}\rangle|a_{L-2}\rangle \ldots |a_0\rangle$ ist. Um die DFT_{2^L} zu konstruieren, benötigt man nur zwei grundlegende unitäre Operationen. Die eine ist die schon bekannte Hadamartransformation (Gleichung (1.1)), die auf das j-te Qubit angewandt wird und gegeben ist durch:

$$H_j = \frac{1}{\sqrt{2}} \begin{array}{c} \begin{array}{cc} |0\rangle & |1\rangle \end{array} \\ \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{array} \begin{array}{c} |0\rangle \\ |1\rangle \end{array} . \tag{6.4}$$

Die andere Operation führt eine Zwei-Bittransformation der Qubits in den Positionen j und k, mit $j < k$ durch, wenn sich diese im Zustand 1 befinden, unabhängig von den Zuständen der anderen Qubits (vgl. Gleichung (1.4)):

$$S_{jk} = \begin{array}{c} \begin{array}{cccc} |0\rangle|0\rangle & |0\rangle|1\rangle & |1\rangle|0\rangle & |1\rangle|1\rangle \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{i\theta_{jk}} \end{pmatrix} \end{array} \begin{array}{c} |0\rangle|0\rangle \\ |0\rangle|1\rangle \\ |1\rangle|0\rangle \\ |1\rangle|1\rangle \end{array} \tag{6.5}$$

mit $\theta_{jk} = \frac{\pi}{2^{k-j}}$. Die Matrix H_j wird wie in Gleichung (6.4) angedeutet in der $|a_j\rangle$ Basis und die Matrix S_{jk} wie in Gleichung (6.5) angedeutet in der $|a_j\rangle|a_k\rangle$ Basis, mit $a_i = 0, 1$, dargestellt.
Um eine DFT_{2^L} auf $|a\rangle$ durchzuführen, muss man die folgende Operation L-mal wiederholen. Man numeriert jeden Durchgang mit einem Index j, der von $j = L-1$ auf $j = 0$ runterläuft. Für den ersten Durchgang setzt man $j = L-1$ und wendet H_j auf das Bit a_{L-1} an. Dann verringert man j um 1, also $j = L-2$, und wendet im zweiten Durchgang $S_{j,L-1}H_j$ auf das Bit a_{L-2} an. Für jeden weiteren Durchgang verringert man j um 1 und wendet $S_{j,j+1}S_{j,j+2}\ldots S_{j,L-1}H_j$ an. Dies wird solange wiederholt, bis $j = 0$ ist und schließt mit der Operation H_0 ab. So ergibt sich die DFT_{2^L} in der Darstellung der unitären Transformationen allgemein zu:

$$(H_0S_{0,1}S_{0,2}\ldots S_{0,L-2}S_{0,L-1})\ldots(H_{L-3}S_{L-3,L-2}S_{L-3,L-1})(H_{L-2}S_{L-2,L-1})(H_{L-1}). \tag{6.6}$$

Das bedeutet nun für einen Input der Größe L, dass L Operationen H_j und $\frac{L(L-1)}{2}$ Operationen $S_{j,k}$ durchgeführt werden müssen, also insgesamt $\frac{L(L+1)}{2}$ elementare Operationen für die gesamte DFT_{2^L}. Somit wächst die Durchführungszeit wie eine quadratische Funktion mit der Inputgröße L; folglich ist dies im Vergleich zu Algorithmen mit exponentiell anwachsendem Aufwand ein effizienter Algorithmus.
Wenn man die Folge (6.6) von Transformationen auf einen Zustand $|a\rangle$ anwendet, so erhält man

$$DFT_q|a\rangle = \frac{1}{\sqrt{q}}\sum_{b=0}^{q-1}\mathrm{e}^{\frac{i2\pi ac}{q}}|b\rangle, \tag{6.7}$$

wobei $|b\rangle$ die umgekehrte Bitfolge von $|c\rangle$ hat, also $|b\rangle = \overline{|c\rangle}$. Also wird eine zusätzliche Operation benötigt, die entweder die Bits von $|b\rangle$ in umgekehrter Reihenfolge anordnet, um $|c\rangle$ zu erhalten, oder das Register von $|b\rangle$ in umgekehrter Reihenfolge liest. Dies ändert jedoch nichts an der Effizienz des Algorithmus, da beide Möglichkeiten einfach umzusetzen sind.
Nun soll gezeigt werden, dass die Operationen H_j und $S_{j,k}$ eine DFT_{2^L} bilden.
Dazu betrachte man den Übergang der Amplituden $|a\rangle = |a_{L-1}\rangle\ldots|a_0\rangle$ zu $|b\rangle = |b_{L-1}\rangle\ldots|b_0\rangle = |c_0\rangle\ldots|c_{L-1}\rangle = \overline{|c\rangle}$. Der Faktor $\frac{1}{\sqrt{2}}$ der Hadamartransformation wird durch die L-malige Anwendung dieser Transformation zu einem Faktor $\frac{1}{\sqrt{q}}$, da $q = 2^L$. Somit muss nun nur noch das Zustandekommen der Phase $\frac{2\pi ac}{q}$ in der Summe (6.7) geklärt werden. Die Matrizen $S_{j,k}$ lassen die Qubits selbst unverändert und verändern lediglich deren Phase. Somit ergibt sich nur die Möglichkeit, das j-te Bit von a_j auf b_j zu überführen, indem man die Hadamartransformation H_j anwendet. Durch den rechten unteren Eintrag in Gleichung (6.4) wird π zu der Phase addiert, wenn beide Bits a_j und b_j den Wert 1 haben; für $a_j \cdot b_j \neq 1$ bleibt alles unverändert. Die Matrix $S_{j,k}$ addiert zusätzlich noch $\frac{\pi}{2^{k-j}}$ zu der Phase,

wenn a_j und b_k beide 1 sind und ändert sonst nichts. Damit ergibt sich die Phasenverschiebung von $|a\rangle$ nach $|b\rangle$ aus der Summe

$$\sum_{0\leq j<l} \pi a_j b_j \quad + \sum_{0\leq j<k<l} \frac{\pi}{2^{k-j}} a_j b_k \quad = \sum_{0\leq j\leq k<l} \frac{\pi}{2^{k-j}} a_j b_k.$$

Da c die umgekehrte Bitreihenfolge von b hat, kann man diese Summe umschreiben zu

$$\sum_{0\leq j\leq k<l} \frac{\pi}{2^{k-j}} a_j c_{l-1-k}.$$

Nun ersetzt man $l-k-1$ durch k und erhält

$$\sum_{0\leq j+k<l} 2\pi \frac{2^j 2^k}{2^l} a_j c_k.$$

Dieses Ergebnis verändert sich nicht, wenn man die Summen über j und k beide von 0 bis $L-1$ laufen lässt, da die zusätzlichen Terme nur Vielfache von 2π addieren und die Phase so unverändert bleibt. Es ergibt sich

$$\sum_{j,k=0}^{L-1} 2\pi \frac{2^j 2^k}{2^l} a_j c_k = \frac{2\pi}{2^l} \sum_{j=0}^{L-1} 2^j a_j \sum_{k=0}^{L-1} 2^k c_k. \tag{6.8}$$

Da $q = 2^L$, $a = \sum_{j=0}^{L-1} 2^j a_j$ und das Entsprechende für c gilt, ist die Gleichung (6.8) gleichbedeutend mit $2\pi\frac{ac}{q}$, was dem Phasenfaktor der Transformation aus (6.7) entspricht.

6.3 Der Quantenalgorithmus von Shor

6.3.1 Theoretische Betrachtung des Quanten–Shor–Algorithmus

Im Folgenden wird der Quantenalgorithmus von Shor beschrieben, der die Ordnung r der Funktion $y^a \pmod N$ in polynomialer Zeit errechnet.
Man beginnt wiederum mit der zu faktorisierenden Zahl N und wählt dazu zunächst eine Zahl q, die $N^2 \leq q < 2N^2$ erfüllt, mit $q = 2^L$. Nun wählt man ein $y < N$, mit $\mathsf{ggT}(y, N) = 1$ und beginnt mit einem L-Bit Register im Zustand $|0\rangle$, auf den eine DFT_q angewandt wird. Man erhält das Register

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle. \tag{6.9}$$

Als nächstes wird $y^a \pmod N$ berechnet (siehe dazu [72, 59, 73]) und das Ergebnis im zweiten Register gespeichert, wodurch sich der Zustand

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle |y^a \pmod N\rangle. \tag{6.10}$$

ergibt. Führt man eine Messung in dieser Basis durch, ist z mit $z = y^l \pmod N$ ein mögliches Ergebnis. Mit der Periode r gilt dann $y^l = y^{l+jr} \pmod N, \forall j \in \mathbb{N}$. Damit werden im ersten Register a Werte mit $a = l, l+r, l+2r, \ldots, l+Ar \leq q-1$ bestimmt, d.h., dass die Messung den Zustand

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle \longmapsto \frac{1}{\sqrt{A+1}} \sum_{j=0}^{A} |l+jr\rangle = |\Phi_l\rangle \tag{6.11}$$

liefert. Ziel ist es, von diesem Zustand ausgehend, die Periode r zu erhalten, und zwar mit einer Wahrscheinlichkeit in der Nähe von 1, wobei die Zahl der Messungen um diese Signifikanz zu erreichen, nicht exponentiell mit der Größe von N wächst. Das bedeutet, dass man den Wert von r mit einer endlichen, d.h. von Null verschiedenen Wahrscheinlichkeit bestimmen möchte, wenn die obige Messung höchstens $\mathrm{poly}(\log(N))$-mal wiederholt wird. Hierzu ist zu bemerken, dass wiederholte Messungen wegen der Besonderheiten des quantenmechanischen Messprozesses bei festem y verschiedene Werte für l ergeben und damit auch $|\Phi_l\rangle$ variiert.
Für das Finden der Ordnung r sei zunächst der vereinfachte Fall $\frac{q}{r} \in \mathbb{N}$, wobei r eine Zweierpotenz ist, betrachtet, um die prinzipielle Vorgehensweise und die Anwendung der DFT_q in diesem Zusammenhang zu verdeutlichen.
Allgemein gilt $\frac{q}{r} - \frac{l}{r} - 1 < A \leq \frac{q}{r} - \frac{l}{r}$, und damit erhält man für A die Bedingungen:

$$l = r \quad : \quad A = \frac{q}{r} - 1 \tag{6.12}$$

$$l < r \quad : \quad A = \frac{q}{r} - 1. \tag{6.13}$$

Damit ergibt sich der Zustand nach der Messung zu

$$|\Phi_l\rangle = \sqrt{\frac{r}{q}} \sum_{j=0}^{\frac{q}{r}-1} |l+jr\rangle = \sum_{a=0}^{q-1} \sum_{j=0}^{\frac{q}{r}-1} \sqrt{\frac{r}{q}}\, \delta_{a,l+jr} |a\rangle = \sum_{a=0}^{q-1} f(a)|a\rangle \tag{6.14}$$

mit

$$f(a) := \sum_{j=0}^{\frac{q}{r}-1} \sqrt{\frac{r}{q}}\, \delta_{a,l+jr}. \tag{6.15}$$

Auf diesen Zustand wendet man nun eine DFT_q an. So erhält man

$$\begin{aligned} DFT_q|\Phi_l\rangle &= DFT_q \sum_{a=0}^{q-1} f(a)|a\rangle \\ &= \sum_{a=0}^{q-1} f(a)\; DFT_q|a\rangle \end{aligned}$$

$$
\begin{aligned}
&\overset{\text{Gl.(6.3)}}{=} \sum_{a=0}^{q-1} f(a) \cdot \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \mathrm{e}^{\frac{i2\pi ac}{q}} |c\rangle \\
&\overset{\text{Gl.(6.15)}}{=} \sum_{j=0}^{\frac{q}{r}-1} \sqrt{\frac{r}{q}} \cdot \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \mathrm{e}^{\frac{i2\pi(l+jr)c}{q}} |c\rangle \\
&= \sum_{c=0}^{q-1} \sum_{j=0}^{\frac{q}{r}-1} \frac{\sqrt{r}}{q} \mathrm{e}^{\frac{i2\pi(l+jr)c}{q}} |c\rangle \\
&= \sum_{c=0}^{q-1} \tilde{f}(c) |c\rangle.
\end{aligned}
\tag{6.16}
$$

wobei

$$
\begin{aligned}
\tilde{f}(c) &= \sum_{j=0}^{\frac{q}{r}-1} \frac{\sqrt{r}}{q} \mathrm{e}^{\frac{i2\pi(l+jr)c}{q}} \\
&= \frac{\sqrt{r}}{q} \mathrm{e}^{\frac{i2\pi lc}{q}} \underbrace{\sum_{j=0}^{\frac{q}{r}-1} \mathrm{e}^{\frac{i2\pi(jrc)}{q}}}_{\frac{q}{r} \cdot \delta_{c,\lambda\frac{q}{r}} \text{ mit } \lambda \in \mathbb{N}} \\
&= \frac{1}{\sqrt{r}} \cdot \mathrm{e}^{\frac{i2\pi lc}{q}} \cdot \delta_{c,\lambda\frac{q}{r}}
\end{aligned}
\tag{6.17}
$$

ist, und somit

$$
DFT_q |\Phi_l\rangle = \frac{1}{\sqrt{r}} \sum_{\lambda=0}^{r-1} \mathrm{e}^{\frac{i2\pi l\lambda}{r}} |\lambda \frac{q}{r}\rangle. \tag{6.18}
$$

Eine Messung auf diesen Zustand liefert ein c, für das $\lambda\frac{q}{r} = c \Leftrightarrow \frac{c}{q} = \frac{\lambda}{r}$ gilt. Wenn $\mathsf{ggT}(\lambda, r) = 1$ gilt, kann r bestimmt werden, da c und q bekannt sind.
Da λ zufällig gewählt wird, ist die Wahrscheinlichkeit, dass $\mathsf{ggT}(\lambda, r) = 1$ gilt, für große r größer als $\frac{1}{\log(r)}$ (siehe Anhang A.4.2). Wiederholt man diese Berechnung $O(\log(r)) < O(\log(N))$-mal, kann man die gewünschte Wahrscheinlichkeit beliebig genau der 1 nähern. Somit ist dies eine effiziente Berechnung der Ordnung r.

Exkurs 2: Das Finden der Periode r wenn r keine Zweierpotenz ist
Rückblickend auf das Anfangsproblem (Zustand 6.11) soll nun die Situation untersucht werden, dass die Periode r keine Zweierpotenz ist. Ausgehend vom Zustand

$$
\frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} \mathrm{e}^{\frac{i2\pi ac}{q}} |c\rangle |y^a \pmod N\rangle \tag{6.19}
$$

nach Anwendung der DFT_q erhält man die Wahrscheinlichkeit, dass man ein $|c, y^a \pmod N\rangle$ mit $0 \leq l < r$ misst, folgendermaßen:
Man summiert über alle möglichen Ergebnisse, die den Zustand $|c, y^a \pmod N\rangle$ ergeben und erhält damit

$$\mathrm{Prob}(c) = \left| \frac{1}{q} \sum_{a:y^a \equiv y^l} \mathrm{e}^{\frac{i2\pi ac}{q}} \right|^2 \text{ mit } 0 \leq a < q, \text{ so dass } y^a \equiv y^l \pmod N. \tag{6.20}$$

Die Ordnung von y ist r, so dass die Summe über alle a läuft, für die gilt: $a \equiv l \pmod r$. Setzt man nun $a = jr + l$, so ergibt sich für die Wahrscheinlichkeit

$$\begin{aligned}
\mathrm{Prob}(c) &= \left| \frac{1}{q} \sum_{j=0}^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi(jr+l)c}{q}} \right|^2 \qquad \text{mit } \lfloor\ \rfloor \text{ Gaußklammer (siehe Anhang A.3)} \\
&= \left| \frac{1}{q} \sum_{j=0}^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi jrc}{q}} \cdot \mathrm{e}^{\frac{i2\pi lc}{q}} \right|^2 \\
&= \left| \frac{1}{q} \sum_{j=0}^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi jrc}{q}} \right|^2 \\
&= \left| \frac{1}{q} \sum_{j=0}^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi j(rc \pmod q)}{q}} \right|^2, \qquad (6.21)
\end{aligned}$$

wobei im letzten Schritt rc durch $rc \pmod q$ mit $-\frac{q}{2} \leq rc \pmod q \leq \frac{q}{2}$ ersetzt wurde.
Jetzt soll gezeigt werden, dass, wenn $rc \pmod q$ klein genug ist, alle Amplituden in der Summe annähernd in die gleiche Richtung zeigen und damit die Summe groß wird. Dazu nähert man die Summe durch ein Integral:

$$\frac{1}{q} \int_0^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi j(rc \pmod q)}{q}} dj \ + \ O\left(\frac{\lfloor \frac{q-1-l}{r} \rfloor}{q} \left(\mathrm{e}^{\frac{i2\pi(rc \pmod q)}{q}} - 1 \right) \right). \tag{6.22}$$

Wenn $rc \pmod q$ mit $-\frac{r}{2} \leq rc \pmod q \leq \frac{r}{2}$ gilt, ist der Fehlerterm von der Größenordnung $O\left(\frac{1}{q}\right)$. Im Folgenden soll gezeigt werden, dass der Wert des Integrals groß wird, wenn $-\frac{r}{2} \leq rc \pmod q \leq \frac{r}{2}$ gilt; damit ist auch die Wahrscheinlichkeit, den Zustand $|c, y^l \pmod N\rangle$ zu messen groß. Anzumerken ist noch, dass die Bedingung nur noch von c abhängt und von l unabhängig ist.
Substituiert man im Integral $u = \frac{rj}{q}$, dann ergibt sich:

$$\begin{aligned}
& \frac{1}{q} \int_0^{\lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi j(rc \pmod q)}{q}} dj \\
= \ & \frac{1}{q} \int_0^{\frac{r}{q} \lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi qu(rc \pmod q)}{rq}} \cdot \frac{q}{r} du \\
= \ & \frac{1}{r} \int_0^{\frac{r}{q} \lfloor \frac{q-1-l}{r} \rfloor} \mathrm{e}^{\frac{i2\pi(rc \pmod q)u}{r}} du. \qquad (6.23)
\end{aligned}$$

Da $l < r$ gilt, kann man die Gaußklammer in der oberen Integrationsgrenze durch $\lfloor \frac{q-1-l}{r} \rfloor \approx \frac{q}{r}$ nähern, was einen Fehler der Größenordnung $O(\frac{1}{q})$ mit sich bringt. Also ergibt sich für Gleichung (6.23)

$$\begin{aligned}
& \frac{1}{r}\int_0^1 \mathrm{e}^{\frac{i2\pi(rc\ (\mathrm{mod} q))u}{r}}\ du \\
= \quad & \frac{1}{r}\left[\frac{r}{i2\pi(rc\ (\mathrm{mod} q))}\ \mathrm{e}^{\frac{i2\pi(rc\ (\mathrm{mod} q))u}{r}}\right]_0^1 \\
= \quad & \frac{1}{r}\left[\frac{r}{i2\pi(rc\ (\mathrm{mod} q))}\ \mathrm{e}^{\frac{i2\pi(rc\ (\mathrm{mod} q))}{r}} - \frac{r}{i2\pi(rc\ (\mathrm{mod} q))}\right] \\
= \quad & \frac{1}{i2\pi(rc\ (\mathrm{mod} q))}\left[\mathrm{e}^{\frac{i2\pi(rc\ (\mathrm{mod} q))}{r}} - 1\right]. \qquad (6.24)
\end{aligned}$$

Es gilt $-\frac{r}{2} \leq rc\ (\mathrm{mod}\ q) \leq \frac{r}{2}$, was gleichbedeutend ist mit $-\frac{1}{2} \leq \frac{rc\ (\mathrm{mod}\ q)}{r} \leq \frac{1}{2}$. Für die Grenzwerte $-\frac{1}{2}$ und $\frac{1}{2}$ wird die rechte Seite der Gleichung (6.24) minimal und man erhält eine untere Grenze für die Wahrscheinlichkeit. Nun wird der Fall $\frac{rc\ (\mathrm{mod}\ q)}{r} = -\frac{1}{2}$ explizit ausgerechnet und davon ausgehend die Wahrscheinlichkeit bestimmt. Das Integral (6.24) nimmt den Wert

$$\begin{aligned}
& \frac{1}{i2\pi - \frac{r}{2}}\left[\mathrm{e}^{i2\pi\frac{-1}{2}} - 1\right] \\
= \quad & \frac{1}{-i\pi r}\left[\cos(\pi) - i\sin(\pi) - 1\right] \\
= \quad & \frac{2}{i\pi r} \qquad (6.25)
\end{aligned}$$

an. Analog erhält man für $\frac{rc\ (\mathrm{mod}\ q)}{r} = \frac{1}{2}$ den Wert $-\frac{2}{i\pi r}$. Das bedeutet, das der absolute Wert des Integrals

$$\left|\frac{1}{r}\int_0^1 \mathrm{e}^{\frac{i2\pi(rc\ (\mathrm{mod} q))u}{r}}\ du\right| = \frac{2}{\pi r} \qquad (6.26)$$

ist. Damit erhält man für die Wahrscheinlichkeit, dass ein $|c, y^a\ (\mathrm{mod}\ N)\rangle$ mit $0 \leq l < r$ gemessen wird, den Wert $\mathrm{Prob}(c) = \frac{4}{\pi^2 r^2}$.
Folglich ist die Wahrscheinlichkeit, den Zustand $|c, y^a\ (\mathrm{mod}\ N)\rangle$ zu messen, mindestens $\frac{1}{3r^2}$, wenn $-\frac{r}{2} \leq rc\ (\mathrm{mod}\ q) \leq \frac{r}{2}$ gilt, d.h. wenn es ein d gibt, so dass

$$\begin{aligned}
-\frac{r}{2} \leq \quad & rc - dq \quad \leq \frac{r}{2} \\
\Leftrightarrow -\frac{1}{2q} \leq \quad & \frac{c}{q} - \frac{d}{r} \quad \leq \frac{1}{2q} \\
\Rightarrow \quad & \left|\frac{c}{q} - \frac{d}{r}\right| \quad \leq \frac{1}{2q}, \qquad (6.27)
\end{aligned}$$

wobei c und q bekannt sind.

Da $q > N^2$, gibt es höchstens einen Bruch $\frac{d}{r}$ mit $r < N$, der Gleichung (6.27) genügt. Also erhält man den Bruch $\frac{d}{r}$ in kleinsten Termen, indem man $\frac{c}{q}$ auf den nächsten Bruch rundet, dessen Nenner kleiner als N ist. Dieser Bruch kann in polynomialer Zeit gefunden

werden, indem man die Kettenbruchdarstellung von $\frac{c}{q}$ berechnet, womit man die besten Approximationen von $\frac{c}{q}$ mit Hilfe von Brüchen findet (Anhang A.5). Wenn man den Bruch $\frac{d}{r}$ in kleinsten Termen berechnet hat und $\mathsf{ggT}(d, r) = 1$ ist, hat man die gesuchte Periode r gefunden.
Es gibt $\varphi(r)$ mögliche Werte dafür, dass d relativ prim zu r ist [2]. Jeder dieser Brüche $\frac{d}{r}$ liegt nahe bei einem Bruch $\frac{c}{q}$, der $\left|\frac{c}{q} - \frac{d}{r}\right| \leq \frac{1}{2q}$ erfüllt. Da r die Ordnung von y ist, gibt es r mögliche Werte für y^l. Also gibt es $r \cdot \varphi(r)$ mögliche Zustände $|c, y^a \pmod N\rangle$, die es ermöglichen, r zu erhalten. Da jeder dieser Zustände mit einer Wahrscheinlichkeit von mindestens $\frac{1}{3r^2}$ vorkommt, erhält man r mit einer Wahrscheinlichkeit von mindestens $\frac{\varphi(r)}{3r}$. Die Beachtung des Großen Primzahlsatzes und seiner Approximation, $\frac{\varphi(r)}{r} > \frac{\mathrm{e}^{-\gamma}}{\log\log(r)}$ mit $\mathrm{e}^{-\gamma}$ konstant (siehe Anhang A.4.2), erfordert eine nur $O(\log\log(r))$-malige Wiederholung der oben beschriebenen Vorgehensweise, um mit einer beliebig großen Wahrscheinlichkeit zum Erfolg zu gelangen.

6.3.2 Zahlenbeispiele für den Quanten–Shor–Algorithmus

In Anlehnung an das Zahlenbeispiel aus Kapitel (6.1) soll nun für $N = 21$ auch der Quanten–Shor–Algorithmus schrittweise vorgeführt werden. Im ersten Beispiel wird der Quanten–Shor–Algorithmus für den Fall, dass die Periode r eine Zweierpotenz ist durchgeführt und im Anschluss daran in Form eines zweiten Beispiels (siehe Exkurs 3) für den Fall, dass r keine Zweierpotenz ist.

Beispiel 1: Zahlenbeispiel für den Quanten–Shor–Algorithmus, wenn die Periode r eine Zweierpotenz ist

1.Schritt: Benutzung des Quantenparallelismus
Man wähle eine beliebige Zahl y. Wenn y keine teilerfremde Zahl zu N ist, ergibt sich ein Faktor von N. Ansonsten gilt $\mathsf{ggT}(y, N) = 1$ und man verfährt nach dem Algorithmus.
Sei L so, dass $N^2 \leq 2^L = q < 2N^2$ gilt. Zu Anfang wird auf das erste Register, das aus L-Qubits im Zustand $|0\rangle$ besteht, die DFT_q angewandt, so dass man das Register

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle \tag{6.28}$$

erhält. Man benutzt den Quantenparallelismus, um $f(a) = y^a \pmod N$ für alle Zahlen von 0 bis $q - 1$ zu berechnen. Die Funktion $f(a)$ ist in dem Quantenzustand

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|f(a)\rangle \tag{6.29}$$

verschlüsselt. Im Folgenden wird angenommen, dass $y = 8$ zufällig gewählt wurde. Da $N^2 = 441$ und $2N^2 = 882$ gilt, findet man $L = 9$ (da $2^9 = 512$), also

[2] $\varphi(r)$ ist dabei die Eulersche φ-Funktion (siehe Anhang A.4.1).

$N^2 = 441 \leq 2^9 < 882 = 2N^2$.
Hat man beispielsweise insgesamt 14 Qubits, werden damit 9 für a und $\lceil \log_2(N) \rceil = 5$ für $f(a)$ benutzt, um die Superposition aus Gleichung (6.29) zu berechnen.

2.Schritt: Konstruktion eines Zustands, dessen Amplitude die gleiche Periode wie $f(a)$ hat

Die diskrete Fouriertransformation wirkt auf die Funktion der Amplitude in Verbindung mit dem Inputzustand. Im Hinblick darauf, dass die diskrete Fouriertransformation genutzt wird, um die Periode r von $f(a)$ zu erhalten, wird ein Zustand konstruiert, dessen Amplitudenfunktion dieselbe Periode wie $f(a)$ hat.
Um einen solchen Zustand zu generieren, misst man die letzten $\lceil \log_2(N) \rceil$ Qubits des Zustandes von (6.29), die $f(a)$ verschlüsseln. Das Ergebnis ist ein zufälliger Wert $z = y^l \pmod N$. Der Wert l ist an sich nicht von Interesse, sondern nur der Effekt, den die Messung auf die Superposition hat. Die Messung projiziert den Zustandsraum auf den Unterraum, der kompatibel mit dem gemessenen Wert ist, so dass der Zustand nach der Messung

$$\sum_{a=0}^{q-1} g(a)|a\rangle|y^a \pmod N\rangle \quad \text{mit } g(a) = k\ \delta_{a,l+jr} \quad (k : \text{Normierungsfaktor}) \tag{6.30}$$

ist. Damit verbleibt im ersten Register eine Überlagerung aller a, die $y^a = y^l \pmod N$ erfüllen, übrig, d.h. es gilt $a \equiv l \pmod r$.
Wenn man zwei aufeinanderfolgende a in der Summe messen könnte, wäre die Periode r gefunden. Aber die Gesetze der Quantenmechanik erlauben nur eine Messung; danach ist der Zustand zerstört und muss neu präpariert werden.
Angenommen es wird $y^l \pmod N \equiv 8$ gemessen; dann verbleiben im ersten Register alle a, für die $8^a \pmod{21} \equiv 8$ gilt.

3.Schritt: Die Benutzung der diskreten Fouriertransformation

Auf das in Gleichung (6.30) erhaltene erste Register

$$\sum_{a=0}^{q-1} g(a)|a\rangle$$

wird nun eine DFT_q angewendet. Dies ergibt

$$DFT_q \sum_{a=0}^{q-1} g(a)|a\rangle = \sum_{c=0}^{q-1} \tilde{f}(c)|c\rangle.$$

Wenn die Periode r eine Zweierpotenz ist, erhält man das exakte Ergebnis

$$DFT_q \sum_{a=0}^{q-1} g(a)|a\rangle = \frac{1}{\sqrt{r}} \sum_{\lambda=0}^{r-1} \mathrm{e}^{\frac{i2\pi l\lambda}{r}} |\lambda \frac{q}{r}\rangle. \tag{6.31}$$

4.Schritt: Das Erhalten der Periode
Der Zustand in dem zuletzt erhaltenen Register (6.31) wird gemessen; das Ergebnis wird c genannt. Die Periodenbestimmung ist einfach, wenn die Periode eine Zweierpotenz ist, da dann die diskrete Fouriertransformation exakt ein Vielfaches von $\frac{q}{r}$ ergibt. In diesem Fall gilt $c = \lambda\frac{q}{r}$ für alle λ. Meistens sind λ und r teilerfremd, so dass der maximal gekürzte Bruch $\frac{c}{q}(= \frac{\lambda}{r})$ ein Bruch ist, dessen Nenner q die Periode r ist.
Angenommen die Messung des Zustands ergibt $c = 768$. Da $q = 512$ gilt (siehe 1.Schritt) ist

$$\frac{c}{q} = \frac{768}{512} = \frac{3}{2} = \frac{\lambda}{r},$$

also die Periode $r = 2$ und somit eine Zweierpotenz.

5.Schritt: Das Finden der Faktoren von N
Wie schon in Kapitel (6.1) beschrieben, findet man die Faktoren von N, indem man mit Hilfe des Euklidischen Algorithmus sowohl den $\mathsf{ggT}(y^{r/2} - 1, N)$ als auch den $\mathsf{ggT}(y^{r/2} + 1, N)$ berechnet, vorausgesetzt r ist eine gerade Zahl.
Somit erhält man für $N = 21$, $y = 8$ und $r = 2$:

$$\begin{array}{lrll} \mathsf{ggT}(7,21): & 21 & = 3 \cdot 7 + 0, & \text{also p=7} \\ \mathsf{ggT}(9,21): & 21 & = 2 \cdot 9 + 3, & \\ & 9 & = 3 \cdot 3 + 0, & \text{also q=3.} \end{array}$$

Die Zahl $N = 21$ läßt sich somit in $21 = 3 \cdot 7$ faktorisieren.

6.Schritt: Eventuelle Wiederholung des Algorithmus
Der Algorithmus muss unter den folgenden Umständen wiederholt werden:

1. Die Periode r und der Faktor λ sind nicht teilerfremd, so dass der Nenner q ein Faktor der Periode ist.
2. Der 5. Schritt ergibt N als Faktor von N.
3. Die erhaltene Periode von $f(a) = y^a \pmod N$ ist ungerade.

Exkurs 3:

Beispiel 2: Zahlenbeispiel für den Quanten–Shor–Algorithmus, wenn die Periode r keine Zweierpotenz ist

1.Schritt: Benutzung des Quantenparallelismus

Man wähle eine beliebige Zahl y. Wenn y keine teilerfremde Zahl zu N ist, ergibt sich ein Faktor von N. Ansonsten gilt $\mathrm{ggT}(y, N) = 1$ und man verfährt nach dem Algorithmus.
Sei L so, dass $N^2 \leq 2^L = q < 2N^2$ gilt. Zu Anfang wird auf das erste Register, das aus L-Qubits im Zustand $|0\rangle$ besteht, die DFT_q angewandt, so dass man das Register

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle \tag{6.32}$$

erhält. Man benutzt den Quantenparallelismus, um $f(a) = y^a \pmod N$ für alle Zahlen von 0 bis $q-1$ zu berechnen. Die Funktion $f(a)$ ist in dem Quantenzustand

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|f(a)\rangle \tag{6.33}$$

verschlüsselt. Im Folgenden wird angenommen, dass $y = 11$ zufällig gewählt wurde. Da $N^2 = 441$ und $2N^2 = 882$ gilt, findet man $L = 9$ (da $2^9 = 512$), also $N^2 = 441 \leq 2^9 < 882 = 2N^2$.
Für den Fall, dass man insgesamt 14 Qubits hat, werden damit 9 für a und $\lceil \log_2(N) \rceil = 5$ für $f(a)$ benutzt, um die Superposition aus Gleichung (6.33) zu berechnen.

2.Schritt: Konstruktion eines Zustands, dessen Amplitude die gleiche Periode wie $f(a)$ hat

Die diskrete Fouriertransformation wirkt auf die Funktion der Amplitude in Verbindung mit dem Inputzustand. Im Hinblick darauf, dass die diskrete Fouriertransformation genutzt wird, um die Periode r von $f(a)$ zu erhalten, wird ein Zustand konstruiert, dessen Amplitudenfunktion dieselbe Periode wie $f(a)$ hat.
Um einen solchen Zustand zu generieren, misst man die letzten $\lceil \log_2(N) \rceil$ Qubits des Zustandes von (6.33), die $f(a)$ verschlüsseln. Das Ergebnis ist ein zufälliger Wert $z = y^l \pmod N$. Der Wert l ist an sich nicht von Interesse, sondern nur der Effekt, den die Messung auf die Superposition hat. Die Messung projiziert den Zustandsraum auf den Unterraum, der kompatibel mit dem gemessenen Wert ist, so dass der Zustand nach der Messung

$$\sum_{a=0}^{q-1} g(a)|a\rangle|y^a \pmod N\rangle \quad \text{mit } g(a) = k\ \delta_{a,l+jr} \quad (k : \text{Normierungsfaktor}) \tag{6.34}$$

ist. Damit verbleibt im ersten Register eine Überlagerung aller a, die $y^a = y^l \pmod N$ erfüllen, übrig, d.h. es gilt $a \equiv l \pmod r$.
Wenn man zwei aufeinanderfolgende a in der Summe messen könnte, wäre die Periode r gefunden. Aber die Gesetze der Quantenmechanik erlauben nur eine Messung; danach ist der Zustand zerstört und muss neu präpariert werden.

Angenommen es wird $y^l \pmod N \equiv 8$ gemessen; dann verbleiben im ersten Register alle a, für die $11^a \pmod{21} \equiv 8$ gilt (siehe Abbildung 6.3.2).

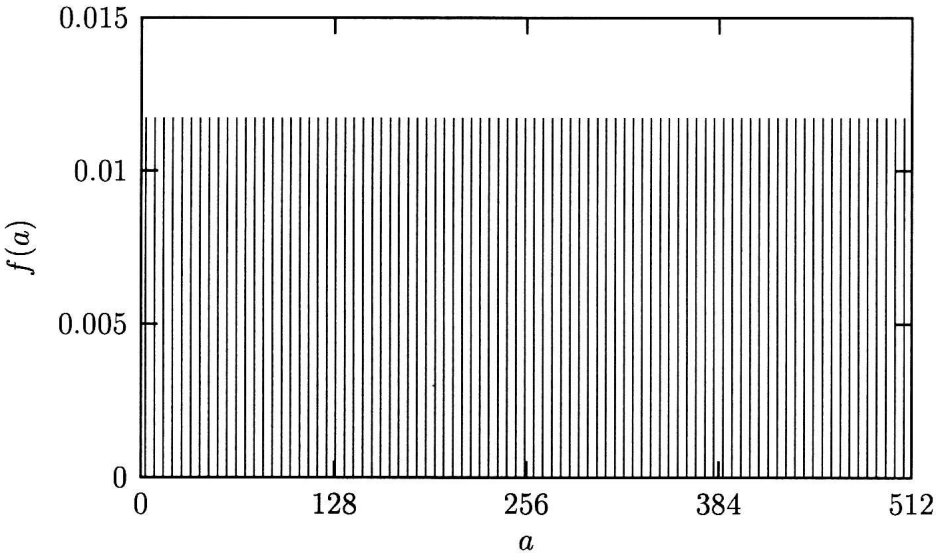


Abbildung 6.1: Nach der Messung verbleiben im ersten Register die Werte von a für die $11^a \pmod{21} \equiv 8$ gilt. Die Abbildung zeigt deutlich die Periodizität von $f(a)$.

3.Schritt: Die Benutzung der diskreten Fouriertransformation
Auf das in Gleichung (6.34) erhaltene erste Register

$$\sum_{a=0}^{q-1} g(a)|a\rangle$$

wird nun eine DFT_q angewendet. Dies ergibt

$$DFT_q \sum_{a=0}^{q-1} g(a)|a\rangle = \sum_{c=0}^{q-1} \tilde{f}(c)|c\rangle.$$

Wenn die Periode r eine Zweierpotenz ist, erhält man das exakte Ergebnis

$$DFT_q \sum_{a=0}^{q-1} g(a)|a\rangle = \frac{1}{\sqrt{r}} \sum_{\lambda=0}^{r-1} \mathrm{e}^{\frac{i2\pi l\lambda}{r}} |\lambda \frac{q}{r}\rangle. \tag{6.35}$$

Ist die Periode r keine Zweierpotenz, was dem Normalfall entspricht, erhält man durch die DFT_q eine gute Näherung der Vielfachen von $\frac{q}{r}$ (siehe Abbildung 6.3.2).

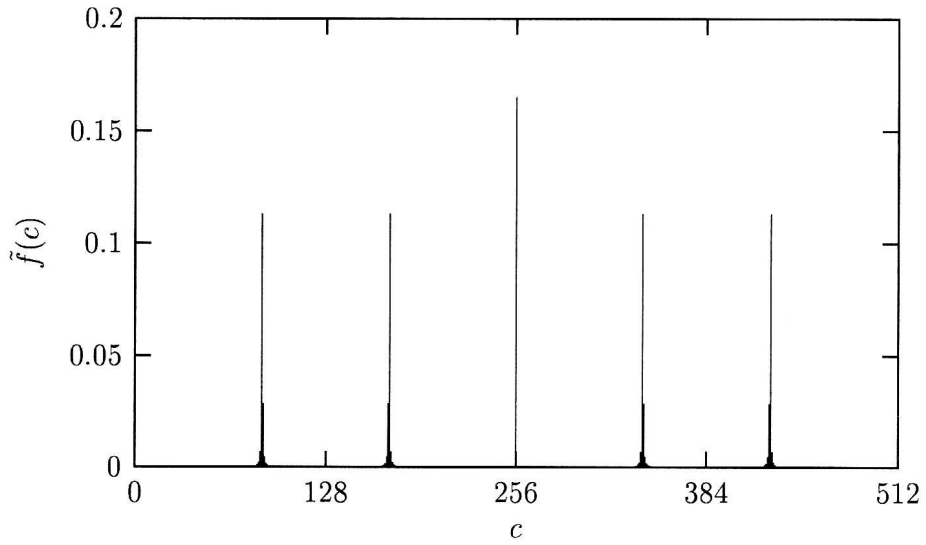


Abbildung 6.2: Nach der Messung verbleiben im ersten Register die Werte von a für die $11^a \pmod{21} \equiv 8$ gilt. Die Abbildung zeigt deutlich die Periodizität von $f(a)$.

Die Abbildungen 6.3.2 und 6.3.2 wurden mit einem C++ Programm erstellt, das in Anhang C zu finden ist.

4.Schritt: Das Erhalten der Periode

Der Zustand in dem zuletzt erhaltenen Register (6.35) wird gemessen; das Ergebnis wird c genannt. Die Periodenbestimmung ist einfach, wenn die Periode eine Zweierpotenz ist, da dann die diskrete Fouriertransformation exakt ein Vielfaches von $\frac{q}{r}$ ergibt. In diesem Fall gilt $c = \lambda\frac{q}{r}$ für alle λ. Meistens sind λ und r teilerfremd, so dass der maximal gekürzte Bruch $\frac{c}{q}(=\frac{\lambda}{r})$ ein Bruch ist, dessen Nenner q die Periode r ist.
Die Tatsache, dass die diskrete Fouriertransformation üblicherweise nur annähernd Vielfache der skalierten Frequenz ergibt, erschwert das Erhalten der Periode aus der Messung. Wenn die Periode r keine Zweierpotenz ist, erhält man die Periode aus der Kettenbruchdarstellung von $\frac{c}{q}$.
Angenommen die Messung des Zustands ergibt $c = 171$; da c und $q = 512$ teilerfremd sind, gilt auch $\mathrm{ggT}(r, q) = 1$, und man muss sich der Approximation durch Kettenbrüche bedienen, indem man folgenden Algorithmus durchläuft:

$$\begin{array}{llll}
a_0 = \left\lfloor \frac{c}{q} \right\rfloor & a_n = \left\lfloor \frac{1}{\epsilon_{n-1}} \right\rfloor & & \\
\epsilon_0 = \frac{c}{q} - a_0 & \epsilon_n = \frac{1}{\epsilon_{n-1}} - a_n & & \\
p_0 = a_0 & p_1 = a_1 a_0 + 1 & \dots & p_n = a_n p_{n-1} + p_{n-2} \\
q_0 = 1 & q_1 = a_1 & \dots & q_n = a_n q_{n-1} + q_{n-2}.
\end{array} \tag{6.36}$$

Der erste Bruch $\frac{p_n}{q_n}$ mit $q_n < N \leq q_{n+1}$ liefert die gewünschte Approximation.
Für das hier betrachtete Beispiel erhält man die folgende Tabelle der oben benutzten Größen:

i	a_i	p_i	q_i	ϵ_i
0	0	0	1	0,3339844
1	2	1	2	0,9941520
2	1	1	3	0,0058824
3	169	170	509	0,9999694

Aus dieser Tabelle ergibt sich mit $3 = q_2 < N \leq q_3 = 509$ die Periode $r = 3$. Da die erhaltene Periode ungerade ist, scheitert der im nächsten Schritt anzuwendende Euklidische Algorithmus und eine Wiederholung des Algorithmus ist notwendig.
Wenn die Messung des Zustands $c = 256$ ergibt, muss der Algorithmus abermals wiederholt werden, da c und q nicht teilerfremd sind.
Angenommen die Messung des Zustands ergibt $c = 427$; da c und q teilerfremd sind, gilt auch hier $\mathsf{ggT}(r, q) = 1$, und man muss sich der Approximation durch Kettenbrüche bedienen, indem man den schon beschriebenen Algorithmus durchläuft. Die Werte der notwendigen Größen enthält die folgende Tabelle:

i	a_i	p_i	q_i	ϵ_i
0	0	0	1	0,8339844
1	1	1	1	0,1990632
2	5	5	6	0,02352941
3	42	211	253	0,5

Aus dieser Tabelle ergibt sich mit $6 = q_2 < N \leq q_3 = 253$ die Periode $r = 6$.

5.Schritt: Das Finden der Faktoren von N
Wie schon in Kapitel (6.1) beschrieben, findet man die Faktoren von N, indem man mit Hilfe des Euklidischen Algorithmus sowohl den $\mathsf{ggT}(y^{r/2}-1, N)$ als auch den $\mathsf{ggT}(y^{r/2}+1, N)$ berechnet, vorausgesetzt r ist eine gerade Zahl.
Somit erhält man für $N = 21$, $y = 11$ und $r = 6$:

$$\begin{array}{rrcrcll}
\mathsf{ggT}(1330, 21): & 1330 & = & 63 \cdot 21 & + & 7, & \\
 & 21 & = & 3 \cdot 7 & + & 0, & \text{also p=7} \\
\mathsf{ggT}(1332, 21): & 1332 & = & 63 \cdot 21 & + & 9, & \\
 & 9 & = & 3 \cdot 3 & + & 0, & \text{also q=3.}
\end{array}$$

Die Zahl $N = 21$ läßt sich somit in $21 = 3 \cdot 7$ faktorisieren.

6.Schritt: Eventuelle Wiederholung des Algorithmus
Der Algorithmus muss unter den folgenden Umständen wiederholt werden:

1. Der Wert von c lag nicht nahe genug bei einem Vielfachen von $\frac{q}{r}$.
2. Die Periode r und der Faktor λ sind nicht teilerfremd, so dass der Nenner q ein Faktor der Periode ist.
3. Der 5. Schritt ergibt N als Faktor von N.
4. Die erhaltene Periode von $f(a) = y^a \pmod N$ ist ungerade.

Anhang A

Zahlentheoretische Grundlagen

A.1 Kongruenzen und Restklassen

Es sei m eine natürliche Zahl. Wenn zwei ganze Zahlen a und b bei Division durch m denselben Rest ergeben, wenn also

$$a = um + r \quad \text{und} \quad b = vm + r \quad \text{mit} \quad u, v, r \in \mathbb{Z} \quad \text{und} \quad 0 \leq r < m,$$

dann nennt man a und b kongruent modulo m und schreibt

$$a \equiv b \pmod{m}.$$

Es gilt offensichtlich

$$a \equiv b \pmod{m} \Longleftrightarrow m|a - b \quad (\text{d.h. } m \text{ teilt } a - b).$$

Statt "n ist von der Form $a + km\,(k \in \mathbb{Z})$" sagt man nach Gauß kurz "$n \equiv a \pmod{m}$".
Die Kongruenz modulo m ist eine Äquivalenzrelation in $\mathbb{Z}$, es gilt nämlich

$$\begin{array}{l} a \equiv a \pmod{m} \quad \text{für alle } a \in \mathbb{Z}; \\ \text{aus} \quad a \equiv b \pmod{m} \quad \text{folgt} \quad b \equiv a \pmod{m}; \\ \text{aus} \quad a \equiv b \pmod{m} \quad \text{und} \quad b \equiv c \pmod{m} \quad \text{folgt } a \equiv c \pmod{m}. \end{array}$$

Die Kongruenz modulo m induziert also eine Klasseneinteilung von $\mathbb{Z}$ in Äquivalenzklassen, welche man Restklassen modulo m nennt. Die Restklasse "a modulo m" oder kurz "$a \pmod{m}$" besteht aus allen ganzen Zahlen, die bei Division durch m den gleichen Rest wie a lassen, also aus allen zu $a \pmod{m}$ kongruenten Zahlen. Statt "$a \pmod{m}$" schreibt man auch kürzer $[a]_m$ bzw. noch kürzer $[a]$, wenn aus dem Zusammenhang klar hervorgeht, bezüglich welchen Moduls die Restklasse zu bilden ist. Es ist also

$$[a]_m = \{x \in \mathbb{Z} | x \equiv a \pmod{m}\}$$

und

$$[a]_m = [b]_m \Leftrightarrow a \equiv b \pmod{m}.$$

Man beschreibt eine Restklasse durch Angabe eines Vertreters, also durch Angabe einer Zahl aus der betreffenden Restklasse. Zum Modul m gibt es genau m verschiedene Restklassen, welche man in der Regel durch ihre kleinsten nichtnegativen Vertreter beschreibt:

$$[0]\,,[1]\,,[2]\,,\ldots,[m-1]\,.$$

Die Menge aller Restklassen modulo m werden im folgenden mit R_m bezeichnet. In R_m soll nun eine Addition "$+$" und eine Multiplikation "$\cdot$" eingeführt werden, so dass $(R_m,+,\cdot)$ eine algebraische Struktur ist. Es gilt:

$$[a]+[b]=[a+b] \quad \text{und} \quad [a]\cdot[b]=[a\cdot b]\,.$$

Man führt also die entsprechenden Operationen mit den Vertretern der Restklassen aus.
Es ist leicht nachzurechnen, dass die Restklassenverknüpfungen beide assoziativ sind, dass neutrale Elemente $([0] \text{ bzw. } [1])$ existieren und das Distributivgesetz gilt. Ferner ist jedes Element bezüglich der Addition invertierbar, das Inverse von $[a]$ ist $-[a]:=[-a]=[m-a]$.

A.2 Euklidischer Algorithmus

Mit Hilfe des Euklidischen Algorithmus berechnet man ohne Kenntnis der Primzerlegung von zwei Zahlen a und b den größten gemeinsamen Teiler dieser beiden Zahlen.
Es seien $a,b\in\mathbb{N}\setminus\{0\}$ mit $a\geq b$. Man setze $a_0:=a,\ a_1:=b$ und bilde sukzessiv folgende Kette von Divisionen mit Rest:

$$\begin{array}{rcllll}
a_0 & = & q_1a_1+a_2 & \text{mit} & q_1,\ a_2\in\mathbb{Z}, & 0\leq a_2<a_1,\\
a_1 & = & q_2a_2+a_3 & \text{mit} & q_2,\ a_3\in\mathbb{Z}, & 0\leq a_3<a_2,\\
 & \vdots & & & \vdots & \vdots\\
a_{n-2} & = & q_{n-1}a_{n-1}+a_n & \text{mit} & q_{n-1},\ a_n\in\mathbb{Z}, & 0\leq a_n<a_{n-1},
\end{array}$$

wo $q_1,q_2,\ldots$ die Quotienten und $a_2,a_3,\ldots$ die Reste sind. Dann gibt es einen ersten Index k, mit $1\leq k\leq b$, so dass gilt: $a_k>0, a_{k+1}=0$. Die Zahl a_k ist dann der größte gemeinsame Teiler von a und b.

Beispiel : a=4081, b=2585

$$\begin{array}{rcrcrcr}
4081 & = & 1 & \cdot & 2585 & + & 1496\\
2585 & = & 1 & \cdot & 1496 & + & 1089\\
1496 & = & 1 & \cdot & 1089 & + & 407\\
1089 & = & 2 & \cdot & 407 & + & 275\\
407 & = & 1 & \cdot & 275 & + & 132\\
275 & = & 2 & \cdot & 132 & + & 11\\
132 & = & 12 & \cdot & 11 & + & 0
\end{array}$$

also $\mathsf{ggT}(a,b)=11$.

A.3 Chinesischer Restesatz

Sind $m_1, m_2, \ldots, m_k$ paarweise teilerfremde natürliche Zahlen und $c_1, c_2, \ldots c_k$ ganze Zahlen, dann existiert genau ein Restklasse $\lfloor x \rfloor$ [1] zum Modul $m = m_1 \cdot m_2 \cdot \ldots m_k$, für welche gilt:

$$x \equiv c_i \pmod{m_i} \text{ mit } i = 1, 2, \ldots, k.$$

A.4 Die Eulersche φ-Funktion, Die Ordnung $\bmod N$, Der große Primzahlsatz

A.4.1 Die Eulersche φ-Funktion

Die durch die Festsetzung $\varphi(a) :=$ Anzahl der zu $a \in \mathbb{N}$ teilerfremden Zahlen aus $\{1, 2, \ldots, a\}$ erklärte Funktion $\varphi : \mathbb{N} \to \mathbb{N}$ heißt Eulersche φ-Funktion.
Es ist also: $\varphi(1) = 1$, $\varphi(2) = 1$, $\varphi(3) = 2$, $\varphi(4) = 2 \ldots$.
Es seien noch zwei wichtige Eigenschaften der Eulerschen φ-Funktion erwähnt:

- Sei p eine Primzahl, dann gilt $\varphi(p) = p - 1$.
- Wenn $\mathsf{ggT}(m, n) = 1$ gilt, dann gilt $\varphi(mn) = \varphi(m)\varphi(n)$, d.h. φ ist multiplikativ.

Ein wichtiges Ergebnis in diesem Zusammenhang stellt der Satz von Fermat-Euler dar.

Satz von Fermat-Euler: Für alle Zahlen $a, N \in \mathbb{N}$ mit $\mathsf{ggT}(a, N) = 1$ gilt: $a^{\varphi(N)} \equiv 1 (\bmod N)$.

Ein Spezialfall ist der kleine Fermatsche Satz: Sei $N = p$ eine Primzahl, so gilt für alle $a \in \mathbb{N}$ mit $\mathsf{ggT}(p, a) = 1$: $a^{p-1} \equiv 1 (\bmod p)$. (Kleiner Fermatscher Satz)

Dies führt zu folgender

Definition: Sei $a, N \in \mathbb{N}$ mit $\mathsf{ggT}(a, N) = 1$. Dann ist die Ordnung r von $a \pmod N$ die kleinste Potenz von a, so dass $a^r \equiv 1 \pmod N$.

Ist hingegen $\mathsf{ggT}(a, N) \neq 1$, dann gibt es keine Potenz m zu a, so dass $a^r \equiv 1 \pmod N$ ist, da $a^m - \lambda N$ auf jeden Fall durch $\mathsf{ggT}(a, N)$ für alle λ, m geteilt wird.

[1] $\lfloor x \rfloor$ Gaußklammer, größte ganze Zahl $\leq x$

A.4.2 Der große Primzahlsatz

Sei $\pi(N)$ die Anzahl der Primzahlen, die kleiner oder gleich N sind

$$\pi(N) = \sum_{p \in \mathbb{P}, p \leq N} 1.$$

Damit ist also: $\pi(1) = 0, \pi(2) = 1, \pi(3) = \pi(4) = 2, \ldots$.
Der große Primzahlsatz besagt, dass

$$\lim_{N \to \infty} \frac{\pi(N)}{\frac{N}{\log(N)}} = 1.$$

Das bedeutet, die elementar-transzendente Funktion $\frac{N}{\log(N)}$ approximiert für große Werte von N die Funktion $\pi(N)$ so gut, dass der Quotient von $\pi(N)$ und dieser Funktion beliebig dicht bei 1 liegt. Man beschreibt diesen Sachverhalt auch häufig suggestiv durch

$$\pi(N) = \frac{N}{\log(N)} \qquad \text{für genügend große } N.$$

Genauer gesagt gibt es für jedes $\epsilon > 0$ ein N_0, so dass

$$\frac{N}{\log(N)} - \epsilon \leq \varphi(N) \leq \frac{N}{\log(N)} + \epsilon \qquad \text{für alle } N > N_0.$$

Daraus folgt offensichtlich, dass $\varphi(N) \geq \pi(N)$ und damit auch

$$\varphi(N) \geq \frac{N}{\log(N)}. \tag{A.1}$$

Also ist die Wahrscheinlichkeit $\frac{\varphi(N)}{N}$, dass eine zufällig gewählte Zahl zwischen 1 und N teilerfremd zu N ist, größer als $\frac{1}{\log(N)}$. Die Abschätzung (A.1) ist jedoch ungenau, da es gewöhnlich mehr Zahlen gibt, die teilerfremd zu N sind als nur die Primzahlen. In ([74], Kapitel 18.4) wird gezeigt, dass

$$\liminf \frac{\varphi N}{\frac{N}{\log\log(N)}} = \mathrm{e}^{-\gamma} \qquad \text{mit } \gamma = 0,577216 \ \ (\text{Eulerkonstante})$$

gilt; damit erhält man für $\varphi(N)$ eine bessere Abschätzung:

$$\varphi(N) > \frac{\mathrm{e}^{-\gamma} N}{\log\log(N)}.$$

Dieses Resultat ist für Betrachtung des Quanten-Shor-Algorithmus von großer Bedeutung.

A.5 Kettenbrüche

Ein (endlicher) Kettenbruch ist von folgender Gestalt:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots + \cfrac{1}{a_N}}}}} =: [a_0, a_1, \ldots a_N]. \tag{A.2}$$

Dabei sind $a_0, a_1, \ldots, a_N \in \mathbb{N}$ die Nenner des Kettenbruchs. In dieser Darstellung ist $[a_0, a_1, \ldots, a_n]$ mit $0 \leq n \leq N$ der n-te Näherungsbruch zu $[a_0, a_1, \ldots, a_N]$. Einen Näherungsbruch kann man mit Hilfe des folgenden Satzes berechnen:

Satz: Definiert man p_n und q_n durch die Gleichungen

$$\begin{array}{lll} p_0 = a_0 & p_1 = a_1 a_0 + 1 & p_n = a_n p_{p-1} + p_{n-2} \\ q_0 = 1 & q_1 = a_1 & q_n = a_n q_{n-1} + q_{n-2}, \end{array} \tag{A.3}$$

dann gilt

$$[a_0, a_1, a_2, \ldots, a_n] = \frac{p_n}{q_n}.$$

Eine wichtige Eigenschaft von Näherungsbrüchen ist die Irreduzibelität, d.h. es gilt $\mathsf{ggT}(p_n, q_n) = 1$.
Außerdem gilt, dass man jede (positive) rationale Zahl durch einen endlichen Kettenbruch mit Hilfe des folgenden Algorithmus darstellen kann:
Es sei $x \in \mathbb{R}$ und $a_0 = \lfloor x \rfloor$, mit $\lfloor x \rfloor \leq x$. Dann gilt

$$x = a_0 + \xi_0 \qquad \text{mit } 0 \leq \xi_0 < 1.$$

Ist $\xi_0 \neq 0$, kann man schreiben

$$\frac{1}{\xi_0} = a_1', \qquad a_1' = a_1 + \xi_1, \qquad 0 \leq \xi_1 < 1.$$

Ist $\xi_1 \neq 0$, kann man schreiben

$$\frac{1}{\xi_1} = a_2' = a_2 + \xi_2, \qquad 0 \leq \xi_2 < 1$$

usw.
Ferner ist $a_n' = \frac{1}{\xi_{n-1}} > 1$, also $a_n \geq 1$ für $n_n \geq 1$. Deshalb ist

$$x = [a_0, a_1'] = \left[a_0, a_1 + \frac{1}{a_2'}\right] = [a_0, a_1, a_2'] = [a_0, a_1, a_2, a_3'] = \cdots,$$

wobei $a_0, a_1, \ldots$ ganze Zahlen sind und $a_1 > 0, a_2 > 0, \ldots$ ist. Das Gleichungssystem

$$\begin{aligned} x &= a_0 + \xi_0 \quad (0 \leq \xi_0 < 1) \\ \frac{1}{\xi_0} &= a_1' = a_1 + \xi_1 \quad (0 \leq \xi_1 < 1) \\ \frac{1}{\xi_1} &= a_2' = a_2 + \xi_2 \quad (0 \leq \xi_2 < 1) \\ &\ldots \end{aligned}$$

ist als Kettenbruchalgorithmus bekannt. Der Algorithmus lässt sich fortsetzen, solange $\xi_n \neq 0$ ist. Kommt man jedoch einmal zu einem Wert von n, etwa N, für den $\xi_N = 0$ ist, bricht der Algorithmus ab, und es ist

$$x = [a_0, a_1, a_2, \ldots, a_N].$$

In diesem Fall wird x durch einen endlichen Kettenbruch dargestellt und ist rational. Die Zahlen a_n' sind die vollständigen Quotienten des Kettenbruchs. Das wichtigste Ergebnis bezüglich der Kettenbrüche ist der folgende Satz:

Satz: Sei $\frac{p}{q}$ eine beliebige rationale Zahl, die

$$\left|\frac{p}{q} - x\right| < \frac{1}{2q^2}$$

erfüllt, dann ist $\frac{p}{q}$ ein Näherungsbruch des Kettenbruchs von x.

Anhang B

Zahlenbeispiel für DFT_q

Hier soll der Zusammenhang zwischen der DFT_q und der Darstellung dieser Operationen über die unitären Transformationen an einem Zahlenbeispiel vorgeführt werden. Es gilt allgemein, dass

$$DFT_q|a\rangle = \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \mathrm{e}^{i2\pi\frac{ac}{q}} |c\rangle \qquad \text{mit} \qquad |c\rangle = |c_{L-1}\rangle \ldots |c_1\rangle \ldots |c_0\rangle$$

über unitäre Transformationen in der folgenden Form dargestellt werden kann:

$$\Pi[(H_0 S_{0,1} \ldots S_{0,j-2} S_{0,j-1} S_{0,j}) \ldots (H_{j-2} S_{j-2,j-1} S_{j-2,j})(H_{j-1} S_{j-1,j})(H_j)|a\rangle].$$

$\Pi|a\rangle$ ist definiert als:

$$\Pi|a\rangle = \Pi|a_{L-1}, a_{L-2}, \ldots, a_1, a_0\rangle := |a_0, a_1, \ldots, a_{L-2}, a_{L-1}\rangle := \overline{|a\rangle}. \tag{B.1}$$

Dies bedeutet, dass Π die Bits in umgekehrter Reihenfolge anordnet.

Für das Zahlenbeispiel setzt man $L = 2$ $(q = 2^L \Rightarrow q = 2^2 = 4)$ und erhält damit $|a\rangle = |a_1\rangle|a_0\rangle$.
Hierfür soll gelten:

$$\frac{1}{\sqrt{4}} \sum_{c=0}^{3} \mathrm{e}^{i2\pi\frac{ac}{4}} |c\rangle = \Pi[H_0 S_{0,1} H_1 |a\rangle]. \tag{B.2}$$

Zunächst betrachtet man den linken Teil der Gleichung (B.2):

$$\frac{1}{\sqrt{4}} \sum_{c=0}^{3} \mathrm{e}^{i2\pi\frac{ac}{4}} |c\rangle = \frac{1}{\sqrt{4}} [|0\rangle + \mathrm{e}^{i2\pi\frac{a}{4}}|1\rangle + \mathrm{e}^{i2\pi\frac{2a}{4}}|2\rangle + \mathrm{e}^{i2\pi\frac{3a}{4}}|3\rangle].$$

Mit

$$\begin{aligned} |0\rangle &= |0\rangle|0\rangle \\ |1\rangle &= |0\rangle|1\rangle \\ |2\rangle &= |1\rangle|0\rangle \\ |3\rangle &= |1\rangle|1\rangle \end{aligned}$$

ergibt sich:

$$\frac{1}{\sqrt{4}}[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{a}{4}}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{2a}{4}}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{3a}{4}}|1\rangle|1\rangle].$$

Für den rechten Teil der Gleichung (B.2) ergibt sich:

$$\begin{aligned} |a_1\rangle|a_0\rangle = |a_1\rangle \otimes |a_0\rangle \quad \text{mit} \quad & |a_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle \\ & |a_0\rangle = \alpha_0|0\rangle + \beta_0|1\rangle \\ \text{und} \quad & |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \end{aligned}$$

also $|a_1\rangle|a_0\rangle = \alpha_1\alpha_0|0\rangle|0\rangle + \alpha_1\beta_0|0\rangle|1\rangle + \beta_1\alpha_0|1\rangle|0\rangle + \beta_1\beta_0|1\rangle|1\rangle$.
Die Anwendungen der Transformationen H und S werden im Folgenden einzeln dargestellt. Man beginnt mit

$$\begin{aligned} H_1|a\rangle = H_1|a_1\rangle|a_0\rangle &= \alpha_1\alpha_0\tfrac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|0\rangle) \\ &+ \alpha_1\beta_0\tfrac{1}{\sqrt{2}}(|0\rangle|1\rangle + |1\rangle|1\rangle) \\ &+ \beta_1\alpha_0\tfrac{1}{\sqrt{2}}(|0\rangle|0\rangle - |1\rangle|0\rangle) \\ &+ \beta_1\beta_0\tfrac{1}{\sqrt{2}}(|0\rangle|1\rangle - |1\rangle|1\rangle). \end{aligned} \tag{B.3}$$

Als nächstes wendet man auf Gleichung (B.3) die Transformation $S_{0,1}$ an. Wenn sich die Qubits im Zustand $|1\rangle|1\rangle$ befinden, erhält man einen zusätzlichen Phasenfaktor $\mathrm{e}^{i\frac{\pi}{2}} = i$:

$$\begin{aligned} S_{0,1}H_1|a\rangle = \tfrac{1}{\sqrt{2}} \quad [\quad & \alpha_1\alpha_0(|0\rangle|0\rangle + |1\rangle|0\rangle) + \alpha_1\beta_0(|0\rangle|1\rangle + i|1\rangle|1\rangle) \\ + \quad & \beta_1\alpha_0(|0\rangle|0\rangle - |1\rangle|0\rangle) + \beta_1\beta_0(|0\rangle|1\rangle - i|1\rangle|1\rangle) \quad]. \end{aligned}$$

Auf diesen Zustand wird nun noch H_0 angewandt. Somit ergibt sich

$$\begin{aligned} H_0S_{0,1}H_1|a\rangle = \tfrac{1}{\sqrt{4}} \quad [\quad & (\alpha_1\alpha_0 + \alpha_1\beta_0 + \beta_1\alpha_0 + \beta_1\beta_0)(|0\rangle|0\rangle) \\ + \quad & (\alpha_1\alpha_0 - \alpha_1\beta_0 + \beta_1\alpha_0 - \beta_1\beta_0)(|0\rangle|1\rangle) \\ + \quad & (\alpha_1\alpha_0 + i\alpha_1\beta_0 - \beta_1\alpha_0 - i\beta_1\beta_0)(|1\rangle|0\rangle) \\ + \quad & (\alpha_1\alpha_0 - i\alpha_1\beta_0 - \beta_1\alpha_0 - i\beta_1\beta_0)(|1\rangle|1\rangle) \quad]. \end{aligned}$$

Wie in (B.3) definiert gilt:

$$\begin{aligned}
\text{für} \quad a = 0 : |a\rangle = |0\rangle|0\rangle \quad &\rightarrow \quad \alpha_1 = 1; \alpha_0 = 1; \beta_1 = 0; \beta_0 = 0 \\
\Longrightarrow \quad H_0 S_{0,1} H_1 |a = 0\rangle \quad &= \quad \tfrac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + |1\rangle|0\rangle + |0\rangle|1\rangle + |1\rangle|1\rangle \right], \\
\\
\text{für} \quad a = 1 : |a\rangle = |0\rangle|1\rangle \quad &\rightarrow \quad \alpha_1 = 1; \alpha_0 = 0; \beta_1 = 0; \beta_0 = 1 \\
\Longrightarrow \quad H_0 S_{0,1} H_1 |a = 1\rangle \quad &= \quad \tfrac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{1}{4}\cdot 1}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{2}{4}\cdot 1}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{3}{4}\cdot 1}|1\rangle|1\rangle \right], \\
\\
\text{für} \quad a = 2 : |a\rangle = |1\rangle|0\rangle \quad &\rightarrow \quad \alpha_1 = 0; \alpha_0 = 1; \beta_1 = 1; \beta_0 = 0 \\
\Longrightarrow \quad H_0 S_{0,1} H_1 |a = 2\rangle \quad &= \quad \tfrac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{1}{4}\cdot 2}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{2}{4}\cdot 2}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{3}{4}\cdot 2}|1\rangle|1\rangle \right], \\
\\
\text{für} \quad a = 3 : |a\rangle = |1\rangle|1\rangle \quad &\rightarrow \quad \alpha_1 = 0; \alpha_0 = 0; \beta_1 = 1; \beta_0 = 1 \\
\Longrightarrow \quad H_0 S_{0,1} H_1 |a = 3\rangle \quad &= \quad \tfrac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{1}{4}\cdot 3}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{2}{4}\cdot 3}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{3}{4}\cdot 3}|1\rangle|1\rangle \right].
\end{aligned}$$

Zusammengefasst bedeutet dies

$$H_0 S_{0,1} H_1 |a\rangle = \frac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{1}{4}\cdot a}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{2}{4}\cdot a}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{3}{4}\cdot a}|1\rangle|1\rangle \right]. \tag{B.4}$$

Nach Definition (B.1) gilt für das hier betrachtete Zahlenbeispiel

$$\Pi|0\rangle|0\rangle = |0\rangle|0\rangle; \; \Pi|1\rangle|0\rangle = |0\rangle|1\rangle; \; \Pi|0\rangle|1\rangle = |1\rangle|0\rangle; \; \Pi|1\rangle|1\rangle = |1\rangle|1\rangle. \tag{B.5}$$

Damit ergibt sich für die Gleichung (B.4) unter Anwendung von (B.5)

$$\Pi[H_0 S_{0,1} H_1 |a\rangle] = \frac{1}{\sqrt{4}} \left[|0\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{1}{4}\cdot a}|0\rangle|1\rangle + \mathrm{e}^{i2\pi\frac{2}{4}\cdot a}|1\rangle|0\rangle + \mathrm{e}^{i2\pi\frac{3}{4}\cdot a}|1\rangle|1\rangle \right],$$

also

$$DFT_4|a\rangle = \frac{1}{\sqrt{4}} \sum_{c=0}^{3} \mathrm{e}^{i2\pi\frac{ac}{4}}|c\rangle = \Pi[H_0 S_{0,1} H_1 |a\rangle],$$

was zu zeigen war.

Anhang C

Programm zur Berechnung der Periodizität und der Wahrscheinlichkeitsverteilung

Das folgende Programm, geschrieben in der Programmiersprache C++, zeigt, wie die diskrete Fouriertransformation explizit berechnet werden kann (siehe auch Abbildungen 6.3.2 und 6.3.2).

```
#include <stdio.h>
#include <math.h>
#include <fstream.h>
#include <iostream.h>
#include <stdlib.h>
#include <Complex.h>

int main(int argc, char* argv[])
{
    ofstream   Result_fa, Result_fc;
    double     a, c, q, r, j, l;
    Complex    fa, fc;


    Result_fa.open ( "QuantumDFT.fa",  ios::out|ios::trunc);
    Result_fc.open ( "QuantumDFT.fc",  ios::out|ios::trunc);

    q = 512;      // ... fixed values
    r = 6;

    l = 3;        // ... arbitrary value



    for ( a=0; a<=q-1; a++ )
```

```
    {
        fa = Complex(     0    , 0);

        for ( j=0; j<=q/r-1; j++ )
        {
            if ( a == j*r + l )
            {
                fa = Complex(  sqrt(r/q), 0);

                cout << "\n a = " << a << " ,  j = " << j
                     << " ,  f(a) = " << fa << flush;
            }
        }

        Result_fa << a << " \t" << real(fa) << " \t"
                                << imag(fa) << " \n" << flush;
    }

    for ( c=0; c<=q-1; c++ )
    {
        fc = Complex(     0    , 0);

        l = 0;     // ... since we are not interested
                   //          in the constant part !


        for ( j=0; j<=q/r-1; j++ )
        {
            fc += Complex ( sqrt(r)/q * cos( 2.0*M_PI*(j*r+l)*c/q ),
                            sqrt(r)/q * sin( 2.0*M_PI*(j*r+l)*c/q ) );

        }

        Result_fc << c << " \t" << real(fc) << " \t"
                                << imag(fc) << " \n" << flush;
    }

}
```

Literaturverzeichnis

[1] J. J. Sakurai: *Modern Quantum Mechanics*, Revised Edition, Addison-Wesley Publishing Company, (1994).

[2] R.P. Feynman: *Simulating physics with computers*, Int. J. Theor. Phys. **21**, 467-488 (1982).

[3] R.P. Feynman: *Quantum mechanical computers*, Found. Phys. **16**, 507-531 (1986).

[4] C. H. Bennett: *Logical reversibility of computation*, IBM J. Res. Develop. **17**, 525-532 (1973).

[5] E. Fredkin, T. Toffoli: *Conservative Logic*, Int. J. Theor. Phys. **21**, 219-253 (1982).

[6] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, H. Weinfurter: *Elementary gates for quantum computation*, preprint quant-ph/9503016.

[7] C. H. Bennett: *Quantum Information and Computation*, Phys. Today **48**(10), 24 (1995).

[8] A. M. Steane: *Quantum computing*, Rept. Prog. Phys. **61**, 117-173 (1998), preprint quant-ph/9708022.

[9] D. P. DiVincenzo: *Quantum Computation*, Science **270**, 255 (1995).

[10] D. Gottesmann: *Theory of fault-tolerant quantum computation*, Phys. Rev. A **57**, 127-137 (1998).

[11] D. Leibfried, T. Schätz: *Ein atomarer Abakus*, Physik Journal, 23-20 (Januar 2004).

[12] J. Preskill: *Reliable Quantum Computers*, preprint quant-ph/9705031.

[13] J. Preskill: *Fault-Tolerant Quantum Computation*, preprint quant-ph/9712048.

[14] E. Rieffel, W. Polak: *An Introduction to Quantum Computing for Non-Physicists*, preprint quant-ph/9809016.

[15] R. Folman et al.: *Controlling Cold Atoms using Nanofabricated Surfaces: Atom Chips*, Phys. Rev. Lett. **84**, 4749 (2000).

[16] H. C. Nägerln et. al.: *Laser adressing of individual ions in a linear ion trap*, Phys. Rev. A **60**, 145 (1999).

[17] J. Reichel, W. Hansel, T. W. Hänsch: *Atomic Micromanipulation with Magnetic Surface Traps*, Phsy. Rev. Lett. **83**, 3398 (1999).

[18] C. A. Sackett et al.: *Experimental entanglement of four particles*, Nature **404**, 256 (2000).

[19] A. Steane et al.: *Speed of ion trap quantum information processors*, preprint quant-ph/0003087.

[20] I. Cirac, P. Zoller: *Quantum computations with cold trapped ions*, Phys. Rev. Lett. **74**, 4091 (1995).

[21] C. Monroe et al.: *Demonstration of a fundamental quantum logic gate*, Phys. Rev. Lett. **75**, 4714 (1995).

[22] Universität Innsbruck: http://www.uibk.ac.at (R. Blatt).

[23] Q. A. Turchette et al.: *Measurement of conditional phase shifts for quantum logic*, Phys. Rev. Lett. **75**, 4710 (1995).

[24] H.-J. Briegel, I. Cirac, P. Zoller: *Quantencomputer*, Phys. Bl. **55**, 37 (1999).

[25] I. Cirac, P. Zoller: *A scalable quantum computer with ions in an array of microtraps*, Nature **404**, 579 (2000).

[26] T. Calarco, J. I. Cirac, P. Zoller: *Entangling ions in arrays of microscopic traps*, preprint quant-ph/0010105.

[27] Physik in unserer Zeit: *Hundert Jahre Quantenmechanik*, **6** (2000).

[28] D. G. Cory et al.: *Ensemble quantum computing by NMR spectroscopy*, Proc. Natl. Acad. Sci. USA **94**, 1634-1639 (1997).

[29] N. A. Gershenfeld. I. L. Chuang: *Bulk spin resonance quantum computation*, Science **275**, 350 (1997).

[30] R. Marx, A. F. Fahmy, J. M. Myers, W. Bermel, S. J. Glaser: *Approaching five-bit NMR quantum computing*, Phys. Rev. A **62**, 012310-1–012310-8 (2000).

[31] I. L. Chuang et al.: *Experimental realization of a quantum algorithm*, Nature **393**, 143-146 (1998).

[32] J. A. Jones, M. Mosca: *Implementation of a Quantum Algorithm to Solve Deutsch's Problem on a Nuclear Magnetic Resonance Quantum Computer*, preprint quant-ph/9801027.

[33] I. L. Chuang, N. Gershenfeld, M. Kubinec: *Experimental Implementation of Fast Quantum Searching*, Phys. Rev. Lett. **80**, 3408 (1998).

[34] S. L. Braunstein et al.: *Separability of very noisy mixed states and implications for NMR quantum computing*, preprint quant-ph/9811018.

[35] B. E. Kane: *A Silicon-based Nuclear Spin Quantum Computer*, Nature **393**, 133 (1998).

[36] D. Loss, D. P. DiVincenzo: *Quantum computation with quantum dots*, Phys. Rev. A **57**, 120 (1998).

[37] A. Shnirman, G. Schön, Z. Hermon: *Quantum Manipulations of Small Josephson Junctions*, Phys. Rev. Lett. **79**, 2371 (1997).

[38] Y. Makhlin, G. Schön, A. Shirman: *Josephson-Junction Qubits with Controlled Couplings*, Nature **398**, 305 (1999).

[39] T. Nakamura, Y. A. Pashkin, J. S. Tsai: *Coherent control of macroscopic quantum states in a single-Cooper-pair box*, Nature **398**, 786 (1999).

[40] T. Toffoli: *Reversible Computing*, MIT Laboratory for Computer Science Tech. Rpt. TM-151, Feb. (1980); Kurzfassung in Springer-Verlag Lecture Notes in Computer Science No. 85, ed. by J.W. de Bakker and J. van Leeuwen, pp. 632-644 (1980).

[41] P. Benioff: *The Computer as a Physical System: A Microscopic Quantum Mechanical Hamiltonian Model of Computers as Represented by Turing Machines*, Journal of Statistical Physics, Vol. **22**, 563-591 (1980).

[42] P. Benioff: *Quantum mechanical Hamiltonian models of Turing machines*, J. Stat. Phys. **29**, 515-546 (1982).

[43] D. Deutsch: *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer*, Proc. Roy. Soc. Lond. A **400**, 97-117 (1985).

[44] S. Lloyd: *Universal quantum simulators*, Science **273**, 1073-1078 (1996).

[45] D. Deutsch: *Quantum computational networks*, Proc. Roy. Soc. Lond. A **425**, 73-90 (1989).

[46] R. Silverman: *The multiple polynomial quadratic sieve*, Mathematical Computing Vol. **48**, 329-339, (1987).

[47] A. Lenstra, M. Manasse, J. Pollard: *The Number Field Sieve*, Proc. of the 22nd ACM Symposium on the Theory of Computing, 564-572 (1990).

[48] A. Lenstra, H. Lenstra: *The development of the number field sieve*, Springer-Verlag (1993).

[49] R. Cleve, A. Ekert, C. Macchiavello, M. Mosca: *Quantum Algorithms Revisited*, preprint quant-ph/9708016.

[50] D. Deutsch, R. Josza: *Rapid solution of problems by quantum computation*, Proc. Roy. Soc. Lond. A **439**, 553-558 (1992).

[51] A. Berthiaume, G. Brassard: *The quantum challenge to structural complexity theory*, Proc. of the 7th Annual Structure in Complexity Theory Conference (IEEE Computer Society Press, Los Alamitos, CA), 132-137 (1992).

[52] A. Berthiaume, G. Brassard: *Oracle quantum computing*, Proc. of the Workshop on Physics of Computation: PhysComp '92 (IEEE Computer Society Press, Los Alamitos, CA), 60-62 (1992).

[53] E. Bernstein, U. Vazirani: *Quantum complexity theory*, Proc. of the 25th Annual ACM Symp. of Theory of Computing (ACM, New York), 11-20 (1993).

[54] D. Simon: *On the power of quantum computation*, Proc. 35th Annual Symp. on Foundations of Computer Science (IEEE Computer Society Press, Los Alamitos), 116-123 (1994).

[55] R. Cleve, A. Ekert, L. Henderson, C. Macchiavello, M. Mosca: *On quantum algorithms*, preprint quant-ph/9903061.

[56] R. Cleve, J. Watrous: *Fast parallel circuits for the quantum Fourier transform*, preprint quant-ph/0006004.

[57] A. Ekert, R. Jozsa: *Quantum computation and Shor's factoring algorithm*, Reviews of Modern Physics, Vol. **68**, No. 3, July, 733-753, (1996).

[58] Yu. I. Manin: *Classical Computing, Quantum Computing, and Shor's Factoring Algorithm*, preprint quant-ph/9903008.

[59] P. W. Shor: *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, Proc. 35th Annual Symp. on Foundations of Computer Science, Santa Fe, IEEE Computer Society Press (1994); revised version preprint quant-ph/9508027.

[60] P. W. Shor: *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*, Proc. 35th Annual Symp. on Foundations of Computer Science, Santa Fe, IEEE Computer Society Press, 124-134 (1994).

[61] P. W. Shor: *Introduction to Quantum Algorithms*, preprint quant-ph/0005003.

[62] D. Coppersmith: *An approximate Fouriertransform useful in quantum factoring*, IBM Research Report RC 19642 (1994).

[63] R. Jozsa: *Quantum Algorithms and the Fourier Transform*, Proc. Roy. Soc. Lond. A, 323-337 (1998).

[64] L. K. Grover: *A Fast Quantum Mechanical Algorithm for Database Search*, Proc. of the 28th Annual ACM Symp. on the Theory of Computing, 212-219, (1996).

[65] L. K. Grover: *A fast quantum mechanical algorithm for estimating the median*, preprint quant-ph/9607024.

[66] L. K. Grover: *Quantum mechanics helps in searching for a needle in a haystack*, Phys. Rev. Lett. **79**, 325-328 (1997).

[67] C. Zalka: *Grover's quantum searching algorithm is optimal*, preprint quant-ph/9711070.

[68] C. Durr, P. Hoyer: *A Quantum Algorithm for Finding the Minimum*, preprint quant-ph/9607014.

[69] A. Yu Kitaev: *Quantum measurements and the Abelian stabilizer problem*, preprint quant-ph/9511026.

[70] R. Jozsa: *Characterizing classes of functions computable by quantum parallelism*, Proc. Roy. Soc. Lond. A, 563-574 (1991).

[71] F. L. Bauer: *Entzifferte Geheimnisse*, Springer-Verlag.

[72] D. Beckmann, A. N. Chari, S. Devabhaktuni, John Preskill: *Efficient networks for quantum factoring*, preprint quant-ph/9602016.

[73] V. Vedral, A. Barenco, A. Ekert: *Quantum Networks for Elementary Arithmetic Operations*, preprint quant-ph/9511018.

[74] Hardy, Wright: *Einführung in die Zahlentheorie*, R. Oldenbourg München 1958.