

Computational Intelligence

LEHRBUCH

Matthias Homeister

# Quantum Computing verstehen

Grundlagen – Anwendungen –  
Perspektiven

*6. Auflage*

 Springer Vieweg

---

# Computational Intelligence

## Reihe herausgegeben von

Barbara Hammer, Bielefeld, Deutschland

Gabriele Kern-Isberner , Dortmund, Deutschland

Sanaz Mostaghim , Magdeburg, Deutschland

Die Reihe Computational Intelligence wird herausgegeben von Prof. Dr. Barbara Hammer, Prof. Dr. Gabriele Kern-Isberner und Prof. Dr. Sanaz Mostaghim.

Aus dem Bereich der „Künstlichen Intelligenz“ bietet die Reihe breitgefächertes Wissen von den Grundlagen bis in die Anwendung, herausgegeben von namhaften Vertretern ihres Faches.

Computational Intelligence hat das weitgesteckte Ziel, das Verständnis und die Realisierung intelligenten Verhaltens voranzutreiben. Die Bücher der Reihe behandeln Themen aus den Gebieten Künstliche Intelligenz, Softcomputing, Robotik, Neuro- und Kognitionswissenschaften. Es geht sowohl um die Grundlagen (in Verbindung mit Mathematik, Informatik, Ingenieurs- und Wirtschaftswissenschaften, Biologie und Psychologie) wie auch um Anwendungen (z.B. Hardware, Software, Webtechnologie, Marketing, Vertrieb, Entscheidungsfindung). Hierzu bietet die Reihe Lehrbücher, Handbücher und solche Werke, die maßgebliche Themengebiete kompetent, umfassend und aktuell repräsentieren.

Weitere Bände in der Reihe <https://link.springer.com/bookseries/12572>

---

Matthias Homeister

# Quantum Computing verstehen

Grundlagen – Anwendungen –  
Perspektiven

6., erweiterte und überarbeitete Auflage



**Springer** Vieweg

Matthias Homeister  
Fachbereich Informatik und Medien  
Technische Hochschule Brandenburg  
Brandenburg, Deutschland

ISSN 2522-0519                      ISSN 2522-0527 (electronic)  
Computational Intelligence  
ISBN 978-3-658-36433-5              ISBN 978-3-658-36434-2 (eBook)  
<https://doi.org/10.1007/978-3-658-36434-2>

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

© Springer Fachmedien Wiesbaden GmbH, ein Teil von Springer Nature 2005, 2008, 2013, 2015, 2018, 2022  
Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von allgemein beschreibenden Bezeichnungen, Marken, Unternehmensnamen etc. in diesem Werk bedeutet nicht, dass diese frei durch jedermann benutzt werden dürfen. Die Berechtigung zur Benutzung unterliegt, auch ohne gesonderten Hinweis hierzu, den Regeln des Markenrechts. Die Rechte des jeweiligen Zeicheninhabers sind zu beachten.

Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen. Der Verlag bleibt im Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutionsadressen neutral.

Planung: David Imgrund + Leonardo Milla

Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.

Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

*Für Kathrin und Paul.*

# Inhaltsverzeichnis

<b>Vorwort</b>	<b>XI</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Eine neue Art des Rechnens . . . . .	1
1.2 Über dieses Buch . . . . .	8
<b>2 Vom Bit zum Quantenregister</b>	<b>9</b>
2.1 Was ist eine Berechnung? . . . . .	10
2.1.1 Die Turingmaschine . . . . .	13
2.1.2 Schaltkreise . . . . .	14
2.1.3 Der Sprung in die Quantenwelt: Schrödingers Katze .	17
2.2 Das Quantenbit . . . . .	20
2.3 Rechenschritte auf einem Quantenbit . . . . .	23
2.4 Der erste Algorithmus: Ein Zufallsgenerator . . . . .	26
2.5 Quantenregister . . . . .	28
2.6 Der zweite Algorithmus: Das Problem von Deutsch . . . . .	33
2.7 Die Rolle des Tensorprodukts . . . . .	37
2.8 Das Messen von Quantenregistern . . . . .	44
2.9 Noch einmal das Problem von Deutsch . . . . .	50
2.10 Bestandsaufnahme: Die drei Prinzipien des Quantum Computing	51
2.11 Verschränkung . . . . .	53
2.12 Die Hadamard-Transformation und mehrere Bits . . . . .	59
2.13 Der Algorithmus von Deutsch-Jozsa . . . . .	62
<b>3 Vom Quantenregister zum Quantenschaltkreis</b>	<b>67</b>
3.1 Laufzeit . . . . .	68
3.2 Klassische Schaltkreise und Komplexität . . . . .	74
3.3 Quantengatter und Quantenschaltkreise . . . . .	76
3.4 Quantenbits kopieren: Das No-Cloning-Theorem . . . . .	81
3.5 Umkehrbare Berechnungen . . . . .	83
3.6 Unterscheidbare Zustände . . . . .	92
3.7 Gestörte Berechnungen . . . . .	95
<b>4 Hilfsmittel aus der Theoretischen Informatik</b>	<b>99</b>
4.1 Komplexitätsklassen . . . . .	99
4.2 Randomisierte Algorithmen . . . . .	104
4.2.1 Mit dem Zufall rechnen . . . . .	104
4.2.2 Ein Primzahltest . . . . .	105

4.2.3	Probabilistische Komplexitätsklassen . . . . .	107
4.3	Unlösbare Probleme? NP-Vollständigkeit . . . . .	111
4.4	Quantenkomplexitätstheorie . . . . .	116
4.5	Die Churchsche These . . . . .	119
<b>5</b>	<b>Teleportation und dichte Kodierung</b>	<b>123</b>
5.1	Quantenteleportation . . . . .	125
5.2	Dichte Kodierung . . . . .	130
5.3	Verschränkte Bits . . . . .	132
<b>6</b>	<b>Suchen</b>	<b>135</b>
6.1	Die Nadel im Heuhaufen . . . . .	136
6.2	Die Grover-Iteration . . . . .	138
6.3	Eine geometrische Veranschaulichung . . . . .	144
6.4	Varianten der Quantensuche . . . . .	151
6.4.1	Suche nach einer von mehreren Lösungen . . . . .	151
6.4.2	Suche bei unbekannt vielen Lösungen . . . . .	153
6.4.3	Die Suche nach dem Minimum . . . . .	154
6.4.4	Zählen . . . . .	157
6.5	Anwendungen von Grovers Algorithmus . . . . .	157
6.6	Grovers Algorithmus ist von optimaler Größenordnung . . . . .	159
6.7	Folgen für die Fähigkeiten von Quantencomputern . . . . .	164
<b>7</b>	<b>Geheime Botschaften</b>	<b>167</b>
7.1	Alice, Bob und Eve . . . . .	168
7.2	Quantenverschlüsselung: das BB84-Protokoll . . . . .	172
7.3	Lauschstrategien . . . . .	180
7.4	Quantenverschlüsselung mit Verschränkung . . . . .	185
<b>8</b>	<b>Klassische Verschlüsselungen knacken: Primfaktorzerlegung</b>	<b>191</b>
8.1	Faktorisierung und Verschlüsselung: RSA-Kryptographie . . . . .	192
8.2	Die Suche nach Perioden . . . . .	197
8.3	Die schnelle Fouriertransformation . . . . .	204
8.4	Die Quanten-Fouriertransformation . . . . .	212
8.5	Simons Algorithmus . . . . .	216
8.6	Shors Algorithmus . . . . .	221
8.7	Jenseits von Shor . . . . .	229
<b>9</b>	<b>Fehler korrigieren</b>	<b>235</b>
9.1	Dekohärenz und Fehler auf Quantenbits . . . . .	235
9.2	Klassische Fehlerkorrektur . . . . .	240
9.3	Herausforderungen bei der Korrektur von Quantenbits . . . . .	240
9.4	Qubits gegen Fehler sichern . . . . .	241
9.4.1	Bitflip-Code mit Syndrom: Korrektur des Codewords . . . . .	241
9.4.2	Der Bitflip-Code: Korrektur des Datenbits . . . . .	243
9.4.3	Korrektur von Phaseflips . . . . .	244
9.5	Shors 9-Qubit-Code . . . . .	245
9.6	Ausblick . . . . .	248



<b>10 Quantenhardware</b>	<b>251</b>
10.1 Anforderungen . . . . .	251
10.2 Photonen . . . . .	253
10.2.1 Mach-Zehnder-Interferometer . . . . .	253
10.2.2 Zufallszahlen . . . . .	256
10.2.3 Kryptographie . . . . .	257
10.3 Kernspinresonanz . . . . .	261
10.4 Ionenfallen . . . . .	263
10.5 Einwegberechnungen mit Clusterzuständen . . . . .	264
10.6 Supraleiter . . . . .	267
<b>11 Ausblick: Optimieren</b>	<b>271</b>
11.1 Adiabatische Quantencomputer . . . . .	271
11.2 Quantum Annealing . . . . .	274
<b>12 Zur Geschichte der Quantenmechanik</b>	<b>281</b>
12.1 Max Planck: das Quantum der Wirkung . . . . .	281
12.2 Albert Einstein: Spukhafte Fernwirkung . . . . .	283
12.3 Niels Bohr: Kopenhagen . . . . .	284
12.4 Werner Heisenberg: Ein großes Quantenei . . . . .	287
12.5 Erwin Schrödinger: Katzen und Wellen . . . . .	289
12.6 Schlaglichter der Geschichte des Quantencomputers . . . . .	290
<b>A Mathematische Grundlagen</b>	<b>293</b>
A.1 Komplexe Zahlen . . . . .	293
A.2 Vektorräume . . . . .	295
A.2.1 Was sind Vektorräume? . . . . .	295
A.2.2 Basen und Unterräume . . . . .	297
A.2.3 Winkel und Abstände in einem Vektorraum . . . . .	298
A.2.4 Projektionen . . . . .	300
A.3 Matrizen . . . . .	301
A.4 Kombinatorik und Wahrscheinlichkeit . . . . .	303
A.5 Ganze Zahlen . . . . .	305
A.5.1 Teiler und Vielfache . . . . .	305
A.5.2 Modulares Rechnen . . . . .	305
A.5.3 Zur Division . . . . .	307
<b>B Lösungen ausgewählter Übungsaufgaben</b>	<b>309</b>
<b>Literatur</b>	<b>320</b>
<b>Symbole und Abkürzungen</b>	<b>329</b>
<b>Quantengatter</b>	<b>330</b>
<b>Namen- und Sachwortverzeichnis</b>	<b>331</b>

# Vorwort

Zwei wissenschaftliche Revolutionen prägten die erste Hälfte des 20. Jahrhunderts. Zum einen legten Pioniere wie Konrad Zuse, Alan Turing und John von Neumann die Grundlagen für den Bau der ersten praktikablen Rechenmaschinen. Zum anderen stürzte das klassische Weltbild der Physik, seit den Tagen Newtons erweitert, aber kaum verändert, in sich zusammen. Um den Aufbau der Materie zu verstehen, wurde eine radikal neue Theorie geschaffen. Die Quantenmechanik veränderte unsere Auffassung von der Realität so sehr, dass auch viele ihrer Schöpfer vor den Konsequenzen zurückschreckten.

Diese wissenschaftlichen Revolutionen zogen sehr schnell technische nach sich. Wie sehr der Computer unsere heutige Gesellschaft, unser Welt- und Menschenbild prägt, steht jedem vor Augen. Weniger bewusst ist vielen, dass die Quantenmechanik unseren Alltag ebenso beeinflusst. Erst die quantenmechanische Beschreibung des Atoms machte es möglich, Halbleiter und den Laser zu entwickeln; das Transistorradio, der CD-Spieler und moderne Computerhardware sind Folgen der Quantenmechanik.

In den letzten Jahrzehnten wurden diese beiden Wissenschaften zusammengeführt, es entstand ein neuer interdisziplinärer Zweig namens *Quantum Computing*. Das Ziel ist, Quantencomputer zu bauen, Quantenalgorithmen zu entwickeln und zu untersuchen, welche Konsequenzen die Quantenmechanik für die Informationsübertragung hat. Es gibt verschiedene Auslöser für diese Entwicklung, am wichtigsten sind die beiden folgenden:

1. Die grundlegenden Bauteile unserer Rechner werden immer kleiner. Wenn diese Entwicklung im aktuellen Tempo fortschreitet, wird es in nicht allzu ferner Zeit Bauteile von der Größe eines einzelnen Atoms geben. Für Hardwarekomponenten dieser Größenordnung gelten die Gesetze der klassischen Physik nicht mehr, und man muss sich auf die Quantenmechanik einlassen.
2. Computer, welche die Gesetze der Quantenmechanik für sich ausnutzen, können Aufgaben erledigen, die für herkömmliche Rechner unmöglich sind. Dazu gehören absolut abhörsichere Nachrichtenübertragung, die Teleportation von Information, das Erzeugen echter Zufallszahlen und das Knacken von Verschlüsselungsmethoden, die zurzeit als sicher gelten. Dabei geht es um Probleme, die herkömmliche Rechner grundsätzlich nicht lösen können, egal wie sich ihre Leistungsfähigkeit steigern wird.

Zudem besteht die Möglichkeit, dass heuristische Quantenverfahren Optimierungsprobleme und Aufgaben aus dem Machine Learning oder der Künstlichen Intelligenz erheblich beschleunigen werden.

Wann Quantencomputer praxisrelevante Probleme besser lösen werden, als unsere herkömmlichen Computer, lässt sich noch nicht sagen.<sup>1</sup> Allerdings sind die Fortschritte der letzten Jahre so beeindruckend, dass neben staatlichen Institutionen und traditionellen Hightech-Konzernen wie Google oder IBM nun auch Automobilhersteller, Finanzdienstleister und Versicherungen hohe Summen in die Forschung und Entwicklung rund um den Quantencomputer investieren. Welche Hoffnungen dabei zunächst enttäuscht werden und in welchen Bereichen der Quantencomputer schon bald einen wirtschaftlichen Mehrwert schaffen wird, ist ohne die berühmte Glaskugel schwer zu sagen. Ohne Zweifel sind wir Zeugen einer wissenschaftlichen und technologischen Revolution, Realismus und kritische Distanz zu Hype-Versprechungen erscheinen trotzdem angeraten.

Am weitesten ist bisher die Umsetzung von Verschlüsselungsverfahren fortgeschritten. Verschiedene Firmen bieten seit vielen Jahren Quantenkryptographiesysteme an und zum Beispiel China plant den Bau eines umfangreichen Quantennetzwerks. Und unabhängig von dem praktischen Einsatz, hat sich schon längst unsere Auffassung vom Rechnen selbst, von dem, was ein Computer prinzipiell leisten kann, verändert. Jeder Informatikstudent lernt im ersten Semester die Churchsche These kennen. Ihre moderne Fassung lautet: Lässt sich ein Problem von irgendeiner physikalisch realisierbaren Rechenmaschine effizient berechnen, so lässt es sich von einer (randomisierten) Turingmaschine effizient berechnen. Das stimmt so nicht mehr. Quantencomputer sind physikalisch realisierbar und sie können einige Probleme lösen, an denen herkömmliche Computer nach dem Stand der Wissenschaft scheitern müssen.

Aber auch für das Verständnis der physikalischen Realität werden Informationsprozesse immer wichtiger. Die Forschung rund um den Quantencomputer hat nicht nur das Potential, eine neue Technologie hervorzubringen, sondern auch unser Weltbild zu verändern. Information könnte ein grundlegendes Prinzip sein als die klassischen physikalischen Begriffe Materie oder Energie.

Wir sind also Zeugen der Entstehung einer neuen Wissenschaft, die sich zuletzt rasant entwickelt hat. Kenntnisse in einigen ihrer Bereiche haben dadurch eine geringe Halbwertszeit. Was ein Quantencomputer ist und was er kann, wird in diesem Buch an Hand von Quantenalgorithmen erläutert, also an konkreten Rechenverfahren. Dieser theoretische Zugang bereitet den Leser auch auf künftige Entwicklungen vor. Man denke daran, dass die Ideen Zuses, Turings und von Neumanns heute nichts von ihrer Gültigkeit verloren haben und aus der Antike stammende Algorithmen noch immer praktisch eingesetzt werden.

Leider trauen sich viele Interessierte nicht zu, sich mit diesem neuen Thema näher auseinander zu setzen. Quantenmechanik ist ein voraussetzungsreiches Gebiet, in das man sich nicht mal eben so einarbeitet. Um jedoch zu verstehen, wie Quantencomputer rechnen, genügen zunächst wenige quantenmechanische Prinzipien. Es sind einfache Prinzipien, auch wenn sie naturgemäß unanschaulich sind. Dieses Buch stellt sie so anwendungs-

---

<sup>1</sup>Zu Googles Veröffentlichung aus 2019 zum Thema Quantenvorteil siehe Abschnitt 12.6.

orientiert wie möglich dar und ist für Leser ohne besondere Vorkenntnisse geschrieben.

Ein typischer Leser dieses Buches könnte ein Informatikstudent nach den Grundvorlesungen sein, aber genauso jede andere Person mit mathematischen Grundkenntnissen. Der Leser wird so schnell wie möglich die Arbeitsweise von Quantenrechnern verstehen können und die wichtigen Algorithmen kennen lernen. Die fortgeschrittenen Abschnitte der einzelnen Kapitel vermitteln vertieftes Wissen und bereiten auf die Forschungsliteratur vor. Um dieses Buch für Leser ohne Informatikkenntnisse lesbar zu machen, werden wichtige Begriffe, wie Berechnung oder Algorithmus, umfassend eingeführt. Lesern, deren Mathematikkenntnisse aus Schule oder Studium verblasst sind, hilft ein Abschnitt im Anhang, der Themen wie komplexe Zahlen, Vektorräume und Matrizen behandelt.

*Zur 6. Auflage:* Für die aktuelle Auflage wurde das Buch durchgehend aktualisiert und an verschiedenen Stellen die Darstellung verbessert. Vollständig überarbeitet und verbessert wurde Kapitel 4. Der Abschnitt zum adiabatischen Quantencomputer wurde in das kurze Ausblickskapitel 11 verlagert. Deutlich erweitert wurde im Zuge dessen das Thema Quantum Annealing. Es gibt einige neue Übungsaufgaben; auf der Homepage zum Buch finden Sie eine Übersicht über Nummerierungsänderungen im Vergleich zu Vorgängerauflagen.

Brandenburg an der Havel, 2022

Matthias Homeister

## Danksagung

Viele Menschen halfen bei der Entstehung dieses Buches, ich möchte mich an dieser Stelle für alle Hinweise bedanken. Namentlich danke ich Ingo Boersch, Dagmar Bruß, Carsten Damm, Christoph Dürr, Martin Gorbahn, Rolf Socher, Johannes Summhammer, Michael Syrjakow, Maike Tech, Philip Walther, Harald Weinfurter, Stephan Waack, Daniel Wolf und insbesondere Hecke Schrobdsdorff, dessen unermüdliche Unterstützung als Korrektor und Gesprächspartner für das Gelingen dieses Buches wesentlich war. Cornelia Caspary danke ich für die wunderbaren Illustrationen. Für viele Hinweise und eine sehr engagierte Zusammenarbeit danke ich dem Herausgeber Prof. Dr. Wolfgang Bibel, Dr. Reinald Klockenbusch vom Vieweg Verlag sowie Herrn Bernd Hansemann, Herrn David Imgrund, Herrn Leonardo Milla und Frau Sybille Thelen von Springer-Vieweg.

# 1 Einleitung

*Eine neue wissenschaftliche Wahrheit pflegt sich nicht in der Weise durchzusetzen, dass ihre Gegner überzeugt werden und sich als belehrt erklären, sondern vielmehr dadurch, dass die Gegner allmählich aussterben, und dass die heranwachsende Generation von vornherein mit der Wahrheit vertraut gemacht ist.*  
Max Planck

Diese Einleitung vermittelt einen ersten Eindruck vom Nutzen der Quantenmechanik für die Informatik. Zur Orientierung geben die Randbemerkungen das Kapitel an, in dem das angesprochene Thema detailliert behandelt wird; wir folgen dabei nicht streng dem Inhaltsverzeichnis.

## 1.1 Eine neue Art des Rechnens

Unsere heutigen Computer, so verzwickelt ihre Technologie auch sein mag, funktionieren letztlich nach den gleichen Grundprinzipien wie die frühen Rechenmaschinen. Das soll nicht heißen, seit Konrad Zuses mechanischer Binärziffernrechenmaschine Z1 von 1938 oder seit der Difference Engine, die Charles Babbage zu Anfang des 19. Jahrhunderts entworfen hat, seien keine Fortschritte gemacht worden. Die Computertechnologie entwickelt sich rasant, und neue Architekturen führen zu immer schnelleren und leistungsfähigeren Rechnern. Trotzdem kann man die Etappen des Wegs vom Zahnrad über Relais, Röhre und Transistor zum miniaturisierten Schaltkreis als verschiedene technische Umsetzungen der gleichen Idee betrachten.

Ein Schritt von ganz anderer Reichweite wird mit der Entwicklung von Quantenrechnern getan, eine ganz neue Art des Rechnens wird hier realisiert. Es ist fast, als würden für Quantencomputer andere Naturgesetze gelten. Das ist natürlich so nicht korrekt, allerdings gelten in der Welt der

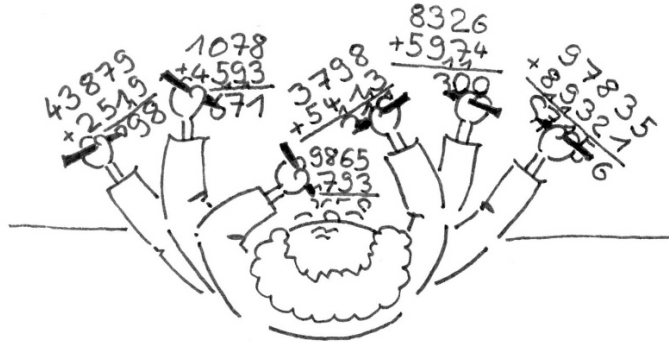


Abbildung 1.1: So ein klassischer Parallelrechner hat mit einem Quantencomputer wenig gemeinsam

kleinsten Teilchen Gesetze, die uns sehr fremd und ungewöhnlich vorkommen. Quantencomputer verhalten sich zu unseren heutigen Rechnern, wie die Quantenphysik zur klassischen Physik in der Nachfolge Newtons. Darum werden wir immer dann den Begriff *klassisch* verwenden, wenn es um die heutigen Rechner geht.

## Kapitel 2: Grundlagen

Wie funktioniert er also, der Quantencomputer? Diese Frage wollen wir so schnell wie möglich beantworten, denn die Antwort darauf ist die Voraussetzung für die andere, ebenso wichtige Frage: Was kann der Quantencomputer tun? Ein Stichwort werden viele Leser schon kennen: die Rede ist vom Quantenparallelismus. Während in einem klassischen Computer ein Bit entweder auf 0 oder auf 1 gesetzt ist, kann ein *Quantenbit* metaphorisch ausgedrückt beide Werte gleichzeitig annehmen. Man spricht von *Superposition*.<sup>1</sup> Noch deutlicher wird der Unterschied bei mehreren Bits: ein klassischer Rechner kann mit  $n$  Bits  $2^n$  verschiedene Zahlen darstellen, zu jedem Zeitpunkt aber nur eine davon speichern. Ein Quantenrechner kann mit ebenso vielen Quantenbits  $2^n$  Zahlen *gleichzeitig* darstellen. Und mehr noch, ein Quantenalgorithmus kann in einem einzelnen Rechengang auf allen möglichen Eingaben gleichzeitig rechnen, und wir erhalten eine Superposition aller Ergebnisse.

Trotzdem sind Quantencomputer etwas ganz anderes als Parallelrechner. In vielen Bereichen werden heute Computer eingesetzt, die mit mehreren Prozessoren ausgestattet sind und mehrere Rechnungen gleichzeitig ausführen, siehe Abbildung 1.1. Mit solchen Rechnern hat ein Quantencomputer keine Ähnlichkeit. Zunächst können wir auf die erwähnte Superposition von Lösungen nicht so einfach zugreifen. Die Welt der Quantenmechanik ist so empfindlich, dass der Versuch, ein Quantenbit zu lesen, im Allgemeinen

<sup>1</sup>Das ist bis hierhin nur eine Veranschaulichung. Was unter einer Superposition zu verstehen ist, werden wir in Kapitel 2 sehen.

seinen Zustand beeinflusst. Wir müssen dazu etwas tun, was *Messen* eines Quantenbits genannt wird und für uns eine große Rolle spielen wird. Beim Messen wird die Superposition zerstört, und wir bekommen, mehr oder weniger zufällig, eines der Rechenergebnisse. Wir erfahren noch nicht einmal, zu welcher der gleichzeitig ausgeführten Berechnungen es gehört. Man muss geschickt vorgehen, um aus einer solchen Superposition die gewünschte Information zu erhalten. Dabei spielt die sogenannte Interferenz eine wichtige Rolle.

Die vielleicht faszinierendste Eigenheit von Quantenbits nennt sich *Verschränkung*. Zwei verschränkte Quantenbits können sich nämlich auf eine eigentümliche Weise beeinflussen, auch wenn Sie sich an weit entfernten Orten befinden. Verändert man nun das erste Bit, kann gleichzeitig das zweite Bit beeinflusst werden. Albert Einstein nannte dieses Phänomen *spukhafte Fernwirkung* und mochte diese Konsequenz der Quantenmechanik nicht hinnehmen. Heutzutage wird Verschränkung praktisch genutzt und wird uns im Laufe des Buches immer wieder begegnen.

Nach Science Fiction könnte zunächst klingen, dass man den Zustand eines Quantenbits *teleportieren* kann. Mit den Hilfsmitteln aus Kapitel 2 lässt sich das Prinzip jedoch leicht erklären und die ersten Teleportationen wurden schon 1997 durchgeführt. Im Jahr 2017 gelang eine Teleportation von der Erdoberfläche zu einem Satelliten und in 2021 wurde auf diese Weise eine Distanz von 4600 km überwunden. Eine so geartete Kommunikation zwischen Erdoberfläche und Satellit wäre für ein zukünftiges Quanteninternet nutzbar und auch für den Bau von Quantencomputern kann Teleportation wichtige Aufgaben erfüllen. Mit der Teleportation eng verbunden ist die Möglichkeit der *dichten Kodierung*. Für die klassische Informationstheorie ist das Bit das elementare Maß für Information. Dagegen lassen sich mit einem einzelnen Quantenbit scheinbar zwei klassische Bits übertragen.

Ein Algorithmus mit einem großen Spektrum von Anwendungen wurde 1996 von Lov Grover veröffentlicht, der für die Bell Labs von Lucent Technologies in New Jersey arbeitet. Er hat festgestellt, dass Quantencomputer sehr viel schneller suchen können als klassische Computer. Versetzen wir uns in folgende Situation: Wir benötigen aus einer Datenbank einen bestimmten Eintrag. Die Einträge sind jedoch völlig unstrukturiert abgelegt, nicht nach einem für uns relevanten Kriterium sortiert. Es ist so, als hätten wir eine New Yorker Telefonnummer und müssten herausbekommen, auf wen der Anschluss angemeldet ist. Wir haben auch schon mehrmals die Nummer gewählt, aber nie hat sich jemand gemeldet. Uns bleibt nun nichts anderes übrig, als im Telefonbuch Anschluss für Anschluss durchzugehen. Hat das Telefonbuch eine Million Einträge, müssen wir im Schnitt 500.000 davon prüfen. Wenn wir Pech haben, ist erst die zuletzt geprüfte Nummer die gesuchte. Ein Quantencomputer müsste nur ungefähr 1000 mal auf die Einträge zugreifen. Suchen wir ein bestimmtes Element in einer Datenbank der Größe  $N$ , kommen wir auf diese Weise mit etwa  $\sqrt{N}$  Aufrufen aus.<sup>2</sup> Grovers Algorithmus zur Datenbanksuche bietet eine quadratische Beschleunigung gegenüber dem klassischen Fall.

Zu Kapitel 5:  
Teleportation

Kapitel 6:  
Schnelle  
Datenbanksuche

<sup>2</sup>Genauer mit  $O(\sqrt{N})$ , siehe Abschnitt 3.1.

Die Grundidee von Grovers Algorithmus lässt sich auf verschiedene ähnliche Situationen übertragen. So sind Quantencomputer im gleichen Maß schneller, wenn unser Suchschema auf mehrere Einträge zutrifft oder wenn wir in diesem Fall die Anzahl der Einträge mit einer bestimmten Eigenschaft zählen wollen. Andererseits ist Grovers Algorithmus in der Größenordnung optimal, das heißt mehr als eine quadratische Beschleunigung ist nicht möglich. So wie der klassische Computer in einer Datenbank der Größe  $N$  im schlechtesten Fall  $N - 1$  Elemente betrachten muss, braucht ein Quantencomputer von der Größenordnung her mindestens  $\sqrt{N}$  Aufrufe.

#### Kapitel 8: Wie man RSA knackt

Noch bekannter als Grovers Algorithmus ist ein Verfahren, das 1993 von Peter Shor an den AT&T Labs entwickelt wurde. Shors Algorithmus zerlegt ganze Zahlen effizient in ihre Primfaktoren, also zum Beispiel 60 in folgendes Produkt von Primzahlen:  $2 \cdot 2 \cdot 3 \cdot 5$ . Das scheint nur auf den ersten Blick ein langweiliges, akademisches Problem zu sein. Denn klassische Computer brauchen dazu nach allgemeiner Annahme exponentielle Rechenzeit. Man ist von dieser Vermutung so überzeugt, dass die Sicherheit von RSA, einem der wichtigsten Public-Key-Verschlüsselungssysteme auf dieser Vermutung beruht. Denn solange Computer ganze Zahlen nur in exponentieller Zeit in ihre Primfaktoren zerlegen können, ist das Verfahren sehr sicher. Shors Quantenalgorithmus faktorisiert Zahlen effizient, somit ist der Quantencomputer in der Lage, eines der wichtigsten Verschlüsselungssysteme zu knacken. Mit dem gleichen Ansatz lässt sich der diskrete Logarithmus berechnen. Das ist ein weiteres Problem, das für den klassischen Computer nach heutigem Wissen nicht lösbar ist und darum zur Verschlüsselung genutzt wird. Am Ende des Kapitels wird auf eine Liste von weiteren Fragestellungen verwiesen, für die Quantenalgorithmen eine Beschleunigung versprechen.

#### Kapitel 7: Kryptographie

Leistungsfähige Quantencomputer sind also fähig, wichtige und weitverbreitete Kryptographieverfahren zu knacken. Gleichzeitig eignen sich Quantenbits aber auch zum Aufbau von neuartigen Verschlüsselungssystemen, die einen Grad an Sicherheit bieten, der mit anderen Mitteln nicht erreichbar ist. Die Grundsituation der Datensicherheit wird klassischerweise wie folgt dargestellt. Alice möchte Bob eine Nachricht schicken, ohne dass Eve (von englisch *eavesdropper*, Lauscher) diese einsehen kann. Verwenden Alice und Bob Quantenbits, können sie ihre Nachricht so präparieren, dass ein Lauschangriff sofort auffliegt. Denn wenn Eve die Nachricht erfahren möchte, muss sie die Quantenbits messen. Sind Alice und Bob geschickt vorgegangen, können sie einen solchen Lauschversuch jedoch entdecken, weil Eves Zugriff die Bits verändert hat.

Nun gut, mag man hier einwenden, man bemerkt es, aber Eve kennt die Nachricht trotzdem. Um zu erläutern, warum ein solches Verfahren trotzdem sehr nützlich ist, müssen wir etwas ausholen. Es gibt ein klassisches Verschlüsselungssystem, das mathematisch beweisbar absolut sicher ist: One-Time Pads, entwickelt 1917 von dem amerikanischen Mathematiker Gilbert Vernam. Es hat nur einen Nachteil: es verwendet einen sehr langen Schlüssel, der nur einmal verwendet werden kann und den sowohl Alice als auch Bob vor der Datenübertragung kennen müssen. Er muss also transportiert werden, und das ist der Schwachpunkt der One-Time Pads. Insbesondere dieses Problem



löst das Quantenverschlüsselungssystem. Wenn Eve versucht, etwas über den Schlüssel zu erfahren, wird das bemerkt. Alice und Bob verwenden schon übertragene Teile des Schlüssels nicht und unternehmen zu einem anderen Zeitpunkt einen neuen Versuch. Quantenkryptographiesysteme werden seit langem kommerziell angeboten. Die Quantenbits werden in der Regel mit Photonen realisiert, mit Lichtquanten also. Diese lassen sich zum Beispiel mit bereits bestehenden Glasfaserleitungen übertragen.

Das Studium von Prozessen auf Quantenebene ist für Chemie und Materialwissenschaften von großer Bedeutung. Klassische Rechner scheitern jedoch an der Aufgabe, Quantensysteme zu simulieren. Im Laufe des Kapitels 2 wird deutlich werden, dass der Aufwand dafür exponentiell wächst. Mit Quantenrechnern andererseits ist dies im Prinzip effizient möglich. Damit könnte die Simulation von Quantensystemen eine der ersten mächtigen und wichtigen Anwendungen von Quantencomputern sein, siehe etwa [126] oder [34]. Der Entwicklung neuer Materialien, zum Beispiel von Katalysatoren oder solcher zum Bau von Batterien, würden dadurch neue Möglichkeiten eröffnet. Interessant ist auch folgender Umkehrschluss: Wenn die Quantenwelt für unsere klassischen Rechner zu komplex ist, müsste dann nicht ein Quantencomputer weitaus komplexere Aufgaben lösen können als ein herkömmlicher?

Simulation von  
Quantenprozessen

Aber wie steht es um den Bau von Quantenrechnern? Die bisher geschilderten Verfahren wurden nicht lang nach Ihrer Veröffentlichung umgesetzt, zunächst unter Laborbedingungen und für kleine Eingaben. So wurde mit Shors Algorithmus in einem IBM-Labor bereits 2001 die Zahl 15 faktorisiert. Grovers Algorithmus lief bereits kurz nach der Veröffentlichung auf einem Drei-Bit-Rechner. Mittlerweile gibt es eine Vielzahl von Firmen, die funktionierende Quantencomputer anbieten. Diese sind für praktische Anwendungen den klassischen Rechnern noch nicht überlegen; die Erfolge der letzten Jahre können einen jedoch diesbezüglich hoffnungsvoll stimmen. Viel Aufmerksamkeit erhalten zur Zeit Quanten-Annealer. Dazu wollen wir erwähnen, dass Kapitel 2 den universellen Quantencomputer in seiner üblichen Form einführt. Man spricht vom *gatterbasierten* Quantencomputer, um solche universellen Rechner von Quanten-Annealern abzugrenzen. Quantum Annealing wiederum ist eine an Simulated Annealing angelehnte Heuristik zur approximativen Lösung von Optimierungsproblemen, siehe Abschnitt 11.2.

Kapitel 10:  
Praktische  
Umsetzung

Zur Zeit werden neben Ionenfallen (siehe Abschnitt 10.4) insbesondere auf Supraleitern basierende Quantenbits verwendet (siehe Abschnitt 10.6). Welche Technik sich mittelfristig durchsetzen wird, ist im Moment noch nicht klar. Das ist einer der Gründe, warum in diesem Buch Quantenhardware nur am Rande eine Rolle spielt. Sicherlich wird das Ergebnis aber kein Quanten-PC sein, der alle Aufgaben heutiger Rechner übernimmt. Quantencomputer sind klassischen Rechnern bei sehr speziellen Fragestellungen überlegen. So wird es mittelfristig auf eine Quanten-Cloud hinauslaufen, in der solche Spezialaufgaben an über das Netz zugreifbare zentrale Quantencomputer delegiert werden. Auch Sie selbst können Experimente auf realen Quantencomputern ausführen. Eine Auswahl von Links finden Sie auf der Homepage zum Buch, siehe Abschnitt 1.2.

Kapitel 3:  
Eigenschaften  
von Qubits

Deutlich leichter umsetzbar sind Verfahren, die Quantenbits zum Datenaustausch verwenden. Wie bereits erwähnt, gibt es seit längerem kommerzielle Anbieter und Anwender von Quantenkryptographiesystemen.

Es gibt auch Aufgaben, die für klassische Rechner selbstverständlich sind, für Quantencomputer hingegen unmöglich. So besagt das *No-Cloning-Theorem*, dass man ein Quantenbit nicht kopieren kann. Eine andere Besonderheit des Quantencomputers ist, dass alle seine Rechenschritte *umkehrbar* oder *reversibel* sind. Für klassische Berechnungen gibt es keine solche Einschränkung.

Dazu betrachten wir eine einfache logische Operation, die Konjunktion oder Verundung zweier Bits  $b_1$  und  $b_2$  mit dem Symbol  $\wedge$ :  $b_1 \wedge b_2$  ist gerade dann 1, wenn beide Bits 1 sind, wie in der folgenden Wahrheitstafel dargestellt.

$\wedge$	0	1
0	0	0
1	0	1

Erhalten wir als Ergebnis dieser Operation das Ergebnis 1, ist zwar klar, dass die beiden Eingabebits den Wert 1 hatten. Im anderen Fall allerdings gibt es drei Möglichkeiten für die Belegungen von  $b_1$  und  $b_2$ . Wir können den Rechenschritt nicht rückgängig machen.

Damit stellt sich die Frage, ob ein Quantencomputer all das berechnen kann, was ein klassischer Computer berechnen kann. Es wäre durchaus denkbar, dass die Forderung von umkehrbaren Berechnungen eine zu große Beschränkung ist und dass bestimmte sehr einfache Aufgaben plötzlich aufwändig werden, wenn jeder Schritt umkehrbar gemacht wird. Glücklicherweise ist es nicht so. Umkehrbare Berechnungen wurden schon untersucht, als man noch nicht an Quantenrechner gedacht hat. Dabei hat sich herausgestellt, dass gewöhnliche Rechnungen in reversible umgeformt werden können.

Interessant ist der Grund für die erwähnten Untersuchungen: Das Löschen eines Bits in einem Computer setzt Wärme frei. Umgekehrt verbrauchen umkehrbare Berechnungen – bei diesen darf nichts gelöscht werden – theoretisch keine Energie. Die Wärmeentwicklung ist angesichts der Miniaturisierung der Schaltkreise ein großes Problem. Weltweit werden allein für die Kühlung gewaltige Energiemengen verbraucht.

Kapitel 4:  
Komplexität

Eine Kernfrage unserer Thematik ist die folgende: Was kann der Quantencomputer berechnen, woran ein klassischer Computer scheitert? Zum einen stehen dem Quantencomputer Möglichkeiten offen, die jedem klassischen Rechner grundsätzlich verwehrt sind. Dazu zählen zum Beispiel Teleportation und die Erzeugung echter Zufallszahlen. Zum anderen gibt es Probleme, die sich im Prinzip berechnen lassen, jedoch nicht in der Praxis. Das Handwerkszeug zur Untersuchung dieser Frage finden wir in der Komplexitätstheorie, wo nach der *Effizienz* von Berechnungen gefragt wird. Wenn wir in dieser Einleitung sagen, ein Problem lasse sich nicht berechnen, meinen wir stets diesen praktischen Begriff.

Ein Schlüsselbegriff bei der Untersuchung, welche Probleme sich effizient berechnen lassen, ist *NP-Vollständigkeit*. Viele in der Praxis wichtige Optimierungsprobleme sind NP-vollständig, genauer: haben NP-vollständige

Entscheidungsvarianten. Ein Beispiel ist das *Problem des Handlungsreisenden*. Dieser Reisende will eine Reihe von Städten besuchen und am Ende wieder am Ausgangspunkt ankommen. Ihn interessiert nun die kürzeste solche Rundreise.<sup>3</sup> Einen Algorithmus für dieses Problem könnte man aber nicht nur als Routenplaner einsetzen, das gleiche Problem lässt sich auch wie folgt ausdrücken. Ein Roboterarm mit Bohrer soll eine Platine mit Tausenden Löchern versehen. Optimierte seinen Weg! Eine Eingabe ist also eine Liste von zu besuchenden Orten samt der Entfernungen zwischen je zwei dieser Orte. Ein Algorithmus für das Problem des Handlungsreisenden muss für jede dieser Eingaben das optimale Ergebnis liefern. Im Prinzip lässt sich dieses finden, indem man alle möglichen Rundreisen durchprobiert. Das ist allerdings praktisch undurchführbar. Denn die Zahl der möglichen Rundreisen wächst ungeheuer schnell. Sind es für fünf Städte nur 120, gibt es zwischen 10 Städten schon über drei Millionen Rundreisen, was für keinen PC ein Problem ist. Aber für 50 Städte, was ja nun nicht sehr viel ist, hat die Anzahl der Rundreisen bereits 64 Nullen. Es gibt eine magische Grenze: die vermutete Anzahl der Atome im Weltall ist eine Zahl mit 78 Nullen. Erhöhen wir die Anzahl der Städte auf 58, hat die Zahl der Rundreisen diese Grenze überschritten. Demgegenüber bezeichnen wir Probleme als effizient berechenbar, deren Berechnungsaufwand in der Eingabegröße deutlich langsamer wächst.

Alle NP-vollständigen Probleme haben zwei Dinge gemein:

1. Zurzeit lässt sich keines effizient berechnen.
2. Wenn man eine effiziente Lösung für *eines* dieser Probleme gefunden hat, lassen sich *alle* effizient lösen.

Die Komplexitätstheoretiker glauben recht einmütig, dass sich *keines* dieser Probleme effizient lösen lässt. Wer dies beweist oder widerlegt, wer das sogenannte *P-NP-Problem* löst, wird dafür eine Million Dollar erhalten, siehe [39]. Wenn es möglich wäre, mit dem Quantencomputer NP-vollständige Probleme effizient zu berechnen, wäre das ein ungeheurer Fortschritt. Mit einem Schlag wäre eine große Zahl sehr wichtiger Probleme mit dem Quantencomputer effizient lösbar, für die das klassisch nicht gilt. Übrigens ist das von Shor untersuchte Faktorisierungsproblem (siehe oben) vermutlich nicht NP-vollständig.

Es gibt Hinweise darauf, dass auch Quantencomputer NP-vollständige Probleme nicht effizient lösen können. Wir diskutieren sie in Abschnitt 6.7, wo unter anderem der Zusammenhang zwischen NP-vollständigen Problemen und Suchalgorithmen erläutert wird. Trotzdem bieten Quantenrechner faszinierende Möglichkeiten. Am Ende von Kapitel 8 verweisen wir auf eine Liste von Algorithmen, für die Quantencomputer eine Beschleunigung ermöglichen. Möglicherweise steht auch der eigentliche Durchbruch, die Entdeckung weiterer hocheffizienter Quantenalgorithmen, erst noch bevor.

---

<sup>3</sup>Korrekt wäre es, die *Entscheidungsvariante* zu betrachten: Gibt es eine Rundreise kürzer als zum Beispiel 500 km?

## 1.2 Über dieses Buch

Dieses Buch führt Leser ohne besondere Vorkenntnisse in ein sich rasant entwickelndes interdisziplinäres Gebiet ein. Die wichtigsten Ergebnisse aus den Bereichen *Quantum Computing* wie etwa Grovers Suchalgorithmus oder Shors Faktorisierungsalgorithmus werden detailliert dargestellt. Zudem werden Ergebnisse wie Teleportation und Quantenkryptographie präsentiert, die man im englischen Sprachraum dem Gebiet *Quantum Information* zuordnet.

### Vorkenntnisse

Benötigt werden nur Grundkenntnisse der Mathematik wie Vektorräume, komplexe Zahlen und Grundbegriffe der Wahrscheinlichkeitstheorie. Insbesondere sind keine besonderen Physikkenntnisse nötig. Zum Verständnis von Quantenalgorithmen genügen wenige quantenmechanische Begriffe, die in Abschnitt 2.10 zu drei Prinzipien zusammengefasst werden.

Da dieses Buch ohne Vorkenntnisse gelesen werden kann und durchgängig anschaulich und verständlich sein soll, ergeben sich zwei Beschränkungen: Dieses Buch ist keine Einführung in die Quantenphysik, und das sehr voraussetzungsreiche Thema *Hardware des Quantencomputers* wird nur oberflächlich behandelt. Sicher wird die Lektüre den Einstieg in diese Themen dennoch vereinfachen.

### Aufbau

Kapitel 2 bildet die Grundlage aller weiteren Kapitel: hier werden die Grundbegriffe eingeführt, und der Leser lernt die ersten Quantenalgorithmen kennen. Auch die Kapitel 3 und 4 haben grundlegenden Charakter; deren Lektüre kann jedoch zunächst zurückgestellt werden. Die den verschiedenen Anwendungen gewidmeten Kapitel 5 bis 8 sind weitgehend unabhängig voneinander. Der mathematische Anhang stellt in komprimierter Form wichtige Begriffe und Aussagen zur Verfügung.

### Übungen

Die Übungsaufgaben bilden einen wichtigen Teil des Buches, und bei einigen sind die Lösungen für die weitere Lektüre nötig. Darum findet der Leser am Ende des Buches Lösungen zu den meisten Aufgaben.

### Online-Service

Unter der Adresse

`informatik.th-brandenburg.de/~mhomeist/qc`

finden Sie Materialien zum Buch, Downloads und Links. Auch die Lösungen von Aufgaben, die sich aus Platzgründen nicht in dem Buch finden, sollen dort nach und nach zum Download bereit gestellt werden (es kann allerdings noch etwas dauern). Sollten Sie Anregungen zu diesem Buch haben oder einen Fehler finden, schicken Sie bitte eine E-Mail an `mhomeist@th-brandenburg.de`, damit Ihre Hinweise in der nächsten Auflage berücksichtigt werden können.

## 2 Vom Bit zum Quantenregister

*Der Anfang enthält also beides, Sein und Nichts; ist die Einheit von Sein und Nichts, – oder ist Nichtsein, das zugleich Sein, und Sein, das zugleich Nichtsein ist.*  
Georg Wilhelm Friedrich Hegel

Dieses Kapitel ist das Fundament für alle weiteren. Wir studieren Quantenbits und die Rechenschritte eines Quantencomputers und erfahren, was Messen und Verschränkung bedeuten. Dabei wenden wir diese Begriffe sofort an und lernen die ersten Algorithmen kennen, und zwar für Probleme, bei denen der Quantencomputer klassischen Rechnern überlegen ist. Als *klassisch* bezeichnen wir alles, was sich verhält, wie wir es aus unserer Alltagswelt gewohnt sind. Davon unterscheiden wir die nichtklassischen Phänomene der Quantenwelt.

Betrachten wir folgendes Spiel. Wir sollen herausbekommen, ob eine Münze echt oder eine plumpe Fälschung ist. Der echten Münze ist auf der Vorderseite eine Zahl eingeprägt, der Rückseite das Brandenburger Tor. Bei der falschen Münze sind beide Seiten gleich. Wie oft müssen wir uns die Münze ansehen, um eine Fälschung zu erkennen? Zweimal, da wir beide Seiten betrachten müssen. In der Quantenwelt genügt ein Blick, wenn wir eine Superposition beider Seiten erzeugt haben. Das ist der Kern des *Problems von Deutsch*. Es ist kein praktisch relevantes Problem, es verdeutlicht jedoch bereits die überraschenden Möglichkeiten von Quantenberechnungen.

Behandelte  
Algorithmen

Das *Problem von Deutsch-Jozsa* ähnelt dem nur nach Deutsch benannten, ist jedoch etwas komplexer. An ihm wird die Stärke von Quantenalgorithmen noch deutlicher. Unsere Aufgabe ist es, etwas über ein Bauteil herauszufinden, das eine mathematische Funktion berechnet; es könnte sich um einen Computerchip handeln. Wir haben keinerlei Informationen über dieses Bauteil: uns bleibt nichts anderes übrig, als die  $n$  Eingänge des Bauteils mit von uns gewählten Bitfolgen zu belegen und aus dem Ergebnis auf die Funktion zu schließen. Man bezeichnet so ein Bauteil als Black Box oder Orakel. Um das Problem von Deutsch-Jozsa zu lösen, muss ein klassischer

Rechner alle  $2^n$  Eingaben ausprobieren, während einem Quantenalgorithmus ein einziger Aufruf des Orakels genügt. Anhand dieses Algorithmus werden wir die Wirkungsweise der *Hadamard-Transformation* studieren, die für Quantenberechnungen eine große Rolle spielt.

Außerdem werden wir sehen, dass Quantencomputer echte *Zufallszahlen* erzeugen können. Der Ausdruck *echt* dient zur Abgrenzung von Pseudozufallszahlen, die uns klassische Rechner liefern. Wir werden kurz auf die Bedeutung der Zufallszahlen für die Informatik eingehen.

Beginnen werden wir aber mit der Frage, was ein Rechner klassischerweise tut, wenn er rechnet. Die Rechenschritte eines Quantenrechners sehen ganz anders aus, als jemand erwartet, der einen klassischen Prozessor oder eine höhere Programmiersprache vor Augen hat. Der Zugang wird erleichtert, wenn wir uns zunächst klassische Berechnungen genau ansehen und in ihre grundlegenden Bestandteile zerlegen. Wir betrachten Turingmaschinen und danach Schaltkreise. Quantenschaltkreise sind am besten geeignet, Quantenalgorithmen zu beschreiben und ihre *Effizienz* zu bewerten.

## 2.1 Was ist eine Berechnung?

Ein Quantencomputer ist ein ziemlich exotisches Gerät. Um ihn verstehen zu können, um einzusehen, dass das, was er tut, mit Fug und Recht rechnen genannt wird, betrachten wir zunächst, was die alltäglichen Computer tun. Diese werden für verschiedene Aufgaben genutzt: Wissenschaftler führen hochkomplexe Rechnungen, wie Wettervorhersagen oder die Simulation geologischer Vorgänge durch. Andere benutzen Computer zur Bearbeitung ihrer Urlaubsfotos oder als Spielkonsolen. In all diesen Anwendungen werden *Berechnungen* ausgeführt: Dem Rechner liegt eine Aufgabenbeschreibung vor, die *Eingabe*, und dazu liefert er eine Lösung als *Ausgabe*. Wir analysieren die dazwischenliegenden Schritte.

Schriftliche  
Addition

Dazu betrachten wir das folgende Beispiel. Ein Mensch löst mit Hilfe eines Stiftes und eines Blattes Papier eine Rechenaufgabe, wie in Abbildung 2.1 dargestellt. Er addiert die Zahlen 39831 und 21584, die untereinander auf dem Blatt stehen. Er geht so vor, wie wir es in der Schule gelernt haben. Er betrachtet zunächst die beiden letzten Stellen und addiert sie:  $4 + 1 = 5$ . Also ist die letzte Stelle der Summe 5, was er auf dem Blatt notiert.  $8 + 3 = 11$ : die nächste Stelle der Summe ist 1 und auch den Übertrag 1 notiert unser Rechner. So fährt er fort: seine Berechnung ist in einfache Schritte unterteilbar. In unserem Beispiel gehört dazu die Addition zweier einstelliger Zahlen. Aber auch das Lesen und Schreiben von Ziffern, der Test, ob eine Zahl größer 9 ist, und die Zerlegung einer solchen Zahl in zwei Stellen betrachten wir als einfache Schritte.

Eine Berechnung ist eine Folge einfacher Rechenschritte.



Abbildung 2.1: Rechner beim Addieren

Diese Definition ist zwar sehr intuitiv, hat aber zwei Probleme. Zunächst ist der Begriff *einfach* nicht klar: in unserem Beispiel haben wir ihn willkürlich festgelegt. Zudem ist es wenig erhellend, den Begriff *Berechnung* unter Rückgriff auf den Begriff *Rechenschritt* zu definieren. Darum folgt nun eine abstraktere Definition; mit ihr lässt sich leicht beschreiben, wie Quantencomputer rechnen.

Betrachten wir den addierenden Menschen mit seinen Hilfsmitteln als eine Einheit, als einen Rechner, so durchläuft er eine Reihe von *Zuständen*. Er beginnt in einem *Anfangszustand*, der durch die Eingabe auf dem Blatt bestimmt ist. Er führt nun Rechenschritte aus: er betrachtet die letzte Stelle der ersten Zahl und ist danach in einen anderen Zustand übergegangen, zuvor kannte er diese Ziffer noch nicht. Er liest die letzte Stelle der anderen Zahl, addiert beide und schreibt das Ergebnis auf das Blatt. Das sind drei weitere Zustandsänderungen. In dieser Weise führt jeder *Rechenschritt* zu einem *neuen Zustand*, den wir als Zwischenergebnis auffassen können. Schließlich bricht die Rechnung in einem *Endzustand* ab, und das Ergebnis steht auf dem Papier. Damit ist auch der Begriff *Rechenschritt* geklärt, den Begriff *einfach* werden wir anhand der Turingmaschine erläutern. Noch eine Beobachtung ist wichtig: die Entscheidung, welcher Schritt als nächstes auszuführen ist, hängt allein vom aktuellen Zustand ab. Und nicht etwa von früheren Zuständen oder irgendeinem intuitiven Wissen. Wir fassen zusammen:

Definition  
Berechnung

Eine Berechnung ist eine Folge von Zuständen des Rechners. Jeder Rechenschritt ist ein Übergang zwischen Zuständen und hängt allein vom aktuellen Zustand ab.



Abbildung 2.2: Ein Quantenrechner: er befindet sich in mehreren Zuständen gleichzeitig

Ein Quantenrechner kann zu jeder Zeit in mehreren klassischen Zuständen gleichzeitig sein, wie in Abbildung 2.2 dargestellt ist. Am Beginn der Einleitung wurde darauf hingewiesen, dass ein Quantenrechner kein Parallelrechner ist. Der Parallelrechner aus Abbildung 1.1 hat viele Arme, jeder muss einzeln angesteuert werden, verbraucht Energie, und alle Arme müssen koordiniert werden. Ein Quantenrechner besitzt nur einen Arm. Er teilt sich dabei auf eine eigentümliche Weise: es ist, als würde er in verschiedenen Welten gleichzeitig rechnen. In jeder Welt bearbeitet er eine andere Eingabe, die Ergebnisse sind uns aber nicht direkt zugänglich. Insofern führt der oft gebrauchte Begriff *Quantenparallelismus* in die Irre, obwohl ein Quantenrechner in gewisser Hinsicht tatsächlich auf mehreren Eingaben gleichzeitig rechnet.

Wir werden sagen, der Zustand eines Quantenrechners bestehe aus einer *Superposition* von Zuständen eines klassischen Rechners. Damit wird auch der eigentümliche Charakter von Quantenberechnungen deutlich. Denn ein Rechenschritt eines solchen Computers ist damit der Übergang zwischen zwei Superpositionen.

#### Definition Algorithmus

Der folgende Begriff ist mit unseren Überlegungen eng verbunden: Berechnungen werden durch *Algorithmen* beschrieben. Ein Algorithmus ist eine Bearbeitungsvorschrift, die so präzise formuliert ist, dass sie von einer Maschine ausgeführt werden kann. Eine solche Maschine werden wir nun kennen lernen. Dadurch wird unsere intuitive Definition der Berechnung konkretisiert. Insbesondere können wir festlegen, was ein elementarer, einfacher Rechenschritt ist.



### 2.1.1 Die Turingmaschine

Die Turingmaschine, man kann diese als einen idealisierten Computer auffassen, stammt von dem britischen Mathematiker Alan Turing (1912-1954). Ihm gelang es damit, Berechnungen so formal zu beschreiben, dass sie wie mathematische Objekte behandelt werden können. Als Vorbild diente ihm der uns schon bekannte Herr am Tisch, der mit Bleistift und Papier eine Matheaufgabe löst.

Eine Turingmaschine besitzt einen *Schreib-Lese-Kopf*, den sie über ein langes *Papierband* bewegen kann. Das Band ist in Kästchen eingeteilt. Ein Kästchen enthält eine 0, eine 1 oder das Leerzeichen. Der Kopf kann nicht nur über das Band bewegt werden, sondern er kann das Zeichen an der aktuellen Stelle auch lesen oder überschreiben. Was die Maschine tut, regelt die *Steuerung*. Diese befindet sich jederzeit in einem internen *Zustand*; diese Zustände bilden so etwas wie das Gedächtnis der Maschine.

Die Steuerung legt nun fest, wie gerechnet wird, indem *feste Regeln zum Übergang zwischen Zuständen* ausgeführt werden. Anfangs steht nur die Eingabe auf dem Band. Wie mit ihr gerechnet wird, ist in der Steuerung realisiert: Abhängig von

Rechenschritte

- dem Kästcheninhalt an der Stelle des Kopfes und
- dem aktuellen Zustand,

wird

- das Zeichen unter dem Kopf überschrieben,
- der Zustand aktualisiert und
- der Kopf einen Schritt nach links oder nach rechts bewegt.

Eine *Konfiguration* der Turingmaschine besteht nun aus folgenden Teilen: der Position des Kopfes, dem Inhalt des Bandes und dem Zustand der Steuerung. Mit jedem Rechenschritt ändert sich diese Konfiguration, die man als Beschreibung des Gesamtsystems auffassen kann. Am Ende der Berechnung, durch einen speziellen, vorher festgelegten Zustand signalisiert, befindet sich das Ergebnis der Berechnung auf dem Band. Abbildung 2.3 verdeutlicht dieses Konzept.

Nun lässt sich eine Turingmaschine konstruieren, die unseren Additionsalgorithmus ausführt; natürlich angepasst an Binärzahlen. Außerdem kann man eine sogenannte *universelle Turingmaschine* angeben, die für alle möglichen Berechnungsaufgaben genutzt werden kann. Neben der Eingabe des zu berechnenden Problems, übergibt man ihr eine Beschreibung des auszuführenden Algorithmus. Das entspricht unserem gewohnten Bild vom Computer. Wir haben nicht für jede Aufgabe ein spezielles Gerät, sondern können verschiedene Programme ausführen, die wir vorher in den Speicher geladen haben.

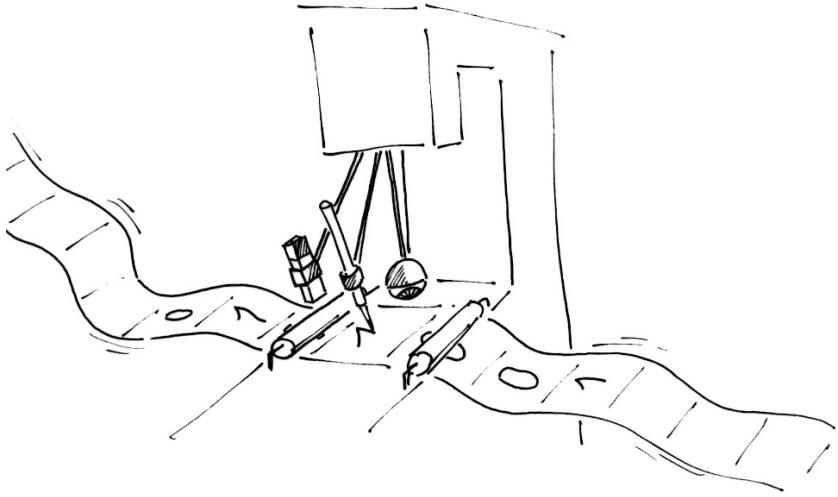


Abbildung 2.3: Turingmaschine bei der Arbeit

**Aufgabe 2.1:** Überlegen Sie sich die Arbeitsweise einer Turingmaschine für das folgende Problem. Zu Beginn befindet sich eine Binärzahl auf dem Band, und der Kopf befindet sich über dem letzten Zeichen. Die Turingmaschine erhöht diese Zahl um den Wert 1, sie berechnet also den Nachfolger einer Zahl.

Alan Turing entwickelte die nach ihm benannte Maschine, um folgende Frage zu untersuchen: Gibt es Probleme, die sich nicht berechnen lassen? Gleichzeitig legte er damit die Grundlage für die ersten universellen Rechner: Rechenmaschinen, die jedes berechenbare Problem lösen können. Ein Vorzug der Turingmaschine ist, dass der Begriff *Berechnung* mathematisch untersucht werden kann. Algorithmen lassen sich mit ihr jedoch nur sehr sperrig beschreiben. Dazu eignet sich ein anderes Modell besser, das zusätzlich dem technischen Aufbau tatsächlicher Computer nahe kommt. Diesem ist der nächste Unterabschnitt gewidmet.

### 2.1.2 Schaltkreise

Ein sehr anschauliches Modell des Quantencomputers liefern Quantenschaltkreise. Darum bilden sie das grundlegende Berechnungsmodell dieses Buches. Dieser Abschnitt führt klassische Schaltkreise am Beispiel der Addition ein.

Schaltkreise sind aus einfachen Bauteilen aufgebaut, den *Gattern*. Wir betrachten Gatter, die einfache Funktionen berechnen. Zum Beispiel hat ein AND-Gatter zwei Eingänge und einen Ausgang und berechnet die logische Operation *Und*. Werden die Eingänge mit zwei Bits  $a$  und  $b$  belegt, liefert der Ausgang genau dann den Wert 1, wenn  $a$  und  $b$  den Wert 1 haben. Wir

verwenden das Symbol  $\wedge$ . Abbildung 2.4 zeigt eine schematische Darstellung solch eines Gatters und eine Tabelle, die sein Verhalten beschreibt.

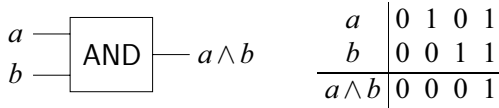


Abbildung 2.4: Gatter für die logische Operation AND

Auf nur einem Bit operiert die *Negation*, das zugehörige Bauteil ist das NOT-Gatter aus Abbildung 2.5. Es liefert gerade dann den Wert 1, wenn das Eingangsbit *nicht* 1 ist. Weitere wichtige logische Operationen sind *Oder* und *Exklusives Oder*. Das logische Oder wird von einem OR-Gatter berechnet. Das Ausgangsbit ist gerade dann 1, wenn mindestens eines der Eingangsbits 1 ist. Bei einem XOR-Gatter für das exklusive Oder zweier Bits ist das Ausgangsbit genau dann 1, wenn *genau* eines der Eingangsbits 1 ist.

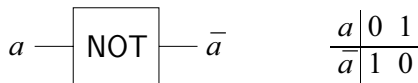


Abbildung 2.5: Gatter für die Negation

Ein *Schaltkreis* ist aus Gattern aufgebaut, ähnlich wie sich komplexe Software aus einfachen Modulen konstruieren lässt. Dabei werden die Eingänge einiger Gatter mit den Ausgängen anderer Gatter durch *Leitungen* oder *Drähte* verbunden, auf diese Weise werden die Bits von einem Bauteil zum nächsten geleitet. Als Beispiel konstruieren wir einen Schaltkreis, der zwei jeweils drei Bit lange Zahlen addiert. Er wird sechs Eingangsbits und vier Ausgangsbits haben, wie schematisch in Abbildung 2.6 dargestellt ist. Die Ausgangsbits  $c_0, \dots, c_3$  stellen binär die Summe der durch die Eingangsbits  $a_0, \dots, a_2$  und  $b_0, \dots, b_2$  kodierten Zahlen dar:

Additionsschalt-  
kreis

$$c_3 \cdot 2^3 + \dots + c_0 \cdot 2^0 = (a_2 \cdot 2^2 + a_1 \cdot 2 + a_0) + (b_2 \cdot 2^2 + b_1 \cdot 2 + b_0).$$

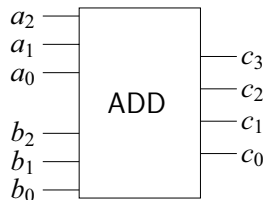


Abbildung 2.6: Schematische Darstellung eines 3-Bit-Addierers

An dieser Stelle ist es nützlich zu betrachten, wie auf der Menge  $\{0,1\}$  gerechnet wird.

Rechnen in  $\{0,1\}$

In der Menge  $\{0,1\}$  bezeichnen wir die Addition mit  $\oplus$  und die Multiplikation mit  $\cdot$ . Diese Operationen sind wie folgt definiert:

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

$\cdot$	0	1
0	0	0
1	0	1

Die Multiplikation entspricht der Operation AND oder  $\wedge$ , die Addition dem XOR.  $+$  und  $\sum$  bezeichnen stets die gewohnte Addition von ganzen oder komplexen Zahlen, während  $\oplus$  die eben definierte Addition von Bits ist.

Der erste Schritt zur Realisierung des Additionsschaltkreises ist in Abbildung 2.7 dargestellt. Aus einem AND-Gatter und einem XOR-Gatter wird ein sogenannter Halbaddierer aufgebaut. Der dicke schwarze Punkt markiert eine Verzweigung, Kreuzungen ohne eine solche Markierung haben keine Bedeutung. Es werden zwei Bits  $a$  und  $b$  addiert, das Ergebnis besteht aus dem Summenbit  $s$  und einem Übertragsbit  $c$  für englisch *carry*. Für die dargestellte Zahl gilt  $a + b = 2c + s$ , wie auch aus der Tabelle zu ersehen ist.

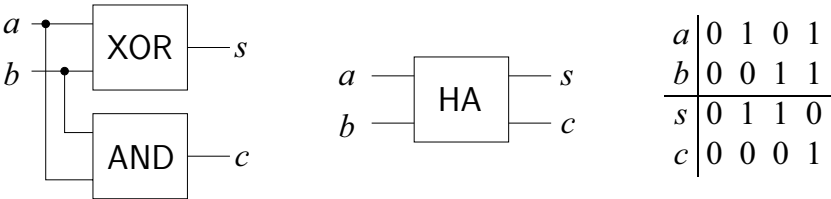


Abbildung 2.7: Der Halbaddierer: Schaltkreis, Symbol und Funktionalität

Der Halbaddierer ist für die Addition der beiden letzten Stellen bestens geeignet. Bei der Addition höherer Stellen müssen wir die Übertragsbits berücksichtigen, die von niedrigeren Stellen geliefert werden. Diese Aufgabe übernimmt ein Volladdierer, wie in Abbildung 2.8 dargestellt. Er setzt um, dass für  $s_i$ , die  $i$ -te Stelle der Summe,

$$s_i = a_i \oplus b_i \oplus \text{Übertrag aus der } (i - 1)\text{-ten Stelle}$$

gilt. Häufig wird OR an der Stelle von XOR verwendet. Wir verwenden letztgenannte Operation, um mit den in der Box oben eingeführten Operationen auszukommen.

Der letzte Schritt zur Konstruktion eines Additionsschaltkreises bleibt dem Leser überlassen.

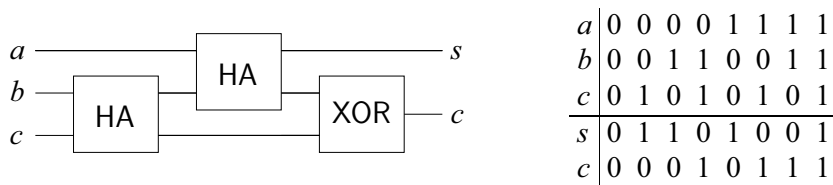


Abbildung 2.8: Der Volladdierer: Schaltkreis und Funktionalität

**Aufgabe 2.2:** Realisieren Sie unseren Additionsschaltkreis gemäß Abbildung 2.6 aus einem Halbaddierer und zwei Volladdierern.

Schaltkreise sind ein so anschauliches Berechnungsmodell, dass man eigentlich nicht näher beschreiben muss, wie sie konkret rechnen. Man stellt sich leicht vor, wie die Bits hindurch fließen. Wir wollen noch darauf eingehen, wie Schaltkreise mit unserer Definition der Berechnung als Folge von Zuständen des Rechners übereinstimmen. Dazu ordnen wir den Leitungen nach und nach Werte zu. Die Berechnung beginnt damit, dass die Eingangsbits und die zugehörigen Leitungen gemäß der Eingabe belegt werden; das ist der Startzustand. Nun sind die Eingänge einiger Gatter belegt, und wir belegen die Leitungen an den Ausgängen entsprechend; das ist der Folgezustand. Und so schreitet die Rechnung voran.

Schaltkreise und  
Berechnung

### 2.1.3 Der Sprung in die Quantenwelt: Schrödingers Katze

Zum Einstieg in das Rechnen mit Quanten betrachten wir eine berühmte Veranschaulichung eines grundlegenden Quantenphänomens. Wir führen die Begriffe *Superposition* und *Messen* an Hand von *Schrödingers Katze* ein. Dieses Tier erleichtert die Annäherung an das Quantenbit und hat folgenden Vorteil: man vergisst es nicht mehr.

Vor uns steht eine verschlossene Kiste; in der Kiste befindet sich eine Katze. Wir wissen, dass gerade ein grausiger „Mechanismus“ ausgelöst wurde. Mit der Wahrscheinlichkeit  $1/2$  ist giftiges Gas ausgeströmt, das die Katze sofort getötet hat. Mit der gleichen Wahrscheinlichkeit ist nichts passiert und eine quicklebendige Katze sitzt in der Kiste. Welches Ereignis eingetreten ist, hängt vom Zerfall eines Atoms ab.

Unserem Alltagsverstand zufolge ist die Katze in der Kiste entweder tot oder lebendig, auch wenn wir es nicht wissen. In der Quantenwelt können die Dinge jedoch grundverschieden sein. Dort ist es metaphorisch gesprochen möglich, dass eine Katze lebendig und tot zugleich ist. Sie ist bezüglich der Frage, ob sie tot oder lebendig ist, unbestimmt. Diese Situation ist in Abbildung 2.9 dargestellt. Öffnen wir die Kiste, ist die Katze entweder tot oder lebendig wie in Abbildung 2.10, solange diese verschlossen ist, bleibt der Zustand uneindeutig.

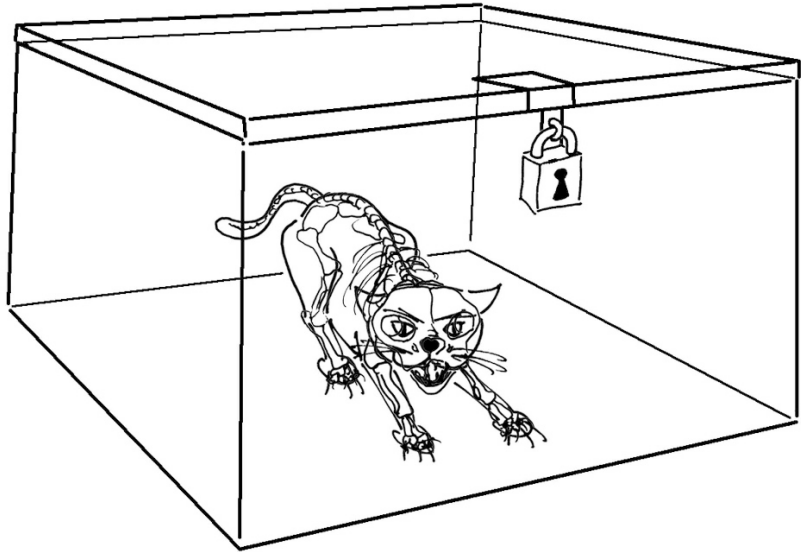


Abbildung 2.9: Katze, lebendig und tot zugleich

In der Quantenwelt ist ein solch zombiehafter Zustand ganz normal. Kleine Teilchen können sich im oben beschriebenen Sinne an zwei Orten gleichzeitig befinden oder andere sich eigentlich ausschließende Eigenschaften vereinen: der genaue Zustand ist unbestimmt, das Teilchen befindet sich in *Superposition* zweier Zustände. Voraussetzung dafür ist, dass das Teilchen von der Außenwelt abgeschottet ist, was in dem Katzenbeispiel die Kiste leistet. Jede Einwirkung von außen, bei uns das Öffnen der Kiste zur Diagnose oder *Messung* des Gesundheitszustands, zerstört die Superposition: Der Zombie kehrt ins Leben zurück oder stirbt ganz und gar. Das Teilchen befindet sich nach der Messung an einem konkreten, wohlbestimmten Ort.

Das Gedankenexperiment mit der Katze stammt von Erwin Schrödinger (1887-1961), einem der Väter der Quantenmechanik. Er wollte verdeutlichen, dass ein solcher Zustand für ein so großes Objekt wie eine Katze völlig absurd ist. Kleine Teilchen verhalten sich aber gerade so. Für uns ist Schrödingers Veranschaulichung ein Bild für ein Phänomen der Quantenwelt, das für unsere Berechnungen entscheidend ist, wenn es auch unserer Intuition widerspricht. Wir betrachten ein Teilchen mit zwei sich ausschließenden Eigenschaften: es kann sich nach unserem Alltagsverstand um eine gedachte Achse *entweder* im Uhrzeigersinn *oder* gegen den Uhrzeigersinn drehen. In der Quantenwelt kann es beide Zustände *zugleich* annehmen: dann dreht es sich mit einem Anteil  $\alpha$  in die eine Richtung und mit einem Anteil  $\beta$  in die entgegengesetzte; das ist die erwähnte Superposition zweier Zustände. Wir können aber niemals erkennen, welchen Wert diese Anteile  $\alpha$  und  $\beta$  genau haben. Wollen wir die Drehrichtung bestimmen, müssen wir sie *messen*. Dabei wird dieser

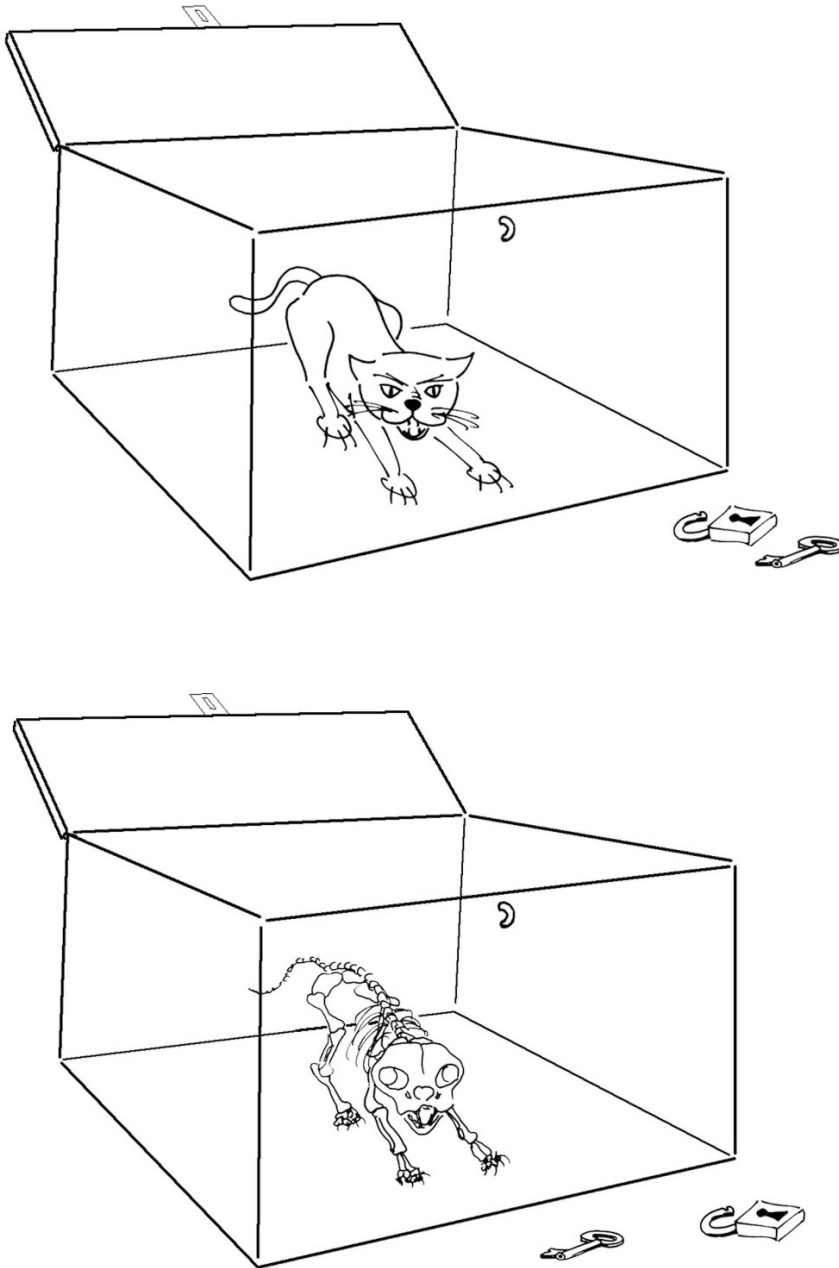


Abbildung 2.10: Bei offener Kiste ist die Katze entweder lebendig oder tot

merkwürdige Quantenzustand, die Superposition, *zerstört*. Mit einer gewissen Wahrscheinlichkeit, die von  $\alpha$  und  $\beta$  abhängt, nimmt das Teilchen eine der beiden Richtungen an; diese ist unser Messergebnis.

Messen stellt eine Wechselwirkung mit der Außenwelt dar: Der Zustand von Schrödingers Katze ist nur so lange unbestimmt, wie sie in der Kiste von der Außenwelt abgeschottet ist. Wollen wir Schrödingers Katze anschauen, muss Licht in die Kiste dringen. Die *splendid isolation* ist beendet, die Wechselwirkung mit den Lichtteilchen kann die Superposition zerstören. Einer der Zustände *tot* oder *lebendig* wird realisiert. Wir selbst gehören stets zur Außenwelt eines Quantenteilchens. Versuchen wir etwas darüber herauszufinden, beeinflussen wir es. In diesem Sinne ist die Kiste ein entscheidender Teil des Gedankenexperiments.

Auf diesen Überlegungen baut die Hauptdefinition des nächsten Abschnittes auf. Ein Quantenbit kann gleichzeitig den Wert 0 und den Wert 1 annehmen. Wollen wir etwas über dieses Bit erfahren, bleibt uns nichts anderes übrig, als es zu *messen*. Die Eigenschaft, gleichzeitig in zwei Zuständen zu sein, geht verloren. Wir erhalten den Wert 0 mit einer Wahrscheinlichkeit  $p$  und den Wert 1 mit der Wahrscheinlichkeit  $1 - p$ .

## 2.2 Das Quantenbit

Ein klassisches Bit ist entweder 0 oder 1. Ein Quantenbit kann beides zugleich sein. Wie das zu verstehen ist, wird in diesem Abschnitt erläutert.

Dirac-Notation

In der Quantenmechanik ist es üblich, Zustände in Klammern der Form  $|\cdot\rangle$  zu setzen. Die Werte eines klassischen Bits werden damit zu  $|0\rangle$  und  $|1\rangle$ . Es ist dies die *ket-Notation* oder auch *Dirac-Notation*, die auf den britischen Physiker Paul Dirac (1902-1984) zurückgeht, der 1933 zusammen mit Schrödinger den Nobelpreis erhielt.

Definition  
Quantenbit

Ein *Quantenbit*, kurz *Qubit*, nimmt Zustände der folgenden Form an:

$$\alpha \cdot |0\rangle + \beta \cdot |1\rangle. \quad (2.1)$$

$\alpha$  und  $\beta$  heißen *Amplituden* und sind komplexe Zahlen mit

$$|\alpha|^2 + |\beta|^2 = 1.$$

Superposition,  
Überlagerung

Ein Quantenbit kann sich also in zwei klassischen Zuständen gleichzeitig befinden; die komplexen Zahlen drücken deren jeweiligen Anteil aus. Man spricht von *Superposition* oder *Überlagerung* der klassischen Zustände  $|0\rangle$  und  $|1\rangle$ . Die Bedingung  $|\alpha|^2 + |\beta|^2 = 1$  schränkt die möglichen Werte der Amplituden ein, zulässige Beispiele sind  $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$  und  $0 \cdot |0\rangle + 1 \cdot |1\rangle = |1\rangle$ . Trotz dieser Einschränkung sind unendlich viele verschiedene Zustände möglich.



Ein klassisches Bit kann man lesen und seinen genauen Zustand feststellen. Das ist bei einem Quantenbit in dieser Form nicht möglich: Möchte man auf ein Quantenbit lesend zugreifen, muss man es *messen*. Das Messergebnis hängt von den Amplituden  $\alpha$  und  $\beta$  ab.

Messen wir ein Quantenbit im Zustand  $\alpha \cdot |0\rangle + \beta \cdot |1\rangle$ , wird die Superposition zerstört. Anschließend ist es mit Wahrscheinlichkeit  $|\alpha|^2$  im Zustand  $|0\rangle$  und mit Wahrscheinlichkeit  $|\beta|^2$  im Zustand  $|1\rangle$ . Diesen Zustand nach dem Messen können wir beobachten.

Messen eines Quantenbits

**Beispiel 2.1:** Die Superposition  $\frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle$  ist ein zulässiger Zustand eines Quantenbits. Denn es gilt

$$\left(\frac{1}{\sqrt{3}}\right)^2 + \left(\sqrt{\frac{2}{3}}\right)^2 = \frac{1}{3} + \frac{2}{3} = 1.$$

Messen wir ein Bit in diesem Zustand, ist das Ergebnis mit Wahrscheinlichkeit  $1/3$  der Zustand  $|0\rangle$  und mit Wahrscheinlichkeit  $2/3$   $|1\rangle$ .

Messen wir ein Bit im Zustand  $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ , sind die Ergebnisse  $|0\rangle$  und  $|1\rangle$  gleichwahrscheinlich.  $\diamond$

Dazu eine Erläuterung: Werfen wir einen sechsseitigen Würfel, ist das Ergebnis eine der Zahlen  $1, 2, \dots, 6$ . Sehen wir von Unregelmäßigkeiten des Würfels ab, erhalten wir jede Zahl mit Wahrscheinlichkeit  $1/6$ ; werfen wir den Würfel sehr oft, erwarten wir, dass wir zum Beispiel die 1 in einem Sechstel der Fälle beobachten. Aber was ist Wahrscheinlichkeit? Man kann wie folgt argumentieren: hätten wir exakte Informationen über den Würfelwurf und die Beschaffenheit der Oberfläche, auf der er auftritt, könnten wir den genauen Bewegungsablauf berechnen, wie wir die Bewegung der Planeten berechnen können. Uns fehlen nur diese exakten Informationen!

Das ist in der Quantenwelt nicht so. Messen wir ein Bit im Zustand  $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ , ist der Ausgang tatsächlich unbestimmt und keine Folge unserer Unwissenheit oder der Grobheit der Messinstrumente. Gibt es auch Quantenzustände, bei denen das Ergebnis feststeht? Messen wir  $0 \cdot |0\rangle + 1 \cdot |1\rangle = |1\rangle$ , so erhalten wir mit Sicherheit den Wert  $|1\rangle$  als Ergebnis. Dieser Zustand verhält sich wie ein klassisches Bit, das 1 gesetzt ist. Wir bezeichnen  $|0\rangle$  und  $|1\rangle$  als *nicht-überlagerte Zustände*, da sie den Zuständen eines klassischen Bits entsprechen.

**Aufgabe 2.3:** 100 Quantenbits befinden sich im Zustand  $\frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle$ . Sie messen alle. Welches Ergebnis erwarten Sie?

Phase

Das Ergebnis der Messung hängt nur von dem Betrag der Amplitude ab. Messungen der Zustände  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  und  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  ergeben jeweils  $|0\rangle$  und  $|1\rangle$  mit Wahrscheinlichkeit  $1/2$ . Zwei reelle Zahlen haben denselben Betrag, wenn sie sich nur im Vorzeichen unterscheiden. Komplexe Zahlen  $z$  haben denselben Betrag, wenn sie sich nur bezüglich der Phase unterscheiden, siehe Seite 295.

Zustände als  
Vektoren

Was nun folgt ist wesentlich, um die Arbeitsweise des Quantencomputers zu verstehen. Den Zustand eines Quantenbits werden wir ab jetzt als Vektor in einem zweidimensionalen Vektorraum über den komplexen Zahlen betrachten; Vektorräume sind immer dann nützlich, wenn eine Beschreibung von mehreren unabhängigen Komponenten abhängt. Auf die folgenden Überlegungen geht auch Abschnitt A.2 im Anhang über Vektorräume ein.

In unserem Fall sind die Komponenten die Amplituden, und die Superposition  $\alpha|0\rangle + \beta|1\rangle$  wird zu dem Vektor aus Abbildung 2.11. Der zugehörige Zustandsvektor ist

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix}.$$

Diesen Vektor können wir als Linearkombination der zweidimensionalen Standardbasisvektoren angeben:

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \alpha|0\rangle + \beta|1\rangle.$$

Die Basis unseres Vektorraumes besteht damit aus den beiden nicht überlagerten Zuständen  $\{|0\rangle, |1\rangle\}$ . Die Superposition  $\alpha|0\rangle + \beta|1\rangle$  wird zu einer *Linearkombination* dieser Basiselemente. Natürlich sind nicht alle Vektoren unseres zweidimensionalen Vektorraums mögliche Zustände eines Quantenbits. Wegen der Bedingung  $|\alpha|^2 + |\beta|^2 = 1$  sind es gerade diejenigen der Länge 1.

Aus Abbildung 2.11 kann man nicht ersehen, dass  $\alpha$  und  $\beta$  komplexe Zahlen sind. Wir stellen uns den Vektorraum einfach so vor, als wäre er über den reellen Zahlen definiert, und rechnen gemäß den Regeln für komplexe Zahlen.

Realisierung

Wie kann man ein Quantenbit bauen, um damit konkret zu rechnen? Dazu muss ein Teilchen von seiner Umwelt strikt abgeschottet werden. Wir erinnern uns an die Kiste, in der Schrödingers Katze steckte. Ohne diese Abschottung würde die Umwelt mit dem Teilchen wechselwirken, und die Superposition würde zerstört werden; ähnlich wie durch eine Messung. Das Stichwort hierzu lautet *Dekohärenz*, siehe Abschnitt 9.1.

Nun nimmt man eine messbare und beeinflussbare Eigenschaft des abgeschotteten Teilchens her, zum Beispiel die Drehrichtung um eine festgelegte Achse oder zwei Energieniveaus. Wir wählen zwei Zustände dieser Eigenschaft, die sich in der klassischen Welt ausschließen: diese bezeichnen wir mit  $|0\rangle$  und  $|1\rangle$ . Fertig. Dieser Zugang ist aus praktischer Sicht unverantwortlich hemdsärmelig. Wir sollten uns jedoch auf genau diesen Standpunkt stellen.

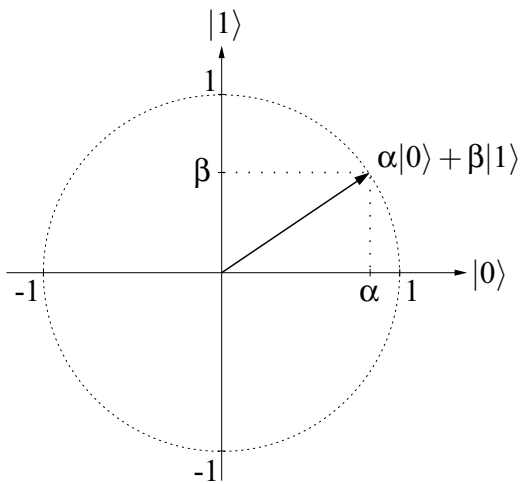


Abbildung 2.11: Die Superposition als Vektor

Quantenbits sind realisierbar; das haben Experimentalphysiker bewiesen. Wir stellen uns im weiteren die Frage, was sich damit anfangen lässt.

Unsere aktuelle Frage lautet: Wie können wir mit Quantenbits rechnen? Oder anders ausgedrückt, was für Rechenschritte können wir ausführen, wie gehen zwei Zustände eines Quantenbits ineinander über?

## 2.3 Rechenschritte auf einem Quantenbit

Wir wissen bereits, was ein Quantenbit ist, aber noch nicht, was man damit machen kann. In Abschnitt 2.1 haben wir festgestellt, dass eine Berechnung eine Folge von Zuständen des Rechners ist. Ein Rechenschritt überführt dabei den aktuellen Zustand in den Folgezustand. Bei klassischen Rechnern gibt es keine weiteren Einschränkungen. Hat das Bit etwa den Wert 0, kann es im nächsten Rechenschritt 1 gesetzt werden oder den Wert 0 behalten. Ein Quantenbit hingegen kann unendlich viele verschiedene Zustände annehmen. Der Übergang zwischen den Zuständen unterliegt dabei besonderen Bedingungen.

Die möglichen Rechenschritte auf einem Quantenbit werden durch quadratische Matrizen beschrieben, genauer durch  $2 \times 2$ -Matrizen, die *unitär* sind. Unitäre Matrizen beschreiben mathematisch, wie sich ein abgeschottetes Quantensystem ändert. Matrizen beschreiben Abbildungen, siehe dazu den Anhang. Aus einer unitären Matrix lässt sich besonders einfach die inverse Matrix beziehungsweise die Umkehrabbildung ableiten.

Dazu benötigen wir folgenden Begriff: Hat eine Matrix  $A = (a_{ij})$  komplexe Einträge, kommen wir zu  $A^*$ , der *komplex konjugierten* von  $A$ , indem wir  $a_{ij}$  durch  $a_{ij}^*$  ersetzen (zu komplexen Zahlen siehe Abschnitt A.1 und zu Matrizen und der verwendeten Notation Abschnitt A.3 im Anhang). Mit  $A^\dagger$  bezeichnen wir die komplex konjugierte und transponierte Matrix  $(A^*)^T$ .  $A^\dagger$  heißt zu  $A$  *adjungierte* Matrix und entsteht aus  $A$ , indem  $a_{ij}$  durch  $a_{ji}^*$  ersetzt wird. Leider gibt es keine einheitliche Notation: in der deutschsprachigen Literatur wird die komplex konjugierte Matrix meistens mit  $\bar{A}$  bezeichnet und die adjungierte mit  $A^*$ . Unsere Schreibweise hat den Vorteil, den Einstieg in die Originalliteratur zu erleichtern.

Definition  
unitäre Matrix

Sei  $A$  eine  $n \times n$ -Matrix, deren Einträge komplexe Zahlen sind.  $A$  heißt *unitär*, falls  $A^\dagger$  zu  $A$  invers ist:

$$A^\dagger = A^{-1}.$$

Eine unitäre Matrix beschreibt eine unitäre Transformation.

Eine Matrix  $A$  mit reellen Einträgen ist genau dann unitär, wenn wir durch Transposition zur Inversen gelangen:  $A^{-1} = A^T$ . Die beiden folgenden Matrizen sind unitär:

**Beispiel 2.2:** Die zweidimensionale Einheitsmatrix

$$I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

ist unitär. Denn es gilt  $I_2^\dagger = I_2 = I_2^{-1}$ .

Hadamard-  
Matrix

Die Matrix

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

ist unitär. Sie heißt Hadamard-Matrix und wird eine wichtige Rolle für uns spielen.

Benannt ist die Matrix  $H$  nach dem französischen Mathematiker Jacques Hadamard (1865-1963).

**Aufgabe 2.4:** Beweisen Sie, dass die Matrix  $H$  unitär ist.

Entwicklung  
eines Qubits

Nun beschreiben wir, wie eine solche Matrix eine Zustandsveränderung eines Quantenbits bewirkt. Auf einem Quantenbit im Zustand  $\alpha|0\rangle + \beta|1\rangle$  soll ein

Rechenschritt ausgeführt werden, der durch eine zweidimensionale (unitäre) Matrix

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

beschrieben wird. Wir erhalten den Folgezustand  $\alpha'|0\rangle + \beta'|1\rangle$ , indem wir den Zustandsvektor mit der Matrix multiplizieren.

$$\begin{pmatrix} \alpha' \\ \beta' \end{pmatrix} = A \cdot \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a \cdot \alpha + b \cdot \beta \\ c \cdot \alpha + d \cdot \beta \end{pmatrix}.$$

Man nennt die zu einer unitären Matrix gehörende Abbildung zwischen Vektoren eine *unitäre Transformation*. In der Folge unterscheiden wir jedoch nicht allzu streng zwischen diesen Begriffen.

**Beispiel 2.3:** Wir wenden die Matrix  $H$  aus Beispiel 2.2 auf die Basiszustände an:

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

Wenden wir  $H$  nun noch einmal an, ergibt sich:

$$\begin{aligned} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) &\xrightarrow{H} |0\rangle \text{ und} \\ \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) &\xrightarrow{H} |1\rangle. \end{aligned}$$

$H$  ist zu sich selbst invers:  $H^{-1} = H$  oder  $HH = I$ . Da sogar  $H^\dagger = H$  gilt, folgt daraus, dass  $H$  unitär ist, siehe Aufgabe 2.4.

**Aufgabe 2.5:** Konstruieren Sie alle unitären Transformationen eines Bits, die den Zustand  $|0\rangle$  auf  $\frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle$  abbilden.

Seit Ende des letzten Abschnitts stellen wir uns ein Quantenbit als von seiner Umwelt abgeschottetes Teilchen vor. Der Zustand gewisser Teilchenarten lässt sich mit Hilfe von Laserstrahlen oder Magnetfeldern ändern. Mit diesen oder ähnlichen Verfahren könnte man beispielsweise eine Superposition zweier Energieniveaus in eine andere Superposition übergehen lassen und so eine *unitäre Transformation* auf dem Quantenbit realisieren. Eine wichtige Möglichkeit, Quantenbits zu konstruieren, stellen polarisierte Photonen oder Lichtquanten dar. Der Zustand so konstruierter Quantenbits lässt sich mit optischen Geräten wie lichtbrechenden Kristallen manipulieren – mehr dazu findet sich in Kapitel 10.2.

Realisierung

### Zusammenfassung

Der Zustand eines Quantenbits wird durch einen Vektor der Länge 1 in einem zweidimensionalen komplexen Vektorraum beschrieben. Wir führen einen Rechenschritt aus, ändern den Zustand des Quantenbits, indem wir eine

unitäre Transformation anwenden beziehungsweise den Zustandsvektor mit einer unitären Matrix multiplizieren. Messen wir ein Quantenbit im Zustand  $\alpha|0\rangle + \beta|1\rangle$ , beobachten wir  $|0\rangle$  mit Wahrscheinlichkeit  $|\alpha|^2$  und  $|1\rangle$  mit Wahrscheinlichkeit  $1 - |\alpha|^2 = |\beta|^2$ . Nach der Messung ist die vorhergehende Superposition zerstört. Insbesondere kann man nie feststellen, welchen Wert die Amplituden  $\alpha$  und  $\beta$  haben.

## 2.4 Der erste Algorithmus: Ein Zufallsgenerator

Bisher stehen uns nur sehr bescheidene Mittel zur Verfügung. Wir können ein Quantenbit manipulieren und messen. Dies genügt jedoch, um einen Quantenalgorithmus für ein Problem zu beschreiben, das für einen klassischen Rechner unmöglich ist: das Erzeugen von Zufallszahlen. Zufallszahlen sind für viele Anwendungen nötig, wovon zwei in diesem Buch behandelt werden. In Abschnitt 4.2 geht es um randomisierte Algorithmen, ein sehr mächtiges Werkzeug. Eine Anwendung aus der Kryptographie lernen wir in Abschnitt 7.1 kennen. Weiter spielen Zufallszahlen für die Vorhersage von Börsenkursen oder anderen wirtschaftlichen Indizes eine große Rolle. Auch für Simulationen, etwa der Klimaentwicklung oder anderer Naturprozesse, werden Zufallszahlen benötigt. Solche Prozesse sind so komplex, dass sie nicht absolut korrekt berechnet werden können. Man kann die Prozesse aber annähern, sie approximieren; dabei kommt der Zufall ins Spiel.

Klassische Computer erzeugen für jede Eingabe eine exakt festgelegte Ausgabe. Mehr noch, jeder Rechenschritt ist durch den aktuellen Zustand determiniert, wie wir an den Berechnungsmodellen in Abschnitt 2.1 gesehen haben. Das Beste, was solche Rechner herstellen können, sind sogenannte *Pseudozufallszahlen*. Das sind Folgen von Zahlen, die mit Mitteln der Statistik nicht von tatsächlichen Zufallszahlen zu unterscheiden sind. Aber der Statistik sind bei solchen Fragen Grenzen gesetzt. Es sind keine echten Zufallszahlen und für Anwendungen äußerst problematisch.

Der folgende Algorithmus erzeugt ein zufälliges Bit. Da die möglichen Ergebnisse beide mit derselben Wahrscheinlichkeit erzeugt werden, verhält er sich wie ein Münzwurf. Die Notation wird in der anschließenden Analyse erklärt.

### Algorithmus Münzwurf

Spezifikation: Wir verwenden ein Quantenbit  $|x\rangle$ . Nach Ablauf des Algorithmus ist es mit Wahrscheinlichkeit  $1/2$  auf  $|0\rangle$  gesetzt und mit Wahrscheinlichkeit  $1/2$  auf  $|1\rangle$ .

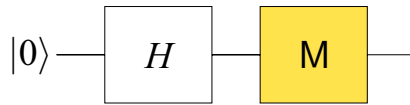


Abbildung 2.12: Schaltkreis für den Algorithmus Münzwurf

1.  $|x\rangle \leftarrow |0\rangle$
2.  $|x\rangle \leftarrow H|x\rangle$
3. Miss  $|x\rangle$

Die verwendete Notation wird durch die Analyse erklärt.

### Analyse:

In Schritt 1 wird das Quantenbit  $|x\rangle$  in den Anfangszustand  $|0\rangle$  versetzt. In Schritt 2 wird die Hadamard-Transformation auf dieses Bit angewendet, danach ist es also im Zustand  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ . Messen wir es, so erhalten wir das in der Spezifikation beschriebene Ergebnis.

**Aufgabe 2.6:** Was ist das Ergebnis des Algorithmus, wenn wir im ersten Schritt das Bit  $|x\rangle$  auf  $|1\rangle$  setzen?

**Aufgabe 2.7:** Was ist das Ergebnis des Algorithmus, wenn wir  $|x\rangle$  im ersten Schritt in einen beliebigen Ausgangszustand  $\alpha|0\rangle + \beta|1\rangle$ ,  $|\alpha|^2 + |\beta|^2 = 1$ , versetzen?

Alle unsere Algorithmen werden von ungefähr dieser Form sein. Man versetzt zunächst einige Quantenbits in einen Ausgangszustand. Dann wendet man eine Reihe von unitären Transformationen an und misst am Ende. Wie angekündigt, ist das grundlegende Berechnungsmodell dieses Buches der Quantenschaltkreis. Ein Gatter eines solchen Schaltkreises führt eine unitäre Transformation aus. Im Abschnitt 3.3 werden wir Quantenschaltkreise näher untersuchen. Der Begriff ist aber so anschaulich, dass wir in Abbildung 2.12 den Schaltkreis angeben, der unserem Algorithmus für den Münzwurf entspricht.

Wir haben bereits eine gewisse Vorstellung davon, was eine Quantenberechnung ist. Folgendermaßen ließe sich der Zufallsgenerator realisieren: man isoliert ein Teilchen, überführt dieses bezüglich seiner Drehrichtung in die Superposition  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  und misst anschließend die Drehrichtung. Das ist alles. Zufälligkeit ist schließlich eines der Charakteristika der Quantenmechanik. Quantenzufallsgeneratoren werden seit längerem produziert und eingesetzt, siehe dazu auch Abschnitt 10.2.2.

Realisierung

## 2.5 Quantenregister

Nun setzen wir mehrere Quantenbits zusammen: Für uns ist ein *Register* nichts anderes als eine Folge von Bits. Ein klassisches Bit kann die Werte 0 und 1 annehmen, die möglichen Zustände eines klassischen Registers mit zum Beispiel drei Bits sind gerade 000, 001, 010 und so weiter bis 111. Mit Hilfe der Binärdarstellung ganzer Zahlen kann ein Register aus  $n$  Bits die Zahlen  $0, \dots, 2^n - 1$  darstellen. Wie nicht anders zu erwarten, ist der Zustand eines Quantenregisters aus  $n$  Quantenbits eine Superposition der  $2^n$  klassischen Zustände.

Bevor wir Quantenregister formal definieren und eine handliche Notation einführen, untersuchen wir am Beispiel eines Registers aus zwei Quantenbits die folgende Frage: *Wie hängt der Zustand eines Quantenregisters mit dem Zustand der einzelnen Bits zusammen?*

Quantenregister  
aus zwei Bits

Unser Register besteht aus den Quantenbits  $|x_0\rangle$  und  $|x_1\rangle$ :

$$R = |x_1\rangle|x_0\rangle.$$

Diese Schreibweise erinnert nicht ohne Grund an ein Produkt. Sei das erste Quantenbit im Zustand

$$|x_0\rangle = \gamma_0|0\rangle + \gamma_1|1\rangle$$

und das zweite im Zustand

$$|x_1\rangle = \beta_0|0\rangle + \beta_1|1\rangle.$$

Dann erhalten wir den Zustand des Quantenregisters durch Ausmultiplizieren; im Abschnitt 2.7 werden wir dieses Vorgehen mathematisch fundieren.

$$\begin{aligned} R &= |x_1\rangle|x_0\rangle \\ &= (\beta_0|0\rangle + \beta_1|1\rangle) \cdot (\gamma_0|0\rangle + \gamma_1|1\rangle) \\ &= \beta_0\gamma_0|0\rangle|0\rangle + \beta_0\gamma_1|0\rangle|1\rangle + \beta_1\gamma_0|1\rangle|0\rangle + \beta_1\gamma_1|1\rangle|1\rangle. \end{aligned}$$

Das ist unübersichtlich: Wir bezeichnen die Amplituden der Zustände des Quantenregisters mit  $\alpha_{ij} = \beta_i\gamma_j$  und erhalten:

$$R = \alpha_{00}|0\rangle|0\rangle + \alpha_{01}|0\rangle|1\rangle + \alpha_{10}|1\rangle|0\rangle + \alpha_{11}|1\rangle|1\rangle.$$

Aus  $|\gamma_0|^2 + |\gamma_1|^2 = 1$  und  $|\beta_0|^2 + |\beta_1|^2 = 1$  folgt  $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$ . Der Einfachheit halber werden wir statt  $R = |x_1\rangle|x_0\rangle$  meistens  $R = |x_1x_0\rangle$  schreiben. Danach sieht der Zustand des Registers folgendermaßen aus:

$$R = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle.$$

Außerdem ist es oft übersichtlicher, anstelle der einzelnen Bits, die in der Binärdarstellung repräsentierten Zahlen anzugeben. Wir erhalten:

$$R = \alpha_0|0\rangle + \alpha_1|1\rangle + \alpha_2|2\rangle + \alpha_3|3\rangle.$$



Es sei noch einmal darauf hingewiesen: die Amplituden  $\alpha_0, \dots, \alpha_3$  ergeben sich aus den Amplituden der beiden Bits.

**Aufgabe 2.8:** Wir betrachten ein Zwei-Bit-Register. Bit  $|x_1\rangle$  ist im Zustand  $\frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle$  und Bit  $|x_0\rangle$  im Zustand  $\frac{1}{2}|0\rangle - \frac{\sqrt{3}}{2}|1\rangle$ . Welche Amplitude haben die Zwei-Bit-Zustände  $|00\rangle, \dots, |11\rangle$ ?

Nun die formale Definition:

Ein Quantenregister  $R = |x_{n-1}\rangle \dots |x_1\rangle |x_0\rangle = |x_{n-1} \dots x_1 x_0\rangle$  kann sich in Zuständen der folgenden Form befinden:

$$R = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle.$$

Dabei bedeutet  $|i\rangle$  für  $i = 0, \dots, 2^n - 1$ : Die Bits sind entsprechend der Binärdarstellung der natürlichen Zahl  $i$  gesetzt.

Es gilt:

$$\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1.$$

Misst man dieses Quantenregister, so beobachtet man mit Wahrscheinlichkeit  $|\alpha_i|^2$  den Zustand  $|i\rangle$ .

Definition  
Quantenregister

Zustände eines Quantenregisters mit  $n$  Bits sind Vektoren in einem  $2^n$ -dimensionalen komplexen Vektorraum. Die Basis bilden die einzelnen Komponenten der Superposition, die möglichen Zustände eines klassischen Registers also:

$$|0 \dots 0\rangle, |0 \dots 01\rangle, \dots, |1 \dots 1\rangle$$

oder kompakter geschrieben

$$|0\rangle, |1\rangle, \dots, |2^n - 1\rangle.$$

Zustandsvektoren

**Beispiel 2.4:** Im Fall von zwei Bits ist die Basis

$$|00\rangle, |01\rangle, |10\rangle, |11\rangle$$

mit der Zuordnung

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Entwicklung  
eines  
Quantenregisters

Die Zustandsvektoren eines  $n$ -Bit Quantenregisters haben  $2^n$  Komponenten. Entsprechend sind die Rechenschritte unitäre Transformationen, die durch  $2^n \times 2^n$ -Matrizen darstellbar sind.

**Beispiel 2.5:** Sei  $n = 2$ . Die folgende Operation namens CNOT ist die unitäre Variante des exklusiven Oder:

$$\text{CNOT} : |x, y\rangle \mapsto |x, x \oplus y\rangle$$

Die Bezeichnung CNOT ist die Abkürzung für *controlled not*: ist das erste Bit 1, so wird das zweite Bit negiert. Anderenfalls bleibt das zweite Bit unverändert.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

ist die zugehörige Matrix. Abbildung 2.13 zeigt das Symbol eines CNOT-Gatters.  $\diamond$

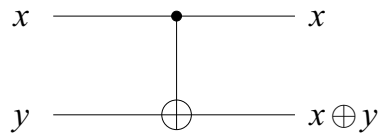


Abbildung 2.13: CNOT-Gatter

Tipp zu Matrizen

Mit unserer Definition eines Zustandsvektors und der einer unitären Transformation als Multiplikation dieses Vektors mit einer Matrix ist eigentlich alles gesagt. Anschaulich verbirgt sich dahinter das Folgende:

Betrachten wir eine Matrix, die einer Transformation auf  $n$  Bits entspricht. Wir stellen uns vor, die Zeilen seien von oben nach unten mit den Zuständen beschriftet,  $|0\rangle, \dots, |2^n - 1\rangle$ , und die Spalten von links nach rechts. Die mit  $|i\rangle$  bezeichnete Spalte beschreibt das Bild des Basisvektors  $|i\rangle$ . Lautet diese

$$a_{0i} \dots a_{2^n-1,i},$$

so wird  $|i\rangle$  auf

$$a_{0i}|0\rangle + a_{1i}|1\rangle + \dots + a_{2^n-1,i}|2^n - 1\rangle$$

abgebildet. Wir demonstrieren die Nummerierung von Zeilen und Spalten an der Operation CNOT.

$$\begin{array}{l} |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{array} \begin{pmatrix} |00\rangle|01\rangle|10\rangle|11\rangle \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

So sieht man sofort, dass diese Operation die Amplituden von  $|10\rangle$  und  $|11\rangle$  tauscht:

$$\text{CNOT} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_2 \end{pmatrix}.$$

Es handelt sich um eine *Permutationsmatrix*.

**Beispiel 2.6:** Wir betrachten Matrizen von der folgenden Form. In jeder Spalte und in jeder Zeile gibt es genau eine 1, alle anderen Einträge sind 0. Diese Matrizen heißen *Permutationsmatrizen*. Angewandt auf einen Vektor, verändert eine solche Matrix die Reihenfolge der Komponenten, ändert aber ihren Wert nicht. Auf einen Zustandsvektor angewandt, wird die Zuordnung von Amplituden zu Basiszuständen geändert.

**Aufgabe 2.9:** Zeigen Sie, dass Permutationsmatrizen unitär sind.

Die folgende Definition ist wesentlich: Zwar wissen wir, wie sich ein Quantenregister entwickelt. Aber wir haben noch nicht gesagt, was ein *einfacher Rechenschritt* ist.

Einfache Rechenschritte auf einem Quantenregister sind *lokale unitäre Transformationen*. Lokal sind Transformationen, an denen höchstens drei Bits beteiligt sind.

Lokalität

Wenn wir also eine unitäre Transformation verwenden, die auf dem ganzen Register operiert, interessiert uns die Zerlegung in lokale Transformationen. Der Grund dafür ist leicht einzusehen: Eine Registeroperation, die von einer  $2^n \times 2^n$ -Matrix beschrieben wird, kann sehr kompliziert und nur mit großem Aufwand ausführbar sein. Dagegen kann eine Drei-Bit-Transformation (zumindest theoretisch) von einfachen physikalischen Systemen realisiert werden.

Die Zahl *drei* haben wir nach praktischen Gesichtspunkten festgelegt. Im Prinzip ist jede unitäre Operation lokal, an der *konstant* viele Bits beteiligt sind; diese Anzahl wächst also nicht mit der Registergröße. Grundsätzlich käme man mit Transformationen zweier Bits aus. Um klassische Rechenverfahren in Quantenalgorithmen zu überführen, ist jedoch ein Drei-Bit-Gatter nötig (siehe Abschnitt 3.5).

Derartige *Zerlegungen* von Transformationen sind das Thema des Abschnitts 2.7. Wir untersuchen dort, was dem Übergang vom Bit zum Register mathematisch entspricht; was also der Zusammenhang zwischen der Entwicklung eines Registers und der Entwicklung seiner einzelnen Bits ist. Das Stichwort dazu lautet *Tensorprodukt*. Zuvor besprechen wir wichtige Eigenschaften unitärer Transformationen:

Eine Matrix  $A$  ist unitär, wenn  $A^{-1} = A^\dagger$  gilt. Das ist eine sehr starke Forderung, woraus sich einige wichtige Eigenschaften der zugehörigen unitären Transformation ergeben.

### Eigenschaften unitärer Transfor- mationen

Für unitäre Transformationen gilt:

1. Jeder zulässige Zustand eines Quantenbits wird wieder auf einen zulässigen Zustand abgebildet; unitäre Transformationen sind län-  
generhaltend.
2. Unitäre Transformationen verändern das Skalarprodukt zweier  
Zustandsvektoren nicht: unitäre Transformationen sind winkeler-  
haltend. Geometrisch betrachtet, dreht eine unitäre Transformation  
Zustandsvektoren um den 0-Punkt oder spiegelt sie an einer Achse  
durch den Ursprung.
3. Unitäre Transformationen sind umkehrbar. Daraus folgt eine wichti-  
ge Beschränkung für Berechnungen eines Quantencomputers: jeder  
Schritt muss umkehrbar sein.
4. Unitäre Transformationen sind linear, da sie durch Matrizen be-  
schrieben werden können, siehe A.3.

### Skalarprodukt

Für das Skalarprodukt zweier Zustandsvektoren  $|u\rangle, |v\rangle$  verwenden wir die  
Notation  $\langle u | v \rangle$ . Das ist die *Braket-Notation* (von engl. *bracket*). Für Vektoren  
 $|u\rangle = (\alpha_0, \dots, \alpha_{n-1})^T$  und  $|v\rangle = (\beta_0, \dots, \beta_{n-1})^T$  mit  $n$  komplexen Kompen-  
ten gilt

$$\langle u | v \rangle = \alpha_0^* \beta_0 + \dots + \alpha_{n-1}^* \beta_{n-1}.$$

Die Länge eines Vektors  $|u\rangle$ , seine *Norm*, ist als

$$\|u\| = \sqrt{\langle u | u \rangle}$$

definiert und ihr Quadrat ist gleich dem uns geläufigen Wert

$$|\alpha_0|^2 + |\alpha_1|^2 + \dots + |\alpha_{n-1}|^2.$$

Abschnitt A.2.3 des Anhangs behandelt diesen Zusammenhang und be-  
schreibt, wie das Skalarprodukt den Winkel zwischen zwei Vektoren bestimmt.  
Die ersten beiden in obiger Box genannten Eigenschaften lauten formal:

1. Für einen Vektor  $|\phi\rangle$  und eine unitäre Transformation  $U$  gilt

$$\|U|\phi\rangle\| = \|\phi\rangle\|.$$

2. Für zwei Vektoren  $|\phi\rangle, |\psi\rangle$  und eine unitäre Transformation  $U$  gilt

$$\langle U\phi | U\psi \rangle = \langle \phi | \psi \rangle.$$

Da der Abstand mit Hilfe des Skalarproduktes definiert ist, folgt die erste Aussage aus der zweiten.

## 2.6 Der zweite Algorithmus: Das Problem von Deutsch

In der Quantenwelt kann man zwei Seiten einer Münze gleichzeitig betrachten. Das wundert uns gar nicht mehr. Nun werden wir sehen, wie man daraus einen Vorteil ziehen kann. Nehmen wir einmal an, wir haben eine Superposition der beiden Münzseiten erzeugt. Wir *messen* diese Superposition, um etwas über die Seiten zu erfahren. Dann erhalten wir doch nur über *eine* Münzseite Informationen. Geht man geschickt vor, lässt sich dennoch etwas über beide Seiten erfahren. Das folgende Problem ist nach David Deutsch benannt, einem der Väter des Quantencomputers; der 1953 in Israel geborene wuchs in England auf und lebt in Oxford.

Wir sollen herausbekommen, ob eine Münze echt oder eine plumpe Fälschung ist. Die Vorderseite der echten Münze zeigt eine Zahl, die Rückseite das Brandenburger Tor. Bei der falschen Münze sind beide Seiten gleich. Wie oft müssen wir uns die Münze ansehen, um eine echte Münze von einer Fälschung zu unterscheiden? Zweimal, da wir beide Seiten betrachten müssen. Ein Quantencomputer kann in einer ähnlichen Situation echte Vorteile bringen.

Ist die Münze echt?

Wir sollen etwas über eine Funktion  $f$  herausfinden, die ein Bit als Eingabe bekommt und ein Bit ausgibt,  $f : \{0, 1\} \rightarrow \{0, 1\}$ . Uns wird ein Bauteil zur Verfügung gestellt, das uns zu einem Bit  $b$  den Wert  $f(b)$  liefert; andere Informationen über  $f$  haben wir nicht. Man sagt,  $f$  ist uns als *Black Box* gegeben. Wir bevorzugen in der Folge das deutsche Wort *Orakel*.

**Aufgabe 2.10:** Geben Sie alle Funktionen  $f : \{0, 1\} \rightarrow \{0, 1\}$  an.

Für eine solche Funktion gilt, sie ist entweder *konstant*, dann gilt  $f(0) = f(1)$  oder sie ist *balanciert*:  $f(0) \neq f(1)$ .

Wir sollen folgende Frage beantworten: Ist die uns als Orakel vorliegende Funktion konstant oder balanciert?

Ein klassischer Computer muss die Funktion an zwei Stellen abfragen. Angenommen, wir beginnen damit, nach  $f(0)$  zu fragen. Dann hilft uns die Antwort nichts, wenn wir nicht auch  $f(1)$  kennen, so wie wir beide Seiten der Münze betrachten müssen. Wir betrachten nun einen Quantenalgorithmus, der dieses Problem mit nur einem Orakelaufruf löst.

## Reversibles Orakel

Der Quantenalgorithmus wird ein Quantenbit in eine Superposition über beide möglichen Eingaben von  $f$  versetzen, dann auf dieses Quantenbit das Orakel anwenden und eine Superposition über beide Funktionswerte bekommen. Da aber alle Rechenschritte umkehrbar sein müssen, brauchen wir die Funktion  $f$  in einer reversiblen Version.  $f$  ist nicht umkehrbar, falls sie konstant ist. Darum verwenden wir zwei Bits  $|x\rangle$  und  $|y\rangle$  und folgende unitäre Form des Funktionsaufrufes

$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle.$$

Die Operation  $\oplus$  ist die dem exklusiven Oder entsprechende Addition von Bits, siehe Seite 16. Es gilt  $U_f^{-1} = U_f$ , also ist  $U_f$  reversibel.

**Aufgabe 2.11:** Zeigen Sie, dass  $U_f$  unitär ist.

Nun können wir den Algorithmus angeben. Er verwendet ein Register aus zwei Quantenbits. Dem Algorithmus entspricht der Schaltkreis aus Abbildung 2.14.

### Der Algorithmus für das Problem von Deutsch

1.  $|x\rangle|y\rangle \leftarrow |0\rangle|1\rangle$
2. Wende die Hadamard-Transformation  $H$  auf beide Bits an:  
 $|x\rangle|y\rangle \leftarrow H|x\rangle H|y\rangle$
3. Wende  $f$  aus:  
 $|x\rangle|y\rangle \leftarrow U_f|x\rangle|y\rangle$
4. Wende die Hadamard-Transformation  $H$  auf beide Bits an:  
 $|x\rangle|y\rangle \leftarrow H|x\rangle H|y\rangle$
5. Miss das Register:  
Hat  $|x\rangle|y\rangle$  den Wert  $|0\rangle|1\rangle$ : Ausgabe *konstant*,  
hat  $|x\rangle|y\rangle$  den Wert  $|1\rangle|1\rangle$ : Ausgabe *balanciert*.

Dem Algorithmus ist nicht auf den ersten Blick anzusehen, warum er unser Problem löst. Darum werden wir sein Vorgehen detailliert betrachten. Festzuhalten ist:  $U_f$  wurde nur einmal ausgeführt! Vielleicht ist noch die folgende Bemerkung zu Schritt 5 angebracht. Der Test, ob das Register einen bestimmten Wert hat, und auch die Ausgabe des Ergebnisses kann von einem klassischen Rechner übernommen werden. Dies erscheint als das praktikabelste Vorgehen. Wir werden jedoch in Kapitel 3.2 sehen, dass ein Quantencomputer theoretisch all das kann, wozu ein klassischer Rechner fähig ist.

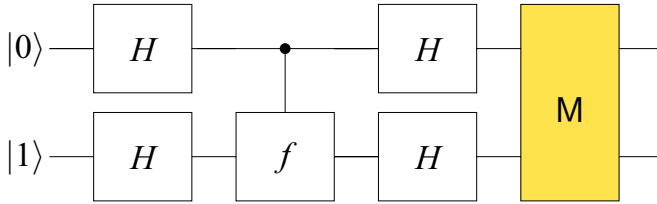


Abbildung 2.14: Schaltkreis für das Problem von Deutsch

**Analyse**

In Schritt 2 wird unser Zwei-Bit Quantenregister  $|x\rangle|y\rangle$  durch die Hadamard-Transformation von dem Zustand  $|0\rangle|1\rangle$  in den Zustand

$$|\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

überführt. Der erste Faktor entspricht dem Zustand von Bit  $|x\rangle$ , der zweite dem von Bit  $|y\rangle$ . Ausmultiplizieren ergibt

$$|\phi_2\rangle = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle).$$

Nun haben wir eine Superposition über alle Basiszustände unseres Registers, wobei die Vorzeichen eine wichtige Rolle spielen. Schritt 3 wendet  $U_f$  an. Der Inhalt des ersten Registers dient als Eingabe der Funktion  $f$ , der Funktionswert wird auf das zweite Register addiert.

$$\begin{aligned} |\phi_2\rangle &= \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle) \\ &\xrightarrow{U_f} \frac{1}{2}(|0\rangle|0 \oplus f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|0 \oplus f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle) \\ &= \frac{1}{2}(|0\rangle \cdot (|f(0)\rangle - |1 \oplus f(0)\rangle) + |1\rangle \cdot (|f(1)\rangle - |1 \oplus f(1)\rangle)) \\ &= |\phi_3\rangle. \end{aligned}$$

Wir haben unseren ursprünglichen Plan ausgeführt und eine Superposition über die Funktionswerte erhalten. Würden wir nun messen, bekämen wir mit einer Wahrscheinlichkeit von jeweils 1/4 einen der Werte  $|0\rangle|0\rangle, |0\rangle|1\rangle, |1\rangle|0\rangle$  oder  $|1\rangle|1\rangle$  geliefert, was uns keine Informationen für unser Problem liefert. Darum wenden wir die Hadamard-Transformation ein zweites Mal an. Zuvor stellen wir fest, dass für die beiden Bitwerte  $x$  gleich 0, 1

$$|f(x)\rangle - |1 \oplus f(x)\rangle = (-1)^{f(x)}(|0\rangle - |1\rangle)$$

gilt. Wir können den Registerzustand umformen in

$$\begin{aligned} |\phi_3\rangle &= \frac{1}{2}((-1)^{f(0)}|0\rangle \cdot (|0\rangle - |1\rangle) + (-1)^{f(1)}|1\rangle \cdot (|0\rangle - |1\rangle)) \\ &= \frac{1}{2}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \cdot (|0\rangle - |1\rangle). \end{aligned}$$

Der Funktionswert ist in das Vorzeichen der Amplitude verlagert worden. Das erste Bit in  $|\phi_3\rangle$  befindet sich im Zustand

$$\frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle).$$

**Schritt 4** Nun betrachten wir den vierten Schritt für beide der möglichen Eigenschaften der Funktion separat.

**Fall: konstant** Wir nehmen zunächst an,  $f$  sei konstant. In diesem Fall gilt  $(-1)^{f(0)} = (-1)^{f(1)}$ , und der Zustand von Bit  $|x\rangle$  ist entweder  $1/\sqrt{2} \cdot (|0\rangle + |1\rangle)$  oder  $-1/\sqrt{2} \cdot (|0\rangle + |1\rangle)$ . Die Hadamard-Transformation bildet  $1/\sqrt{2} \cdot (|0\rangle + |1\rangle)$  auf  $|0\rangle$  und  $-1/\sqrt{2} \cdot (|0\rangle + |1\rangle)$  auf  $-|0\rangle$  ab. Denn unitäre Transformationen sind linear, siehe Abschnitt A.3.

$|y\rangle$  befindet sich vor Schritt 4 im Zustand  $1/\sqrt{2} \cdot (|0\rangle - |1\rangle)$  und wird auf  $|1\rangle$  abgebildet. Schritt 4 überführt das ganze Register somit in den Zustand

$$|\phi_4^k\rangle = \pm|0\rangle|1\rangle.$$

Dabei bedeutet  $a = \pm b$ , dass einer der Fälle  $a = b$  oder  $a = -b$  zutrifft.

**Fall: balanciert** Ist  $f$  balanciert, ist  $|x\rangle$  vor Schritt 4 im Zustand  $1/\sqrt{2} \cdot (|0\rangle - |1\rangle)$  oder im Zustand  $-1/\sqrt{2} \cdot (|0\rangle - |1\rangle)$ . Die Hadamard-Transformation bildet  $1/\sqrt{2} \cdot (|0\rangle - |1\rangle)$  auf  $|1\rangle$  ab, und analog zum anderen Fall bekommen wir als Folge von Schritt 4

$$|\phi_4^b\rangle = \pm|1\rangle|1\rangle.$$

**Schritt 5** Messen liefert also für eine balancierte Funktion  $|1\rangle|1\rangle$  und für eine konstante Funktion  $|0\rangle|1\rangle$ .  $\diamond$

Der erste Quantenalgorithmus für dieses Problem wurde 1985 von Deutsch veröffentlicht.

**Aufgabe 2.12:** Wenden Sie die Hadamard-Transformation auf  $-(|0\rangle + |1\rangle)$  an.

### Beispielrechnung

Falls noch Unklarheiten bestehen, ist das folgende Beispiel vielleicht hilfreich. Danach sollte man dann die Analyse noch einmal durchgehen. Uns sei die Funktion  $f$  mit  $f(0) = 1$  und  $f(1) = 0$  gegeben,  $f$  ist also die Negation. Die Schritte 1 und 2 sind von der Funktion unabhängig, und wir verweisen auf die Analyse. Schritt 3 überführt unser Register aus zwei Bits dann aus dem Zustand

$$|\phi_2\rangle = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle)$$

in

$$\begin{aligned} |\phi_3\rangle &= \frac{1}{2}(|0\rangle|0 \oplus f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|0 \oplus f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle) \\ &= \frac{1}{2}(|0\rangle|1\rangle - |0\rangle|0\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle). \end{aligned}$$



Um die Wirkung von  $H$  auf das erste Bit zu untersuchen, klammern wir es aus:

$$\begin{aligned} |\phi_3\rangle &= \frac{1}{2} \left( -|0\rangle \cdot (|0\rangle - |1\rangle) + |1\rangle \cdot (|0\rangle - |1\rangle) \right) \\ &= \frac{1}{2} (|1\rangle - |0\rangle) \cdot (|0\rangle - |1\rangle). \end{aligned}$$

$H$  bildet  $|1\rangle - |0\rangle$  auf  $-\sqrt{2}|1\rangle$  ab und  $|0\rangle - |1\rangle$  auf  $\sqrt{2}|1\rangle$ . Schritt 4 erzeugt also den Registerzustand

$$|\phi_4\rangle = \frac{1}{2} (-\sqrt{2}|1\rangle \cdot \sqrt{2}|1\rangle) = -|1\rangle|1\rangle.$$

Messen liefert  $|1\rangle|1\rangle$  mit Wahrscheinlichkeit  $(-1)^2 = 1$ . Der Algorithmus gibt *balanciert* aus.

**Aufgabe 2.13:** Rechnen Sie den Algorithmus für die 1-Funktion mit  $f(0) = f(1) = 1$  durch. Betrachten Sie alle fünf Schritte.

Die meisten Quantenalgorithmen nutzen ein Phänomen namens *Interferenz*. Amplituden können sich zu einem größeren Wert addieren, das nennt man konstruktive Interferenz, aber auch gegenseitig auslöschen; man spricht von destruktiver Interferenz. Wir untersuchen dahingehend unseren Algorithmus für das Problem von Deutsch. In der Analyse ist erwähnt, dass wir aus einer Messung nach Schritt 3 nicht erfahren, ob die Funktion  $f$  konstant oder balanciert ist. Wie gelingt es nun, mit der zweiten Hadamardtransformation diese Information aus der Superposition  $|\phi_3\rangle$  zu extrahieren? Dazu betrachten wir die Wirkung auf das erste Bit im Fall, dass  $f$  konstant ist:

$$\pm \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \xrightarrow{H} \pm \frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right).$$

Im Resultat dieser Transformation addieren sich für  $|0\rangle$  und  $|1\rangle$  je zwei Amplituden. Die von  $|0\rangle$  ist  $1/2 + 1/2 = 1$ , die von  $|1\rangle$  ist  $1/2 - 1/2 = 0$ . Ist  $f$  hingegen balanciert, lässt die Anwendung von  $H$  die Amplitude von  $|0\rangle$  durch destruktive Interferenz verschwinden und vergrößert die von  $|1\rangle$ . Wir kommen in den Abschnitten 4.4 und 10.2 darauf zurück.

Die Analyse des Algorithmus hat gezeigt, dass er das gestellte Problem löst. Vielleicht ist beim Leser noch kein intuitives Verständnis seines Vorgehens entstanden. Darum werden wir noch einmal auf das Problem von Deutsch zurückkommen, und zwar in Abschnitt 2.9.

## 2.7 Die Rolle des Tensorprodukts

Im vorletzten Abschnitt haben wir mehrere Bits zusammengesteckt und sind so zum Register gelangt. Nun tun wir das gleiche mit den Rechenschritten.

Wir werden sehen, wie sich die Operationen auf den Bits zu Operationen auf dem ganzen Register verhalten.

Der Zustand eines Quantenregisters aus  $n$  Bits wird durch einen Vektor eines  $2^n$ -dimensionalen Vektorraums beschrieben; das haben wir bereits gesehen. Es gibt ein mathematisches Instrument, mit dem Vektorräume für Register aus solchen für einzelne Bits zusammengesetzt werden können: das Tensorprodukt. Damit können außerdem Rechnungen auf einem Register aus solchen auf den einzelnen Bits konstruiert werden.

Produkt von  
Räumen

Seien  $V_1$  und  $V_2$  zwei Vektorräume. Sei  $e_0, \dots, e_{m-1}$  eine Basis von  $V_1$  und  $f_0, \dots, f_{n-1}$  eine Basis von  $V_2$ . Das *Tensorprodukt*

$$V_1 \otimes V_2$$

dieser Räume ist ein  $(m \cdot n)$ -dimensionaler Vektorraum. Seine Basisvektoren bezeichnen wir mit

$$\begin{array}{cccc} e_0 \otimes f_0, & e_0 \otimes f_1, & \dots, & e_0 \otimes f_{n-1}, \\ e_1 \otimes f_0, & e_1 \otimes f_1, & \dots, & e_1 \otimes f_{n-1}, \\ \vdots & \vdots & & \vdots \\ e_{m-1} \otimes f_0, & e_{m-1} \otimes f_1, & \dots, & e_{m-1} \otimes f_{n-1}. \end{array}$$

Produkt von  
Vektoren

Die Rolle dieser Basisvektoren wird durch die folgende Definition deutlich: wir betrachten das Tensorprodukt zweier *Vektoren* der Ausgangsräume. Sei

$$v_1 = \alpha_0 e_0 + \dots + \alpha_{m-1} e_{m-1} = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{m-1} \end{pmatrix}$$

ein Vektor aus  $V_1$  und

$$v_2 = \beta_0 f_0 + \dots + \beta_{n-1} f_{n-1} = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_{n-1} \end{pmatrix}$$

aus  $V_2$ . Ihr Tensorprodukt ist

$$v_1 \otimes v_2 = \left( \sum_{i=0}^{m-1} \alpha_i e_i \right) \otimes \left( \sum_{j=0}^{n-1} \beta_j f_j \right) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \alpha_i \beta_j (e_i \otimes f_j).$$

**Beispiel 2.7:** Im Fall  $n = m = 2$  gilt

$$\begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} \otimes \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} \alpha_0 \beta_0 \\ \alpha_0 \beta_1 \\ \alpha_1 \beta_0 \\ \alpha_1 \beta_1 \end{pmatrix}.$$

**Beispiel 2.8:** Für unsere Zwecke ist das Tensorprodukt einfach und anschaulich zu handhaben. Aus den Räumen für zwei Quantenbits  $|x\rangle$  und  $|y\rangle$  mit Basen  $\{|0\rangle, |1\rangle\}$  erhalten wir durch das Tensorprodukt einen Raum für das Register aus diesen beiden Bits. Die Basis ist

Produkt von  
Qubits

$$\begin{aligned} & \{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\} \\ &= \{|0\rangle|0\rangle, |0\rangle|1\rangle, |1\rangle|0\rangle, |1\rangle|1\rangle\} \\ &= \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}. \end{aligned}$$

Zur Verdeutlichung dienen die Koordinatenvektoren aus Beispiel 2.4: Beispielsweise entspricht  $|1\rangle \otimes |0\rangle = |10\rangle$  gerade

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

Nun betrachten wir zwei Zustände, die nicht zur Basis gehören:

$$|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle, |\psi\rangle = \beta_0|0\rangle + \beta_1|1\rangle.$$

Gemäß Beispiel 2.7 gilt:

$$|\phi\rangle \otimes |\psi\rangle = \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle.$$

Dieses Resultat ergibt sich auch durch Ausmultiplizieren von

$$(\alpha_0|0\rangle + \alpha_1|1\rangle) \cdot (\beta_0|0\rangle + \beta_1|1\rangle).$$

**Aufgabe 2.14:** Berechnen Sie  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ .

Die Beschreibung eines Registers aus  $m$  Bits lässt sich aus dem  $m$ -fachen Tensorprodukt der Beschreibung eines Bits erzeugen. Sind die Bits

Produkt von  
Zuständen

$$|x_1\rangle, \dots, |x_m\rangle$$

in den Zuständen

$$|\phi_1\rangle, \dots, |\phi_m\rangle,$$

so befindet sich das Register

$$|x_1, \dots, x_m\rangle$$

im Zustand

$$|\phi_1\rangle \otimes \dots \otimes |\phi_m\rangle.$$

Wir können die Amplituden des neuen Zustands wie gewohnt durch Ausmultiplizieren errechnen. Darum verwenden wir statt  $\otimes$  meistens die gewöhnliche Produktschreibweise.

Genauso erhält man aus einer  $2^m$ -dimensionalen Beschreibung eines  $m$ -Bit-Registers und einer  $2^n$ -dimensionalen eines  $n$ -Bit-Registers eine Beschreibung für das gemeinsame  $m+n$ -Bit-Register in einem  $2^{m+n}$ -dimensionalen Vektorraum.

Produkt von  
Matrizen

Wie angekündigt, lässt sich das Tensorprodukt auch für Transformationen definieren. Seien  $A$  und  $B$  Matrizen mit

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}.$$

Das Tensorprodukt der Matrizen ist

$$A \otimes B = \begin{pmatrix} a_{11} \cdot B & \cdots & a_{1n} \cdot B \\ a_{21} \cdot B & \cdots & a_{2n} \cdot B \\ \vdots & \ddots & \vdots \\ a_{m1} \cdot B & \cdots & a_{mn} \cdot B \end{pmatrix}.$$

Damit können wir die Matrizen aus Beispiel 2.2 auf zwei Bits erweitern.

**Beispiel 2.9:** Das Tensorprodukt der Einheitsmatrix  $I_2$  mit sich selbst ergibt

$$I_2 \otimes I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes I_2 = \begin{pmatrix} I_2 & 0 \\ 0 & I_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I_4.$$

In der mittleren Matrix kürzt 0 eine 0-Matrix der Größe  $2 \times 2$  ab. Entsprechend ist

$$I_2 \otimes I_2 \otimes I_2 = \begin{pmatrix} I_4 & 0 \\ 0 & I_4 \end{pmatrix} = I_8 = \begin{pmatrix} I_2 & 0 & 0 & 0 \\ 0 & I_2 & 0 & 0 \\ 0 & 0 & I_2 & 0 \\ 0 & 0 & 0 & I_2 \end{pmatrix}.$$

Die  $2^n$ -dimensionale Hadamard-Transformation  $H_n$  ist als

$$H_n = \bigotimes_{i=1}^n H$$

definiert. Damit ist

$$H_2 = H \otimes H = \frac{1}{\sqrt{2}} \begin{pmatrix} H & H \\ H & -H \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}.$$

Wendet man  $H_n$  auf ein  $n$ -Bit Quantenregister an, so entspricht dies der Anwendung von  $H$  auf jedes einzelne Quantenbit.

Die Bedeutung des Tensorproduktes für Berechnungen beschreibt die folgende Definition.

Die beiden folgenden Aktionen haben dasselbe Resultat:

- Wir wenden auf die Bits

$$|x_1\rangle, \dots, |x_m\rangle$$

Transformationen an, die durch die Matrizen

$$A_1, \dots, A_m,$$

beschrieben sind (jeweils  $A_i$  auf  $|x_i\rangle$ ),

Produkt von  
Operationen

- wir wenden auf das Register

$$|x_1, \dots, x_m\rangle$$

die Transformation

$$A_1 \otimes \dots \otimes A_m$$

an.

Damit haben wir beschrieben, wie Rechenschritte auf einzelnen Bits mit solchen auf dem ganzen Register zusammenhängen. Betrachten wir zwei Bits  $|x\rangle|y\rangle$ .

- $H \otimes H = H_2$  beschreibt auf Registerebene die Anwendung von  $H$  auf  $|x\rangle$  und auf  $|y\rangle$ .
- $H \otimes I_2$  beschreibt die Anwendung von  $H$  auf  $|x\rangle$  und
- $I_2 \otimes H$  die Anwendung von  $H$  auf  $|y\rangle$ .

**Aufgabe 2.15:** Berechnen Sie die  $4 \times 4$ -Matrizen  $H \otimes I_2$  und  $I_2 \otimes H$ .

Allgemein gilt für Vektorräume: Agiert die Matrix  $A$  auf dem Vektorraum  $V_1$  und die Matrix  $B$  auf  $V_2$ , so ist die Matrix  $A \otimes B$  eine Abbildung von  $V_1 \otimes V_2$  auf sich selbst. Für Vektoren  $v_1$  aus dem Raum  $V_1$  und  $v_2$  aus  $V_2$  gilt:

$$(Av_1) \otimes (Bv_2) = (A \otimes B)(v_1 \otimes v_2).$$

Rechnung mit  $H_2$ 

**Beispiel 2.10:** Wir rechnen mit dem Zwei-Bit-Register aus Beispiel 2.8 im Zustand

$$|x\rangle|y\rangle = |0\rangle|1\rangle = |01\rangle.$$

Wenden wir  $H$  auf das erste Bit an, erhalten wir

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|1\rangle = \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|11\rangle.$$

Wir wenden  $H$  erneut an, und zwar auf das zweite Bit und bekommen

$$\begin{aligned} & \frac{1}{\sqrt{2}}|0\rangle \left( \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right) + \frac{1}{\sqrt{2}}|1\rangle \left( \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right) \\ &= \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle) \\ &= H_2|01\rangle. \end{aligned}$$

Die Reihenfolge, in der wir die Hadamardtransformationen auf die Quantenbits anwenden, spielt für das Endresultat keine Rolle.

**Aufgabe 2.16:** Beginnen Sie mit zwei Bits im Zustand  $|10\rangle$  und wenden Sie  $H$  auf das erste Bit an, dann auf das zweite, dann noch einmal auf das erste und zuletzt erneut auf das zweite.

### Anwendung: ein $n$ -Bit Zufallsgenerator

Zufallszahlen

Unser Algorithmus aus Abschnitt 2.4 erzeugte ein zufälliges Bit. Wir können ihn wie folgt verallgemeinern: Der Schaltkreis aus Abbildung 2.15 erzeugt Zufallszahlen zwischen 0 und  $2^n - 1$ . Jede Zahl wird mit der gleichen Wahrscheinlichkeit  $1/2^n$  erzeugt, die Zahlen werden unter Gleichverteilung gezogen.

1.  $R = |x_{n-1}, \dots, x_0\rangle \leftarrow |0 \dots 0\rangle$
2.  $R \leftarrow H_n R$
3. Miss  $R$

Analyse

### Analyse

Unser Register hat  $n$  Bits, alle haben den Anfangswert  $|0\rangle$  (Schritt 1). Schritt 2 wendet auf jedes einzelne die Hadamard-Transformation  $H$  an, was der Anwendung von  $H_n$  auf das Gesamtregister entspricht:

$$|0\rangle \dots |0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \dots \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

Für die ersten beiden Bits gilt

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle).$$

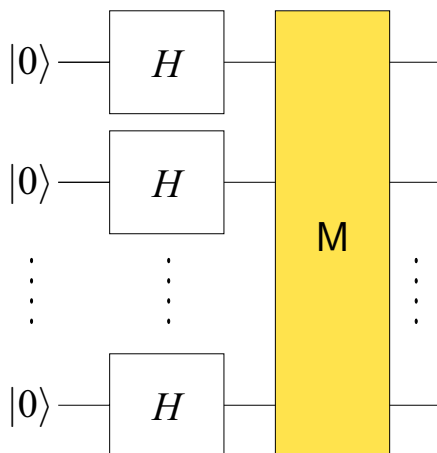


Abbildung 2.15: Der Schaltkreis des Zufallsgenerators

Multipliziert man weiter aus, erkennt man, dass sich  $R$  nach Schritt 2 im Zustand

$$\begin{aligned} & \frac{1}{\sqrt{2^n}} (|0\dots 0\rangle + |0\dots 01\rangle + \dots + |1\dots 1\rangle) \\ &= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \end{aligned}$$

befindet. Beim Messen wird jeder Basiszustand des Quantenregisters mit Wahrscheinlichkeit  $1/2^n$  beobachtet.  $\diamond$

Man nennt  $H_n|0\dots 0\rangle = H|0\rangle\dots H|0\rangle$  die *gleichgewichtete Superposition*, da jeder möglichen Bitbelegung die gleiche Amplitude zugeordnet wird. Allerdings hätte man auch nacheinander die einzelnen Bits messen können und hätte nach und nach die Binärdarstellung der Zufallszahl erzeugt. Das Messen einzelner Bits ist eines der Themen des nächsten Abschnitts.

Zum Abschluss kommen wir noch einmal auf das Skalarprodukt zu sprechen. Eine wichtige Eigenschaft des Tensorproduktes ist, dass es mit dem Skalarprodukt verträglich ist.

Für Vektoren  $|\phi\rangle, |\phi'\rangle$  derselben Länge  $m$  und Vektoren  $|\psi\rangle, |\psi'\rangle$  derselben Länge  $n$  gilt:

$$\langle \phi \otimes \psi | \phi' \otimes \psi' \rangle = \langle \phi | \phi' \rangle \cdot \langle \psi | \psi' \rangle.$$

Tensor- und  
Skalarprodukt

## 2.8 Das Messen von Quantenregistern

Ein Quantenzustand ist nur abgeschottet stabil. Wollen wir als Mitglieder der Außenwelt eines Quantenbits etwas über dessen Zustand herausfinden, müssen wir es messen. Eine Messung ist ein Prozess, der uns Zugang zu einem Quantenbit verschafft, wenn wir auch nicht den genauen Zustand ermitteln können.

Bisher haben wir nur einen Spezialfall solcher Prozesse kennen gelernt: eine Messung resultierte für uns stets in einem klassischen Zustand. Aber das ist nicht die ganze Wahrheit, man kann auch bezüglich bestimmter überlagerter Zustände messen. Außerdem muss keineswegs immer das ganze Register gemessen werden, wir können auch einzelne Bits abfragen.

Messbasis

Kurzgefasst lautet unsere bisherige Auffassung: ein Zugriff auf ein Quantenregister zerstört die Superposition, und ein klassischer Zustand wird angenommen und beobachtet. Dies ist nur eine Möglichkeit unter vielen, man kann bezüglich *verschiedener Basen* messen. Eine Voraussetzung muss so eine Basis erfüllen: sie muss orthogonal sein, alle Elemente stehen senkrecht aufeinander. Und da alle unsere Zustandsvektoren die Länge 1 haben, betrachten wir letztlich *Orthonormalbasen*.

Beschränken wir uns zunächst auf ein Bit. In Abbildung 2.16 sehen wir den Zustandsvektor eines Quantenbits, die Koordinatenachsen sind mit  $|0\rangle$  und  $|1\rangle$  beschriftet.  $\alpha$ , der  $|0\rangle$ -Anteil oder – mathematisch ausgedrückt – die Länge der *Projektion* des Zustandsvektors auf die  $|0\rangle$ -Achse, bestimmt die Wahrscheinlichkeit, dass wir  $|0\rangle$  beobachten. Ebenso verhält es sich mit dem  $|1\rangle$ -Anteil. Wir können jedoch ein anderes Koordinatensystem zu Grunde legen. In Abbildung 2.17 sind die neuen Koordinatenachsen Richtung

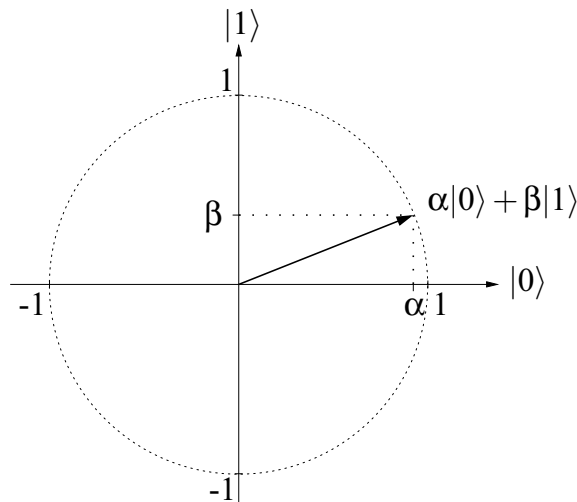


Abbildung 2.16: Zustandsvektor in der Standardbasis  $|0\rangle, |1\rangle$



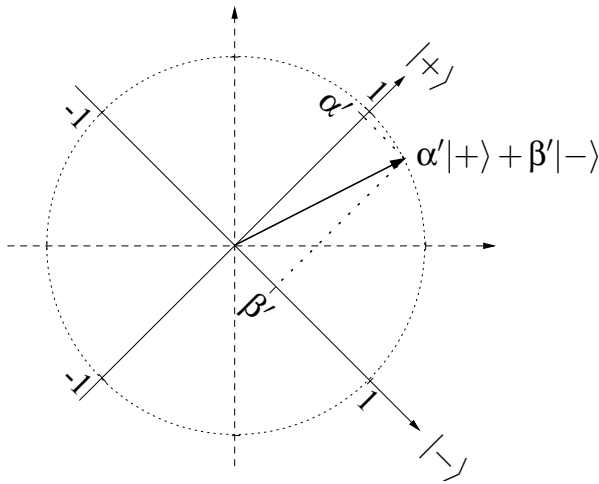


Abbildung 2.17: Basiswechsel

$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  und  $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  ausgerichtet. Der Zustandsvektor ändert sich nicht, aber die Projektionen auf die Koordinatenachsen haben sich geändert.

**Aufgabe 2.17:** Zeigen Sie, dass die Basis  $|+\rangle, |-\rangle$  orthogonal ist.

$\{|0\rangle, |1\rangle\}$  heißt *Standardbasis* und  $\{|+\rangle, |-\rangle\}$  *Hadamard-Basis*, da die Elemente durch die Hadamard-Transformation aus der Standardbasis entstehen. Messen wir nun bezüglich der neuen Basis, bestimmen die Projektionen  $\alpha', \beta'$  auf die neuen Koordinatenachsen das Ergebnis.

Messen wir  $\alpha'|+\rangle + \beta'|-\rangle$  bezüglich der Hadamard-Basis, so beobachten wir  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  mit der Wahrscheinlichkeit  $|\alpha'|^2$  und  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  mit Wahrscheinlichkeit  $|\beta'|^2$ , siehe Abbildung 2.17.

Weiterhin können wir nicht die Superposition selbst ermitteln; beim Messen wird diese zerstört, und einer der beiden Werte der zu Grunde gelegten Basis wird angenommen. Man kann die Werte  $\alpha', \beta'$  durch eine einfache Umrechnung in das neue Bezugssystem bestimmen; durch *Basistransformation*. Dazu lösen wir folgende Gleichung:

Basiswechsel

$$\alpha|0\rangle + \beta|1\rangle = \alpha' \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \beta' \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

**Aufgabe 2.18:** Zeigen Sie, dass  $\alpha' = \frac{1}{\sqrt{2}}(\alpha + \beta)$  und  $\beta' = \frac{1}{\sqrt{2}}(\alpha - \beta)$  gilt.

Man kann diese Basistransformation auch durch folgende *Übergangsmatrix* beschreiben: in der  $j$ -ten Spalte steht der Koordinatenvektor des alten Basisvektors  $|j\rangle$  bezüglich der neuen Basis. In unserem Fall ist die Übergangsmatrix gerade die Hadamard-Matrix  $H$ . Denn es gilt

$$|0\rangle = \frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) \text{ und } |1\rangle = \frac{1}{\sqrt{2}}(|+\rangle - |-\rangle).$$

Das alles gilt genauso für Register mit beliebig vielen Bits. Wir fassen unsere Überlegungen in folgender Definition zusammen:

Definition  
Messen

Register  $R$  bestehe aus  $n$  Quantenbits und befinde sich im Zustand

$$|\phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle.$$

Wir messen bezüglich einer Basis

$$|0'\rangle, |1'\rangle, \dots, |(2^n - 1)'\rangle$$

aus zueinander orthogonalen Vektoren der Länge 1. Dabei wird die Superposition von  $|\phi\rangle$  zerstört. Hat  $|\phi\rangle$  bezüglich der Messbasis die Darstellung

$$\sum_{i=0}^{2^n-1} \alpha'_i |i'\rangle,$$

so finden wir das Register nach der Messung mit Wahrscheinlichkeit  $|\alpha'_i|^2$  im Zustand  $|i'\rangle$  vor. Sämtliche anderen Informationen gehen dabei verloren.

### Messen eines einzelnen Bits eines Registers

Es ist auch möglich, nur eines der Bits eines Quantenregisters zu messen. Wie leitet man aus dem Zustand des Registers das Ergebnis dieser teilweisen Messung ab und wie ändert sich der Zustand des Registers? Um diese Fragen zu beantworten, konzentrieren wir uns erneut auf zwei Bits  $|xy\rangle$ , diese sind im Zustand

$$|\phi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle.$$

Wir messen das erste Bit: das Ergebnis ist  $|0\rangle$  oder  $|1\rangle$ . Im ersten Fall,  $|x\rangle = |0\rangle$ , geht das Register in eine Superposition von  $|00\rangle$  und  $|01\rangle$  über. Die Wahrscheinlichkeit für dieses Ergebnis berechnet sich aus den beteiligten Amplituden: sie beträgt  $|\alpha_{00}|^2 + |\alpha_{01}|^2$ . Wie genau sieht der Folgezustand aus? In ihm bleibt das Verhältnis der Amplituden  $\alpha_{00}$  und  $\alpha_{01}$  gleich. Um einen zulässigen Quantenzustand zu erreichen, in dem die Summe der Quadrate des Amplitudenbetrags gleich 1 ist, *normalisieren* wir den Zustand.

Messen wir im obigen Zustand  $|\phi\rangle$  das erste Bit, so geht das Register mit Wahrscheinlichkeit

$$|\alpha_{00}|^2 + |\alpha_{01}|^2$$

in den Zustand

$$|\phi'\rangle = \frac{\alpha_{00}|00\rangle + \alpha_{01}|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}$$

über und mit Wahrscheinlichkeit

$$|\alpha_{10}|^2 + |\alpha_{11}|^2$$

in den Zustand

$$|\phi'\rangle = \frac{\alpha_{10}|10\rangle + \alpha_{11}|11\rangle}{\sqrt{|\alpha_{10}|^2 + |\alpha_{11}|^2}}.$$

Wir erfahren bei der Messung nur den Wert des gemessenen Bits, nicht die Amplituden des Folgezustandes.

Messen eines Bits

### Messen als Projektion

Der Rest dieses Abschnitts ist etwas abstrakter. Er enthält eine allgemeinere Beschreibung davon, was für Messungen an Quantenregistern möglich sind. Die unseren Messungen zu Grunde liegende Mathematik ist *Projektion auf einen Unterraum*, siehe dazu Abschnitt A.2.4 im Anhang. Dort wird auch der von Vektoren  $b_1, \dots, b_m$  aufgespannte Vektorraum  $\text{Span}\{b_1, \dots, b_m\}$  eingeführt.

Betrachten wir dazu noch einmal die Abbildungen 2.16 und 2.17. Der Zustandsvektor wird in der ersten Abbildung auf die Unterräume  $\text{Span}\{|0\rangle\}$  und  $\text{Span}\{|1\rangle\}$  projiziert und in der anderen auf die Unterräume  $\text{Span}\{|+\rangle\}$  und  $\text{Span}\{|-\rangle\}$ . Das Ergebnis dieser Projektionen sind die Vektoren

$$\alpha|0\rangle, \beta|1\rangle, \alpha'|+\rangle \text{ und } \beta'|-\rangle.$$

Messen wir in einem Zwei-Bit-Register beide Bits, bestimmen die Projektionen auf die eindimensionalen Unterräume

$$\text{Span}\{|00\rangle\}, \text{Span}\{|01\rangle\}, \text{Span}\{|10\rangle\} \text{ und } \text{Span}\{|11\rangle\}$$

das Ergebnis. Messen wir in dem Zwei-Bit-Zustand  $|\phi\rangle$  das erste Bit, so sind die Projektionen auf

$$\text{Span}\{|00\rangle, |01\rangle\} \text{ und } \text{Span}\{|10\rangle, |11\rangle\}$$

maßgeblich. Das sind gerade  $\alpha_{00}|00\rangle + \alpha_{01}|01\rangle$  und  $\alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ . Messen wir das zweite Bit, projizieren wir auf

$$\text{Span}\{|00\rangle, |10\rangle\} \text{ und } \text{Span}\{|01\rangle, |11\rangle\}.$$

### Definition Observable

Wir verwenden diese Überlegungen zur Messung als Projektion für eine allgemeinere Definition. Grundsätzlich gilt: Wollen wir etwas über einen Quantenzustand herausbekommen, müssen wir ihn messen. Eine *Observable* beschreibt die Art, wie gemessen wird.

Sei  $e_0, \dots, e_{n-1}$  eine Orthonormalbasis des Zustandsraums eines Quantenregisters. Sei  $E_1, \dots, E_k$  eine disjunkte Zerlegung dieser Basis, das heißt, jeder Basisvektor kommt in genau einer der Mengen  $E_1, \dots, E_k$  vor. Wir nennen die Folge von Unterräumen

$$\text{Span } E_1, \dots, \text{Span } E_k$$

eine *Observable*. Mit  $|\phi_1\rangle, \dots, |\phi_k\rangle$  bezeichnen wir die Projektionen von  $|\phi\rangle$  in die Unterräume der Observablen. Es gilt

$$|\phi\rangle = |\phi_1\rangle + \dots + |\phi_k\rangle.$$

Messen wir den Zustandsvektor  $\phi$  bezüglich dieser Observablen,

- geht das Register mit Wahrscheinlichkeit

$$\| |\phi_i\rangle \|^2$$

in den Zustand

$$\frac{|\phi_i\rangle}{\| |\phi_i\rangle \|}$$

über. Wir können mit diesem Zustand weiterrechnen und

- wir erfahren den Index  $i$ .

$\| |\phi\rangle \|$  bezeichnet dabei die Länge oder *Norm* des Zustandsvektors  $|\phi\rangle$ , siehe Anhang. Für  $|\phi\rangle = (\alpha_0 \dots \alpha_{n-1})^T$  ist  $\| |\phi\rangle \| = \sqrt{|\alpha_0|^2 + \dots + |\alpha_{n-1}|^2}$ .

**Aufgabe 2.19:** Man kann unsere bisherige Art zu messen mit obigem Begriff der Observablen in Einklang bringen. Tun Sie das! Und zwar für die normale Messung eines Registers bezüglich der Standardbasis und für die Messung eines Bits eines Zwei-Bit-Registers.

### Ausblick

In der Folge werden wir mit der Messung in der Standardbasis oder einer anderen Orthonormalbasis auskommen, manchmal auf einzelne Bits eingeschränkt. In der weiterführenden Literatur wird Ihnen vielleicht einmal die *Observable* als hermitescher Operator begegnen. Im Falle von Messungen an Qubits besteht folgender Zusammenhang: Die Eigenvektoren hermitescher Operatoren bilden eine Orthonormalbasis wie oben.

### Ergebnis der Messung

Nun ist es an der Zeit, unser ursprüngliches Bild von der Messung etwas zu korrigieren. Um nicht zu viel auf einmal verstehen zu müssen, war unsere Ausgangsdefinition des Messens: Messen wir ein Quantenbit im Zustand  $\alpha|0\rangle + \beta|1\rangle$ , wird die Superposition zerstört und wir beobachten  $|0\rangle$  oder  $|1\rangle$ . An dieser Auffassung nehmen wir nun folgende Korrekturen vor.

1. Die Superposition wird in gewissem Sinne tatsächlich zerstört, weil eine Messung nicht rückgängig gemacht werden kann. Aber das Ergebnis kann durchaus wieder eine Superposition in unserem bisherigen Sinne sein; etwa wenn wir bezüglich der Basis  $\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$  messen.
2. Damit hat eine Messung im Grunde zwei Ergebnisse. Einen Quanten-Folgezustand mit dem wir weiterrechnen können, der aber ebenso gefährdet und unberührbar ist wie alle anderen Quantenzustände. Und klassische Informationen über das Ergebnis der Messung: Nummerieren wir die Unterräume durch, zeigt uns die Messapparatur die Nummer des angenommen Unterraumes an.

Ergab bisher eine Messung von  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  bezüglich der Standardbasis etwa das Ergebnis  $|0\rangle$ , bedeutet dies im Grunde zweierlei: Das Quantenbit ist in den Quantenzustand  $|0\rangle$  übergegangen, und wir werden darüber informiert. Der Hinweis auf diese zwei Aspekte mag subtil erscheinen, hat aber einen Grund. Messen wir einen unbekannten Zustand bezüglich einer Observablen, erfahren wir, in welchen Unterraum der Zustand projiziert wurde, der genaue Folgezustand aber bleibt uns weiterhin verborgen. Zur Veranschaulichung kann die Messung eines einzelnen Bits eines Quantenregisters dienen, siehe Seite 46.

Den genauen Zustand eines Quantensystems können wir jedoch nicht ermitteln. Stellen wir uns vor, wir öffnen eine Kiste und finden eine tote Katze. Das Tier kann sich vor dem Öffnen in jeder Superposition

Einschränkungen

$$|\phi_{Katze}\rangle = \alpha \cdot |\text{tot}\rangle + \beta \cdot |\text{lebendig}\rangle, \alpha \neq 0$$

befunden haben. Vielleicht war sie bereits ausschließlich tot ( $\alpha = 1, \beta = 0$ ), vielleicht auch nicht.

Messen wir bezüglich einer anderen Basis bleibt diese Einschränkung erhalten. Wir messen zum Beispiel eine tote Katze

$$|\phi_{Katze}\rangle = |\text{tot}\rangle$$

bezüglich der Basis  $B = \{|b_1\rangle, |b_2\rangle\}$  mit

$$|b_1\rangle = \frac{1}{\sqrt{2}}(|\text{tot}\rangle + |\text{lebendig}\rangle), |b_2\rangle = \frac{1}{\sqrt{2}}(|\text{tot}\rangle - |\text{lebendig}\rangle).$$

Dann wird unsere Messung die Katze in einen Zombie verwandeln. Denn

$$|\text{tot}\rangle = \frac{1}{\sqrt{2}}(|b_1\rangle + |b_2\rangle).$$

Die Katze ist nach der Messung im Zustand  $|b_1\rangle$  oder  $|b_2\rangle$ .

## 2.9 Noch einmal das Problem von Deutsch

Unsere neue Auffassung des Messens soll gleich erprobt werden, und zwar an einer neuen Version des Algorithmus für das Problem von Deutsch. Die Berechnung aus Abschnitt 2.6 wird abgewandelt, die verkürzte Analyse blickt aus einem anderen Winkel auf den Algorithmus.

Problemstellung

Uns steht ein Quantenorakel  $U_f$  für eine Funktion  $f : \{0,1\} \rightarrow \{0,1\}$  zur Verfügung. Das heißt, wir können die Funktion auf ein Quantenbit anwenden. Dieses Orakel hat die Form

$$U_f : |x,y\rangle \mapsto |x,y \oplus f(x)\rangle.$$

Wir sollen herausfinden, ob diese Funktion konstant ist – dann gilt  $f(0) = f(1)$  – oder ob sie balanciert ist:  $f(0) \neq f(1)$ .

### Der neue Algorithmus für das Problem von Deutsch

1.  $R = |x,y\rangle \leftarrow |01\rangle$
2. Wende die Hadamard-Transformation  $H_2 = H \otimes H$  an:  
 $R \leftarrow H_2 R$
3. Wende  $f$  aus:  
 $R \leftarrow U_f R$
4. Miss das erste Bit  $|x\rangle$  bezüglich der Basis  $\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$ :  
Ist das Ergebnis  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ : Ausgabe *konstant*,  
Ist das Ergebnis  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ : Ausgabe *balanciert*.

Diesem Algorithmus entspricht der Schaltkreis in Abbildung 2.18. Das mit  $M'$  markierte Gatter ist eine Messung in der Hadamard-Basis.

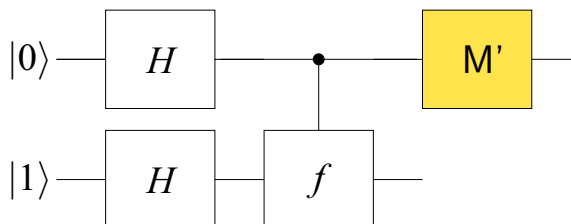


Abbildung 2.18: Schaltkreis für das Problem von Deutsch:  $M'$  ist eine Messung in der Hadamard-Basis

### Analyse

Wie in dem Algorithmus aus Abschnitt 2.6 versetzen die ersten beiden Schritte das Register  $|x\rangle|y\rangle$  in den Zustand

$$|\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Um die Wirkung von  $U_f$  zu untersuchen, lassen wir  $|x\rangle$  zunächst einmal unbestimmt.

$$\begin{aligned} |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) &\xrightarrow{U_f} |x\rangle \frac{1}{\sqrt{2}}(|f(x)\rangle - |1 \oplus f(x)\rangle) \\ &= |x\rangle (-1)^{f(x)} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned}$$

Das ist der Kern dieses Schrittes:  $f(x)$  wird in das Vorzeichen verlagert.

Nun ist  $|x\rangle$  allerdings im Zustand  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ , und das Ergebnis von Schritt 2 ist

$$\begin{aligned} |\phi_3\rangle &= \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= \begin{cases} \pm \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot |y\rangle & \text{für } f(0) = f(1), \\ \pm \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \cdot |y\rangle & \text{für } f(0) \neq f(1). \end{cases} \end{aligned}$$

*Bemerkung:*

1. Das Ergebnis ist dasselbe, egal ob wir bezüglich der Hadamard-Basis  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ ,  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  messen oder die Hadamard-Transformation ausführen und bezüglich der Standardbasis  $|0\rangle, |1\rangle$  messen.
2. Messen wir bezüglich der falschen Basis (oder transformieren nicht in die passende Basis), liefert uns die Messung nicht die gewünschten Informationen. Ob ein Quantenbit im Zustand  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  oder  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  ist, macht für eine Messung in der Standardbasis keinen Unterschied.

## 2.10 Bestandsaufnahme: Die drei Prinzipien des Quantum Computing

Mit den zurückliegenden neun Abschnitten sind die Grundlagen für das Verständnis von Quantenberechnungen gelegt. Wir wissen, welche Zustände ein Quantenregister annehmen kann, was für Rechenschritte ausgeführt werden können, und wir haben die zentrale Rolle des Messens verstanden. Nun können wir von der Pflicht zur Kür übergehen. Zuvor fassen wir alles zusammen: in den *drei Prinzipien des Quantum Computing*.

**Registerzustände** *Prinzip 1.* Ein Quantenregister aus  $n$  Quantenbits  $R = |x_{n-1} \dots x_1 x_0\rangle$  wird durch einen  $2^n$ -dimensionalen Vektorraum über den komplexen Zahlen beschrieben. Die Standardbasis bildet die Menge der durch  $n$  klassische Bits darstellbaren Zahlen

$$|0\rangle, |1\rangle, \dots, |2^n - 1\rangle$$

oder die entsprechenden Binärdarstellungen

$$\{|0 \dots 00\rangle, |0 \dots 01\rangle, \dots, |1 \dots 11\rangle\} = \{|i\rangle \mid i \in \{0, 1\}^n\}.$$

Ein Zustand eines Quantenregisters  $R$  ist ein Vektor

$$\begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{pmatrix}$$

der Länge 1. Diesem Vektor entspricht die Superposition oder Überlagerung von Basiszuständen

$$R = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$$

mit der Eigenschaft:

$$\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1.$$

**Rechenschritte** *Prinzip 2.* Mögliche Veränderungen an Quantenregistern werden durch unitäre Transformationen beschrieben. Dadurch ist jeder Rechenschritt auf einem Quantencomputer umkehrbar. Die möglichen Rechenschritte sind lokal: sie lassen sich in unitäre Operationen zerlegen, an denen höchstens zwei der Bits beteiligt sind. Die Zerlegung geschieht gemäß dem Tensorprodukt.

**Messen** *Prinzip 3.* Misst man ein Quantenregister im Zustand

$$R = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$$

bezüglich der Standardbasis

$$|0\rangle, |1\rangle, \dots, |2^n - 1\rangle,$$

erhält man mit Wahrscheinlichkeit  $|\alpha_i|^2$  das Ergebnis  $|i\rangle$ . Misst man bezüglich einer Orthonormalbasis

$$|b_0\rangle, |b_1\rangle, \dots, |b_{2^n-1}\rangle$$

und hat der Registerzustand gemäß dieser Basis die Form

$$R = \sum_{i=0}^{2^n-1} \alpha'_i |b_i\rangle,$$

so erhält man mit Wahrscheinlichkeit  $|\alpha'_i|^2$  das Ergebnis  $|b_i\rangle$ .



Messen bezüglich einer *Observablen* wird am Ende von Abschnitt 2.8 eingeführt. Dieses Konzept beschreibt, wie bezüglich einzelner Bits und nach noch allgemeineren Zerlegungen gemessen werden kann. Eine Messung hat zwei Resultate:

1. wir erhalten das Ergebnis der Messung in Form klassischer Information;
2. die gemessenen Quantenbits nehmen einen Folgezustand an, mit dem wir weiterrechnen können.

Im nächsten Abschnitt werden wir eine weitere Eigenschaft von Quantensystemen kennen lernen, die für unsere Berechnungen sehr wichtig sind. Das Stichwort lautet *Verschränkung*. Wir brauchen die Liste dieses Abschnitts jedoch nicht zu erweitern: die Verschränkung folgt aus den drei Prinzipien.

## 2.11 Verschränkung

Eine der merkwürdigsten Eigenschaften von Quantenregistern ist, dass einzelne Bits *verschränkt* sein können. Manipulationen eines Bits können den Ausgang der Messung eines anderen Bits beeinflussen; darauf beruht zum Beispiel die Quantenteleportation, wie wir in einem späteren Kapitel sehen werden. Verschränkung, der Begriff wurde erstmals 1937 von Schrödinger verwendet, gilt als das eigentliche Charakteristikum der Quantenwelt, und in der Anfangszeit der Quantenmechanik war sie Gegenstand erregter Debatten. Es handelt sich dabei um ein Phänomen, das schwer mit unserer alltäglichen Auffassung von Realität in Übereinstimmung zu bringen ist, aber mathematisch sehr leicht aus dem uns schon Bekannten folgt. Darum verschieben wir die Diskussion über das Wesen der Verschränkung (engl. *entanglement*) auf das Ende des Abschnitts.

Wir betrachten ein Zwei-Bit-Register  $|b_1 b_2\rangle$  im Zustand  $|00\rangle$ , wenden auf das erste Bit die Hadamard-Transformation an und anschließend auf beide Bits die Operation CNOT aus Beispiel 2.5 (siehe Abbildung 2.19):

Erzeugung eines verschränkten Zustands

$$\text{CNOT} : |x, y\rangle \rightarrow |x, y \oplus x\rangle.$$

Das ergibt:

$$\begin{aligned} |00\rangle &\xrightarrow{H \otimes I_2} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) \\ &\xrightarrow{\text{CNOT}} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \end{aligned}$$

Messen wir nun das erste Bit, ist das Ergebnis dieser Messung  $|0\rangle$  oder  $|1\rangle$ , und zwar jeweils mit Wahrscheinlichkeit  $1/2$ . Im ersten Fall ist der Folgezustand  $|00\rangle$ . Haben wir für das erste Bit  $|1\rangle$  beobachtet, geht das Zwei-Bit-Quantenregister in den Zustand  $|11\rangle$  über. Messen wir danach das zweite Bit, erhalten wir dasselbe Ergebnis wie für das erste Bit: entweder sind am Ende beide Bits 0 oder beide 1.

Gekoppelte Bits

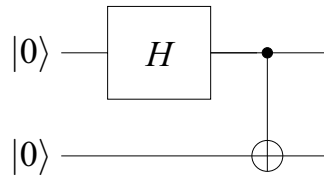


Abbildung 2.19: Schaltkreis zur Verschränkung zweier Bits

Bevor  $|b_1\rangle$  gemessen wurde, war der Ausgang einer Messung an  $|b_2\rangle$  offen: beide Ergebnisse waren gleich wahrscheinlich. Ist  $|b_1\rangle$  hingegen bereits gemessen worden, steht das Ergebnis der Messung an  $|b_2\rangle$  fest. Das gleiche Phänomen tritt auf, wenn wir zuerst  $|b_2\rangle$  messen.

#### Fernwirkung

Dass sich der Zustand  $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  verhält wie eben beschrieben, folgt aus unseren Voraussetzungen; es lässt sich dem Zustand direkt ansehen. Um noch deutlicher heraus zu arbeiten, wie erstaunlich das Verhalten verschränkter Quantenbits ist, trennen wir die Bits und geben sie zwei verschiedenen Personen. Wie in der Literatur zum Thema üblich, nennen wir die beiden fiktiven Personen Alice und Bob. Gemeinsam erzeugen sie wie oben beschrieben zwei verschränkte Quantenbits. Alice erhält das erste und Bob das zweite Bit. Dieser geht damit ins Nebenzimmer; wenn sein Quantenbit gut von der Außenwelt abgeschirmt ist – so wie Schrödingers Katze durch die Kiste – ändert sich der Zustand beider Bits nicht; er ist weiterhin

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

Alice und Bob messen die mittlerweile räumlich getrennten Bits. Für beide gilt: mit Wahrscheinlichkeit  $1/2$  ist das Ergebnis  $|0\rangle$  und mit der gleichen Wahrscheinlichkeit  $|1\rangle$ . Keiner kann feststellen, ob der andere sein Bit bereits gemessen hat. Egal ob Bob zuerst misst, oder ob Alice zuerst misst: jeder für sich beobachtet  $|0\rangle$  oder  $|1\rangle$  mit gleicher Wahrscheinlichkeit. Für jede der beiden Messungen ist das Ergebnis unbestimmt.

Wenn Alice und Bob sich jedoch anschließend treffen und über ihre Messungen sprechen, werden sie feststellen: *ihre Ergebnisse sind identisch*. Die Bits sind vollständig aneinander gekoppelt, trotz räumlicher Distanz; dazu gibt es in der klassischen Welt kein Analogon. Es spielt dabei auch keine Rolle, wer zuerst misst oder ob beide gleichzeitig messen (wobei diese Begriffe durch die Relativitätstheorie ohnehin an Absolutheit verloren haben).

#### Bell-Zustände

Alice und Bob haben einen *Bell-Zustand* erzeugt, benannt nach dem in Belfast geborenen Physiker John Bell (1928-1990). Insgesamt gibt es vier solche Bell-Zustände:

$$\begin{aligned}
|\Phi^+\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \\
|\Phi^-\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \\
|\Psi^+\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \\
|\Psi^-\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).
\end{aligned}$$

Diese bilden zusammen eine Orthonormalbasis des Zustandsraumes der zwei Bits, die sogenannte *Bell-Basis*.

**Aufgabe 2.20:** Rechnen Sie nach: Wendet man wie in Abbildung 2.19 auf das erste Bit  $H$  an und anschließend CNOT, so wird die Standardbasis  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$  auf die Bell-Basis abgebildet.

Man sieht leicht, dass in  $|\Phi^-\rangle, |\Psi^+\rangle$  und  $|\Psi^-\rangle$  die beiden Bits auf dieselbe Weise gekoppelt sind wie in  $|\Phi^+\rangle$ . Der einzige Unterschied: Bei  $|\Psi^+\rangle$  und  $|\Psi^-\rangle$  erhalten Alice und Bob grundsätzlich entgegengesetzte Ergebnisse. Man bezeichnet zwei Bits im Zustand  $|\Phi^+\rangle$  oder  $|\Psi^+\rangle$  auch als *EPR-Paar*, benannt nach Einstein, Podolsky und Rosen, siehe unten. In der obigen Situation teilen sich Alice und Bob ein EPR-Paar.

Wie lässt sich diese Kopplung mathematisch beschreiben? Dazu unterscheiden wir verschränkte Zustände von unverschränkten.

$$\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$

und

$$H_2|11\rangle = \frac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle)$$

sind Beispiele für Superpositionen, in denen die Bits nicht gekoppelt sind. Das wird aus Messungen an den einzelnen Bits ersichtlich. Wir definieren:

Sei  $|\phi\rangle$  der Zustand eines Quantenregisters aus  $n$  Bits. Der Zustand  $|\phi\rangle$  heißt *unverschränkt*, wenn er das Produkt von Zuständen der einzelnen Bits ist:

$$|\phi\rangle = |\phi_{n-1}\rangle \otimes |\phi_{n-2}\rangle \otimes \dots \otimes |\phi_0\rangle.$$

Ein Zustand heißt *verschränkt*, wenn es keine solche Zerlegung gibt.

Definition  
Verschränkung

**Beispiel 2.11:** Wir zerlegen die beiden vor der Definition erwähnten unverschränkten Zustände. Es gilt

$$\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle,$$

(bisher haben wir der schwerfälligen Notation  $\otimes$  meistens das gewöhnliche Produktzeichen  $\cdot$  vorgezogen). Für den zweiten der oben genannten Zustände gilt:

$$\begin{aligned} \frac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle) &= \frac{1}{2}(|0\rangle \otimes (|0\rangle - |1\rangle) - |1\rangle \otimes (|0\rangle - |1\rangle)) \\ &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned}$$

**Aufgabe 2.21:** Zeigen Sie, dass  $|\Phi^+\rangle$  nicht in das Produkt zweier Zustände jeweils eines Bits zerlegt werden kann.

Verschränkung  
von mehr als  
zwei Bits

Um  $|\Phi^+\rangle$  zu erzeugen, haben wir CNOT auf den unverschränkten Zustand  $1/\sqrt{2}(|00\rangle + |10\rangle)$  angewendet und so einen verschränkten Zustand erhalten. Nun stimmt CNOT gerade mit der Ein-Bit-Operation  $U_f$  zum Orakel  $f(0) = 0, f(1) = 1$  überein. Ähnlich lassen sich verschränkte Zustände mehrerer Bits erzeugen.

**Beispiel 2.12:** Wir rechnen mit zwei Registern  $|x\rangle$  und  $|y\rangle$  aus jeweils  $n$  Quantenbits. Wir initialisieren alle Bits mit dem Zustand  $|0\rangle$ . Wenden wir auf das erste Register die Hadamard-Transformation an, ändert sich der Zustand zu

$$\left( \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \right) |0 \dots 0\rangle$$

über. Wir wenden die Operation

$$U_f : \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}, |x, y\rangle \rightarrow |x, y \oplus f(x)\rangle$$

für eine beliebige Funktion  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  an ( $\oplus$  meint hier die komponentenweise Addition in  $\{0, 1\}^n$ ; es ist  $(x_{n-1}, \dots, x_0)^T \oplus (y_{n-1}, \dots, y_0)^T = (x_{n-1} \oplus y_{n-1}, \dots, x_0 \oplus y_0)^T$ ). Das Register befindet sich anschließend im Zustand

$$\frac{1}{\sqrt{2^n}} \left( \sum_{i=0}^{2^n-1} |i\rangle |f(i)\rangle \right).$$

Messen wir nun das erste Register, beobachten wir irgendeinen Wert  $|i\rangle$ . Gleichzeitig nimmt das zweite Register den Wert  $|f(i)\rangle$  an.  $\diamond$

Unverschränkte Zustände lassen sich in Produkte von Zuständen von je *einem* Bit zerlegen. Folgende Übung behandelt die Zerlegung eines verschränkten Zustands in Anteile von je zwei Bits:

**Aufgabe 2.22:** Zerlegen Sie

$$1/2(|0\rangle + |3\rangle + |12\rangle + |15\rangle)$$

in ein Produkt von verschränkten Zwei-Bit-Zuständen.

Bisher haben wir verschränkte und unverschränkte Zustände kennengelernt. Es gibt auch solche, die nach unserer Definition verschränkt sind, deren Bits aber weniger stark gekoppelt sind als in den Bell-Zuständen. Messungen an einem Zwei-Bit-Register im Zustand

Grad der  
Verschränkung

$$|\phi\rangle = \frac{1}{2}|00\rangle + \frac{\sqrt{3}}{2}|11\rangle$$

liefern für beide Bits stets dasselbe Ergebnis, aber mit unterschiedlichen Wahrscheinlichkeiten. Mit Wahrscheinlichkeit  $1/4$  erhalten wir  $|00\rangle$  und mit Wahrscheinlichkeit  $3/4$  das Ergebnis  $|11\rangle$ . Der Einfluss der ersten Messung ist geringer: der Ausgang der zweiten Messung ist von vornherein weniger unsicher. Wir betrachten zur Verdeutlichung ein Zwei-Bit-Register im Zustand

$$|11\rangle.$$

Auch hier liefern getrennte Messungen stets dasselbe Ergebnis. Aber für beide Messungen steht das Ergebnis von vornherein fest,  $|11\rangle$  ist ein unverschränkter Zustand. Und da für  $|\phi\rangle$  der Ausgang der Messungen weniger unsicher ist als für  $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ , ist  $|\phi\rangle$  weniger verschränkt als  $|\Phi^+\rangle$ . Die Bell-Zustände sind *maximal verschränkt*.

Wir verzichten darauf, das Maß an Verschränkung formal zu definieren. Wir begnügen uns mit der Feststellung, dass ein Zustand der Form

$$\frac{1}{\sqrt{k}}|00\rangle + \frac{\sqrt{k-1}}{\sqrt{k}}|11\rangle, k \geq 2$$

für wachsendes  $k$  immer weniger verschränkt ist.

Was bedeutet das Phänomen „Verschränkung“ für unser Weltbild? Eine knapp gehaltene Diskussion dieser Frage soll diesen Abschnitt beschließen. Dazu begeben wir uns in das Jahr 1935, als eine berühmte Arbeit von Albert Einstein, Boris Podolsky und Nathan Rosen erschien [52]. Die Bezeichnung EPR-Paar geht auf diese drei damals am Institute for Advanced Studies in Princeton arbeitenden Forscher zurück. Der Grund: sie schlugen ein Gedankenexperiment vor, das der oben beschriebenen Situation ähnelt, in der Alice und Bob ihre Bits in verschiedenen Räumen messen. Die Kopplung der beiden Bits, diese im Originalton *spukhafte Fernwirkung* oder *spooky action at a distance*, hielten sie für eine absurde Folge der Quantenmechanik. Das später so benannte *Einstein-Podolsky-Rosen-Paradoxon* sollte zeigen, dass die Quantenmechanik keine vollständige Beschreibung der Wirklichkeit liefert. Denn das Verhalten von räumlich getrennten verschränkten Bits scheint dem Lokalitätsprinzip zu widersprechen. Nicht nur die drei Autoren waren Anhänger dieses Prinzips, es scheint ein Grundsatz des gesunden Menschenverstandes zu sein.

Philosophisches

Wirken zwei Objekte aufeinander, müssen sie nach klassischer Auffassung in Verbindung stehen; das fordert das Lokalitätsprinzip. Eine rollende Billardkugel kann eine ruhende Billardkugel durch *direkten Kontakt* in Bewegung setzen. Die Wirkung zwischen zwei Objekten kann aber auch über ein *Medium* vermittelt werden, zum Beispiel über ein elektromagnetisches Feld. Auch

Lokalität

die Anziehungskraft zwischen zwei entfernten Körpern lässt sich so erklären. Die Schwerkraft wird über Gravitonen vermittelt, die im Einklang mit der Relativitätstheorie nicht schneller als das Licht sind.

Sind hingegen zwei Objekte auf eine Art getrennt, die keinen Informationsaustausch zwischen ihnen zulässt, so sind diese Objekte unabhängig und können sich nicht beeinflussen. Diese plausibel wirkenden Annahmen bilden das Fundament des *Lokalitätsprinzips*: jede Wirkung beruht auf direktem Kontakt oder wird durch ein Medium weitergeleitet, und zwar höchstens so schnell wie das Licht.

Wie verhält es sich nun mit den Messungen, die Alice und Bob an ihren verschränkten Quantenbits vorgenommen haben? Die erste Messung wird ausgeführt, und sofort steht das Ergebnis der zweiten Messung fest. Es kann kein Medium geben, über das Information ausgetauscht wird, denn keine lokale Wirkung ist schneller als das Licht. Auch der klassische Begriff *Kausalität* ist fehl am Platze. Man kann nur sagen: bei den Messungen äußert sich die Kopplung der Bits in identischen Ergebnissen; verschränkte Bits bilden räumlicher Trennung zum Trotz eine Einheit.

Realismus

Einstein, Podolsky und Rosen störte an der Quantenmechanik noch etwas anderes. Sie gingen davon aus, dass physikalische Eigenschaften von ihrer Beobachtung unabhängig seien. Diese Auffassung bezeichnet man in diesem Zusammenhang als *Realismus*. Der Zustand eines Quantensystems ist jedoch in der Regel vor der Messung unbestimmt und der Zufall spielt eine für einen Realisten ungebührliche Rolle.

Verborgene  
Variablen

Die drei Autoren folgerten: Da die Quantenmechanik keine lokal-realistische Theorie ist, und zum Beispiel die Existenz von nicht-lokalen Wirkungen nahe legt, kann mit ihr etwas nicht stimmen. Die quantenmechanische Beschreibung der Natur ist nicht vollständig, sie zeichnet nicht das ganze Bild. In der erwähnten Veröffentlichung schlugen sie die Existenz *verborgener Variablen* vor, die die Ergebnisse der beiden Messungen festlegt, bevor Alice und Bob die Bits trennen. Die Herleitung von korrelierten Messergebnissen aus verborgenen Variablen könnte man mit der Erklärung der Ähnlichkeit von Zwillingen aus dem gemeinsamen Erbgut vergleichen. Mit dem Unterschied, dass sich besagte Variablen nicht beobachten lassen, sie sind nunmal verborgen. Lässt sich mit Hilfe verborgener Variablen ein lokal-realistisches Weltbild aufrechterhalten oder sollten wir davon Abschied nehmen? Das scheint zunächst eine weltanschauliche oder sogar eine Geschmacksfrage zu sein, die experimentell nicht entscheidbar ist.

Bells Theorem

Überraschenderweise stimmt das nicht ganz. John Bell (siehe oben) war Mitarbeiter des europäischen Kernforschungszentrums CERN in Genf und beschäftigte sich unter anderem mit der Frage, wie eine im Sinne von Einstein, Podolsky und Rosen vollständige Beschreibung der Wirklichkeit aussehen könne. Eine solche Beschreibung sollte lokal realistisch sein und zugleich quantenmechanische Phänomene erklären können. Im Jahr 1964 bewies Bell rein theoretisch: laut Quantenmechanik ist die Kopplung verschränkter Teilchen stärker, als durch irgendeine lokal-realistische Theorie erklärbar wäre, auch wenn diese auf verborgene Variablen zurückgreift, [13]. Bell betrachtete dazu verschiedene Messungen an verschränkten Bits. Er konnte

zeigen, dass die Ergebnisse in jeder möglichen lokal-realistischen Theorie eine Ungleichung erfüllen müssen, welche jedoch nach den Voraussagen der Quantenmechanik verletzt würde. Also könnte experimentell überprüft werden, ob diese Ungleichung verletzt wird.

*Bellsche Ungleichung* ist heute der Begriff für eine ganze Reihe solcher Ungleichungen. So verallgemeinerten Clauser, Horne, Shimony und Holt Bells ursprüngliches Resultat zu der *CHSH-Ungleichung*, die experimentell leichter zu überprüfen ist. Die Ergebnisse vieler Experimente – das erste stammt aus dem Jahr 1972 ([62]) – stimmen mit den Voraussagen der Quantenmechanik überein: Bells Ungleichung gilt nicht und wir sollten uns gegen den lokalen Realismus entscheiden. Gezeigt ist damit, dass nicht Realismus *und* Lokalität zutreffen können. Wir müssen uns also von einem der beiden Prinzipien oder auch von beiden verabschieden.

Abgeschlossen ist diese an die Grundlagen der Physik und unser Weltbild rührende Debatte allerdings nicht, im Gegenteil ist die Interpretation der experimentellen Ergebnisse Gegenstand einer kontroversen Diskussion. So herrscht Uneinigkeit, ob die Lokalität, der Realismus oder beides aufgegeben werden sollten. Und bei den früheren experimentellen Überprüfungen, hätte die Verletzung von Bells Ungleichung im Prinzip durch *Lücken* – engl. *loopholes* – im experimentellen Aufbau erklärt werden können. Auch unausgesprochene Hintergrundannahmen von Bells Formalisierung des lokalen Realismus führten dazu, dass immer ausgefeiltere Experimente durchgeführt wurden. Berühmt ist ein 1982 von Aspect, Dalibard und Roger durchgeführter Nachweis der Verletzung der Voraussagen des lokalen Realismus, [9]. Von den jüngeren Arbeiten sollen [76] und [74] erwähnt werden, die als Einstieg in die aktuelle Diskussion dienen können.

Verschränkung ist eine der wesentlichen Eigenschaften von Quantensystemen. Formal folgt es aus den drei Prinzipien des letzten Abschnitts. Teleportation und Quantenkryptographie werden durch den Einsatz verschränkter Quantenbits möglich, und auch für andere Verfahren spielen diese eine Rolle. Die Diskussion dieses Themas wird in Abschnitt 5.3 fortgesetzt.

Zusammenfassung und Ausblick

## 2.12 Die Hadamard-Transformation und mehrere Bits

In Beispiel 2.9 haben wir die Hadamard-Transformation auf mehreren Bits kennen gelernt. Die meisten bekannten Algorithmen verwenden diese Operation oder eine Verallgemeinerung (die Quanten-Fouriertransformation, siehe Kapitel 8). Ziel des Abschnitts ist, etwas Fingerfertigkeit im Umgang mit dieser Transformation zu gewinnen.

Beginnen wir damit, die Definition zu wiederholen. Die Hadamard-Transformation auf einem Bit wird durch die Matrix

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

beschrieben. Man kann die Hadamard-Transformation auf  $n$  Bits anwenden, indem für jedes der Bits  $H$  ausgeführt wird. Mit Hilfe des Tensorproduktes ausgedrückt erhalten wir

$$H_n = \bigotimes_{i=1}^n H.$$

Wir untersuchen die Wirkung der  $n$ -Bit-Version.

Sei  $|x\rangle$  ein Register aus  $n$  Quantenbits  $|x_{n-1}\dots x_0\rangle$ . Belegungen des Registers drücken wir auf zwei verschiedene Weisen aus: Entweder als  $n$ -Tupel von 0en und 1en ( $x \in \{0,1\}^n$ ) oder als binär kodierte Zahl aus dem Bereich  $0, \dots, 2^n - 1$ . Wir wechseln je nach Bequemlichkeit zwischen diesen beiden Schreibweisen.

Wirkung auf  
zwei Bits

Ist ein Register aus zwei Bits im Zustand  $|x_1, x_0\rangle$  mit  $(x_1, x_0) \in \{0,1\}^2$ , so hat die Hadamard-Transformation  $H_2$  die folgende Wirkung:

$$|x_1, x_0\rangle \xrightarrow{H_2} \frac{1}{2}(|0\rangle + (-1)^{x_1}|1\rangle) \cdot (|0\rangle + (-1)^{x_0}|1\rangle)$$

Da  $(-1)^{x_0} \cdot (-1)^{x_1} = (-1)^{x_0 \oplus x_1}$  ist, ergibt Ausmultiplizieren

$$\begin{aligned} & \frac{1}{2}(|00\rangle + (-1)^{x_0}|01\rangle + (-1)^{x_1}|10\rangle + (-1)^{x_0 \oplus x_1}|11\rangle) \\ = & \frac{1}{2}((-1)^{(0,0) \cdot x}|00\rangle + (-1)^{(0,1) \cdot x}|01\rangle \\ & + (-1)^{(1,0) \cdot x}|10\rangle + (-1)^{(1,1) \cdot x}|11\rangle), \end{aligned}$$

mit  $x = (x_1, x_0)^T$ . Für zwei Vektoren  $x, y \in \{0,1\}^n$  bezeichnet  $x \cdot y$  das Skalarprodukt  $\bigoplus_{i=1}^n x_i y_i$ .  $\oplus$  ist die auf Seite 16 eingeführte Addition in  $\{0,1\}$ .

Wirkung auf  $n$   
Bits

Auf ein  $n$ -Bit-Register im Zustand  $x \in \{0,1\}^n$  hat die Hadamard-Transformation folgende Wirkung:

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

Es entsteht eine Superposition über alle klassischen Zustände des Registers, die Information über  $|x\rangle$  wird in das Vorzeichen verlagert.

**Beispiel 2.13:** Wir kennen bereits die Wirkung auf ein Register, dessen Bits alle den Wert 0 haben:

$$H_n|00\dots 0\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^0 |y\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} |y\rangle.$$

Das Ergebnis wird als gleichgewichtete Superposition bezeichnet.



Gehen wir von einem Register aus, in dem nur das Bit mit dem kleinsten Index den Wert 1 hat, erhalten wir

$$H_n |0\dots 01\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} (-1)^{y_0} |y\rangle$$

für  $y = (y_{n-1}, \dots, y_0)$ . Das ist die alternierende Superposition

$$\frac{1}{\sqrt{2^n}} \left( |0\dots 00\rangle - |0\dots 01\rangle + |0\dots 10\rangle - |0\dots 11\rangle \pm \dots + |1\dots 10\rangle - |1\dots 11\rangle \right).$$

◇

Die Hadamard-Transformation lässt sich auch als Basistransformation auffassen, wie in Abschnitt 2.8 beschrieben. Der von uns zuerst betrachtete Algorithmus für das Problem von Deutsch geht wie folgt vor: er führt auf einem Ausgangszustand des Zwei-Bit-Registers  $H_2$  aus, ruft dann das Orakel  $U_f$  auf, um vor dem Messen erneut die Transformation  $H_2$  anzuwenden.  $U_f$  wird also bezüglich der durch  $H_2$  beschriebenen Basis berechnet; danach wird zurücktransformiert.

Diese doppelte Ausführung der Hadamard-Transformation wird uns noch häufiger begegnen. Zur Vorbereitung des nächsten Abschnitts führen wir jetzt  $H_n$  auf einem Register beliebigen Inhalts zweimal hintereinander aus. Uns interessiert dabei, wie aus der in das Vorzeichen verlagerten Information der Ausgangszustand wiederhergestellt wird.

### Die Hadamard-Transformation zweimal ausgeführt

Unser Quantenregister aus  $n$  Bits sei anfangs im Zustand  $|x\rangle$ :

$$|x\rangle \xrightarrow{H_n} \frac{1}{\sqrt{2^n}} \sum_{z=0}^{2^n-1} (-1)^{x \cdot z} \cdot |z\rangle = |\phi_1\rangle.$$

Der Ausgangszustand  $x$  ist im Vorzeichen aufgehoben. Da die Hadamard-Transformation unitär ist, können wir diesen Schritt rückgängig machen, und zwar durch erneute Anwendung von  $H_n$ . Dabei geht jedes Element  $|z\rangle$  der Superposition in

$$H_n |z\rangle = \frac{1}{\sqrt{2^n}} \sum_{w=0}^{2^n-1} (-1)^{z \cdot w} |w\rangle$$

über. Wir erhalten also

$$\begin{aligned} |\phi_1\rangle &\xrightarrow{H_n} \frac{1}{\sqrt{2^n}} \sum_{z=0}^{2^n-1} (-1)^{x \cdot z} \left( \frac{1}{\sqrt{2^n}} \sum_{w=0}^{2^n-1} (-1)^{z \cdot w} |w\rangle \right) \\ &= \frac{1}{2^n} \sum_{z=0}^{2^n-1} \sum_{w=0}^{2^n-1} (-1)^{x \cdot z} (-1)^{z \cdot w} |w\rangle \\ &= |\phi_2\rangle. \end{aligned}$$

Wir vertauschen die Reihenfolge der Summation und nutzen die Linearität des Skalarproduktes:  $u \cdot (v \oplus w) = u \cdot v \oplus u \cdot w$ .

$$|\phi_2\rangle = \frac{1}{2^n} \sum_{w=0}^{2^n-1} \sum_{z=0}^{2^n-1} (-1)^{z \cdot (x \oplus w)} |w\rangle.$$

Für  $w = x$  ist  $z \cdot (x \oplus w) = 0$ ; wir zerlegen die Summe über  $w$ :

$$|\phi_2\rangle = \frac{1}{2^n} \left( \sum_{z=0}^{2^n-1} (-1)^0 \cdot |x\rangle + \sum_{\substack{w=0 \\ w \neq x}}^{2^n-1} \sum_{z=0}^{2^n-1} (-1)^{z \cdot (x \oplus w)} |w\rangle \right).$$

Wir untersuchen

$$\sum_{z=0}^{2^n-1} (-1)^{z \cdot (x \oplus w)}$$

für den Fall  $x \neq w$  und alle möglichen Werte von  $z$ . Dann ist  $x \oplus w$  nicht der Nullvektor. Wir nehmen an, in  $x \oplus w$  seien gerade die Bits  $b_1, \dots, b_m$  gleich 1. Es gilt:

$$z \cdot (x \oplus w) = \begin{cases} 0 & \text{wenn in } z \text{ eine gerade Anzahl} \\ & \text{der Bits } b_1, \dots, b_m \text{ den Wert 1 hat} \\ 1 & \text{sonst.} \end{cases}$$

Wir betrachten alle möglichen Werte von  $z$ : für die Hälfte ist die Anzahl der Bits  $b_1, \dots, b_m$  mit Wert 1 gerade, für die andere Hälfte ungerade. Damit gilt für alle  $w$ , die ungleich  $x$  sind:

$$\sum_{z=0}^{2^n-1} (-1)^{z \cdot (x \oplus w)} = 0,$$

und der Zustand nach der zweiten Hadamard-Transformation ist

$$|\phi_2\rangle = \frac{1}{2^n} \left( 2^n \cdot |x\rangle + \sum_{\substack{w=0 \\ w \neq x}}^{2^n-1} 0 \cdot |w\rangle \right) = |x\rangle.$$

Wir fassen zusammen:

$$|x\rangle \xrightarrow{H_n} \frac{1}{\sqrt{2^n}} \sum_{z=0}^{2^n-1} (-1)^{x \cdot z} \cdot |z\rangle \xrightarrow{H_n} |x\rangle.$$

## 2.13 Der Algorithmus von Deutsch-Jozsa

Im vorherigen Abschnitt wurde die Wirkungsweise der Hadamard-Transformation auf ein Register untersucht. Das hilft uns, eine Verallgemeinerung des Problems von Deutsch zu lösen.

Uns liegt ein Bauteil vor, das eine mathematische Funktion berechnet. Es bildet Eingaben aus  $n$  Bits auf ein Ausgabebit ab, das heißt eine Funktion  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  wird berechnet.

Aufgaben-  
stellung

Wir wissen über die berechnete Funktion  $f$  folgendes:

- Entweder ist  $f$  *konstant* und alle Eingaben werden auf die gleiche Ausgabe abgebildet,
- oder  $f$  ist *balanciert*: die Hälfte der Eingaben wird auf 1 abgebildet, die andere Hälfte auf 0. Dann gilt für die Anzahl der Urbilder  $|f^{-1}(0)| = |f^{-1}(1)| = 2^{n-1}$ .

Unsere Aufgabe: Wir sollen entscheiden, welche der Alternativen zutrifft.

Wir überlegen zunächst, für wie viele Eingaben ein klassischer Computer die Funktion auswerten muss. Auch wenn er geschickt vorgeht, kann folgende Situation eintreten: er hat bereits für  $v = 2^{n-1}$  verschiedene Eingaben  $b_1, \dots, b_v \in \{0, 1\}^n$  den Funktionswert bestimmt und herausbekommen, dass  $f(b_1) = \dots = f(b_v)$  gilt. Er weiß dann allerdings noch immer nicht, ob die Funktion balanciert oder konstant ist. Der klassische Rechner muss also im schlechtesten Fall für  $2^{n-1} + 1$  Eingaben den Funktionswert bestimmen. Die Anzahl der Aufrufe ist im schlechtesten Fall exponentiell in der Eingabegröße  $n$ .

Aufwand im klassischen Fall

Liegt uns die Funktion hingegen als Quantenbauteil

$$U_f : |x_{n-1} \dots x_0, y\rangle \mapsto |x_{n-1} \dots x_0, y \oplus f(x)\rangle$$

vor, kommt ein Quantencomputer mit nur einem Aufruf aus. Der folgende Algorithmus stammt von David Deutsch und Richard Jozsa von der Universität Bristol (siehe [46]).

### Der Algorithmus von Deutsch-Jozsa

Wir verwenden ein Register mit  $n + 1$  Quantenbits  $|x_{n-1} \dots x_0\rangle|y\rangle$ , siehe auch Abbildung 2.20.

1.  $|x_{n-1} \dots x_0\rangle|y\rangle \leftarrow |0 \dots 0\rangle|1\rangle$
2. Wende die Hadamard-Transformation  $H_{n+1}$  an:  
 $|x\rangle|y\rangle \leftarrow H_{n+1}|x\rangle|y\rangle$
3. Wende  $f$  aus:  
 $|x\rangle|y\rangle \leftarrow U_f|x\rangle|y\rangle$
4. Wende die Hadamard-Transformation auf  $|x\rangle$  an:  
 $|x\rangle|y\rangle \leftarrow (H_n|x\rangle)|y\rangle$
5. Miss das Register:  
Ist  $|x\rangle = |0 \dots 0\rangle$ : Ausgabe *konstant*,  
Sonst: Ausgabe *balanciert*.

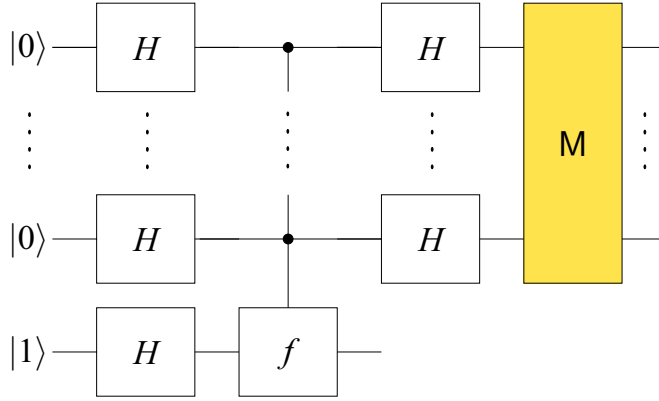


Abbildung 2.20: Schaltkreis für den Algorithmus von Deutsch-Jozsa

**Analyse**

Schritt 2

Wir beginnen mit Schritt 2, der Anwendung von  $H_{n+1}$  auf  $|0 \dots 0\rangle|1\rangle$ . Dies entspricht der Anwendung von  $H_n$  auf  $|0 \dots 0\rangle$  zusammen mit  $H$  auf  $|1\rangle$ :

$$|0 \dots 0\rangle|1\rangle \xrightarrow{H_{n+1}} \left( \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \right) \cdot \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |\phi_2\rangle.$$

Schritt 3

Zunächst betrachten wir, wie die Anwendung von  $U_f$  im Schritt 3 für ein fixiertes  $x$  wirkt:

$$\begin{aligned} |x\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) &\xrightarrow{U_f} |x\rangle \frac{1}{\sqrt{2}} (|f(x)\rangle - |1 \oplus f(x)\rangle) \\ &= (-1)^{f(x)} \cdot |x\rangle \cdot \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle). \end{aligned}$$

Also hat  $U_f$  auf die Superposition  $|\phi_2\rangle$  die Wirkung:

$$|\phi_2\rangle \xrightarrow{U_f} \left( \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \right) \cdot \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |\phi_3\rangle.$$

Schritt 4

Im vierten Schritt wird die Hadamard-Transformation auf den  $|x\rangle$ -Teil angewendet, für Zwischenschritte verweisen wir auf den vorhergehenden Abschnitt.

$$|\phi_3\rangle \xrightarrow{H_n \otimes I_2} \left( \frac{1}{2^n} \sum_{z=0}^{2^n-1} \sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot z} |z\rangle \right) \cdot \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |\phi_4\rangle.$$

Schritt 5

Um die Auswirkung der Messung zu untersuchen, betrachten wir  $|x_{n-1} \dots x_0\rangle$  für ein festes  $|z\rangle$ , also die (nicht normierte) Summe

$$|\phi'_4\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot z} |z\rangle.$$

Nehmen wir zunächst an,  $f$  sei *konstant*. Dann ist  $f(x)$  für alle  $x$  gleich. Fall: konstant  
Im Fall  $z = 0$  gilt außerdem  $x \cdot z = 0$  und wir erhalten in diesem Fall:

$$|\phi'_4\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} \pm |0\rangle = \pm |0\rangle.$$

Messen von  $|x_{n-1} \dots x_0\rangle$  liefert also das Ergebnis  $|0\rangle$ . Da  $|\phi'_4\rangle$  ein zulässiger Zustand von  $|x_{n-1}, \dots, x_0\rangle$  ist, verschwindet die Amplitude von jedem  $|z\rangle$  mit  $z \neq 0$ . Es ist instruktiv, das nachzurechnen:

**Aufgabe 2.23:** Zeigen Sie, dass  $|z\rangle$  für  $z \neq 0$  die Amplitude 0 hat.

Wenn  $f$  balanciert ist, mit welcher Wahrscheinlichkeit beobachten wir dann  $|0\rangle$ ? Für  $z = 0$  ist Fall: balanciert

$$|\phi'_4\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |0\rangle.$$

Für die eine Hälfte der  $x$  ist  $f(x) = 0$ , für die andere ist  $f(x) = 1$ . Also ist die Amplitude von  $|0\rangle$  gleich 0.  $\diamond$

Das folgende Problem lässt sich auf ganz ähnliche Weise lösen: wieder müssen wir etwas über eine Funktion  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  herausfinden, die uns als Black Box gegeben ist. Wir wissen, dass es sich um eine der  $2^n$  Funktionen Bernstein-Vazirani

$$f_a(x) = a \cdot x = a_0 x_0 \oplus \dots \oplus a_{n-1} x_{n-1} \text{ für } a \in \{0, 1\}^n$$

handelt. Die Funktion ist also über einen Vektor  $a \in \{0, 1\}^n$  definiert und bildet jede Eingabe auf das Skalarprodukt mit  $a$  ab.

### Der Algorithmus von Bernstein-Vazirani

Wir verwenden ein Register mit  $n + 1$  Quantenbits  $|x_{n-1} \dots x_0\rangle |y\rangle$

1.  $|x_{n-1} \dots x_0\rangle |y\rangle \leftarrow |0 \dots 0\rangle |1\rangle$
2. Wende die Hadamard-Transformation  $H_{n+1}$  an:  
 $|x\rangle |y\rangle \leftarrow H_{n+1} |x\rangle |y\rangle$
3. Werte  $f$  aus:  
 $|x\rangle |y\rangle \leftarrow U_f |x\rangle |y\rangle$
4. Wende die Hadamard-Transformation auf  $|x\rangle$  an:  
 $|x\rangle |y\rangle \leftarrow (H_n |x\rangle) |y\rangle$
5. Miss das Register:  
Ist  $|x\rangle = |a\rangle$ :  
Ausgabe Das Bauteil berechnet die Funktion  $f_a(x) = a \cdot x$ .

Die Ähnlichkeit mit dem Algorithmus von Deutsch-Jozsa ist so groß, dass die Analyse dem Leser überlassen wird.

**Aufgabe 2.24:** Analysieren Sie den Algorithmus von Bernstein-Vazirani und zeigen Sie, dass er korrekt arbeitet.

Die Idee stammt von Ethan Bernstein und Umesh Vazirani (1993).

# 3 Vom Quantenregister zum Quantenschaltkreis

*Information is physical.*  
Rolf Landauer

Dieses Kapitel legt notwendige Grundlagen für Verständnis und Entwicklung von Quantenverfahren; es enthält aber auch einige überraschende Ergebnisse. Wir beginnen mit einem Schnellkurs „Laufzeitanalyse von Algorithmen“. Daran schließt sich eine genauere Betrachtung von Quantenschaltkreisen an, die unser grundlegendes Berechnungsmodell bilden. In Abschnitt 3.5 werden wir sehen, wie sich jeder effiziente klassische Algorithmus in einen Quantenschaltkreis überführen lässt, der seinerseits *effizient* rechnet. Dieses Ergebnis werden wir regelmäßig anwenden und Aufgaben, die klassisch realisierbar sind, in Quantenalgorithmen integrieren.

Der Rest dieses Kapitels beschäftigt sich mit einigen Unterschieden zwischen Quantenbits und klassischen Bits. Abschnitt 3.4 zeigt, dass sich Quantenbits nicht kopieren lassen. Das ist eine Einschränkung, die bei dem Entwurf von Quantenalgorithmen beachtet werden muss. Diese spielt auch eine entscheidende Rolle für die Sicherheit von Quantenverschlüsselungsverfahren. Des Weiteren lassen sich die Zustände von Quantenbits nur unter bestimmten Voraussetzungen unterscheiden (Abschnitt 3.6).

Im letzten Abschnitt dieses Kapitels untersuchen wir, wie störanfällig Quantenberechnungen sind. Diese Frage stellt sich dringlicher als bei klassischen Berechnungen, da verschiedene Zustände eines Quantenbits beliebig nah beieinanderliegen können. Wie beeinflussen geringe Veränderungen eines solchen Zustandes das Ergebnis einer Berechnung? Wir werden erfahren: Quantenberechnungen sind recht robust. Dies ist eine wichtige Voraussetzung für den Bau praktisch einsetzbarer Quantenrechner.

### 3.1 Laufzeit

Eine unserer Grundfragen lautet: *Was können wir mit einem Quantencomputer tun?* Daraus ergibt sich eine weitere Frage: *Welche Aufgaben können wir mit einem Quantencomputer besser erledigen, als mit einem herkömmlichen Rechner?* Den Ausdruck *besser* interpretieren wir dabei als *effizienter*. Darum interessiert uns der Speicherverbrauch und insbesondere die Laufzeit eines Rechenverfahrens, sei es nun für einen klassischen oder für einen Quantencomputer konzipiert.

Starten wir ein Rechenverfahren auf einem konkreten Computer, könnten wir die Rechenzeit mit einer Stoppuhr bestimmen. Problematisch ist dabei, dass die gestoppte Zeit von dem verwendeten Rechner abhängt. Sie ist nicht geeignet, um verschiedene Rechenverfahren zu vergleichen. Außerdem benötigen wir eine Methode zur Bewertung der Laufzeit von Quantenverfahren, die ohne einen konkreten Quantenrechner auskommt.

Definition

Laufzeit

Wir definieren die *Laufzeit* einer Berechnung als die Anzahl ihrer *Schritte*. Damit ist die Laufzeit eines Algorithmus für eine Eingabe  $x$  die Anzahl der Schritte, die bei der Berechnung auf  $x$  ausgeführt werden. Sofort entstehen zwei Fragen: Wie sieht so ein Schritt aus? Was für einen Rechner legen wir zu Grunde? Wir werden in diesem Abschnitt sehen, dass diese Fragen für uns keine Rolle spielen werden. Es ist möglich, die Laufzeit hardwareunabhängig zu betrachten. Dann können wir uns vorstellen, dass unsere Algorithmen von einem gewöhnlichen PC ausgeführt werden (das theoretische Modell dazu nennt man *Registermaschine*). Als Schritt betrachten wir dabei Befehle einer höheren Programmiersprache, deren Zeitaufwand nicht von der Eingabegröße abhängt. Ein gleichwertiger Zugang könnte auch die Schritte einer Turingmaschine zählen. Wir kommen am Ende des Abschnitts darauf zurück.

Eingabegröße

Die Laufzeit eines Algorithmus ist von der Eingabegröße abhängig: Zwei 8-Bitzahlen lassen sich schneller addieren als zwei 100-Bitzahlen. Wie die *Eingabegröße* definiert ist, hängt von dem Problem ab. Sollen wir ein Feld sortieren oder aus einer Menge von Zahlen die kleinste herausfinden, ist die *Anzahl der Elemente* die Eingabegröße. Für andere Probleme verwenden wir die Anzahl der Bits, die nötig ist, um die Eingabe zu speichern. Sind zwei Zahlen  $m$  und  $n$  zu addieren oder zu multiplizieren, ist die Eingabegröße damit  $\log_2 m + \log_2 n$ . Im konkreten Fall liegt die natürlichste Definition meist auf der Hand.

Ab jetzt betrachten wir die Laufzeit eines Algorithmus als Funktion  $T(n)$ , die uns zu der Eingabegröße die Anzahl der Schritte liefert. Das folgende Beispiel verdeutlicht, welche Unterschiede im Laufzeitverhalten relevant sind und welche wir vernachlässigen können.

**Beispiel 3.1:** Wir betrachten zwei fiktive Verfahren, die zwei  $m$ -Bit-Zahlen auf irgendeine Art addieren. Die Eingabegröße ist  $n = 2 \cdot m$ . Der erste Algorithmus addiert mit

$$t_1(n) = 10 \cdot n + 1000$$



einfachen Operationen. Es scheint kein sehr gutes Verfahren zu sein, da selbst für sehr kleine Zahlen mehr als tausend Schritte ausgeführt werden müssen. Ein zweites Verfahren addiert  $m$ -Bitzahlen mit

$$t_2(n) = n^2$$

derartiger Operationen; für kleine Eingaben ist das zweite Verfahren sicherlich besser.

Das erste Verfahren benötigt für 4-Bit Zahlen  $t_1(8) = 1080$  Operationen, das zweite  $t_2(8) = 64$  und ist damit deutlich schneller. Ist die Eingabe jedoch 50 Bits lang, ist das erste Verfahren besser. Werte für verschiedene Eingabegrößen finden sich in der folgenden Tabelle.

$n$	$t_1(n)$	$t_2(n)$
8	1080	64
50	1500	2500
100	2000	10000
1000	11000	$10^6$

Für große Eingaben ist das erste Verfahren das bei weitem bessere. Der hier auftretende Unterschied ist von viel größerer Bedeutung, als der bei kleinen Eingaben, wie ein Blick auf die Tabelle zeigt. Die Funktion  $t_2$  wächst *quadratisch* in der Eingabegröße,  $t_1$  hingegen *linear*. Betrachten wir statt  $t_1$  und  $t_2$  beispielsweise die Funktionen

$$s_1(n) = 20 \cdot n + 100$$

und

$$s_2(n) = 3 \cdot n^2 + n + 50,$$

ergibt sich ein sehr ähnliches Resultat. Außerdem lässt sich erkennen, dass sich  $t_1$  und  $s_1$  sehr ähnlich verhalten, ebenso gleichen sich  $t_2$  und  $s_2$ . Zwischen diesen Paaren von Funktion besteht jedoch eine große Differenz.

**Aufgabe 3.1:** Ermitteln Sie die Werte von  $s_1$  und  $s_2$  für  $n = 8, 50, 100, 1000$ .

Das *Wachstum* der Laufzeit in der Eingabegröße scheint der entscheidende Faktor zu sein. Die Funktion  $n^2$  wächst so viel schneller als  $n$ , dass die *konstanten* Faktoren, beispielsweise bei den Funktionen  $3n^2$  und  $20n$ , nicht ins Gewicht fallen. Die Funktionen  $t_1$  und  $s_1$  sind von der Form

$$a \cdot n + b,$$

$t_2$  und  $s_2$  haben die Gestalt

$$c \cdot n^2 + d \cdot n + e.$$

Die Konstanten  $a, b, c, d, e$  spielen nur für sehr kleine Eingaben eine Rolle. Entscheidender ist der Unterschied in der *Größenordnung*:  $t_1$  und  $s_1$  besitzen keinen quadratischen Anteil  $n^2$ .

Folgende Tabelle enthält wichtige Funktionen, wobei nur zwei von gleicher Größenordnung sind; diesen Begriff werden wir im Anschluss formal definieren (die Werte sind gerundet).

$n$	$\log_2 n$	$\sqrt{n}$	$n$	$n^2$	$2n^2 + 3n - 5$	$n^3$	$2^n$
8	3	3	8	64	147	512	256
30	5	6	30	900	1885	27000	$10^9$
100	7	10	100	10000	20295	$10^6$	$10^{30}$
200	8	15	200	40000	80595	$8 \cdot 10^6$	$10^{60}$
1000	10	32	1000	$10^6$	$(2 + \frac{3}{1000}) \cdot 10^6$	$10^9$	$10^{301}$
10000	14	100	10000	$10^8$	$(2 + \frac{3}{10000}) \cdot 10^8$	$10^{12}$	$10^{3010}$
$10^6$	20	1000	$10^6$	$10^{12}$	$(2 + \frac{3}{10^6}) \cdot 10^{12}$	$10^{18}$	$10^{30100}$
$10^9$	30	31623	$10^9$	$10^{18}$	$10^{18}$	$10^{27}$	$10^{3 \cdot 10^8}$

Interpretation  
der Tabelle

Was können wir dieser Tabelle entnehmen? Die sechs Funktionen

$$\log_2 n, \sqrt{n}, n, n^2, n^3 \text{ und } 2^n$$

unterscheiden sich deutlich in ihrem Wachstum. Der Unterschied zwischen  $n^2$  und  $2n^2 + 3n - 5$  tritt mit wachsendem  $n$  immer mehr in den Hintergrund. Dagegen wird die Differenz zwischen  $n^2$  und  $n$ , und genauso die zwischen  $n^2$  und  $n^3$ , mit wachsendem  $n$  immer relevanter.

Wir stellen weiter fest:  $\log_2 n$  ist eine sehr langsam wachsende Funktion.<sup>1</sup> Auch ein Ressourcenverbrauch von der Größenordnung  $\sqrt{n}$  oder  $n$  ist als eher gering einzustufen. Die Funktionen  $n^2$  und  $n^3$  wachsen deutlich schneller: Algorithmen mit quadratischer oder kubischer Laufzeit wirken nicht gerade flink, scheinen aber noch ausführbar zu sein. Wächst der Ressourcenverbrauch eines Verfahrens jedoch wie die Funktion  $2^n$  exponentiell, ist es nur für sehr kleine Eingaben anwendbar.

O-Notation

Nun folgt die exakte Definition des Begriffes *Größenordnung*, wir verwenden ihn synonym mit *asymptotisches Wachstum*. Die sogenannte Groß-O-Notation beschreibt eine *obere Grenze* des Verhaltens von Funktionen. Dabei betrachten wir Funktionen, die die Eingabegröße eines Berechnungsproblems auf die Zahl der Schritte abbildet, das heißt Funktionen

$$f : \mathbb{N} \rightarrow \mathbb{N}.$$

Wir sagen,

$$f(n) \text{ ist } O(g(n)), \text{ oder } f(n) = O(g(n))$$

wenn  $f(n)$  asymptotisch nicht schneller wächst als  $g(n)$ . Das ist der Fall, wenn die Funktion  $f(n)$  ab einer bestimmten Stelle  $n_0$  kleiner als  $g(n)$  ist oder sich nur um einen konstanten Faktor von  $g(n)$  unterscheidet. Abbildung 3.1 illustriert die formale Definition.

<sup>1</sup>Wie bei Laufzeitbetrachtungen üblich, werden wir in der Folge statt  $\log_2 n$  nur  $\log$  verwenden.

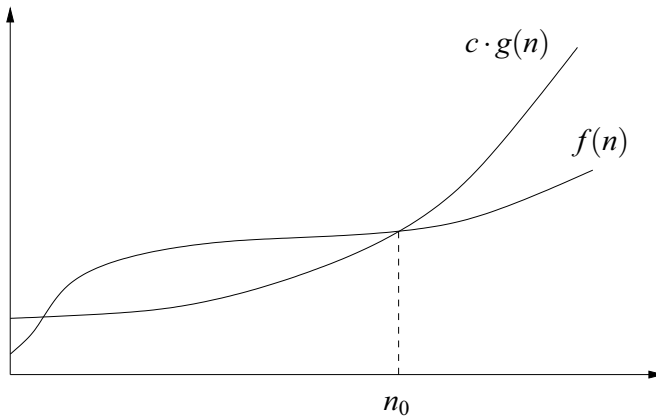


Abbildung 3.1: Zwei Funktionen: es gilt  $f = O(g(n))$

Für zwei Funktionen  $f, g$  gilt, dass  $f$  asymptotisch nicht stärker wächst als  $g$ ,

$$f = O(g(n)),$$

wenn es eine natürliche Zahl  $n_0$  gibt und eine Konstante  $c$ , so dass

$$f(n) \leq c \cdot g(n) \text{ für alle } n \geq n_0.$$

### Beispiel 3.2:

1. Die Funktion  $f(n) = 6n$  wächst schneller als  $g(n) = 3n$ . Sie unterscheiden sich jedoch nur um einen konstanten Faktor: Für alle  $n$  gilt

$$f(n) \leq 2 \cdot g(n),$$

also gilt  $f(n) = O(g(n))$ . Ebenso gilt  $g(n) = O(f(n))$ . Beide Funktionen sind von der *linearen* Größenordnung  $O(n)$ .

2. Die lineare Funktion  $f(n) = 6n + 100$  hat für kleine  $n$  größere Funktionswerte als  $g(n) = n^2 - 12$ . Jenseits der Stelle  $n_0 = 14$  – hier schneidet die Parabel  $g(n)$  die Gerade  $f(n)$  – ist jedoch  $g(n)$  größer. Also gilt

$$f(n) = O(g(n)).$$

Die umgekehrte Aussage gilt nicht:  $g(n) \neq O(f(n))$ , da sich  $f(n)$  und  $g(n)$  nicht durch einen konstanten Faktor, sondern um einen Faktor der Größenordnung  $n$  unterscheiden.

Die Relation  $f = O(g(n))$  bedeutet  $f$  *kleiner oder gleich*  $g$  in Bezug auf die Größenordnung. Das Verhältnis *größer oder gleich* drücken wir mit dem

Symbol  $\Omega$  aus.  $\Theta$  findet Verwendung, wenn zwei Funktionen von *gleicher* Größenordnung sind.

Für zwei Funktionen  $f, g$  gilt, dass  $f$  asymptotisch mindestens so stark wächst wie  $g$ ,

$$f = \Omega(g(n)),$$

wenn  $g = O(f(n))$  gilt.  $f$  und  $g$  sind von der gleichen Größenordnung,

$$f = \Theta(g(n)),$$

wenn  $f = O(g(n))$ , und  $f = \Omega(g(n))$  gilt.

Wir verwenden diese Schreibweise stets in Aussagen der Form „Algorithmus A hat Laufzeit  $O(n \log n)$ “. Das heißt, die Laufzeit des Algorithmus wächst asymptotisch betrachtet wie  $n \log n$ .

**Aufgabe 3.2:** Zeigen Sie die folgenden Aussagen.

- $n = O(n^2)$ ,  $n^2 \neq O(n)$
- $3n^2 + 1000n + 12 \log n = O(n^2)$
- $n \log n \neq O(n)$ , aber  $n(\log n)^k = O(n^2)$  für alle  $k$ .

### Einheitskostenmaß

Wir betrachten die Laufzeit nur der Größenordnung nach. Dabei werden Konstanten eliminiert, und darum vernachlässigen wir, ob eine Operation einen einfachen Schritt, zwei, fünf oder hundert einfache Schritte benötigt, sofern diese Zahl von der Eingabegröße unabhängig ist. Einfache Schritte mit Laufzeit  $O(1)$  sind etwa Zuweisungen, Vergleiche, Feldzugriffe und arithmetische Operationen wie Additionen oder Multiplikationen. Wir sprechen vom *Einheitskostenmaß*. Abhängig von der Eingabegröße sind in der Regel Schleifen und Rekursionen.

Allerdings ist das Einheitskostenmaß eine Vereinfachung, die in einigen Fällen nicht zulässig ist. So können wir den arithmetischen Operationen nur dann konstanten Aufwand zuordnen, wenn die Zahlen klein sind – genau genommen kleiner einer zuvor festgesetzten Konstanten. Addieren oder multiplizieren wir  $m$ -Bitzahlen, sind dafür  $T(m)$  Schritte notwendig. Diese Zahl hängt von der Länge der Zahlen ab. Kommen wir mit einer festen Länge aus – der Java-Typ `int` zum Beispiel verwendet 32 Bit – ist diese Laufzeit eine Konstante.  $T(32)$  ist eine konkrete ganze Zahl, und wir ordnen einer Addition zweier Zahlen den Aufwand  $O(1)$  zu. Addieren wir zehn Zahlen, ist der Aufwand weiterhin  $O(1)$ ; erst wenn die Anzahl der arithmetischen Operationen von der Eingabegröße abhängt, ändert sich die Größenordnung.

Betrachten wir Algorithmen, die mit beliebig großen Zahlen rechnen, muss die Länge der Binärdarstellung der Zahlen in die Laufzeitanalyse eingehen.

Man spricht von dem *logarithmischen Kostenmaß*. Zum Beispiel ist es für viele moderne Verschlüsselungsverfahren wichtig, große Zahlen darauf zu testen, ob sie Primzahlen sind. Ist  $n$  die Eingabe eines Primzahltests, so ist die Eingabegröße  $\log n$ . Für arithmetische Operationen mit so großen Zahlen können wir nicht Kosten  $O(1)$  veranschlagen. Wir müssen die Anzahl der Bitoperationen zählen; für Additionen ergibt sich Größenordnung  $O(\log n)$  und Multiplikationen kommen mit  $O((\log n)^2)$  Bitoperationen aus.

**Beispiel 3.3:** Wir bestimmen den Mittelwert von  $n$  Zahlen, die in einem Feld  $F[0, \dots, n-1]$  gespeichert sind. Die Zahlen sind klein, und wir legen das Einheitskostenmaß zu Grunde. Die Eingabegröße ist damit die Anzahl  $n$ .

Mittelwert-  
bestimmung

1.  $R \leftarrow 0$ ,  
    $i \leftarrow 0$
2. Solange  $i < n$  gilt:
  - 2.1.  $R \leftarrow R + F[i]$
  - 2.2.  $i \leftarrow i + 1$
3.  $R \leftarrow R/n$

Schritt 1 ist für alle Eingaben gleich, also von deren Größe unabhängig: die Laufzeit ist konstant beziehungsweise  $O(1)$ . In Schritt 2 durchläuft eine Schleife die  $n$  Feldelemente. Für jedes Feldelement wird dieses auf das Register addiert, wird der Schleifenzähler inkrementiert und ein Vergleich ausgeführt, von dem abhängt, ob die Schleife abgebrochen wird. Jeder Durchlauf der Schleife hat im Einheitskostenmaß Laufzeit  $O(1)$  und Schritt 2 insgesamt  $O(n)$ . Schritt 3 hat im Einheitskostenmaß Laufzeit  $O(1)$  und die Gesamtlaufzeit ist  $O(1) + O(n) + O(1) = O(n)$ .

*Bemerkung:* Dem aufmerksamen Leser ist vielleicht Folgendes aufgefallen: wenn wir auch die Feldelemente als klein voraussetzen können, so sind doch Schritt 3 und der Vergleich  $i < n$  in Schritt 2 nicht von der Eingabegröße unabhängig. Hier ist es dennoch sinnvoll das Einheitskostenmaß zu Grunde zu legen, da man einen Vergleich oder die Division zweier Zahlen, die zum Beispiel aus 32 Bits bestehen, als feste Folge von Operationen ansehen kann, wohingegen die Laufzeit der Schleife unmittelbar von der Länge des Feldes abhängt.  $\diamond$

### Erwartete Laufzeit und Laufzeit im schlechtesten Fall

Formal haben wir die Laufzeit eines Algorithmus bisher auf *einer* Eingabe betrachtet. Interessant ist das Laufzeitverhalten bezüglich *aller* Eingaben. Die Analyse in Beispiel 3.3 ist zwar von konkreten Eingaben unabhängig, trotzdem müssen wir berücksichtigen, dass ein Verfahren für verschiedene Eingaben Laufzeiten unterschiedlicher Größenordnung haben kann.

Zunächst nehmen wir einen etwas pessimistischen Standpunkt ein: Was hilft es uns, wenn unser Algorithmus für viele Eingaben recht gut ist, zum Beispiel linear, für einige aber sehr lange braucht, etwa Laufzeit  $\Omega(n^3)$  hat? In konkreten Anwendungen ist es möglich, dass wir ausschließlich ungünstige Eingaben zu bewältigen haben. Um eine Aussage zu treffen, auf die wir uns stets verlassen können, betrachten wir die *Laufzeit im schlechtesten Fall*.

worst-case

Wir verwenden die Eingabe, auf der der Algorithmus am längsten rechnet; in unserem Fall ergibt sich  $\Omega(n^3)$ .

average-case

Oft gibt es gute Gründe, etwas großzügiger zu sein. Beispielsweise wenn ein Algorithmus für fast alle Eingaben sehr schnell und nur für sehr wenige langsam ist. Um das zu berücksichtigen, stellen wir uns vor, wir wählen die Eingaben zufällig, und zwar jede mit derselben Wahrscheinlichkeit; die Eingaben werden gemäß der Gleichverteilung gezogen. Die *erwartete Laufzeit* ist der Mittelwert der Laufzeiten der Eingaben. Die erwartete Laufzeit ist immer dann aussagekräftig, wenn ein Algorithmus auf sehr verschiedenen Eingaben ausgeführt wird.

Ein Beispiel wird zu Beginn von Abschnitt 6.1 betrachtet. Dort werden eine worst-case und eine average-case Analyse aller möglichen Algorithmen, die ein Suchproblem lösen, durchgeführt.

Damit haben wir den Begriff Laufzeit (als Funktion der Eingabegröße) auf Algorithmen übertragen und formal definiert. Es ist die Laufzeit als Funktion der Eingabegröße. Schlechtester und mittlerer Fall sind alternative Betrachtungsweisen. Nun können wir zwei wichtige mit einander zusammenhängende Begriffe definieren:

Polynomiale  
Laufzeit

Eine Funktion  $f: \mathbb{N} \rightarrow \mathbb{N}$  heißt *polynomial*, falls

$$f = O(n^k) \text{ für konstantes } k.$$

Wir nennen einen Algorithmus *effizient*, wenn seine Laufzeit als Funktion der Eingabegröße polynomial ist.

*Zusammenfassung:* Um die Qualität verschiedener Rechenverfahren zu vergleichen, betrachten wir deren Laufzeit. Genauer: das *asymptotische Wachstum* bzw. die *Größenordnung der Laufzeit* in der Eingabegröße. Der Schlüssel zur Laufzeitanalyse: Kommt ein bestimmter Vorgang mit einer konstanten Anzahl an Schritten aus, deren Aufwand jeweils von der Eingabegröße unabhängig ist, ordnen wir ihm konstante Laufzeit  $O(1)$  zu. Ein solcher Vorgang kann im Weiteren selbst wie ein Schritt behandelt werden.

## 3.2 Klassische Schaltkreise und Komplexität

Es gibt viele Möglichkeiten, Algorithmen zu beschreiben; dies geschieht mit verschiedenen Berechnungsmodellen. Bisher haben wir Turingmaschinen kennen gelernt (Abschnitt 2.1.1), Registermaschinen im letzten Abschnitt kurz erwähnt und im Abschnitt 2.1.2 Schaltkreise definiert. Dieser kurze theoretische Abschnitt beschreibt den Zusammenhang zwischen ebendiesen Schaltkreisen und dem Begriff *Laufzeit*. Dazu beschränken wir uns auf klassische Schaltkreise, *deren Gatter höchstens zwei Eingabebits haben*. Unser Interesse gilt der Anzahl solcher Gatter.

Die *Größe* eines Schaltkreises ist die Anzahl seiner Gatter.

Es liegt auf der Hand, dass die so definierte Größe eines Schaltkreises etwas mit dem Aufwand des beschriebenen Algorithmus zu tun hat. Schaltkreise unterscheiden sich von anderen Berechnungsmodellen in einem wichtigen Punkt. Eine Turingmaschine muss für Eingaben jeder Größe die korrekte Lösung berechnen. Ein Schaltkreis ist hingegen für Eingaben einer festen Größe konzipiert. Zum Beispiel rechnet ein 32-Bit-Addierer mit binären Eingaben der Länge 32. Für längere Zahlen muss ein neuer Schaltkreis konstruiert werden.

Aus diesem Grund betrachtet man Folgen von Schaltkreisen  $(C_n)$ . Der Index  $n$  durchläuft dabei die natürlichen Zahlen größer gleich 1.  $C_i$  – der  $i$ -te Schaltkreis der Folge – rechnet mit Eingaben der Größe  $i$ . Eine Folge von Additionsschaltkreisen  $(C_n)$  besteht zum Beispiel aus Addierern von je zwei Zahlen der Länge  $n$ .

Die von uns betrachteten Schaltkreise müssen eine entscheidende Bedingung erfüllen: sie müssen effizient berechenbar sein. Wir definieren: Eine Folge  $(C_n)$  von Schaltkreisen heißt *uniform*, wenn es einen *effizienten* Algorithmus  $A$  gibt, der aus dem Wert  $i \geq 1$  die Gestalt des Schaltkreises  $C_i$  berechnet.  $A$  berechnet aus der Eingabegröße, welche Gatter verwendet werden und wie sie durch Leitungen verbunden sind.

Uniforme  
Schaltkreise

Wir müssen uns um den Algorithmus  $A$  nicht viele Gedanken machen, da wir nur Schaltkreisfolgen  $(C_n)$  mit den folgende Eigenschaften betrachten:

- Sie sind *polynomial*, das heißt, es gibt ein konstantes  $k$ , so dass die Größe jedes Schaltkreises  $C_i$  von der Ordnung  $O(i^k)$  ist.
- Der Aufbau ist regelmäßig: aus der Gestalt des Schaltkreises für Eingaben der Größe  $i$  ist die Gestalt für alle anderen Eingabegrößen ablesbar. Also kann Algorithmus  $A$  nicht kompliziert sein.

Damit haben wir das Ziel dieses Abschnitts erreicht. Wir können den Zusammenhang zwischen Schaltkreisgröße und Laufzeit formulieren.

Ein Berechnungsproblem lässt sich dann und nur dann in polynomialer Zeit lösen, wenn es von uniformen Schaltkreisen polynomialer Größe berechnet werden kann.

Wir betrachten die Beweisidee der für uns interessanten Richtung dieser Aussage: wenn wir für ein Problem polynomiale (uniforme) Schaltkreise angeben können, ist es effizient – sprich: in Polynomialzeit – lösbar.

Die Idee lautet: wir nehmen die Schaltkreisfolge her und simulieren deren Berechnung mit einer Turingmaschine. Wir sprechen von *simulieren*, wenn die Turingmaschine die Berechnung des Schaltkreises direkt nachäfft, ohne dass weitere Ideen oder Fähigkeiten eingebracht werden. Besagte Turingmaschine arbeitet wie folgt: Wird ihr eine Eingabe vorgelegt, bestimmt sie deren Größe.

Daraus berechnet sie effizient die Gestalt des passenden Schaltkreises  $C$ ; das ist effizient möglich, da wir uniforme Schaltkreise betrachten. Die Gestalt von  $C$  befindet sich auf dem Band. Sie beginnt an den Eingabebits und durchläuft den Schaltkreis gatterweise. Auf diese Weise kann sie die Ausgaben jedes einzelnen Gatters und schließlich die des gesamten Schaltkreises  $C$  bestimmen.

Dieses Ergebnis gilt auch für Quantenschaltkreise, die wir im folgenden Abschnitt einführen. Ein Quantenalgorithmus ist *effizient*, wenn er durch einen Quantenschaltkreis polynomialer Größe beschrieben wird.

### 3.3 Quantengatter und Quantenschaltkreise

Klassische Schaltkreise bestehen aus Leitungen und Gattern. Ganz analog bestehen Quantenschaltkreise aus Quantenleitungen und Quantengattern. Jede Quantenleitung entspricht einem Quantenbit und ein Quantengatter führt eine unitäre Transformation aus, siehe Abbildung 3.2.

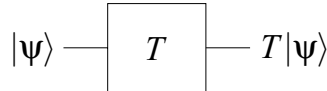


Abbildung 3.2: Quantengatter für die unitäre Transformation  $T$

Einer Berechnung mit mehreren parallelen Leitungen wie in Abbildung 3.3 ergibt sich aus dem Tensorprodukt der einzelnen Gatter. Der Endzustand  $|x', y', z'\rangle$  des Drei-Bit-Registers ist gerade

$$(I_2 \otimes W \otimes I_2)(U \otimes V)|x, y, z\rangle.$$

Damit ist der Aufbau von Quantenschaltkreisen im Wesentlichen geklärt, wir können auf eine formale Definition verzichten. Im Laufe des Abschnitts wird sich unser Bild vom Quantenschaltkreis nach und nach vervollständigen.

Auf zwei Dinge sei schon an dieser Stelle hingewiesen:

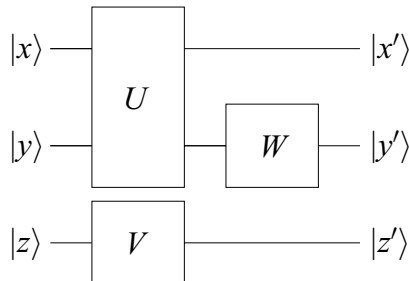


Abbildung 3.3: Quantenschaltkreis mit drei Gattern



1. Da Quantenschaltkreise umkehrbare Berechnungen ausführen, haben sie stets ebenso viele Ausgabebits wie Eingabebits.
2. Wir werden in Abschnitt 3.4 sehen, dass sich Quantenbits nicht kopieren lassen. Darum dürfen sich die Leitungen unserer Quantenschaltkreise nicht verzweigen. Ebenso wenig darf das Ergebnis eines Gatters mehrfach verwendet werden. Ein Quantengatter führt eine unitäre Transformation auf den beteiligten Bits aus. Es ist nicht möglich, die Anzahl der Bits zu erhöhen.

Quantenschaltkreise stellen Quantenalgorithmen sehr übersichtlich dar. Mit einem Blick lässt sich die Folge der Transformationen erkennen, an denen ein bestimmtes Bit beteiligt ist. Zudem erleichtern sie es, Aufwand oder Ressourcenverbrauch von Quantenalgorithmen zu ermitteln.

Wie im klassischen Fall ist die Größe eines Schaltkreises die Anzahl seiner Gatter. Dabei setzen wir voraus, dass jedes Gatter höchstens drei Bits in seine Berechnung einbezieht: ein Schritt einer Quantenberechnung ist *lokal*. Wieso gerade drei? Klarerweise darf die Anzahl der beteiligten Bits nicht mit der Anzahl der Eingabebits eines Schaltkreises wachsen. Sonst könnte man einzelne Gatter von gewaltiger Komplexität erdenken, die nicht praktisch realisierbar sind. Konkret konnte gezeigt werden, dass Quantengatter mit zwei Bits genügen (um alles zu berechnen, was mit Quantengatter mit konstant vielen Bits erreichbar ist). In Abschnitt 3.5 verwenden wir allerdings das sogenannte Toffoli-Gatter, um Gatter klassischer Schaltkreise durch umkehrbare Gatter zu ersetzen. Dieses verwendet drei Bits.

Lokale Transformationen

Der Zusammenhang zwischen Quantenschaltkreisen und Laufzeit ist der gleiche wie im klassischen Fall.

Ein Berechnungsproblem lässt sich dann und nur dann von einem Quantenalgorithmus in polynomialer Zeit lösen, wenn es von uniformen Quantenschaltkreisen polynomialer Größe berechnet werden kann.

In Abschnitt 3.5 werden wir zeigen, wie sich ein effizienter klassischer Algorithmus ( $\approx$  ein Schaltkreis polynomialer Größe) in einen effizienten Quantenalgorithmus ( $\approx$  ein Quantenschaltkreis polynomialer Größe) überführen lässt.

Nun werden wir die Grundbausteine aller Quantenalgorithmen studieren: Messungen und unitäre Transformationen.

### Messungen

Inwieweit gehören Messungen zu einer Berechnung? Bei den Algorithmen aus dem einführenden Kapitel 2 wird am Ende gemessen. Aus der Messung ergibt sich das Ergebnis der Berechnung. Es hindert uns jedoch nichts daran, auf das Ergebnis einer solchen Messung weitere unitäre Transformationen anzuwenden. Zudem ist es möglich, nur einzelne Quantenbits zu messen (siehe Abschnitt 2.8). In diesem Fall enthält der Folgezustand des Quantenregisters Informationen, die einer weiteren Messung zugänglich sind. Ein

Verfahren, das davon Gebrauch macht, ist die Quantenteleportation in Kapitel 5. Es wird zweimal gemessen, und zwischen den Messungen werden unitäre Transformationen angewendet, die von dem Ergebnis der ersten Messung abhängen.

Das legt folgende Sichtweise nahe: Eine Messung ist kein Rechenschritt im Sinne eines festen und planbaren Überganges von einem Zustand des Quantenregisters zu einem anderen. Eine Messung ist keine unitäre Transformation und auch nicht umkehrbar. Trotzdem sind Messungen Teil der Berechnung. Darum drücken wir Messungen durch ein Symbol aus, das den Quantengattern ähnlich sieht, siehe Abbildung 3.4. Um die Skizzen von Schaltkreisen nicht mit Information zu überfrachten, geben wir die Basis, bezüglich der gemessen wird, nicht an.



Abbildung 3.4: Symbol für die Messung eines Bits

Mathematisch korrekt formuliert ist der Folgezustand einer Messung eine Wahrscheinlichkeitsverteilung über Quantenzustände, ein so genannter *gemischter Zustand*. Für unsere Zwecke ist auch ohne formale Definition klar, wie nach einer Messung vorgegangen werden kann.

### Grundlegende unitäre Transformationen

Hadamard-  
Transformation

Eine wichtige unitäre Transformation haben wir schon oft verwendet: die Hadamard-Transformation. Die Variante, die auf ein Bit wirkt, bezeichnen wir mit  $H$ . Sie wird durch die Matrix

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

beschrieben. Die Version auf  $n$  Bits ist als  $n$ -faches Tensorprodukt

$$H_n = H \otimes \dots \otimes H$$

definiert und bildet einen Zustand  $|x\rangle$  auf

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

ab. Die Herleitung befindet sich in Abschnitt 2.12. Eine wichtige Verallgemeinerung der Hadamard-Transformation lernen wir im Zusammenhang mit Shors Algorithmus zur Faktorisierung ganzer Zahlen kennen: Abschnitt 8.4 ist der Quanten-Fouriertransformation gewidmet.

CNOT

Eine wichtige klassische Operation ist die Negation.

NOT überführt wahre Aussagen in falsche und bildet ein Bit  $b$  auf das Komplement  $b \oplus 1$  ab. Schon der Name sagt es: wir können im Quantenfall die kontrollierte Negation

$$\text{CNOT} : |a\rangle|b\rangle \rightarrow |a\rangle|a \oplus b\rangle$$

verwenden (siehe Abschnitt 2.3). Das zweite Bit wird negiert, wenn das erste im Zustand  $|1\rangle$  ist. In einer allgemeinen Superposition werden die Amplituden von  $|10\rangle$  und  $|11\rangle$  getauscht.

$$\begin{aligned} & \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \\ \xrightarrow{\text{CNOT}} & \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{11}|10\rangle + \alpha_{10}|11\rangle. \end{aligned}$$

Dieser Zustand ist im Allgemeinen verschränkt, siehe Abschnitt 2.11.

Für das Gatter verwenden wir das Symbol aus Abbildung 3.5, das deutlich macht, wie ein Bit auf das andere addiert wird.

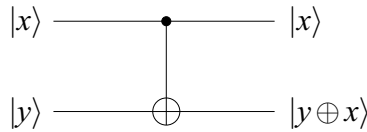


Abbildung 3.5: Symbol des CNOT-Gatters

Die Negation lässt sich auch mit einer unitären Transformation auf einem Bit ausführen. Um die Amplituden von  $|0\rangle$  und  $|1\rangle$  zu tauschen, können wir die Matrix

Pauli-Matrizen

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

verwenden. Die Operation  $X$  wird auch als *Bitflip* bezeichnet. Es ist eine von drei *Pauli-Matrizen*. Neben dem genannten Bitflip  $X$  sind das gerade

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \text{ und } Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Benannt sind die Operationen nach dem österreichischen Physiker Wolfgang Pauli (1900-1958; 1945 Nobelpreis für Physik). Abbildung 3.6 stellt ihre Wirkung auf drei Bits im Zustand  $\alpha|0\rangle + \beta|1\rangle$  dar.  $Z$  negiert gerade die Amplitude der Komponente  $|1\rangle$ . Darum bezeichnet man  $Z$  als *Phaseflip*.

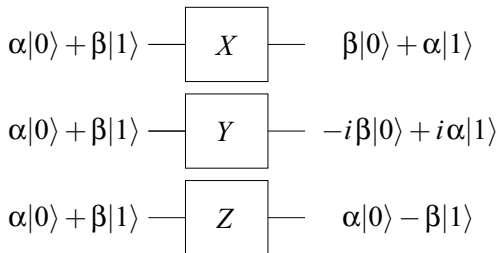
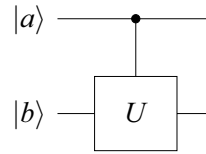


Abbildung 3.6: Schaltkreis mit Pauli-Transformationen

Abbildung 3.7: Anwendung von  $U$  in Abhängigkeit vom ersten Bit

**Aufgabe 3.3:** Zeigen Sie, dass die Transformationen  $X, Y$  und  $Z$  unitär sind.

### Kontrollierte Operationen

Wir betrachten noch einmal die Operation CNOT aus Abbildung 3.5. Sind  $|a\rangle$  und  $|b\rangle$  klassische Zustände, wird das zweite Bit gerade dann negiert, wenn das erste Bit  $|1\rangle$  ist. Für Superpositionen bedeutet das: die Amplituden von  $|10\rangle$  und  $|01\rangle$  werden getauscht. Anders ausgedrückt wird die Negation  $X$  gerade auf die Zustände angewendet, für die das erste Bit  $|1\rangle$  ist.

Diese Idee lässt sich verallgemeinern: Abbildung 3.7 zeigt das Symbol eines Gatters, das gerade dann die unitäre Transformation  $U$  auf das Bit  $|b\rangle$  anwendet, wenn  $|a\rangle = |1\rangle$  gilt.

Diese Operation wird durch die Matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & & U \\ 0 & 0 & & \end{pmatrix}$$

beschrieben. Für  $U = X$  erhalten wir gerade CNOT. Damit könnten wir diese Operation alternativ gemäß Abbildung 3.8 schreiben. Die kontrollierte Hadamard-Transformation – das entspricht  $U = H$  – wird durch die Matrix

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

beschrieben.

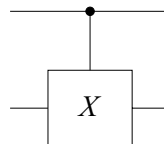


Abbildung 3.8: Eine alternative Schreibweise des CNOT-Gatters

*Bemerkung:* Die kontrollierten Operationen lassen sich im Allgemeinen nicht als Tensorprodukt zweier 1-Bit-Transformationen darstellen. Denken Sie daran, dass wir in Abschnitt 2.11 mit Hilfe von CNOT einen unverschränkten Zustand in einen verschränkten überführt haben.

### 3.4 Quantenbits kopieren: Das No-Cloning-Theorem

In diesem Abschnitt betrachten wir eine vermeintlich einfache Aufgabe: wir bekommen ein Quantenbit in einem Zustand  $\alpha|0\rangle + \beta|1\rangle$  und sollen dieses kopieren. Dazu steht uns ein zweites Qubit als Ziel zur Verfügung. Wir nehmen an, es sei im Zustand  $|0\rangle$ . Wir beginnen mit dem Zustand

$$(\alpha|0\rangle + \beta|1\rangle) \cdot |0\rangle \quad (3.1)$$

und wir sollen den Zustand

$$(\alpha|0\rangle + \beta|1\rangle) \cdot (\alpha|0\rangle + \beta|1\rangle) \quad (3.2)$$

herstellen. Der Übergang zwischen (3.1) und (3.2) muss nach den Gesetzen der Quantenmechanik unitär sein.

In einem klassischen Schaltkreis ist es kein Problem, ein Bit zu kopieren. Wollen wir ein klassisches Bit *umkehrbar* kopieren, können wir die kontrollierte Negation verwenden:

$$\text{CNOT} : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus x\rangle.$$

So wird  $|00\rangle$  auf  $|00\rangle$  und  $|10\rangle$  auf  $|11\rangle$  abgebildet: das erste Bit wird auf das zweite kopiert. CNOT ist unitär, also ein Kandidat auch für das Kopieren eines Quantenbits.

Wir wenden diese Operation probeweise auf den Ausgangszustand

$$\alpha|00\rangle + \beta|10\rangle$$

an. Das Ergebnis ist

$$\alpha|00\rangle + \beta|11\rangle. \quad (3.3)$$

Das ist allerdings nicht der gewünschte Zustand (3.2), der ausmultipliziert wie folgt lautet:

$$\alpha^2|00\rangle + \alpha\beta|01\rangle + \alpha\beta|10\rangle + \beta^2|11\rangle. \quad (3.4)$$

Wir sehen: es wurde nur korrekt kopiert, wenn  $\alpha\beta = 0$  gilt. Das ist für klassische Bits oder die entsprechenden Basiszustände der Fall. Der Zustand (3.3) - das Ergebnis der CNOT-Operation - entstand hingegen nach der Art des Quantenparallelismus; wie in dem Übergang

$$\sum_x \alpha_x |x\rangle |0\rangle \mapsto \sum_x \alpha_x |x\rangle |f(x)\rangle$$

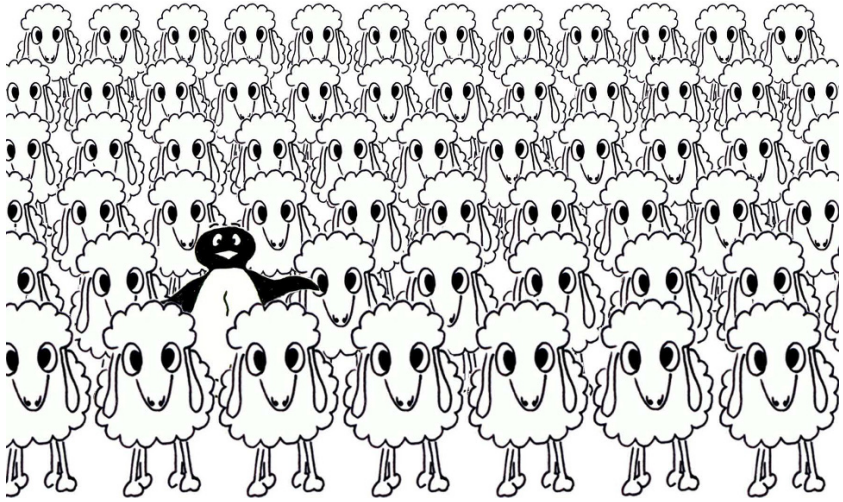


Abbildung 3.9: Quantenbits können nicht korrekt kopiert werden.

ist ein verschränkter Zustand entstanden (falls  $\alpha$  und  $\beta$  ungleich 0 sind).

CNOT eignet sich offenbar nicht zum Kopieren von Quantenbits. Es gilt sogar, dass kopieren überhaupt nicht möglich ist. Man nennt dieses Ergebnis das *No-Cloning-Theorem*, illustriert in Abbildung 3.9.

#### No-Cloning-Theorem

Wir betrachten ein Quantensystem im Zustand

$$|\psi\rangle \otimes |s\rangle,$$

wobei  $|\psi\rangle$  beliebig und  $|s\rangle$  beliebig aber fest gewählt ist. Es gibt keine unitäre Transformation, die dieses System für *jedes*  $|\psi\rangle$  in den Zustand

$$|\psi\rangle \otimes |\psi\rangle$$

versetzt.

Warum gibt es kein Verfahren, dass zwei verschiedene Zustände  $|\phi\rangle$  und  $|\psi\rangle$  auf ein Zielbit im Zustand  $|s\rangle$  kopiert? Zunächst stellen wir fest, dass diese Aufgabe nur von einer unitären Transformation übernommen werden kann. Wir nehmen an, es gäbe eine unitäre Transformation  $U$ , die

$$|\psi\rangle \otimes |s\rangle \text{ auf } |\psi\rangle \otimes |\psi\rangle$$

und

$$|\phi\rangle \otimes |s\rangle \text{ auf } |\phi\rangle \otimes |\phi\rangle$$

abbildet. Dann müssen das Skalarprodukt der Urbilder und das der Bilder übereinstimmen (denn unitäre Transformationen sind winkelerhaltend:  $\langle Uu|Uv \rangle = \langle u|v \rangle$ , siehe Abschnitt 2.3). Es gilt

$$\langle \psi \otimes s | \phi \otimes s \rangle = \langle \psi \otimes \psi | \phi \otimes \phi \rangle.$$

Da Skalarprodukt und Tensorprodukt verträglich sind (siehe Seite 43) folgt:

$$\langle \psi | \phi \rangle \langle s | s \rangle = \langle \psi | \phi \rangle \langle \psi | \phi \rangle$$

Für jeden Einheitsvektor  $|s\rangle$  gilt  $\langle s | s \rangle = 1$  und damit

$$\langle \psi | \phi \rangle = (\langle \psi | \phi \rangle)^2.$$

Diese Gleichung hat die Lösungen 0 und 1. Das heißt,  $U$  kann  $\phi$  und  $\psi$  nur dann kopieren, wenn  $\phi = \psi$  gilt oder  $\phi$  und  $\psi$  orthogonal sind.

Zwei verschiedene Zustände können von einem Quantenschaltkreis nur dann kopiert werden, wenn sie orthogonal sind.

Angenommen, wir wollen ein Quantenbit kopieren. Wir wissen genau, welche Zustände es annehmen kann, es sind dies  $|\phi\rangle$  und  $|\psi\rangle$ . Auch mit diesem Wissen können wir einen auf  $|\phi\rangle$  und  $|\psi\rangle$  spezialisierten Quantenkopierer nur dann bauen, wenn  $|\phi\rangle$  und  $|\psi\rangle$  orthogonal sind.

Damit ist auch erklärt, wie das Klonverbot mit der Erfahrung verträglich ist, dass wir klassische Zustände sehr wohl kopieren können: solche sind bezüglich jeder beliebigen Basis orthogonal. Nun stellt sich die Frage: kann ich zwei nicht orthogonale Zustände vielleicht *annähernd* kopieren? Vielleicht genügt für Anwendungen eine Kopie, die nahe am Original liegt. Diese Fragestellung ist intensiv untersucht worden. Die Ergebnisse lauten dem Geiste nach: Quantenbits lassen sich nicht zufriedenstellend kopieren. Leser mit fortgeschrittener mathematischer Intuition spüren vielleicht schon, dass leichte Zugeständnisse an die Exaktheit der Kopie nur leichte Zugeständnisse an die Orthogonalität der kopierbaren Vektoren erlauben.

## 3.5 Umkehrbare Berechnungen

Quantencomputer haben Fähigkeiten, die klassische Rechner nicht besitzen. Zwei Stichworte sind Parallelität und Verschränkung. Nun gilt es, eine wichtige Eigenschaft von Quantenberechnungen zu untersuchen, die eine Einschränkung bedeuten könnte: die Rechenschritte entsprechen unitären Transformationen, und diese sind *umkehrbar*. Aus dem Ergebnis einer Berechnung lässt sich die Eingabe eindeutig berechnen, jeder Rechenschritt lässt sich rückgängig machen.

Für klassische Rechner gilt diese Einschränkung nicht. Zum Beispiel ist das logische Und (AND) zweier Bits  $(b_1, b_2)$  eine nicht umkehrbare Operation.

Die drei Eingaben  $(0,0)$ ,  $(0,1)$  und  $(1,0)$  werden allesamt auf 0 abgebildet. Ist das Ergebnis einer Verundung 0, können wir daraus nicht erschließen, welche der drei Situationen vorlag und wie also die Eingabe aussieht.

Das könnte ein ernstes Problem sein: womöglich gibt es Aufgaben, die sich mit klassischen Rechnern effizient lösen lassen, mit einem Quantencomputer hingegen nicht. Es wäre möglich, dass *umkehrbar* rechnen in vielen Fällen zu viel Speicherplatz und Rechenzeit benötigt.

Jede (klassische) Operation lässt sich umkehren, wenn wir uns die Eingabe merken. Eine umkehrbare Version von AND könnte die beiden Eingabebits  $(b_1, b_2)$  auf drei Ausgabebits abbilden, nämlich auf  $(b_1 \wedge b_2, b_1, b_2)$ . Die Frage lautet: Wie wirkt sich eine solche Umformung in einem komplexen Schaltkreis aus? Falls wir uns alle Zwischenergebnisse merken müssen, wird dann der Schaltkreis nicht zu groß?

Dieser Abschnitt zeigt, dass jede effiziente Berechnung in eine effiziente *umkehrbare* Berechnung umgeformt werden kann. Daraus werden wir folgern, dass Quantenrechner trotz der Unitarität der Rechenschritte klassische Rechner *effizient simulieren* können (siehe Abschnitt 3.2 zu diesem Begriff). Das sagt natürlich nichts über die Fälle, in denen Quantenrechner klassischen Computern echt überlegen sind!

## Ziel

Ein Problem, das mit einem klassischen Schaltkreis der Größe  $m$  berechenbar ist, lässt sich von einem Quantenschaltkreis der Größe  $O(m)$  berechnen.

Wir erreichen unser Ziel in *zwei Schritten*.

*Schritt 1:* Wir zeigen: Jeder klassische Schaltkreis mit umkehrbaren Gattern kann als Quantenschaltkreis aufgefasst werden.

*Schritt 2:* Danach formen wir einen beliebigen klassischen Schaltkreis so um, dass jeder Rechenschritt umkehrbar wird und die Größe höchstens linear wächst.

*Bemerkung:* Wir betrachten im gesamten Kapitel 3 nur *deterministische* Berechnungen. Im Abschnitt 4.2 werden wir Berechnungen untersuchen, in denen Zufallsschritte verwendet werden. Dort werden wir das Ergebnis dieses Abschnittes verallgemeinern: auch jede *randomisierte* klassische Berechnung kann in eine Quantenrechnung überführt werden.

## Umkehrbar heißt unitär

### Schritt 1

Wie genau sieht eine umkehrbare – oder anglophon *reversible* – klassische Berechnung aus? Jeder Rechenschritt führt in Abhängigkeit von dem aktuellen Zustand zu einem Folgezustand. *Umkehrbar* bedeutet, ein Folgezustand darf nicht von zwei verschiedenen Zuständen aus erreicht werden. In Abbildung 3.10 sind einige Übergänge zwischen den Zuständen eines einzelnen Bits dargestellt. Der letzte Schritt, in dem 0 und 1 den gleichen Nachfolgezustand 0 haben, ist nicht umkehrbar. Da Rechenschritte Abbildungen auf der Menge der möglichen Zustände sind, können wir im Folgenden den Begriff *bijektive Abbildung* nutzen.



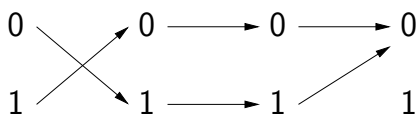


Abbildung 3.10: Der letzte Schritt der Berechnung ist nicht umkehrbar

Rechnen wir mit  $n$  Bits, so gibt es  $2^n$  mögliche Zustände. Wir verwenden die Quantenschreibweise  $|0\rangle, \dots, |2^n - 1\rangle$ : nicht überlagerte Quantenzustände (in der Standardbasis) entsprechen gerade den klassischen Zuständen. Ein Rechenschritt ist umkehrbar, wenn er eine bijektive Abbildung der Menge  $\{|0\rangle, \dots, |2^n - 1\rangle\}$  auf sich selbst ist. Oder anders ausgedrückt, wenn diese Abbildung eine *Permutation* ist, siehe Abschnitt 2.5.

Klassische  
Schritte und  
Quantenregister

**Beispiel 3.4:** Wir erzeugen eine reversible Variante des exklusiven Oder, indem wir uns eines der Eingabebits merken:

$$(b_1, b_2) \mapsto (b_1, b_1 \oplus b_2).$$

Wenn beide Bits 0 sind, ändert sich nichts. Wenn das erste Bit 1 ist, ergibt sich der Folgezustand mit unverändertem ersten und gekipptem zweiten Bit. Die einzelnen Zustandsübergänge lauten

$$\begin{array}{ll} 00 & \mapsto 00 \\ 01 & \mapsto 01 \\ 10 & \mapsto 11 \\ 11 & \mapsto 10 \end{array}$$

Jedes Urbild kommt genau einmal als Bild vor. Wir können erkennen, falls wir uns im Zustand 11 befinden, war der Zustand vor der Operation 10. Es handelt sich um eine Permutation der vier Zustände der beiden (klassischen) Bits; diese Transformation ist eine alte Bekannte namens CNOT.  $\diamond$

In unserem Schaltkreismodell entspricht ein Rechenschritt einem Gatter, an dem nur konstant viele Bits beteiligt sind. Damit gilt:

Ein Schaltkreis führt genau dann eine umkehrbare Rechnung aus, wenn jedes Gatter eine Permutation der Zustände der beteiligten Bits ist.

Wir haben bereits gesehen, dass eine Permutation auf einem Vektorraum als unitäre Transformation wirkt (siehe Aufgabe 2.9). Wir folgern: Eine klassische Berechnung, in der jeder Schritt umkehrbar ist, taugt auch für Quantenberechnungen. Umkehrbare klassische Schritte sind in der Quantenwelt unitäre Transformationen.

Jede klassische (deterministische) umkehrbare Berechnung lässt sich als Quantenrechnung ausführen.

### Klassische Berechnungen umkehrbar gestalten

#### Schritt 2

Im zweiten Schritt formen wir einen beliebigen klassischen Schaltkreis in einen umkehrbaren um. An dem entstehenden Schaltkreis interessieren uns zwei Aspekte:

1. Jeder Rechenschritt – jedes Gatter also – ist umkehrbar, und für eine klassische Eingabe ergibt sich das gleiche Resultat wie im Falle des Ausgangsschaltkreises.
2. In Schritt 1 haben wir gesehen, dass umkehrbare klassische Berechnungen unitären Transformationen entsprechen. Somit sind sie im Prinzip als Quantenberechnungen ausführbar. Da wir auf theoretischer Ebene argumentieren, sind umkehrbare klassische Schaltkreise für uns Quantenschaltkreise. Für klassische – nichtüberlagerte – Eingaben ergibt sich das Ergebnis des klassischen Falls. Zu welchem Ergebnis eine Superposition als Eingabe führt, sehen wir später.

Wir beginnen mit einer Definition.

Ein Gatter  $G$  heißt *universell*, wenn jede Funktion  $\{0,1\}^n \rightarrow \{0,1\}^m$  mit einem Schaltkreis berechenbar ist, der nur Gatter  $G$  verwendet.

Mit einem universellen Gatter lassen sich also alle anderen Gatter ersetzen. Kann es so etwas geben? Ja; und um das einzusehen, überlegen wir uns: Jede Funktion mit Werten aus  $\{0,1\}$  lässt sich als Formel ausdrücken, die nur  $\wedge, \vee$  und die Negation verwendet.

**Beispiel 3.5:** Wir betrachten die Funktion  $f : \{0,1\}^3 \rightarrow \{0,1\}$ , die die Eingaben  $x=0, y=1, z=0$  und  $x=1, y=1, z=1$  auf 1 abbildet und alle anderen Eingaben auf 0. Dann gilt:

$$f(x,y,z) = (\bar{x} \wedge y \wedge \bar{z}) \vee (x \wedge y \wedge z).$$

Jede Funktion lässt sich nach dem Prinzip von Beispiel 3.5 ausdrücken, man spricht von der *disjunktiven Normalform*. Aus dieser Darstellung ergibt sich sofort ein Schaltkreis für  $f$ , der die Gatter AND, OR, NOT verwendet.

**Aufgabe 3.4:** Geben Sie einen Schaltkreis für die Funktion aus Beispiel 3.5 an.

### Das Toffoli-Gatter

Wir geben ein umkehrbares Gatter an, das universell ist: wir können die Operationen AND, OR, NOT damit ersetzen. Es hat drei Eingänge  $a, b, c$  und drei Ausgänge, die mit  $a, b$  und  $c \oplus (a \wedge b)$  belegt sind, wie in Abbildung 3.11.

Dieses Gatter heißt *Toffoli-Gatter*, benannt nach Tommaso Toffoli, der es 1981 veröffentlicht hat. Eine Anekdote dazu: Die Grundidee wurde bereits 1967 von Carl Adam Petri veröffentlicht, allerdings in deutscher Sprache und in einem wenig verbreiteten Tagungsband. So konnte sie mehr als ein Jahrzehnt später von Toffoli neu entdeckt werden.

*Das Toffoli-Gatter ist universell:* Wir können mit dem Toffoli-Gatter ein Bit negieren, wenn wir zwei andere Bits konstant setzen: Gilt  $b = 1$  und  $c = 1$  ist das dritte Ausgabebit

$$1 \oplus (a \wedge 1) = \bar{a}.$$

Für  $c = 0$  ist dieses Bit

$$0 \oplus (a \wedge b) = a \wedge b.$$

Abbildung 3.12 zeigt diese Konstruktion. Die Disjunktion ergibt sich nicht ganz so leicht. Um ein OR-Gatter zu ersetzen, benötigen wir drei Toffoli-Gatter.

**Aufgabe 3.5:** Wie lässt sich mit dem Toffoli-Gatter die Disjunktion  $a \vee b$  berechnen?

So wie wir AND und NOT mit einem und OR mit drei Toffoli-Gattern berechnen können, lässt sich jedes klassische Gatter ersetzen. Sehr direkt ist dies bei XOR möglich:

$$c \oplus (a \wedge 1) = a \oplus c.$$

Es gilt allgemein:

Wir können jedes Gatter für eine klassische logische Operation mit zwei Eingabebits und einem Ausgabebit durch konstant viele Toffoli-Gatter ersetzen.

*Das Toffoli-Gatter ist umkehrbar:* es bildet die Eingabe  $(a, b, c) = (1, 1, 0)$  auf  $(1, 1, 1)$  ab und umgekehrt,  $(1, 1, 1)$  auf  $(1, 1, 0)$ . Alle anderen Eingabe-

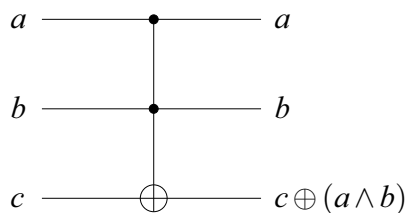


Abbildung 3.11: Das Toffoli-Gatter

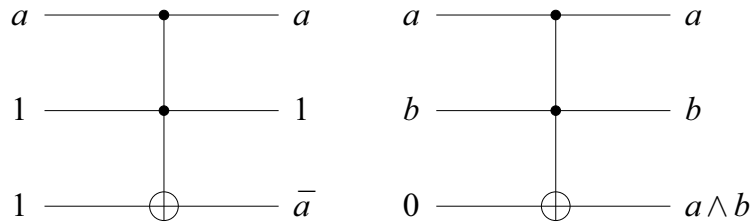


Abbildung 3.12: Negation und Konjunktion klassischer Bits mit einem Toffoli-Gatter

ben bleiben unverändert. Somit ist aus dem Ergebnis die Eingabe immer rekonstruierbar. In Abbildung 3.11 ist auch direkt erkennbar, dass zweifache Anwendung des Gatters die Identität ergibt. Damit führt das Toffoli-Gatter wie jede umkehrbare klassische Berechnung auf dem Raum der Zustandsvektoren eine unitäre Transformation aus.

Das Toffoli-Gatter kann als eine Verallgemeinerung der Operation CNOT betrachtet werden. Das spiegeln die Symbole für die Gatter wieder, siehe Abbildung 3.5. Eine Alternative zu dem Toffoli-Gatter stammt von Edward Fredkin:

**Aufgabe 3.6:** Untersuchen Sie das *Fredkin-Gatter*. Es hat drei Eingänge  $a, b, c$  und drei Ausgänge  $a', b', c'$ . Es gilt

1.  $a' = b$  und  $b' = a$ , falls  $c = 1$ ,
2.  $a' = a$  und  $b' = b$ , falls  $c = 0$ ,
3.  $c' = c$  für alle Eingaben.

Überlegen Sie, warum dieses Gatter für reversible Berechnungen ebenso gut geeignet ist wie das Toffoli-Gatter.  $\diamond$

### Schaltkreise aus Toffoli-Gattern

#### Verzweigungen

Nun können wir Schritt 2 abschließen, indem wir in einem Schaltkreis alle Gatter durch das umkehrbare Toffoli-Gatter ersetzen. Beachten müssen wir allerdings eines: Gatter klassischer Schaltkreise haben im Allgemeinen *mehrere Ausgänge* und jede Leitung kann sich *verzweigen*. Das entspricht dem Kopieren eines Bits. Auch das müssen wir als unitäre Transformation realisieren. Möglich ist dies wiederum mit dem Toffoli-Gatter: Setzen wir etwa  $b = 1$  und  $c = 0$ , wird das erste Bit auf das dritte kopiert, Abbildung 3.13 (dazu ist auch das CNOT-Gatter geeignet, siehe Abbildung 3.5).

In Abschnitt 3.4 haben wir festgestellt: Quantenbits lassen sich nicht kopieren. Aber unser Ziel ist zunächst ein umkehrbarer *klassischer* Schaltkreis. Was geschieht, wenn wir einen solchen als Quantenschaltkreis auffassen, betrachten wir am Ende des Abschnitts.

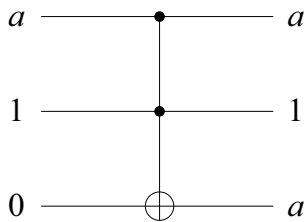


Abbildung 3.13: Ein klassisches Bit mit dem Toffoli-Gatter kopieren

Abbildung 3.14 zeigt rechts eine reversible Variante des Halbaddierers links (siehe Abschnitt 2.1). Es werden zwei Toffoli-Gatter verwendet, und es sind zwei Hilfsbits nötig (die reversible Variante hat vier Ausgänge). Die Verzweigungen müssen nicht wie in Abbildung 3.13 umgesetzt werden, da das Ergebnis des ersten Gatters auf ein Hilfsbit kopiert wird und die Bits  $a$  und  $b$  vom ersten Gatter nicht verändert werden.

Um einzusehen, dass jeder beliebige Schaltkreis auf diese Weise umkehrbar gemacht werden kann, überlegen wir uns ein Vorgehen, der diese Aufgabe nach Kochrezept ausführt.

Schaltkreise  
umkehrbar  
machen

Durchlaufe, an den Eingängen beginnend, den Schaltkreis. Ersetze jedes Gatter und jede Verzweigung wie oben beschrieben durch Toffoli-Gatter.

Im Falle des Halbaddierers (Abbildung 3.14) ist das Ergebnis der Schaltkreis in Abbildung 3.15. Zunächst werden die beiden Verzweigungen durch je ein Toffoli-Gatter und je zwei Hilfsbits ersetzt. Das XOR-Gatter wird folgendermaßen ersetzt: da wir die Gatter lokal ersetzen, überprüfen wir nicht, ob die Bits  $a$  und  $b$  überschrieben werden dürfen. Wir verwenden ein

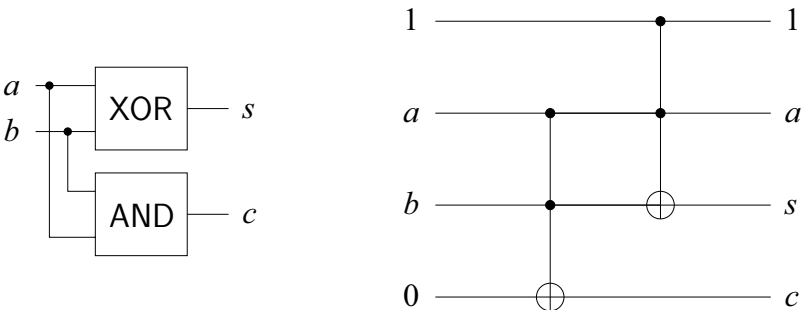


Abbildung 3.14: Halbaddierer und reversibler Halbaddierer aus zwei Toffoli-Gattern

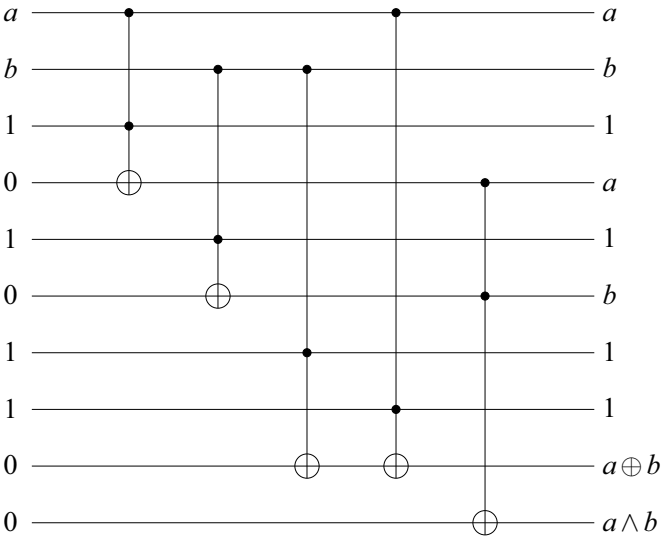


Abbildung 3.15: Reversibler Halbaddierer, schrittweise erzeugt

zusätzliches Ausgabebit, auf das wir zunächst  $b$  kopieren. Zuletzt ersetzen wir die Konjunktion.

Ergebnis

Das eben beschriebene Verfahren überführt jeden Schaltkreis in einen umkehrbaren. Berechnet der Ausgangsschaltkreis die Funktion

$$x \mapsto f(x),$$

so gibt es eine Lösung, die drei Register verwendet und

$$|x\rangle|h\rangle|0\rangle \text{ auf } |x\rangle|h^x\rangle|f(x)\rangle$$

abbildet.  $|x\rangle$  ist die Eingabe,  $|h\rangle$  enthält konstant gesetzte Hilfsbits, und das dritte Register enthält am Schluss die Lösung. Entscheidend ist, dass die Anzahl der Gatter nur linear wächst; und ebenso die Anzahl der Bits. Das ergibt sich daraus, dass wir jedes Gatter genau einmal durch konstant viele Toffoli-Gatter ersetzen. Dieses Ersetzen ist lokal; die Gesamtstruktur des Schaltkreises wird nicht berührt. Neu eingeführte Hilfsbits spielen für die anderen Gatter keine Rolle.

**Aufgabe 3.7:** Betrachten Sie die Berechnung des Schaltkreises in Abbildung 3.15, wenn  $a$  und  $b$  (beziehungsweise  $|a\rangle$  und  $|b\rangle$ ) jeweils mit  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  belegt sind.

Löschen von Zwischenergebnissen

Einige der Hilfsbits des Schaltkreises in Abbildung 3.15 werden in Abhängigkeit von der Eingabe  $x$  (dort  $a, b$ ) verändert, das ist durch den oberen Index in  $|h^x\rangle$  angedeutet. Das kann in konkreten Anwendungen unangenehme Auswirkungen haben. Es gibt eine Möglichkeit, das zu verhindern.

Grob gesagt wird nach Beendigung der Berechnung das Ergebnis mit Hilfe von CNOT-Gattern auf Ausgabebits kopiert, die an der Berechnung bis dahin nicht beteiligt waren. Anschließend wird jeder der Rechenschritte rückgängig gemacht. Alle Bits außer den Ergebnisbits befinden sich dann im gleichen Zustand wie zu Beginn der Berechnung.

*Zusammenfassung:* Wir gehen den Weg zurück:

1. Wir können in einem klassischen Schaltkreis jedes Gatter durch konstant viele Toffoli-Gatter ersetzen.
2. Das Toffoli-Gatter ist umkehrbar.
3. Jede umkehrbare Operation ist eine Permutation der Eingabebits und damit unitär.

So können wir jeden Schaltkreis in einen Quantenschaltkreis überführen, wobei die Anzahl der Gatter höchstens um einen konstanten Faktor wächst. Auf diese Weise lässt sich feststellen:

- Sind wir aus Gründen der Miniaturisierung dazu gezwungen, bei unseren Berechnungen die Gesetze der Quantenmechanik zu beachten, entstehen daraus keine theoretischen Probleme: Eine klassische Berechnung kann mit geringem Aufwand in eine Folge umkehrbarer Schritte überführt werden.
- Wir wollen einen Quantenschaltkreis konstruieren, der eine Funktion  $f$  verwendet.  $f$  ist mit einem effizienten klassischen Schaltkreis berechenbar. Daraus folgt mit dem Ergebnis dieses Abschnitts:  $f$  ist mit einem Quantenschaltkreis effizient berechenbar, der eine Eingabe

$$|x\rangle|0\rangle \text{ auf } |x\rangle|f(x)\rangle$$

abbildet (und eventuell weitere Hilfsbits verwendet). Eine Superposition

$$\sum_x \alpha_x |x\rangle |0\rangle$$

wird auf den verschränkten Zustand

$$\sum_x \alpha_x |x\rangle |f(x)\rangle$$

abgebildet.

**Aufgabe 3.8:** Wir belegen die Eingabebits des Schaltkreises in Abbildung 3.14 mit  $|a\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  und  $|b\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ . Was ist das Ergebnis der Berechnung?

Reversible Berechnungen wurden zu einem Zeitpunkt untersucht, als man noch nicht an Quantenrechner dachte. Warum? Wir stellen uns vor, wir befänden uns gerade auf einem Spaziergang. Den bisher zurückgelegten Weg können wir in umgekehrter Richtung im Prinzip zurückgehen (wir lassen Spitzfindigkeiten wie Tore, die nur in eine Richtung geöffnet werden können,

Informations-  
verlust

beiseite). Wenn wir jedoch mit dem Kopf vor einen Ast laufen und einen plötzlichen Gedächtnisverlust erleiden, finden wir eventuell nicht mehr nach Hause. So können wir uns viele Situationen denken, in denen das *Vernichten von Information* die Wiederherstellung eines früheren Zustandes unmöglich macht. Der amerikanische Physiker Rolf Landauer hat sich mit diesem Thema intensiv auseinander gesetzt. *Information is physical* ist eine für uns interessante Prämisse seiner Arbeit. Es scheint auf der Hand zu liegen, dass wir Information nur in physikalischer Form speichern oder kommunizieren können. Daraus entsteht die Frage: Was sind die physikalischen Konsequenzen einer Informationsvernichtung? Vereinfacht gesagt: Es wird Energie frei, in der Regel Wärme! *Landauers Prinzip* (1961) gibt an, welche Energiemenge beim Löschen eines Bits mindestens abgegeben wird, oder thermodynamisch ausgedrückt: auf welche Weise sich die der Umgebung mindestens erhöht, wenn ein Bit Information vernichtet wird.

Der Umkehrschluss dazu lautet: ein Computer, der nie ein Bit löscht, erwärmt sich (im Prinzip) nicht. Die freigesetzte Wärme war im Rechnerbau immer ein Problem, gerade angesichts der permanenten Miniaturisierung. Darum wurde schon sehr früh darüber nachgedacht, ob sich jede Berechnung im Prinzip umkehrbar gestalten lässt.

Eingeschränkte  
Berechnungen

Das Ergebnis dieses Abschnitts ist nicht selbstverständlich. Es gibt Berechnungsmodelle, für die die Quantenversion nicht alles klassisch mögliche bewältigen kann. Wir haben lediglich gezeigt: klassische Schaltkreise können in Quantenschaltkreise umgewandelt werden, die unwesentlich (linear) größer sind.

QOBDDs (*quantum ordered binary decision diagrams*) beschreiben Quantenalgorithmen, die jedes Bit der Eingabe höchstens einmal lesen. Außerdem ist die Reihenfolge vorgegeben, in der die Bits abgefragt werden, und neben der Eingabe der Größe  $n$  dürfen nur  $\log n$  Hilfsbits verwendet werden. Diese Einschränkungen mögen willkürlich wirken: klassische OBDDs spielen eine zentrale Rolle im Schaltkreisentwurf und verwandten Anwendungen.

Man kann zeigen: Es gibt Aufgaben, die mit klassischen OBDDs effizient lösbar sind, von Quanten-OBDDs jedoch exponentiellen Aufwand erfordern (siehe [130]). Ähnliches gilt für Versionen von *endlichen Quantenautomaten* (siehe [95]).

### 3.6 Unterscheidbare Zustände

Den Zustand eines klassischen Bits können wir jederzeit problemlos feststellen. Bei Quantenbits geht das nicht: ist ein solches etwa im Zustand  $\alpha|0\rangle + \beta|1\rangle$ , können wir die genauen Werte von  $\alpha$  und  $\beta$  nicht exakt bestimmen (siehe Abschnitt 2.8). Es gilt sogar: Nur unter bestimmten Voraussetzungen können wir feststellen, ob zwei Quantenbits in demselben Zustand sind oder sich unterscheiden.

Ein Quantenbit kann unendlich viele verschiedene Zustände annehmen. So fein wir ein Messinstrument bauen, man kann immer zwei Zustandsvektoren angeben, die zu nahe beieinanderliegen, als dass unser Instrument sie



unterscheiden könnte. Aber es ist auch aus theoretischen Gründen nicht möglich!

In diesem Abschnitt sehen wir, dass wir zwei Quantenzustände durch Messungen nur dann unterscheiden können, wenn die Zustandsvektoren orthogonal sind. Von den unendlich vielen Zuständen eines Quantenregisters lassen sich demnach nur endlich viele unterscheiden. Haben wir ein Register aus  $n$  Quantenbits, sind dies  $2^n$  verschiedene Zustände; wie bei einem klassischen Register.

Welche Rolle spielt die Unterscheidbarkeit von Quantenzuständen? Angenommen, eine Berechnung hat fünf mögliche Ergebnisse (ein Beispiel: fünf Personen spielen ein kompliziertes Gesellschaftsspiel, und ein Quantenrechner ermittelt den Spieler mit den meisten Punkten). Jedem der Ergebnisse ist ein Zustand  $|\phi_i\rangle, i = 1, \dots, 5$  eines Quantenregisters zugeordnet. Damit wir den Gewinner erfahren, müssen wir das Register messen. Wie wir oben erfahren haben, müssen die Zustände  $|\phi_1\rangle, \dots, |\phi_5\rangle$  paarweise orthogonal sein. Für das Ergebnisregister benötigen wir also mindestens drei Quantenbits: die Zustände eines Quantenregisters aus zwei Bits sind Vektoren eines lediglich vierdimensionalen Vektorraums.

Rolle der Unterscheidbarkeit

Wir sind im Besitz zweier Quantenbits und sollen entscheiden, ob sie in demselben Zustand sind. Betrachten wir zum Beispiel die beiden Zustände

Aufgabenstellung

$$|\phi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ und } |\psi_1\rangle = |0\rangle.$$

Wir messen beide Bits bezüglich der Standardbasis.  $|\psi_1\rangle$  liefert das Ergebnis  $|0\rangle$ . Bei der Messung von  $|\phi_1\rangle$  erhalten wir mit Wahrscheinlichkeit  $1/2$  das Resultat  $|1\rangle$ ; damit wissen wir: die Zustände waren verschieden (bevor unsere Messungen sie zerstört haben). Aber mit der gleichen Wahrscheinlichkeit liefert uns die Messung von  $|\phi_1\rangle$  das gleiche Ergebnis wie die an  $|\psi_1\rangle$ . Dann sind wir so schlau wie zuvor.

Sollen wir die gleiche Aufgabe für

$$|\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ und } |\psi_2\rangle = -\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

erledigen, können wir mit Messungen bezüglich der Standardbasis gar keinen Aufschluss darüber erlangen, ob die Zustände verschieden sind.

Aber vielleicht mit Hilfe der Messung bezüglich einer anderen Basis? Wir werden sehen, dass sich nicht feststellen lässt, ob  $|\phi_2\rangle$  und  $|\psi_2\rangle$  identisch oder verschieden sind.

**Aufgabe 3.9:** : Geben Sie ein Verfahren an, das die Zustände

$$|\phi_3\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ und } |\psi_3\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

unterscheidet.

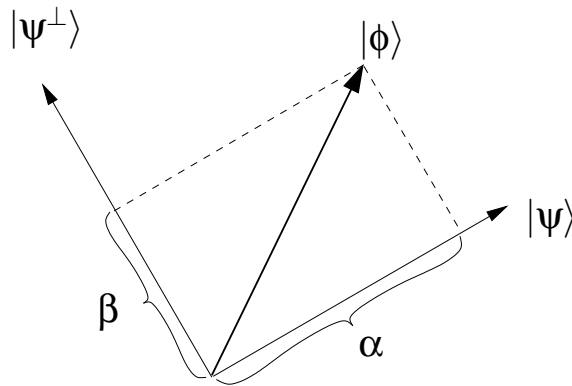


Abbildung 3.16: Geometrische Veranschaulichung von  $|\phi\rangle = \alpha|\psi\rangle + \beta|\psi^\perp\rangle$ .

Nun formulieren und veranschaulichen wir das Resultat dieses Abschnitts:

Unterscheidbare  
Zustände

Zwei Zustände lassen sich nur dann zweifelsfrei unterscheiden, wenn sie zueinander orthogonal sind.

Wir beweisen die Feststellung nicht formal, sondern veranschaulichen sie geometrisch. Wenn wir zwei Zustände  $|\phi\rangle$  und  $|\psi\rangle$  unterscheiden können, ist folgendes möglich: Uns wird ein Qubit  $|x\rangle$  übergeben. Es befindet sich im Zustand  $|\phi\rangle$  oder im Zustand  $|\psi\rangle$ ,  $|\phi\rangle \neq |\psi\rangle$ . Wir sollen entscheiden, welches der tatsächliche Zustand von  $|x\rangle$  ist.

Wir beginnen damit, eine aussichtsreiche Orthonormalbasis zu konstruieren. Wir wählen einen auf  $|\psi\rangle$  senkrecht stehenden Einheitsvektor  $|\psi^\perp\rangle$  und erhalten eine Orthonormalbasis

$$\{|\psi\rangle, |\psi^\perp\rangle\}.$$

Wir können  $|\phi\rangle$  als Linearkombination dieser Vektoren darstellen, siehe Abbildung 3.16:

$$|\phi\rangle = \alpha|\psi\rangle + \beta|\psi^\perp\rangle.$$

Sind  $|\phi\rangle$  und  $|\psi\rangle$  *orthogonal*, können wir unsere Aufgabe mit einer Messung bezüglich dieser Basis lösen: Ist  $|x\rangle = |\psi\rangle$  beobachten wir  $|\psi\rangle$ , ist  $|x\rangle = |\phi\rangle$  beobachten wir  $|\psi^\perp\rangle$ .

Wenn  $|\phi\rangle$  und  $|\psi\rangle$  *nicht orthogonal* sind, hat  $|\phi\rangle$  stets einen nicht verschwindenden  $|\psi\rangle$ -Anteil  $\alpha \neq 0$ . Eine Messung bezüglich der Basis  $\{|\psi\rangle, |\psi^\perp\rangle\}$  liefert für  $|x\rangle = |\psi\rangle$  stets das Ergebnis  $|\psi\rangle$  und für  $|\phi\rangle$  erhalten wir mit der positiven Wahrscheinlichkeit  $|\alpha|^2$  dasselbe Ergebnis. Wir können  $|\phi\rangle$  und  $|\psi\rangle$  also nicht mit Sicherheit unterscheiden. Sind sie *fast orthogonal*, ist die Unsicherheit eher gering; wenn sie weit davon entfernt sind orthogonal zu sein, ist die Unsicherheit sehr groß.

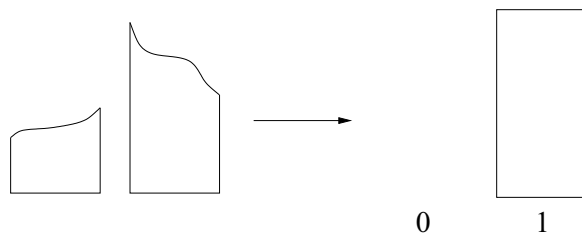


Abbildung 3.17: Ein klassisches Bit ist störungsresistent

Der von uns ausgelassene formale Beweis zeigt: Dieses Problem lässt sich nicht durch die Wahl einer anderen Basis beseitigen (siehe etwa [114]). Anschaulich gilt, dass der Wechsel zu einer anderen Orthonormalbasis als  $\{|\psi\rangle, |\psi^\perp\rangle\}$  die Unterscheidungskraft verringert.

Wir haben somit gesehen, dass nicht-orthogonale Zustände *prinzipiell* nicht unterscheidbar sind. Egal wie fein unsere Messinstrumente sind, es ist theoretisch unmöglich, nicht-orthogonale Zustände auseinanderzuhalten.

## 3.7 Gestörte Berechnungen

Ein Vorteil von klassischen Binärrechnern ist: sie sind ziemlich störungsresistent. Ein Bit hat nur zwei mögliche Zustände. Wir repräsentieren zum Beispiel 1 durch eine Spannung von 5 V und 0 durch keine Spannung. Liegt nun in einem physikalisch realisierten Schaltkreis eine Abweichung von diesem Ideal vor, angenommen wir haben eine Spannung von 4,3 V, können wir diese Abweichung leicht korrigieren, siehe Abbildung 3.17. Das liegt daran, dass 4,3 V nach unserer Festlegung kein gültiger Zustand ist. Wir können folgende fehlerrobuste Vereinbarung treffen: Der Bereich kleiner 2,5 V entspricht dem Bitzustand 0 und der größer gleich 2,5 V der 1.

Ein Quantenrechner ist kein Binärrechner in diesem Sinne. Ein Quantenbit kann nicht nur die Zustände  $|0\rangle$  und  $|1\rangle$  annehmen, sondern auch unendlich viele mögliche Überlagerungen davon. Ist nun ein Gatter eines Quantenschaltkreises gestört und wir erhalten statt des Folgezustands  $|\phi\rangle$  den Zustand  $|\phi'\rangle$  – siehe Abbildung 3.18 – gibt es keine so einfache Methode, diese Abweichung auszugleichen. Jeder Vektor der Länge 1 ist ein zulässiger Zustand. Wenn wir uns also über die Realisierung von Quantenrechnern Gedanken machen, müssen wir die Frage nach dem Umgang mit Fehlern besonders berücksichtigen. Dieser Abschnitt liefert dazu ein erstes befriedigendes Resultat:

In einer Folge von gestörten Quantengattern addiert sich der Fehler. Die Abweichung, die ein fehlerhaftes Gatter verursacht, wird in der weiteren Berechnung nicht verstärkt.

Ziel

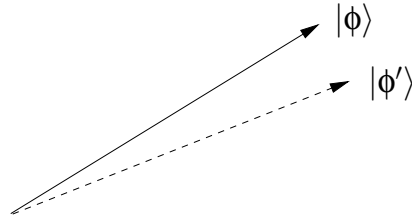


Abbildung 3.18: Der Zustand eines Quantenbits ist störanfällig

Wir stellen uns vor, wir hätten einen wichtigen Quantenalgorithmus zu realisieren. Wir haben diesen Algorithmus bereits in eine Folge von einfachen unitären Transformationen zerlegt, die wir mit unseren Mitteln ausführen können. Unser Algorithmus beginnt im Startzustand  $|\phi_0\rangle$ . Dann soll er die Transformationen  $U_1, \dots, U_T$  ausführen. Tatsächlich berechnen unsere Gatter Transformationen, die von den im Entwurf des Algorithmus verwendeten abweichen. Die Abweichung jedes einzelnen Gatters ist gering. Aber wie verhalten sie sich hintereinander geschaltet, wie in Abbildung 3.19?

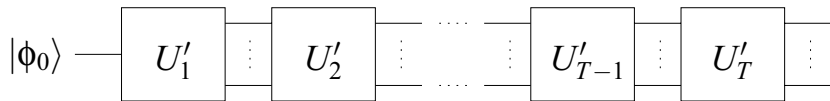


Abbildung 3.19: Der physikalisch realisierte Algorithmus

Es wäre möglich, dass sich der Fehler multipliziert, wie im folgenden Beispiel.

**Beispiel 3.6:** Wir betrachten die Operation  $N$ , die eine ganze Zahl negiert:  $N(z) = -z$ . Wir wenden  $N$  mehrmals auf 1 an.  $N(1) = -1, N^2(1) = N(N(1)) = 1$  und so weiter.

Die physikalische Ausführung  $N'$  weicht von  $N$  allerdings um 10 Prozent ab. 1 wird auf  $-1,1$  und allgemein  $z$  auf  $-(z + z/10)$  abgebildet. Wie wächst der Fehler bei mehrfacher Anwendung?

Bei zweifacher erhalten wir

$$N'(N'(1)) = N'(-1,1) = 1,1 \cdot 1,1 = 1,21.$$

Hätten sich die Fehler der beiden Rechnungen addiert, wäre das Ergebnis  $1 + 0,1 + 0,1 = 1,2$ . Auch wenn der Unterschied klein erscheint: der additive Fehler wächst linear in der Anzahl der Anwendungen, der multiplikative exponentiell.

Wenden wir  $N'$  zehnmal an, ist das Ergebnis  $1,1^{10} \approx 2,59$ , während das Ergebnis mit additivem Fehler 2 wäre. Führen wir  $N'$  hundertmal aus, erhalten wir  $1,1^{100} \approx 13780$ , wogegen eine Rechnung mit additivem Fehler das Ergebnis 11 erbrächte.  $\diamond$

Unsere Berechnung beginnt im Startzustand  $|\phi_0\rangle$ . Im ersten Schritt soll  $U_1|\phi_0\rangle$  realisiert werden. Das tatsächliche Ergebnis ist allerdings  $U'_1|\phi_0\rangle$ . Die Abweichung bestimmen wir als den Abstand der beiden Vektoren:

$$\|U_1|\phi_0\rangle - U'_1|\phi_0\rangle\| = \|(U_1 - U'_1)|\phi_0\rangle\|.$$

Wir definieren

$$|\phi_i\rangle = U_i \dots U_1 |\phi_0\rangle,$$

das Zwischenergebnis nach  $i$  ungestörten Schritten. Wenn nur das  $i$ -te Gatter gestört ist, beträgt die dort entstehende Abweichung damit

$$\|(U_i - U'_i)|\phi_{i-1}\rangle\|.$$

Nun untersuchen wir den Gesamtfehler: den Abstand zwischen den Berechnungen  $U_T \dots U_1 |\phi_0\rangle$  und  $U'_T \dots U'_1 |\phi_0\rangle$ .

Dazu betrachten wir die Berechnungen  $|\psi_i\rangle$ , bei der die ersten  $i$  Schritte korrekt ausgeführt werden und alle danach gestört sind:

$$\begin{aligned} |\psi_T\rangle &= U_T U_{T-1} \dots U_3 U_2 U_1 |\phi_0\rangle \\ |\psi_{T-1}\rangle &= U'_T U_{T-1} \dots U_3 U_2 U_1 |\phi_0\rangle \\ &\vdots \\ |\psi_2\rangle &= U'_T U'_T U'_{T-1} \dots U'_3 U'_2 U'_1 |\phi_0\rangle \\ |\psi_1\rangle &= U'_T U'_T U'_{T-1} \dots U'_3 U'_2 U'_1 |\phi_0\rangle \\ |\psi_0\rangle &= U'_T U'_T U'_{T-1} \dots U'_3 U'_2 U'_1 |\phi_0\rangle \end{aligned}$$

Damit ist der Gesamtfehler gerade

$$\| |\psi_T\rangle - |\psi_0\rangle \|.$$

Zur Analyse betrachten wir den Abstand zwischen zwei Berechnungen, die sich in genau einem Schritt unterscheiden:

$$\begin{aligned} \| |\psi_{i+1}\rangle - |\psi_i\rangle \| &= \| U'_T \dots U'_{i+2} (U_{i+1} - U'_{i+1}) U_i \dots U_1 |\phi_0\rangle \| \\ &= \| (U_{i+1} - U'_{i+1}) U_i \dots U_1 |\phi_0\rangle \| \\ &= \| (U_{i+1} - U'_{i+1}) |\phi_i\rangle \|. \end{aligned}$$

Das ist der entscheidende Schritt! Die Transformationen  $U'_T \dots U'_{i+1}$  sind unitär und damit normerhaltend! Der Fehler im  $(i+1)$ -ten Gatter – die Differenz  $(U_{i+1} - U'_{i+1})$  also – wird durch die weiteren fehlerhaften Gatter nicht vergrößert!

Der Rest ist jetzt eine einfache Rechnung. Für den Gesamtfehler gilt

$$\begin{aligned} \| |\psi_T\rangle - |\psi_0\rangle \| &= \| |\psi_T\rangle - |\psi_{T-1}\rangle + |\psi_{T-1}\rangle \pm \dots - |\psi_1\rangle + |\psi_1\rangle - |\psi_0\rangle \| \\ &\leq \| |\psi_T\rangle - |\psi_{T-1}\rangle \| + \dots + \| |\psi_1\rangle - |\psi_0\rangle \|. \end{aligned}$$

Das sieht man mit der Dreiecksungleichung (Abschnitt A.2.3); der Gesamtfehler ist somit kleiner gleich der Summe der Einzelfehler:

$$\| |\psi_T\rangle - |\psi_0\rangle \| \leq \| (U_1 - U'_1) |\phi_0\rangle \| + \dots + \| (U_T - U'_T) |\phi_{T-1}\rangle \|.$$

Praktisch bedeutet das: Wenn wir an irgendeiner Stelle einer Rechnung mit einem gestörten Gatter konfrontiert werden, können wir mögliche Abweichungen in der bisherigen Rechnung ignorieren. Die neue Abweichung vom Ideal addiert sich zu den bisherigen. Dieses Ergebnis lässt sich auch veranschaulichen. Unitäre Transformationen sind winkelerhaltend, wirken also auf Zustandsvektoren ähnlich wie Drehungen oder Spiegelungen. Drehen wir nun in einem Rechenschritt um einen vom Ideal abweichenden Winkel, wirkt sich diese Abweichung auf die nächste Drehung nicht aus.

Fehlerkorrigierende  
Codes

Am Anfang dieses Abschnitts haben wir argumentiert, dass sich Rechnungen eines klassischen Rechners Fehlern gegenüber ziemlich robust verhalten. Aber trotzdem kann es auch hier zu Störungen kommen. Besonders anschaulich, wenn Daten über eine Leitung verschickt werden, die defekt ist oder Einflüssen wie etwa Magnetfeldern ausgesetzt ist. Um mit solchen Situationen fertig zu werden, verwendet man *fehlerkorrigierende Codes*. Man verlängert die Nachricht um einige Bits. Diese enthalten Zusatzinformationen, mit denen im Falle einer Störung die ursprüngliche Nachricht wiederhergestellt werden kann.

Die Frage, wie mit Fehlern umzugehen sei, stellt sich bei Quantenrechnern noch dringlicher als bei klassischen. Quantenbits können unendlich viele Zustände annehmen. Dazu kommt das Problem der *Dekohärenz*, siehe Abschnitt 9.1. Zudem stellt uns das No-Cloning-Theorem vor gewisse Schwierigkeiten. Eine einfache Idee zur Fehlerkorrektur lautet: schicke jedes Bit mehrmals, um zu prüfen, ob am Zielort Abweichungen auftreten. Um ein Quantenbit mehrfach zu schicken, müssten wir es kopieren. Und das ist nicht möglich! Peter Shor war der erste, der 1995 einen fehlerkorrigierenden Quantencode veröffentlicht hat. Mit der Existenz solcher Codes ist eine notwendige Bedingung für den Bau praktisch verwendbarer Quantenrechner erfüllt. Kapitel 9 führt in dieses Thema ein.

## 4 Hilfsmittel aus der Theoretischen Informatik

*Nicht zu sagen Hypothese, noch weniger Theorie, sondern VORSTELLUNGS-ART.*  
Georg Christoph Lichtenberg

Bei welchen Aufgaben sind Quantencomputer klassischen Rechnern überlegen? Gibt es Probleme, die für letztere *schwer*, für erstere aber *einfach* sind? Dieses Kapitel führt in aller Kürze in die Komplexitätstheorie ein, die danach fragt, welche Probleme effizient lösbar sind. Diese Theorie hilft uns auch dabei, über die Grenzen des Quantencomputers nachzudenken. Die Betrachtung schließt direkt an die Abschnitte 3.1 und 3.2 an.

Abschnitt 4.2 beschäftigt sich mit randomisierten Algorithmen. Diese nutzen Zufallsprozesse und dürfen sich in einem gewissen Rahmen irren. Für unsere Zwecke ist es wichtig, den Nutzen solcher Verfahren zu verstehen und die Rolle des Fehlers einschätzen zu können. Denn auch Quantenalgorithmen enthalten ein randomisiertes Element: die Messung.

### 4.1 Komplexitätsklassen

In Kapitel 2 haben wir den Begriff Berechnung anhand der ganzzahligen Addition erläutert. Wie jedes *Berechnungsproblem* ist es durch die Eingabe und die Ausgabe definiert. Erstere besteht in diesem Fall aus zwei ganzen Zahlen  $a$ ,  $b$  und die Ausgabe ist deren Summe  $a + b$ . Addition

Wie aufwändig ist es, zwei Zahlen zu addieren? Wie viele Rechenschritte benötigt ein Algorithmus, der diese Aufgabe für alle möglichen Eingaben löst? Das ist die Frage nach der dem Problem *innewohnenden Schwierigkeit* oder nach der *inhärenten Komplexität*.

Wir bestimmen zunächst die Eingabegröße des Problems *Addition*. Wir gehen davon aus, dass die Zahlen  $a$  und  $b$  in Binärdarstellung vorliegen. Dazu

sind etwa  $\log_2 a + \log_2 b$  Bits nötig. Die Eingabegröße  $m$  ist damit von der Größenordnung  $O(\log a)$ , wenn  $a$  die größere Zahl ist.

In Kapitel 2 haben wir das Additionsverfahren betrachtet, das Schüler in der Grundschule lernen. Beide Ziffernfolgen werden von rechts nach links durchlaufen, und je zwei *Stellen* werden unter Berücksichtigung des Übertrags addiert. Wir können uns leicht überlegen, dass dieses Verfahren für jede Stelle konstant viele Schritte ausführen muss. Die Laufzeit ist linear; also von der Größenordnung  $O(m) = O(\log a)$ .

Zudem kann es kein Verfahren geben, das der Größenordnung nach schneller ist. Für die Summe spielt jede Ziffer der beiden Zahlen  $a$  und  $b$  eine Rolle, und alle müssen einmal angesehen werden. Die Laufzeit ist also mindestens  $\Omega(m)$ . Damit haben wir die Laufzeit-Komplexität der Addition hinreichend genau beschrieben: sie beträgt  $\Theta(m)$ .

Nicht immer lässt sich die Komplexität so einfach bestimmen. Gerade bei Problemen, für die kein schneller Algorithmus bekannt ist, ist es oft schwierig, den Ressourcenbedarf exakt zu bestimmen. Hilfreich erweist es sich, Berechnungsprobleme anhand ihrer Schwierigkeit vergrößernd einzuteilen, in *Komplexitätsklassen*.

Entscheidungs-  
probleme

Hierfür werden wir uns in der Folge auf *Entscheidungsprobleme* beschränken. Das sind Berechnungsprobleme, deren Ergebnis entweder 0 oder 1 ist.

**Beispiel 4.1:** Die folgenden Berechnungsprobleme sind Entscheidungsprobleme:

1. *Eingabe:* Eine ganze Zahl  $n$

*Ausgabe:*

$\text{PRIMES}(n) = 1$ , falls  $n$  eine Primzahl ist,

$\text{PRIMES}(n) = 0$ , sonst.

2. *Eingabe:* Drei ganze Zahlen  $a, b, n$

*Ausgabe:*

$\text{MULT}(a, b, n) = 1$ , falls  $n = a \cdot b$ ,

$\text{MULT}(a, b, n) = 0$ , sonst.

3. *Eingabe:* Ein Feld von ganzen Zahlen  $a_1, \dots, a_n$ , eine Schranke  $k$

*Ausgabe:*

$\text{MINIMUM}(a_1, \dots, a_n, k) = 1$ , falls das Feld ein Element kleiner gleich  $k$  enthält

$\text{MINIMUM}(a_1, \dots, a_n, k) = 0$ , sonst.

Zum Beispiel gilt  $\text{PRIMES}(7) = 1$ ,  $\text{PRIMES}(9) = 0$ . Wir können auch die entgegengesetzte Eigenschaft testen, nämlich ob eine ganze Zahl zusammengesetzt ist; dann gilt  $\text{COMPOSITES}(7) = 0$ ,  $\text{COMPOSITES}(9) = 1$ .

$\text{MULT}$  und  $\text{MINIMUM}$  sind *Entscheidungsvarianten* von Problemen, deren Ausgabe eigentlich nicht mit 0 oder 1 beziehungsweise *Ja* oder *Nein* auskommt. So ist im Falle von  $\text{MINIMUM}$  nicht das tatsächliche Minimum zu bestimmen, sondern nur die Frage zu beantworten, ob es die Schranke  $k$  nicht überschreitet. Auf ähnliche Weise lässt sich vielen Optimierungsproblemen eine Entscheidungsvariante zuordnen: Beispielhaft wird dies für die



Routenplanung (genauer das Problem des Handlungsreisenden) in Abschnitt 4.3 getan.

Unser Ziel ist es, die Komplexität oder Schwierigkeit von Problemen zu untersuchen. Können wir zeigen, dass die abgeleitete Entscheidungsvariante aufwändig ist, ist die ursprüngliche *Optimierungsvariante* auch schwierig: denn ein Algorithmus für die ursprüngliche Variante kann auch zur Lösung der Entscheidungsvariante eingesetzt werden. Tatsächlich geht die praktische Verknüpfung der beiden Varianten noch weiter, worauf wir ebenfalls im Abschnitt 4.3 kurz eingehen wollen.

In der komplexitätstheoretischen Literatur wird meistens die Schreibweise der *formalen Sprachen* verwendet. Eine zu einem Berechnungsproblem  $E$  gehörende formale Sprache enthält dann alle Eingaben  $x$  mit  $E(x) = 1$ , ist also die Menge  $\{x \mid x \text{ ist eine Eingabe von } E \text{ mit } E(x) = 1\}$ .

Unsere erste Komplexitätsklasse soll beschreiben, was in unserem Sinne *effizient* entscheidbar ist.

Die Klasse P enthält alle Entscheidungsprobleme, für die es Polynomialzeitalgorithmen gibt.

Polynomialzeit

Damit ist ein Problem mit Eingabegröße  $n$  dann ein Element der Menge P, wenn es einen Algorithmus  $A$  gibt, der es in Zeit  $O(n^k)$  löst. Dabei muss  $k$  eine Konstante sein und  $A$  für *alle* Eingaben das korrekte Ergebnis mit  $O(n^k)$  Schritten liefern.

Somit formalisiert die Klasse P eine Festlegung, die wir schon im letzten Kapitel getroffen haben. P enthält alle Entscheidungsprobleme, die im Sinne von Abschnitt 3.1 effizient lösbar sind und in der Folge werden wir *effizient* weiterhin als Synonym für Polynomialzeit verwenden. Nun könnte diese Laufzeitanforderung zu locker wirken: Neben linearer oder quadratischer Laufzeit ist theoretisch auch solche mit Aufwand  $O(n^{10})$  zugelassen. Polynome mit so großen Exponenten weisen ein Wachstum auf, welches der landläufigen Auffassung von Effizienz völlig widerspricht.

Dieses Konzept hat sich trotzdem aus verschiedenen Gründen bewährt:

1. Die Klasse P ist sehr robust. Rufen sich zwei Polynomialzeitalgorithmen gegenseitig auf oder wird ein solches Verfahren polynomial oft aufgerufen, ist die Gesamtlaufzeit wiederum polynomial: das Produkt zweier Polynome ist wieder ein Polynom ( $n^k \cdot n^l = n^{k+l}$ ). Durch diese Robustheit ist P maschinenunabhängig.
2. Die Erfahrung zeigt, dass praktisch bedeutsame Probleme aus P keinen sehr großen Exponenten haben. Lässt sich von einem solchen praktisch relevanten Problem erstmals zeigen, dass es überhaupt in P liegt, findet sich meistens auch ein Verfahren mit kleinen Exponenten.
3. Ist ein Problem *nicht* in Polynomialzeit berechenbar, kann man es in der Praxis nur für kleine Eingaben lösen.

Die Klasse P ist der Ausgangspunkt einer umfangreichen Theorie. Viele tiefe Einsichten blieben verborgen, hätte man sich vor der diskutierten Vergrößerung gescheut, allgemeine Polynome als Laufzeitschranke zuzulassen.

NP

Und schon folgt die nächste, genauso wichtige Komplexitätsklasse: NP enthält Probleme, die sich *effizient verifizieren* lassen.

Wir betrachten wieder das Entscheidungsproblem COMPOSITES. Wir lassen zunächst offen, wie sich für eine Eingabe – zum Beispiel die Zahl  $n = 1309$  – konkret ausrechnen lässt, ob sie zusammengesetzt ist. Wir nehmen stattdessen an, dass ein Freund von uns behauptet: „Die Zahl 1309 ist zusammengesetzt.“ Es handelt sich zwar um einen guten Freund, allerdings haben wir die Erfahrung gemacht, dass ihm die Wahrheit manchmal weniger wichtig ist, als selbst gut dazustehen. Darum fordern wir ihn auf: „Beweis es!“

Er antwortet: „1309 ist das Produkt der Zahlen 119 und 11.“ Das lässt sich allerdings leicht nachprüfen. Wir multiplizieren 119 mit 11 und vergleichen dieses Produkt mit 1309. Die Werte stimmen überein, und damit ist bewiesen: 1309 ist keine Primzahl,  $\text{COMPOSITES}(1309) = 1$ . Diesen Beweis zu führen – wenn wir die Zusatzinformation 119, 11 besitzen – ist offenbar einfacher, als ohne dieses Hilfsmittel die Frage zu beantworten.

Ähnlich verhält es sich mit dem Entscheidungsproblem MINIMUM aus Beispiel 4.1. Am Tag nach dem Betriebskegeln liegt uns eine Liste der Ergebnisse der mehr als tausend Teilnehmer vor. Der Generaldirektor trägt uns auf, folgendes herauszubekommen: Gibt es einen darunter, der weniger als einmal alle neun Kegel umwerfen konnte? Unser Freund hilft uns: „Ja“, sagt er zunächst nur, vielleicht um uns zu ärgern. Nach einer Pause fügt er hinzu: „Niegelmann, Norbert. Zeile 768“. Wir brauchen nur in der betreffenden Zeile nachschlagen, um uns zu überzeugen. Wir verifizieren die Aussage mit einem Hilfsmittel. Wir verwenden in der folgenden formalen Definition der Klasse NP für dieses Hilfsmittel den Begriff *Zertifikat*.

Ein Entscheidungsproblem  $E$  ist in NP, wenn eine Funktion  $f(x, y)$  existiert, so dass:

- Für Eingaben  $x$  mit  $E(x) = 1$  gibt es ein Zertifikat  $y$  polynomialer Größe mit  $f(x, y) = 1$ ,
- für Eingaben  $x$  mit  $E(x) = 0$  gibt es kein solches Zertifikat ( $f(x, y) = 0$  für alle möglichen Zertifikate),
- $f$  ist in polynomialer Zeit berechenbar.

Wir brauchen uns um die formalen Details nicht zu viele Gedanken machen. Unsere Art zu verifizieren, dass eine Zahl zusammengesetzt ist, stimmt mit der Definition überein. Da 1309 zusammengesetzt ist ( $\text{COMPOSITES}(1309) = 1$ ) gibt es das Zertifikat  $y = (119, 11)$ .  $f(n, (a, b))$  ist der Test, ob  $n = a \cdot b$  gilt. Ist  $n$  eine Primzahl, gibt es kein Zertifikat, das diesen Test erfüllt.

Asymmetrie von  
NP

Aber was ist mit der entgegengesetzten Fragestellung, ob eine Zahl *nicht* zusammengesetzt ist? Das ist gerade PRIMES aus Beispiel 4.1. Nach Definition von NP genügt es, für alle  $x$  mit  $E(x) = 1$  nachzuweisen, dass diese

Aussage zutrifft. Für Eingaben  $x$  mit  $E(x) = 0$  ist nichts zu zeigen. Wie kann ein Zertifikat aussehen, das uns überzeugt, dass eine Zahl eine Primzahl ist, dass also *kein* echter Teiler existiert? Liegt PRIMES in NP? Denken Sie darüber eine Weile nach. Offenbar ist in diesem Fall das Verifizieren schwieriger.

Das ist ein Unterschied zur Klasse P: Können wir entscheiden, dass eine Aussage zutrifft, haben wir damit auch entschieden, dass die verneinte Aussage nicht zutrifft. Können wir rechnerisch bestimmen, ob eine Zahl eine Primzahl ist, folgt aus der gleichen Berechnung, ob sie zusammengesetzt ist. Das ist beim Verifizieren nicht der Fall: Hilft ein Zertifikat bei dem Beweis, dass etwas zutrifft, folgt nicht, dass es ein Zertifikat gibt, das die gegenteilige Aussage beweisen lässt<sup>1</sup>.

Betrachten wir dazu Problem MINIMUM aus Beispiel 4.1. Mit dem Index  $i$  als Zertifikat eines Elementes  $a_i \leq k$  lässt sich  $\text{MINIMUM}(a_1, \dots, a_k, k)$  leicht überprüfen. Das ist eine lokale Eigenschaft. Ist hingegen gefragt, ob es *kein* Element kleiner-gleich  $k$  gibt – ob also alle Elemente größer  $k$  sind – ist dies eine globale Eigenschaft, und im schlechtesten Fall müssen wir alle Elemente  $a_i$  mit  $k$  vergleichen.

Andererseits: Wenn wir ein Problem effizient entscheiden können, können wir es erst recht effizient verifizieren:

$$P \subseteq NP$$

Das ist anschaulich klar: Ein P-Algorithmus entscheidet ein Problem in Polynomialzeit ohne weitere Information. Damit sind erst recht die niedrigeren Anforderungen an eine Verifikation im Sinne von NP erfüllt.

Bei klassischen Algorithmen wie in Abschnitt 2.1 beschrieben, ist durch eine Eingabe die Folge der Rechenschritte eindeutig festgelegt. Man nennt diese Algorithmen *deterministisch* und sie liegen auch der Klasse P zugrunde: P enthält die durch deterministische Polynomialzeitalgorithmen entscheidbaren Probleme. Der Name NP bedeutet nun *nichtdeterministische Polynomialzeit*. NP wurde ursprünglich über nichtdeterministische Berechnungen mit besagter Zeitbeschränkung definiert. Wir bleiben bei dem äquivalenten Konzept der Verifikation in Polynomialzeit.

Deterministische  
Berechnungen

Die Klasse NP spielt in der Komplexitätstheorie eine entscheidende Rolle, weil sie eine gewaltige Zahl von in der Praxis wichtigen Problemen enthält, für die man keinen effizienten Algorithmus kennt. Enthält die Klasse NP Probleme, die nicht in P enthalten sind? Können Quantenrechner alle Probleme in NP berechnen? Wir kommen in den Abschnitten 4.3 und 6.7 auf diese Fragen zurück.

Wichtigkeit von  
NP

<sup>1</sup>Tatsächlich liegt PRIMES sogar in P, siehe [3].

## 4.2 Randomisierte Algorithmen

In diesem Abschnitt lernen wir klassische Algorithmen kennen, die eine besondere Fähigkeit haben: Sie können Zufallszahlen erzeugen und den nächsten Schritt einer Berechnung von einer solchen Zahl abhängig machen. Solche Algorithmen sind nicht *deterministisch* im Sinne des vorhergehenden Abschnittes. Wir diskutieren zunächst, warum das eine sinnvolle Fähigkeit ist und betrachten Beispiele, in denen Zufallsschritte nützlich sind.

### 4.2.1 Mit dem Zufall rechnen

Wir beginnen mit einem bekannten Sortierverfahren.

Randomisiertes  
Quicksort

**Beispiel 4.2:** *Quicksort* gilt für viele Anwendungen als das in der Praxis schnellste Sortierverfahren. Es hat jedoch einen Nachteil: Während zum Beispiel Mergesort jede Folge mit Laufzeit  $O(n \log n)$  sortiert, benötigt Quicksort im schlechtesten Fall die Laufzeit  $\Theta(n^2)$ .

Quicksort zerlegt das zu sortierende Feld wiederholt in zwei Teilfelder. Dazu wird zunächst ein *Pivotelement*  $p$  gewählt und unsere Folge so aufgeteilt, dass ein Teilfeld nur Elemente kleiner gleich  $p$  enthält und das andere nur Elemente größer gleich  $p$ . Im optimalen Fall sind anschließend beide Felder gleich groß. Schlecht geteilt, befinden sich fast alle Elemente in demselben Teilfeld. Wird immer wieder schlecht geteilt, ist die Laufzeit quadratisch.

Tatsächlich tritt dies nur für einen kleinen Teil der Eingaben ein, und im mittleren Fall hat Quicksort natürlich auch die Laufzeit  $O(n \log n)$ . Allerdings ist der mittlere Fall über eine Gleichverteilung aller möglichen Eingaben definiert. Bei realen Anwendungen haben wir es nicht unbedingt mit Daten zu tun, die diesem abstrakten Ideal genügen (zum Beispiel sind häufig teilsortierte Folgen zu sortieren). Wir bekommen Probleme, wenn wir immer wieder mit Daten zu tun haben, für die unsere Wahl des Pivotelements ungünstig ist. Zum Glück kann man mit einer einfachen Idee verhindern, systematisch immer wieder in den schlechtesten Fall zu geraten.

Stellen wir uns ein Spiel vor. Wir haben einen sehr mächtigen Gegner, nennen wir diesen Big Brother. Zu einer festen Wahl des Pivotelements, liefert dieser uns stets Daten, die quadratische Laufzeit benötigen. Haben wir da überhaupt eine Chance? Wenn uns die Daten vorliegen, wählen wir das Pivotelement zufällig! Dieses Vorgehen nennt sich *randomisiertes Quicksort*. Damit schlagen wir Big Brother ein Schnäppchen, da auch dieser den Ausgang zukünftiger Zufallsexperimente nicht vorhersehen kann. Auf reale Daten angewandt heißt das: Ob der schlechteste Fall eintritt oder nicht, hängt nicht länger von der Eingabe ab, sondern von unserer zufälligen Wahl. Beim deterministischen Quicksort gibt es einzelne Eingaben mit schlechter Laufzeit. Das randomisierte Quicksort verhält sich im Mittel für alle Eingaben gleich: Die *erwartete Laufzeit*, siehe unten, ist  $O(n \log n)$ . Den schlechtesten Fall gibt es weiterhin, dieser hängt aber von den *Zufallsbits* ab.

Wir vermeiden es, ein neues abstraktes Berechnungsmodell einzuführen, sondern legen einfach fest:

Ein Algorithmus heißt *randomisiert*, wenn er eine Reihe von Zufallsbits  $r_0, \dots, r_{k-1}$  zu Hilfe nimmt.

Randomisierter Algorithmus

Jedes Bit  $r_i$  soll mit Wahrscheinlichkeit  $1/2$  den Wert 0 bekommt und mit derselben Wahrscheinlichkeit 1. Es handelt sich um einen fairen Münzwurf. Interpretieren wir die Zufallsbits als Binärzahl  $\sum r_i 2^i$ , ergibt sich damit die Gleichverteilung über  $0, \dots, 2^k - 1$  (siehe Abschnitt A.4 im Anhang).

Bei randomisierten Algorithmen hängt die Laufzeit  $T$  wie im Beispiel 4.2 nicht nur von der Eingabe  $x$  ab, sondern auch von der Wahl der Zufallsbits. Wir schreiben  $T(x, R)$ , wobei  $R = (r_0, \dots, r_{k-1})$  die Folge der zufälligen Bits ist.

Für einen randomisierten Algorithmus  $A$  und eine Eingabe  $x$  ist die *erwartete Laufzeit* der Erwartungswert

Erwartete Laufzeit

$$1/2^k \sum_{R \in \{0,1\}^k} T(x, R).$$

Die erwartete Laufzeit von  $A$  ist das Maximum dieser Werte unter allen Eingaben  $x$ .

Diese Definition weicht folgendermaßen von der im Abschnitt 3.1 ab. Dort ergibt sich die Laufzeit im mittleren Fall aus der Gleichverteilung aller Eingaben. Hier wird für *jede* Eingabe zugesichert, dass der Erwartungswert der Laufzeit dem angegebenen Wert entspricht.

**Beispiel 4.3:** Man kann beweisen: Wird für *Quicksort* das Pivotelement fest gewählt, so sind schlechte Eingaben unwahrscheinlich, wenn diese unabhängig und gleichverteilt ausgewählt werden. Wählen wir das Pivotelement zufällig, ist die erwartete Laufzeit  $O(n \log n)$ . Ob wir eine schlechte Laufzeit bekommen, hängt nicht mehr von der Eingabe ab, sondern von den Zufallsbits.

Noch einmal Quicksort

Die Idee der Randomisierung ist auch in der Spieltheorie nützlich. Bei einem Spiel wie *Schere-Stein-Papier* können wir den Vorteil eines psychologisch überlegenen Gegners ausgleichen, wenn wir unseren Spielzug auswürfeln. Kann dieser sonst aus dem Verlauf der vorhergehenden Spiele und unserem Gesichtsausdruck unsere nächste Aktion ziemlich genau vorhersehen, nützen ihm diese Fähigkeiten nichts mehr. Egal wie er vorgeht, die Gewinnwahrscheinlichkeit ist exakt  $1/2$ . Auch bei komplexeren Spielen, wie sie für wirtschaftswissenschaftliche und militärische Fragestellungen eine Rolle spielen, ist die zufällig gewählte Strategie oft die optimale.

Spieltheorie

### 4.2.2 Ein Primzahltest

Bei der randomisierten Version von Quicksort hängt die Laufzeit von der Wahl der Zufallsbits ab. Das Ergebnis ist jedoch immer dasselbe und korrekt.

Ihr ganzes Können entfalten randomisierte Algorithmen dann, wenn wir die Anforderungen an die Korrektheit des Ergebnisses abschwächen. Wir lassen einen Fehler zu: natürlich darf dieser nicht beliebig groß werden. Welche Anforderungen an den Fehler sinnvoll sind, diskutieren wir im nächsten Unterabschnitt. Zunächst untersuchen wir, welchen Vorteil es hat, Fehler zuzulassen.

Wir wollen entscheiden, ob eine Zahl  $n$  eine Primzahl ist. Alle möglichen Teiler auszuprobieren, ist nicht effizient. Deren Anzahl ist nicht polynomial in der Eingabegröße  $\log n$ . Hilfreich wäre nun eine Eigenschaft  $E$  für die gilt: Die *meisten* Zahlen mit der Eigenschaft  $E$  sind Primzahlen. Wenn wir für eine Zahl  $n$  die Eigenschaft  $E$  *effizient* überprüfen können, kann das sehr hilfreich sein. Wie kann so eine Eigenschaft aussehen? In Abschnitt A.5 des Anhangs findet sich der Satz von Fermat:

Wenn  $p$  eine Primzahl ist, dann gilt mit jeder Zahl  $a$  aus  $1, \dots, p-1$ :

$$a^{p-1} \equiv 1 \pmod{p}.$$

In erwähntem Abschnitt ist auch die verwendete Terminologie eingeführt. Leser, denen Berechnungen modulo  $p$  nicht vertraut sind, können sich für das Weitere auf die logische Struktur der *wenn-dann* Aussage beschränken.

### Vorstufe eines Primzahltests

Eingabe ist eine ungerade Zahl  $n > 2$ .

1. Wähle zufällig eine Zahl  $a$  aus  $2, \dots, n-1$ .
2. Falls  $a^{n-1} \equiv 1 \pmod{n}$ : Ausgabe *Primzahl*  
 Falls  $a^{n-1} \not\equiv 1 \pmod{n}$ : Ausgabe *keine Primzahl*

Wir analysieren das Ergebnis dieses einfachen Algorithmus (die Laufzeit ist polynomial: auf Seite 196 wird besprochen, wie sich  $a^{n-1} \pmod{n}$  effizient berechnen lässt). Kehren wir die *wenn-dann* Aussage von Fermats Satz um, erhalten wir für  $1 \leq a \leq n-1$ : Wenn  $a^{n-1} \not\equiv 1 \pmod{n}$  gilt, dann ist  $n$  keine Primzahl. Daraus folgt für unseren Algorithmus: Die Ausgabe *keine Primzahl* ist immer korrekt.

Und wenn die Ausgabe *Primzahl* lautet? Dann wissen wir über die Eingabe  $n$ , dass für irgendein  $a$  die Gleichung  $a^{n-1} \equiv 1 \pmod{n}$  gilt. Nehmen wir einmal an, wir könnten beweisen:

*Für jede zusammengesetzte Zahl  $n$  gilt für den überwiegenden Teil der Zahlen  $a$  aus  $2, \dots, n-1$ , dass  $a^{n-1} \not\equiv 1 \pmod{n}$ . Dann wäre die Ausgabe *Primzahl* mit hoher Wahrscheinlichkeit korrekt. Leider ist es nicht so einfach. Es gibt zusammengesetzte Zahlen  $n$ , für die obiger Test meistens scheitert. Diese Zahlen heißen Carmichael-Zahlen und davon gibt es unendlich viele,*

für Details siehe etwa [33]. Eine zufällig erzeugte Zahl ist nur mit geringer Wahrscheinlichkeit eine Carmichael-Zahl, trotzdem ergibt sich hier ein Problem für unseren Ansatz. Wir fordern von einem randomisierten Algorithmus nämlich folgendes:

Ein randomisierter Algorithmus liefert für *jede* Eingabe das korrekte Ergebnis mit höherer Wahrscheinlichkeit als das falsche.

Ein randomisierter Algorithmus darf also nicht für einige Eingaben korrekt und für andere falsch antworten. Aussagen über die Fehlerwahrscheinlichkeit oder die erwartete Laufzeit eines randomisierten Algorithmus beziehen sich nur auf die Zufallsbits und nicht auf die Eingaben.

Obige Vorstufe des Primzahltests versagt bei den Carmichael-Zahlen ganz. Um einen praktikablen Algorithmus zu konstruieren, ist ein feineres Kriterium nötig, wie bei dem Miller-Rabin-Test, siehe etwa [40].

Es gibt einen Algorithmus, der als Eingabe eine ganze Zahl  $n$  erwartet. Antwortet der Algorithmus *zusammengesetzt*, so stimmt dies immer, antwortet der Algorithmus *nicht zusammengesetzt* oder *Primzahl*, so ist dies in mehr als der Hälfte aller Fälle korrekt.

Damit ist es weniger ein Primzahltest, als ein Test auf Zusammengesetztheit. Müssen wir uns darauf verlassen können, dass eine Zahl zusammengesetzt ist, ist das Verfahren bestens geeignet. Zusammengesetzte Zahlen werden hin und wieder fälschlich abgelehnt, aber nie wird eine Primzahl diesen Test bestehen. Im nächsten Abschnitt werden wir sehen, wie sich der Fehler durch wiederholte Anwendung drastisch verringern lässt. Dadurch ist der Miller-Rabin-Test als Primzahltest geeignet.

### 4.2.3 Probabilistische Komplexitätsklassen

Wir beginnen mit folgender abstrakter Definition, die wir im Anschluss auf den Primzahltest anwenden.

Ein Entscheidungsproblem  $E$  liegt in der Klasse RP, wenn es einen randomisierten Algorithmus  $A$  gibt, so dass:

Einseitiger Fehler

- Falls  $E(x) = 0$ , so ist  $A(x) = 0$ ,
- falls  $E(x) = 1$ , so ist  $A(x) = 1$  mit Wahrscheinlichkeit größer gleich  $1/2$ , und
- die Laufzeit von  $A$  ist polynomial (für jede Eingabe und jede Ausprägung von  $A$ s Zufallsbits).

RP kürzt *randomized polynomial time* ab.

Andersherum betrachtet: Ist die Ausgabe  $A(x) = 1$ , so gilt immer  $E(x) = 1$ . Und aus  $A(x) = 0$  folgt  $P(E(x) = 1) < \frac{1}{2}$ .

Das trifft gerade auf den Miller-Rabin-Test aus dem letzten Abschnitt zu und somit folgt:

$$\text{COMPOSITES} \in \text{RP}.$$

Ihre praktische Relevanz erhalten RP-Algorithmen durch die folgende, leicht einzusehende Eigenschaft:

Probability  
amplification

Der Fehler eines RP-Algorithmus lässt sich mit  $k$  Wiederholungen auf den Wert  $1/2^k$  verringern.

Diese Idee heißt *probability amplification*:

Führe den RP-Algorithmus  $A$  bis zu  $k$ -mal aus:

- Liefert  $A$  das erste Mal 1, gib 1 zurück.
- Ist das Ergebnis  $k$ -mal 0, gib 0 zurück.

Wichtig ist, dass bei jeder Wiederholung neue Zufallsbits verwendet werden und die Ergebnisse voneinander unabhängig sind. Wird in dieser Schleife das erste Mal 1 ermittelt, können wir abbrechen. Einem RP-Algorithmus mit diesem Ergebnis kann man voll vertrauen. Falls davor der Rückgabewert schon einige Male 0 war, war dies ein Irrtum aufgrund ungünstiger Zufallsbits.

Analysieren wir also den Fall, dass 1 ist die richtige Antwort ist,  $A$  hingegen  $k$ -mal 0 liefert. Wie groß ist die Wahrscheinlichkeit, dass sich  $A$   $k$ -mal in Folge irrt? Im ersten Durchlauf ist die Irrtumswahrscheinlichkeit kleiner gleich  $1/2$ . Dass  $A$  beim ersten *und* beim zweiten Mal 0 liefert hat eine Wahrscheinlichkeit kleiner gleich  $1/2 \cdot 1/2 = 1/4$ . Die Wahrscheinlichkeiten multiplizieren sich aufgrund der unabhängigen Wahl der Zufallsbits. Abbildung 4.1 stellt dies für einen Algorithmus für COMPOSITES dar, zum Beispiel den Miller-Rabin-Test. Dieses Argument lässt sich fortsetzen und wir erhalten, dass der einseitige Fehler nach  $k$  Wiederholungen kleiner gleich  $1/2^k$  ist.  $\diamond$

Mit der gleichen Idee können wir in der Definition von RP die Wahrscheinlichkeit  $1/2$  durch jede positive Konstante  $c < 1$  ersetzen. Haben wir zum Beispiel einen Algorithmus mit dem Fehler  $3/4$ , kann man den Fehler durch drei Wiederholungen auf  $(3/4)^3 \approx 0,42$  senken und erhält so einen RP-Algorithmus gemäß unserer Definition.

RP-Algorithmen kommen mit polynomialer Zeit aus, und somit benötigt für jede Konstante  $k$  auch die  $k$ -fache Ausführung nur polynomielle Zeit. Dies gilt auch dann noch, wenn  $k$  nicht konstant, sondern durch ein Polynom in der Eingabegröße beschränkt ist.



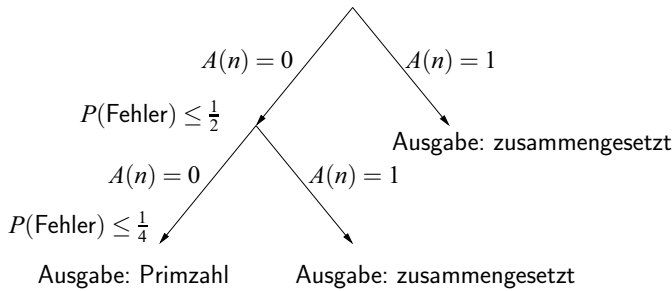


Abbildung 4.1: Zweifache Ausführung eines RP-Algorithmus für zusammengesetzte Zahlen.

Randomisierte Algorithmen mit einseitigem Fehler sind offenbar uneingeschränkt nützlich. Sie entsprechen Polynomialzeitalgorithmen, deren Fehler wir beliebig senken können. Auch Algorithmen mit zweiseitigem Fehler können nützlich sein. Voraussetzung ist, dass der Fehler durch eine Konstante beschränkt ist.

Ein Entscheidungsproblem  $E$  liegt in der Klasse BPP, wenn es einen randomisierten Algorithmus  $A$  gibt, mit

- falls  $E(x) = 1$ , so  $A(x) = 1$  mit Wahrscheinlichkeit größer gleich  $2/3$ ,
- falls  $E(x) = 0$ , so  $A(x) = 0$  mit Wahrscheinlichkeit größer gleich  $2/3$ , und
- die Laufzeit von  $A$  ist polynomial.

BPP kürzt *bounded error probabilistic polynomial time* ab.

Beschränkter Fehler

*Bemerkung:* Statt  $2/3$  kann jede Konstante echt größer  $1/2$  verwendet werden.

Der Fehler eines BPP-Algorithmus  $A$  lässt sich durch ein *Mehrheitsvotum* verringern. Dazu betrachten wir den folgenden Algorithmus  $A'$ . Er führt für eine Eingabe  $x$  fünfmal  $A(x)$  aus. Er merkt sich die fünf Ergebnisse; die Ausgabe ist jenes, das am häufigsten ermittelt wurde.  $A$  hat die Fehlerschranken wie in der obigen Definition. Wie groß ist die Wahrscheinlichkeit, dass sich  $A'$  irrt?

Wir nehmen an,  $A'(x)$  antwortet 0, obwohl  $E(x) = 1$  (die andere Art zu irren –  $A'(x) = 1$  und  $E(x) = 0$  – ist symmetrisch). Das Ergebnis  $A'(x) = 0$  ergibt sich, wenn  $A$  dreimal, viermal oder fünfmal 0 ergibt. Wegen  $E(x) = 1$ ,

tritt  $A(x) = 0$  mit Wahrscheinlichkeit kleiner gleich  $1/3$  ein. Also irrt  $A'$  mit Wahrscheinlichkeit kleiner gleich

$$\binom{5}{3} \cdot (1/3)^3 \cdot (2/3)^2 + \binom{5}{4} \cdot (1/3)^4 \cdot (2/3) + (1/3)^5 = 17/81.$$

In dem Beispiel verringert sich der Fehler. Aber in welcher Größenordnung sinkt er bei steigender Zahl der Wiederholungen?

Der Fehler eines BPP-Algorithmus lässt sich durch wiederholte Ausführung samt Mehrheitsvotum verringern; er sinkt exponentiell mit der Anzahl der Wiederholungen.

Wir passen den Beweis aus [137] für unsere Zwecke an.  $A'$  führe  $k$ -mal den BPP-Algorithmus  $A$  aus,  $k$  ungerade. Dies liefert eine Folge  $S$  von  $k$  Ergebnissen. Mit  $r$  bezeichnen wir die Anzahl der korrekten und mit  $f$  die der falschen Ergebnisse. Damit ist klar, dass  $r + f = k$  gilt und dass sich der Mehrheitsvotumsalgorithmus  $A'$  irrt, falls  $f > r$ . Diese Fehlerwahrscheinlichkeit von  $A'$  wollen wir nun (grob) abschätzen. Sei  $x$  eine Eingabe und  $e_x$  die Fehlerwahrscheinlichkeit von  $A$  auf  $x$ . Da  $A$  ein BPP-Algorithmus ist, gilt  $e_x < 1/3$ . Für die Wahrscheinlichkeit  $p_S$  einer beliebigen Ergebnisfolge  $S$  mit  $f > r$  gilt nun:

$$p_S = e_x^f (1 - e_x)^r \stackrel{1.}{\leq} (1/3)^f \cdot (2/3)^r \stackrel{2.}{\leq} (1/3)^{k/2} \cdot (2/3)^{k/2}.$$

Das wollen wir einsehen. Zunächst gilt  $a(1-a) \leq b(1-b)$  für  $a \leq b < 1/2$ . Dazu kann man etwa die Ableitung von  $a(1-a)$  im Intervall  $0 \leq a < 1/2$  betrachten. Für  $a \leq b$  folgt weiter  $a^n(1-a)^n \leq b^n(1-b)^n$ , da  $x^n$  für  $x \geq 0$  ebenfalls monoton wächst. Aus  $e_x < 1/3$  erhalten wir damit zunächst  $e_x^r (1 - e_x)^f \leq (1/3)^r \cdot (2/3)^f$ . Da auch  $e_x^{f-r} \leq (1/3)^{f-r}$  gilt, erhalten wir nun durch Multiplizieren Ungleichung 1. Ungleichung 2. gilt, da  $f \geq k/2 \geq r$  und  $1/3 \leq 2/3$ .

Nun haben wir die Fehlerwahrscheinlichkeit einer Ergebnisfolge  $S$ , für die sich  $A'$  irrt, ermittelt und müssten nun über alle derartigen Folgen summieren. Stattdessen verwenden wir folgende grobe Abschätzung: Die Anzahl der Folgen ist kleiner gleich  $2^k$ . Also gilt:

$$P(A' \text{ irrt für eine Eingabe } x) \leq 2^k \cdot (1/3)^{k/2} \cdot (2/3)^{k/2} = (8/9)^{k/2}.$$

Ermitteln wir für ein vorgegebenes  $t$  nun noch  $k$ , so dass  $(8/9)^{k/2} \leq 1/2^t$ . Logarithmieren ergibt  $(k/2) \cdot \log_2(8/9) \leq -t$ , woraus  $k > -(2t)/\log_2(8/9) \approx 11,77 \cdot t$  folgt.  $\diamond$

Ein exponentiell sinkender Fehler bedeutet, dass wir in Polynomialzeit einen verschwindend geringen Fehler erreichen können. Darum werden wir in der Praxis mit Problemen in der Klasse BPP fertig. Die Grenze der mit klassischen Rechnern entscheidbaren Aufgaben liegt nicht bei P, sondern eher bei BPP.

Komplexitätstheoretiker interessieren sich für den Zusammenhang der Komplexitätsklassen. Wir beginnen mit den einfachen Relationen. So gilt

$$P \subseteq RP \subseteq BPP.$$

Der Unterschied zwischen RP und BPP liegt in der Fehlerbeschränkung. Diese ist für BPP laxer, da der Fehler zweiseitig ist. Also besteht jedes Problem in RP den Test auf Mitgliedschaft in der Klasse BPP. Ebenso lässt sich ein P-Algorithmus als ein RP-Algorithmus auffassen, der von seinen Zufallsbits keinen Gebrauch macht und sich somit auch nicht irrt. Allerdings ist nicht bekannt, ob die Subsumtionen echt sind, ob zum Beispiel  $P=RP$  oder  $P \neq RP$  gilt. Weiterhin gilt

$$RP \subseteq NP,$$

während der Zusammenhang zwischen BPP und NP unbekannt ist. Diese Zusammenhänge sind in Abbildung 4.2 zusammengefasst. Die gestrichelte Linie markiert die Grenze der Probleme, für die wir Polynomialzeitalgorithmen kennen. Der Frage, welchen Status die Probleme in NP haben, widmet sich der folgende Abschnitt.

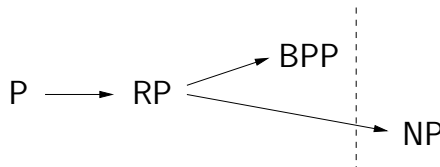


Abbildung 4.2: Probabilistische Komplexitätsklassen, P und NP

## 4.3 Unlösbare Probleme? NP-Vollständigkeit

Wenn ein Problem  $E$  in der Klasse NP enthalten ist, gilt für  $E$ : Ist  $x$  eine Eingabe mit  $E(x) = 1$ , so können wir das effizient verifizieren; vorausgesetzt, uns steht ein Zertifikat  $y$  zur Verfügung (siehe Seite 102). Nun ist diese Eigenschaft von großem theoretischen Interesse, in der Praxis müssen wir jedoch normalerweise ohne Hilfsmittel berechnen, welchen Wert  $E(x)$  ist. Sind alle Probleme in NP auch effizient entscheidbar? Gilt

$$P = NP?$$

Man weiß es nicht. Es ist klar, dass  $P \subseteq NP$  gilt, aber die Frage ob  $P=NP$  oder  $P \neq NP$  gehört zu den wichtigen offenen Fragen der mathematischen Wissenschaften. So ist es eines der sieben mathematischen Probleme, für deren Klärung das *Clay Mathematics Institute (CMI)* eine Million Dollar ausgelobt hat, siehe [39].

Allerdings gibt es zahlreiche Probleme in NP, von denen man vermutet, dass sie nicht in P liegen. Dazu gehört das *Problem des Handlungsreisenden* oder englisch *traveling salesman problem*, TSP:

Ein Handlungsreisender sitzt in seinem Büro und befasst sich mit folgender Aufgabe: er muss eine Reihe von Städten besuchen und anschließend in

TSP

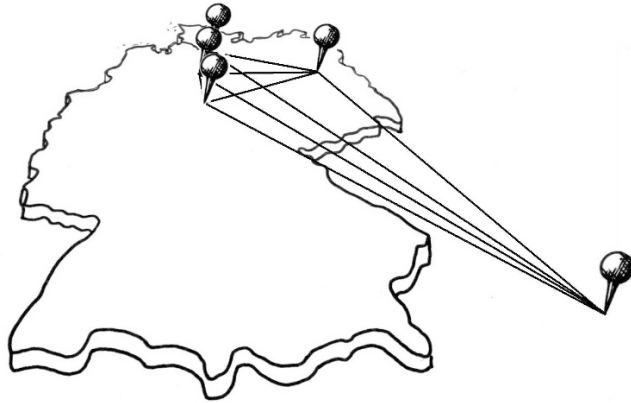


Abbildung 4.3: Eine Eingabe für das Problem des Handlungsreisenden

sein Büro zurückkehren. Abbildung 4.3 zeigt eine mögliche Eingabe für das Problem des Handlungsreisenden. Hier hat er einige Städte im norddeutschen Raum und die Hauptstadt von Österreich zu bereisen. Der Handlungsreisende interessiert sich nun für eine möglichst kurze Rundreise und könnte sich eine der folgenden Fragen stellen:

1. Welches ist die kürzeste Rundreise?
2. Gibt es eine Rundreise die nicht länger als 2200 km ist? Allgemein, die nicht länger als ein vorgegebener fester Schwellenwert  $k$  ist?

Diese beiden Ausprägungen nennt man *Optimierungsvariante* und *Entscheidungsvariante*. Letztere nennen wir TSP und beschränken uns in der Folge auf diese. Es besteht zwischen den Varianten folgender Zusammenhang: Wenn die Optimierungsvariante in Polynomialzeit lösbar wäre, dann natürlich erst recht die Entscheidungsvariante. Aus  $\text{TSP} \notin \text{P}$  würde also folgen, dass es auch keine Polynomialzeitlösung für die Optimierungsvariante gibt<sup>2</sup>.

Man weiß nicht, ob TSP in der Klasse P liegt. Bekommen wir hingegen eine Lösung vorgeschlagen – sprich eine Rundreise, die nicht länger als 2200 km ist – können wir das effizient nachprüfen. Die Rundreise, also die Reihenfolge, in der die Städte besucht werden, ist das Zertifikat aus der Definition der Klasse NP. Wir addieren die Entfernungen zwischen den Städten gemäß dem Zertifikat und vergleichen mit der Schranke  $k$ , die oben beispielhaft auf 2200 gesetzt wurde. Also gilt  $\text{TSP} \in \text{NP}$ .

Wie oben erwähnt ist unklar, ob  $\text{P}=\text{NP}$  gilt, ob also effiziente Verifikation immer effiziente Entscheidbarkeit bedeutet, oder aber  $\text{P} \neq \text{NP}$ , ob also mindestens ein Problem in NP nicht in P liegt. Immerhin ist es gelungen, die

<sup>2</sup>Für TSP kann durch wiederholten Aufruf aus einem (fiktiven) effizienten Algorithmus für die Entscheidungsvariante auch die Lösung der Optimierungsvariante, also die konkrete optimale Route, in Polynomialzeit gefunden werden. Ein ähnlicher Zusammenhang gilt für viele Optimierungsprobleme

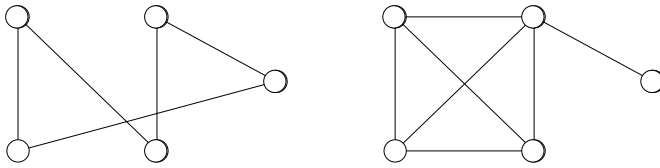


Abbildung 4.4: Zwei Graphen: Der linke enthält einen Hamiltonkreis, der rechte nicht.

*schwersten* Probleme in NP zu charakterisieren. Das wären Kandidaten für solche, die nicht in P liegen. Ein Spoiler: TSP gehört dazu, ist *NP-vollständig*.

Wir führen ein Vergleichskriterium ein, um zu bestimmen, wann in Bezug auf P, ein Problem  $E_2$  *mindestens so schwer wie* ein Problem  $E_1$  ist.

Wir definieren für  $E_1$  und  $E_2$  aus NP:

$$E_1 \leq_P E_2,$$

wenn es einen Polynomialzeitalgorithmus  $A$  gibt, der

1. Eingaben  $x$  von  $E_1$  auf Eingaben  $A(x)$  von  $E_2$  abbildet, so dass
2.  $E_1(x) = E_2(A(x))$ .

Der Algorithmus  $A$  heißt *Polynomialzeitreduktion* von  $E_1$  auf  $E_2$ .

Polynomialzeit-  
reduktion

Anschaulich bedeutet  $E_1 \leq_P E_2$ , dass ein Algorithmus für  $E_2$  das Problem  $E_1$  lösen kann. Und zwar, indem zuvor die Eingabe von  $E_1$  mittels des Polynomialzeitalgorithmus  $A$  in eine Eingabe für  $E_2$  überführt wurde. Eigenschaft 2. obiger Definition stellt sicher, dass die Ausgabe von  $E_2$  auf  $A(x)$  gerade das Ergebnis von  $E_1$  ergibt. Damit können wir folgern: Nach dem Kriterium der Entscheidbarkeit in Polynomialzeit ist  $E_2$  mindestens so schwer wie  $E_1$ .

*Zur Benennung:* Falls Sie den Ausdruck *Reduktion* verwirrend finden, befinden Sie sich in bester Gesellschaft; schließlich wird  $E_1$  nicht kleiner, leichter, eingedickt oder welche Alltagsbedeutung *Reduktion* sonst noch hat. Man sollte es als technischen Begriff davon losgelöst verwenden. Am ehesten passt eine Formulierung der Art: Um  $E_1$  zu lösen, genügt es  $E_2$  zu entscheiden, also reduziert sich die Aufgabe  $E_1$  zu erledigen, darauf  $E_2$  zu tun.

Nun folgt ein Graphenproblem, dass wir auf TSP reduzieren wollen.

**Beispiel 4.4:** Abbildung 4.4 zeigt zwei *Graphen*. Ein Graph besteht zum einen aus Knoten, zum anderen aus Kanten: Zwischen je zwei Knoten kann es eine Kante geben oder auch nicht. Die beiden Graphen aus der Abbildung besitzen gerade fünf Knoten.

Hamiltonkreis

Formal besteht ein Graph aus zwei Mengen: den Knoten  $V$  und den Kanten  $E$ . Eine Kante  $e$  wird als Paar von Knoten  $\{u, v\}$  beschrieben. Zwischen zwei

beliebigen Knoten  $u$  und  $v$  gibt es einen Weg, wenn man entlang der Kanten von  $u$  nach  $v$  gelangt.

Der linke Graph in Abbildung 4.4 hat eine besondere Eigenschaft. Es gibt einen Weg, der jeden Knoten genau einmal besucht und am Ausgangspunkt endet: so einen Weg nennt man *Hamiltonkreis*. Entfernt man irgendeine Kante, existiert in dem Beispiel kein solcher Kreis mehr; fügt man Kanten hinzu, existiert natürlich weiterhin ein Hamiltonkreis.

Formal ist ein Hamiltonkreis in einem Graph mit  $k$  Knoten eine Folge von Kanten  $\{u_1, u_2\}, \{u_2, u_3\}, \dots, \{u_{k-1}, u_k\}, \{u_k, u_1\}$  mit  $u_i \neq u_j$  für  $i \neq j$ .

In dem Graphen auf der rechten Seite von Abbildung 4.4 gibt es keinen Hamiltonkreis. Der Knoten ganz rechts ist nämlich nur über eine einzige Kante zu erreichen, an dieser Stelle endet jeder Kreis. Nun können wir das Entscheidungsproblem HC definieren. Eingabe ist ein Graph  $G$ . Besitzt der Graph  $G$  einen Hamiltonkreis gilt  $HC(G) = 1$ , sonst  $HC(G) = 0$ .

**Aufgabe 4.1:** Zeigen Sie, dass HC in NP liegt.

**Beispiel 4.5:** Jetzt wollen wir zeigen, dass HC nicht schwerer ist, als das Problem des Handlungsreisenden:

$$HC \leq_P TSP.$$

Dazu beschreiben wir einen Algorithmus an, der aus jeder Eingabe  $G$  von HC eine Eingabe  $x_G$  für TSP konstruiert, so dass gilt:  $HC(G) = TSP(x_G)$ .

Dieses Verfahren arbeitet wie folgt: Für jeden Knoten  $v$  aus  $G$  erzeugen wir eine Stadt  $s(v)$ . Die Entfernungen zwischen den Städten richtet sich danach, ob es zwischen den zugehörigen Knoten in  $G$  eine Kante gibt. Gibt es in  $G$  zwischen  $v$  und  $w$  eine Kante, setzen wir die Distanz zwischen  $s(v)$  und  $s(w)$  auf 1, gibt es keine Kante, setzen wir sie auf 2. Die Schranke  $k$  legen wir als die Anzahl der Knoten  $V$  fest.

Dieses Verfahren ist offensichtlich in Polynomialzeit ausführbar. Ein Hamiltonkreis in  $G$  entspricht einer Rundreise der Länge  $|V|$  in der Eingabe  $x_G$  für TSP und umgekehrt. Damit haben wir  $HC \leq_P TSP$  bewiesen.  $\diamond$

Vorstehendes Beispiel macht deutlich, wie man nun mit einem Algorithmus  $A_{TSP}$  für TSP das Entscheidungsproblem HC lösen kann. Wäre Algorithmus  $A_{TSP}$  ein Polynomialzeitalgorithmus, so würde  $HC \in P$  folgen.

Allgemein:

$$\text{Aus } E_1 \leq_P E_2 \text{ und } E_2 \in P \text{ folgt } E_1 \in P,$$

und andererseits:

$$\text{Aus } E_1 \leq_P E_2 \text{ und } E_1 \notin P \text{ folgt } E_2 \notin P.$$

Tatsächlich kennt man keinen Algorithmus, der TSP für alle Eingaben in Polynomialzeit löst. TSP hat aber eine andere Eigenschaft:

Ein Problem  $E$  heißt NP-vollständig, wenn

1.  $E$  in NP liegt, und
2.  $E' \leq_P E$  für alle Probleme  $E'$  in NP gilt,

wenn  $E$  also bzgl.  $\leq_P$  zu den schwersten Problemen in NP gehört.

NP-vollständig

Gilt für ein Problem nur die zweite Bedingung, so heißt es *NP-schwer*. Es ist dann mindestens so schwer, wie ein NP-vollständiges Problem, kann aber außerhalb von NP liegen.

Wie beweist man die NP-Vollständigkeit eines Entscheidungsproblems  $E \in \text{NP}$ ? Laut Definition ist zu zeigen, dass sich *alle* Probleme in NP auf  $E$  reduzieren lassen. Das klingt schwierig; hat man es aber für ein Problem  $E$  geschafft, vereinfacht sich die Aufgabe. Dann würde zum Beispiel aus  $E \leq_P \text{HC}$  folgen, dass auch HC NP-vollständig ist und durch Beispiel 4.5 auch TSP. Verketteten wir zwei Polynomialzeitreduktionen, ergibt sich eine ebensolche. Somit lassen sich über  $E$  alle Probleme in NP auf HC und weiter auf TSP reduzieren.

Es gilt tatsächlich, dass HC und damit auch TSP NP-vollständig sind, siehe z.B. [40]. Ein weiteres Beispiel ist 3-SAT, das im Abschnitt 11.1 knapp beschrieben wird.

Es sind so viele NP-vollständige Probleme bekannt, dass damit schon vor Jahrzehnten ganze Bücher gefüllt wurden, siehe [66]. Gerade viele Entscheidungsvarianten von Optimierungsproblemen gehören dazu. Und all diese NP-vollständigen Probleme sind durch Polynomialzeit-Algorithmen miteinander verbunden. Wenn wir ein NP-vollständiges Problem effizient entscheiden können, dann gilt das für alle. Ist  $E$  NP-vollständig, so würde aus  $E \in P$  folgen, dass  $P = \text{NP}$ . Aus der Sicht von  $P$  könnte man sagen, dass alle NP-vollständigen Probleme einen gemeinsamen schweren Kern besitzen, für den wir keinen effizienten Algorithmus kennen – vermutlich wird es einen solchen Algorithmus auch nicht geben. Noch plakativer: So unterschiedlich die NP-vollständigen zum Teil erscheinen, kann man diese von  $P$  aus als verschiedene Kostümierungen eines einzelnen schweren Problems ansehen.

Die Bedeutung der NP-Vollständigkeit

Aber für kein NP-vollständiges Problem wurde bisher ein effizienter Algorithmus gefunden. Die Fachwelt geht im Gegenteil davon aus, dass  $P \neq \text{NP}$  gilt (ohne Beweis sind jedoch beide Optionen weiterhin denkbar). Abbildung 4.5 stellt diese Situation dar.  $P$  und die Menge der NP-vollständigen Probleme sind Teilmengen von NP. Im Falle  $P = \text{NP}$  sind die drei Mengen identisch, im Fall  $P \neq \text{NP}$  sind es drei verschiedene Mengen.

Probleme vom Typ TSP begegnen uns im Zusammenhang mit der Quantensuche (Kapitel 6) erneut. Ein anderes wichtiges Berechnungsproblem ist die Faktorisierung oder Primfaktorzerlegung ganzer Zahlen. Es geht darum, eine Zahl  $n$  als Produkt nicht weiter zerlegbarer Zahlen darzustellen, siehe Abschnitt 8.1. Kennen wir die Primfaktorzerlegung einer Zahl – zum Beispiel  $28644 = 2 \cdot 2 \cdot 3 \cdot 7 \cdot 11 \cdot 31$  – können wir diese effizient verifizieren. Analog ist die zugehörige Entscheidungsvariante in NP. Für eine Eingabe  $n, k, l$  ist zu entscheiden, ob  $n$  einen Primteiler  $p$  besitzt, für den  $k \leq p \leq l$  gilt.

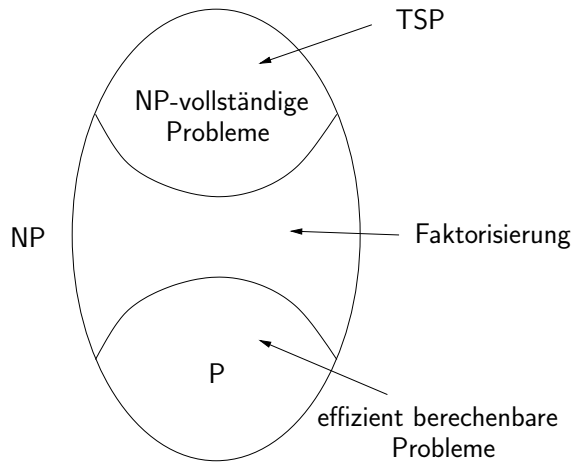


Abbildung 4.5: Die Klasse P ist in NP enthalten, ebenso die Menge der NP-vollständigen Probleme

Es wird allgemein vermutet, dass das Faktorisierungsproblem weder in P liegt, noch dass es so schwer wie NP-vollständige Probleme ist. Mit Shors Algorithmus ist es in Polynomialzeit lösbar: auf einem Quantenrechner (Kapitel 8). Damit können Quantenrechner ein Problem mit polynomialem Aufwand lösen, für das kein klassischer effizienter Algorithmus bekannt ist. Es stellt sich die Frage, ob Quantenrechner auch NP-vollständige Probleme lösen können. Diese Frage diskutieren wir in Zusammenhang mit Grovers Algorithmus in Abschnitt 6.7. Im nächsten Abschnitt untersuchen wir den Zusammenhang zwischen der Klasse der mit Quantenrechnern effizient lösbaren Probleme und den klassischen Komplexitätsklassen.

## 4.4 Quantenkomplexitätstheorie

Wir beginnen damit, die Klasse der durch Quantenalgorithmen effizient berechenbaren Probleme zu definieren. In Abschnitt 4.2 haben wir gesehen, wie randomisierte Berechnungen mit beschränktem Fehler nützliche Ergebnisse liefern: Der Fehler lässt sich verringern, indem wir die Berechnungen mehrfach ausführen. Er sinkt exponentiell in der Anzahl der Wiederholungen. Wir gestehen Quantenalgorithmen einen beschränkten Fehler zu und definieren die von Quantenalgorithmen effizient berechenbaren Probleme analog zu BPP.



Ein Entscheidungsproblem  $E$  liegt in der Klasse BQP, wenn es einen Quantenalgorithmus  $A$  gibt, mit

- falls  $E(x) = 1$ , so  $A(x) = 1$  mit Wahrscheinlichkeit größer gleich  $2/3$ ,
- falls  $E(x) = 0$ , so  $A(x) = 0$  mit Wahrscheinlichkeit größer gleich  $2/3$ , und
- $A$  ist durch uniforme Quantenschaltkreise polynomialer Größe berechenbar.

BQP kürzt *bounded error quantum polynomial time* ab.

Statt  $2/3$  kann jede Konstante echt größer  $1/2$  verwendet werden.

In Abschnitt 3.5 haben wir effiziente deterministische Berechnungen in effiziente Quantenberechnungen überführt. In unserer komplexitätstheoretischen Terminologie heißt das:

$$P \subseteq BQP.$$

In Abschnitt 4.2 haben wir gesehen, wie klassische Berechnungen durch Zufall angereichert werden können. Im Grunde enthält die Klasse BPP das, was wir mit klassischen Rechnern realisieren können.

Wie verhalten sich Quantenalgorithmen zu randomisierten Algorithmen? Die einzelnen Rechenschritte auf einem Quantenregister mittels unitärer Transformationen sind im Sinne unserer Definition *deterministisch*. Allerdings kommt kein Quantenalgorithmus ohne Messungen aus. Das können wir nutzen, um eine randomisierte Berechnung in eine Quantenrechnung umzuformen: Was kann ein randomisierter Algorithmus, was ein deterministischer nicht kann? Münzen werfen; vornehmer ausgedrückt: Zufallsbits erzeugen. Also gerade das, was wir mit unserem ersten Quantenalgorithmus erledigt haben (siehe Abschnitt 2.4).

Unsere Aufgabe ist es, einen randomisierten Algorithmus zu simulieren. Dazu passen wir den Beweis aus Abschnitt 3.5 an. Immer wenn ein Zufallsbit erzeugt, versetzt unser Quantenalgorithmus ein Hilfsbit in den Zustand

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = H|0\rangle$$

und misst dieses Quantenbit. Mit dieser einfachen Idee lässt sich beweisen:

$$BPP \subseteq BQP.$$

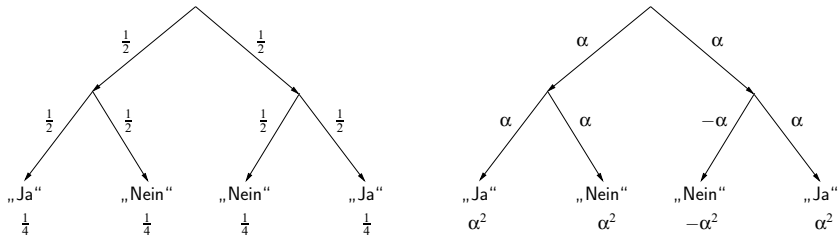


Abbildung 4.6: Eine randomisierte und eine Quantenberechnung ( $\alpha = \frac{1}{\sqrt{2}}$ ).

Was aber ist der Unterschied zwischen Quantenberechnungen und randomisierten Berechnungen? Es gibt deutliche Parallelen zwischen einer Wahrscheinlichkeitsverteilung und einer Superposition. Würfeln wir eine Zahl  $x$  zwischen 0 und 64 aus und berechnen den Funktionswert  $f(x)$ , bekommen wir dasselbe Ergebnis, wie mit der Quantenberechnung

$$|0\rangle|0\rangle \rightarrow \sum_{x=0}^{64} |x\rangle|0\rangle \rightarrow \sum_{x=0}^{64} |x\rangle|f(x)\rangle$$

nach einer abschließenden Messung.

Zu den Unterschieden gehört, dass klassische Berechnungen nur Pseudozufallszahlen erzeugen können. Zudem gibt es in der klassischen Welt nichts, was dem Quantenphänomen Verschränkung ähnelt. Und ganz entscheidend für die Algorithmen ist *Interferenz*: Klassische Wahrscheinlichkeiten addieren sich nur konstruktiv, während Amplituden auch negative Werte annehmen können. Das verdeutlicht Abbildung 4.6. Der Baum auf der linken Seite stellt einen randomisierten Prozess dar, bei dem zweimal Münzen geworfen werden; bei Kopf gehen wir jeweils nach links und bei Zahl nach rechts. Es gibt vier mögliche Ausgängen mit Wahrscheinlichkeit  $1/4$ . Zweien ordnen wir das Ergebnis *Ja* zu und zweien *Nein*. Beide Antworten erhalten wir dann mit Wahrscheinlichkeit  $1/4 + 1/4 = 1/2$ .

Im Quantenfall sind Wahrscheinlichkeiten durch Amplituden ersetzt.  $\alpha$  steht für  $\frac{1}{\sqrt{2}}$ . Hier ergibt sich für *Nein* die Amplitude  $\alpha^2 - \alpha^2 = 0$  (destruktive Interferenz) und für *Ja* die Amplitude  $\alpha^2 + \alpha^2 = 1/2 + 1/2 = 1$  (konstruktive Interferenz). Wie Interferenz für Quantenberechnungen genutzt wird, konnten wir bereits am Algorithmus von Deutsch sehen (siehe Abschnitt 2.6).

## PSPACE

An dieser Stelle definieren wir eine weitere deterministische Komplexitätsklasse. Bisher haben wir Probleme nach ihrer Laufzeit klassifiziert. Eine interessante weitere Ressource ist der *Speicherverbrauch*. Diesen definieren wir als Funktion in der Eingabegröße. Wir zählen dazu die Anzahl der Speicherzellen, die von einer Turingmaschine oder einem anderen Computer irgendwann während einer Berechnung verwendet wurden. Wir können diese Definition so vage halten, da uns wiederum nur die Größenordnung interessiert.

Die Klasse PSPACE enthält alle Probleme, die sich mit deterministischen Algorithmen berechnen lassen, die mit polynomial viel Platz auskommen.

PSPACE ist eine große Klasse. Sie enthält natürlich alles in P, denn man kann in *einem* Rechenschritt nur *eine* Speicherzelle verwenden. Sie enthält sogar alle Probleme in NP: Ein PSPACE-Algorithmus kann ein Problem aus NP lösen, indem er nacheinander alle möglichen Zertifikate ausprobiert. Der Zeitverbrauch ist zwar exponentiell; auf dem Band steht jedoch immer nur ein Zertifikat polynomialer Größe, und auch die Verifikation kommt mit polynomialen Platz aus, da wir in einem Rechenschritt auf nur eine Speicherzelle zugreifen können. Es gilt also:

$$\text{NP} \subseteq \text{PSPACE}.$$

Dass ein Problem in der Klasse PSPACE enthalten ist, bedeutet nicht, dass es auf einem klassischen Rechner praktisch ausführbar ist. Wie wir gesehen haben, kann der Zeitverbrauch exponentiell wachsen. Aufschlussreich ist, dass sich die Berechnungskraft von Quantenverfahren in die von deterministischen und randomisierten Algorithmen einordnen lässt. So gilt  $\text{BQP} \subseteq \text{PSPACE}$ , siehe etwa [17].

Wir fassen die Ergebnisse zusammen:

$$\text{BPP} \subseteq \text{BQP} \subseteq \text{PSPACE}.$$

Zusammenfassung

Siehe dazu auch Abbildung 4.7. Die interessante offene Frage ist, ob  $\text{BPP} \neq \text{BQP}$  gilt oder ob randomisierte Algorithmen Quantenalgorithmen effizient simulieren können, worauf zurzeit nichts hinweist.

## 4.5 Die Churchsche These

Es gibt verschiedene Möglichkeiten, den Begriff Algorithmus zu präzisieren. In Abschnitt 2.1.1 haben wir die Turingmaschinen als Berechnungsmodell kennengelernt. Aus dem gleichen Jahrzehnt – den 30er Jahren des 20. Jahrhunderts – stammt ein anderes Berechnungsmodell, der Lambda-Kalkül des amerikanischen Logikers Alonzo Church. Es stellte sich heraus, dass beide Berechnungsmodelle äquivalent sind: es sind jeweils dieselben Probleme berechenbar. Und es gibt noch mehr äquivalente Modelle, wie zum Beispiel die Registermaschine.

Wir formulieren die Churchsche These (oder Church–Turing–These) als: *Jede im intuitiven Sinne berechenbare Funktion ist mit einer Turingmaschine*

*berechenbar*. Anschaulich lässt sich danach zu jeder hinreichend exakt formulierten Rechenvorschrift eine Turingmaschine angeben, die diese Rechnung ausführt. Für die Informatik spielt die Churchsche These eine wesentliche Rolle: Ihr zufolge kann man sich bei der Untersuchung von Berechnungen auf *ein* Modell beschränken, das heißt der Begriff Berechnung ist modellunabhängig. Beweisbar ist die Churchsche These in dieser Formulierung nicht, sie ist auch gar kein mathematischer Satz. Der Begriff *intuitiv berechenbar* verweist auf die Vermutung, dass sich zukünftig beschriebene Berechnungsmodelle ebenfalls von Turingmaschinen simulieren lassen werden.

Jüngeren Datums sind *physikalische* Formulierungen, die den Begriff *intuitiv berechenbar* eliminieren. Eine mögliche Formulierung: *Jede durch einen physikalisch umsetzbaren Algorithmus berechenbare Funktion ist mit einer Turingmaschine berechenbar*, siehe etwa [116] für eine Diskussion verschiedener Ausprägungen. Damit kann die Churchsche These als physikalische Aussage gedeutet werden.

Keine der bisher genannten Versionen der Churchschen These verliert durch die Möglichkeit, Quantenrechner zu bauen, ihre Gültigkeit. Die Quantenmechanik lässt sich mit klassischen mathematischen Begriffen beschreiben und kann auf einem klassischen Rechner simuliert werden (allerdings mit großem Aufwand). Auf diese Weise wird auch die Aussage  $BQP \subseteq PSPACE$  aus dem letzten Abschnitt bewiesen.

Quanten-  
computer und  
Berechenbarkeit

Mit Quantenrechnern lassen sich dieselben Funktionen berechnen wie mit klassischen Rechnern.

Quantitative  
Version

Wenn nun etwas grundsätzlich berechenbar ist, ist weiterhin interessant, ob so eine Berechnung auch *effizient* ausführbar ist. Als effizient sehen

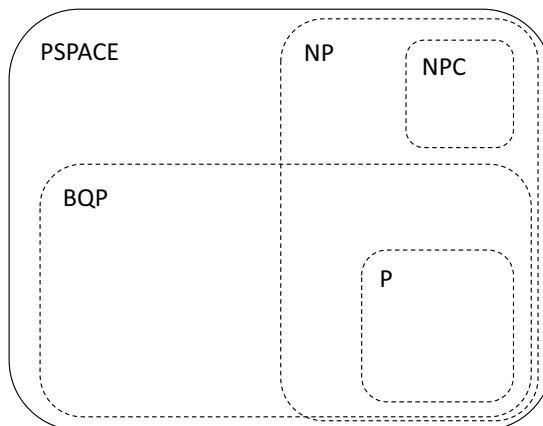


Abbildung 4.7: Einordnung von BQP.

wir Polynomialzeit-Berechnungen an. In Abschnitt 4.2 haben wir unser Blickfeld etwas erweitert und den Nutzen von randomisierten Polynomialzeit-Algorithmen mit beschränktem Fehler gesehen. Damit lässt sich die *quantitative Church–Turing-These* formulieren: *Jede Berechnung lässt sich effizient durch eine probabilistische Turingmaschine simulieren.*

Diese Version wird durch die neuen Quantenrechner allerdings herausgefordert. Vermutlich können klassische randomisierte Turingmaschinen Quantenrechner nicht effizient simulieren. Der Algorithmus von Shor, siehe Kapitel 8, löst ein Problem, für das kein effizienter klassischer Algorithmus bekannt ist.

## 5 Teleportation und dichte Kodierung

*Take me apart, take me apart, You must  
be off your head,  
And if you try to take me apart to get  
me there,  
I'll stay right here in bed.*

*I teleported home one night,  
With Ron and Sid and Meg,  
Ron stole Meggie's heart away,  
And I got Sidney's leg.  
Douglas Adams*

Unter Teleportation versteht man die Überwindung des Raumes, ohne dass Zeit vergeht oder ein Weg zurückgelegt wird. Ein Teleportationsverfahren ist durch die Serie *Raumschiff Enterprise* populär geworden. *Scotty, beam me up, there is no intelligent life on this planet!* ist unter Science Fiction Fans zu einem geflügelten Wort geworden. Auf dieses Kommando hin wird eine exakte Beschreibung Captain Kirks ermittelt. Mit diesen Informationen wird die Person an Bord des Raumschiffs wieder zusammengesetzt, während sie zeitgleich von dem trostlosen Planeten verschwindet (eine ähnliche Situation illustriert Abbildung 5.1).

Verschiedene Argumente legen nahe, dass derartige Teleportationen außerhalb der Science Fiction ausgeschlossen sind. Nur eines davon: Teleportation verstößt gegen eine berühmte Folgerung der Relativitätstheorie: Nichts kann sich schneller bewegen als das Licht! In der Quantenwelt sind Teleportationen jedoch möglich, und wir werden sehen, inwiefern sie mit der Relativitätstheorie verträglich sind.

Quantenteleportation kann ein Quantenbit von einem Ort direkt zu einem anderen befördern, ohne dass der Zustand des Quantenbits bekannt sein muss. Dazu wird die starke Kopplung zwischen verschränkten Quantenbits

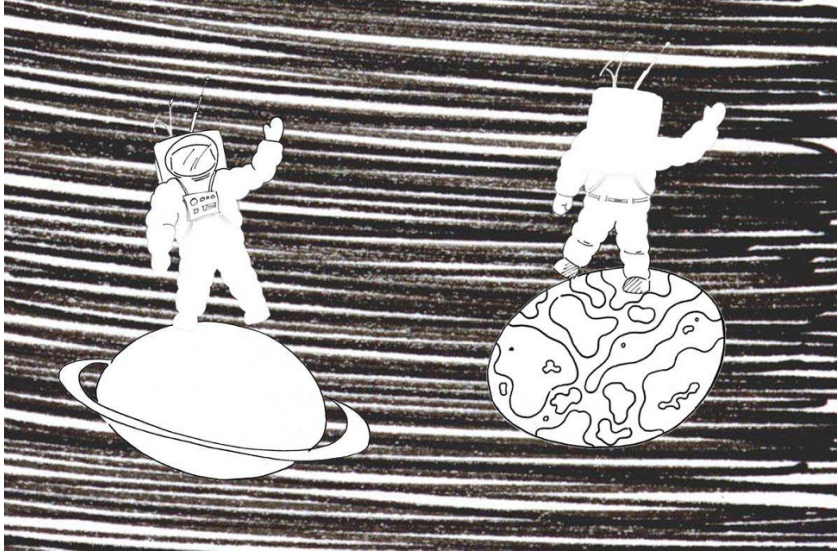


Abbildung 5.1: Teleportation eines Weltraumreisenden; technisch möglich ist die Teleportation einzelner Quantenbits.

genutzt, zusätzlich ist allerdings eine klassische Informationsübertragung nötig. Durch diese wird ein Konflikt mit der Relativitätstheorie vermieden, auf klassische Weise allein kann der Zustand eines Quantenbits jedoch nicht kommuniziert werden.

Die im folgenden Abschnitt vorgestellte Technik ist nicht nur sehr überraschend, sondern bietet viele praktische Anwendungen: so etwa zur Informationsübertragung zwischen zwei Quantenrechnern oder innerhalb eines solchen.

Ein zentraler Begriff der Theorie der Informationsübertragungen ist der *Kanal*.

## Quantenkanal

Ein *Quantenkanal* kann ein Quantenbit von einem Ort an einen anderen transportieren; vom Beginn des Quantenkanals an sein Ende.

Ein Quantenkanal könnte eine Glasfaserleitung sein, die Photonen transportiert. Entscheidend ist, dass die Superposition der für  $|0\rangle$  und  $|1\rangle$  gewählten Zustände (im Idealfall) nicht beeinflusst wird. Allgemein sind Quantenkanäle so abstrakt wie möglich aufzufassen. Ist es etwa möglich, ein Quantenbit auf einem Rollwagen von einem Labor ins andere zu fahren, so existiert zwischen Ausgangs- und Zielpunkt ein Quantenkanal.

Ein *klassischer Kanal* kann klassische Bits transportieren. Klassische Kommunikationsverfahren gibt es in großer Zahl. Telefon oder Postkarten können hier genauso wie reitende Boten oder DFÜ-Verbindungen genannt werden.

Ein Kanal heißt *verrauscht*, wenn die Bits während des Transports Störungen ausgesetzt sind. Ein verrauschter Quantenkanal liefert ein Quantenbit  $\alpha|0\rangle + \beta|1\rangle$  in der Form  $\alpha'|0\rangle + \beta'|1\rangle$  beim Adressaten ab, wobei zu hoffen ist, dass  $\alpha'$  und  $\beta'$  von  $\alpha$  und  $\beta$  nicht zu sehr abweichen.

## 5.1 Quantenteleportation

Alice ist im Besitz eines Quantenbits und möchte dieses Bob mitteilen. Leider gibt es zu diesem Zeitpunkt keinen Quantenkanal zwischen den beiden. Alice und Bob können lediglich telefonieren. Damit sind sie über einen klassischen Kanal verbunden; der Versuch, das Quantenbit damit zu übertragen muss allerdings scheitern. Um etwas über das Quantenbit herauszubekommen, muss Alice es messen. Dabei wird der Zustand des Quantenbits eventuell zerstört. In Abschnitt 2.8 haben wir gesehen, dass es prinzipiell unmöglich ist, den genauen Zustand eines Qubits zu ermitteln.

Besitzen Alice und Bob jedoch jeweils ein Bit eines *verschränkten* Paares von Quantenbits – die Hälfte eines EPR-Paares – so ist folgendes möglich: Alice verschränkt das zu übermittelnde Quantenbit mit ihrer Hälfte des EPR-Paares. Sie misst ihre Bits, und ohne zeitlichen Abstand ändert sich das Quantenbit in Bobs Besitz. Dieses Verhalten haben wir bereits in Abschnitt 2.11 über Verschränkung studiert. Um die Teleportation zu vollenden, muss Alice ihr Messergebnis an Bob schicken; dazu genügt jedoch ein klassisches Kommunikationsmedium, zum Beispiel ein Telefon.

### Der Algorithmus zur Quantenteleportation

**Aufgabenstellung:** Alice besitzt ein Quantenbit  $|x\rangle$  im Zustand  $|\psi\rangle$ . Dieses Qubit soll Bob übermittelt werden.

**Voraussetzung:** Alice besitzt ein Quantenbit  $|a\rangle$  und Bob ein Quantenbit  $|b\rangle$ , die sich in dem verschränkten Zustand  $|ab\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  befinden. Alice kann Bob zwei klassische Bits senden.

**Ergebnis:** Nach Ausführung der folgenden Schritte befindet sich Bobs Bit im Zustand  $|\psi\rangle$ .

$X$  ist dabei der Bitflip

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

und  $Z$  der Phaseflip

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

siehe Abschnitt 3.3.



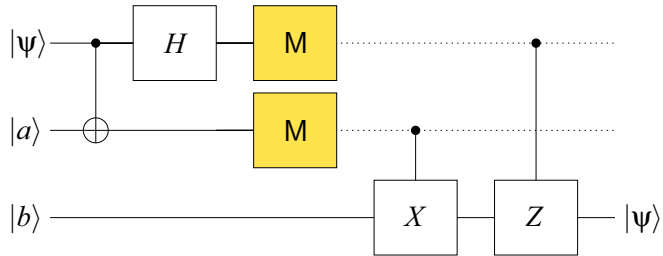


Abbildung 5.2: Schaltkreis zur Teleportation. Die gebrochene Linie stellt einen klassischen Kanal dar

1. Alice wendet ein CNOT-Gatter an:  
 $|x\rangle|a\rangle \leftarrow |x\rangle|a \oplus x\rangle$
2. Alice wendet auf das zu übermittelnde Bit die Hadamard-Transformation an:  
 $|x\rangle \leftarrow H|x\rangle$
3. Alice misst ihre Bits und übermittelt die Ergebnisse  $x$  und  $a$  über einen klassischen Kanal an Bob.
4. Ist  $a = 1$ , wendet Bob den Bitflip  $X$  auf sein Bit an:  
 $|b\rangle \leftarrow X|b\rangle$
5. Ist  $x = 1$ , wendet Bob den Phaseflip  $Z$  auf sein Bit an:  
 $|b\rangle \leftarrow Z|b\rangle$

Dem Algorithmus entspricht der Schaltkreis aus Abbildung 5.2. Die linke Hälfte ist der Anteil von Alice, die rechte ist Bobs Berechnung. Die gebrochene Linie entspricht der klassischen Übermittlung der Messergebnisse an Bob. An dem Schaltkreis wird außerdem deutlich, dass das Bit  $|b\rangle$  zu Bob gelangen muss. Dazu bedarf es im Prinzip eines Quantenkanals; allerdings kann dieses Bit schon lange Zeit vor der Teleportation transportiert worden sein. Wir stellen uns vor, Bob hat es beim Abschied von Alice mitgenommen.

Es ist auch möglich, dass Alice und Bob nie über einen Quantenkanal verbunden waren, wenn eine dritte Person namens Charlotte Quantenbits an Alice und Bob schicken kann. Charlotte hat irgendwann ein EPR-Paar erzeugt und die Hälften Alice und Bob übermittelt.

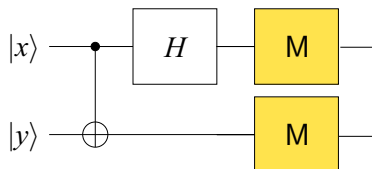


Abbildung 5.3: Bell-Messung

Die von Alice ausgeführten Operationen gemäß Abbildung 5.3 bezeichnet man auch als *Bell-Messung*. In Aufgabe 2.20 haben wir  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$  in die vier Bell-Zustände überführt. Dazu genügte es,  $H$  auf Bit 1 gefolgt von CNOT anzuwenden. In Abbildung 5.3 werden zunächst diese Schritte in umgekehrter Reihenfolge ausgeführt und damit rückgängig gemacht.  $|\Phi^+\rangle$  wird auf  $|00\rangle$ ,  $|\Psi^+\rangle$  auf  $|01\rangle$ ,  $|\Phi^-\rangle$  auf  $|10\rangle$  und  $|\Psi^-\rangle$  wird auf  $|11\rangle$  abgebildet.

Also ergibt sich mit der abschließenden Messung in der Standardbasis  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$  eine Messung in der Bell-Basis.

### Analyse

Bei verschränkten Quantenbits spielt die räumliche Distanz keine Rolle. Wir betrachten bei der Analyse stets den Zustand des „Registers“  $|x\rangle|a\rangle|b\rangle$ . Es gelte  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ . Dann befindet sich das Register  $|x\rangle|a\rangle|b\rangle$  anfangs im Zustand

$$\begin{aligned} |\phi_0\rangle &= (\alpha|0\rangle + \beta|1\rangle) \cdot \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ &= \frac{\alpha}{\sqrt{2}}(|000\rangle + |011\rangle) + \frac{\beta}{\sqrt{2}}(|100\rangle + |111\rangle). \end{aligned}$$

Im ersten Schritt wendet Alice CNOT mit dem Folgezustand

Alice' Schritte

$$|\phi_1\rangle = \frac{\alpha}{\sqrt{2}}(|000\rangle + |011\rangle) + \frac{\beta}{\sqrt{2}}(|110\rangle + |101\rangle)$$

an. Die Hadamard-Transformation überführt  $|\phi_1\rangle$  in

$$\begin{aligned} |\phi_2\rangle &= \frac{\alpha}{\sqrt{2}} \left( \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \right) (|00\rangle + |11\rangle) \\ &\quad + \frac{\beta}{\sqrt{2}} \left( \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right) (|10\rangle + |01\rangle) \\ &= \frac{\alpha}{2} (|000\rangle + |011\rangle + |100\rangle + |111\rangle) \\ &\quad + \frac{\beta}{2} (|010\rangle + |001\rangle - |110\rangle - |101\rangle). \end{aligned}$$

Wir stellen den Term so um, dass wir das Ergebnis der Messung der ersten beiden Bits erkennen:

$$\begin{aligned} |\phi_2\rangle &= \frac{1}{2} \left( |00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\beta|0\rangle + \alpha|1\rangle) \right. \\ &\quad \left. + |10\rangle(\alpha|0\rangle - \beta|1\rangle) - |11\rangle(\beta|0\rangle - \alpha|1\rangle) \right). \end{aligned}$$

Alice' Messung im dritten Schritt ergibt jeweils mit Wahrscheinlichkeit  $1/4$  eines der Ergebnisse  $|00\rangle, |10\rangle, |01\rangle$  oder  $|11\rangle$ . Gleichzeitig geht das dritte Bit – im Besitz von Bob – in einen an Alice' Ergebnis gekoppelten Zustand über.

Alice' Ergebnis	Zustand von Bobs Bit
$ 00\rangle$	$\alpha 0\rangle + \beta 1\rangle$
$ 01\rangle$	$\alpha 1\rangle + \beta 0\rangle$
$ 10\rangle$	$\alpha 0\rangle - \beta 1\rangle$
$ 11\rangle$	$\alpha 1\rangle - \beta 0\rangle$

**Bobs Schritte**            Erhält Alice das Ergebnis  $|00\rangle$ , ist Bobs Bit durch die Messung in den Zustand  $|\psi\rangle$  übergegangen, und die Teleportation ist vollzogen. In den anderen Fällen kann Bob den Zustand  $|\psi\rangle$  leicht herstellen, vorausgesetzt er kennt das Ergebnis der Messung. Das trifft zu: es wurde ihm über einen klassischen Kanal mitgeteilt. Ist Alice' Ergebnis  $|01\rangle$ , muss er die Amplituden von  $|0\rangle$  und  $|1\rangle$  tauschen. Das gelingt ihm mit dem Bitflip  $X$ . Im Fall  $|10\rangle$ , negiert er mit  $Z$  das Vorzeichen der Amplitude von  $|1\rangle$ , und im Fall  $|11\rangle$  führt er  $X$  und  $Z$  aus.◊

Alice muss Bob für den Vollzug der Teleportation etwas mitteilen. Das hat etwas Beruhigendes. Denn nach Einsteins Relativitätstheorie kann sich nichts schneller bewegen als das Licht. Ein Transport von Information ohne Zeitverzug würde diesem Grundprinzip widersprechen: *information is physical*, siehe Abschnitt 3.5. Bei unserem Verfahren kann Bob auf die unmittelbar übermittelten Informationen erst zugreifen, wenn er das langsamer als das Licht übermittelte Messergebnis kennt.

Alice kann nach ihrer Messung den Zustand  $|\psi\rangle$  nicht mehr rekonstruieren. Ein anderes Ergebnis stünde auch im Widerspruch zu dem Ergebnis des Kapitels 3.4: Quantenbits lassen sich nicht kopieren.

**Realisierung**            Das Verfahren zur Quantenteleportation stammt von der international zusammengesetzten Forschergruppe Bennett, Brassard, Crépeau, Jozsa, Peres und Wootters und ist aus dem Jahr 1993, [19]. Die erste praktische Umsetzung gelang 1997 in Innsbruck. Bouwmeester, Pan, Mattle, Eibl, Weinfurter und Zeilinger teleportierten ein Quantenbit etwa einen Meter weit, [25]. Die Zustände  $|0\rangle$  und  $|1\rangle$  wurden als Polarisationen eines Photons verwirklicht, siehe Abschnitt 10.2.

Heutzutage wird Quantenteleportation als Standardverfahren angesehen, dessen einzige Grenze durch die Distanz der verschränkten Bits gegeben ist. Bei Teleportationen über größere Entfernungen und außerhalb des Labors wird die Verschränkung der Bits  $|a\rangle$  und  $|b\rangle$  zum Problem. Theoretisch ist das Verhalten verschränkter Quantenbits von ihrer Entfernung völlig unabhängig. In der Praxis entstehen Probleme beim Transport und durch Wechselwirkung mit der Umgebung: reale Quantenkanäle sind immer verrauscht.

Abhilfe schafft ein Verfahren namens *entanglement purification*, *Verschrankungsreinigung*, auch als *Verschrankungsdestillation* bezeichnet. Wir nehmen an, Alice stellt ein verschränktes Paar von Quantenbits her und schickt eines davon an Bob. Ihr steht ein *verrauschter Kanal* zur Verfügung: das Quantenbit ist Störungen ausgesetzt und kommt verändert bei Bob an. Dadurch sind die beiden Bits eventuell nicht mehr maximal verschränkt. Zur Verschrankungsreinigung wiederholt Alice dieses Verfahren. Aus zwei nicht maximal verschränkten Paaren stellt sie eines her, dessen Grad an Verschrankung deutlich höher ist, siehe etwa [21] oder [75].

Verschrankungs-  
reinigung

Theoretisch lässt sich so aus einem verrauschten Quantenkanal und einem klassischen Kanal ein Quantenkanal nahezu ohne Rauschen konstruieren. Dazu werden über den verrauschten Kanal so viele Hälften von EPR-Paaren geschickt, dass sich daraus mit Hilfe der Quantenreinigung ein sehr stark verschränktes Paar von Quantenbits erzeugen lässt. Die beiden Hälften dieses Paares befinden sich an den beiden Enden des verrauschten Kanals. Das eben beschriebene Verfahren zur Quantenteleportation ist anwendbar: Wir können ein Quantenbit mit nur geringem Fehler übertragen, und ein Kanal mit geringem Rauschen ist entstanden.

2017 gelang einem chinesischen Team die Teleportation eines Quantenbits von der Erdoberfläche in den Satellitenorbit, siehe [127]. Dabei wurde die Distanz zwischen einer in Tibet auf einer Höhe von 5100 m platzierten Station zu einem je nach Position 500 - 1400 km entfernten Satelliten überbrückt. Das erklärte Ziel war es, satellitengestützte Teleportation in ein landesweites Quanten-Netzwerk zu integrieren, mehr dazu am Ende dieses Abschnitts.

Das Quanteninternet würde die Kopplung von Quantenrechnern ermöglichen sowie die Anwendung von Quantenkryptographieverfahren, siehe Kapitel 7. Dazu bietet sich die Umsetzung von Qubits durch Photonen an, die über Glasfaserleitungen ausgetauscht werden. Zu großen technischen Herausforderungen führt, dass die Fehlerrate exponentiell mit der Länge der Leitung wächst. Repeater-Techniken, die bei klassischer Kommunikation über dieses Medium Anwendung finden, verbieten sich aufgrund des No-Cloning-Theorems, siehe Abschnitt 3.4. Die theoretischen Gegenmittel sind allerdings bereits bekannt. In [29] und [30] wird der Aufbau eines Quanten-Repeaters vorgeschlagen. Dazu wird Verschrankungsreinigung mit *Entanglement swapping* kombiniert.

Quanteninternet

Dieser Verschrankungsaustausch ähnelt der Teleportation: Stellen wir uns vor, dass die Entfernung zwischen Alice und Charlotte ausreichend klein ist, um diese mit verschränkten Qubits auszustatten und das gleiche gilt für die Distanz zwischen Charlotte und Bob. Es teilen sich nun Alice und Charlotte ein EPR-Paar und genauso Charlotte und Bob. Dann lässt sich Verschrankung zwischen Alice und Bob erzeugen, ohne dass diese jemals direkt in Kontakt getreten sind. Dazu nimmt Charlotte eine Bell-Messung an ihren beiden Bits vor, so wie Alice in unserem Teleportationsprotokolls, siehe Seite 127 und Abbildung 5.3. Charlotte nimmt hier die Rolle des Repeaters ein. Mit einer Folge derartiger Repeater ließe sich theoretisch Verschrankung über beliebige weite Strecken erzeugen.

## Entanglement Swap

Die Teilnehmer und ihre Qubits:

Alice	Charlotte	Bob
$a$	$c_1, c_2$	$b$

1. Zu Beginn ist  $a$  mit  $c_1$  verschränkt und  $c_2$  mit  $b$ .
2. Charlotte misst ihre Bits  $c_1$  und  $c_2$  in der Bell-Basis.
3. Danach sind  $a$  und  $b$  verschränkt.

Die Umsetzung dieser Idee sieht sich allerdings noch technischen Schwierigkeiten gegenüber. Zur Drucklegung dieser Auflage existieren verschiedene Quantennetzwerke, die sogenannte *trusted repeater* verwenden. Diese sind zum Schlüsselaustausch (siehe Kapitel 7) geeignet, müssen allerdings perfekt abgesichert sein. Das ist bei den eben beschriebenen Quanten-Repeatern nicht der Fall, mit diesen wäre Ende-zu-Ende-Verschlüsselung möglich. Zum Beispiel sind seit 2017 Peking und Shanghai über eine 2000 km lange Glasfaserleitung mit 32 solchen *trusted nodes* verbunden. In 2021 wurde durch Satellitenunterstützung die Distanz von 4600 km überwunden ([37]).

Der Einstieg in das Thema Quanteninternet ist zum Beispiel durch [147] oder [118] möglich.

## 5.2 Dichte Kodierung

Das in diesem Abschnitt vorgestellte Verfahren hat mit der Teleportation große Ähnlichkeit. Die Aufgabe ist allerdings eine andere.

- Eine Teleportation überträgt ein Quantenbit und nutzt dazu einen klassischen Kanal (Voraussetzung ist, dass die Kommunikationspartner ein verschränktes Bitpaar teilen).
- *Dichte Kodierung* macht es möglich, mit Hilfe eines Quantenkanals und eines Quantenbits *zwei klassische Bits* zu übertragen.

Bei dieser Art von Kodierung geht es um „Komprimierung“: von zwei klassischen Bits auf ein Quantenbit. Uns interessieren weniger mögliche Anwendungen, sondern die Erkenntnis: mit einem Quantenbit lässt sich unter bestimmten Umständen die doppelte Informationsmenge übermitteln, wie mit einem klassischen Bit.

### Der Algorithmus zur dichten Kodierung

Alice möchte eine der vier Botschaften 00, 01, 10 und 11 an Bob übermitteln. Alice besitzt ein Quantenbit  $|a\rangle$  und Bob ein Quantenbit  $|b\rangle$ , die sich in dem verschränkten Zustand  $|ab\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  befinden.

Die Transformation  $iY$  entspricht der Matrix

$$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

1. *Kodierung*: Alice wendet je nach zu übermittelnder Nachricht  $N$  eine der folgenden Operationen auf ihr Bit an:

Ist  $N = 01$ :  $|a\rangle \leftarrow Z|a\rangle$

Ist  $N = 10$ :  $|a\rangle \leftarrow X|a\rangle$

Ist  $N = 11$ :  $|a\rangle \leftarrow iY|a\rangle$

Ist  $N = 00$ : sie tut nichts.

2. Alice schickt ihr Bit  $|a\rangle$  an Bob.
3. Bob misst die beiden Bits  $|a\rangle|b\rangle$  bezüglich der Bell-Basis (Abschnitt 2.11).
4. *Dekodierung*: Bob erschließt Alice' Nachricht wie folgt aus dem Messergebnis:

$$\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \text{ bedeutet } N = 01,$$

$$\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \text{ bedeutet } N = 10,$$

$$\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \text{ bedeutet } N = 11,$$

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \text{ bedeutet } N = 00.$$

**Aufgabe 5.1:** Analysieren Sie den Algorithmus und überprüfen Sie so dessen Korrektheit.

**Beispiel 5.1:** Alice befindet sich auf einer Wetterstation und muss Bob mitteilen, woher der Wind weht. Es genügt, wenn sie Bob eine der vier Nachrichten *Norden*, *Süden*, *Osten*, *Westen* schickt. Genauer braucht die Richtung nicht angegeben zu werden. Wie zu erwarten war, verfügt Alice über keine klassische Kommunikationsmöglichkeit. Aber Sie ist in der Lage, ein einzelnes Quantenbit an Bob zu schicken. Und vor einigen Monaten haben sie gemeinsam ein EPR-Paar  $|ab\rangle$  mit Zustand  $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  erzeugt, aufgeteilt, und beide tragen ihre Hälfte immer bei sich. Zudem gibt es zwischen beiden folgende Vereinbarung: In ihren Kommunikationen steht 00 für *Norden*, 01 für *Süden*, 10 für *Osten*, und 11 für *Westen*.

Es ist Ostwind. Dafür stehen die beiden klassischen Bits 10. Wie im Algorithmus beschrieben, wendet sie auf ihr Bit die Transformation  $X$  an. Der gemeinsame Zustand des Paares  $|ab\rangle$  wechselt zu

$$\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle).$$

Alice schickt  $|a\rangle$  an Bob. Der misst beide Bits in der Bell-Basis und erhält das Ergebnis  $\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$ . Gemäß Schritt 4 schließt er: der Wind weht aus Osten.  $\diamond$

Klassische  
Informations-  
übermittlung

Zu diesem Verfahren gibt es kein klassisches Gegenstück. Die Aufgabenstellung, mehr als ein Bit an klassischer Information mit einem klassischen Bit zu übermitteln, ist in sich widersprüchlich und damit sinnlos. Denn der Begriff „Bit“ hat eine doppelte Bedeutung. Es ist üblich, damit einen elementaren – nicht weiter zerlegbaren – Informationsträger zu bezeichnen. Den gleichen Namen trägt die Einheit, mit der man die Informationsmenge angibt. Diese wollen wir „bit“ schreiben. Ein bit ist die maximale Informationsmenge, die mit zwei Zuständen eines (klassischen) physikalischen Systems gespeichert, ausgedrückt oder übermittelt werden kann. Das wollen wir an einem Beispiel verdeutlichen.

**Beispiel 5.2:** Mit einem Bit lassen sich zwei Zustände kodieren und mit  $n$  Bits gerade  $2^n$  viele. So genügen für die vier Himmelsrichtungen aus Beispiel 5.1 zwei Bits. Wir nehmen an, Alice und Bob verwenden aus uns nicht bekannten Gründen drei Bits und folgende Kodierung 000 für *Norden*, 010 für *Süden*, 101 für *Osten*, und 110 für *Westen*. In diesem Fall werden mit einer Nachricht trotzdem nur zwei bit an Information übertragen. Genauer gesagt: maximal zwei bit. Dieses Maximum wird angenommen, wenn alle vier Windrichtungen gleich wahrscheinlich sind. Wenn Bob vor der Nachricht in völliger Unwissenheit über die Windrichtung ist. Angenommen Alice' Wetterstation befindet sich in einer Schlucht, durch die der Wind nur aus Norden oder aus Süden hindurchwehen kann. In diesem Fall ist der Informationsgehalt von Alice Nachricht nur ein bit, egal was für eine Kodierung sie wählt.  $\diamond$

Das in vorstehendem Beispiel erwähnte Informationsmaß ist Gegenstand der von dem amerikanischen Mathematiker Claude Shannon (1916-2001) begründeten Informationstheorie. Während man zur Kodierung von vier Nachrichten zwei klassische Bits benötigt, kommt man mit einem Quantenbit aus; allerdings nur, wenn sich die Kommunikationspartner ein EPR-Paar teilen. Das Verfahren der dichten Kodierung stammt von Charles Bennett und Stephen Wiesner [22].

## 5.3 Verschränkte Bits

Teleportation und dichte Kodierung sind aus klassischer Sicht sehr erstaunliche Verfahren. Beide beruhen auf der Verwendung verschränkter Bits. In diesem Abschnitt werden wir folgende Eigenschaft verschränkter Bits untersuchen:

Verteilte  
Information

In einem Paar maximal verschränkter Quantenbits ist keines allein Träger der Information. Diese besteht in der Korrelation beider Bits.

Angenommen, Alice besitzt zwei Quantenbits  $|x\rangle|y\rangle$  und weiß, dass diese in einem der Basiszustände  $|00\rangle, |01\rangle, |10\rangle, |11\rangle$  sind. Dann erfährt sie aus einer Messung in dieser Basis den exakten Zustand. Allgemein gilt für jede Basis: weiß Alice, dass sich zwei Quantenbits in einem Basiszustand befinden, kann sie den genauen Zustand mit einer Messung in dieser Basis ermitteln. Nun kommen Bob und Charlotte ins Spiel. Charlotte erzeugt Zwei-Bit-Zustände in der Basis  $|00\rangle, |01\rangle, |10\rangle, |11\rangle$  oder in einer anderen Orthonormalbasis aus unverschränkten Zuständen. Eines erhält Alice, das andere Bob. Jeder misst sein Bit in der Basis  $|0\rangle, |1\rangle$ ; auf diese Weise können sie genau ermitteln, welchen der vier Werte das Bitpaar hat, wenn sie ihre Ergebnisse zum Beispiel über einen klassischen Kanal austauschen. Die Bits sind unabhängig voneinander. Die Messungen enthüllen die kodierte Information.

Als nächstes erzeugt Charlotte Bitpaare in einem der verschränkten Zustände der Bell-Basis:

$$\begin{aligned} |\Phi^+\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), & |\Phi^-\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \\ |\Psi^+\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), & |\Psi^-\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \end{aligned}$$

Das erste Bit schickt sie an Alice, das zweite an Bob.

Wenn Alice und Bob ihre Bits zusammensetzen und gemeinsam in der Bell-Basis messen, können sie problemlos den genauen Zustand ermitteln. Das geht aber nicht mehr, wenn sie sich an unterschiedlichen Orten befinden und jeder nur eine Messung an seinem Bit vornehmen kann.

Wir nehmen an, Charlotte hat die beiden Bits in den Zustand  $|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$  versetzt und verteilt. Alice und Bob wissen, dass Charlotte einen Zustand aus der Bell-Basis verteilt hat, und sollen herausfinden, welchen. Alice misst in der Standardbasis  $|0\rangle, |1\rangle$ : die Ergebnisse  $|0\rangle$  und  $|1\rangle$  sind gleich wahrscheinlich. Bob misst sein Bit ebenso und erhält gerade das zu Alice' Ergebnis entgegengesetzte Resultat. Sie telefonieren und schließen: der Zustand ist  $|\Psi^+\rangle$  oder  $|\Psi^-\rangle$ . Welcher der beiden Zustände es ist, muss ihnen verborgen bleiben. Wie sie auch die Messungen vornehmen und egal, welche Informationen sie austauschen, sie können nur feststellen, ob die Bits korreliert ( $|\Phi^+\rangle$  und  $|\Phi^-\rangle$ ) oder antikorreliert ( $|\Psi^+\rangle$  und  $|\Psi^-\rangle$ ) sind.

Und welche Information enthält ein einzelnes Quantenbit eines maximal verschränkten Paares? Keine! Wenn Charlotte einen der Zustände der Bell-Basis verteilt hat und Alice ihr Quantenbit misst, ist das Ergebnis ein Zufallsbit. Sie erhält keinen Hinweis darauf, in welchem Zustand sich die beiden Quantenbits befinden!

Wir formulieren das letzte Ergebnis als:

Eine Hälfte eines maximal verschränkten Quantenbitpaares enthält keinerlei Information.



## Holevoschranke

Aus einer Messung an nur einem der Bits erfährt man in diesem Fall nichts. Die Information befindet sich bildlich gesprochen *zwischen* den Bits, sie besteht in der Korrelation. Einem Satz der Quanteninformationstheorie zufolge lässt sich mit einem Quantenbit nur ein bit<sup>1</sup> Information übertragen: dieses Ergebnis aus dem Jahr 1973 trägt den Namen *Holevoschranke*, benannt nach Alexander Holevo vom Moskauer Steklov Institut. Wie verträgt sich diese Aussage mit der dichten Kodierung, bei der zwei bit, also der Informationsgehalt von zwei klassischen Bits, mit nur einem Quantenbit übermittelt werden können?

Die Holevoschranke beschreibt die Kapazität eines Quantenkanals, über den ein einzelnes Quantenbit verschickt wird, und macht eine Aussage über die auf diese Weise übermittelbare Information. Für die dichte Kodierung ist es wesentlich, dass sich Alice und Bob bereits ein verschränktes Bitpaar teilen. Dieses muss zuvor ausgetauscht worden sein, und damit werden bei der dichten Kodierung zwei Quantenbits ausgetauscht. Aus klassischer Sicht ist es trotzdem erstaunlich, dass es verschränkte Zustände erlauben, einen Teil des Informationsaustauschs auf diese Weise vorzuverlegen. Das erste Bit kann zu einem Zeitpunkt verschickt werden, zu dem die zu verschickende Nachricht noch nicht bekannt ist.

---

<sup>1</sup>Siehe S. 132 unter *Klassische Informationsübermittlung*.

## 6 Suchen

*Suchen und finden schließen einander so lange im Moment der Enttäuschung aus, bis sie sich im Moment der (Er-)Findung einschließen.*

Olaf Kuschniok

Suchen ist eine menschliche Grundtätigkeit. Zerstreute Menschen suchen ihren Zimmerschlüssel, alle Menschen suchen (zumindest insgeheim) das Glück. Der Kommissar sucht den Mörder, und in dem 4000 Jahre alten babylonischen Gilgamesch-Epos macht sich die Titelfigur auf die Suche nach der Unsterblichkeit. Für Informatiker stellt sich diese Aufgabe prosaischer dar. Dort muss meist aus einer Menge von Datensätzen derjenige mit einer bestimmten Eigenschaft gesucht werden. Suchvorgänge dieser Art verbrauchen in heutigen EDV-Systemen sehr viel Rechenzeit.

Die Fähigkeiten klassischer Rechner sind unter anderem dadurch begrenzt, dass diese nicht schneller suchen können. Denn viele Berechnungsprobleme stellen sich letzten Endes als Suchprobleme heraus. So ist der Versuch, in ein fremdes Sicherheitssystem einzubrechen, meistens die Suche nach dem Schlüssel. Ein anderes Beispiel sind *Optimierungsprobleme*, die für ökonomische und technische Anwendungen eine sehr große Rolle spielen, wie etwa das Problem des Handlungsreisenden: Gegeben ist eine Menge von Städten. *Gesucht* ist die kürzeste Route, die alle Städte genau einmal besucht. Allgemein ist ein Optimierungsproblem die Suche nach der optimalen Lösung.

In diesem Kapitel werden wir untersuchen, wie sich mit Quantencomputern suchen lässt. Im Zentrum steht ein berühmtes Resultat, das mit dem Namen Lov Grover verbunden ist. Er hat gezeigt, dass Quantencomputer viel schneller suchen können als klassische Rechner. Gleichzeitig ist sein Suchalgorithmus aus dem Jahr 1996 ein besonders anschauliches Beispiel für die Möglichkeiten von Quantenrechnern. Heute gibt es eine Vielzahl an Ergebnissen zum Thema Quantensuche, die Grovers Grundidee variieren; einige davon werden in diesem Kapitel dargestellt.

Wir erwähnten, dass Quantencomputer viel schneller suchen können als klassische. Aber sie suchen andererseits nicht so viel schneller, dass der

Quantencomputer dadurch zum Allheilmittel wird. Mit Suchalgorithmen kann man nämlich eine sehr wichtige Klasse von Problemen angehen, für die keine effizienten Verfahren für klassische Rechner bekannt sind. Für alle, die sich mit Komplexitätstheorie auskennen oder Kapitel 4 gelesen haben: es handelt sich um NP-vollständige Probleme. Könnten Quantencomputer diese Probleme effizient lösen, hätte das sehr weitreichende Konsequenzen! Leider deutet einiges darauf hin, dass das nicht möglich ist. Das ist das Thema der beiden letzten Abschnitte dieses Kapitels.

## Übersicht

Wir beginnen in Abschnitt 6.1 mit einer Analyse der Aufgabenstellung. Die Abschnitte 6.2 und 6.3 stellen Grovers Algorithmus detailliert dar und Abschnitt 6.4 beschäftigt sich mit Verallgemeinerungen der Grundidee.

## 6.1 Die Nadel im Heuhaufen

Wir wollen herausfinden, wem eine bestimmte Telefonnummer gehört. Wir haben nur die Nummer und ein Telefonbuch zur Verfügung. Wir haben bereits mehrmals die Nummer gewählt, aber es hat sich nie jemand gemeldet. Uns bleibt nichts anderes übrig, als im Telefonbuch Anschluss für Anschluss durchzugehen.

Man bezeichnet diese Aufgabenstellung als *unstrukturierte Datenbanksuche*. Die Einträge der Datenbank sind nach einem Kriterium sortiert, das für uns unbrauchbar ist. So wie es uns in der beschriebenen Situation nichts nützt, dass die Einträge im Telefonbuch nach den Namen sortiert sind. Einen Eintrag der Datenbank darauf zu prüfen, ob es der gesuchte ist, bezeichnen wir als *Anfrage* an die Datenbank.

Suchalgorithmen dieser Art haben viele Anwendungen; in der Einleitung wurde bereits die Kryptographie erwähnt. Einige Verschlüsselungssysteme lassen sich knacken, indem man alle möglichen Schlüssel ausprobiert. Dieses Ausprobieren fassen wir als Datenbankanfrage auf: passt der Schlüssel, ist es der gesuchte Eintrag. Das Ausprobieren des Schlüssels entspricht dem Überprüfen der Nummer hinter einem Namen im Telefonbuch.

## Optimierungsprobleme

Auch Optimierungsprobleme sind im Grunde Suchprobleme. In der Menge aller möglichen Wege, eine Aufgabe zu lösen, suchen wir die optimale. In der Einleitung wurde bereits das *Problem des Handlungsreisenden* erwähnt und in Abschnitt 4.3 genauer betrachtet. Wir müssen eine Reihe von Städten besuchen und am Ende wieder am Ausgangspunkt ankommen. Darum interessieren wir uns für die kürzeste derartige Rundreise. Nun wandeln wir das Problem etwas ab: wir hoffen, dass es für unser Ensemble von Städten eine Rundreise gibt, die kürzer als 100 Kilometer ist. Eine solche *suchen* wir! Für jeden möglichen Reiseplan können wir leicht die Entfernung ausrechnen und somit ermitteln, ob diese Route kürzer 100 Kilometer ist: das entspricht der Datenbankanfrage.

## Aufgabenstellung

Um leichter Algorithmen für diese Art der Datenbanksuche beschreiben zu können, formalisieren wir die Aufgabenstellung: Wir gehen davon aus, dass die Datenbank  $N$  Elemente  $\{0, \dots, N-1\}$  enthält, wobei die Anzahl eine Zweierpotenz  $N = 2^n$  ist. Den Datensätzen ordnen wir die Elemente in  $\{0, 1\}^n$

zu, das gesuchte Element bezeichnen wir als  $\hat{x}$ . Die Datenbank modellieren wir als eine Funktion  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  mit folgender Eigenschaft: es gibt genau ein Element  $\hat{x}$ , für das die Funktion 1 ist, für alle  $x \in \{0, 1\}^n$  mit  $x$  ungleich  $\hat{x}$  gilt  $f(x) = 0$ . Die Funktion – und damit die Datenbank – ist uns als Quantenorakel  $U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle$  gegeben. Wir sollen  $\hat{x}$  bestimmen, eine Datenbankanfrage entspricht dem Auswerten von  $f$ .

Wie die hier diskutierten Arten von Suchalgorithmen zusammenhängen, verdeutlicht die folgende Tabelle.

Wähle ein Datenbankelement.	Frage an, ob es das gesuchte ist.
Wähle einen Telefonbucheintrag.	Ist es die gesuchte Nummer?
Wähle eine Rundreise.	Rechne nach, ob sie kurz genug ist.
Wähle ein $x$ aus $\{0, 1\}^n$ .	Ist $f(x) = 1$ ?

Wie aufwändig ist dieses Problem für klassische Computer? Als *Aufwand* legen wir die Anzahl der Anfragen an die Datenbank fest; für die von uns betrachteten Algorithmen wird diese Anzahl die Laufzeit dominieren. Da wir über  $f$  nur wissen, dass genau ein Element den Funktionswert 1 hat, müssen wir blind nach diesem Element suchen und im schlechtesten Fall  $N - 1$  Elemente betrachten. Mathematisch korrekt überlegen wir uns das folgendermaßen: Wir stellen uns einen klassischen Algorithmus  $A$  vor, der nach  $N - 2$  Aufrufen des Datenbankorakels  $f$  schon das gesuchte Element kennt. Wie kann er das herausgefunden haben? Allein dadurch, dass sich  $\hat{x}$  unter den  $N - 2$  schon betrachteten Elementen befindet. Da  $A$  die Elemente für jede Eingabe in der gleichen Reihenfolge testet – wir haben keine Informationen, die uns eine bestimmte Vorgehensweise ans Herz legen – können wir ein  $f$  angeben, für das dieser fiktive Algorithmus scheitert:  $\hat{x}$  mit  $f(\hat{x}) = 1$  ist eines der Elemente, die der Algorithmus für die ersten  $N - 2$  Anfragen nicht wählt. Damit muss er mindestens  $N - 1$  Elemente ausprobieren.

Aufwand im schlechtesten Fall

Diese Situation, der schlechteste Fall, ist allerdings unwahrscheinlich. In der Regel werden wir nicht so viel Pech haben, dass wir das gesuchte Element als letztes in die Hände bekommen. Wie viele Aufrufe braucht ein klassischer Algorithmus also im Mittel? Die Wahrscheinlichkeit, im ersten Versuch das gesuchte Element zu erwischen, ist  $1/N$ : wir haben keine Informationen, und jedes Element ist mit der gleichen Wahrscheinlichkeit das gesuchte. Die Wahrscheinlichkeit dafür, dass der zweite Versuch erfolgreich ist, ist  $(1 - 1/N) \cdot 1/(N - 1)$ : der erste Versuch war erfolglos und von den verbleibenden  $N - 1$  Möglichkeiten raten wir die richtige. Diese Situation ist in der Wahrscheinlichkeitstheorie unter dem Namen *Ziehen ohne Zurücklegen*

Durchschnittlicher Aufwand

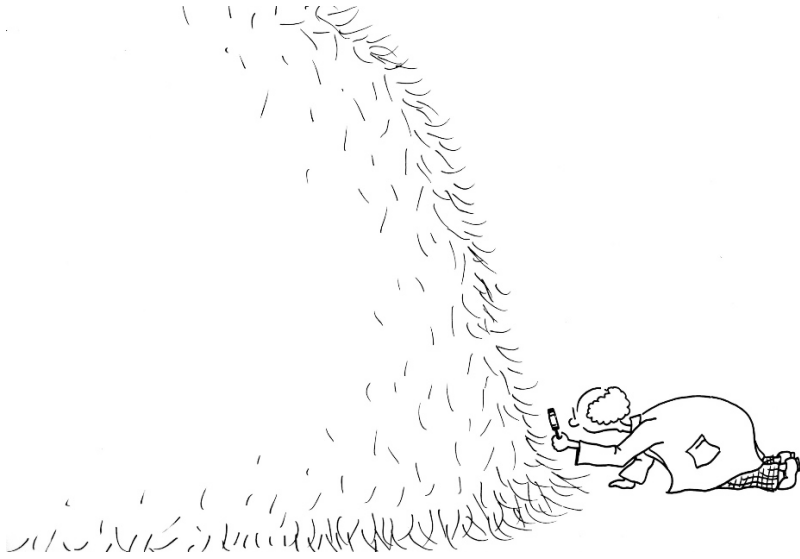


Abbildung 6.1: Unstrukturierte Suche: die Nadel im Heuhaufen

modelliert, siehe zum Beispiel [96]. Im Schnitt benötigt man  $(N+1)/2$  Versuche, asymptotisch (größenordnungsmäßig) ausgedrückt also  $\Omega(N)$ . Nun sieht man leicht, dass die Anzahl der Orakelaufufe die Laufzeit dominiert.

*Bemerkung:* Wir werden an anderer Stelle diskutieren, dass Quantenalgorithmen für die tatsächliche Suche in Datenbanken nur bedingt geeignet sind. Die in diesem Kapitel dargestellten Suchalgorithmen scheinen für andere Aufgaben besser geeignet zu sein. Dennoch hat sich der Begriff *unstrukturierte Datenbanksuche* für die oben beschriebene Aufgabe durchgesetzt, da er die Ausgangssituation auf den Punkt bringt (Abbildung 6.1).

## 6.2 Die Grover-Iteration

Amplituden-  
verstärkung

Ein allgemeiner Ansatz, mit Quantenalgorithmen ein bestimmtes Problem zu berechnen, lautet: Beginne mit einer Superposition möglicher Lösungen für die konkrete Eingabe. Vergrößere nach und nach die Amplitude der tatsächlichen Lösung (genauer: den Betrag der Amplitude) und verkleinere gleichzeitig alle anderen Amplituden. Man nennt diesen Ansatz *Amplitudenverstärkung*, engl. *amplitude amplification*.

Grundidee

Der Suchalgorithmus von Grover setzt diesen Ansatz mustergültig um: Schritt für Schritt wird die Amplitude des gesuchten Elementes  $\hat{x}$  größer. Beginnen wir mit der gleichverteilten Superposition über alle Datenbankelemente:

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle.$$

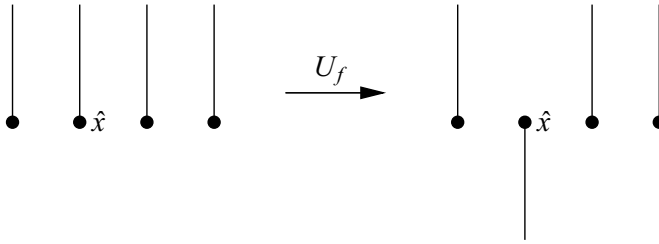


Abbildung 6.2: Das Datenbankorakel kippt die Amplitude von  $\hat{x}$

Alle verfügbaren Informationen über die Datenbank und das fragliche Element  $|\hat{x}\rangle$  stecken in dem Quantenorakel  $U_f$ , unserer Eingabe. Um es auf  $|s\rangle$  anzuwenden benötigen wir ein weiteres Bit  $|y\rangle$ , das wir wie bei den Algorithmen von Deutsch und von Deutsch-Jozsa in den Zustand  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = H|1\rangle$  versetzen. Dann wirkt das Orakel wie folgt (siehe auch Seite 51):

Bedingter Vorzeichenwechsel

$$\begin{aligned} |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) &\xrightarrow{U_f} |x\rangle \frac{1}{\sqrt{2}}(|f(x)\rangle - |1 \oplus f(x)\rangle) \\ &= |x\rangle (-1)^{f(x)} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned}$$

Da nur  $\hat{x}$  den Funktionswert 1 hat, erhält die Amplitude dieses Elements ein negatives Vorzeichen; die anderen Amplituden ändern sich nicht. Abbildung 6.2 illustriert das Ergebnis für den Fall  $N = 4$ . Die Ausgangssuperposition ist  $|s\rangle = \frac{1}{2} \sum_{x=0}^3 |x\rangle$ . Die Amplitude von  $\hat{x}$  wechselt von  $1/2$  zu  $-1/2$ . Damit ist der Superposition das gesuchte Element anzusehen. Eine Messung würde uns jedoch wie in der Ausgangssituation  $|s\rangle$  jedes der Elemente mit der gleichen Wahrscheinlichkeit liefern.

Es gibt eine Möglichkeit, die eben beschriebene Situation auszunutzen. Dazu berechnen wir den Mittelwert der Amplituden und *spiegeln* jede einzelne

Spiegelung am Mittelwert

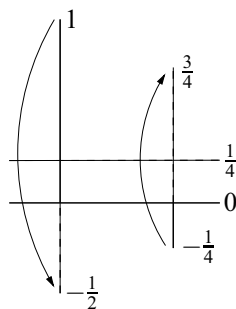


Abbildung 6.3: Spiegeln der Werte 1 und  $-1/4$  am Wert  $1/4$

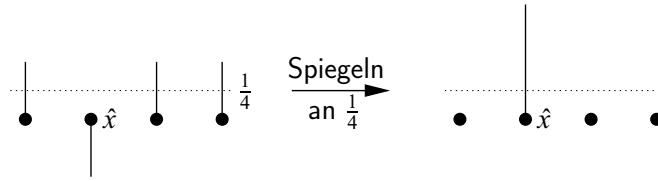


Abbildung 6.4: Anwendung der Spiegelung am Mittelwert im Fall  $N = 4$  auf den Zustand  $1/2, -1/2, 1/2, 1/2$

Amplitude daran, was Abbildung 6.3 illustriert. Spiegeln einer Amplitude  $a$  an dem Wert  $m$  entspricht der Abbildung

$$a \mapsto 2 \cdot m - a. \quad (6.1)$$

Die Abbildung  $a \mapsto -a$  entspricht der Spiegelung an der dem Wert 0 entsprechenden Achse, die Addition von  $2 \cdot m$  ist die Korrektur aufgrund der verschobenen Spiegelachse.

Im Beispiel aus Abbildung 6.2 ist der Mittelwert gerade  $1/4 \cdot (1/2 - 1/2 + 1/2 + 1/2) = 1/4$ . Spiegelung an diesem Mittelwert bildet  $1/2$  auf 0 ab und  $-1/2$  auf 1. Abbildung 6.4 zeigt das Ergebnis für den Fall  $N = 4$ , allerdings nicht ganz maßstabsgerecht. Die Amplitude des gesuchten Elements  $|\hat{x}\rangle$  ist gleich 1, und alle anderen sind verschwunden. Messen der Superposition liefert das erwünschte Resultat.

Bevor wir die Umsetzung der Spiegelung am Mittelwert als unitäre Transformation untersuchen, wenden wir dieses Vorgehen auf den Fall  $N = 8$  an. Die Ausgangssuperposition  $\frac{1}{\sqrt{8}} \sum_{x=0}^7 |x\rangle$  wird durch einen Orakelaufruf  $U_f$  zu

$$\frac{1}{\sqrt{8}} (-|\hat{x}\rangle + \sum_{\substack{x=0 \\ x \neq \hat{x}}}^7 |x\rangle).$$

Der Mittelwert der Amplituden ist gerade  $\frac{1}{8} (\frac{7}{\sqrt{8}} - \frac{1}{\sqrt{8}}) = \frac{6}{8\sqrt{8}}$ . Spiegeln wir die Amplituden daran, bekommt  $|\hat{x}\rangle$  die Amplitude  $(1 + \frac{6}{4}) \frac{1}{\sqrt{8}} = \frac{5}{2\sqrt{8}}$  und alle anderen Elemente  $(-1 + \frac{6}{4}) \frac{1}{\sqrt{8}} = \frac{1}{2\sqrt{8}}$ .

Wir beobachten zweierlei:

- Spiegeln an einem positiven Mittelwert bevorzugt negative Amplituden
- in unserem Fall sind anschließend alle Amplituden wieder positiv

Das ist in Abbildung 6.5 angedeutet. Es bietet sich also an, den Vorgang zu wiederholen, um die Amplitude von  $|\hat{x}\rangle$  nochmals zu erhöhen. Erneut kippen wir zunächst das Vorzeichen von  $|\hat{x}\rangle$  und spiegeln dann am Mittelwert.

**Aufgabe 6.1:** Wir führen im eben beschriebenen Fall  $N = 8$  beide Schritte zweimal aus. Zeigen Sie, dass eine Messung danach das gesuchte Element  $|\hat{x}\rangle$  mit Wahrscheinlichkeit  $121/128$  liefert.



Abbildung 6.5: Spiegelung am Mittelwert der Amplituden

Die Grundidee des Verfahrens lässt sich also wie folgt formulieren:

1. Beginne mit der gleichverteilten Superposition

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle.$$

2. Führe solange aus, bis die Amplitude von  $|\hat{x}\rangle$  groß ist:
  - 2.1. Negiere die Amplitude des gesuchten Elementes  $|\hat{x}\rangle$
  - 2.2. Spiegle jede Amplitude der aktuellen Superposition am Mittelwert aller Amplituden.
3. Miss  $|x\rangle$  und gib das Ergebnis aus.

Grundidee der  
Quantensuche

Schritt 2 des obigen Algorithmus bezeichnet man als *Grover-Iteration*. Daraus entstehen verschiedene Fragen. Zunächst ist zu klären, wie sich Schritt 2 dieses Verfahrens auf einem Quantenregister ausführen lässt: Wir müssen dazu Schritt 2 des obigen Algorithmus mit Hilfe einfacher unitärer Transformationen realisieren. Danach ist zu klären, wie oft genau dieser Schritt ausgeführt werden muss, bevor die Amplitude *groß genug* ist, um bei der Messung mit hoher Wahrscheinlichkeit das gesuchte Element zu erhalten. Wenn der korrekte Abbruchzeitpunkt verpasst und Schritt 2 zu oft ausgeführt wird, verkleinert sich die Amplitude von  $|\hat{x}\rangle$  nämlich wieder. Betrachten wir dazu wieder den Fall  $N = 4$ . Nach einer Anwendung sind wir von der gleichverteilten Superposition in den Zustand  $|\hat{x}\rangle$  übergegangen. Führen wir die Grover-Iteration noch einmal aus, führt 2.1 zu dem Zustand  $-|\hat{x}\rangle$ . Der Mittelwert der Amplituden ist dann  $-1/4$ . Die Amplitude von  $-|\hat{x}\rangle$  wird durch Schritt 2.2 zu  $1/2$ , alle anderen Elemente bekommen die Amplitude  $-1/2$ . Also haben wir das Ergebnis zerstört. Misst man nun, erfährt man nichts über das gesuchte Element. Der Physiker John Preskill vergleicht Grovers Algorithmus mit einem Soufflé, das während des Backens langsam aufgeht. Lässt man es jedoch zu lange im Ofen, fällt es in sich zusammen.



### Die Realisierung der Grover-Iteration

Zunächst überlegen wir, wie die Spiegelung am Mittelwert ausführbar ist. Auf den ersten Blick scheint es eine eher komplexe Operation zu sein. Auf eine Superposition von  $N$  Zuständen wirkt sie wie folgt:

$$\sum_{i=0}^{N-1} \alpha_i |i\rangle \mapsto \sum_{i=0}^{N-1} \left( 2 \cdot \left( \sum_{j=0}^{N-1} \frac{\alpha_j}{N} \right) - \alpha_i \right) |i\rangle.$$

Das folgt aus Gleichung (6.1), denn  $\sum_{j=0}^{N-1} \frac{\alpha_j}{N}$  ist der Mittelwert der Amplituden. Diese Operation entspricht der Matrix

$$D_N = \begin{pmatrix} -1 + \frac{2}{N} & \frac{2}{N} & \cdots & \frac{2}{N} \\ \frac{2}{N} & -1 + \frac{2}{N} & \cdots & \frac{2}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{2}{N} & \frac{2}{N} & \cdots & -1 + \frac{2}{N} \end{pmatrix}.$$

Das sieht man durch Anwendung auf einen Zustandsvektor:

$$\begin{pmatrix} \alpha'_0 \\ \alpha'_1 \\ \vdots \\ \alpha'_{N-1} \end{pmatrix} = D_N \cdot \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix}.$$

Für die  $i$ -te Komponente des Bildes gilt

$$\alpha'_i = -\alpha_i + 2 \cdot \sum_{j=0}^{N-1} \frac{\alpha_j}{N}.$$

Wir haben die Spiegelung am Mittelwert als  $N \times N$ -Matrix dargestellt. Was noch fehlt, ist eine Zerlegung in einfache unitäre Matrizen gemäß dem Lokalitätsprinzip, siehe Seite 31.  $D_N$  ist das folgende Produkt aus einfacheren Matrizen:

$$D_N = -H_n \cdot R_N \cdot H_n \text{ mit } R_N = \begin{pmatrix} -1 & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}.$$

**Aufgabe 6.2:** Beweisen Sie, dass  $D_N = -H_n \cdot R_N \cdot H_n$  gilt.

$H_n$  ist die Hadamard-Transformation auf  $n$  Bits,  $N = 2^n$ .  $R_N$  ändert das Vorzeichen von  $|0 \dots 0\rangle$  (genauer: das Vorzeichen der Amplitude), und es stellt sich die Frage, wie sich diese Operation durch lokale Operationen auf konstant vielen Bits bewerkstelligen lässt. Man kann sich zum Beispiel leicht überlegen, dass  $R_4$  nicht als Tensorprodukt von  $2 \times 2$ -Matrizen geschrieben werden kann.

Wir nutzen, dass  $R_N$  ein Spezialfall des bedingten Vorzeichenwechsels ist, den wir oben betrachtet haben. Wir definieren dazu die Funktion

$$g(x_{n-1}, \dots, x_0) = \begin{cases} 1 & \text{für } x_{n-1} = \dots = x_0 = 0 \\ 0 & \text{sonst.} \end{cases}$$

Bringen wir ein Hilfsbit  $|y\rangle$  in den Zustand  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  und wenden wir das Orakel  $U_g : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus g(x)\rangle$  an, haben wir auf  $|x\rangle$  die Transformation  $R_N$  ausgeführt.

**Aufgabe 6.3:** Zeigen Sie, dass  $U_g$  tatsächlich  $R_N$  ausführt, verwendet man das Orakel wie eben beschrieben.

Das Orakel  $U_g$  lässt sich leicht mit einem klassischen Schaltkreis der Größe  $O(n)$  berechnen. In Abschnitt 3.5 wird beschrieben, wie sich dieser in einen Quantenschaltkreis umwandeln lässt, der eventuell weitere Hilfsbits verwendet, aber ebenfalls Größe  $O(n)$  hat. Für den Fall  $N = 4$  können wir die Konstruktion in Abbildung 6.6 verwenden.

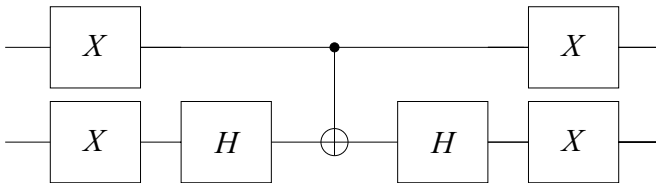


Abbildung 6.6: Schaltkreis, der das Vorzeichen von  $|0\rangle|0\rangle$  negiert.

**Aufgabe 6.4:** Zeigen Sie, dass der Schaltkreis in Abbildung 6.6 die Transformation  $R_4$  berechnet.

Ab jetzt erwähnen wir die Hilfsbits nicht mehr explizit. Für den bedingten Vorzeichenwechsel führen wir die neue Bezeichnung  $V_f$  ein.

Sei  $f$  eine Funktion  $\{0, 1\}^n \rightarrow \{0, 1\}$ . Der bedingte Vorzeichenwechsel ist definiert als

$$V_f : |x\rangle \mapsto (-1)^{f(x)}|x\rangle.$$

$V_f$  lässt sich mit einem Hilfsbit und der Operation  $U_f$  realisieren.

Bedingter Vorzeichenwechsel

Damit ist genau genug beschrieben, wie sich die Grundidee der Quantensuche durch unitäre Transformationen realisieren lässt, und wir können den Algorithmus exakt formulieren. Nur eine Lücke besteht noch: wir wissen nicht, wie oft wir die Grover-Iteration ausführen müssen. In der nun folgenden Formulierung wird diese noch zu bestimmende Anzahl als  $G(N)$  bezeichnet.

$G(N)$

## Grovers Suchalgorithmus

Spezifikation: Gegeben sind eine Menge  $\{0, \dots, N-1\}$  mit  $N = 2^n$  und das Orakel  $U_f$  einer Funktion  $f$ ;  $f$  bildet genau ein Element  $\hat{x}$  auf 1 ab und alle anderen Elemente auf 0.  $\hat{x}$  ist zu bestimmen.

Wir verwenden ein  $n$ -Bit-Register  $|R\rangle$  und linear viele Hilfsbits.

1. Beginne mit der gleichverteilten Superposition:  
 $|R\rangle \leftarrow H_n|0\dots 0\rangle$
2. Wende  $G(N)$ -mal die Grover-Iteration auf  $|R\rangle$  an:  
 $|R\rangle \leftarrow -H_n R_N H_n V_f |R\rangle$
3. Miss  $|R\rangle$  und gib das Ergebnis aus.

Der Schaltkreis in Abbildung 6.7 führt diesen Algorithmus mit einer Iteration aus. Grovers Suchalgorithmus wurde erstmals in [71] vorgestellt. Die geometrische Veranschaulichung im nächsten Abschnitt geht auf Richard Jozsa zurück [84].

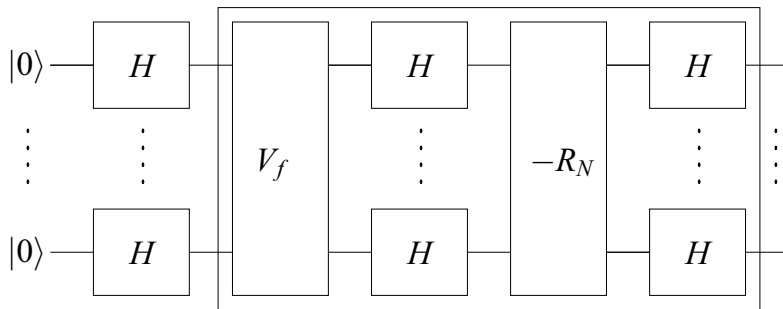


Abbildung 6.7: Schaltkreis, der den Anfangszustand herstellt und einmal die Grover-Iteration (umrahmt) ausführt. Ohne Hilfsbits.

## 6.3 Eine geometrische Veranschaulichung

In diesem Abschnitt bestimmen wir, wie oft wir die Grover-Iteration ausführen müssen, um das gesuchte Element mit hoher Wahrscheinlichkeit zu finden: wir berechnen  $G(N)$ . Die Anzahl  $G(N)$  benötigen wir außerdem, um die Laufzeit der Quantensuche zu bestimmen und diese mit der klassischen Datenbanksuche zu vergleichen. Um diese Anzahl zu bestimmen, interpretieren wir die beiden Schritte der Grover-Iteration geometrisch. Nebenher entwickeln wir dabei ein alternatives Verständnis von Grovers Algorithmus und den Möglichkeiten von Quantenalgorithmen.

Die Spiegelung am Mittelwert der Amplituden ist implizit schon eine geometrische Operation. Allerdings haben wir sie bisher bezüglich der einzelnen

Amplituden betrachtet. Die Operation  $D_N$  kann auch als eine Spiegelung im Raum der Zustandsvektoren angesehen werden. An der Matrix

$$D_N = \begin{pmatrix} -1 + \frac{2}{N} & \frac{2}{N} & \cdots & \frac{2}{N} \\ \frac{2}{N} & -1 + \frac{2}{N} & \cdots & \frac{2}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{2}{N} & \frac{2}{N} & \cdots & -1 + \frac{2}{N} \end{pmatrix}$$

sehen wir:

$$D_N = -I_N + 2 \cdot \begin{pmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix}.$$

Das verdeutlicht nochmals, dass es sich um die Spiegelung am Mittelwert  $m$  handelt. Ein Vektor  $|v\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$  wird auf

$$-|v\rangle + 2 \cdot \sum_{i=0}^{N-1} \frac{\alpha_i}{N} (1, \dots, 1)^T = -|v\rangle + (2m, \dots, 2m)^T$$

abgebildet, wobei

$$m = \frac{1}{N} \sum_{i=0}^{N-1} \alpha_i$$

der Mittelwert der Amplituden von  $|v\rangle$  ist. Wir interpretieren den Operator  $D_N$  auf andere Weise. Zunächst betrachten wir das Skalarprodukt von  $|v\rangle$  mit der gleichverteilten Superposition

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle.$$

Es gilt

$$\langle s | v \rangle = \sum_{i=0}^{N-1} \frac{1}{\sqrt{N}} \alpha_i = \sqrt{N} \cdot m.$$

Wir gelangen also zum Mittelwert der Amplituden, indem wir  $\langle s | v \rangle$  mit  $\frac{1}{\sqrt{N}}$  multiplizieren. Multiplizieren wir statt dessen mit dem Vektor  $|s\rangle$ , ist das Ergebnis gerade

$$|s\rangle \langle s | v \rangle = m \cdot \sum_{x=0}^{N-1} |x\rangle,$$

der Zustandsvektor dessen Komponenten sämtlich den Wert  $m$  haben. Wir betrachten  $\langle s |$  als Abbildung eines Vektors  $|v\rangle$  auf das Skalarprodukt  $\langle s | v \rangle$  und können die Spiegelung am Mittelwert wie folgt ausdrücken:

Es gilt

$$D_N = -I_N + 2|s\rangle \langle s|.$$

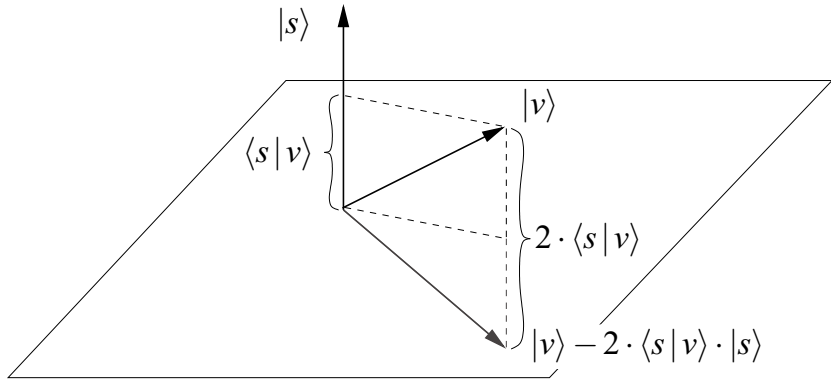


Abbildung 6.8: Spiegelung von  $|v\rangle$  an der Hyperebene, auf der  $|s\rangle$  senkrecht steht

Rechnerisch ist das klar. Was aber bedeutet der Operator  $-I_N + 2|s\rangle\langle s|$  anschaulich? Wir betrachten zunächst die Negation

$$-D_N = I_N - 2|s\rangle\langle s|.$$

Wir erinnern uns daran, dass  $|s\rangle\langle s|x\rangle$  gerade die Projektion von  $|x\rangle$  auf den Unterraum ist, der von  $|s\rangle$  aufgespannt wird.<sup>1</sup> Anschaulich ist nämlich  $\langle s|x\rangle$  die Länge des  $|s\rangle$ -Anteils von  $|x\rangle$ ; das ist sofort klar, wenn  $|s\rangle$  nicht die gleichverteilte Superposition, sondern ein Basisvektor ist. Damit zieht der Operator  $I_N - 2|s\rangle\langle s|$  – wie in Abbildung 6.8 – von einem Vektor  $|x\rangle$  zweimal die Projektion auf den von  $|s\rangle$  aufgespannten Raum ab! Die  $|s\rangle$ -Komponente wird negiert, alles andere bleibt beim Alten, oder:  $|x\rangle$  wird an der Hyperebene gespiegelt, auf der  $|s\rangle$  senkrecht steht.

**Aufgabe 6.5:** Untersuchen Sie  $I_N - 2|s\rangle\langle s|$  im zweidimensionalen Fall. Einmal für die gleichverteilte Superposition und auch für

$$|s\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Interessant ist nun, dass  $V_f$  im Falle des auf 1 abgebildeten Elements  $|\hat{x}\rangle$  ebenfalls als Spiegelung im Raum der Zustandsvektoren aufgefasst werden kann. Gerade die Amplitude von  $|\hat{x}\rangle$  wird negiert. Also gilt mit unseren Voraussetzungen

$$V_f = I_N - 2|\hat{x}\rangle\langle \hat{x}|.$$

$V_f$  spiegelt an der zu  $|\hat{x}\rangle$  orthogonalen Hyperebene: die Komponente von  $|\hat{x}\rangle$  wird negiert, der Rest bleibt erhalten. Wir bezeichnen mit  $S_v$  die Spiegelung

<sup>1</sup>Siehe dazu Abschnitt A.2.4 im Anhang.

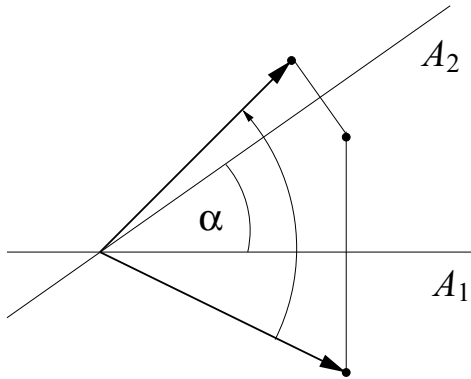


Abbildung 6.9: Zwei Spiegelungen in der Ebene ergeben eine Drehung

an der Hyperebene  $\text{Span}\{|v\rangle\}^\perp$ , die aus allen Vektoren besteht, die orthogonal zu  $|v\rangle$  sind.

Wir können die Negation der Grover-Iteration als Folge zweier Spiegelungen  $S_s$  und  $S_{\hat{x}}$  auffassen.

### Zwei Spiegelungen sind eine Drehung

$-D_N V_f$ , die Negation der Grover-Iteration, ist eine Spiegelung an der Hyperebene  $\text{Span}\{|\hat{x}\rangle\}^\perp$  gefolgt von einer Spiegelung an  $\text{Span}\{|s\rangle\}^\perp$ . Der Algorithmus beginnt mit der Superposition  $|s\rangle$ . Bei der ersten Spiegelung wird die  $|\hat{x}\rangle$ -Komponente negiert. Alle anderen Amplituden bleiben unverändert: das Ergebnis ist ein Vektor, der sich in der Ebene befindet, die von  $|s\rangle$  und  $|\hat{x}\rangle$  aufgespannt wird. Wenn wir danach an  $\text{Span}\{|s\rangle\}^\perp$  spiegeln und die Spiegelungen wiederholen, verlassen wir diese Ebene nicht!

Wenden wir die Spiegelungen  $S_s S_{\hat{x}}$  wiederholt auf  $|s\rangle$  an, befindet sich jedes Zwischenergebnis in der Ebene, die von  $|s\rangle$  und  $|\hat{x}\rangle$  aufgespannt wird, also in  $\text{Span}\{|s\rangle, |\hat{x}\rangle\}$ .

Damit findet die gesamte Berechnung in einem zweidimensionalen Unterraum des Raumes der Zustandsvektoren statt. Für uns hat das die erfreuliche Konsequenz, dass wir Ergebnisse der ebenen Geometrie anwenden können.

Wie wirken zwei Spiegelungen in der Ebene? In Abbildung 6.9 wird ein Vektor an zwei Achsen  $A_1$  und  $A_2$  gespiegelt.  $\alpha$  ist der Winkel zwischen den beiden Achsen; die beiden Spiegelungen entsprechen einer Drehung um den Winkel  $2\alpha$ . Das wenden wir auf die Spiegelungen  $S_s$  und  $S_{\hat{x}}$  und ihre Wirkung in der Ebene  $\text{Span}\{|s\rangle, |\hat{x}\rangle\}$  an.

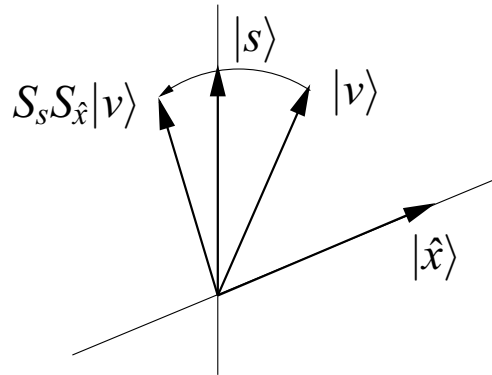


Abbildung 6.10: Die Spiegelungen  $S_s S_{\hat{x}}$  wirken zusammen als Drehung in der von  $|s\rangle$  und  $|\hat{x}\rangle$  aufgespannten Ebene

Der Winkel zwischen den Spiegelachsen entlang  $|s^\perp\rangle$  und  $|\hat{x}^\perp\rangle$  ist gleich dem Winkel  $\alpha$  zwischen den Vektoren  $|s\rangle$  und  $|\hat{x}\rangle$ . Für diesen gilt<sup>2</sup>

$$\cos \alpha = \langle s | \hat{x} \rangle = \frac{1}{\sqrt{N}}.$$

Das bedeutet, dass der Winkel von dem gesuchten Element unabhängig ist. Grovers Algorithmus verfährt demnach dem Geiste nach wie folgt (siehe Abbildung 6.10):

Wir beginnen mit der gleichverteilten Superposition über alle Datenbank-elemente  $|s\rangle$  und drehen diesen Vektor in Richtung  $|\hat{x}\rangle$ . Sind wir dem gesuchten Element nah genug, messen wir.

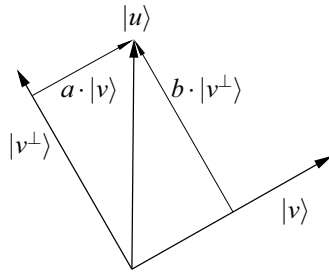
Aber einen Unterschied gibt es! Wir haben bisher  $S_s S_{\hat{x}}$  betrachtet. Grovers Algorithmus verwendet jedoch  $-S_s S_{\hat{x}}$ . Das hat folgenden Grund: Für großes  $N$  wird  $\frac{1}{\sqrt{N}}$  sehr klein, folglich ist dann  $\alpha$  nahe  $\pi$  beziehungsweise  $180^\circ$ . Eine Drehung um  $2\alpha$  ist sehr unbeholfen.

Um  $-S_s$  zu verstehen, hilft uns erneut, dass unsere Spiegelungen in einem zweidimensionalen Unterraum wirken. Und dort gilt folgendes: Sei  $v$  ein beliebiger Vektor und  $v^\perp$  ein Einheitsvektor, der auf  $v$  senkrecht steht (es gibt genau zwei davon). Dann kann jeder Vektor  $u$  in Komponenten bezüglich  $v$  und  $v^\perp$  zerlegt werden. Die Darstellung  $u = a \cdot v + b \cdot v^\perp$ , wie in Abbildung 6.11 illustriert, ist eindeutig.

An dieser Darstellung lässt sich die Wirkung von  $S_v$  gut erkennen:

$$S_v(a \cdot v + b \cdot v^\perp) = -a \cdot v + b \cdot v^\perp.$$

<sup>2</sup>Siehe Abschnitt A.2.3 im Anhang.

Abbildung 6.11: Zerlegung des Vektors  $u$ 

Das heißt

$$-S_v(a \cdot v + b \cdot v^\perp) = a \cdot v - b \cdot v^\perp$$

oder

$$-S_v = S_{v^\perp}.$$

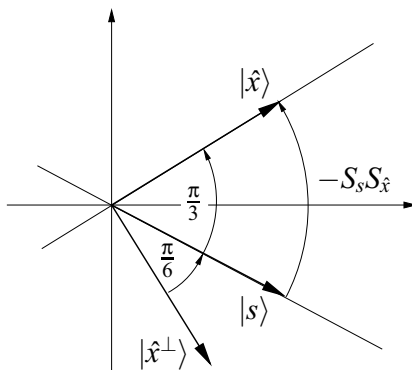
$-S_s S_{\hat{x}}$  entspricht also den Spiegelungen  $S_s S_{\hat{x}^\perp}$ ; d.h. die Ebene  $\text{Span}\{|s\rangle, |\hat{x}\rangle\}$  wird ebenso wenig verlassen wie bei Anwendung von  $S_s S_{\hat{x}}$ . Für den neuen Drehwinkel  $\beta$  folgt

$$\cos \beta = \langle s | \hat{x}^\perp \rangle$$

oder

$$\sin \beta = \langle s | \hat{x} \rangle = \frac{1}{\sqrt{N}}.$$

**Beispiel 6.1:** Im Falle einer Datenbank mit  $N = 4$  Elementen gilt  $\sin \beta = 1/2$ . Der Winkel zwischen  $|s\rangle$  und  $|\hat{x}^\perp\rangle$  beträgt damit  $\beta = \frac{\pi}{6}$  oder  $30^\circ$  wie in Abbildung 6.12.

Abbildung 6.12: Die Grover-Iteration in der Ebene im Fall  $N = 4$ .

Eine Grover-Iteration dreht den Ausgangsvektor  $|s\rangle$  um den Winkel  $\frac{\pi}{3}$ . Danach ist der Zielvektor auch schon erreicht: der Winkel zwischen  $|\hat{x}^\perp\rangle$  und dem Bild von  $|s\rangle$  beträgt  $\frac{\pi}{6} + \frac{\pi}{3} = \frac{\pi}{2}$ .



Das Ergebnis einer Grover-Iteration steht also auf  $|\hat{x}^\perp\rangle$  senkrecht. Messen ergibt das gesuchte Element, und für die Anzahl der Iterationen folgt  $G(4) = 1$ .  $\diamond$

*Bemerkung:* Bei diesen Betrachtungen könnte Verwirrung stiften, dass es zwei Vektoren gibt, die in der Ebene  $\text{Span}\{|s\rangle, |\hat{x}\rangle\}$  auf  $|\hat{x}\rangle$  senkrecht stehen. Das ändert allerdings zum einen nichts an der Spiegelachse; außerdem kann man sich überlegen, dass der Ausgangsvektor tatsächlich auf  $|\hat{x}\rangle$  und nicht auf  $-|\hat{x}\rangle$  gedreht wird. Außerdem liefern Messungen an  $|\hat{x}\rangle$  und  $-|\hat{x}\rangle$  identische Ergebnisse.  $\diamond$

#### Allgemeine Analyse

Wie in Beispiel 6.1 lässt sich für jedes  $N$  die Zahl  $G(N)$  genau bestimmen. Wirksam wird die Überlegenheit eines Quantencomputers gegenüber klassischen Rechnern bei Datenbanken mit vielen Elementen. Mit  $\arcsin$  bezeichnen wir die Umkehrfunktion des Sinus, die mathematisch lax auch als  $\sin^{-1}$  bezeichnet wird. Für großes  $N$  ist  $\arcsin \frac{1}{\sqrt{N}}$  fast gleich  $\frac{1}{\sqrt{N}}$ , wie Abbildung 6.13 verdeutlicht.

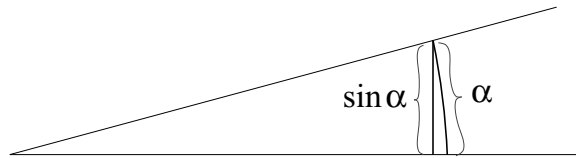


Abbildung 6.13: Je kleiner  $\alpha$  ist, desto näher ist die Länge des Bogens bei  $\sin \alpha$

Im allgemeinen Fall beträgt der Winkel zwischen dem Anfangszustand  $|s\rangle$  und  $|\hat{x}^\perp\rangle$  also etwa  $\frac{1}{\sqrt{N}}$ . Unser Ziel ist es,  $|s\rangle$  so weit zu drehen, dass das Ergebnis auf  $|\hat{x}^\perp\rangle$  senkrecht steht. Nach  $T$  Grover-Iterationen hat der Winkel den Wert

$$(2T + 1) \frac{1}{\sqrt{N}},$$

da jeder Iterationsschritt eine Drehung um etwa den Winkel  $2/\sqrt{N}$  ist. Damit erhalten wir nach

$$T = \frac{\pi}{4} \sqrt{N}$$

Schritten den Winkel

$$\frac{\pi}{2} + \frac{1}{\sqrt{N}}.$$

Das heißt, der dabei erhaltene Vektor steht auf  $|\hat{x}^\perp\rangle$  fast senkrecht. Beim Messen erhalten wir  $|\hat{x}\rangle$  mit hoher Wahrscheinlichkeit, da wir uns in der von  $|s\rangle$  und  $|\hat{x}\rangle$  aufgespannten Ebene befinden.

Für großes  $N$  gilt

$$G(N) \approx \frac{\pi}{4} \sqrt{N} = O(\sqrt{N}).$$

### Die Größe des Schaltkreises

Wir haben bewiesen, dass Grovers Algorithmus mit  $O(\sqrt{N})$  Orakelaufrufen auskommt. Das entspricht einer quadratischen Beschleunigung gegenüber der klassischen Datenbanksuche im mittleren Fall. Wie groß ist der Schaltkreis für Grovers Algorithmus? Dazu betrachten wir Abbildung 6.7. Das Orakel  $V_f$  ist uns geliefert worden, und wir müssen uns keine Gedanken darüber machen, wie aufwändig die darin ausgeführte Berechnung ist: es zählt als ein Gatter. Die Operation  $-R_N$  können wir mit  $n = \log N$  lokalen Gattern realisieren; das haben wir auf Seite 143 gesehen. Also genügen für einen Schritt der Grover-Iteration  $O(\log N)$  Gatter und die Größe des ganzen Schaltkreises ist

$$O(\log N \cdot \sqrt{N}).$$

## 6.4 Varianten der Quantensuche

Lov Grovers ursprüngliche Arbeit hat eine große Zahl weiterer Untersuchungen nach sich gezogen. Diese zeigen, wie Grovers Grundidee an verwandte Problemstellungen angepasst werden kann. Dieser Abschnitt präsentiert Verfahren für die Suche in einer Datenbank, in der mehrere Elemente dem Suchkriterium entsprechen und die Suche nach dem kleinsten Element eines Feldes. Abschließend zitieren wir ein Ergebnis, das sich mit dem Zählen der *markierten*, also dem Suchkriterium entsprechenden Elemente, in einer Datenbank befasst.

### 6.4.1 Suche nach einer von mehreren Lösungen

Hilft Grovers Idee auch, wenn mehrere Elemente der Datenbank das Ergebnis unserer Suche sein können? Angenommen, wir haben die Telefonnummer einer Wohngemeinschaft bekommen, und diese Nummer ist unter den Namen aller vier Mitglieder verzeichnet. Im klassischen Fall wird die Aufgabe dadurch einfacher: um *einen* der vier Einträge zu finden, ist die erwartete Anzahl der Datenbankaufrufe ein Viertel der Versuche, die eine Suche nach nur einem markierten Eintrag erwarten lässt.

Bei der Quantensuche ist es genauso. Seien  $\hat{x}_1, \dots, \hat{x}_k$  die gesuchten Elemente. In der Grover-Iteration wird die Spiegelung  $S_{\hat{x}}$  durch  $S_{\tilde{x}}$  ersetzt mit

$$|\tilde{x}\rangle = \frac{1}{\sqrt{k}} \sum_{i=1}^k |\hat{x}_i\rangle.$$

Wir spiegeln also an der Hyperebene, auf der die gleichgewichtete Superposition aller gesuchten Elemente senkrecht steht.

Aufgabenstellung

Analyse

**Aufgabe 6.6:** Überlegen Sie sich, wie man die Spiegelung an der zu  $\tilde{x}$  orthogonalen Hyperebene ausführen kann.

Nun kann man mit dem Ansatz des letzten Abschnitts die Anzahl der Schritte bestimmen, mit der wir  $|\tilde{x}\rangle$  erreichen. Zunächst gilt

$$\langle s | \tilde{x} \rangle = \sqrt{k} \cdot \frac{1}{\sqrt{N}} = \sin \beta$$

für den Winkel  $\beta$  zwischen den Spiegelachsen. Es folgt

$$\beta \approx \sqrt{\frac{k}{N}}$$

wenn  $k/N$  klein ist. Die erste Grover-Iteration dreht den Ausgangsvektor  $|s\rangle$  um den Winkel  $2\beta$  in Richtung  $|\tilde{x}\rangle$ . Nach

$$T \approx \frac{\pi}{4} \sqrt{\frac{N}{k}}$$

Wiederholungen sind wir nah am Ziel. Messen ergibt *eines* der gesuchten Elemente. Interessant ist der Fall, in dem genau ein Viertel der Elemente gesucht sind:

**Aufgabe 6.7:** Zeigen Sie, dass wir im Fall  $k = N/4$  nach einer Grover-Iteration eine Lösung finden.

### Algorithmus zur Suche nach mehreren Elementen

Der Algorithmus

Spezifikation: Gegeben sind eine Menge  $\{0, \dots, N-1\}$  mit  $N = 2^n$  und das Orakel  $U_f$  einer Funktion  $f$ ;  $f$  bildet genau  $k$  Elemente  $\hat{x}_1, \dots, \hat{x}_k$  auf 1 ab und alle anderen Elemente auf 0. Wir setzen voraus, dass  $k \leq 3/4N$  ist. Eines der Elemente  $\hat{x}_1, \dots, \hat{x}_k$  ist zu bestimmen.

1. Beginne mit der gleichverteilten Superposition:  
 $|R\rangle \leftarrow H_n |0 \dots 0\rangle$
2. Wende  $G(N, k)$ -mal die Grover-Iteration auf  $|R\rangle$  an:  
 $|R\rangle \leftarrow -H_n R_N H_n V_f |R\rangle$
3. Miss  $|R\rangle$  und gib das Ergebnis aus.

Für die Zahl der Iterationen folgt mit der Betrachtung zu Beginn dieses Unterabschnitts:

$G(N, k)$  ist diejenige ganze Zahl, die am nächsten an

$$\frac{\pi}{4} \arcsin \sqrt{N/k} - \frac{1}{2}$$

liegt. Falls  $N$  deutlich größer als  $k$  ist, gilt

$$G(N, k) \approx \frac{\pi}{4} \sqrt{\frac{N}{k}}.$$

Was tun wir, wenn  $k > 3/4N$  ist? Dann wählen wir zufällig ein Element  $x$  aus und berechnen  $f(x)$ . Die Wahrscheinlichkeit für  $f(x) = 1$  ist größer  $3/4$ .

**Aufgabe 6.8:** Überlegen Sie sich, wie wir alle gesuchten Elemente aufzählen können.

### 6.4.2 Suche bei unbekannt vielen Lösungen

Wir haben beim letzten Algorithmus vorausgesetzt, dass die Anzahl der Lösungen bekannt ist. Das ist aber in der Praxis nicht immer gegeben: dann ist es nicht möglich, die korrekte Anzahl der Iterationen zu bestimmen. Abhilfe schafft ein Verfahren, das nach seinen Schöpfern Michel Boyer, Gilles Brassard, Peter Høyer und Alain Tapp als *G-BBHT-Suche* bezeichnet wird [26].

Die Anzahl der Iterationen wird geraten. Wir testen nach Abschluss der Berechnung, ob der Algorithmus das gesuchte Element geliefert hat, indem wir in der Datenbank nachsehen. Falls das Ergebnis nicht korrekt war, versuchen wir es mit einer anderen Zahl. Die erwartete Laufzeit ist dabei nicht schlechter, als bei bekannter Anzahl  $k$ . Es ist demnach nicht sehr wahrscheinlich, die Zahl der Iterationen schlecht zu raten.

#### Die G-BBHT-Suche

Spezifikation: Gegeben sind eine Menge  $\{0, \dots, N-1\}$  mit  $N = 2^n$  und das Orakel  $U_f$  einer Funktion  $f$ ;  $f$  bildet genau  $k$  Elemente  $\hat{x}_1, \dots, \hat{x}_k$  auf 1 ab und alle anderen Elemente auf 0. Die Zahl  $k$  ist uns nicht bekannt.

Eines der Elemente  $\hat{x}_1, \dots, \hat{x}_k$  ist zu bestimmen.

1. Wähle zufällig ein  $x \in \{0, 1\}^n$ .  
Falls  $f(x) = 1$ , gib  $x$  aus und stoppe.
2. Wähle  $r$ , die Zahl der Iterationen zufällig aus  $1, \dots, \sqrt{N}$ .
3. Beginne mit der gleichverteilten Superposition:  
 $|R\rangle \leftarrow H_n|0 \dots 0\rangle$
4. Wende  $r$ -mal die Grover-Iteration auf  $|R\rangle$  an:  
 $|R\rangle \leftarrow -H_n R_N H_n V_f |R\rangle$
5. Miss  $|R\rangle$  und werte  $f$  für das Ergebnis  $x$  aus:  
Ist  $f(x) = 1$  gib  $x$  aus und stoppe.  
Anderenfalls, beginne erneut mit Schritt 1.

Das Auswerten eines zufällig ausgewählten Elementes zu Anfang dient folgendem Zweck: falls  $k \geq 3/4N$  ist, finden wir in diesem Schritt mit hoher Wahrscheinlichkeit eines der gesuchten Elemente.

Die Analyse in [26] zeigt, dass nach dem ersten Durchlauf des Algorithmus mit Wahrscheinlichkeit größer als  $1/4$  eine Lösung gefunden wird. Also ist die erwartete Anzahl der Ausführungen der Schritte 1-5 gerade 4, wie sich aus dem Erwartungswert der geometrischen Verteilung ergibt. Die erwartete Laufzeit ist also gerade  $O(\sqrt{N})$ , auch wenn wir die Zahl der Iterationen raten.

Geht man etwas cleverer vor, kommt man mit  $O(N/k)$  Orakelaufrufen aus. Man beginnt mit einer Iteration und steigert die Zahl der Wiederholungen von Fehlversuch zu Fehlversuch langsam. Konkret wählt man diese zufällig aus einer Menge  $\{1, \dots, m\}$ . Zu Beginn ist  $m = 1$ ,  $m$  steigert sich vor jeder neuen Wahl um den Faktor  $6/5$ . Wir fassen dieses Ergebnis zusammen, ohne auf Details einzugehen:

In einer Datenbank sind  $k$  Elemente markiert, die Zahl  $k$  ist jedoch unbekannt. Dann findet die G-BBHT-Suche eines der Elemente mit der erwarteten Anzahl von Orakelaufrufen  $O(N/k)$ .

### 6.4.3 Die Suche nach dem Minimum

In einem Feld soll das kleinste Element bestimmt werden. Dies ist mit einem Quantenalgorithmus möglich, der mit unseren Kenntnissen über Grovers Algorithmus leicht zu verstehen ist. Während die bisherige Quantensuche lokal ist – ein bestimmter Eintrag ist gesucht, unabhängig von den Eigenschaften anderer Elemente – ist für die Suche nach dem Minimum eine globale Eigenschaft ausschlaggebend; die Eigenschaft, kleinstes Element zu sein, hängt von dem gesamten Feld ab. Die geschilderte Aufgabe lässt sich mit einem von Christoph Dürr und Peter Høyer (1996) stammenden Algorithmus lösen, der natürlich leicht für die Suche nach dem Maximum abgewandelt werden kann.

### Algorithmus zur Suche nach dem Minimum

- Eingabe ist ein Orakel  $U$ , das auf ein Feld  $T[0], \dots, T[N-1], N = 2^n$ , zugreift.  $U$  kennzeichnet alle Indizes  $i$  aus  $0, \dots, N-1$  als markiert, für die  $T[i] < T[j]$  für einen Schrankenindex  $j$  gilt.
- Wir rechnen mit zwei Quantenregistern  $|x\rangle|y\rangle$  der Länge  $n$  und einem Hilfsbit  $|h\rangle$ . Mit dem Orakel

$$U : |x\rangle|y\rangle|h\rangle \mapsto |x\rangle|y\rangle|h \oplus f(x,y)\rangle,$$

$$f(x,y) = \begin{cases} 1 & \text{falls } T[x] < T[y] \\ 0 & \text{sonst.} \end{cases}$$

wird die im letzten Abschnitt beschriebene G-BBHT-Suche ausgeführt.  $|y\rangle$  speichert dabei das Zwischenergebnis, das kleinste bisher gefundene Element.

- Das Orakel wird  $G_{DH}(N)$ -mal aufgerufen; diesen Wert geben wir im Anschluss an.
- Ausgabe ist das kleinste Element des Feldes.

1. Wähle einen zufälligen Feldindex  $j$  als anfänglichen *Schrankenindex*. Setze  $|y\rangle \leftarrow |j\rangle$ .
2. Führe die folgenden Schritte aus, bis  $G_{DH}(N)$ -mal auf  $T$  zugegriffen wurde:
  - 2.1.  $|x\rangle \leftarrow |0 \dots 0\rangle$
  - 2.2. Führe die G-BBHT-Suche auf dem Register  $|x\rangle$  mit dem Quantenorakel  $U$  aus.
  - 2.3. Miss Register  $|x\rangle$ .  
Gilt für das Ergebnis  $i$ , dass  $T[i] < T[y]$ , setze  $|y\rangle \leftarrow |i\rangle$ .
3. Miss Register  $|y\rangle$  und gib das Ergebnis aus.

In Schritt 2.2 wird von allen Feldelementen, die kleiner als die Schranke  $T[y]$  sind, eines zufällig ausgewählt (dass Grovers Algorithmus und damit die G-BBHT-Suche manchmal ein falsches Ergebnis liefert – wenn auch mit geringer Wahrscheinlichkeit – wird durch den Test in Schritt 2.3 abgefangen).

Wie das Orakel umsetzen?

Ist die Definition des in Schritt 2.2 verwendeten Orakels auch für die Praxis sinnvoll? Um diese Frage zu beantworten gehen wir davon aus, dass uns das Feld  $T$  selbst als Orakel

$$U_T : |i\rangle|0\rangle \mapsto |i\rangle|T[i]\rangle$$

gegeben ist. Um daraus das im Algorithmus verwendete Orakel  $U$  zu konstruieren, verwenden wir ein drittes  $n$ -Bit-Register  $|z\rangle$ , dem Feldinhalte  $T[i]$  zugewiesen werden. Nun können wir aus klassischen (reversiblen) Gattern einen Vergleich zwischen dem Schranken-Element und dem Hilfsregister vornehmen. Ein weiteres Hilfsbit speichert das Ergebnis des Orakelaufrufs. Wir verwenden also insgesamt  $3n + 1$  Quantenbits

$$|x\rangle|y\rangle|0\rangle|0\rangle.$$

In einem ersten Schritt wendet  $U$  das Feldorakel  $U_T$  mit Ergebnis

$$|x\rangle|y\rangle|T[x]\rangle|0\rangle$$

an. Danach werden  $T[x]$  und  $T[y]$  verglichen; ist  $T[x]$  kleiner, wird das letzte Bit negiert. Da so ein Vergleich klassisch möglich ist, können wir ihn auch in einen Quantenschaltkreis integrieren.

Ist eine Funktion  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  durch einen Schaltkreis gegeben, können wir mit dem beschriebenen Verfahren die Eingabe mit dem minimalen Funktionswert finden.

Laufzeit

Analysiert man den Algorithmus zur Suche nach dem Minimum, ergibt sich, dass mit  $G_{DH}(N) = O(\sqrt{N})$  Orakelzugriffen in mehr als der Hälfte der Fälle das Minimum gefunden wird.

Aus Platzgründen müssen wir uns mit einigen, das Ergebnis veranschaulichenden Hinweisen begnügen und für Details auf die Ursprungsarbeit [50] verweisen. Angenommen, wir führen den Algorithmus unendlich lange aus. Wie groß ist die Wahrscheinlichkeit, dass irgendein Element  $T[i]$  jemals zur Schranke  $y$  wird? Es lässt sich beweisen: Wenn es  $t - 1$  kleinere Elemente als  $T[i]$  gibt, wird  $T[i]$  mit Wahrscheinlichkeit  $1/t$  jemals zur Schranke.

Wenn nun für unsere Schranke  $y$  gilt, dass  $t - 1$  Elemente kleiner sind, können wir ein solches mit der G-BBHT-Suche in Zeit

$$O\left(\sqrt{\frac{N}{t-1}}\right)$$

finden. Damit ergibt sich für die erwartete Zahl an Feldzugriffen

$$\sum_{t=2}^N \frac{1}{t} O\left(\sqrt{\frac{N}{t-1}}\right) = O(\sqrt{N}) \cdot \sum_{t=2}^N \frac{1}{t\sqrt{t}} = O(\sqrt{N}).$$

Anwendungen

Die Quantensuche nach dem Minimum eines Feldes ist für Graphalgorithmen verwendbar. In [49] wird unter anderem das Problem *Single Source Shortest Path* untersucht: in gewichteten Graphen sollen kürzeste Wege bestimmt werden. Im klassischen Fall muss  $\Theta(n^2)$  oft auf die Kantengewichte zugegriffen werden, während im Quantenfall  $O(n^{3/2} \log n)$  Aufrufe genügen. Es wird außerdem gezeigt, dass Quantenalgorithmen für dieses Problem mindestens  $\Omega(n^{3/2})$  Zugriffe benötigen.

### 6.4.4 Zählen

Brassard, Høyer und Tapp haben in einer Arbeit von 1998 einen Algorithmus vorgestellt, der die Anzahl der markierten Elemente in einer Datenbank ermittelt. Ein klassischer Algorithmus braucht für diese Aufgabe  $\Theta(N)$  Anfragen, wenn  $N$  die Größe der Datenbank ist.

In einer Datenbank der Größe  $N$  lässt sich die Anzahl der markierten Elemente mit  $O(\sqrt{N})$  Anfragen ermitteln.

Quanten-Zählen

Auf den ersten Blick erscheint dieses Ergebnis noch überraschender als das über die Quantensuche. Wir zählen mit weniger Schritten, als es zu zählende Elemente gibt. Aber wir wissen bereits, dass ein Quantenorakel auf alle Elemente zugleich zugreifen kann. Somit könnten wir ebenso gut fragen: Geht das nicht schneller? Diese Frage stellen wir für die Quantensuche in Abschnitt 6.6.

Wir beschränken uns darauf, die Grundidee des Quantenzählens anzudeuten. Der Quantenzählalgorithmus verbindet Ideen von Grovers Algorithmus und Shors Algorithmus, Kapitel 8. Wir haben John Preskill zitiert, der Grovers Algorithmus mit einem Soufflé vergleicht. Während des Backens geht dies langsam auf, wird die Backzeit überschritten, fällt es in sich zusammen. Zu viele Grover-Iterationen entfernen den aktuellen Zustand von dem Ziel der Rechnung. Irgendwann sind wir so schlau wie zu Beginn. Im Unterschied zu dem Soufflé verbessert sich das Ergebnis in der Quantensuche jedoch, wenn wir an dieser Stelle zu rechnen fortfahren. Die Amplituden entwickeln sich *periodisch*. Ebenso ist die Amplitude der Menge aller gesuchten Elemente eine periodische Funktion in der Anzahl der Iterationen.

Wenn nun  $k$  Einträge der Datenbank gesucht beziehungsweise markiert sind, hängt die Periode von der Zahl  $k$  ab. Der Kern von Shors Algorithmus ist das schnelle Auffinden der Periode einer ganzzahligen Funktion.

## 6.5 Anwendungen von Grovers Algorithmus

Zu Beginn dieses Kapitels haben wir verschiedene Anwendungen von Grovers Suchalgorithmus betrachtet. Als anschaulicher Oberbegriff diente uns die *Datenbanksuche*. Aber wie können wir die Quantensuche konkret nutzen? Durchforsten bald Quantensuchmaschinen das Internet? Der Knackpunkt ist, dass wir bisher das Datenbankorakel  $U_f$  vorausgesetzt haben, ohne uns Gedanken über dessen Realisierung zu machen.



Quanten- datenbanken	Eine Quantendatenbank ist nach unserem bisherigen Verständnis ein Quantenschaltkreis, der das zugehörige Datenbankorakel realisiert. Denn Grovers Algorithmus sucht letztlich in der Ergebnismenge eines ihm übergebenen Algorithmus. Diesen Umstand sollten wir im Auge behalten, wenn wir praktische Anwendungen von Grovers Idee untersuchen.
Klassische Datenbanken	Es ist prinzipiell möglich, mit dem Quantenalgorithmus auch klassische Datenbanken zu durchsuchen, vorausgesetzt: solch eine Datenbank ist über ein Quantenadressierungssystem ansprechbar, das auf jeden Eintrag direkt zugreifen kann. Schon die Beschreibung macht deutlich, dass so ein Adressierungssystem aufwändig zu bauen ist. Ob dieser Ansatz eine reale Option ist, hängt von der zukünftigen Quantenhardware und deren Preis ab. Eine detailliertere Darstellung findet sich in [114].
Optimierungs- probleme	<p>Bereits zu Beginn des Kapitels wurden Optimierungsprobleme erwähnt, auch die Entscheidungsvariante des <i>Problems des Handlungsreisenden</i> (TSP). Wir sollen entscheiden, ob es für das uns übergebene Ensemble von <math>n</math> Städten eine Rundreise kürzer als eine Konstante <math>k</math> gibt. Ein Orakel, das Grovers Algorithmus anwendbar macht, erwartet eine aus zwei Teilen bestehende Eingabe:</p> <ul style="list-style-type: none"> <li>• die Abstände zwischen den Städten und die Schranke <math>k</math>: die Problemspezifikation und</li> <li>• eine Rundreise, also die Reihenfolge in der die Städte besucht werden.</li> </ul> <p>Die Ausgabe des Orakels ist 1, wenn die Rundreise kürzer als <math>k</math> ist, und 0 sonst. Es ist leicht, einen effizienten klassischen Algorithmus anzugeben, der die beschriebene Aufgabe erledigt. Mit dem Ergebnis von Abschnitt 3.5 können wir daraus einen Quantenschaltkreis konstruieren, der als Orakel effizient einsetzbar ist. Für eine Eingabe des Problems des Handlungsreisenden findet dann die G-BBHT-Suche eine Rundreise kürzer <math>k</math> mit Laufzeit <math>O(\sqrt{N})</math>, wenn <math>N</math> die Zahl der Rundreisen ist. Die Suche nach dem Minimum aus Abschnitt 6.4.3 findet mit einem entsprechenden Orakel sogar die <i>kürzeste</i> Rundreise.</p>
NP-Probleme	Die eben betrachtete Entscheidungsvariante des Problems des Handlungsreisenden liegt in der Komplexitätsklasse NP. Grovers Algorithmus ist für viele ähnliche Optimierungsprobleme anwendbar und grundsätzlich für jedes Problem in der Klasse NP. Ein Problem ist gerade dann in NP, wenn eine Lösung mit Hilfe eines Zertifikats in polynomialer Zeit verifiziert werden kann (siehe Abschnitt 4.3). Solange man keine bessere Idee hat, könnte man die Menge aller möglichen Zertifikate durchlaufen und zu verifizieren versuchen. Das wäre die Brute-Force-Methode. Diese Zertifikatsmenge kann von exponentieller Größe sein. Man nimmt an, dass es Probleme in NP gibt, die nicht effizient berechenbar sind, die sogenannten NP-vollständigen Probleme (Abschnitt 4.3). Mit einem ähnlichen Ansatz wie für das Problem des Handlungsreisenden können Quantensuchalgorithmen das Auffinden eines Zertifikats immerhin quadratisch beschleunigen. Allerdings ergibt sich daraus im Allgemeinen noch kein effizienter Algorithmus. Es wird vielmehr angenommen, dass nicht alle Probleme in NP von Quantenalgorithmus effizient gelöst werden können. Dem Grund dafür sind die beiden nächsten Abschnitte gewidmet.

Ähnlich kann Grovers Methode verwendet werden, um für einen gegebenen Algorithmus eine Eingabe mit bestimmten Eigenschaften zu bestimmen. Dieser Algorithmus bildet dazu die Grundlage für das Quantenorakel. Damit eignen sich Quanten-Suchalgorithmen prinzipiell dazu, Teilaufgaben unterschiedlicher algorithmischer Fragestellungen zu beschleunigen.

## 6.6 Grovers Algorithmus ist von optimaler Größenordnung

Mit Quantenrechnern lässt sich ein einzelnes Datenbankelement schneller finden als mit klassischen Rechnern. Die Beschleunigung ist quadratisch, und zwar auch für die Varianten des Ausgangsproblems, die wir im letzten Abschnitt kennengelernt haben. Außerdem ist die Idee sehr einfach. Man könnte vermuten, mit raffinierteren Ideen ließe sich eine noch größere Beschleunigung erreichen: der Traum vieler Informatiker wäre ein Suchalgorithmus, der aus  $N$  Datenbankelementen das gesuchte in Zeit  $O(\log N)$  finden könnte. Die Auswirkungen, die solch ein Algorithmus hätte, werden wir im nächsten Abschnitt studieren.

Allerdings ist das nicht möglich! Grovers Algorithmus ist optimal; zumindest was die Größenordnung der Laufzeit gemessen in der Anzahl der Orakelaufrufe betrifft.

Um uns die Analyse zu vereinfachen, erweitern wir das Modell der Suche in einer Datenbank mit  $N$  Elementen etwas. Die Eingaben sind Schaltkreise für Orakelfunktionen, die genau ein Element auf 1 abbilden, oder das *leere Orakel*, das alle Elemente auf 0 abbildet. Das leere Orakel modelliert eine Datenbank, in der kein Element unserem Suchkriterium entspricht.

Erweiterte Suche

Wir zeigen zunächst, dass Grovers Algorithmus auch dieses leicht erweiterte Problem löst. Wie verhält sich dieser, wenn die Eingabe die leere Datenbank ist? Der Anfangszustand  $|s\rangle$  wird durch eine Groveriteration auf  $-|s\rangle$  abgebildet. Das Ergebnis der abschließenden Messung ist also ein Zufallselement. Wir betrachten zunächst einen Fall, in dem Grovers Algorithmus die Fehlerwahrscheinlichkeit 0 hat, zum Beispiel  $N = 4$  (siehe Beispiel 6.1). Wir führen den Algorithmus zweimal aus. Sind die beiden Ergebnisse verschieden, ist die Eingabe das leere Orakel. Ist das Ergebnis in beiden Ausführungen dasselbe, etwa  $a$ , argumentieren wir wie folgt: Handelt es sich um das leere Orakel, ist die Wahrscheinlichkeit für dieses Ergebnis nur  $\frac{1}{N}$ . Die Aussage „Die Eingabe ist nicht das leere Orakel und das gesuchte Element ist  $a$ “ hat beschränkten Fehler. Durch weitere Ausführungen des Algorithmus lässt sich der Fehler weiter verringern. Für ein  $N$ , bei dem die Erfolgswahrscheinlichkeit nicht 1, sondern nur „groß“ ist, können wir ganz ähnlich vorgehen.

Jeder Quantenalgorithmus zur Datenbanksuche benötigt mindestens  $\Omega(\sqrt{N})$  Datenbankabfragen. In diesem Sinne ist Grovers Algorithmus optimal.

Ziel des Abschnitts

Um diese Aussage zu beweisen, müssen wir unser mathematisches Instrumentarium etwas abstrakter handhaben als bei den bisherigen Analysen. Die Beweisidee ist allerdings nicht schwierig.

Beweisansatz

Wir müssen eine Behauptung über jeden möglichen Algorithmus für unser Problem beweisen. Wir nehmen an, wir haben einen Algorithmus  $A$  für die Datenbanksuche. Er ist uns als Quantenschaltkreis gegeben. An diesem Schaltkreis interessiert uns nur eines: Wie oft wird das Datenbankorakel  $U_f$  aufgerufen? Diese Anzahl bezeichnen wir als  $T$ , da die Datenbankaufrufe der uns interessierende Aspekt der Laufzeit sind. Unser Ansatz funktioniert für alle Weisen, auf die das Orakel implementiert sein kann; wir diskutieren das am Ende des Abschnitts.

Die allgemeine Form der Quantensuche

Wie sieht eine Berechnung von  $A$  aus, die Orakel  $U_f$  als Eingabe bekommen hat?  $A$  ruft  $T$ -mal das Orakel auf und führt dazwischen beliebige Berechnungen aus. Die Berechnungen zwischen den Orakelaufrufen können wir zu jeweils *einer* unitären Transformation zusammenfassen: mit  $U_i$  bezeichnen wir die unitäre Transformation, die vor dem  $i$ -ten Aufruf des Orakels ausgeführt wird. Wir beginnen mit einem beliebigen Startzustand  $|\phi_0\rangle$ . Unsere Berechnung ist dann von der Form

$$U_T \cdot U_f \cdot U_{T-1} \cdot U_f \cdots \cdots U_1 \cdot U_f |\phi_0\rangle. \quad (6.2)$$

Das ist tatsächlich die Grundform aller möglichen Algorithmen zur Suche mit einem Datenbankorakel. Die Transformationen  $U_T, \dots, U_1$  bestimmen die Art, in der gesucht wird.

Ein Algorithmus wie in Formel (6.2) beschrieben, muss für alle möglichen Eingaben, sprich Datenbankorakel, die richtige Lösung mit ausreichend hoher Wahrscheinlichkeit liefern. Was heißt das allgemein? Grovers Algorithmus erfüllt diese Bedingung. Es könnte allerdings andere Algorithmen geben, die ein Ergebnis liefern, aus dem man indirekt die richtige Lösung erschließen könnte. Allgemein ausgedrückt: die Ergebnisse nach Berechnungen mit verschiedenen Eingaben müssen *unterscheidbar* sein. In Abschnitt 3.6 haben wir gesehen, dass nur orthogonale Vektoren vollständig unterscheidbar sind. Sei  $v_x$  die Ausgabe des Algorithmus, wenn  $x$  gesucht ist. Bildet die Menge aller  $v_x, x \in \{0, 1\}^n$  eine Basis, messen wir bezüglich dieser Basis. Aus dem Ergebnis können wir das gesuchte Element erschließen.

Anforderungen

Wir verfolgen einen etwas anderen Ansatz und fordern von einem anwendbaren Quantensuchalgorithmus: für je zwei verschiedene Ergebnisse  $v_x$  und  $v_y$  muss der *Abstand*

$$\|v_x - v_y\|$$

durch eine Konstante  $c$  beschränkt sein, also von  $N$  unabhängig. Man kann sich anschaulich vorstellen, dass orthogonale Vektoren immer einen großen Abstand haben. Somit ist konstant beschränkter Abstand eine Minimalforderung an unseren Algorithmus, damit sich verschiedene Ergebnisse mit ausreichend großer Wahrscheinlichkeit unterscheiden lassen. Außerdem lässt sich zeigen, dass sich zwei Zustände nur dann mit *konstanter Wahrscheinlichkeit* unterscheiden lassen, wenn sie konstant weit auseinander liegen: Ist

der Abstand zweier Vektoren gering, so liegen auch die Projektionen auf die Achsen einer Messbasis dicht beieinander, und messen liefert für beide Vektoren ähnliche Ergebnisse, egal welche Basis zu Grunde liegt.

Konstante Wahrscheinlichkeit meint das folgende: Legen wir die gleichverteilte Superposition zu Grunde, erhalten wir beim Messen jedes  $|x\rangle$  mit Wahrscheinlichkeit  $\frac{1}{N}$ ; dieser Wert wird mit wachsendem  $N$  immer kleiner. Erkennen wir nun  $|\hat{x}\rangle$  für eine konstante, also von  $N$  unabhängige Wahrscheinlichkeit, haben wir einen großen Fortschritt gemacht (siehe auch Abschnitt 4.2). Andererseits ist das klarerweise eine Mindestforderung an unseren Suchalgorithmus.

Jeder Quantenalgorithmus  $A$  zur Datenbanksuche benötigt  $\Omega(\sqrt{N})$  Schritte, um  $|\hat{x}\rangle$  mit einer konstanten Wahrscheinlichkeit  $p > 0$  zu bestimmen.

Wir haben bereits gesehen, dass ein beliebiger Quantenalgorithmus  $A$  von der in (6.2) beschriebenen Form ist. Welche Transformationen  $U_1, \dots, U_T$  gewählt sind, legt das Vorgehen eines konkreten Suchalgorithmus  $A$  fest. Wir untersuchen einen solchen gegebenen Algorithmus  $A$  genauer und ermitteln ein zu suchendes Element  $\hat{x}$ , für das dieses festgelegte Vorgehen ungünstig ist. Wir konstruieren eine Eingabe  $U_g$ , für die  $T = \Omega(\sqrt{N})$  gilt; für die also das gesuchte Element  $|\hat{x}\rangle$  erst nach  $\Omega(\sqrt{N})$  Aufrufen des Orakels  $U_g$  mit positiver Wahrscheinlichkeit erkannt wird.

Beweisidee

Um  $A$  zu untersuchen, entnehmen wir die Orakelaufufe und setzen dafür die Identität  $I_N$  ein:

$$U_T \cdot I_N \cdot U_{T-1} I_N \cdot \dots \cdot U_1 I_N \cdot |\phi_0\rangle.$$

Wir erhalten einen Algorithmus, den wir  $A_0$  nennen. Er ist von der Eingabe unabhängig, und wir werden ihn in der Folge mit  $A$  vergleichen.

Wir ermitteln ein Datenbankelement, das zu den Zeitpunkten, zu denen in  $A$  das Orakel aufgerufen wird, eine sehr kleine Amplitude hat. Dabei hilft folgende Definition: Für ein Datenbankelement  $|x\rangle$  und einen Zeitpunkt  $t \leq T$  ist

$$\alpha_{t,x}$$

die Amplitude von  $|x\rangle$  nach  $t$  Schritten, also die im Zustand

$$U_t \cdot I_N \cdot U_{t-1} I_N \cdot \dots \cdot U_1 \cdot I_N |\phi_0\rangle.$$

Die Summe der Wahrscheinlichkeiten für unser  $|x\rangle$  über die ganze Berechnung bezeichnen wir als

$$a_x = \sum_{t=1}^T |\alpha_{t,x}|^2.$$

Als Summe von Wahrscheinlichkeiten ist

$$\sum_{x \in \{0,1\}^n} a_x = T.$$

Nach dem Schubfachprinzip<sup>3</sup> gibt es einen Datenbankeintrag  $|\hat{x}\rangle$  mit

$$a_{\hat{x}} \leq T/N.$$

Aus diesem  $|\hat{x}\rangle$  konstruieren wir eine Eingabe für  $A$ : wir definieren das Orakel  $U_g$  als jenes, für das  $|\hat{x}\rangle$  das gesuchte Element ist.

Wir können die Ergebnisse über gestörte Berechnungen aus Abschnitt 3.7 anwenden, um die Zustände während der Berechnung von Algorithmus  $A$  mit der Eingabe  $U_g$  mit den Zwischenzuständen von  $A_0$  zu vergleichen. Dazu fassen wir einen Schritt des Algorithmus  $A$  der Form

$$U_i U_g$$

als eine „fehlerhafte“ Abweichung von dem entsprechenden Schritt in  $A_0$

$$U_i I_N$$

auf und untersuchen, wie sich die Abweichungen summieren. Da wir wissen, wie die Auswirkung einer tatsächlich fehlerhaften Berechnung beschränkt ist, ist auch der wünschenswerte Unterschied zwischen den Ergebnissen von  $A_0$  und  $A$  beschränkt und wird erst bei langer Laufzeit hinreichend groß.

Sei  $|\phi_t\rangle$  das Zwischenergebnis der „ungestörten Berechnung“ ohne Orakel nach  $t$  Schritten:

$$|\phi_t\rangle = U_t I_N U_{t-1} \dots U_1 I_N |\phi_0\rangle.$$

Dann gilt

$$\|U_g |\phi_t\rangle - I_N |\phi_t\rangle\| \leq 2 \cdot |\alpha_{t,\hat{x}}|. \quad (6.3)$$

Denn  $U_g$  und  $I_N$  unterscheiden sich nur hinsichtlich ihres Verhaltens auf der Komponente  $|\hat{x}\rangle$ . In der gewohnten Form als  $V_g$  zum Beispiel nimmt Orakel  $U_g$  hier einen Vorzeichenwechsel vor; die anderen Komponenten werden wie von  $I_N$  nicht verändert.<sup>4</sup>

Wir wenden an, dass sich der „Fehler“ nur addiert (siehe Abschnitt 3.7): Sei  $|\psi_0\rangle$  der Endzustand nach der Rechnung mit dem Orakel  $I_N$  und  $|\psi_{\hat{x}}\rangle$  derjenige der Berechnung mit  $U_g$ . Dann gilt

$$\| |\psi_{\hat{x}}\rangle - |\psi_0\rangle \| \leq 2 \sum_{t=1}^T |\alpha_{t,\hat{x}}| \leq 2T/\sqrt{N}.$$

Verwendet wurde Gleichung (6.3) und die Tatsache, dass aus

$$\sum_{t=1}^T |\alpha_{t,\hat{x}}|^2 \leq T/N \quad (6.4)$$

die Ungleichung

$$\sum_{t=1}^T |\alpha_{t,\hat{x}}| \leq T/\sqrt{N}$$

<sup>3</sup>Siehe Abschnitt A.4 im Anhang.

<sup>4</sup>Die Ungleichung gilt für jede Form, in der das Orakel gegeben sein kann: siehe Bemerkung 1 am Ende des Abschnitts.

folgt, siehe Bemerkung 2. Wir haben eingangs gefordert, dass  $\| |\psi_{\hat{x}}\rangle - |\psi_0\rangle \| \geq p$  gelten soll. Daraus folgt

$$2T/\sqrt{N} \geq p$$

und wir erhalten

$$T \geq \frac{p}{2} \sqrt{N} = \Omega(\sqrt{N}).$$

*Zusammenfassung:* Für jeden Quantensuchalgorithmus der Form (6.2) können wir eine Eingabe konstruieren (ein Datenbankorakel), für die das Ergebnis der Berechnung erst nach  $\Omega(\sqrt{N})$  Schritten von einer Berechnung auf dem leeren Orakel (es gibt kein gesuchtes Element) unterschieden werden kann.

*Bemerkung 1:* Bisher haben wir Orakel stets in der Form des bedingten Vorzeichenwechsels angewendet, siehe Seite 138. Wir wollen uns vergewissern, dass das Ergebnis dieses Abschnitts davon nicht abhängt. Allgemein verwendet ein Orakel wie folgt Eingabebits  $|x\rangle$ , ein Orakelbit  $|y\rangle$  und eventuell noch einige Hilfsbits  $|z\rangle$ . Darauf wirkt es als

Allgemeine Form  
des Orakels

$$U_f : |x\rangle|y\rangle|z\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle|z\rangle.$$

Hat vor dem Aufruf der Zustand  $|x\rangle|0\rangle|z\rangle$  die Amplitude  $\alpha$  und  $|x\rangle|1\rangle|z\rangle$  die Amplitude  $\beta$ , so hat  $|x\rangle|0\rangle|z\rangle$  danach die Amplitude

$$(1 - f(x)) \cdot \alpha + f(x) \cdot \beta$$

und  $|x\rangle|1\rangle|z\rangle$  die Amplitude

$$f(x) \cdot \alpha + (1 - f(x)) \cdot \beta.$$

und die Abschätzung aus Gleichung (6.3) folgt.

*Bemerkung 2:* Aus der Cauchy-Schwarzschen Ungleichung

$$\sum_{t=1}^T |\alpha_t \beta_t| \leq \left( \sum_{t=1}^T |\alpha_t|^2 \right)^{\frac{1}{2}} \left( \sum_{t=1}^T |\beta_t|^2 \right)^{\frac{1}{2}}$$

folgt

$$\sum_{t=1}^T |\alpha_t| \leq \left( \sum_{t=1}^T |\alpha_t|^2 \right)^{\frac{1}{2}} \cdot \sqrt{T},$$

indem wir alle  $\beta_t$  gleich 1 setzen. Damit folgt aus Gleichung (6.4), dass

$$\sum_{t=1}^T |\alpha_{t,\hat{x}}| \leq \sqrt{T}/\sqrt{N} \cdot \sqrt{T}.$$

*Bemerkung 3:* Diese Bemerkung richtet sich an Leser mit großem Interesse an Komplexitätstheorie. Die eben präsentierte untere Schranke für die Quantensuche ist die einzige dieses Buches. Eine allgemeine Methode, untere Schranken für Quantenalgorithmen abzuleiten, basiert wie im klassischen Fall auf dem Rang von Polynomen. Mit einem ähnlichen Ansatz wie er in diesem Abschnitt im Anschluss an Formel (6.2) verfolgt wurde, kann zum Beispiel bewiesen werden, dass  $\Omega(2^n)$  Aufrufe einer Black Box Funktion  $f$  nötig sind, um zu entscheiden, ob  $|f^{-1}(1)|$  gerade ist. Siehe dazu etwa [12] oder [6].

## 6.7 Folgen für die Fähigkeiten von Quantencomputern

In diesem Kapitel wurde schon mehrfach erwähnt, dass Quantensuchalgorithmen sehr unterschiedliche Aufgaben lösen können, zum Beispiel Optimierungsprobleme. Letztlich kann jedes Berechnungsproblem als Suche in der Menge der möglichen Lösungen angesehen werden. Gesucht ist die tatsächliche Lösung. Da Suchalgorithmen derart universell sind, hat das Ergebnis des letzten Abschnitts weitreichende Konsequenzen. Dass jeder Quantenalgorithmus zur Orakel-Datenbanksuche  $\Omega(\sqrt{N})$  Anfragen benötigt, ist für die Frage relevant, ob Quantenrechner NP-vollständige Probleme effizient lösen können.

Wir betrachten noch einmal das Problem des Handlungsreisenden: Gibt es zwischen  $n$  Städten eine Rundreise kürzer einer Konstante  $k$ ? Dieses Problem ist NP-vollständig. Damit folgt: Ist es mit einem Quantenrechner effizient lösbar, dann sind *alle Probleme in NP* mit effizienten Quantenalgorithmen berechenbar.

In Abschnitt 6.5 haben wir ein darauf anwendbares Orakel konstruiert: Diesem wird eine Rundreise übergeben. Das Orakel antwortet mit 1, wenn diese kürzer als  $k$  ist; sonst mit 0. Mit diesem Orakel findet ein Quantenalgorithmus eine Rundreise kürzer  $k$  mit  $\Theta(\sqrt{N})$  Anfragen, wenn  $N$  die Anzahl der Rundreisen ist. Aber wie groß ist  $N$ ? Zwischen  $n$  Städten gibt es  $1 \cdot 2 \cdot \dots \cdot (n-1)$  Rundreisen,  $N$  ist damit von der Größenordnung  $\Theta(n^n)$  und  $\sqrt{N}$  ist immer noch eine verheerend große Zahl. Die quadratische Beschleunigung genügt nicht, die Rundreise in allen Fällen mit polynomialem Aufwand zu finden. Eine exponentielle Beschleunigung hingegen würde zu einem effizienten Algorithmus führen:  $\log n^n = \Theta(n \log n)$ .

Wäre die Quantensuche mit einem Datenbankorakel in Laufzeit  $O(\log N)$  möglich, könnten Quantenrechner alle Probleme in NP in Polynomialzeit lösen. Die Folge wäre  $\text{NP} \subseteq \text{BQP}$ .

Nun haben wir im letzten Abschnitt bewiesen, dass dies unmöglich ist: Grovers Algorithmus mit der Laufzeit  $O(\sqrt{N})$  ist optimal.

Wie in Abschnitt 4.3 beschrieben, gibt es eine große Zahl NP-vollständiger Probleme, die auf den ersten Blick sehr unterschiedlich aussehen. Für diese gilt nun: Entweder ist *kein* NP-vollständiges Problem mit Quantenalgorithmen effizient berechenbar ( $\text{NP} \not\subseteq \text{BQP}$ ) oder *alle* Probleme in NP liegen auch in BQP ( $\text{NP} \subseteq \text{BQP}$ ). Demnach lösen Quantenrechner alle NP-vollständigen Probleme oder keines. Dazwischen gibt es keine Alternative. Um alle zu lösen, müssen Quantenalgorithmen für extrem unterschiedliche Probleme aus einer exponentiell großen Menge von möglichen Lösungen die korrekten herausfischen. Und zwar in Polynomialzeit! Das wirkt so, als müsste man bei der unstrukturierten Suche mit  $O(\log N)$  Anfragen auskommen, was ja nun ausgeschlossen ist.

Wenn nun trotzdem  $\text{NP} \subseteq \text{BQP}$  gelten sollte, bedeutete das: *Alle* NP-vollständigen Probleme haben eine uns nicht bekannte gemeinsame Struktur, die ziemlich genau zu den Fähigkeiten von Quantenrechnern passt; diese Struktur erkennen wir, wenn wir für *ein* solches Problem einen effizienten Quantenalgorithmus finden.

Die meisten Komplexitätstheoretiker glauben nicht an die Existenz einer solchen Struktur und vermuten, dass Quantenrechner NP-vollständige Probleme nicht lösen können. Aber solange nichts bewiesen ist, ist die Frage ob

$$\text{NP} \subseteq \text{BQP} \text{ oder } \text{NP} \not\subseteq \text{BQP}.$$

offen.



## 7 Geheime Botschaften

*Die Welt ist alles, was der Fall ist, und  
auch alles, was der Fall sein kann.*

Anton Zeilinger

Wie kann ich vertraulich kommunizieren und sichergehen, dass nur der Adressat meiner Nachricht Kenntnis von dieser erlangt? Diese Frage lässt sich bis in die Antike zurückverfolgen und spielte insbesondere für das Militär eine wichtige Rolle. Unsere heutige Welt ist von digitaler Kommunikation, Internet und elektronischer Speicherung von persönlichen Daten geprägt. Somit werden Verschlüsselungsmethoden auch für Privatpersonen immer wichtiger.

Quantenverfahren bieten sich aus zwei Gründen für die sichere Datenübermittlung an: Quantenbits lassen sich nach dem No-Cloning-Theorem nicht kopieren, sofern die möglichen Zustände nicht orthogonal sind. Heimliches Belauschen einer Nachricht ist im Grunde ein Versuch, diese zu kopieren. Möchte man hingegen mittels Messungen etwas über ein Quantenbit herausbekommen, wird dieses im Allgemeinen verändert; dann kann ein unberechtigter Zugriff auf eine Nachricht im Nachhinein aufgedeckt werden. Das ist eine Eigenschaft von Quantenverschlüsselungsverfahren, die kein Gegenstück in der klassischen Welt hat. Wir lernen zwei der ersten veröffentlichten Verfahren kennen, die bis heute die praktisch wichtigsten sind.

Die in diesem Kapitel geschilderten Techniken lösen das kritischste Problem klassischer Verschlüsselungsverfahren: die Schlüsselerzeugung und Verteilung. Wir werden sehen, dass es sehr gute klassische Verschlüsselungsverfahren gibt, die auf geheimen Schlüsseln beruhen. Das Problem besteht darin, die Schlüssel auch tatsächlich geheim zu halten.

Wir beginnen mit einer kurzen Einführung in die Kryptographie und betrachten Voraussetzungen und Methoden zur Bewertung von Verschlüsselungstechniken: Wir stellen zunächst die Frage, auf die Quantenkryptographie eine Antwort gibt.

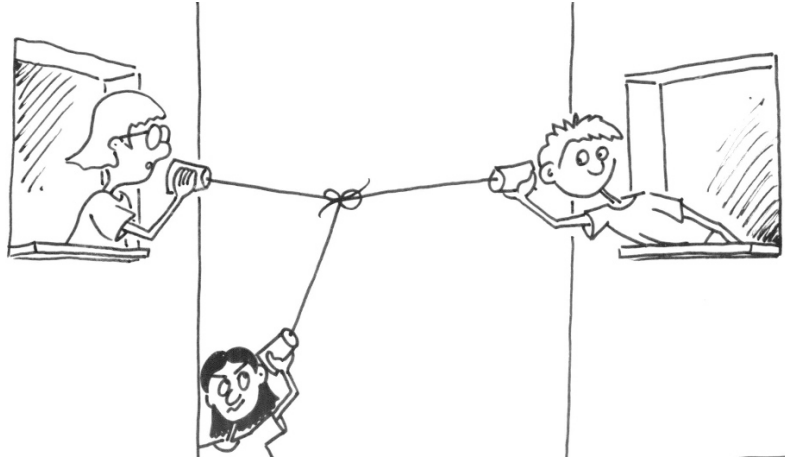


Abbildung 7.1: Kryptographische Grundsituation: Alice hat Bob Wichtiges vertraulich mitzuteilen und auch Eve zeigt sich interessiert.

## 7.1 Alice, Bob und Eve

Alice hat Bob etwas Wichtiges mitzuteilen. Eine dritte Person namens Eve interessiert sich ebenfalls für die Nachricht; Alice und Bob wollen mit allen Mitteln verhindern, dass Eve die Nachricht erfährt (*to eavesdrop* bedeutet *lauschen*, siehe Abbildung 7.1).

Das ist eine klassische Situation. Man könnte sich vorstellen, Alice sei die Geliebte von Bob und Eve Bobs misstrauische Ehefrau. Oder Alice ist das militärische Hauptquartier einer kriegführenden Partei, Bob gehört dem gleichen Lager an und Eve ist der gegnerische Nachrichtendienst. Ein ziviles Beispiel bilden Bankgeschäfte im Internet. Alice übermittelt einen Überweisungsauftrag an ihre Bank (Bob) und will unbedingt verhindern, dass ein anonymer Lauscher im Netz (Eve) Zugriff auf ihre Daten erhält. Zunächst dürfen Höhe des Geldbetrags und der Adressat nicht herauskommen, denn dafür interessieren sich das Finanzamt und auch der politische Gegner. Außerdem muss verhindert werden, dass Eve den zu überweisenden Betrag verändert oder das Geld auf ihr eigenes Konto leitet. Wir können demnach zwei Arten des unberechtigten Zugriffs unterscheiden.

1. Eve hört die Nachricht ab.
2. Eve manipuliert die Nachricht unbemerkt.

Klar- und  
Kryptotext

Dagegen können kryptographische Verfahren schützen. Alice' Mitteilung für Bob, nennen wir ab jetzt *Klartext* (englisch *plain text*). Der Klartext wird verschlüsselt, und es entsteht der *Kryptotext* (*cipher text*). Dieser wird an Bob geschickt. Fängt Eve den Kryptotext ab, soll sie daraus nicht erschließen können, was Alice mitzuteilen hat. Niemand außer Bob soll den Klartext aus dem Kryptotext erschließen können.

Es geht also nicht um Geheimhaltung durch organisatorische Maßnahmen, wie tote Briefkästen oder konspirative Treffen, und auch nicht um physikalische Vorkehrungen wie Siegel oder Schränke aus Panzerstahl. Es wird davon ausgegangen, dass die Kommunikation belauscht wird, dass Eve den Kryptotext erfährt. Eves Versuch, daraus den Klartext zu erschließen und damit das Verschlüsselungssystem zu brechen, heißt *Angriff*.

Kryptographische Verfahren wurden bereits von den alten Ägyptern vor fast 4000 Jahren eingesetzt. Auch ist bekannt, dass im antiken Griechenland militärische Geheimnisse verschlüsselt wurden. Ein wichtiger Schritt auf dem Weg zur modernen Kryptographie wurde im 19. Jahrhundert getan: der niederländische Militärkryptologe Auguste Kerckhoffs von Nieuwenhof (1835–1903) beschäftigte sich mit vertraulicher Kommunikation über Telegraphenleitungen. Ein auf ihn zurückgehender Grundsatz bei der Entwicklung und Bewertung von Verschlüsselungstechniken lautet: man muss stets davon ausgehen, dass das gerade verwendete Verschlüsselungsverfahren dem Lauscher bekannt ist. Das ist das sogenannte *Kerckhoffsche Prinzip*. In militärischen Apparaten oder Geheimdiensten sichern stets Informationen nach draußen. Existiert ein allgemein verkäufliches Verschlüsselungssystem auf dem Markt, kann man davon ausgehen, dass die Baupläne wenige Zeit später weit verbreitet sind. Das ist ein Problem einiger Dekodiergeräte im Bezahlfernsehen.

Wollen Alice und Bob ihre Nachricht, den Klartext, trotzdem geheim halten, empfiehlt sich nach dem Kerckhoffschen Prinzip die Verwendung eines *Schlüssels*, der leicht zu wechseln ist und Eve die Nachricht auch bei Kenntnis des Verfahrens verbirgt. Wir betrachten ein einfaches Beispiel:

Alice möchte eine Nachricht verschlüsseln, die aus großen lateinischen Buchstaben besteht. Dazu verwendet sie die sogenannte *Ersetzungschiffre*. Sie legt fest, wie jeder Buchstabe durch einen anderen ersetzt wird. Zum Beispiel gemäß der Regel:

Ersetzungschiffre

$$\begin{array}{llllllll} A \mapsto K, & B \mapsto N, & C \mapsto V, & D \mapsto A, & E \mapsto F, & F \mapsto C, & G \mapsto O, \\ H \mapsto S, & I \mapsto Z, & J \mapsto Y, & K \mapsto L, & L \mapsto B, & M \mapsto G, & N \mapsto U, \\ O \mapsto E, & P \mapsto X, & Q \mapsto J, & R \mapsto H, & S \mapsto W, & T \mapsto I, & U \mapsto M, \\ V \mapsto P, & W \mapsto T, & X \mapsto R, & Y \mapsto Q, & Z \mapsto D \end{array}$$

Damit wird aus dem Klartext

GUTENTAG

der Kryptotext

OMIFUIKO.

Dieser verschlüsselten Mitteilung ist der Klartext nicht anzusehen. Weiß der Adressat jedoch, nach welcher Regel die Buchstaben ersetzt wurden, kann er den Klartext wieder zugänglich machen.

Ein Schlüssel für die Ersetzungschiffre ist damit eine konkrete Ersetzungsregel wie oben beschrieben. Alice und Bob müssen beide den Schlüssel kennen und ihn vor Eve geheim halten. Weiß ein Spion, dem der Text OMIFUIKO in

die Hände fiel, dass er mit einer Ersetzungschiffre verschlüsselt wurde, kann er alle möglichen Schlüssel ausprobieren, bis er einen sinnvollen Klartext erhalten hat. Es gibt  $1 \cdot 2 \cdot 3 \cdot \dots \cdot 26$  mögliche Schlüssel; eine sehr große Zahl, mit heutigen Rechnern sollte der geschilderte Angriff jedoch ausführbar sein. Es gibt jedoch einen weit geschickteren Angriff; dieser nutzt statistische Methoden. Untersucht man die deutsche Sprache, stellt man fest, dass in einem typischen Text mehr als ein Viertel der verwendeten Buchstaben E oder N sind. Etwa 34 Prozent der Buchstaben sind I, S, R, A oder T.

Ordnen wir die Buchstaben nach abfallender Häufigkeit an, ergibt sich in etwa die Folge:

E N I S R A T D H U L C G M O B W F K Z P V J Y X Q

Daraus ergibt sich für Eve folgender Angriff auf die Ersetzungschiffre: Zähle in dem Geheimtext die Anzahl der vorkommenden Buchstaben und stelle folgende Vermutung über den verwendeten Schlüssel auf: der am häufigsten vorkommende Buchstabe ersetzt das E, der am zweithäufigsten vorkommende das N und so weiter. Wenn der Geheimtext nicht zu kurz ist, wird man mit diesem Verfahren sehr schnell den Klartext ermitteln können, da sich die häufigsten Buchstaben mit hoher Wahrscheinlichkeit an der richtigen Stelle befinden und schnell eine Identifizierung der Wörter erlauben.

Redundanz

Dieser Angriff nutzt die Redundanz der natürlichen Sprache aus: so kommt nur ein kleiner Teil der möglichen Buchstabenkombinationen darin vor. EI, ER, AN sind mögliche deutsche Wörter, EE, EF, EG hingegen nicht. Auch deutlich veränderte Texte einer natürlichen Sprache können noch verständlich sein:

DIE MTISEEN MCEENHSN KNNEÖN DSIEN STAZ FSAT FILSSÜG  
LSEEN.

In vorstehendem Text befinden sich nur der erste und der letzte Buchstabe eines Wortes an der korrekten Stelle; trotzdem gibt er seine Information vollständig preis. Mit mathematischen Methoden und Rechnerunterstützung können auch aus auf den ersten Blick raffiniert verschlüsselten Texten Informationen über den Klartext gewonnen werden.

Ergebnis

Die Ersetzungschiffre ist ein unsicheres kryptographisches Verfahren, weil der Kryptotext zu viel Information über den Klartext enthält. Ein Angriff mit statistischen Methoden bricht die Chiffre im Handumdrehen.

Die Ersetzungschiffre ist ein sehr einfaches Verfahren. Einem cleveren Verfahren muss man mit einem seinerseits cleveren Angriff begegnen. Das ist der Wettbewerb zwischen dem *Codemaker* und dem *Codebreaker*. Die Güte eines Verschlüsselungsverfahrens ergibt sich aus der Güte des besten Angriffs. Hilfsmittel sind dabei mathematische Analysen gepaart mit Rechenkraft. Die *National Security Agency (NSA)*, eine Unterabteilung des Department of Defense (Pentagon) der USA, das den internationalen Nachrichtenverkehr auswertet, gilt als einer der größten Abnehmer von Computerhardware und Arbeitgeber von Mathematikern. Ein Grund, bei Angreifern optimale Voraussetzungen anzunehmen.

Es gibt jedoch ein klassisches Kryptographieverfahren, das mathematisch beweisbar sicher ist, bei dem also auch die Möglichkeiten der NSA versagen. Allerdings ist es so aufwändig, dass die Einsetzbarkeit stark eingeschränkt ist. Darum werden *One-Time Pads* nur für Nachrichten eingesetzt, die einen hohen Aufwand rechtfertigen.

One-Time Pads

## One-Time Pads

Alice und Bob wollen eine Nachricht verschicken und kodieren diese als eine Folge von Bits. Texte können sie zum Beispiel mit dem ASCII-Code in eine Binärfolge umwandeln.

### 1. Schlüsselerzeugung

Der Schlüssel ist eine Folge von unabhängigen Zufallsbits. Alice erzeugt den Schlüssel  $k_1, \dots, k_m$  durch  $m$  Münzwürfe und übergibt Bob eine Kopie. Genauso gut kann Bob den Schlüssel erzeugen und an Alice schicken oder eine vertrauenswürdige dritte Partei könnte Zufallsbits herstellen und sowohl Alice als auch Bob zukommen lassen.

### 2. Verschlüsselung

Alice' Klartext ist die Folge von Bits  $a_1, \dots, a_m$ . Sie verwendet den ebenso langen Schlüssel  $k_1, \dots, k_m$ . Der Kryptotext lautet  $a_1 \oplus k_1, \dots, a_m \oplus k_m$ .

### 3. Entschlüsselung

Bob hat den Kryptotext  $b_1, \dots, b_m$  erhalten und addiert die Schlüsselbits, um Alice' Klartext zu erhalten:  $a_1, \dots, a_m = b_1 \oplus k_1, \dots, b_m \oplus k_m$ . Die Entschlüsselung ist korrekt, da

$$b_i \oplus k_i = a_i \oplus k_i \oplus k_i = a_i$$

ist.

Dieses Verfahren wurde 1918 von dem amerikanischen Mathematiker Gilbert Vernam (1890–1960) entwickelt, der für die AT&T Labs tätig war. Es wird für Mitteilungen der höchsten Sicherheitsstufe angewendet. So wurden während des kalten Krieges Mitteilungen zwischen Moskau und Washington auf diese Weise verschlüsselt.

One-Time Pads garantieren absolute Sicherheit, vorausgesetzt

Voraussetzungen

- die Schlüssel sind zufällig erzeugt,
- nur Alice und Bob kennen die Schlüssel,
- die Schlüssel werden nur einmal benutzt.

*Absolut sicher* bedeutet: Eve weiß gemäß dem Kerkhoffsschen Prinzip, dass Alice und Bob One-Time Pads verwenden, und den Kryptotext  $b_1, \dots, b_m$  hat sie auch abgefangen. Trotzdem kann sie *nichts* über den Klartext  $a_1, \dots, a_m$  herausbekommen, sofern sie den Schlüssel nicht kennt.

Wir betrachten ein einzelnes Bit: Eve hat das verschlüsselte Bit  $b_i = a_i \oplus k_i$  abgefangen. Sie weiß:  $k_i$  wurde ausgewürfelt. Da sie es nicht kennt, nützt ihr das abgefangene Bit  $b_i$  nichts. Hat dieses zum Beispiel den Wert 1, gibt es zwei Möglichkeiten. Ist das Schlüsselbit  $k_i = 0$ , so ist das ursprüngliche Bit  $a_i = 1$ . Ist  $k_i = 1$  gilt  $a_i = 0$ . Beide Fälle sind aus Eves Sicht gleich wahrscheinlich; somit enthält der Wert des verschlüsselten Bits  $b_i$  keine Information über  $a_i$ .

Da  $k_i$  ein Zufallsbit ist, ist ohne Kenntnis des Schlüssels auch  $a_i \oplus k_i$  eines, und genauso ist der gesamte Kryptotext  $a_1 \oplus k_1, \dots, a_m \oplus k_m$  eine Zufallsfolge! Es gibt keine Muster, Redundanzen und statistische Auffälligkeiten, da jedes Bit unabhängig von den anderen mit einem Zufallsbit überschrieben wurde.

Das gilt allerdings nur, wenn der Schlüssel tatsächlich nur einmal verwendet wird und genauso lang wie der Klartext ist. Verwenden Alice und Bob immer die gleichen Schlüssel, ist die Sicherheit dahin: Die Folge der mit demselben Schlüssel entstandenen Kryptotexte enthält Hinweise auf den Schlüssel und damit auf die Klartexte.

Nachteile der  
One-Time Pads

Den oben beschriebenen drei Voraussetzungen für die Sicherheit von One-Time Pads, entsprechen drei Nachteile:

- klassische Rechner können nur Pseudozufallszahlen erzeugen,
- das Problem der sicheren Kommunikation verschiebt sich auf die Übertragung der Schlüssel,
- die Schlüssel sind genauso lang wie die Nachricht und müssen für jede Kommunikation neu erzeugt werden.

Rolle der Quan-  
tenkryptographie

Quantenkryptographie kann diese Lücke füllen: sie erlaubt die Erzeugung und sichere Verteilung von *echten* Zufallsschlüsseln. Das ist auch dann möglich, wenn Alice und Bob das erste Mal miteinander kommunizieren. Versucht Eve, die Quantenschlüsselverteilung zu belauschen, können Alice und Bob das feststellen. In diesem Fall wird das Prozedere abgebrochen und zu einem späteren Zeitpunkt von vorn begonnen.

## 7.2 Quantenverschlüsselung: das BB84-Protokoll

Im Jahr 1984 haben Charles Bennett von IBM und Gilles Brassard von der Universität Montreal ein Quantenkryptographieverfahren veröffentlicht, das heute unter dem Namen BB84 bekannt ist. Fünf Jahre später wurde es das erste Mal experimentell umgesetzt, wobei die Quantenbits als polarisierte Photonen (siehe Abschnitt 10.2) realisiert wurden. In diesem Abschnitt stellen wir das BB84-Protokoll auf eine Weise dar, die unabhängig von der Umsetzung ist. Wir nehmen dazu an, dass zwischen Alice und Bob ein perfekter, störungsfreier Quantenkanal existiert. Da Kanäle in der Praxis nie störungsfrei sind, betrachten wir am Ende des Abschnittes die Auswirkung von Rauschen auf das geschilderte Verfahren.

Die Aufgabe lautet, Alice und Bob mit einer Folge von *zufälligen* und geheimen Bits auszustatten. Diese können dann als One-Time Pads zur sicheren klassischen Verschlüsselung genutzt werden.

Wir beginnen mit der Erzeugung eines einzelnen Bits und setzen danach das eigentliche Schlüsselübertragungsprotokoll aus einer Folge solcher Schritte zusammen. Die Biterzeugung beginnt mit einer Aktion von Alice.

### A. Alice verschickt ein Quantenbit

Verwendet wird ein Quantenbit  $|x\rangle$ .

1. Erzeuge ein zufälliges klassisches Bit  $a$ ;  
Versetze das Quantenbit in den Zustand  $|x\rangle \leftarrow |a\rangle$
2. Erzeuge ein zweites klassisches Zufallsbit  $a'$ :  
Ist  $a' = 1$ , wende die Hadamard-Transformation an:  $|x\rangle \leftarrow H|x\rangle$
3. Schicke  $|x\rangle$  an Bob.

Wir bezeichnen im folgenden

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = H|0\rangle \text{ mit } |+\rangle$$

und

$$\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = H|1\rangle \text{ mit } |-\rangle.$$

In welchem Zustand befindet sich das Bit  $|x\rangle$  nach Schritt 3? Mit Wahrscheinlichkeit von jeweils  $1/4$  in einem der Zustände

$$|0\rangle, |1\rangle, |+\rangle, |-\rangle.$$

Das Quantenbit  $|x\rangle$  wurde gemäß folgender Tabelle aus den Zufallsbits erzeugt:

	$a' = 0$	$a' = 1$
$a = 0$	$ 0\rangle$	$ +\rangle$
$a = 1$	$ 1\rangle$	$ -\rangle$

Das Zufallsbit  $a$  soll an Bob übermittelt werden. Das Zufallsbit  $a'$  bestimmt, in welcher Basis  $a$  dargestellt wird. Mit  $B$  bezeichnen wir ab jetzt die Standardbasis  $|0\rangle, |1\rangle$ , mit  $B'$  die Hadamardbasis  $|+\rangle, |-\rangle$ . Alice' zweiter Münzwurf entscheidet also, welche Basis für die Übertragung gewählt wird.

Nun hat Bob das Quantenbit  $|x\rangle$  erhalten. Wie kann er etwas über das Bit  $a$  herausfinden? Er muss es messen. Da er nicht weiß, bezüglich welcher Basis es kodiert wurde, überlässt er diese Entscheidung dem Zufall.

Verwendete  
Basen

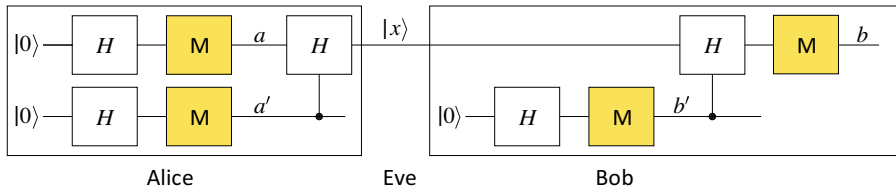


Abbildung 7.2: Schaltkreis für die Schritte A. und B.: Der Quantenkanal zwischen Alice und Bob ist der unsichere Teil.

### B. Bob misst Alice' Quantenbit

$|x\rangle$  ist das von Alice in Schritt A erzeugte und verschickte Quantenbit.

1. Erzeuge ein zufälliges Bit  $b'$ :  
Ist  $b' = 0$ , miss  $|x\rangle$  bezüglich  $B = \{|0\rangle, |1\rangle\}$ ,  
ist  $b' = 1$ , miss  $|x\rangle$  bezüglich  $B' = \{|+\rangle, |-\rangle\}$ .
2. Teile Alice über einen klassischen Kanal mit, bezüglich welcher Basis gemessen wurde.

Das Ergebnis der Messung ergibt ein Bit  $b$ . Dieses ist 0, falls  $|x\rangle$  vor der Messung im Zustand  $|0\rangle$  war und in  $B$  gemessen wurde, oder im Zustand  $|+\rangle$  war und in  $B'$  gemessen wurde. Entsprechend ist es 1 im Falle der Kombinationen  $|1\rangle$  und  $B$  bzw.  $|-\rangle$  und  $B'$ . Die Schritte A. und B. lassen sich wie in dem Schaltkreis in Abbildung 7.2 ausführen.

**Aufgabe 7.1:** Zeigen Sie: Ist  $b' = a'$ , so ist auch  $b = a$ .

Aufgabe 7.1 zeigt: Haben Alice und Bob die gleiche Basis gewählt, sind sie im Besitz desselben zufälligen Bits. Im anderen Fall hat Bob nichts über Alice' Bit erfahren. Hat Alice etwa  $|+\rangle$  verschickt und misst Bob in der Basis  $B$ , erhält er die beiden möglichen Ergebnisse mit jeweils derselben Wahrscheinlichkeit.

**Aufgabe 7.2:** Zeigen Sie: Ist  $b' \neq a'$ , so sind die Ergebnisse  $b = a$  und  $b \neq a$  gleich wahrscheinlich.

Verwenden Alice und Bob verschiedene Basen, ist das Ergebnis von Bobs Messung nach Aufgabe 7.2 ein Zufallsbit, das unabhängig von dem Wert von Alice' Bit  $a$  ist. Darum teilt Bob in Schritt B.2 Alice die gewählte Basis mit. Entscheidend ist nun: Im weiteren verwenden Alice und Bob die Bits  $a$  und  $b$  nur dann für den Schlüssel, wenn sie die gleiche Basis verwendet haben.



### C. Alice und Bob entscheiden, ob das Zufallsbit verwendbar ist

1. Alice teilt Bob mit, ob sie die gleiche Basis verwendet haben.  
Falls nicht, nutzen sie das Ergebnis nicht.

In etwa der Hälfte der Fälle benutzt Bob die gleiche Basis wie Alice. Tauschen die beiden auf diese Weise eine Reihe Bits aus, können sie erwarten, dass dies für die Hälfte zutrifft.

Die folgende Tabelle fasst das Vorgehen zusammen. Der Eintrag Z steht für ein Zufallsbit: mit Wahrscheinlichkeit von jeweils  $1/2$  ist dessen Wert 0 oder 1. Im Fall  $a' \neq b'$  ist  $b$  – das Ergebnis von Bobs Messung – ein solches Zufallsbit und  $a$  beziehungsweise  $b$  werden nicht als Schlüssel verwendet. Statt von *Schlüsselverteilung* spricht man deshalb besser von *Schlüsselerzeugung*. Es ist nämlich nicht so, dass der Schlüssel schon zu Beginn feststeht und nur verschickt wird: er entsteht erst im Verlauf des Kommunikationsprozesses, da Bobs Wahl von  $b'$  die Gestalt des Schlüssels mitbestimmt.

$a$	0	0	1	1	0	0	1	1
$a'$	0	1	0	1	0	1	0	1
$ x\rangle$	$ 0\rangle$	$ +\rangle$	$ 1\rangle$	$ -\rangle$	$ 0\rangle$	$ +\rangle$	$ 1\rangle$	$ -\rangle$
$b'$	0	0	0	0	1	1	1	1
$b$	0	Z	1	Z	Z	0	Z	1
Schlüssel	0		1			0		1

### Eves Zug

Nach den Schritten A bis C – eventuell müssen diese mehrfach ausgeführt werden – besitzen Alice und Bob ein gemeinsames Zufallsbit. Nun kommen wir zu dem interessanten Punkt des Verfahrens, zu Eve. Das Ziel ist ja gerade, dass diese nichts über den Schlüssel, sprich das Bit  $a$  erfährt. Eve steht damit natürlich stellvertretend für alle unbefugten Personen.

Gemäß dem Kerkhoffsschen Prinzip gehen wir davon aus, dass Eve das Prozedere genau kennt. Sie hat vollen Zugriff auf den Quantenkanal, und auch den klassischen Kanal kann sie belauschen, auf dem sich Alice und Bob über die verwendete Basis austauschen. Dadurch erfährt sie die Bits  $a'$  und  $b'$ ; aber erst *nachdem* Bob gemessen hat. Kann Eve mit diesem Wissen das Bit  $a$  bestimmen?

Messen und  
weeterschicken

Nach dem No-Cloning-Theorem (Abschnitt 3.4) kann Eve das Quantenbit  $|x\rangle$  nicht kopieren, denn dessen vier mögliche Zustände sind nicht orthogonal. Damit bietet sich folgender Angriff an: Sie misst das Quantenbit und schickt es dann an Bob weiter – oder erzeugt gemäß dem Messergebnis ein neues Quantenbit, das sie Bob übermittelt, als käme es von Alice.

1. Eve wählt ein Bit  $e'$ .
2. Ist  $e' = 0$ , misst sie bezüglich  $B$ , sonst bezüglich  $B'$ .

Die Erzeugung von  $e'$  lassen wir offen. Hier könnte irgendeine raffinierte Strategie oder auch ein Zufallsverfahren zum Zuge kommen. Eve kann dabei ausschließlich ihr erlaushetes Wissen verwenden; das Bit  $e'$  ist darum *unabhängig* von den Bits, die Alice und Bob erzeugen. Natürlich könnte Eve auch in anderen Basen als  $B$  und  $B'$  messen. Man kann sich jedoch überlegen, dass daraus kein Vorteil entsteht: so lässt sich das Grundproblem nicht umgehen, dass die möglichen Zustände des zu belauschenden Bits nicht orthogonal sind. Im nächsten Abschnitt diskutieren wir allgemeinere Lauschstrategien, die diesen Fall einschließen.

Wüsste Eve, bezüglich welcher Basis Alice das Zufallsbit kodiert hat, könnte sie erfolgreich und unbemerkt lauschen. Denn eine Messung in der korrekten Basis beeinflusst den Zustand nicht. Da dies nicht der Fall ist, wird sie – wie Bob – in der Hälfte der Fälle die falsche Basis wählen, egal nach welcher Strategie sie  $e'$  wählt.

**Beispiel 7.1:** Wir nehmen an, dass

$$|x\rangle = |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

ist, das heißt  $a = 0, a' = 1$ .

1. Wählt Eve  $e' = 1$  und misst in der Basis  $B' = \{|+\rangle, |-\rangle\}$ , ist das Ergebnis  $|+\rangle$ . Sie folgert  $a = 0$  und das Quantenbit  $|x\rangle$  wurde nicht verändert.
2. Anderenfalls –  $e' = 0$  – misst Eve in der Basis  $B = \{|0\rangle, |1\rangle\}$ . Das Ergebnis ist mit jeweils gleicher Wahrscheinlichkeit  $|0\rangle$  oder  $|1\rangle$ , und sie hat nichts über das Bit  $a$  erfahren. Mehr noch: Bit  $|x\rangle$  wurde verändert. Es ist in gerade den Zustand übergegangen, den Eve beobachtet hat.  $\diamond$

Bei wiederholtem Lauschen, kann Eve auf diese Weise dennoch einiges über den ausgetauschten Schlüssel herausbekommen. Alice und Bob verwenden ein Bit nur im Fall  $a' = b'$ . Da  $e'$  von der Wahl der Zufallsbits von Alice und Bob unabhängig ist, hat  $e'$  mit Wahrscheinlichkeit  $1/2$  den gleichen Wert, und Eve erfährt aus dem Basenaustausch in Schritt C, ob das der Fall ist. Anderenfalls ist ihr Messergebnis ein wertloses Zufallsbit, aber immerhin: sie

erfährt die Hälfte des Schlüssels. Wir folgern: die Schritte A bis C allein sind unsicher!

Aber die Bits, die Eve in der falschen Basis gemessen hat, sind verändert. Das können Alice und Bob ausnutzen, um einen Test auf unbefugten Zugriff auszuführen. Dazu müssen sie ein bereits korrekt übertragenes Bit opfern.

#### D. Alice und Bob opfern ein Bit, um Eve zu entlarven

Alice und Bob haben ein übertragenes Bit  $|x\rangle$  bezüglich derselben Basis gemessen: Es gilt  $a' = b'$  und deshalb erwarten sie  $a = b$ .

1. Alice und Bob tauschen  $a$  und  $b$  über den klassischen Kanal aus.
2. Stimmen die Ergebnisse nicht überein, brechen sie die Schlüssel-erzeugung ab und verwenden die bereits übertragenen Schlüssel nicht.
3. Stimmen die Ergebnisse überein, fahren Sie mit der Schlüsselübertragung fort, verwenden aber das ausgetauschte Bit nicht.

Wie groß ist die Wahrscheinlichkeit, Eves Lauschversuch aufzudecken? Voraussetzung von Teil D ist  $a' = b'$ . Hat Eve die Basis korrekt geraten –  $a' = b' = e'$  – stimmen die Bits  $a$  und  $b$  überein und Eve wird nicht entlarvt. Hat Eve falsch geraten, projiziert die Messung das ausgetauschte Bit  $|x\rangle$  in die falsche Basis. Damit ist Bobs Messung ein Zufallsbit – wie in dem Fall, in dem er selbst die falsche Basis wählt – und mit Wahrscheinlichkeit  $1/2$  gilt  $a \neq b$ . Insgesamt wird Eve in diesem Schritt also mit Wahrscheinlichkeit  $1/4$  entdeckt.

**Aufgabe 7.3:** Rechnen Sie nach: falls Eve bezüglich der falschen Basis gemessen hat, gilt mit Wahrscheinlichkeit  $1/2$ , dass  $a \neq b$ .

Die folgende Tabelle zeigt, wie Bits zur Enttarnung Eves verwendet werden. Im Falle 'Z' wird Eves Zugriff mit Wahrscheinlichkeit  $1/2$  aufgedeckt.

$a$	0	0	1	1	0	0	1	1
$a'$	0	1	0	1	0	1	0	1
$b'$	0	1	0	1	0	1	0	1
$e'$	0	0	0	0	1	1	1	1
$b$	0	Z	1	Z	Z	0	Z	1

Die Situation des  
Lauschers

Wir fassen Eves Situation zusammen. Da Alice das Bit  $a'$  zufällig wählt, gibt es für Eve keine gute Strategie, in der richtigen Basis zu messen. In der Hälfte der Fälle wählt sie die falsche und erhält ein Zufallsbit. Das ist aus Eves Sicht nicht schlimm, da sie so immerhin die Hälfte des Schlüssels zuverlässig ertauscht. Schlimm für Eve und gut für Alice und Bob ist: Eves Lauschangriff wird dann in der Hälfte der Fälle aufgedeckt. Alice und Bob brechen in diesem Fall die Schlüsselerzeugung ab.

Nun ist es an der Zeit, das Protokoll als Ganzes zu diskutieren.

### Das BB84-Protokoll zur Schlüsselerzeugung

Benötigt werden ein Quantenkanal und ein klassischer Kanal.

Falls das Protokoll nicht wegen Lauschverdachts abgebrochen werden muss, ergibt sich der Schlüssel aus Alice' Bits  $a_1, \dots, a_m$  und Bobs Bits  $b_1, \dots, b_m$ , die in den Schritten 4 und 5 nicht verworfen werden.

1. Alice erzeugt Zufallsbits  $a_1, \dots, a_m$  und  $a'_1, \dots, a'_m$ .
2. Alice erledigt für  $i = 1, \dots, m$ :
  - kodiere  $a_i$  als  $|0\rangle$  beziehungsweise  $|1\rangle$  falls  $a'_i = 0$
  - kodiere  $a_i$  als  $|+\rangle$  oder  $|-\rangle$  falls  $a'_i = 1$
  - schicke das entstandene Quantenbit an Bob.
3. Bob erzeugt Zufallsbits  $b'_1, \dots, b'_m$ .  
Das  $i$ -te Quantenbit von Alice misst er
  - in der Basis  $B = \{|0\rangle, |1\rangle\}$  falls  $b'_i = 0$
  - in der Basis  $B' = \{|+\rangle, |-\rangle\}$  falls  $b'_i = 1$
 und speichert das Ergebnis als  $b_i$ .
4. Alice und Bob vergleichen für  $i = 1, \dots, m$  die Bits  $a'_i$  und  $b'_i$  über einen klassischen Kanal.  
Ist  $a'_i \neq b'_i$  werden  $a_i$  beziehungsweise  $b_i$  nicht verwendet.
5. Alice und Bob tauschen  $k$  der nicht gelöschten Bits  $a_i, b_i$  aus und ermitteln die Fehlerrate, definiert als *Anzahl der sich unterscheidenden Bits geteilt durch  $k$* .  
Ist die Fehlerrate zu hoch, verwenden sie die erzeugten Bits nicht, da Verdacht auf einen Angriffe seitens Eve besteht.

Den letzten Schritt sollten wir noch eingehender betrachten. Wir haben bereits gesehen, dass Eves oben beschriebene Lauschstrategie für jedes der  $k$  geopfert Bits mit Wahrscheinlichkeit  $1/4$  einen Fehler verursacht. Zu

Beginn dieses Abschnitts haben wir einen perfekten Quantenkanal vorausgesetzt. Somit hat ein unbelauschter Ablauf die Fehlerrate 0. Belauscht Eve jedes Bit, bleibt sie nur mit Wahrscheinlichkeit  $(3/4)^k$  unerkannt. Die Wahrscheinlichkeit, einen Lauschangriff zu entdecken, wächst exponentiell in der Zahl der dafür geopfert Bits; das Restrisiko, dass Eve mit dem Schlüssel unbemerkt entwischt, wird bei geeignet groß gewähltem  $k$  so klein, dass es praktisch irrelevant ist. Zumal wir ihr bereits optimale Voraussetzungen zugestanden haben. Aus Zweckpessimismus ignorieren wir, dass Eves Zugriff auf die Kanäle in realen Fällen meist weniger absolut sein wird. Wir folgern: *Das BB84-Protokoll erlaubt sichere Schlüsselverteilung.* Aber nur unter den Voraussetzungen, von denen unsere bisherige Analyse ausging:

1. Eve verwendet die oben beschriebene Lauschstrategie.
2. Der Quantenkanal ist perfekt.

Punkt 1 werden wir im nächsten Abschnitt diskutieren. Punkt 2 ist deshalb wichtig, weil in der Praxis keine perfekten Quantenkanäle zur Verfügung stehen. Bei einem realen Kanal hat unser Protokoll daher auch unbelauscht eine Fehlerrate größer 0. Belauscht Eve jedes Bit, ist die erwartete Fehlerrate  $1/4$ . Heutige Übertragungsmethoden lassen unbelauscht deutlich niedrigere Fehlerraten zu, so dass die Fälle *belauscht* und *nicht belauscht* trotz Rauschens unterschieden werden können.

Kryptographie  
über verrauschte  
Kanäle

Allerdings bietet sich dann für Eve die folgende Strategie an: Sie misst nur wenige Bits. Auf diese Weise erhält sie nur einen geringen Teil des Schlüssels, die Auswirkungen ihrer Lauschtätigkeit könnten – geschickt kalkuliert – in dem allgemeinen Rauschen unentdeckt bleiben. Diesem Problem begegnet man mit dem Einsatz fehlerkorrigierender Codes und dem folgenden Verfahren.

### Verstärkung der Sicherheit

Eine Methode namens *privacy amplification* erlaubt es, aus einem nur teilweise geheimen Schlüssel einen kürzeren, sehr geheimen Schlüssel zu erzeugen; und zwar mittels öffentlicher Diskussion.

Privacy  
amplification

Wir stellen uns vor, Alice und Bob haben zwei Schlüsselbits  $b_1$  und  $b_2$  ausgetauscht. Eve hat eines davon erfahren. Darum beschließen Alice und Bob, aus den zwei Schlüsselbits ein einzelnes neues zu erzeugen:

$$b_3 = b_1 \oplus b_2.$$

Angenommen, Eve weiß, dass  $b_1 = 0$  gilt. Über  $b_2$  weiß sie hingegen nichts: aus ihrer Sicht sind die Fälle  $b_2 = 0$  und  $b_2 = 1$  gleichwahrscheinlich. Dann weiß sie über  $b_3$  ebenso wenig! Ihre Kenntnis von  $b_1$  verrät nichts über  $b_3$ , da  $b_1$  mit dem unbekannten, aus Eves Sicht zufälligen Bit  $b_1$  überschrieben ist; wie bei der Verschlüsselung mittels One-Time Pads.

Genauso ist die Summe  $b_1 \oplus b_2 \oplus \dots \oplus b_k$  ein ihr unbekanntes Zufallsbit, wenn sie *eines* der  $k$  Bits nicht kennt. Um aus einer Folge von  $m$  Zufallsbits  $n$  neue zu erzeugen ( $n < m$ ), wählt man zufällig  $n$  Teilmengen der Bits aus. Die Summe der Bits jeweils einer Teilmenge ergeben ein neues Zufallsbit. In [20] ist dieses Verfahren analysiert worden und hat sich als effizient erwiesen, insbesondere in Verbindung mit fehlerkorrigierenden Codes. Die Fehlerrate des BB84-Protokolls gibt einen deutlichen Hinweis darauf, wieviele Bits Eve kennen kann. Verkürzen Alice und Bob mittels *privacy amplification* den Schlüssel um diese Zahl, kennt Eve mit großer Wahrscheinlichkeit nur noch sehr wenige Bits.

Zusammenfassung

Was bringt die Quantenkryptographie Neues? Unberechtigte Versuche, auf die übermittelten Daten zuzugreifen, werden aufgedeckt. Ein Lauschangriff auf ein einzelnes Bit wird zwar nur mit einer bestimmten Wahrscheinlichkeit bemerkt. Wird jedoch ein längerer Schlüssel übertragen ist es praktisch nicht möglich, relevante Informationen über diesen zu erlangen, ohne entdeckt zu werden. Verlassen wir zur Erläuterung kurz das Gebiet der Kryptographie: Ein Einbruch in ein hochmodernes Sicherheitssystem, zum Beispiel in den Safe einer Zentralbank, kann theoretisch unbemerkt bleiben. Nach den Gesetzen der klassischen Physik spricht nichts dagegen, dass Eve das System in exakt dem gleichen Zustand zurücklässt, in dem sich dieses vor dem Eindringen befand. Dagegen folgt aus den Gesetzen der Quantenmechanik, dass ihre Messungen am Quantenkanal deutliche Spuren hinterlassen, möchte sie Wesentliches über den ausgetauschten Schlüssel erfahren.

Die Möglichkeit, Lauschangriffe aufzudecken, ist der wesentliche Beitrag der Quantenmechanik. Dazu gibt es im klassischen Fall kein Pendant. Durch das beschriebene Verfahren werden die sicheren, aber durch das Problem der Schlüsselverteilung unhandlichen, One-Time Pads praktikabel.

## 7.3 Lauschstrategien

Im letzten Abschnitt wurde das BB84-Protokoll eingeführt. Um zu analysieren, wie sicher das Verfahren ist, müssen wir die möglichen Angriffe von Eve untersuchen. Wir sind bisher davon ausgegangen, dass Eve das übermittelte Quantenbit misst und weiterschickt. Diese Strategie liegt nah, da sich Quantenbits nicht kopieren lassen. Ein Vorteil dieser Lauschstrategie ist, dass Eve kein Quantenbit speichern muss: das wäre mit beträchtlichem Aufwand verbunden und ist zurzeit nur für kurze Zeitspannen möglich. Allerdings sind auch Angriffe interessant, die bisher nur theoretisch möglich sind. So können Alice und Bob nicht wissen, ob Eves technische Fähigkeiten nicht viel weiter fortgeschritten sind, als sie glauben.

Im letzten Abschnitt haben wir gesehen: Wenn Eve die übermittelten Bits misst, werden diese Eingriffe mit hoher Wahrscheinlichkeit entdeckt. Darum untersuchen wir einen anderen Ansatz. Wir setzen voraus, dass Eve im Besitz einer Reihe von Quantenbits  $|e_1\rangle, |e_2\rangle, \dots$  ist. Sie wendet unitäre Transformationen auf den Kanal und diese Quantenbits an und versucht, so viel Information über das gesendete Bit wie möglich auf ihre Bits zu

übertragen. Das andere wesentliche Ziel ist natürlich weiterhin, unentdeckt zu bleiben.

Die Sicherheit des BB84-Protokolls – angereichert um die Verwendung von fehlerkorrigierenden Codes und *privacy amplification* – allgemein zu untersuchen, ist eine Aufgabe, die jenseits der in diesem Buch vermittelten Methoden liegt. Doch bereits im letzten Abschnitt haben wir erfahren: Wenn Eve die gesendeten Quantenbits misst und weiterschickt, kann sie die Sicherheit des Protokolls nicht gefährden. Dieser Abschnitt soll ohne einen formalen Beweis Argumente dafür liefern, dass auch allgemeinere Angriffe entweder bemerkt werden oder nur sehr wenig über den Schlüssel in Erfahrung bringen.

Wir beginnen mit einem einfachen Beispiel.

**Beispiel 7.2:** Eve erinnert sich daran, dass man mit einem CNOT-Gatter Quantenbits miteinander verschränken kann. Wäre das nicht die Lösung? Sie verschränkt ein Quantenbit in ihrem Besitz mit dem gesendeten. Dann wartet sie ab, bis Alice und Bob ihre Basen vergleichen. Sie belauscht diesen Schritt und verwendet die gewonnene Information, wenn sie ihr eigenes Quantenbit misst: sie nutzt es für die Wahl einer Messbasis.

1. Wir nehmen an, Alice sendet  $|x\rangle = |1\rangle$ . Eve führt die Operation CNOT auf diesem Bit und einem vorbereiteten Quantenbit  $|e\rangle$  im Zustand  $|0\rangle$  aus. Das Ergebnis: Die Bits sind im Zustand

$$|x\rangle|e\rangle = |1\rangle|1\rangle.$$

Eve wartet Bobs Messung ab und belauscht, welche Basis er verwendet hat. Misst Bob in der richtigen – in  $B$  also – tut sie es ihm gleich und erfährt das Schlüsselbit  $a = b = 1$ . Misst Bob in der anderen Basis, verwenden Alice und Bob das Bit nicht, und Eve ignoriert es ebenso.

2. Sendet Alice hingegen  $|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ , überführt das CNOT-Gatter die Bits in den Zustand

$$|x\rangle|e\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle).$$

BB84 deckt  
Verschränkung  
auf

2.1 Bob misst in der Basis  $B$ . Er beobachtet  $|0\rangle$  oder  $|1\rangle$  mit gleicher Wahrscheinlichkeit. Eve misst ihr Bit und erhält dasselbe Ergebnis wie Bob. Das ist schließlich der Grund, ihr Quantenbit mit dem gesendeten zu verschränken. Allerdings wird dieses Bit nicht verwendet, da Bobs Basis nicht mit der von Alice übereinstimmt.

2.2 Das nächste übermittelte Bit ist ebenfalls im Zustand  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ , diesmal misst Bob in der richtigen Basis  $B'$  – und das BB84-Protokoll enttarnt Eve in der Hälfte der Fälle!

Das zeigt folgende Rechnung: Ein Bit in der Basis  $B'$  zu messen, entspricht einer Anwendung der Hadamard-Transformation mit an-

schließender Messung in  $B$ . Bob misst das erste Bit des verschränkten Paares  $|x\rangle|e\rangle$ .

$$\begin{aligned} H(|x\rangle)|e\rangle &= \frac{1}{\sqrt{2}} \left( (H|0\rangle)|0\rangle - (H|1\rangle)|1\rangle \right) \\ &= 1/2(|00\rangle - |01\rangle + |10\rangle + |11\rangle) \end{aligned}$$

Das Ergebnis stimmt in der Hälfte der Fälle nicht mit Alice' Bit überein, obwohl die gleichen Basen verwendet wurden.

In diesem Fall hat die Anwendung des CNOT-Gatters die gleiche Auswirkung auf Bobs Ergebnis wie eine Messung in der falschen Basis  $B$ .

Es gibt ein Analogon zu CNOT, dessen Einfluss in der Basis  $B$  aufgedeckt wird und in der Basis  $B'$  unbemerkt bleibt. Die Situation ist die gleiche wie beim direkten Messen: da die möglichen Zustände des Quantenbits  $|x\rangle$  nicht orthogonal sind, wird die Verschränkung mit Eves Bits bei Bobs Messung in einem Viertel der Fälle aufgedeckt.  $\diamond$

Ohne es formal bewiesen zu haben, hat uns Beispiel 7.2 folgendes einsehen lassen: Verschränkt Eve eines ihrer Quantenbits mit  $|x\rangle$ , dem über den Quantenkanal versendeten Bit, wird das Ergebnis von Bobs Messung genauso beeinflusst, als würde Eve das Bit  $|x\rangle$  messen. Damit haben wir einen weiteren speziellen Angriff untersucht. Nun wollen wir uns der Aufgabe nähern, mögliche Angriffe von Eve allgemein zu beschreiben.

Angriffe ohne  
Verschränkung

Wir verallgemeinern nun Eves Möglichkeiten. Die folgende Angriffsart schöpft sie zwar noch nicht ganz aus, ist dafür jedoch sehr einfach zu analysieren.

Eve möchte unterscheiden, ob das von Alice an Bob gesendete Quantenbit  $|x\rangle$  im Zustand  $|0\rangle$  oder im Zustand  $H|0\rangle = |+\rangle$  ist. Dazu wendet sie eine unitäre Zwei-Bit-Transformation  $U$  auf den Kanal (das heißt, auf  $|x\rangle$ ) und ihr Bit an, insgesamt also auf  $|x\rangle|e\rangle$ . Den Anfangszustand von  $|e\rangle$  nennen wir  $|\phi_a\rangle$ . Anschließend misst Eve  $|e\rangle$  in irgendeiner Weise.

1. Eve will absolut unbemerkt bleiben und fordert, dass die Transformation  $U$  das Bit  $|x\rangle$  unverändert lassen soll. Dann gibt es zwei von der Wahl von  $U$  abhängige Einheitsvektoren  $|\phi_0\rangle, |\phi_+\rangle$ , so dass

$$|0\rangle|\phi_a\rangle \xrightarrow{U} |0\rangle|\phi_0\rangle \text{ und } |+\rangle|\phi_a\rangle \xrightarrow{U} |+\rangle|\phi_+\rangle.$$

Mit der Wahl dieser Vektoren ist die unitäre Transformation bereits exakt bestimmt; da  $U$  linear ist, folgen mit einer einfachen Rechnung die Bilder von zum Beispiel  $|1\rangle|\phi_a\rangle$  und  $|-\rangle|\phi_a\rangle$ . Welchen der beiden möglichen Folgezustände  $|\phi_0\rangle$  und  $|\phi_+\rangle$  Eves Bit angenommen hat, ist die einzige Information, die sie bei dem Lauschversuch gewonnen hat. Aus dieser allein muss ent-



schieden werden, welchen Zustand  $|x\rangle$  hat. Da unitäre Transformationen winkelerhaltend sind, gilt<sup>1</sup>

$$\langle 0|+\rangle\langle\phi_a|\phi_a\rangle = \langle 0|+\rangle\langle\phi_0|\phi_+\rangle.$$

Da  $|0\rangle$  und  $|+\rangle$  nicht orthogonal sind, können wir durch  $\langle 0|+\rangle$  teilen; es folgt

$$\langle\phi_0|\phi_+\rangle = 1.$$

Das heißt (siehe Aufgabe A.8 im Anhang):

$$|\phi_0\rangle = |\phi_+\rangle.$$

Mit der  $|x\rangle$  unverändert lassenden Transformation  $U$  kann Eve nicht entscheiden, ob  $|x\rangle$  den Wert  $|0\rangle$  oder  $|+\rangle$  hat. Daraus folgt: Wenn Eve in dieser Methode das Risiko entdeckt zu werden, ganz ausschließen will, erfährt sie nichts über  $|x\rangle$ .

2. Eve beschließt, eine gewisse Störung in Kauf zu nehmen. Sie verwendet eine unitäre Transformation  $U'$ , die Bit  $|x\rangle$  verändert, wenn auch nach Möglichkeit nicht zu sehr. Ist dieses im Zustand  $|0\rangle$ , so ist das Ergebnis  $|\tilde{0}\rangle$ , aus dem Zustand  $|+\rangle$  wird  $|\tilde{+}\rangle$ .

$$|0\rangle|\phi_a\rangle \xrightarrow{U'} |\tilde{0}\rangle|\phi_0\rangle \text{ und } |+\rangle|\phi_a\rangle \xrightarrow{U'} |\tilde{+}\rangle|\phi_+\rangle.$$

Es folgt

$$\langle 0|+\rangle = \langle\tilde{0}|\tilde{+}\rangle\langle\phi_0|\phi_+\rangle.$$

Die Zustände  $|\phi_0\rangle$  und  $|\phi_+\rangle$  lassen sich umso besser unterscheiden, je kleiner  $\langle\phi_0|\phi_+\rangle$  ist. Also müsste Eve die Transformation  $U'$  entsprechend wählen. Von dieser Wahl unabhängig ist jedoch das Skalarprodukt  $\langle 0|+\rangle = 1/\sqrt{2}$ . Dadurch wird  $\langle\tilde{0}|\tilde{+}\rangle$  in gleichem Maße größer, wie  $\langle\phi_0|\phi_+\rangle$  kleiner wird. Der Wert  $\langle\tilde{0}|\tilde{+}\rangle$  ist ein Maß für die Störung des bei Bob ankommenden Bits. Das heißt:

Je mehr Information Eve erhält, desto stärker wird  $|x\rangle$  verändert.

Unsere Argumentation ähnelt der in den Abschnitten 3.4 und 3.6. Allerdings schöpft Eve mit der beschriebenen Strategie ihre Möglichkeiten wie erwähnt nicht aus. Deutlich wird ihr Zielkonflikt zwischen *viel Information* erhalten und *unbemerkt* bleiben. Diesen kann sie mit keiner der eben beschriebenen Strategien umgehen. Wenn Eve unbemerkt bleiben will, kann sie nur wenig über das übermittelte Quantenbit  $|x\rangle$  in Erfahrung bringen. Wenn sie über dieses viel erfährt, wird sie aller Wahrscheinlichkeit nach ertappt.

Eves Zielkonflikt

### Eine Lauschstrategie mit verschränkten Bits

Wir können diesen Ansatz noch verallgemeinern, indem wir Eve erlauben, ihr Bit mit dem versendeten zu verschränken. Einen speziellen Fall dieser Idee

<sup>1</sup>Die linke Seite der Gleichung besteht aus dem Skalarprodukt der beiden Ausgangszustände  $|0\rangle|\phi_a\rangle$  und  $|+\rangle|\phi_a\rangle$ , die rechte aus dem der beiden Bilder unter der Transformation  $U$ , siehe auch Seite 43.

haben wir bereits im Beispiel 7.2 kennen gelernt. Eve verwendet ein Quantenregister  $|E\rangle$  im Anfangszustand  $|\phi_a\rangle$  und wendet eine unitäre Transformation  $V$  auf  $|x\rangle|E\rangle$  an. Dadurch hat sie die Möglichkeit ihr Quantenregister mit dem übermittelten Bit zu verschränken. Mit der Messung ihres Quantenregisters kann sie warten, bis Alice und Bob die verwendeten Basen öffentlich ausgetauscht haben. Die Nachfolgezustände von Eves Transformation  $V$  haben die Form

$$|0\rangle|\phi_a\rangle \xrightarrow{V} |0\rangle|\phi_{00}\rangle + |1\rangle|\phi_{01}\rangle$$

und

$$|1\rangle|\phi_a\rangle \xrightarrow{V} |0\rangle|\phi_{10}\rangle + |1\rangle|\phi_{11}\rangle.$$

Eves Anteile  $|\phi_{00}\rangle, |\phi_{01}\rangle$  etc. sind in diesen Formeln keine normalisierten Zustände: Gilt beispielsweise

$$|0\rangle|\phi_a\rangle \xrightarrow{V} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle),$$

so wäre  $|\phi_{00}\rangle = 1/\sqrt{2} \cdot |0\rangle$  und  $|\phi_{01}\rangle = 1/\sqrt{2} \cdot |1\rangle$ . Aus der Forderung, dass Eves Zugriff so unauffällig wie möglich sein soll, folgt sofort dass  $\langle\phi_{01}|\phi_{01}\rangle$  und  $\langle\phi_{10}|\phi_{10}\rangle$  so klein wie möglich zu sein haben.

Das gleiche gilt, wenn Alice das Bit in der Hadamardbasis kodiert hat. Wir setzen an:

$$|+\rangle|\phi_a\rangle \xrightarrow{V} |+\rangle|\phi_{++}\rangle + |-\rangle|\phi_{+-}\rangle$$

und

$$|-\rangle|\phi_a\rangle \xrightarrow{V} |+\rangle|\phi_{-+}\rangle + |-\rangle|\phi_{--}\rangle,$$

wobei  $|\phi_{++}\rangle, |\phi_{+-}\rangle, \dots$  vier weitere nichtnormalisierte Zustände sind. Wenden wir die Linearität von  $V$  an, können wir diese als Linearkombination von  $|\phi_{00}\rangle$  und  $|\phi_{10}\rangle$  darstellen. Damit lässt sich der Wunsch  $\langle\phi_{+-}|\phi_{+-}\rangle$  und  $\langle\phi_{-+}|\phi_{-+}\rangle$  klein zu halten, als weitere Bedingung an  $|\phi_{00}\rangle, |\phi_{10}\rangle$  etc. formulieren.

Auf der anderen Seite will Alice natürlich möglichst viel über das gesendete Bit herausfinden. In einer Reihe von Veröffentlichungen ist dieser Ansatz intensiv untersucht worden (eine Übersicht findet der Leser in [24]). Eve hat demnach denselben Zielkonflikt, den wir bei den Angriffen ohne Verschränkung beobachtet haben. In dem Maß, in dem Eve Informationen über den Schlüssel gewinnt, verändert sie die übermittelten Bits. Werden Fehlerkorrektur und privacy amplification verwendet, bietet das BB84-Protokoll gegen derartige Angriffe hinreichende Sicherheit. Ist die Anzahl der ausgetauschten Bits hinreichend groß, ist die Anzahl der für den Erhalt der Sicherheit aufzuwendenden Bits tolerierbar.

Die allgemeinste mögliche Lauschsituation würde nur voraussetzen, dass Eve mit den ausgetauschten Quantenbits in Kontakt tritt und dass die Gesetze der Quantenmechanik gelten. Die bisher beschriebenen Angriffe sind noch nicht so generell modelliert. Sie werden als *inkohärente* oder *unzusammenhängende* Angriffe bezeichnet, da Eve die von Alice an Bob geschickten Quantenbits nacheinander, jedes für sich analysiert. Man kann nun noch allgemeinere Szenarien erdenken, in denen Eve die ausgetauschten Quantenbits als ein System auffassen kann und Messungen vornehmen darf,

die allgemeiner als die in diesem Buch präsentierten sind (siehe zum Beispiel [107]). Nach unseren bisherigen Überlegungen dürfen wir zu Recht vermuten, dass daraus kein Vorteil entsteht.

Einen Angriff gibt es, dem das BB84-Protokoll zunächst schutzlos ausgeliefert ist. Vielleicht ist der eine oder andere Leser selbst auf die Idee gekommen: Wenn Eve sich Alice gegenüber als Bob ausgibt und alle seine Schritte simuliert, wenn Eve außerdem in gleicher Weise Bob gegenüber Alice' Rolle im BB84-Protokoll übernimmt, kann sie natürlich nicht entdeckt werden. Allerdings benötigt Eve dazu eine vollkommene Kontrolle des Geschehens. Und glücklicherweise gibt es klassische Verfahren, die vor einem solchen *man-in-the-middle attack* schützen: das sind sogenannte *Authentisierungsverfahren*. Ein Stichwort in diesem Zusammenhang lautet *digitale Unterschrift*.

Man-in-the-middle  
attack

## 7.4 Quantenverschlüsselung mit Verschränkung

Die bisher beschriebene BB84-Verschlüsselung beruht darauf, dass Quantenzustände mit nicht-orthogonalen Zuständen verändert werden, möchte man etwas über sie herausfinden. 1991 hat Artur Ekert von der Universität Oxford ein Verfahren vorgeschlagen, das verschränkte Quantenbits nutzt. Diese nur zweieinhalb Seiten lange Publikation [53] zählt mittlerweile zu den meist zitierten Veröffentlichungen zur Kryptographie.

Wir erinnern uns an das Gedankenexperiment aus Abschnitt 2.11. Alice erzeugt ein Paar verschränkter Quantenbits

Vorüberlegung

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

Eines davon schickt sie Bob. Messen beide ihre Hälften des verschränkten Paares, erhalten sie dasselbe Ergebnis. Mit Wahrscheinlichkeit  $1/2$  ist dieses 0, mit der gleichen Wahrscheinlichkeit 1. Nach der Messung besitzen beide das gleiche, zufällig erzeugte Bit, ohne dass dieses im gewohnten Sinn ausgetauscht werden musste. Zum Zeitpunkt der Übertragung stand das Ergebnis der Messungen allerdings noch nicht fest. Das Zufallsbit ist in diesem Sinne nie übertragen worden und konnte also nicht belauscht werden.

Damit scheinen verschränkte Bits für die Erzeugung von Schlüsseln gut geeignet zu sein. Nach den Erfahrungen, die wir in den vorhergehenden Abschnitten mit Eves Raffinesse gemacht haben, erscheint uns das jedoch als zu einfach. Bitte denken Sie vor dem Weiterlesen eine Weile über die folgende Aufgabe nach:

**Aufgabe 7.4:** Wie sieht Eves Angriff aus?

Eine mögliche Antwort: Eve versetzt ein Quantenbit  $|e\rangle$  in den Anfangszustand  $|0\rangle$  und wendet das CNOT-Gatter auf dieses und das von Alice an Bob

gesandte Gatter an. Danach befinden sich die drei Bits in dem verschränkten Zustand

$$\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle).$$

In welcher Reihenfolge auch immer die drei ihre Bits messen: das Ergebnis ist das gleiche, Eve erfährt auf diese Weise unbemerkt den genauen Schlüssel.

Verschiedene  
Basen

Allerdings haben wir im Beispiel 7.2 gesehen, dass Eves verschränktes Quantenbit das Ergebnis von Messungen beeinflusst, wenn die möglichen Zustände nicht orthogonal sind. Das nutzt Ekerts Schlüsselverteilungsprotokoll. Alice und Bob würfeln darin aus, in welcher Basis sie messen. Nur wenn die Ergebnisse der Messung eine bestimmte Bedingung erfüllen, verlief die Schlüsselverteilung unbelauscht, das heißt ohne Verschränkung durch weitere Bits.

Ekerts Verfahren verwendet Konzepte und Notation, die in diesem Buch nicht eingeführt werden. Darum beschreiben wir nur die Grundidee. Wir gleichen diesen Mangel dadurch aus, dass wir gegen Ende des Abschnitts die auf den ersten Blick erstaunlichen Ähnlichkeiten mit dem Ansatz von Bennett und Brassard diskutieren. Dadurch wird die Sicherheit von Verfahren nach der Art des Ekert-Protokolls erhellt.

### Das Schema des Ekert-Protokolls

Alice und Bob steht jeweils eine Auswahl verschiedener Messverfahren zur Verfügung. Folgende Schritte „übertragen“ ein Bit und sind entsprechend der gewünschten Schlüssellänge zu wiederholen.

1. Erzeuge ein Paar von Quantenbits im Zustand

$$\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).$$

Ein Bit erhält Alice, das andere Bob.

2. Alice wählt ein Messverfahren aus ihrer Menge zufällig und misst ihr Bit.
3. Bob wählt ein Messverfahren aus seiner Menge zufällig und misst sein Bit.
4. Alice und Bob tauschen sich über das gewählte Messverfahren aus:
  - 4.1. haben sie das gleiche Verfahren gewählt, so wird das Ergebnis Teil des Schlüssels;
  - 4.2. haben sie verschiedene Verfahren gewählt, dienen ihre Ergebnisse dazu, eine Bedingung  $C$  zu testen. Ist diese Bedingung  $C$  nicht erfüllt, wird der gesamte bisher übertragene Schlüssel verworfen.

Die in 4.2 zu überprüfende Bedingung  $C$  ergibt sich aus der CHSH-Ungleichung, die experimentell zugängliche Variante von Bells Ungleichung aus Abschnitt 2.11. Wir erinnern uns: John Bell fand ein Kriterium, anhand dessen sich entscheiden lässt, ob die von der Quantenmechanik vorhergesagte Fernwirkung verschränkter Teilchen korrekt ist, oder ob nicht doch eine lokal-realistische Erklärung hinter allem schlummert (Stichwort: versteckte Variablen). Damit lässt sich feststellen, ob die Quantenbits von Alice und

Bob im Sinne der Quantenmechanik maximal verschränkt sind, oder ob Eves Dazwischenfunken diese Eigenschaft vermindert hat. Die Auswahl der Messverfahren ist an die CHSH-Ungleichung angepasst.

Für die Sicherheit des Verfahrens ist es nicht wichtig, dass Alice das verschränkte Bitpaar erzeugt. Das kann irgendeine dritte Dienstleistungsinstanz sein. Die Messungen decken auf, ob jemand versucht hat, an dem Schlüsselaustausch teilzuhaben.

Wir untersuchen nun, wie Lauschversuche von Eve prinzipiell aufgedeckt werden können. Dabei hilft uns unsere Analyse des BB84-Protokolls. In dem Maß, in dem Eve die versendeten Quantenbits durch Messungen oder unitäre Transformationen manipuliert, weichen auch beim Ekert-Protokoll die Messergebnisse von Alice und Bob vom Erwarteten ab. Wenn Eve hingegen ein Quantenbit in ihrem Besitz mit einem ausgetauschten verschränkt, ändert sie nur auf den ersten Blick nichts. Beispiel 7.2 zeigt, wie ein solches Vorgehen vom BB84-Protokoll aufgedeckt wird. Anschaulich vermindert Eves verschränktes Quantenbit die ursprüngliche Verschränkung der Bits von Alice und Bob. Die Information, die sich ursprünglich in der Korrelation zwischen zwei Bits befand (siehe Abschnitt 5.3), ist nun auf drei Bits verteilt. Der Test der CHSH-Ungleichung deckt diese verminderte Verschränkung auf. An dieser Stelle wird deutlich, warum auf eine detaillierte Darstellung des Ekert-Protokolls verzichtet wird: es zu verstehen, hängt vom Verständnis der genannten Ungleichung ab. Wir gehen hier einen anderen Weg. Wir untersuchen Gemeinsamkeiten mit dem uns mittlerweile vertrauten BB84-Protokoll, um das Geschehen mit uns bekannten Begriffen anzunähern und auf diese Weise ein Gefühl für auf Verschränkung beruhende Kryptographieverfahren zu gewinnen.

Eves Einfluss

Die Arbeit von Bennett und Brassard aus dem Jahr 1984 hat anfangs kaum Verbreitung unter Physikern gefunden, sondern war nur Kryptologen und Informatikern bekannt. Somit entwickelte Ekert sein Verfahren ohne Kenntnis des BB84-Protokolls. Dabei gibt es zwischen beiden eine überraschende Gemeinsamkeit, aber auch einen wichtigen Unterschied. Beides wollen wir hier diskutieren.

Zusammenhang mit BB84

Wir beginnen mit der Gemeinsamkeit. Dazu betrachten wir eine Variante des Ekert-Protokolls, die ebenfalls Verschränkung nutzt.

### Eine Verschränkung nutzende Variante

1. Erzeuge ein Paar von Quantenbits im Zustand  $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ . Eines erhält Alice, das andere Bob.
2. Alice misst ihr Bit in einer der zufällig gewählten Basen  $B = \{|0\rangle, |1\rangle\}$  oder  $B' = \{|+\rangle, |-\rangle\}$ .
3. Bob misst sein Bit in einer der zufällig gewählten Basen  $B$  oder  $B'$ .
4. Alice und Bob tauschen sich über das gewählte Messverfahren aus:

- 4.1. haben sie in der gleichen Basis gemessen, ist das Ergebnis Teil des Schlüssels;

- 4.2. haben sie verschiedene Basen gewählt, vergleichen sie ihre Bits. Sind diese ungleich, wird der bisher übertragene Schlüssel verworfen.

Die Gemeinsamkeiten mit dem BB84-Protokoll lassen sich leicht nachvollziehen, wenn wir dessen Darstellung auf Seite 178 betrachten. In Schritt 2 unserer „Ekert-Variante“ erzeugt Alice implizit ein Zufallsbit, das im BB84-Protokoll  $a'$  heißt. Das Messergebnis entspricht dem Bit  $a$  des BB84-Protokolls. Die weiteren Gemeinsamkeiten der Verfahren sollte der Leser an dieser Stelle selbst nachvollziehen. Auf diese Weise nimmt das hier nur ange-deutete Ekert-Protokoll etwas konkretere Konturen an. Da wir die Sicherheit des BB84-Protokolls ausführlich diskutiert haben, bekommen wir außerdem einen Hinweis darauf, warum das Ekert-Protokoll sicher ist.

Unterschied zu  
BB84

Hier soll nun aber ein wichtiger Unterschied erwähnt werden. Im Fall des BB84-Protokolls existiert Alice' Schlüsselbit Bit  $a$ , bevor das davon abhängende Quantenbit verschickt wird. Bei dem auf Verschränkung beruhenden Protokoll ist es theoretisch möglich, die verschränkten Quantenbits zu verschicken; Alice und Bob messen diese erst dann, wenn die Schlüssel benötigt werden. Falls zwischen der Schlüsselverteilung – also dem Zeitpunkt, zu dem zwischen den beiden ein Quantenkanal existiert – und der Verwendung ein längerer Zeitraum liegt, hat das Ekert-Protokoll einen Vorteil: die Schlüsselbits können unmittelbar vor Gebrauch erzeugt werden. Beim BB84-Protokoll müssen die Schlüssel zwischen Verteilung und Verwendung in klassischer Form sicher verwahrt werden. Zwar hindern die Gesetze der Physik Eve daran, durch Belauschen des Quantenbits relevante Informationen zu erlangen, sofern Alice und Bob das Protokoll korrekt einhalten. Verwahren Alice und Bob die erzeugten, dann klassischen, Schlüssel anschließend in Panzerschränken, um sie zu einem späteren Zeitpunkt zu verwenden, spricht gemäß der klassischen Physik nichts dagegen, dass Eve einen davon knackt, die Bits kopiert und schließlich alles in demselben Zustand wie vor ihrem Zugriff hinterlässt – mag das in der Praxis auch mit Schwierigkeiten verbunden sein.

**Aufgabe 7.5:** Was ist an folgender Argumentation falsch, die abweichend vom vorhergehenden Absatz behauptet, dass die eben beschriebenen Protokolle doch gleichwertig seien: „Im BB84-Protokoll könnte Bob die ihm zugesandten Quantenbits speichern und erst dann messen, wenn die Schlüssel benötigt werden. Da seine Wahl der Basis den entstehenden Schlüssel mitbestimmt, wird dieser auch erst nach Versendung der Quantenbits erzeugt. Aus Eves Sicht gibt es keinen Unterschied zwischen diesem Vorgehen und dem Protokoll auf Seite 187.“

Ausblick

Wir haben hier zwei historisch frühe, aber gleichzeitig für Anwendungen und das Grundverständnis wichtige Beispiele kryptographischer Verfahren auf Quantenbasis kennen gelernt<sup>2</sup>. Wie in allen Bereichen der Quanteninformatik gibt es auch hier eine sehr große Zahl an weiteren Ansätzen, Ergebnissen

<sup>2</sup>Der erste Ansatz zur Quantenkryptographie stammt bereits aus dem Jahr 1969. Stephen Wiesner entwickelte einen Ansatz für fälschungssichere Banknoten, der allerdings erst 1983 zur Publikation angenommen wurde, [149].

und Ideen. Quantenkryptographie hat sich zu einem sehr umfangreichen Forschungsgebiet entwickelt.

Bereits in der Einleitung wurde erwähnt, dass quantenkryptographische Systeme schon seit längerem käuflich zu erwerben sind. Abbildung 7.2 verdeutlicht, dass für die Umsetzung des BB84-Protokolls sehr einfache unitäre Transformationen genügen. In Abschnitt 10.2.3 wird beispielhaft eine frühe Umsetzung beschrieben: Im Oktober 2002 gelang Harald Weinfurter und seinem Team von der LMU München mittels kleiner und preiswerter Geräte eine Schlüsselübertragung zwischen der Zug- und der Karwendelspitze über 23,4 Kilometer. Bei solchen Freiraumverfahren werden die Photonen direkt durch die Luft gesendet.

Es gab bereits erfolgreiche Angriffe auf Umsetzungen des BB84-Protokolls. Das ist kein Widerspruch zu den Ergebnissen unserer Untersuchung der Sicherheit des BB84-Protokolls. Der theoretische Aufbau geht selbstverständlich von Voraussetzungen aus, die in realen Umsetzungen nicht immer gegeben sind. Ein sehr einfaches Beispiel: wir gehen selbstverständlich davon aus, dass Alice *genau ein* Quantenbit an Bob schickt. Als Quantenbits werden in den Umsetzungen in der Regel Photonen verwendet, siehe Abschnitt 10.2. In frühen Umsetzungen kamen als Photonenquellen stark abgeschwächte Laser zum Einsatz, die meistens kein, häufig genau ein und manchmal mehrere Photonen emittieren. Verschickt nun Alice unbeabsichtigt mehrere identische Quantenbits, so kann Eve eines davon abfangen, ohne bemerkt zu werden. Deshalb ist für die Quantenkryptographie die Entwicklung verlässlicher *Einzelphotonenquellen* von großer Bedeutung. Einen vielversprechenden Ansatz dazu bieten die sogenannten *Quantenpunkte*, siehe Abschnitt 10.2.3. Dort finden Sie auch die Verweise auf einige erfolgreiche Angriffe.

Zu weiteren Angriffen auf das BB84-Protokoll gibt es Hinweise im Abschnitt 10.2. Zum Einstieg in ein vertiefendes Studium der Quantenkryptographie ist noch immer [67] empfehlenswert, wo auch die Sicherheit der thematisierten Verfahren genauer analysiert wird, als es in diesem Kapitel möglich ist. In diesem Zusammenhang sei wir auch auf das Ende von Abschnitt 5.1 verwiesen, wo sich Bemerkungen zum Thema Quanteninternet finden.

# 8 Klassische Verschlüsselungen knacken: Primfaktorzerlegung

*O Sicherheit, der Teufel wartet deiner!*  
Jakob Böhme

In diesem Kapitel lernen wir den berühmtesten Quantenalgorithmus kennen: Shors Verfahren zur Faktorisierung ganzer Zahlen. Bei der Veröffentlichung 1994 handelte es sich um den ersten effizienten Quantenalgorithmus für ein Problem, für das kein effizientes klassisches Verfahren bekannt ist und das zugleich wichtig ist. Mit Shors Algorithmus lässt sich zu einer ganzen Zahl ein Teiler finden. Das soll wichtig sein? Um das einzusehen, beginnen wir unsere Untersuchung mit einem bekannten Verschlüsselungsverfahren, der RSA-Kryptographie.

In diesem Kapitel spielen Begriffe und Ergebnisse aus der Zahlentheorie eine Rolle. Diese sind in Abschnitt A.5 des Anhangs ausgelagert, um den Lesefluss nicht zu stören. In Abschnitt 8.2 werden wir feststellen, dass wir Zahlen dann faktorisieren können, wenn wir die Periode einer Funktion bestimmen können. Shors Algorithmus ermöglicht es, solche Perioden effizient zu berechnen. Zur Vorbereitung auf diesen Quantenalgorithmus lernen wir in Abschnitt 8.3 ein klassisches Verfahren von großer praktischer Relevanz kennen: die diskrete Fouriertransformation. Denn ein wichtiges Element von Shors Faktorisierungsalgorithmus ist die Quanten-Fouriertransformation, siehe Abschnitt 8.4.

Abschnitt 8.5 untersucht *Simons Algorithmus*, dessen Aufgabenstellung an das Verfahren von Deutsch-Jozsa erinnern mag. Er berechnet die Periode einer Funktion mit Booleschen Eingabewerten (also aus der Menge  $\{0, 1\}$ ) und ist strukturell mit dem Kern von Shors Algorithmus identisch. Nach diesen Vorarbeiten ist es nicht mehr schwierig, die Arbeitsweise von Shors Algorithmus zu verstehen. Im letzten Abschnitt werden wichtige Varianten und Verallgemeinerungen von Shors Faktorisierungsalgorithmus vorgestellt.



## 8.1 Faktorisierung und Verschlüsselung: RSA-Kryptographie

Definition  
Faktorisierung

Finden wir zu einer Zahl  $n$  einen Teiler  $a$ , können wir  $n$  als Produkt ganzer Zahlen schreiben:

$$n = a \cdot b, \text{ mit } b = n/a.$$

Die Faktoren  $a$  und  $b$  können wir weiter zerlegen, bis ein Produkt von Primzahlen entstanden ist. Zum Beispiel gilt

$$100 = 10 \cdot 10 = 2 \cdot 2 \cdot 5 \cdot 5.$$

Das ist die sogenannte *Primfaktorzerlegung*. Jede ganze Zahl ist auf diese Weise als Produkt nicht weiter zerlegbarer Zahlen darstellbar. Dieses Produkt ist eindeutig (lässt man die Reihenfolge der Faktoren außer acht).

Verfahren zur Berechnung der Primfaktorzerlegung einer Zahl  $n$  sind seit langem bekannt. So können wir wie oben beschrieben zunächst einen Teiler bestimmen und das entstehende Produkt gegebenenfalls weiter zerlegen. Allerdings könnte jede Zahl zwischen 2 und  $\sqrt{n}$  ein echter Teiler von  $n$  sein. Die Eingabegröße des Faktorisierungsproblems ist die Anzahl der Bits von  $n$ , die wir mit  $m$  bezeichnen. Wir müssten  $\Omega(\sqrt{2^m})$  mögliche Teiler ausprobieren, und die Laufzeit wächst beinahe exponentiell. Auch der beste bekannte Algorithmus hat Laufzeit  $\Omega(2^{\frac{1}{3}m})$ , und es wird allgemein angenommen, dass die Primfaktorzerlegung mit klassischen Rechnern nicht effizient berechnet werden kann. Man ist von dieser Annahme so überzeugt, dass auf ihr die Sicherheit eines bekannten Verschlüsselungssystems beruht: die der RSA-Kryptographie, benannt nach Ronald Rivest, Adi Shamir und Leonard Adleman, die dieses Verfahren 1978 veröffentlicht haben, [128].

Bei der RSA-Kryptographie handelt es sich um ein *Public Key-Verfahren*; man spricht auch von *asymmetrischer Verschlüsselung*. Bei solchen Verfahren gibt es zwei Schlüssel: einen *öffentlichen*, den die Absender verwenden, um Nachrichten zu verschlüsseln. Und einen geheimen, den der Adressat niemandem mitteilen darf: damit entschlüsselt er Nachrichten, die mit dem öffentlichen verschlüsselt wurden. Das Ziel des Ganzen ist: für eine unbefugte Person, diese heißt bei uns Eve, muss die verschlüsselte Nachricht unlesbar sein. Dies entspricht der Situation in Abbildung 7.1.

Public Key-  
Kryptographie

Schematisch sieht das folgendermaßen aus: Bob erwartet vertrauliche Nachrichten von eventuell mehreren Absendern.

1. Er erzeugt zwei Schlüssel: einen geheimen, den er für sich behält, und einen öffentlichen, den er zum Beispiel Alice und Charlotte zukommen lässt.
2. Alice verschlüsselt ihre Mitteilung an Bob mit dem öffentlichen Schlüssel.
3. Bob verwendet seinen geheimen Schlüssel, um Alice' Nachricht zu entschlüsseln.

Können wir sicherstellen, dass eine mit dem öffentlichen Schlüssel kodierte Nachricht ohne den geheimen Schlüssel nicht lesbar ist, hat dieses Verfahren zwei große Vorteile:

1. Es muss kein geheimer Schlüssel zwischen Alice und Bob ausgetauscht werden. Dieser Vorgang ist eine Sicherheitslücke von Verfahren mit geheimen Schlüsseln (siehe Abschnitt 7.1).
2. Viele verschiedene Absender können denselben öffentlichen Schlüssel verwenden. Sie sind alle in das Kryptographiesystem eingebunden; Alice kann aber von Charlotte verschlüsselte Nachrichten nicht lesen und umgekehrt.

Ein Beispiel für ein Public Key-Kryptographiesystem ist das zur Verschlüsselung von E-Mails dienende PGP-Verfahren (engl. *pretty good privacy*). Wir betrachten nun, wie die drei Schritte (Schlüsselerzeugung, Chiffrierung, Dechiffrierung) im Falle des RSA-Verfahrens umgesetzt sind.

### A. Bob erzeugt einen öffentlichen Schlüssel

1. Wähle zwei zufällige Primzahlen  $p \neq q$ .
2. Berechne das Produkt und dessen Eulerfunktionswert:  

$$n \leftarrow pq,$$

$$\phi(n) \leftarrow (p-1)(q-1)$$
3. Wähle eine kleine ungerade Zahl  $e$ , die teilerfremd zu  $\phi(n)$  ist.
4. Löse die Gleichung  $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$ .
5. Veröffentliche  $e$  (*encryption*) und  $n$  als *öffentlichen Schlüssel*.
6. Halte  $d$  (*decryption*) als *geheimen Schlüssel* unter strengem Verschluss, vernichte  $p, q$  und  $\phi(n)$ .

### Bemerkungen:

*Schritt 1:* Die Zahlen  $p$  und  $q$  müssen sehr groß sein. Eine Länge von 2000 Bits ist empfehlenswert. Wir wählen zwei große Zahlen  $p$  und  $q$  zufällig und testen, ob es sich um Primzahlen handelt. Dazu gibt es ein randomisiertes Verfahren (siehe Abschnitt 4.2, kürzlich wurde auch ein deterministisches Verfahren entdeckt, [3]).

*Schritt 2:* Die Eulersche  $\phi$ -Funktion ist auf Seite 308 definiert.

*Schritt 3:* Wir können  $e$  als Primzahl größer  $p$  und  $q$  wählen. Die Verschlüsselung wird effizienter, wenn  $e$  klein ist; ein beliebter Kandidat ist 65537.

*Schritt 4* lässt sich mit einer erweiterten Version von Euklids Algorithmus in Laufzeit  $O((\log n)^3)$  ausführen, siehe [40].  $d$  ist das multiplikative Inverse von  $e$  modulo  $\phi(n)$ .

*Schritt 6:* Die Werte  $p, q$  und  $\phi(n)$  können aus  $e, d$  und  $n$  effizient rekonstruiert werden. Werden auch die Zahlen  $p, q$  als Teil des geheimen Schlüssels gespeichert, wird eine beschleunigte Entschlüsselung möglich. Wir beschränken uns in der Folge auf die ursprüngliche Grundvariante des Verfahrens.

Bob hat einen öffentlichen Schlüssel erzeugt und zum Beispiel auf seiner Homepage veröffentlicht. Alice hat sich die Zahlen  $e$  und  $n$  heruntergeladen. Bevor sie ihre Nachricht verschlüsseln kann, muss sie diese in eine ganze Zahl umwandeln. Ist diese als Folge von Bits  $b_{n-1} \dots b_0$  gegeben, bietet sich die binär dargestellte Zahl  $m = \sum_{i=0}^{n-1} b_i 2^i$  an.

### B. Alice verschlüsselt ihre Nachricht

Eingabe beziehungsweise Ausgangspunkt ist Alice' Nachricht, die ganze Zahl  $m$ . Wir setzen voraus, dass  $m$  kleiner  $n$  ist. Anderenfalls zerlegt sie die Ziffernfolge  $m$  in Blöcke von Ziffern  $m_1, \dots, m_k$ , so dass jeder der  $k$  Zahlen kleiner  $n$  ist.

Ausgabe ist die verschlüsselte Nachricht  $\tilde{m}$ .

$$1. \tilde{m} = m^e \bmod n$$

### C. Bob entschlüsselt die Nachricht

Eingabe: die verschlüsselte Nachricht  $\tilde{m}$

Ausgabe: die entschlüsselte Nachricht  $m$

Bob benutzt seinen geheimen Schlüssel  $d$ .

$$1. m \leftarrow \tilde{m}^d \bmod n$$

**Beispiel 8.1:** Das folgende Beispiel stammt aus der Ursprungsarbeit von Rivest, Shamir und Adleman, [128].

A. Bob wählt die Primzahlen

$$p = 47, q = 59$$

und ermittelt

$$n = pq = 2773 \text{ und } \phi(n) = (p-1)(q-1) = 2668.$$

Er wählt  $e = 17$  und zeigt mit Euklids Algorithmus

$$\text{ggT}(2668, 17) = 1.$$

Mit der erweiterten Variante von Euklids Algorithmus stellt er fest, dass

$$d \cdot 17 \equiv 1 \bmod 2668$$

die Lösung  $d = 157$  hat. Bob veröffentlicht  $e = 17$  und  $n = 2773$ .

B. Alice möchte Bob die Nachricht ITS ALL GREEK TO ME schicken und verwendet den öffentlichen Schlüssel  $e = 17, n = 2773$ . Sie wählt die Kodierung

$$\text{Leerzeichen} = 00, A = 01, B = 02, \dots, Z = 26.$$

Das ergibt

$$m = 0920 \ 1900 \ 0112 \ 1200 \ 0718 \ 0505 \ 1100 \ 2015 \ 0013 \ 0500$$

Sie teilt  $m$  in 10 Blöcke der Länge 4 ein, damit jeder Block einer Zahl kleiner  $n = 2773$  entspricht.  $m_1 = 920$  verschlüsselt sie zu

$$\tilde{m}_1 = 920^{17} = 948 \bmod 2773.$$

Die verschlüsselte Nachricht lautet dann

$$\tilde{m} = 0948 \ 2342 \ 1084 \ 1444 \ 2663 \ 2390 \ 0778 \ 0774 \ 0219 \ 1655.$$

- C. Bob hat  $\tilde{m}$  empfangen. Er wendet den geheimen Schlüssel  $d = 157$  auf  $\tilde{m}_1$  an:

$$948^{157} \bmod n = 920$$

ergibt zum Beispiel die ersten beiden Buchstaben von Alice' Nachricht.

*Bemerkung:* Damit dieses Beispiel nachvollziehbar wird, wurden die Zahlen  $p, q$  und  $e$  sehr klein gewählt. So ist jedoch keine sichere Verschlüsselung möglich. Zudem erlaubt die Einteilung in Blöcke aus zwei Zeichen eine einfache Häufigkeitsanalyse. In der Praxis wird RSA häufig zusammen mit anderen Verfahren verwendet, zum Beispiel zum Austausch des geheimen Schlüssels eines anderen Protokolls.

### Analyse

Die Analyse besteht aus drei Teilen. Zunächst stellen wir fest, dass Bob die Nachricht korrekt entschlüsselt. Danach betrachten wir sehr knapp die Laufzeit dieses Verfahrens. Und zuletzt untersuchen wir die Sicherheit: kann wirklich nur Bob die Nachricht von Alice entschlüsseln?

Bob hat die Nachricht korrekt entschlüsselt.  $d$  ist  $e^{-1} \bmod (p-1)(q-1)$ , das multiplikativ Inverse von  $e$  modulo  $\phi(n)$ . Das heißt

Korrekte  
Entschlüsselung

$$ed \equiv 1 \bmod (p-1)(q-1)$$

oder

$$ed = 1 + k(p-1)(q-1)$$

für eine ganze Zahl  $k$ . Wir können ausnutzen, dass  $p$  und  $q$  Primzahlen sind:

$$\begin{aligned} \tilde{m}^d &= m^{ed} \\ &= m^{1+k(p-1)(q-1)} \\ &= m \cdot (m^{p-1})^{k(q-1)} \\ &\equiv m \cdot 1^{k(q-1)} \bmod p \\ &\equiv m \bmod p \end{aligned}$$

Im vorletzten Schritt haben wir für den Fall  $m \not\equiv 0 \bmod p$  den kleinen Satz von Fermat angewendet, siehe Seite 308. Falls  $m \equiv 0 \bmod p$  ist die Aussage ohnehin klar. Mit der gleichen Rechnung folgt

$$\tilde{m}^d \equiv m \bmod q.$$

An dieser Stelle verwenden wir eine aus der Zahlentheorie bekannte Tatsache, die direkt aus dem *chinesischen Restsatz* folgt (siehe etwa [40]):

Sind  $k$  und  $l$  teilerfremd, so folgt für alle ganzen Zahlen  $x$  und  $a$ :

$$x \equiv a \pmod{k} \text{ und } x \equiv a \pmod{l}$$

genau dann, wenn

$$x \equiv a \pmod{k \cdot l}.$$

Damit folgt

$$\tilde{m}^d \equiv m \pmod{n}.$$

Laufzeit

Wir überzeugen uns rasch davon, dass alle drei Schritte effizient ausführbar sind. Mehrfach wird der Euklidische Algorithmus angewendet, der in der Eingabegröße linear viele arithmetische Operationen ausführt, die ihrerseits quadratische Laufzeit haben. Den Primzahltest haben wir bereits oben erwähnt. Wir müssen uns nur noch davon überzeugen, dass wir in Polynomialzeit potenzieren können. Wir dürfen etwa

$$m^e \pmod{n}$$

nicht durch  $e$  Multiplikationen berechnen: das wären exponentiell viele in der Eingabegröße.

Effiziente

Potenzierung

Es geht schneller, wenn wir die Binärdarstellung des Exponenten verwenden. So ist  $100 = 2^6 + 2^5 + 2^2$ . Damit gilt

$$m^{100} = m^{2^6} m^{2^5} m^{2^2}.$$

Für die Faktoren gilt  $m^{2^i} = m^{2^{i-1}} m^{2^{i-1}}$ . Damit lassen sich die potentiellen Faktoren  $m^{2^0}, m^{2^1}, \dots, m^{2^k}$  mit  $k = O(\log e)$  Multiplikationen bestimmen. Daraus lässt sich mit  $O(\log e)$  weiteren Multiplikationen  $m^e$  bestimmen. Es genügen also  $O(\log e)$  Multiplikationen; diese sind mit  $O((\log m)^2)$  Bitoperationen ausführbar. Mit dieser Idee lässt sich  $m^e \pmod{n}$  mit kubischer Laufzeit berechnen:

Die Potenz  $m^e \pmod{n}$  ist für  $L$ -Bitzahlen mit Laufzeit  $O(L^3)$  berechenbar.

1.  $\text{int } x \leftarrow 1$
2. solange  $e \neq 0$ 
  - 2.1. falls  $e$  ungerade, so  $x \leftarrow x \cdot m \pmod{n}$
  - 2.2.  $e \leftarrow e/2$
  - 2.3.  $m \leftarrow m \cdot m \pmod{n}$
3. return  $x$

Selbstverständlich hat auch die unbefugte Person Eve den Schlüssel  $e, n$  heruntergeladen, und Alice' verschlüsselte Botschaft  $\tilde{m}$  abzufangen war auch kein Problem. Warum kann sie die Klartextbotschaft  $m$  nicht bestimmen? Um den geheimen Schlüssel  $d$  herauszufinden, müsste sie die Gleichung

$$ed \equiv 1 \pmod{(p-1)(q-1)}$$

lösen. Offenbar benötigt sie dazu das Produkt  $(p-1)(q-1) = \phi(n)$ .

Ihre erste Idee ist, die Zahl  $n$  zu faktorisieren. Aber wir haben bereits erfahren, dass Eve dazu nicht in der Lage ist, wenn  $n$  groß ist. Und Bob hat dafür gesorgt, dass  $n$  *sehr* groß ist.

Die Folgefrage lautet: Kann Eve  $\phi(n)$  berechnen, ohne  $n$  zu faktorisieren? Das ist im Prinzip nicht ausgeschlossen:  $\phi(n)$  ist die Größe der Menge  $\mathbb{Z}_n^*$ . Könnte dieser Wert ohne Kenntnis von  $p$  und  $q$  bestimmt werden? Nein: jeder der  $\phi(n)$  kennt, kann daraus  $p$  und  $q$  effizient berechnen. Wir erhalten  $\phi(n) = (p-1)(q-1) = n - (p+q) + 1$  durch Ausmultiplizieren. Die Kenntnis von  $\phi(n)$  und  $n$  liefert also  $p+q$ . Damit lässt sich  $p-q$  ermitteln:

$$(p-q)^2 = p^2 - 2pq + q^2 = (p+q)^2 - 4pq.$$

Es sei daran erinnert, dass Eve  $pq = n$  kennt. Jetzt ergibt sich der erste Faktor durch  $(p+q) - (p-q) = 2q$ .

Allerdings folgt daraus noch nicht, dass Eve  $n$  faktorisieren muss, um  $\tilde{m}$  zu entschlüsseln. Es wurde jedoch schon seit Jahrzehnten über die Sicherheit der RSA-Kryptographie nachgedacht, und man hat keinen effizienten, allgemeinen Angriff gefunden.

Wir schließen:

1. Bob kann mit seinem geheimen Schlüssel die Nachricht dechiffrieren, weil er  $p$  und  $q$  kennt, die Primfaktoren von  $n$ .
2. Eve kann diese Nachricht (nach dem aktuellen Wissensstand) nur dann effizient entschlüsseln, wenn sie  $p$  und  $q$  kennt.

Darauf beruht die Sicherheit der RSA-Kryptographie. Und auf der ebenfalls unbewiesenen Vermutung, dass es keinen effizienten Algorithmus zur Faktorisierung von  $n$  gibt. Eines aber lässt sich beweisen: Verfügt Eve über einen Quantencomputer, der auf ausreichend vielen Bits rechnet, kann sie  $n$  faktorisieren und Alice' Botschaft entschlüsseln.

## 8.2 Die Suche nach Perioden

In Abbildung 8.1 ist der Verlauf einer periodischen Funktion an die Tafel gezeichnet. Es gibt ein wiederkehrendes Muster, das der Wissenschaftler als Periode  $p$  identifiziert hat:

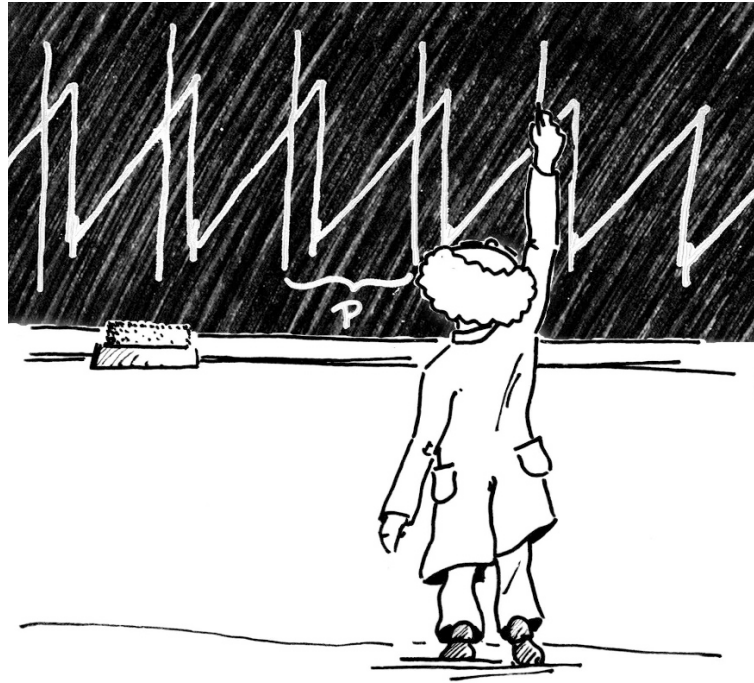


Abbildung 8.1: Die Periode einer Funktion zu finden, ist eine wichtige Aufgabe

Definition  
Periode

Die Periode einer Funktion  $f$  ist die kleinste Zahl  $p > 0$ , so dass

$$f(x+p) = f(x) \text{ für alle } x.$$

Dieser Begriff wird in Abbildung 8.1 illustriert.

### Beispiel 8.2:

1. Die Sinusfunktion hat die Periode  $2\pi$ :

$$\sin(x+2\pi) = \sin(x).$$

2. Die Funktion

$$f(x) = 2^x \bmod 5$$

bildet ganze Zahlen in die Menge  $\{0, 1, \dots, 4\}$  ab.

Es gilt  $f(0) = 1$ ,  $f(1) = 2$  und  $f(2) = 4$ . Wegen

$$2^3 = 8 \equiv 3 \bmod 5$$

ist  $f(3) = 3$ . Der nächste Funktionswert ergibt sich aus

$$2^4 = 16 \equiv 1 \bmod 5,$$

$f(4) = 1 = f(0)$ . Damit hat  $f$  Periode  $p = 4$ . Denn ab dieser Stelle gilt

$$f(4+i) = 2^4 2^i \equiv 1 \cdot 2^i \pmod{5} = f(i).$$

◇

Die Periode einer Funktion zu finden, ist nicht immer einfach. Wir werden sehen: Das am Beginn des letzten Abschnitts formulierte Faktorisierungsproblem lässt sich effizient lösen, wenn wir die Periode von bestimmten Funktionen berechnen können. Da das Faktorisierungsproblem schwierig ist, ist damit auch die Suche nach Perioden schwierig, zumindest im Allgemeinen. Wie Perioden beim Faktorisieren helfen, beschreibt der Rest des Abschnitts.

### Perioden und Faktoren

Wir haben uns vorgenommen, eine ganze Zahl  $n$  zu faktorisieren. Dazu nehmen wir uns vor, in einem Schritt einen *echten Teiler* zu bestimmen. Also eine Zahl  $t$ , die  $n$  teilt und ungleich 1 und  $n$  ist. Das ergibt eine Zerlegung

$$n = t \cdot n'.$$

Falls  $t$  oder  $n'$  keine Primzahlen sind, können wir diese Zahlen weiter zerlegen und so die Primfaktorzerlegung ermitteln. Wir werden davon ausgehen, dass  $n$  ungerade ist. Das ist keine Einschränkung: Wenn  $n$  gerade ist, lässt sich dies leicht feststellen – zum Beispiel an der Binärdarstellung – und wir kennen sofort zwei echte Teiler: 2 und  $n/2$ .

Da wir nicht alle möglichen Teiler von  $n$  durchprobieren können, analysieren wir folgenden randomisierten Ansatz (der allerdings noch nicht zum Erfolg führen wird). Für jede ganze Zahl  $a$  ist  $\text{ggT}(a, n)$  ein Teiler von  $n$ , gerade der größte gemeinsame mit  $a$ . Also könnten wir eine Zahl  $a$  zufällig aus  $\{2, \dots, n-1\}$  wählen und mit dem Euklidischen Algorithmus den Teiler  $\text{ggT}(a, n)$  bestimmen. Das Problem ist allerdings: sehr oft werden wir das Ergebnis 1 erhalten; immer dann, wenn  $a$  und  $n$  teilerfremd sind. Angenommen,  $n$  ist das Produkt zweier großer Primzahlen  $p$  und  $q$ . Dann sind fast alle Zahlen  $\{2, \dots, n-1\}$  mit  $n$  teilerfremd.

Es gibt jedoch ein Verfahren, das mit hoher Wahrscheinlichkeit eine Zahl findet, die mit  $n$  nicht teilerfremd ist. Um es effizient ausführen zu können, müssen wir jedoch folgende Fähigkeit haben: für eine Zahl  $1 < a < n$  können wir die Periode  $p$  der Funktion

Voraussetzung:  
wir können  
Perioden  
bestimmen

$$f(x) = a^x \pmod{n}$$

effizient berechnen.

Aus  $f(x+p) = f(x)$  folgt  $f(p) = f(0)$ . Für die Periode  $p$  gilt also

$$a^p \equiv 1 \pmod{n}. \quad (8.1)$$



Man nennt  $p$  auch die *Ordnung* modulo  $n$  von  $a$ .  $p$  gibt an, wie oft  $a$  mit sich selbst multipliziert werden muss, bis das Ergebnis modulo  $n$  den Wert  $a^0 = 1$  ergibt. Anders ausgedrückt: die Folge  $a^0, a^1, a^2, a^3, \dots$  ergibt  $p$  verschiedene Werte:

$$a^0, a^1, \dots, a^{p-1}, a^p = a^0, a^{p+1} = a^1, \dots$$

Aus Gleichung (8.1) folgt:

$$a^p = 1 + kn \text{ für ein } k.$$

Das heißt,  $n$  teilt  $a^p - 1$ . Was hilft uns das bei der Suche nach einem echten Teiler von  $n$ ? Wenn die Periode  $p$  gerade ist, können wir gemäß (8.1)  $a^p - 1$  faktorisieren:

$$a^p - 1 = (a^{p/2} - 1)(a^{p/2} + 1).$$

Die Zahl  $n$  teilt also das Produkt auf der rechten Seite der Gleichung. Aus einer Gleichung der Form

$$k \cdot n = (x - 1)(x + 1)$$

folgt, dass  $n$  mit  $x - 1$  oder mit  $x + 1$  einen Teiler ungleich 1 gemeinsam hat. Es können also nicht sowohl  $a^{p/2} - 1$  als auch  $a^{p/2} + 1$  mit  $n$  teilerfremd sein. Allerdings könnte  $a^{p/2} + 1$  ein Vielfaches von  $n$  sein: in diesem Fall ist  $\text{ggT}(a^{p/2} + 1, n) = n$  und liefert keinen echten Teiler.

**Aufgabe 8.1:** Zeigen Sie, dass  $a^{p/2} - 1$  kein Vielfaches von  $n$  ist.

Mit Übung 8.1 folgern wir:

$$\text{ggT}(a^{p/2} + 1, n) \text{ oder } \text{ggT}(a^{p/2} - 1, n)$$

ergeben einen echten Teiler von  $n$ , außer

- $a^{p/2} - 1$  ist mit  $n$  teilerfremd, und
- $a^{p/2} + 1$  ist ein Vielfaches von  $n$ .

Dieser Fall ist jedoch unwahrscheinlich:

**Tatsache der Zahlentheorie:** Ist  $n$  ungerade und keine Primpotenz, so gilt für mindestens die Hälfte der Zahlen  $a$  in  $\{0, \dots, n-1\}$ , die mit  $n$  teilerfremd sind:

- die Periode  $p$  der Funktion  $a^x \bmod n$  ist gerade, und
- $n$  teilt  $a^{p/2} + 1$  nicht (anders ausgedrückt:  $a^{p/2} \not\equiv -1 \bmod n$ ).

Diese Aussage werden wir nicht beweisen (siehe dazu etwa [114]). Stattdessen konstruieren wir aus unseren bisherigen Überlegungen einen Algorithmus (siehe dazu Abbildung 8.2).

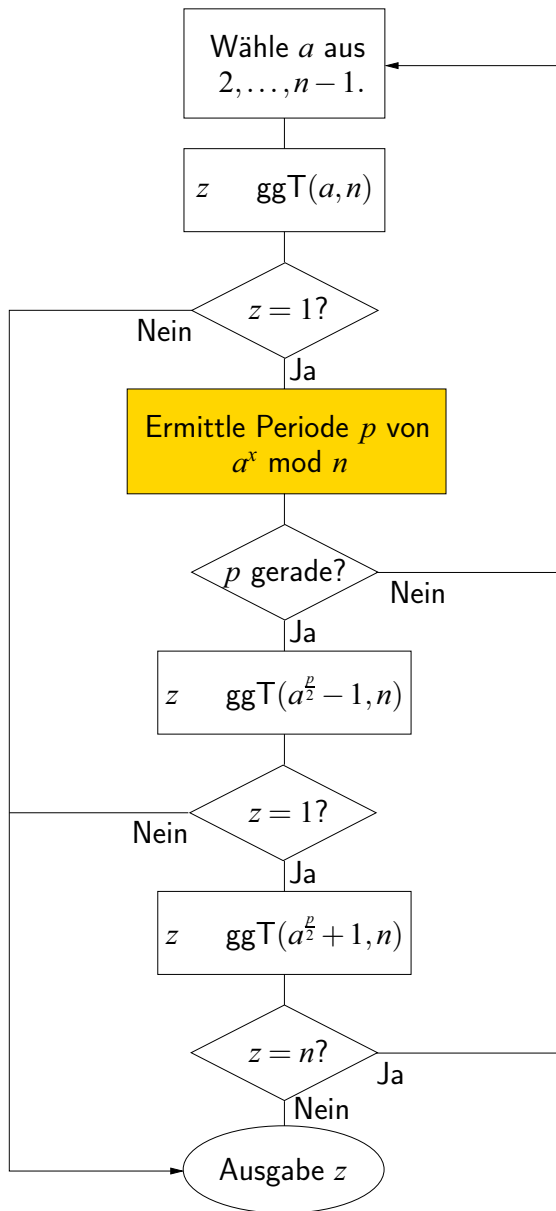


Abbildung 8.2: Flussdiagramm zu Shors Algorithmus. Die hervorgehobene Suche nach der Periode wird von einem Quantenalgorithmus ausgeführt.

**Algorithmus Faktorisierung, klassischer Teil**

Eingabe: eine ungerade ganze Zahl  $n$ , die keine Primpotenz ist  
Ausgabe: ein echter Teiler von  $n$

1. Wähle zufällig eine Zahl  $a$  aus  $\{2, \dots, n-1\}$ .

2.  $z \leftarrow \text{ggT}(a, n)$ .  
Falls  $z \neq 1$ : Ausgabe  $z$ . Abbruch.

3. Ermittle die Periode  $p$  von  $a^x \bmod n$ .

4. Falls  $p$  ungerade ist: Beginne erneut mit Schritt 1.

5. Ermittle  
 $\text{ggT}(a^{p/2} + 1, n)$  und  $\text{ggT}(a^{p/2} - 1, n)$   
  
Hat sich kein echter Teiler ergeben, beginne erneut mit Schritt 1.  
Sonst: Ausgabe  $z$ .

Haben wir im ersten Schritt die Zahl  $a$  ungünstig gewählt, lässt sich daraus kein Teiler von  $n$  ermitteln. Dann wählen wir erneut eine Zufallszahl und beginnen von vorn. Während in Schritt 2 alle  $a$  mit  $\text{ggT}(a, n) \neq 1$  sofort zum Erfolg führen, ist die Wahrscheinlichkeit, im weiteren Verlauf einen echten Teiler zu finden größer gleich  $1/2$ : das ist die Folge der obigen *Tatsache aus der Zahlentheorie*.

**Beispiel 8.3:** Die kleinste Zahl, die sich mit dieser Methode zerlegen lässt ist  $n = 15$ . Wir wählen im ersten Schritt eine zufällige Zahl  $a$  aus  $\{2, \dots, 14\}$  und berechnen mit dem Euklidischen Algorithmus  $\text{ggT}(a, 15)$ . Für  $a = 3$  erhalten wir  $\text{ggT}(3, 15) = 3$  und sind fertig: 3 ist ein echter Teiler von 15. Das gleiche gilt für  $a = 5$ .

Alle anderen möglichen Werte für  $a$  sind mit 15 teilerfremd. Nehmen wir einmal an,  $a = 7$  wurde gewählt. Die Werte der Funktion  $7^x \bmod 15$  sind gerade:

$x$	0	1	2	3	4	5	6	...
$7^x \bmod 15$	1	7	4	13	1	7	4	...

Also hat  $7^x \bmod 15$  die Periode  $p = 4$ .  $p$  ist gerade und wir ermitteln  $7^{p/2} = 49$ . Wir berechnen

$$\text{ggT}(15, 49 + 1) = 5, \text{ggT}(15, 49 - 1) = 3$$

und haben  $n = 15$  faktorisiert.

Führen wir diese Berechnung für alle  $a$  aus  $\{2, \dots, 14\}$  durch, stellen wir fest, dass wir nur mit  $a = 14$  scheitern.

$x$	0	1	2	3	4	5	6	...
$14^x \bmod 15$	1	14	1	14	1	14	1	...

In diesem Fall ist  $14^{p/2} = 14 \bmod 15$ .

$$\text{ggT}(15, 14 + 1) = 15, \text{ggT}(15, 14 - 1) = 1.$$

Es ergibt sich kein echter Teiler. ◇

Wie wahrscheinlich ist es zu scheitern? Wie hoch ist der Anteil der Elemente  $a$ , die eine ungerade Periode haben oder für die  $a^{p/2} \equiv \pm 1 \bmod n$  gilt? Wir nehmen zur Kenntnis, dass mit Ergebnissen Zahlentheorie folgende Aussage bewiesen werden kann:

Ist  $n$  ungerade und nicht die Potenz einer Primzahl ( $n \neq p^k$ , für eine Primzahl  $p$ ), so führt der klassische Teil von Shors Algorithmus mit Wahrscheinlichkeit größer  $1/2$  im ersten Anlauf zum Erfolg.

Fehlerwahrscheinlichkeit

Damit ist die Wahrscheinlichkeit, nach  $k$  Versuchen noch keine Lösung gefunden zu haben, kleiner  $1/2^k$ . Die Erfahrung zeigt, dass in der Praxis der Anteil der Werte  $a$ , für die sich keine Lösung ergibt, noch geringer ist.

Eine Einschränkung haben wir bisher nicht kommentiert:  $n$  darf keine Primpotenz sein. Diesen Fall können wir jedoch zuvor ausschließen: es gibt einen effizienten klassischen Test, der diesen Fall erkennen und auch lösen kann. Falls den Leser all die Voraussetzungen und Einschränkungen verwirren: Der in der RSA-Kryptographie verwendete Fall –  $n$  ist das Produkt zweier unterschiedlicher Primzahlen – erfüllt alle Voraussetzungen und kann als Musteranwendung betrachtet werden.

### Laufzeit

Der klassische Algorithmus dieses Abschnitts bildet den Rahmen von Shors Algorithmus. Was noch fehlt ist der Quantenalgorithmus zur Bestimmung der Periode. Um den Ressourcenverbrauch von Shors Algorithmus mit den bekannten klassischen Verfahren vergleichen zu können, analysieren wir jetzt den Aufwand des klassischen Teils.

Laufzeit

Wir beziehen uns auf den Algorithmus, der in Abbildung 8.2 visualisiert ist. Dieser rechnet ausschließlich mit Zahlen kleiner  $\log n$ , der Eingabegröße unseres Problems. Den ersten Schritt können wir als konstant voraussetzen. Der Euklidische Algorithmus – und damit Schritt 2 – hat Laufzeit  $O((\log n)^3)$ , siehe Abschnitt A.5. Der Aufwand des Quantenverfahrens im dritten Schritt wird in Abschnitt 8.6 ermittelt. Schritt 4 bedeutet konstanten Aufwand, und wir fahren mit Schritt 5 fort:  $a^{p/2}$  kann eine sehr große Zahl sein. Wir berechnen stattdessen  $a^{p/2} \bmod n$ , was mit  $O((\log n)^3)$  Bitoperationen möglich ist, wie wir auf Seite 196 gesehen haben.

**Aufgabe 8.2:** Zeigen Sie, dass für ganze Zahlen  $x, n$  gilt:

$$\text{ggT}(x, n) = \text{ggT}(x \bmod n, n).$$

Ebenso aufwändig ist anschließend Euklids Algorithmus. Also können wir Schritt 5 in Zeit  $O((\log n)^3)$  ausführen.

*Wir fassen zusammen:* Eine Ausführung des Algorithmus hat Laufzeit  $O((\log n)^3)$  und findet mit Wahrscheinlichkeit größer gleich  $1/2$  einen Teiler von  $n$ . In Abschnitt 4.2 wird untersucht, wie sich die Fehlerwahrscheinlichkeit randomisierter Algorithmen verringern lässt. Die Wahrscheinlichkeit, nach  $k$  Wiederholungen noch keinen Teiler gefunden zu haben, ist kleiner gleich  $1/2^k$ . Die erwartete Laufzeit ist also

$$\sum_{k \geq 1} \frac{k}{2^k} \cdot O((\log n)^3) = O((\log n)^3).$$

Wir wollen einen echten Teiler einer ungeraden Zahl  $n$  bestimmen, wobei  $n$  keine Primpotenz ist. Steht uns die Periode von  $a^x \bmod n$  zur Verfügung, erledigen wir diese Aufgabe in Zeit  $O((\log n)^3)$  mit Wahrscheinlichkeit größer  $1/2$ .

Das ist allerdings nur das Vorspiel zu dem eigentlichen Ergebnis dieses Kapitels und schon lange bekannt, siehe etwa [111]. Da man kein klassisches Verfahren zur effizienten Ermittlung der Periode kennt, entsteht daraus erst mit Shors Quantenalgorithmus in Abschnitt 8.6 ein effizientes Verfahren.

*Bemerkung:* Wir haben bei der Laufzeitanalyse vorausgesetzt, dass wir  $l$ -Bitzahlen mit der Schulmethode multiplizieren. Diese benötigt  $O(l^2)$  Bitoperationen und wird von Euklids Algorithmus und der modularen Potenzierung jeweils  $l$ -mal aufgerufen. Es gibt ein raffinierteres Verfahren [132], das in Zeit  $O(l \cdot \log l \cdot \log \log l)$  multipliziert, womit der klassische Teil von Shors Algorithmus in Zeit  $O(l^2 \cdot \log l \cdot \log \log l)$  ausführbar wird ( $l = \log n$ , wenn  $n$  die Eingabe des Algorithmus ist).

*Zusammenfassung:* Das Ziel ist, eine Zahl  $n$  zu faktorisieren. Primpotenzen und gerade Zahlen können wir ausschließen: in diesen Fällen lässt sich leicht ein Teiler finden. Können wir mit Laufzeit  $T(n)$  die Periode der Funktion

$$a^x \bmod n$$

für ein  $a$  bestimmen, gibt es einen randomisierten Algorithmus, der mit Laufzeit

$$O((\log n)^3) + T(n)$$

einen echten Teiler von  $n$  ermittelt. Die folgenden drei Abschnitte führen uns an den Quantenalgorithmus zur Periodenbestimmung heran.

## 8.3 Die schnelle Fouriertransformation

Die diskrete Fouriertransformation (DFT) ist ein sehr wichtiger klassischer Algorithmus. Vor allem eine Variante namens *schnelle Fouriertransformation*

ist aus vielen Anwendungen nicht mehr wegzudenken (z.B. Signalverarbeitung und Datenkompression; Stichworte MP3 und jpg); wir lernen die *fast fourier transformation* (FFT) am Ende des Abschnitts kennen. An dieser Stelle ist die DFT interessant, weil sie uns die Wirkungsweise der Quanten-Fouriertransformation besser verstehen lässt.

Die Aufgabenstellung: wir sollen Polynome multiplizieren, wie wir es aus der Schule kennen:

$$\begin{aligned}(6x^3 - 10x) \cdot (x^2 + 5) &= 6x^5 - 10x^3 + 30x^3 - 50x \\ &= 6x^5 + 20x^3 - 50x.\end{aligned}$$

Ein Polynom vom Grad  $n - 1$  hat die Form

$$\begin{aligned}&a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0 \\ &= \sum_{i=0}^{n-1} a_i x^i.\end{aligned}$$

Dabei ist der Koeffizient  $a_{n-1}$  ungleich 0. Um ein Polynom zu speichern, verwenden wir das Feld seiner Koeffizienten

$$(a_0, a_1, \dots, a_{n-1}).$$

Wir sprechen von der *Koeffizientendarstellung*. Wir betrachten Polynome vom Grad  $n - 1$ , da diese durch  $n$  Koeffizienten bestimmt sind und der Eingabegröße  $n$  entsprechen.

Damit lautet unsere Aufgabenstellung: Das Produkt der Polynome

Polynom-  
multiplikation

$$A(x) = \sum_{i=0}^{n-1} a_i x^i, \quad B(x) = \sum_{i=0}^{n-1} b_i x^i$$

ist

$$C(x) = \sum_{i=0}^{2n-2} c_i x^i \quad \text{mit} \quad c_i = \sum_{j=0}^i a_j b_{i-j}. \quad (8.2)$$

Der Koeffizient des Terms  $x^i$  ist gerade

$$a_0 b_i + a_1 b_{i-1} + \dots + a_i b_0,$$

weil

$$x^j x^{i-j} = x^i$$

ist. Gleichung 8.2 beschreibt eine Methode zur Multiplikation zweier Polynome, die

$$1 + 2 + \dots + (2n - 2) = \Theta(n^2)$$

Multiplikationen von Koeffizienten verwendet. Wir nennen dieses direkte Verfahren die *Schulmethode*. Wir werden ein Verfahren untersuchen, das mit  $O(n \log n)$  Multiplikationen auskommt und die *Schnelle Fouriertransformation* verwendet. Die Zeitersparnis ist immens, da der Logarithmus sehr langsam wächst. Algorithmen mit Laufzeit  $O(n \log n)$  bezeichnet man als *quasilinear*.

Wir beginnen mit einer alternativen Darstellung von Polynomen. Es liegt nahe, Polynome als Funktionen oder als Kurvenzüge in der Ebene zu betrachten. Solche vom Grad 1 sind, geometrisch betrachtet, Geraden. Durch zwei Punkte verläuft genau eine Gerade. Also können wir zwei solche Punkte zur Beschreibung einer Geraden verwenden. Allgemein lässt sich ein Polynom vom Grad  $n-1$  durch  $n$  Punkte beschreiben; man spricht von der *Stützstellendarstellung*.

Stützstellendarstellung

Gegeben seien  $n$  Punkte

$$(x_0, y_0), (x_1, y_1), \dots, (x_{n-1}, y_{n-1}),$$

so dass die Werte  $x_i$  paarweise verschieden sind. Dann gibt es genau ein Polynom  $A$  vom Grad  $n-1$ , für das

$$A(x_i) = y_i \text{ für } i = 0, \dots, n-1$$

gilt.

Der Beweis enthält eine interessante Beschreibung der Polynommultiplikation als Produkt von Matrizen. Die Gleichung

$$A(x_i) = y_i \text{ für } i = 0, \dots, n-1$$

ist mit der Gleichung

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

gleichwertig. Die Matrix auf der linken Seite bezeichnen wir mit  $V(x)$ ; man nennt sie *Vandermonde Matrix*. Man weiß:  $V(x)$  ist umkehrbar! Also lässt sich die Koeffizientendarstellung  $a = (a_0, \dots, a_{n-1})^T$  eindeutig aus dem Vektor  $y = (y_0, \dots, y_{n-1})^T$  berechnen:

$$a = V(x)^{-1}y.$$

Den Schluss von den Stützstellen auf die Koeffizientendarstellung bezeichnet man auch als *Interpolation*. Für Leser, die der Beweis interessiert, dass  $V(x)$  umkehrbar ist: Die Determinante ist

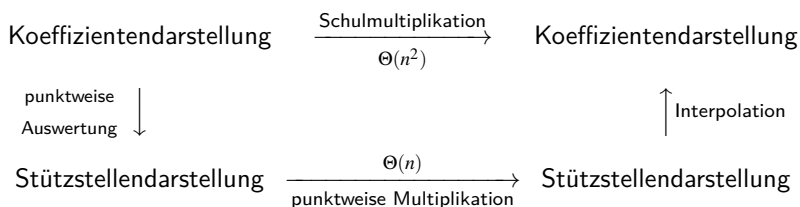
$$\prod_{i < j} (x_j - x_i).$$

In der Stützstellendarstellung lassen sich Polynome schneller multiplizieren. Ist nämlich  $(x_0, A(x_0))$  Stützstelle von  $A$  und  $(x_0, B(x_0))$  von  $B$ , so ist  $(x_0, A(x_0) \cdot B(x_0))$  Stützstelle des Produktes  $C$ . Allerdings kann  $C$  vom Grad

$2n - 2$  sein, so dass wir eine entsprechende Anzahl an Stützstellen benötigen, um  $C$  zu bestimmen.

Aus jeweils  $2n - 1$  gemeinsamen Stützstellen zweier Polynome vom Grad  $n - 1$  lässt sich in Zeit  $\Theta(n)$  eine Stützstellendarstellung des Produkts berechnen.

Das folgende Schema fasst die Situation zusammen.



**Aufgabe 8.3:** Überlegen Sie, wie die punktweise Auswertung und die Interpolation in Zeit  $\Theta(n^2)$  ausführbar sind.

Das Problem ist also: Wie lässt sich die Koeffizientendarstellung schnell in eine Stützstellendarstellung umwandeln und umgekehrt? Verwenden wir eine Methode mit quadratischer Laufzeit – gemäß Übung 8.3 – haben wir nichts gewonnen. Die Gesamtlaufzeit bleibt quadratisch und scheint schlechter als die Schulmethode zu sein.

### Einheitswurzeln

Unser Problem lässt sich mit einer besonderen Klasse von komplexen Zahlen lösen. Eine Lösung der Gleichung

$$x^n = 1$$

in  $\mathbb{C}$  heißt *komplexe  $n$ -te Einheitswurzel*. In der Algebra werden Nullstellen als Wurzeln bezeichnet. Alle Nullstellen von  $x^n - 1$  haben Länge 1 und sind von der Form  $\sqrt[n]{1}$ ; daher stammt der Name. Den Zusatz *komplex* lassen wir in der Folge oft fort.

Die (komplexen)  $n$ -ten Einheitswurzeln sind die Potenzen der komplexen Zahl

$$\omega_n = e^{2\pi i/n}.$$

Es gibt genau  $n$  von ihnen.

Einheitswurzeln

Im Anhang (Abschnitt A.1) wird die komplexe Exponentialfunktion dargestellt. Damit lässt sich einsehen, dass die  $n$ -ten Einheitswurzeln

$$\omega_n^0, \omega_n^1, \dots, \omega_n^{n-1}$$



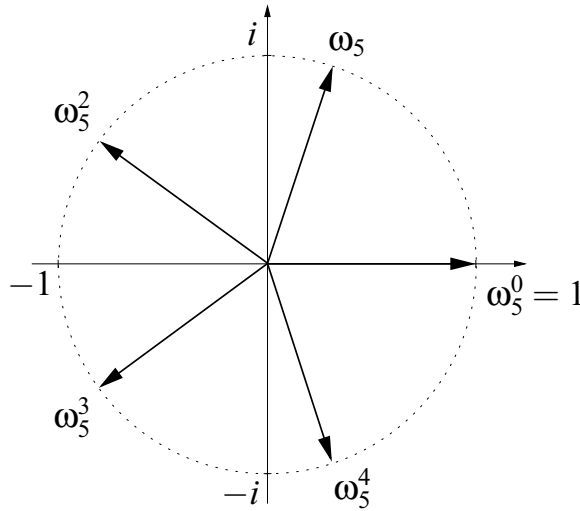


Abbildung 8.3: Die komplexen 5-ten Einheitswurzeln

oder

$$e^{2\pi i k/n} \quad (k = 0, \dots, n-1),$$

den Einheitskreis in gleich große Segmente unterteilen, wie Abbildung 8.3 zeigt. Da  $\omega_n^n = e^{2\pi i} = 1$  gilt, ist jede  $n$ -te Einheitswurzel eine Nullstelle von  $x^n - 1$ .

**Aufgabe 8.4:** Visualisieren Sie die  $k$ -ten Einheitswurzeln für  $k = 2, 3, 4$  und  $8$ .

Wir interessieren uns für Einheitswurzeln, weil sich Polynome an diesen Stellen besonders einfach auswerten lassen. Hat man ein Polynom  $A$  an der Stelle  $\omega_n$  berechnet, lassen sich die Zwischenergebnisse für die Auswertung an der Stelle  $\omega_n^2$  verwenden und so fort. Es folgt der entscheidende Begriff:

DFT

Sei  $A$  ein Polynom in Koeffizientendarstellung  $a = (a_0, \dots, a_{n-1})^T$ . Der Vektor  $y = (y_0, \dots, y_{n-1})^T$  mit

$$y_k = A(\omega_n^k)$$

heißt *diskrete Fouriertransformation* von  $a$ :

$$y = \text{DFT}_n(a).$$

Die DFT eines Polynoms  $A$  vom Grad  $n-1$  ist dessen Stützstellendarstellung bezüglich der  $n$ -ten Einheitswurzeln. Es gilt für die zu  $\omega_n^k$  gehörende Komponente

$$y_k = \sum_{j=0}^{n-1} a_j \omega_n^{kj},$$

oder

$$\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \omega_n^3 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \omega_n^6 & \cdots & \omega_n^{2(n-1)} \\ 1 & \omega_n^3 & \omega_n^6 & \omega_n^9 & \cdots & \omega_n^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \omega_n^{3(n-1)} & \cdots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

$\text{DFT}_n$  wird also durch die Matrix mit Eintrag  $\omega_n^{(j-1)(k-1)}$  an der Stelle  $(j, k)$  beschrieben. Die Umkehrabbildung  $\text{DFT}_n^{-1}$  ist dann die Matrix mit dem Eintrag  $\frac{1}{n} \omega_n^{-(j-1)(k-1)}$  an der Stelle  $(j, k)$ .

### Die schnelle Fouriertransformation

Nun kommen wir zu einem Verfahren, dass  $\text{DFT}_n$  in Zeit  $\Theta(n \log n)$  berechnet, in dieser Zeit also ein Polynom an  $n$  Stellen auswertet. Wir nutzen aus, dass Potenzen von  $n$ -ten Einheitswurzeln wieder Einheitswurzeln ergeben.

Sei  $n$  gerade und positiv. Quadrieren wir alle  $n$ -ten Einheitswurzeln, erhalten wir die  $(n/2)$ -ten Einheitswurzeln:

$$\{1^2, (\omega_n)^2, (\omega_n^2)^2, \dots, (\omega_n^{n-1})^2\} = \{1, \omega_{n/2}, \omega_{n/2}^2, \dots, \omega_{n/2}^{n/2-1}\}.$$

Halbierungs-  
lemma

Die Gleichung  $(\omega_n^k)^2 = \omega_{n/2}^k$  lässt sich an einer Zeichnung wie Abbildung 8.3 verdeutlichen (zum Beispiel für den Fall  $n = 8$ ). Formal gilt für jedes  $k \in \mathbb{N}$

$$(\omega_n^k)^2 = \left(e^{\frac{2\pi i}{n}}\right)^{2k} = \left(e^{\frac{2\pi i}{n/2}}\right)^k = \omega_{n/2}^k.$$

Wir erhalten durch Quadrieren der  $n$ -ten Einheitswurzeln jede  $(n/2)$ -te genau zweimal.

**Aufgabe 8.5:** Zeigen Sie:

$$(\omega_n^k)^2 = (\omega_n^{k+n/2})^2.$$

Was folgt aus diesem Lemma für die Berechnung der diskreten Fouriertransformation? Wir müssen Funktionswerte für  $n$  verschiedene Eingaben

bestimmen, für die  $n$ -ten Einheitswurzeln. Wenn wir nun *einmal* alle Werte quadrieren, *halbiert* sich die Anzahl der Werte. Die Situation ähnelt jenen rekursiven Sortierverfahren, bei denen in jedem Schritt das zu sortierende Feld halbiert wird (zum Beispiel *Mergesort* oder *Quicksort* im mittleren Fall).

Wir können Ähnliches erreichen und zerlegen ein Polynom

$$A(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1}$$

mit Grad  $n - 1$  in zwei Polynome mit (etwa) halbiertem Grad:

$$A^0(x) = a_0 + a_2x + a_4x^2 \dots + a_{n-2}x^{n/2-1},$$

$$A^1(x) = a_1 + a_3x + a_5x^2 \dots + a_{n-1}x^{n/2-1}.$$

Es gilt

$$A(x) = A^0(x^2) + x \cdot A^1(x^2). \quad (8.3)$$

Damit können wir  $A$  bei

$$\omega_n^0, \omega_n^1, \dots, \omega_n^{n-1}$$

auswerten, indem wir die Polynome  $A^0$  und  $A^1$  mit halbiertem Grad bei

$$(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{n-1})^2 \quad (8.4)$$

auswerten. Dazu wenden wir  $\text{DFT}_{n/2}$  auf  $A_0$  und  $A_1$  an. Denn nach dem Halbierungslemma ergibt (8.4) gerade die  $n/2$ -ten Einheitswurzeln,  $n/2$  an der Zahl. Dann setzen wir die Teilergebnisse gemäß (8.3) zusammen.

### Algorithmus $\text{FFT}_n$

Eingabe: Ein Feld  $(a_0, \dots, a_{n-1})$ ,  $n$  ist eine Zweierpotenz

Ausgabe:  $y = \text{DFT}_n(a_0, \dots, a_{n-1})$

FFT

1. Falls  $n - 1 = 0$ : Ausgabe  $a$ .
2.  $a^0 \leftarrow (a_0, a_2, \dots, a_{n-2})$ ,  $a^1 \leftarrow (a_1, a_3, \dots, a_{n-1})$
3.  $y^0 \leftarrow \text{FFT}_{n/2}(a^0)$ ,  $y^1 \leftarrow \text{FFT}_{n/2}(a^1)$
4. Setze  $y^0$  und  $y^1$  gemäß (8.3) zu  $y = \text{DFT}_n(a)$  zusammen.

Wie bei dem Divide-and-conquer-Algorithmus *Mergesort* ergibt sich für die Laufzeit  $T$ :

$$T(n) = 2T(n/2) + \Theta(n). \quad (8.5)$$

Der erste Summand ist der zeitliche Aufwand zweier rekursiver Aufrufe mit halbiertem Eingabelänge.  $\Theta(n)$  ist der Aufwand des Zusammensetzens gemäß (8.3). Es ist bekannt, dass die Rekursion (8.5) die Lösung  $T(n) = \Theta(n \log n)$  hat.

Der Algorithmus findet sich in vielen Büchern zum Thema *Algorithmen und Datenstrukturen*. Um Schritt 4 in linearer Zeit zu bewerkstelligen, werden weitere Eigenschaften der  $n$ -ten Einheitswurzeln verwendet. Eine wichtige beschreibt das Summationslemma, das wir noch verwenden werden.

Die  $n$ -ten Einheitswurzeln addieren sich zu 0:

$$\sum_{i=0}^{n-1} \omega_n^i = 0.$$

Es gilt für ganzzahlige  $k \geq 0$ , die kein Vielfaches von  $n$  sind:

$$\sum_{i=0}^{n-1} (\omega_n^k)^i = 0.$$

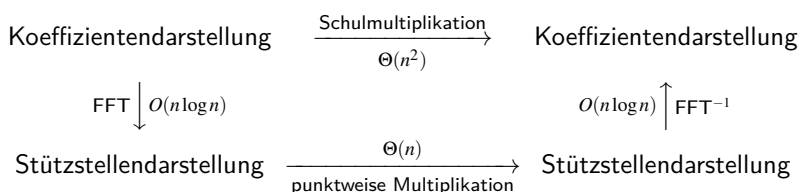
Summationslemma

**Aufgabe 8.6:** Veranschaulichen Sie das Summationslemma mit Hilfe von Abbildung 8.3 und beweisen Sie es formal. Verwenden Sie die Formel für die geometrische Reihe

$$\sum_{i=0}^n x^i = \frac{x^{n+1} - 1}{x - 1} \quad \text{für } x \neq 1.$$

Die Ergebnisse dieses Abschnitts sind in dem folgenden Schema zusammengefasst.

Zusammenfassung



Wir können aus zwei Polynomen in Koeffizientendarstellung mit Laufzeit  $O(n \log n)$  Stützstellendarstellungen berechnen, nämlich deren diskrete Fouriertransformation. Diese lassen sich in linearer Zeit multiplizieren und in  $O(n \log n)$  gewinnen wir daraus die Koeffizientendarstellung des Produktes. Das gesamte Verfahren benötigt Zeit  $\Theta(n \log n)$ .

Dieses Verfahren ähnelt unserem Vorgehen bei dem Problem von Deutsch. Dort haben wir den Ausgangszustand mittels der Matrix  $H$  transformiert. Unsere Berechnung wurde in der Hadamard-Basis ausgeführt, und vor der Messung haben wir das Ergebnis in die Standardbasis zurücktransformiert (mittels  $H^{-1} = H$ ). Im weiteren werden wir sehen, dass es eine Beziehung zwischen Hadamard-Transformation und Fouriertransformation gibt.

## 8.4 Die Quanten-Fouriertransformation

In diesem Abschnitt lernen wir die Quantenvariante der diskreten Fouriertransformation DFT kennen.

QFT

Die Quanten-Fouriertransformation der Ordnung  $N$  wird durch die unitäre Matrix

$$\text{QFT}_N = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_N & \omega_N^2 & \omega_N^3 & \cdots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \omega_N^6 & \cdots & \omega_N^{2(N-1)} \\ 1 & \omega_N^3 & \omega_N^6 & \omega_N^9 & \cdots & \omega_N^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \omega_N^{3(N-1)} & \cdots & \omega_N^{(N-1)(N-1)} \end{pmatrix}$$

beschrieben, wobei  $\omega_N$  die  $N$ -te Einheitswurzel  $e^{2\pi i/N}$  bezeichnet. Ist  $|0\rangle, \dots, |N-1\rangle$  eine Orthonormalbasis, operiert  $\text{QFT}_N$  auf dem  $N$ -dimensionalen Basisvektor  $|j\rangle$  folgendermaßen:

$$\text{QFT}_N |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{j \cdot k} |k\rangle.$$

Der letzten Identität sieht man die Ähnlichkeit mit der Hadamard-Transformation an:

$$H_n |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle,$$

siehe Seite 60. Das Bild eines Basisvektors ist eine Summe über alle Basisvektoren; das Urbild geht in die Phase ein. Dabei gilt  $(-1)^{x \cdot y} = \omega_2^{x \cdot y}$ . Ein wesentlicher Unterschied ist, dass bei  $H_n$  Boolesche Vektoren  $x, y$  Verwendung finden – die Multiplikation ist das Skalarprodukt – während bei der Quanten-Fouriertransformation ganze Zahlen  $j, k$  multipliziert werden. In diesem Sinne operiert  $\text{QFT}_N$  in der Menge der ganzen Zahlen. Genauer: In der Menge  $\{0, \dots, N-1\}$  mit der Multiplikation modulo  $N$ . Es gilt

$$\omega_N^j = \omega_N^{j \bmod N},$$

da

$$\omega_N^{k+N} = \omega_N^k \omega_N^N = \omega_N^k.$$

Auf den Vektor  $|0 \dots 0\rangle$  wirken beide Transformationen gleich: das Ergebnis ist die gleichgewichtete Superposition. Für einen beliebigen Zustandsvektor

$$|v\rangle = \sum_{j=0}^{N-1} \alpha_j |j\rangle$$

ist das Ergebnis

$$\text{QFT}_N |v\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \left( \sum_{j=0}^{N-1} \alpha_j \omega_N^{j \cdot k} \right) |k\rangle.$$

Der Zusammenhang mit der DFT wird klar, wenn wir uns die Amplituden als Koeffizienten eines Polynoms vorstellen. Im klassischen Fall ist der Normalisierungsfaktor  $\frac{1}{\sqrt{N}}$  nicht nötig. Die QFT wirkt auch als Basistransformation; sie überführt die Standardbasis  $|0\rangle, \dots, |N-1\rangle$  in die Basis  $\text{QFT}_N |0\rangle, \dots, \text{QFT}_N |N-1\rangle$ . Mag die QFT anfangs etwas kompliziert wirken, ihre Wirkungsweise wird durch ihre Verwendung in Shors Algorithmus klarer.

Die inverse Quanten-Fouriertransformation  $\text{QFT}^{-1} = \text{QFT}^\dagger$  – es handelt sich um eine unitäre Transformation – wirkt auf einen Basiszustand  $|j\rangle$  als

Inverse QFT

$$\begin{aligned} \text{QFT}_N^{-1} |j\rangle &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \frac{1}{\omega_N^{j \cdot k}} |k\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-2\pi i \cdot j \cdot k / N} |k\rangle. \end{aligned}$$

Bevor wir die Quanten-Fouriertransformation im nächsten Abschnitt anwenden können, müssen wir sie noch als Schaltkreis aus lokalen Quantengattern konstruieren.

### Realisierung der Quanten-Fouriertransformation

Die Quanten-Fouriertransformation eines Zustands erzeugt eine Superposition, in der für jede Amplitude zwei ganze Zahlen multipliziert werden. Sie ist unitär, aber können wir sie auch effizient ausführen?

$\text{QFT}_N$  lässt sich mit  $n(n+1)/2 = O(n^2)$  vielen lokalen Quantengattern ausführen ( $N = 2^n$ ).

Ziel

Um die Operation QFT aus lokalen Gattern aufzubauen, müssen wir sie nach Art des Tensorproduktes zerlegen. Dann können wir erkennen, wie mit den einzelnen Quantenbits zu verfahren ist. Was ist dafür zu tun? Basisvektoren interpretieren wir als Binärdarstellung einer ganzen Zahl. In die Phase  $\omega_N^{j \cdot k}$  geht das Produkt zweier ganzer Zahlen ein und jedes Bit ist relevant. Somit ist zunächst einmal völlig unklar, wie sich die QFT durch lokale Operationen realisieren lassen soll.

Wir beginnen unsere Betrachtung mit  $\text{QFT}_4 |x\rangle$  für einen beliebigen Zweibitzustand  $|x_1 x_0\rangle$ .

$$\begin{aligned} \text{QFT}_4 |x\rangle &= \frac{1}{2} \sum_{y=0}^3 \omega_4^{xy} |y\rangle \\ &= \frac{1}{2} (\omega_4^0 |00\rangle + \omega_4^1 |01\rangle + \omega_4^2 |10\rangle + \omega_4^3 |11\rangle). \end{aligned}$$

Das lässt sich als Produkt schreiben:

$$1/2 \left( |0\rangle + \omega_4^{2x} |1\rangle \right) \cdot \left( |0\rangle + \omega_4^x |1\rangle \right),$$

wie wir leicht nachrechnen. Die erste Klammer beschreibt den Zustand von Bit  $y_1$  und die zweite den von  $y_0$ . Interessantes zeigt folgende Rechnung:

$$\omega_4^{2x} = e^{2\pi i \cdot 2x/4} = \omega_2^x.$$

Wir folgern:

$$\text{QFT}_4 |x\rangle = 1/2 \left( |0\rangle + \omega_2^x |1\rangle \right) \cdot \left( |0\rangle + \omega_4^x |1\rangle \right),$$

und die lästige Ganzzahlmultiplikation im Exponenten ist zu einem Wechsel der Einheitswurzel mutiert. Mit dieser Idee lässt sich zeigen:

Der Zustand  $|x\rangle$  bestehe aus  $n$  Bits,  $N = 2^n$ . Dann gilt

$$\text{QFT}_N |x\rangle = \frac{1}{\sqrt{N}} \left( |0\rangle + \omega_2^x |1\rangle \right) \cdot \left( |0\rangle + \omega_4^x |1\rangle \right) \cdot \dots \cdot \left( |0\rangle + \omega_N^x |1\rangle \right). \quad (8.6)$$

Die Klammern entsprechen den Zuständen der Bits  $|y_{n-1}\rangle, |y_{n-2}\rangle, \dots, |y_1\rangle, |y_0\rangle$ .

**Aufgabe 8.7:** Zeigen Sie (8.6) für den Fall  $N = 8$ .

Wie können wir ein Quantenregister in den von Gleichung (8.6) beschriebenen Zustand versetzen? Es gibt einen Zusammenhang zwischen der QFT und der Hadamard-Transformation: Wenden wir  $H$  auf ein Quantenbit  $|x\rangle$  im Zustand 0 oder 1 an, ist das Ergebnis

$$\begin{aligned} H|x\rangle &= \frac{1}{\sqrt{2}} (|0\rangle + (-1)^x |1\rangle) \\ &= \frac{1}{\sqrt{2}} (|0\rangle + \omega_2^x |1\rangle). \end{aligned}$$

Für die Fouriertransformation

$$\text{QFT}_N : |j\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{j \cdot k} |k\rangle,$$

genügen allerdings die 2-ten Einheitswurzeln nicht. Abhilfe schafft folgende lokale unitäre Transformation:

Phasen-Rotation

Die Operation  $R_m$  auf einem Bit wird von der Matrix

$$R_m = \begin{pmatrix} 1 & 0 \\ 0 & \omega_m \end{pmatrix}$$

beschrieben. Ist das Bit 1, wird die Amplitude mit  $\omega_m$  multipliziert, die Phase wird also um den Winkel  $2\pi/m$  gedreht.

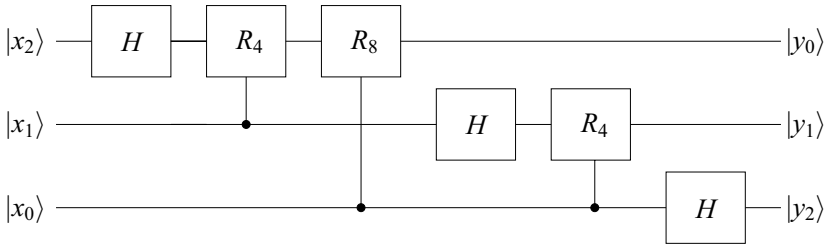


Abbildung 8.4: Quanten-Fouriertransformation auf drei Bits

Der Übersichtlichkeit halber weichen wir von der allgemein gebräuchlichen Bezeichnung ab, in der  $R_d$  mit  $\omega_{2^d}$  multipliziert.

**Aufgabe 8.8:** Berechnen Sie

$$R_4 \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

**Aufgabe 8.9:** Zeigen Sie, dass  $R_m$  unitär ist.

Nun müssen wir nur noch betrachten, wie sich die von  $x$  abhängige Phase  $\omega_m^x$  erzeugen lässt. Wir müssen gemäß der binären Kodierung der Zahl  $x$  die korrekten Transformationen anwenden. Dazu betrachten wir

$$\text{QFT}_8 |x\rangle = \frac{1}{\sqrt{8}} \left( |0\rangle + \omega_2^x |1\rangle \right) \cdot \left( |0\rangle + \omega_4^x |1\rangle \right) \cdot \left( |0\rangle + \omega_8^x |1\rangle \right).$$

Der Vektor  $x = (x_2, x_1, x_0)$  hat den ganzzahligen Wert

$$4x_2 + 2x_1 + x_0.$$

Im Term  $\omega_2^x$  interessiert jedoch nur, ob  $x$  gerade oder ungerade ist:  $\omega_2^{2k} = 1$  und  $\omega_2^{2k+1} = -1$  für alle  $k$ . Wir folgern:

$$\omega_2^x = \omega_2^{x_0}$$

ergibt den Zustand von Bit  $y_2$  und ist mit einer Hadamard-Transformation auf  $x_0$  realisierbar. Des Weiteren ergeben sich für die Bits  $y_1$  und  $y_0$ :

$$\omega_4^x = \omega_4^{2x_1+x_0} = \omega_2^{x_1} \omega_4^{x_0} \text{ und } \omega_8^x = \omega_8^{4x_2+2x_1+x_0} = \omega_2^{x_2} \omega_4^{x_1} \omega_8^{x_0}.$$

Die Phasenrotation  $\omega_m^{x_i}$  (mit Wirkung auf die  $|1\rangle$ -Komponente) lässt sich mit einem  $x_i$ -gesteuerten  $R_m$ -Gatter realisieren. Bit  $y_1$  ergibt sich aus Bit  $x_1$ , indem zunächst  $H$  angewendet wird, gefolgt von einem  $x_0$ -gesteuerten  $R_4$ -Gatter. Einen Schaltkreis für  $\text{QFT}_8$  zeigt Abbildung 8.4. Der Index  $m$  der  $R_m$ -Gatter ergibt sich aus dem Abstand der beteiligten Bits  $x_i$  und  $x_j$ ,  $i > j$ : dann gilt  $m = 2^{i-j+1}$ .



Auf  $|x_2\rangle$  wenden wir zwei  $R_m$ -Gatter an. Das erste ist durch  $|x_1\rangle$  gesteuert und führt  $R_4$  aus mit  $4 = 2^{2-1+1}$ . Das zweite wird durch  $|x_0\rangle$  gesteuert und führt  $R_8$  aus mit  $8 = 2^{2-0+1}$ .

**Aufgabe 8.10:** Beweisen Sie, dass der Schaltkreis in Abbildung 8.4 die Transformation  $\text{QFT}_8$  realisiert.

So lässt sich  $\text{QFT}_N$  für jede Zweierpotenz  $N = 2^n$  realisieren. Auf das Bit  $|x_i\rangle$  werden  $i$  verschiedene  $R_m$ -Gatter angewendet. Diese werden von den Bits  $|x_0\rangle, \dots, |x_{i-1}\rangle$  gesteuert. Für  $j = 1, \dots, i-1$  steuert  $|x_j\rangle$  ein  $R_m$ -Gatter mit  $m = 2^{i-j+1}$ . Im Schaltkreis in Abbildung 8.4 sind die Gatter so angeordnet, dass diejenigen Bits zuerst transformiert werden, die keinen Einfluss mehr auf andere Bits haben. Wir beweisen die folgende Aussage nicht formal, da der analysierte Fall  $N = 8$  bereits alle Ideen enthält:

$\text{QFT}_N$  lässt sich mit  $n(n+1)/2 = O(n^2)$  vielen lokalen Quantengattern ausführen ( $N = 2^n$ ).

Auch die klassische Fouriertransformation ist bei periodischen Funktionen äußerst nützlich. Jedoch benötigt die schnelle Fouriertransformation  $\text{FFT}_N$  Zeit  $\Theta(N \log N)$ , während die Quantenversion mit  $\Theta((\log N)^2)$  Gattern auskommt; der Grund ist die Parallelität. Dieser exponentielle Unterschied wird von Shors Algorithmus genutzt.

## 8.5 Simons Algorithmus

Zur Vorbereitung auf das Hauptergebnis dieses Kapitels studieren wir einen Algorithmus, der uns sehr vertraute Methoden verwendet und strukturell Shors Algorithmus sehr ähnlich ist. Die Problemstellung erinnert an die intensiv diskutierten Algorithmen von Deutsch und Deutsch-Jozsa.

Problemstellung      Uns ist eine Funktion

$$f : \{0, 1\}^n \rightarrow \{0, 1\}^n$$

gegeben, für die eine der beiden Alternativen gilt:

1.  $f$  ist bijektiv (verschiedene Vektoren aus  $\{0, 1\}^n$  haben unterschiedliche Bilder).
2. Je zwei Vektoren aus  $\{0, 1\}^n$  haben dasselbe Bild, und es gibt einen Vektor  $s \in \{0, 1\}^n, s \neq 0$  mit

$$f(x) = f(x') \text{ genau dann, wenn } x \oplus s = x'.$$

Wir sollen entscheiden, welcher Fall für ein gegebenes  $f$  gilt. Im zweiten Fall ist  $s$  zu bestimmen.

Dieser Vektor  $s$  beschreibt, wie zwei Vektoren mit demselben Bild auseinander hervorgehen. Wir bezeichnen ihn als die *Periode* der Funktion  $f$ , denn für jede Eingabe  $x$  gilt

$$f(x \oplus s) = f(x).$$

**Beispiel 8.4:** Wir betrachten zwei Funktionen  $\{0,1\}^2 \rightarrow \{0,1\}^2$ :

1. Die Funktion

$$f(x,y) = (0, x \oplus y)$$

operiert als

00	$\mapsto$	00
01	$\mapsto$	01
10	$\mapsto$	01
11	$\mapsto$	00

Wir sehen, dass je zwei Vektoren dasselbe Bild haben: 00 und 11 werden auf 00, 01 und 10 auf 01 abgebildet. Die Periode ist der Vektor  $(1,1)$ .

2. Die Funktion  $\text{CNOT}(x,y) = (x, y \oplus x)$  ist bijektiv.

Der folgende Algorithmus (siehe auch Abbildung 8.5) stammt von dem kanadischen Kryptologen Daniel R. Simon, der für Microsoft Research in Redmond tätig ist.

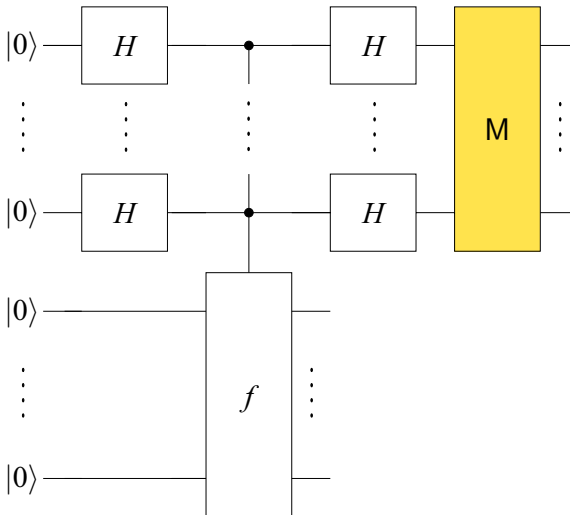


Abbildung 8.5: Schaltkreis für Simons Algorithmus

### Simons Algorithmus

Wir rechnen mit zwei  $n$ -Bit Quantenregistern  $|a\rangle|b\rangle$ . Eingabe ist ein Quantenorakel  $U_f : |a\rangle|b\rangle \mapsto |a\rangle|b \oplus f(a)\rangle$

1.  $R = |a\rangle|b\rangle \leftarrow |0 \dots 0\rangle|0 \dots 0\rangle$
2. Wende die Hadamard-Transformation auf Register  $|a\rangle$  an:  
 $R \leftarrow \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|0 \dots 0\rangle$
3. Wende das Orakel an:  
 $R \leftarrow U_f R = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|f(x)\rangle$
4. Miss Register  $|b\rangle$ , das zweite Register
5. Wende die Hadamard-Transformation auf  $|a\rangle$  an:  
 $|a\rangle \leftarrow H_n |a\rangle$
6. Miss  $|a\rangle$ . Das Ergebnis ist ein Vektor  $z$ ; gib diesen aus.

Wiederhole diese Prozedur bis  $n - 1$  linear unabhängige Ergebnisse  $z_1, \dots, z_{n-1}$  erzeugt wurden.

Dies ist der Quantenteil von Simons Algorithmus. Das Endergebnis ergibt sich mit einer klassischen Berechnung, die wir nach der Analyse dieses Verfahrens studieren. Die Messung in Schritt 4 ist für das Resultat ohne Belang; darum wird diese in dem Schaltkreis in Abbildung 8.5 nicht vorgenommen. Sie erleichtert uns jedoch die Analyse des Algorithmus.

### Analyse

**Schritt 4** Die Ergebnisse der ersten drei Schritte sind dem Algorithmus direkt anzusehen. Die Analyse beginnt mit Schritt 4. Was passiert beim Messen des Registers  $|b\rangle$ ? Das Orakel  $U_f$  hat die beiden Register in einen verschränkten Zustand versetzt; interessant ist die Wirkung auf Register  $|a\rangle$ . Wir nehmen an, die Messung in Schritt 4 hat das Ergebnis  $y$ .

$f$  bijektiv Ist  $f$  bijektiv, so werden wir bei einer anschließenden Messung von  $|a\rangle$  den Wert  $f^{-1}(y)$  beobachten.

$f$  periodisch Im weiteren konzentriert sich diese Analyse auf den Fall, dass  $f$  nicht bijektiv ist. Dann hat  $y$  genau zwei Urbilder. Wir nennen sie  $x$  und  $x'$  und wissen: es gilt  $x \oplus x' = s$ . Wir folgern: Hätten wir beide Register gemessen, wäre das Ergebnis entweder  $|x\rangle|f(x)\rangle$  oder  $|x \oplus s\rangle|f(x)\rangle$ . Da wir tatsächlich nur das zweite messen, ist  $R$  nach Schritt 4 in einem Zustand der Form

$$|\phi_4\rangle = \frac{1}{\sqrt{2}}(|\hat{x}\rangle + |\hat{x} \oplus s\rangle)|f(\hat{x})\rangle,$$

siehe Abschnitt 2.8.  $|\hat{x}\rangle$  könnte dabei jeder Zustandsvektor sein, da  $f(\hat{x})$  gleichverteilt unter allen Bildern der Funktion  $f$  gewählt wurde.

Messen wir an dieser Situation das erste Register, können wir dem Ergebnis nichts entnehmen. Wir müssen den störenden *Offset*  $\hat{x}$  aus der Superposition  $|\phi_4\rangle$  entfernen. Durch die Anwendung der Hadamard-Transformation lautet der Folgezustand

Schritt 5

$$|\phi_5\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} \frac{1}{\sqrt{2}} ((-1)^{\hat{x} \cdot z} + (-1)^{(\hat{x} \oplus s) \cdot z}) |z\rangle |f(\hat{x})\rangle.$$

Wir bezeichnen die Amplitude von  $|z\rangle |f(\hat{x})\rangle$  als

$$\alpha_z = 1/\sqrt{2^{n+1}} \cdot ((-1)^{\hat{x} \cdot z} + (-1)^{\hat{x} \cdot z \oplus s \cdot z}).$$

Wir unterscheiden die Fälle, dass  $z \cdot s$  gerade oder ungerade ist.

Es gelte  $z \cdot s \equiv 0 \pmod{2}$ . Dann gilt

Fall 1

$$\begin{aligned} \alpha_z &= 1/\sqrt{2^{n+1}} \cdot ((-1)^{\hat{x} \cdot z} + (-1)^{\hat{x} \cdot z}) \\ &= \pm 1/\sqrt{2^{n-1}}. \end{aligned}$$

Dementsprechend gilt für  $z \cdot s \equiv 1 \pmod{2}$

Fall 2

$$\begin{aligned} \alpha_z &= 1/\sqrt{2^{n+1}} \cdot ((-1)^{\hat{x} \cdot z} + (-1)^{\hat{x} \cdot z \oplus 1}) \\ &= 0. \end{aligned}$$

Wir sehen, der störende Einfluss von  $\hat{x}$  wurde beseitigt. Aber wissen wir jetzt mehr über die Periode  $s$ ? Ja! Nach der abschließenden Messung im letzten Schritt gilt für das Ergebnis  $z$ : Das Skalarprodukt mit  $s$  ist eine gerade Zahl,

$$z \cdot s \equiv 0 \pmod{2}.$$

Es gibt genau  $2^{n-1}$  Vektoren  $z$  mit dieser Eigenschaft. Kennen wir  $n-1$  linear unabhängige davon, können wir unsere Berechnung wie folgt beenden:

### 7. Löse das lineare Gleichungssystem

$$z_1 \cdot t = 0, \dots, z_{n-1} \cdot t = 0$$

in den Unbekannten  $(t_1, \dots, t_n) = t$ .

$s$  sei die Lösung ungleich dem Nullvektor.

### 8. Ist $f(0) = f(s)$ :

Ausgabe:  $f$  ist periodisch bezüglich  $s$ .

Anderenfalls Ausgabe:  $f$  ist bijektiv.

Abschluss von  
Simons  
Algorithmus

Der Vektorraum, der von einem Vektor  $z \in \{0,1\}^n$  aufgespannt wird, besitzt nur zwei Elemente:  $0 \cdot z$  und  $1 \cdot z$ . Darum hat das Gleichungssystem aus Schritt 7 zwei Lösungen und nur eine ungleich 0.

$f$  bijektiv

Bisher haben wir außer acht gelassen, dass  $f$  eventuell gar nicht periodisch ist. Dann sind die  $z_i$  völlig gleichverteilt gezogen worden, und die gewählte Lösung  $s$  des Gleichungssystems ist ein Zufallsvektor. Es gilt dann

$$f(0) \neq f(s).$$

Die Funktion ist bijektiv, und  $s$  ist ungleich dem Nullvektor.

Laufzeit

Eine wichtige Frage ist noch offen: wie lange dauert es, bis wir  $n - 1$  linear unabhängige Vektoren gefunden haben? Wir erhalten in einem Durchlauf von Simons Algorithmus jeden der  $2^{n-1}$  Vektoren  $z \cdot s \equiv 0 \pmod 2$  mit gleicher Wahrscheinlichkeit. Es ist sehr unwahrscheinlich, dass die ersten beiden Vektoren  $z_1$  und  $z_2$  linear abhängig sind. Dazu müssten beide gleich oder einer von ihnen der Nullvektor sein. Ein dritter Vektor ist von den ersten beiden linear abhängig, wenn er  $z_1, z_2, z_1 \oplus z_2$  oder  $0$  lautet. Das sind alle Linearkombinationen. Das deutet darauf hin, dass wir durch blindes Auswählen leicht  $n - 1$  linear unabhängige Vektoren finden. Man kann formal zeigen: im obigen Fall sind mit Wahrscheinlichkeit größer  $1/4$  bereits die ersten  $n - 1$  Vektoren linear unabhängig.

Das Gleichungssystem aus Schritt 7 lässt sich mit dem Gaußschen Eliminationsverfahren in Zeit  $O(n^3)$  lösen; dabei wird nebenher ermittelt, ob die Vektoren bereits linear unabhängig sind.

**Beispiel 8.5:** Wir betrachten die Funktion  $f: \{0, 1\}^3 \rightarrow \{0, 1\}^3$  mit

$$f(x_2, x_1, x_0) = (x_2, x_1, 0).$$

Die Periode ist  $s = (0, 0, 1)^T$ ;  $f$  ist von der Komponente  $x_0$  unabhängig. Nach Schritt 3 befindet sich das Register im Zustand:

$$\frac{1}{\sqrt{8}}(|000\rangle|000\rangle + |001\rangle|000\rangle + |010\rangle|010\rangle + |011\rangle|010\rangle + \dots + |111\rangle|110\rangle).$$

Wir messen das zweite Register: Das Ergebnis ist eine Bitfolge  $|x_2 x_1 x_0\rangle$  mit  $x_0 = 0$ . Zum Beispiel

$$|110\rangle.$$

Dann lautet der neue Zustand

$$\frac{1}{\sqrt{2}}(|110\rangle + |111\rangle)|110\rangle.$$

Wir lassen ab jetzt das zweite Register weg. Dessen Zustand ändert sich nicht mehr. Schritt 5 erbringt

$$\frac{1}{\sqrt{8}} \left( \frac{1}{\sqrt{2}} ((-1)^{(110) \cdot (000)^T} + (-1)^{(111) \cdot (000)^T}) |000\rangle + \dots \right).$$

Berechnet man mehr Summanden dieser Superposition, erkennt man: nur Summanden  $|z\rangle$  mit  $z \cdot (001)^T = 0$  haben Amplitude ungleich 0. Damit hat das Ergebnis der letzten Messung diese Eigenschaft.

Nun haben wir zwei linear unabhängige Lösungen erhalten, etwa

Schritt 7

$$z_1 = (1, 1, 0)^T, z_2 = (0, 1, 0)^T.$$

Das Gleichungssystem lautet

$$t_2 \oplus t_1 = 0, t_1 = 0.$$

Für die Lösung gilt  $t_2 = t_1 = 0$ .  $t_0$  ist nicht festgelegt, aber nur  $t_0 = 1$  unterscheidet vom Nullvektor. Der Test

$$f(0, 0, 0) = f(0, 0, 1)?$$

ergibt:  $f$  hat die Periode  $(0, 0, 1)^T$ .

◇

**Intuition:** Zwei Register befinden sich im verschränkten Zustand

Intuition

$$\sum_x |x\rangle |f(x)\rangle,$$

wobei  $f$  periodisch ist:  $f(x \oplus s) = f(x)$ . Dabei fallen die Funktionswerte gemäß der Periode zusammen. Messen wir das zweite Register, geht das erste in

$$\frac{1}{\sqrt{2}}(|\hat{x}\rangle + |\hat{x} \oplus s\rangle)$$

über. Die Periode  $s$  ist Teil dieses Zustands, allerdings nur implizit: wir müssen den störenden Einfluss des zufälligen Vektors  $\hat{x}$  beseitigen; dies gelingt mit einer Hadamard-Transformation. Diese Grundidee ist der Kern von Simons Algorithmus und leicht abgewandelt auch von Shors Algorithmus. Dort betrachten wir allerdings eine periodische Funktion in den ganzen Zahlen und verwenden statt der Hadamard- die Fouriertransformation.

Simons Algorithmus stammt aus dem Jahre 1994 und war das erste Beispiel für einen Quantenalgorithmus, der jedem klassischen deterministischen Verfahren für die gleiche Aufgabenstellung exponentiell überlegen ist.

## 8.6 Shors Algorithmus

Wir nähern uns dem Höhepunkt dieses Kapitels: mit den Vorbereitungen in den vorhergehenden Abschnitten, können wir Shors Algorithmus leicht verstehen. Das Resultat des Abschnitts 8.2 lautet: wir können eine Zahl  $n$  faktorisieren, wenn wir zu einer Zahl  $a$  kleiner  $n$  die Periode der Funktion

$$f(x) = a^x \bmod n$$

bestimmen können. Zudem haben wir gesehen, wie Simons Algorithmus die Periode einer Funktion mit Eingaben aus  $\{0, 1\}^n$  berechnen kann.

Die Funktion  $a^x \bmod n$  bildet die ganzen Zahlen  $\mathbb{Z}$  in die Menge  $\{0, \dots, n-1\}$  ab. Um die Funktion implementieren zu können, beschränken wir die

Eingabemenge auf  $\{0, \dots, N-1\}$  für eine Zweierpotenz  $N$  von der Größenordnung  $n^2$ . Auf diese Menge werden wir die Quanten-Fouriertransformation anwenden; wir wählen  $N$  quadratisch in  $n$ , um mit hoher Wahrscheinlichkeit ein verwertbares Ergebnis zu erhalten.

### Shors Algorithmus zur Suche nach Perioden, Quantenteil

*Eingabe:* Eine Funktion  $f: \{0, \dots, N-1\} \rightarrow \{0, \dots, n-1\}$  mit Periode  $p$ :

$$f(x+p) = f(x) \text{ für alle } x \in \{0, \dots, N-1\}.$$

$f$  steht als Quantenorakel

$$U_f: |a\rangle|b\rangle \mapsto |a\rangle|b \oplus f(a)\rangle$$

zur Verfügung. Für die Faktorisierung ist  $f(x)$  von der Form  $a^x \bmod n$ .

*Ausgabe:*

Im Fall  $p$  teilt  $N$ : Ein ganzzahliges Vielfaches  $y$  von  $N/p$ ;  $y = jN/p$ .

Anderenfalls: Eine Zahl nahe einem ganzzahligen Vielfachen von  $N/p$ .

Wir rechnen mit zwei Quantenregistern  $|a\rangle|b\rangle$ , aus  $\log N$  beziehungsweise  $\log n$  Bits. Wir konstruieren das Orakel so, dass  $N \approx n^2$  gilt; dann ist die Zahl der Quantenbits  $3 \log N$

1.  $R = |a\rangle|b\rangle \leftarrow |0 \dots 0\rangle|0 \dots 0\rangle$
2. Wende die Hadamard-Transformation auf  $|a\rangle$  an:  
 $R \leftarrow \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle|0 \dots 0\rangle$
3. Wende das Orakel an:  
 $R \leftarrow U_f R = \frac{1}{\sqrt{N}} \sum_{x=0, \dots, N-1} |x\rangle|f(x)\rangle$
4. Miss Register  $|b\rangle$
5. Wende die Quanten-Fouriertransformation auf  $|a\rangle$  an:  
 $|a\rangle \leftarrow \text{QFT}_N |a\rangle$
6. Miss  $|a\rangle$  und gib den Inhalt aus.

Vergleichen wir diesen Algorithmus mit Simons auf Seite 218, wird die gemeinsame Struktur erkennbar. Bei der Analyse können wir von Beobachtungen des letzten Abschnitts profitieren. Die Messung in Schritt 4 ist für das Ergebnis ohne Belang, vereinfacht jedoch die Analyse.

Der Schaltkreis in Abbildung 8.6 implementiert den eben beschriebenen Algorithmus. Die Quanten-Fouriertransformation lässt sich mit  $O(n^2)$  Gattern realisieren, siehe Abbildung 8.4.

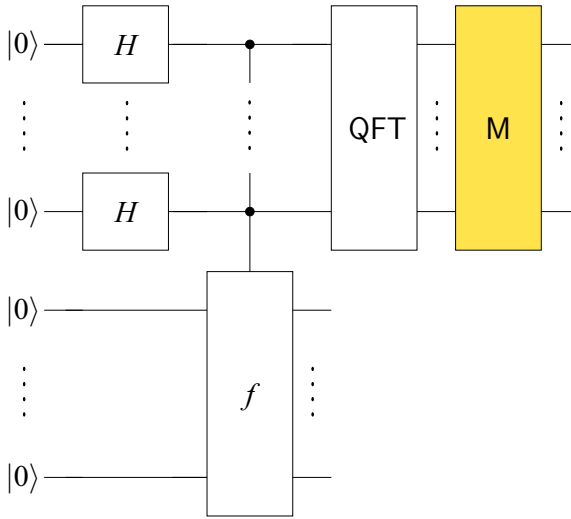


Abbildung 8.6: Schaltkreis für Shors Algorithmus.

Laut Spezifikation ist das Ergebnis des Algorithmus nicht die Periode. Im Fall  $p$  teilt  $N$  lautet es  $y = jN/p$  für eine ganze Zahl  $j$ . Daraus lässt sich die Periode  $p$  mit einem effizienten klassischen Verfahren bestimmen. Zur Erinnerung: auch bei Simons Algorithmus ist eine klassische Nachbearbeitung nötig: Schritte 7 und 8 auf Seite 219.

Nutzen des Ergebnisses

Wir kennen  $y$  und  $N$ , und es gilt

$$\frac{j}{p} = \frac{y}{N}. \quad (8.7)$$

Haben  $j$  und  $p$  keine gemeinsamen Teiler – das heißt der Bruch  $\frac{j}{p}$  ist gekürzt – können wir  $j$  und, weitaus interessanter  $p$ , durch Kürzen des Bruches  $y/N$  bestimmen. Das wird am Ende des folgenden Beispiels durchgeführt.

**Beispiel 8.6:** Shors Algorithmus wurde im Jahr 2000 auf einem experimentellen Quantenrechner ausgeführt [145]. Damit gelang es, die Zahl 15 zu faktorisieren. Darum führen wir Beispiel 8.3 fort und betrachten die Arbeitsweise des eben geschilderten Algorithmus mit der Eingabe

$$f(x) = 7^x \bmod 15.$$

Die Zahl 7 ist unter den zu  $n = 15$  teilerfremden Zahlen aus  $\{2, \dots, 14\}$  beliebig gewählt (siehe Beispiel 8.3). Wir legen  $N = 16$  fest. Dann führen die Schritte 1 und 2 zu dem Zustand

$$\frac{1}{4}(|0\rangle|0\rangle + |1\rangle|0\rangle + \dots + |15\rangle|0\rangle).$$



Das Orakel erzeugt

$$\begin{aligned} & \frac{1}{4} \sum_{x=0}^{15} |x\rangle |7^x \bmod 15\rangle \\ &= \frac{1}{4} (|0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + |4\rangle|1\rangle + |5\rangle|7\rangle + \dots \end{aligned}$$

Wir sehen an dieser Stelle: die Periode  $p$  lautet 4, da  $f(4) = f(0)$  gilt und weiter

$$f(5) = 7 \cdot f(0) = f(1).$$

Wir geben dem Zustand eine Form, die das Ergebnis der ersten Messung erkennbar macht:

$$\begin{aligned} & \frac{1}{4} ( (|0\rangle + |4\rangle + |8\rangle + |12\rangle) |1\rangle + \\ & (|1\rangle + |5\rangle + |9\rangle + |13\rangle) |7\rangle + \\ & (|2\rangle + |6\rangle + |10\rangle + |14\rangle) |4\rangle + \\ & (|3\rangle + |7\rangle + |11\rangle + |15\rangle) |13\rangle ). \end{aligned}$$

Schritt 4

Die Messung in Schritt 4 ergibt als Ergebnis eine der (normalisierten) Zeilen der vorstehenden Superposition. So etwa

$$\frac{1}{2} (|1\rangle + |5\rangle + |9\rangle + |13\rangle) |7\rangle.$$

Die Periode ist dem Zustand anzusehen. Messen wir jedoch, erhalten wir keinen Aufschluss über  $p$ .

Schritt 5

Darum wird die Fouriertransformation ausgeführt, und zwar mit dem Ergebnis:

$$\text{QFT}_{16} \frac{1}{2} (|1\rangle + |5\rangle + |9\rangle + |13\rangle) = \frac{1}{2} (|0\rangle + i|4\rangle - |8\rangle - i|12\rangle).$$

Das einzusehen, erfordert eine etwas anstrengende Rechnung, die uns nach der allgemeinen Analyse leichter fällt (Aufgabe 8.11). Wir sehen jedoch, es handelt sich um eine Superposition von Vielfachen der Periode 4. Ein gleichwertiges Ergebnis erhalten wir, wenn Schritt 4 ein anderes Ergebnis hatte. So ist

$$\begin{aligned} \text{QFT}_{16} \frac{1}{2} (|0\rangle + |4\rangle + |8\rangle + |12\rangle) &= \frac{1}{2} (|0\rangle + |4\rangle + |8\rangle + |12\rangle), \\ \text{QFT}_{16} \frac{1}{2} (|2\rangle + |6\rangle + |10\rangle + |14\rangle) &= \frac{1}{2} (|0\rangle - |4\rangle + |8\rangle - |12\rangle) \text{ und} \\ \text{QFT}_{16} \frac{1}{2} (|3\rangle + |7\rangle + |11\rangle + |15\rangle) &= \frac{1}{2} (|0\rangle - i|4\rangle - |8\rangle + i|12\rangle). \end{aligned}$$

Auf diese Weise macht die QFT die Periode zugänglich.

Die abschließende Messung ergibt einen Wert  $y$  aus

Schritt 6

$$\{0, 4, 8, 12\} = \{jN/p \mid 0 \leq j \leq 3\},$$

in der Bezeichnungsweise aus Gleichung (8.7).

Nun können wir das Ergebnis bestimmen, wenn  $j$  und  $p$  keine gemeinsamen Teiler haben. 1 und 3 sind mit  $p = 4$  teilerfremd. Ist das Ergebnis der letzten Messung  $y = 1 \cdot 4$ , setzen wir an:

$$\frac{j}{p} = \frac{4}{16} = \frac{1}{4}.$$

Der gekürzte Wert ergibt  $p = 4$ . Ist das Ergebnis  $3 \cdot 4 = 12$ , ergibt sich

$$\frac{j}{p} = \frac{12}{16} = \frac{3}{4}.$$

Bei den Ergebnissen 0 und 8 können wir die Periode nicht korrekt bestimmen. Ein Test klärt uns über den Fehler auf. Allerdings müssen wir noch die Frage klären: Wie hoch ist die Fehlerwahrscheinlichkeit?

Aus der Zahlentheorie ist folgende Aussage bekannt.

Für eine große Zahl  $m$  ist ein Anteil der Größenordnung

$$\Omega(1/\log \log m)$$

der Zahlen  $1, \dots, m-1$  teilerfremd mit  $m$ .

Also ist es ziemlich wahrscheinlich, dass wir aus  $jN/p$  die Periode  $p$  bestimmen können. Nun werden wir zeigen, dass der Algorithmus auf Seite 222 tatsächlich ein Vielfaches von  $N/p$  ergibt.

### Analyse

Das Ergebnis der ersten drei Schritte ist

Schritt 4

$$|\phi_3\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle |f(x)\rangle.$$

Wir beginnen die Analyse mit der Messung des ersten Registers im vierten Schritt. Register  $|b\rangle$  geht in den Zustand  $|f(x)\rangle$  für ein mit Wahrscheinlichkeit  $1/N$  gewähltes  $x$  über. Da die Register in einem verschränkten Zustand waren, wird  $|a\rangle$  beeinflusst; es geht in eine Superposition aller Urbilder von  $f(x)$  über. Da  $f$  periodisch ist, sind diese von der Form  $|\hat{x}\rangle, |\hat{x}+p\rangle, |\hat{x}+2p\rangle$  und so fort. Die Anzahl der Urbilder bezeichnen wir mit  $A$ .

Wir führen die Analyse zunächst für den Fall durch, dass  $N$  ein Vielfaches der Periode  $p$  ist. In diesem Fall ist  $A = N/p$ . Da  $f$  periodisch ist, hat  $R$  nach Schritt 4 einen Zustand der Form

Fall  $p$  teilt  $N$

$$|\phi_4\rangle = \frac{1}{\sqrt{A}} (|\hat{x}\rangle + |\hat{x}+p\rangle + \dots + |\hat{x}+(A-1)p\rangle) |f(\hat{x})\rangle.$$

Wobei wir der Lesbarkeit halber ignoriert haben, dass wir modulo  $N$  rechnen. Korrekter ist

$$|\phi_4\rangle = \frac{1}{\sqrt{A}} \sum_{j=0}^{A-1} |\hat{x} + j \cdot p \bmod N\rangle |f(\hat{x})\rangle.$$

$|\hat{x}\rangle$  könnte dabei jede ganze Zahl sein, da bei der Messung des zweiten Registers jedes Element aus  $\{0, \dots, N-1\}$  das Urbild von  $f(\hat{x})$  sein könnte. Würden wir an dieser Stelle auch das erste Register messen, wäre dem Ergebnis nichts zu entnehmen.

Schritt 5

Darum ist das Ziel von Schritt 5, den störenden Offset  $\hat{x}$  aus  $|\phi_4\rangle$  zu entfernen. Die Anwendung der QFT bildet

$$|\hat{x} + j \cdot p\rangle$$

auf

$$\frac{1}{\sqrt{N}} \sum_y \omega_N^{(\hat{x}+j \cdot p) \cdot y} |y\rangle$$

ab. Der Nachfolgezustand von Schritt 5 lautet also

$$\begin{aligned} |\phi_5\rangle &= \frac{1}{\sqrt{NA}} \sum_{y=0}^{N-1} \sum_{j=0}^{A-1} \omega_N^{(\hat{x}+j \cdot p) \cdot y} |y\rangle |f(\hat{x})\rangle \\ &= \frac{1}{\sqrt{NA}} \sum_{y=0}^{N-1} \omega_N^{\hat{x} \cdot y} \sum_{j=0}^{A-1} \omega_N^{j \cdot p \cdot y} |y\rangle |f(\hat{x})\rangle. \end{aligned}$$

Das heißt,

$$\omega_N^{\hat{x} \cdot y}$$

spielt als Faktor keine Rolle für die Wahrscheinlichkeit von  $|y\rangle$ , sondern beeinflusst als Vorfaktor der Summe über  $j$  nur die Phase! Die Wahrscheinlichkeit von  $|y\rangle$  ist gerade

$$P(y) = \frac{1}{NA} \left| \omega_N^0 + \omega_N^{py} + \dots + \omega_N^{(A-1)py} \right|^2$$

und von dem störenden Offset  $\hat{x}$  unabhängig. Da wir  $N$  als ein Vielfaches der Periode  $p$  angenommen haben – damit ist  $A = N/p$  eine ganze Zahl – gilt

$$\omega_N^{jpy} = \omega_A^{jy},$$

und die Wahrscheinlichkeit  $P(y)$  ergibt sich aus einer Summe über  $A$ -te Einheitswurzeln.

$$P(y) = \frac{1}{NA} \left| \omega_A^0 + \omega_A^y + \dots + \omega_A^{(A-1)y} \right|^2.$$

Wir untersuchen  $P(y)$  für die zwei Fälle, in denen  $y$  ein Vielfaches von  $A$  ist beziehungsweise nicht ist.

Ist  $y$  ein Vielfaches von  $A$  – es gelte  $y = kA$  – wird die  $A$ -te Einheitswurzel auf 1 abgebildet: Fall 1

$$\omega_A^{jy} = (\omega_A^A)^{kj} = 1.$$

Es folgt

$$P(y) = \frac{1}{NA} A^2 = \frac{A}{N} = \frac{1}{p}$$

Anderenfalls können wir das Summationslemma anwenden (Seite 211): Wir erhalten  $P(y) = 0$ , falls  $A$  kein Teiler von  $y$  ist. Anschaulich ist Fall 2

$$\omega_N^0 + \omega_N^{py} + \dots + \omega_N^{(A-1)py}$$

eine Summe von gleichmäßig über den Einheitskreis verteilten  $N$ -ten Einheitswurzeln. Diese addieren sich zu 0.

Wir folgern: Im Fall  $p$  teilt  $N$  arbeitet der Algorithmus korrekt: eine Messung liefert ein Vielfaches von  $A = N/p$ .

**Aufgabe 8.11:** Zeigen Sie, dass wie in Beispiel 8.6 behauptet

$$\text{QFT}_{16}(|1\rangle + |5\rangle + |9\rangle + |13\rangle) = \frac{1}{4}(|0\rangle + i|4\rangle - |8\rangle - i|12\rangle)$$

gilt.

**Der Fall:  $N$  ist kein Vielfaches von  $p$**

Allerdings können wir nicht sicherstellen, dass  $N$  ein Vielfaches der unbekannten Periode  $p$  ist.  $A$  – die Anzahl der Urbilder des Ergebnisses der Messung am zweiten Register – ist dann die größte ganze Zahl kleiner  $N/p$  oder die kleinste ganze Zahl größer  $N/p$ .

Der allgemeine Fall

**Beispiel 8.7:** Ein Beispiel für den Fall  $N = 10$  und  $p = 4$  liefert eine Funktion  $f$  mit

$$f(0) = f(4) = f(8), f(1) = f(5) = f(9) \text{ und } f(2) = f(6).$$

Wir sehen: die Anzahl der Urbilder eines Wertes  $f(\hat{x})$ ,  $\hat{x} \in \{0, \dots, 9\}$  ist 2 oder 3.

$\omega_N^{jpy}$  lässt sich in diesem Fall nicht als  $A$ -te Einheitswurzel ausdrücken. Allerdings ist  $A$  nicht weit von  $N/p$  entfernt, und auch in diesem Fall bevorzugt die Superposition  $|\phi_5\rangle$  jene Werte  $y$ , die nahe bei einem Vielfachen von  $N/p$  liegen. Die Analyse bedeutet einigen technischen Aufwand; die Summe

$$\sum_{j=0}^{N-1} \omega_N^{jpy}$$

wird über die Formel für die geometrische Reihe (siehe Aufgabe 8.6) bestimmt und abgeschätzt, was wir uns hier ersparen.

Wiederholte  
Kettenbrüche

Wie lässt sich aus dem Ergebnis  $y$  die Periode  $p$  bestimmen, wenn sie den Wert  $N$  nicht teilt? Wie in Beispiel 8.6 den Term  $N/y$  zu kürzen, ist nicht möglich. Da wir  $N$  – die Dimension der Quanten-Fouriertransformation – von der Größenordnung  $\Theta(n^2)$  gewählt haben, liegt das Ergebnis  $y$  so nah bei einem Vielfachen von  $N/p$ , dass wir mit einem Verfahren namens *wiederholte Kettenbrüche* (engl. *continued fractions*) die Periode  $p$  mit hoher Wahrscheinlichkeit ermitteln können, siehe zum Beispiel [114].

### Abschließende Ermittlung der Periode

Das Ergebnis von Schritt 6 ist die Zahl  $y$ , nahe bei einem Vielfachen von  $N/p$ .

Abschluss von  
Shors  
Algorithmus

7. Ermittle aus  $y$  und  $N$  mit Hilfe wiederholter Kettenbrüche die Periode  $p$ .

### Die Laufzeit

Im wesentlichen besteht Shors Quantenalgorithmus aus drei Schritten, siehe Abbildung 8.6:

1. der Hadamard-Transformation,
2. der Anwendung des Orakels für  $a^x \bmod n$  zur Faktorisierung von  $n$ ,
3. der Quanten-Fouriertransformation.

Schritte 1 und 2 werden auf  $2 \cdot \log n$  Quantenbits ausgeführt, Schritt 3 auf  $\log n$  Bits. Damit ist die Hadamard-Transformation mit  $2 \cdot \log n$  und die Quanten-Fouriertransformation mit  $O((\log n)^2)$  Gattern realisierbar, siehe Abschnitt 8.4. Am aufwändigsten ist Schritt 2: es gibt klassische Schaltkreise, die modulare Potenzen mit  $O((\log n)^3)$  Gattern realisieren.

Damit ist der Quantenalgorithmus für Shors Algorithmus mit  $O((\log n)^3)$  Gattern realisierbar. Wir haben allerdings die konkrete Berechnung der Periode mit Hilfe von Kettenbrüchen noch nicht betrachtet. Diese ist in Zeit  $O((\log n)^3)$  ausführbar, führt auf das Ergebnis des Quantenalgorithmus angewandt jedoch nicht immer zum Erfolg: Die Erfolgswahrscheinlichkeit ist etwa  $1/(20 \cdot \log \log n)$ , hier spielt die zahlentheoretische Aussage von Seite 225 eine große Rolle. Das bedeutet: nach  $O(\log \log n)$  Wiederholungen ergibt sich mit hoher Wahrscheinlichkeit die Periode. Es gilt:

Shors Quantenalgorithmus zur Bestimmung der Periode von  $a^x \bmod n$  hat erwartete Laufzeit

$$O(\log \log n \cdot (\log n)^3).$$

Nun verbinden wir dieses Ergebnis mit dem Resultat aus Abschnitt 8.2. Der klassische Teil von Shors Algorithmus benötigt  $O(\log n)$  Multiplikationen und hat die Erfolgswahrscheinlichkeit  $1/2$ .

Shors Algorithmus ermittelt mit erwarteter Laufzeit

$$O(\log \log n \cdot (\log n)^3) = O((\log n)^4)$$

einen echten Teiler einer ganzen Zahl  $n$ .

*Bemerkung:* Wird das beste bekannte Multiplikationsverfahren verwendet, ist die erwartete Laufzeit  $O((\log n)^2 \cdot \log \log n \cdot \log \log \log n)$ , siehe Seite 204.

# 8.7 Jenseits von Shor

Shors Algorithmus verwendet die Quanten-Fouriertransformation, um ein Problem effizient zu lösen, für das keine klassischen Polynomialzeitalgorithmen bekannt sind. Dieser Abschnitt beschreibt Problemstellungen, auf die Shors Grundeinsicht übertragen werden kann.

## Der diskrete Logarithmus

Schon in Shors Ursprungsarbeit aus dem Jahr 1994 wird neben dem eben beschriebenen Faktorisierungsalgorithmus ein zweites Quantenrechenverfahren beschrieben: mit einer ähnlichen Methode können Quantenschaltkreise polynomialer Größe den *diskreten Logarithmus* berechnen.

Ist  $p$  eine Primzahl und  $a$  eine positive Zahl mit  $a \leq p-1$ , bezeichnen wir die Funktion  $\{0, \dots, p-1\} \rightarrow \{0, \dots, p-1\}$ , Problemstellung

$$x \mapsto a^x \bmod p$$

als *diskrete Exponentialfunktion zur Basis  $a$* .

Wir haben diese Funktion bereits kennen gelernt: auf Seite 196 wird ein Verfahren beschrieben, das diese Funktion in Zeit  $O((\log p)^3)$  berechnet. Schwieriger ist es, die Umkehrfunktion zu berechnen, zu einer Zahl  $y$ , den Wert  $x$  zu bestimmen, für den  $a^x \bmod p = y$  gilt. Da  $p$  eine Primzahl ist, ist dieser Wert  $x$  eindeutig bestimmt. Wir definieren die *diskrete Logarithmusfunktion* als

$$\text{dlog}_{p,a}(y) = x \text{ mit } a^x \bmod p = y.$$

Der bekannte reelle Logarithmus  $\log_b y$  ist effizient berechenbar, da die reelle Exponentialfunktion  $b^x$  stetig ist. Dagegen macht die diskrete Exponentialfunktion unvorhersehbare „Sprünge“. Das verdeutlicht folgende Tabelle anhand einiger Werte der Funktion  $5^x \bmod 101$ :

$x$	0	1	2	3	4	5	6	7	...
$5^x \bmod 101$	1	5	25	24	19	95	71	52	...

Man kennt keinen effizienten klassischen Algorithmus für den diskreten Logarithmus. Darum spielt diese Funktion in der Kryptographie eine ähnliche Rolle wie die Faktorisierung: Die Exponentialfunktion ist effizient berechenbar, für die Umkehrfunktion ist das nicht bekannt. Das kann für ein Public Key-Verschlüsselungsverfahren ähnlich der RSA-Kryptographie ausgenutzt werden. In [135] beschreibt Shor einen auf der Quanten-Fouriertransformation beruhenden Algorithmus, der den diskreten Logarithmus effizient berechnet.

**Aufgabe 8.12:** Arbeiten Sie die Ähnlichkeit der Suche nach einer Periode und dem diskreten Logarithmus heraus.

### Phasenbestimmung

Quantenschaltkreise realisieren unitäre Transformationen. Die Quanten-Fouriertransformation erlaubt es, eine Eigenschaft unitärer Transformationen zu berechnen: Man kann die Phase eines Eigenwertes zu einem gegebenen Eigenvektor bestimmen. Diese etwas abstrakte Problemstellung werden wir gleich erläutern. Vorausgeschickt sei bereits: dadurch wird Shors Algorithmus verallgemeinert, da die Phasenbestimmung zur effizienten Faktorisierung anwendbar ist.

Eigenwert

Für eine  $n \times n$ -Matrix  $M$  heißt eine reelle Zahl  $\lambda$  *Eigenwert* von  $M$ , falls ein Vektor  $x \neq 0$  existiert, mit  $Mx = \lambda x$ . Der Vektor  $x$  ist der zu  $\lambda$  gehörende *Eigenvektor*. Wir betrachten folgende Problemstellung:

**Eingabe:** Eine unitäre Transformation  $U$  und ein Eigenvektor  $|u\rangle$  von  $U$ .

**Ausgabe:** Die Phase  $\phi$  des Eigenwertes  $\lambda$  mit  $U|u\rangle = \lambda|u\rangle$ .

Lautet der Eigenwert  $\lambda = e^{2\pi i \phi}$ , so ist der Winkel  $2\pi\phi$  oder damit gleichwertig der Wert  $\phi$  gefragt.  $U$  wird als eine Folge von gesteuerten Quantenbausteinen für die Transformationen  $U^{2^0}, U^{2^1}, U^{2^2}, \dots$  verwendet. Wir deuten an, wie sich die Eigenwertbestimmung zur Faktorisierung verwenden lässt.

Phasen-  
bestimmung und  
Faktorisierung

Shors Algorithmus findet die Periode einer Funktion der Form  $a^x \bmod n$ , mit  $\text{ggT}(a, n) = 1$ . Betrachten wir die Transformation

$$U : |x\rangle \mapsto |x \cdot a \bmod n\rangle, x = 0, \dots, n-1,$$

so gilt

$$U^p = I.$$

Daraus folgt, dass die Eigenwerte von  $U$  die  $p$ -ten Einheitswurzeln sind. Außerdem gilt, dass die Superposition *aller* Eigenvektoren von  $U$  der Vektor  $|1\rangle$  ist. Damit kann man sich des Problems entledigen, dass dem Algorithmus zur Phasenbestimmung ein Eigenvektor übergeben werden muss. Da die Einheitswurzeln

$$e^{\frac{2\pi i}{p} k}, k = 0, \dots, p-1$$

lauten, lässt sich aus der Phase die Periode bestimmen: aus einem Vielfachen von  $1/p$  ist  $p$  berechenbar. Für Details verweisen wir auf [83].

## Die allgemeine Fouriertransformation

Die Algorithmen von Simon und Shor sind sich so ähnlich, dass sich die Frage nach der gemeinsamen Struktur stellt. Der Quantenteil unterscheidet sich in einem einzigen Punkt: Simons Algorithmus verwendet die Hadamard-Transformation, die auf der Menge  $\{0,1\}^n$  definiert ist, und Shors Verfahren operiert mit der Quanten-Fouriertransformation auf der Menge der ganzen Zahlen  $\mathbb{Z}$ . Die Problemstellung und die verwendete Transformation können so verallgemeinert werden, dass die Algorithmen von Shor und Simon als Spezialfälle eines allgemeinen algorithmischen Ansatzes erscheinen.

Wir betrachten eine Menge  $G$ , auf der wir eine abstrakte Operation definieren: sind  $a$  und  $b$  Elemente aus  $G$ , so ist auch  $a \diamond b$  aus  $G$ . Eine solche Struktur nennt man eine *Gruppe*, wenn folgendes gilt:

Definition  
Gruppe

1.  $a \diamond (b \diamond c) = (a \diamond b) \diamond c$ ,
2. es gibt ein neutrales Element  $e$  in  $G$  mit  $e \diamond a = a$  für alle  $a \in G$ , und
3. zu jedem  $a \in G$  gibt es ein Inverses  $a^{-1}$  in  $G$  mit  $a \diamond a^{-1} = e$ .

**Aufgabe 8.13:** Zeigen Sie:  $\mathbb{Z}$  mit der ganzzahligen Addition  $+$  ist eine Gruppe und ebenso  $\{0,1\}^n$  mit der komponentenweisen Addition  $\oplus$ .

Eine Gruppe heißt *abelsch*<sup>1</sup> oder *kommutativ*, wenn für das Ergebnis ihrer Operation  $\diamond$  die Reihenfolge der Operanden keine Rolle spielt:

Abelsch

$$a \diamond b = b \diamond a \text{ für alle } a, b \in G.$$

Die Gruppen aus Aufgabe 8.13 sind abelsch, ebenso wie alle weiteren in diesem Abschnitt betrachteten Gruppen.

Eine Teilmenge von  $G$  heißt *Untergruppe*, wenn diese selbst eine Gruppe ist. Eine solche Untergruppe entsteht zum Beispiel, wenn wir eine Funktion  $f$  betrachten, die Gruppenelemente in eine beliebige Menge  $X$  abbildet und definieren:

$$K = \{k \in G \mid f(g \diamond k) = f(g) \text{ für alle } g \in G\}.$$

Die Menge  $K$  ist eine Untergruppe von  $G$ , die *symmetrische Gruppe* von  $f$  genannt wird.

<sup>1</sup>Benannt nach dem dänischen Mathematiker Niels Henrik Abel, 1802–1829.



Aufgabenstellung

Unsere Aufgabe ist die folgende:

**Eingabe:**  $G$  und  $f$

**Ausgabe:**  $K$

Die symmetrische Gruppe  $K$  beschreibt die Periodizität der Funktion  $f$ . Man nennt die beschriebene Aufgabe auch Suche nach der *hidden subgroup*.

### Beispiel 8.8:

1. Ist  $G = \{0, 1\}^n$  mit der Addition  $\oplus$ ,  $X$  beliebig und  $f$  eine Funktion mit  $f(x \oplus s) = f(x)$ , so ist  $K = \{0, s\}$ .
2. Ist  $G = \mathbb{Z}$  mit der ganzzahligen Addition,  $X$  beliebig und  $f$  eine Funktion mit  $f(x \oplus p) = f(x)$ , so ist  $K = \{0, p, 2p, \dots\}$ .
3. Ist  $G = \{0, 1\}$  mit  $\oplus$  und  $X = \{0, 1\}$ , so gilt folgendes: Ist  $f$  eine konstante Funktion ( $f(0) = f(1)$ ), so gilt  $K = \{0, 1\}$ . Ist  $f$  eine balancierte Funktion ( $f(0) = f(1) \oplus 1$ ), so gilt  $K = \{0\}$ .  
Also können wir das *Problem von Deutsch* dadurch lösen, dass wir die entsprechende symmetrische Gruppe bestimmen.
4. Auch die Aufgabe, den diskreten Logarithmus zu bestimmen, lässt sich als die Suche nach der symmetrischen Gruppe einer Funktion formulieren.

Wie lässt sich die Untergruppe  $K$  bestimmen? Wir leiten eine Transformation ab, die für die abelsche Gruppe  $G$  eine ähnliche Wirkung hat, wie die Hadamard-Transformation für  $\{0, 1\}^n$  und die Quanten-Fouriertransformation für  $\mathbb{Z}$ .

Erzeugen wir eine Superposition über alle Gruppenelemente und wenden  $f$  darauf an, erhalten wir den Zwei-Register-Zustand

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f(g)\rangle.$$

Wir setzen voraus:  $f(g) = f(h)$  dann und nur dann, wenn  $g \diamond h^{-1}$  Element der Untergruppe  $K$  ist. Dann ergibt eine Messung an dem zweiten Register den Folgezustand

$$\frac{1}{\sqrt{|K|}} \sum_{k \in K} |\hat{g} \diamond k\rangle |f(\hat{g})\rangle, \quad (8.8)$$

für ein beliebiges Gruppenelement  $\hat{g}$ . Dies ist die gleiche Situation wie bei den Algorithmen von Simon und Shor. Wieder gibt es einen störenden Offset  $\hat{g}$ . Diesen entfernen wir mit einer zu der Gruppe  $G$  passenden Transformation.

Dazu definieren wir eine Basis  $|b_1\rangle, \dots, |b_m\rangle$ , die sich bezüglich  $\diamond$  in folgendem Sinne invariant verhält:

$$|g \diamond b_j\rangle = e^{i\phi(g,j)} |b_j\rangle, \text{ für alle } g \in G.$$

Nur die Phase ändert sich. Geben wir Gleichung (8.8) in dieser Basis an, wandert der Offset  $\hat{g}$  in die Phase. Die *Fouriertransformation über der Gruppe*  $G$  transformiert nun diese Basis in die Standardbasis.

Fouriertransformation über einer Gruppe

**Beispiel 8.9:**

1. Wir betrachten die Basis  $|b_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ ,  $|b_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  über der Gruppe  $\{0, 1\}, \oplus$ . Es gilt

$$|b_0 \oplus 1\rangle = \frac{1}{\sqrt{2}}(|0 \oplus 1\rangle + |1 \oplus 1\rangle) = |b_0\rangle, \quad (8.9)$$

$$|b_1 \oplus 1\rangle = \frac{1}{\sqrt{2}}(|0 \oplus 1\rangle - |1 \oplus 1\rangle) = -|b_1\rangle. \quad (8.10)$$

2. Wir betrachten  $G = \mathbb{Z}_N$  und die Basis  $|b_0\rangle, \dots, |b_{N-1}\rangle$  mit

$$|b_j\rangle = \omega_N^{j \cdot 0}|0\rangle + \omega_N^{j \cdot 1}|1\rangle + \omega_N^{j \cdot 2}|2\rangle + \dots + \omega_N^{j \cdot (N-1)}|N-1\rangle.$$

Addieren wir ein Element  $\hat{x} \in G$ , erhalten wir

$$\begin{aligned} |b_j + \hat{x}\rangle &= \omega_N^{j \cdot 0}|\hat{x}\rangle + \omega_N^{j \cdot 1}|1 + \hat{x}\rangle + \omega_N^{j \cdot 2}|2 + \hat{x}\rangle + \dots + \omega_N^{j \cdot (N-1)}|N-1 + \hat{x}\rangle \\ &= \omega_N^{-\hat{x}}(\omega_N^{j \cdot 0}|0\rangle + \omega_N^{j \cdot 1}|1\rangle + \omega_N^{j \cdot 2}|2\rangle + \dots + \omega_N^{j \cdot (N-1)}|N-1\rangle). \end{aligned}$$

Dem entspricht die Definition der Quanten-Fouriertransformation

$$\text{QFT}_N |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{j \cdot k} |k\rangle$$

mit

$$\text{QFT}_N |j + \hat{x}\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{\hat{x} \cdot k} \omega_N^{j \cdot k} |k\rangle.$$

◇

Die Problemstellung der Algorithmen von Shor und Simon lässt sich als die Suche nach einer bestimmten Untergruppe auffassen. Um dieses Problem allgemein anzugehen, wird eine Transformation definiert, die in den Spezialfällen „Shor“ und „Simon“ der Quanten-Fouriertransformation beziehungsweise der Hadamard-Transformation entsprechen. Details findet der Leser in [83].

Zusammenfassung

**Die Pellische Gleichung**

In einer alten Chronik soll über die Truppen des Sachsenkönigs Harald, der 1066 in der Schlacht von Hastings auf die Normannen stieß, zu lesen sein (zitiert nach [139]): „Haralds Mannen standen tapfer zusammen und bildeten 61 Quadrate mit gleich vielen Recken in jedem Quadrat ... Als Harald sich in die Schlacht warf, bildeten die Sachsen mit ihm zusammen ein einziges mächtiges Quadrat von Männern.“

Nun lautet die Frage: wie groß ist Haralds Armee mindestens, damit diese Bedingung erfüllt werden kann? Dazu ist die kleinste ganzzahlige Lösung der Gleichung

$$y^2 = 61x^2 + 1$$

gesucht. Eine Lösung  $(x_0, y_0)$  definieren wir als die kleinste, wenn sie  $x_0 + y_0\sqrt{d}$  minimiert. Gleichungen dieses Typs sind nach dem englischen Mathematiker John Pell (1610–1685) benannt, der allerdings selbst nichts zu diesem Gebiet beigetragen hat; solche Fehlbenennungen sind in der Mathematik nicht selten. Die Theorie der Gleichung

$$y^2 = d \cdot x^2 + 1$$

geht im wesentlichen auf Fermat, Euler und Joseph Louis Lagrange (1736–1813) zurück, wobei bereits der antike Mathematiker Diophant Lösungen für  $d = 26$  und  $30$  gefunden hat. Man setzt  $d$  als quadratfrei voraus. Damit ist es sinnvoll, den Wert  $A = x_0 + y_0\sqrt{d}$  als Ausgabe zu fordern, da sich dann aus  $A$  alle Lösungen eindeutig bestimmen lassen. Allerdings kann selbst die kleinste Lösung exponentiell in der Eingabegröße  $\log d$  sein, so dass sich diese nicht in Polynomialzeit aufschreiben lässt. Darum berechnet man statt dessen eine andere zahlentheoretische Größe: den *Regulator*  $\ln A$ , den wir mit einer festgelegten Genauigkeit annähern wollen. Der beste bekannte klassische Algorithmus für diese Aufgabe ist nicht polynomial in  $\log d$ .

Hallgrens  
Algorithmus

Sean Hallgren gibt in [73] einen Quantenalgorithmus an, der das Problem in Polynomialzeit löst. Dazu wird das Problem, die Pellsche Gleichung zu lösen, auf die Suche nach der Periode einer reellwertigen Funktion reduziert. Hallgren zeigt, wie sich die Quanten-Fouriertransformation in dieser Situation anwenden lässt. [85] bietet eine Einführung für Leser, die sich mit algebraischer Zahlentheorie nicht gut auskennen.

Weitere  
Algorithmen

Der Leser könnte sich nun fragen, ob er sich für so spezielle Fragestellungen wie die Lösung der Pellschen Gleichung interessieren sollte. Eine Erkenntnis aus der Theorie der NP-Vollständigkeit (siehe Abschnitt 4.3) ist, dass verschiedene schwierige Probleme oft durch effiziente Transformationen, den sogenannten Reduktionen, miteinander verbunden sind. Damit können effiziente Algorithmen für ein schwieriges Problem zu effizienten Verfahren für Fragestellungen führen, die auf den ersten Blick nichts mit dem Ausgangsproblem zu tun zu haben scheinen. Es gibt noch eine Reihe weiterer Probleme, für die Quantenalgorithmen bekannt sind, die klassischen Verfahren überlegen sind. Unter der Internetadresse [82] veröffentlicht Stephen Jordan eine sehr nützliche Übersicht dieser Probleme.

## 9 Fehler korrigieren

*Form follows failure.*

Henry Petrosky

Wer einen Quantencomputer bauen will, wird ohne Fehlerkorrektur nicht auskommen. Das werden wir in dem ersten Abschnitt sehen. Das Forschungsgebiet Quantenfehlerkorrektur entwickelt sich rasant, wie sich auch an den umfangreichen Monografien [48] und [102] zeigt. Deshalb kann es hier nur darum gehen, ein Grundverständnis für Probleme und Prinzipien von fehlerkorrigierenden Codes zu erwerben. Dazu werden wir Shors 9-Qubit-Code ausführlich herleiten und diskutieren. Die Arbeit mit den dabei auftretenden Codewörtern ist einfach, aber technisch etwas aufwändig. Darum wird der Leser aufgefordert, sich über Übungsaufgaben Details selbst zu erarbeiten. Tatsächlich ist es einfacher, die nötigen Schritte selbst auszuführen, als fertig vorbereiteten Ableitungen zu folgen. Das werden Sie sehen, wenn Sie sich die Musterlösungen ansehen. Das Kapitel schließt mit einem Ausblick auf weiterführende Aspekte dieses wichtigen und anspruchsvollen Themas.

### 9.1 Dekohärenz und Fehler auf Quantenbits

Wir kehren an den Anfang zurück: zu Schrödingers Katze. Solange diese in der Kiste isoliert ist, bleibt ihr Zustand unbestimmt. Es lässt sich nicht sagen, ob sie lebendig ist oder tot; ihr Befinden ist eine Superposition beider Zustände. Realisieren wir ein Quantenbit, ist eine solche dauerhafte Abschottung nicht möglich: wie wir sehen werden, ist für jedes Quantenobjekt die Wechselwirkung mit der natürlichen Umgebung relevant. Wir sollten ein solches daher als sogenanntes *offenes System* ansehen, genauer noch als ein offenes Teilsystem eines größeren Gesamtsystems, wobei auch letzteres den Gesetzen der Quantenmechanik unterworfen ist.

Wir untersuchen ein Quantenteilchen, das sich an zwei Orten befinden kann. Diese beiden Positionen bezeichnen wir als  $|L\rangle$  und als  $|R\rangle$  und ignorieren zunächst die Frage, was es mit weiteren möglichen Orten auf sich hat. Nun

Hinführung

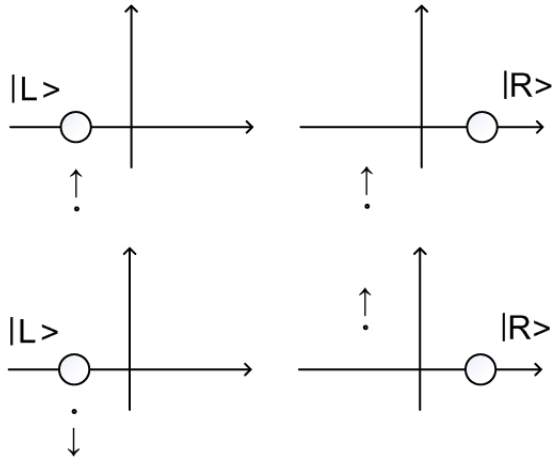


Abbildung 9.1: Annäherung an das Phänomen Dekohärenz.

bewegt sich ein Photon in Richtung  $|\uparrow\rangle$  auf den Ort  $|L\rangle$  zu, siehe Abbildung 9.1 oben. Befindet sich das eingangs beschriebene Teilchen bei  $|L\rangle$ , kommt es zu einer Kollision und das Photon wird reflektiert: dessen Bewegungsrichtung ändert sich zu  $|\downarrow\rangle$ .<sup>1</sup> Befindet sich das Teilchen bei  $|R\rangle$  kommt es zu keiner Berührung und damit zu keiner Veränderung. Diese beiden Alternativen sind im unteren Teil von Abbildung 9.1 dargestellt (dieses einführende Beispiel folgt [56]).

Da wir uns für Quantenzustände und deren Fragilität interessieren, betrachten wir die Superposition  $\frac{1}{\sqrt{2}}(|L\rangle + |R\rangle)$ . Die Wechselwirkung mit dem Photon können wir nun wie folgt beschreiben:

$$\frac{1}{\sqrt{2}}(|L\rangle + |R\rangle)|\uparrow\rangle = \frac{1}{\sqrt{2}}(|L\rangle|\uparrow\rangle + |R\rangle|\uparrow\rangle) \quad (9.1)$$

$$\xrightarrow{\text{Interaktion}} \frac{1}{\sqrt{2}}(|L\rangle|\downarrow\rangle + |R\rangle|\uparrow\rangle). \quad (9.2)$$

Man erkennt, dass die Interaktion eine unverschränkte Superposition in einen verschränkten Zustand überführt hat: der Zustand des Photons hängt vom Ort des Teilchens ab und umgekehrt. Damit ist die Eigenständigkeit der Ortsbeschreibung des Teilchens verloren gegangen, obwohl wir in dem Modell vereinfachend festgelegt hatten, dass sich nur der Photonzustand ändert. Das Photon trägt nun Information über den Ort des Teilchens. Im folgenden untersuchen wir den Einfluss der Außenwelt allgemeiner.

Allgemeine  
Situation

Stellen wir uns folgende allgemeine Situation vor: Ein Quantenobjekt implementiert ein Qubit; wir realisieren die Zustände  $|0\rangle$  und  $|1\rangle$ . Nun betrachten wir das Gesamtsystem aus unserem Qubit und dessen Umgebung. In der Folge bezeichnen wir den Zustand der „Außenwelt“ als  $|e\rangle$ , wie *environment*. Diesen sollten wir uns komplex und viele Dimensionen besitzend vorstellen,

<sup>1</sup>Der Einfachheit halber nehmen wir an, dass der Ort  $|L\rangle$  unseres Teilchens nicht verändert wird; eine realitätsnähere Modellierung werden wir später untersuchen.

aber davon werden wir hier abstrahieren. Kommt es zu einer Wechselwirkung zwischen Qubit und Umgebung (z.B. Kollision mit einem Gasmolekül, Wärmestrahlung o.ä.), so hängt der Folgezustand des Gesamtsystems davon ab, ob der Qubitzustand  $|0\rangle$  oder  $|1\rangle$  war, aber auch davon, welchen Zustand das Qubit nach der Interaktion haben wird. Dabei ändern sich zum einen das Qubit und zum anderen das erwähnte Gasmolekül. Wir beschreiben das wie folgt:

$$\begin{aligned} |0\rangle|e\rangle &\xrightarrow{\text{Interaktion}} |0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle, \\ |1\rangle|e\rangle &\xrightarrow{\text{Interaktion}} |0\rangle|e_{10}\rangle + |1\rangle|e_{11}\rangle. \end{aligned}$$

Ist also unser Bit zunächst im Zustand  $|i\rangle$  und hat nach der Wechselwirkung mit der Umgebung den Wert  $|j\rangle$ ,  $i, j \in \{0, 1\}$ , so wird diese Umgebung anschließend im Zustand  $|e_{ij}\rangle$  sein. Vielleicht ist es eine sinnvolle Annahme, dass die Amplituden von  $|e_{00}\rangle$  und  $|e_{11}\rangle$  jeweils einen hohen Betrag besitzen, aber auch davon abstrahieren wir.

Wir untersuchen, wie obige Interaktion die Superposition unseres Qubits verändert.

Dekohärenzgleichung

$$\begin{aligned} (\alpha|0\rangle + \beta|1\rangle)|e\rangle &\xrightarrow{\text{Interaktion}} \alpha(|0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle) + \beta(|0\rangle|e_{10}\rangle + |1\rangle|e_{11}\rangle) \\ &= (\alpha|0\rangle + \beta|1\rangle) \cdot \frac{1}{2}(|e_{00}\rangle + |e_{11}\rangle) \\ &+ (\alpha|0\rangle - \beta|1\rangle) \cdot \frac{1}{2}(|e_{00}\rangle - |e_{11}\rangle) \\ &+ (\alpha|1\rangle + \beta|0\rangle) \cdot \frac{1}{2}(|e_{01}\rangle + |e_{10}\rangle) \\ &+ (\alpha|1\rangle - \beta|0\rangle) \cdot \frac{1}{2}(|e_{01}\rangle - |e_{10}\rangle) \end{aligned}$$

Die Korrektheit dieser Gleichung lässt sich leicht nachrechnen. Interessanter ist ein anderer Aspekt: Sowohl die Außenwelt als auch unser Qubit haben sich verändert. Gehen wir davon aus, dass die Außenwelt das verschmerzen kann, wir aber unser Bit korrigieren müssen: Kommt dem Leser diese Aufgabe bekannt vor? Er könnte sich an die Teleportation erinnert fühlen: Bob muss dort mit Hilfe von Alice' klassischen Bits eine ähnliche Situation korrigieren. Wir haben dort vier Fälle:

Bitzustand	Fehlerart	Operation
$\alpha 0\rangle + \beta 1\rangle$	kein Fehler	$I$
$\alpha 0\rangle - \beta 1\rangle$	Phaseflip	$Z$
$\alpha 1\rangle + \beta 0\rangle$	Bitflip	$X$
$\alpha 1\rangle - \beta 0\rangle$	beides	$X \cdot Z$

Die Definition von Bitflip  $X$  und Phaseflip  $Z$  finden Sie ab Seite 79. Wir halten fest: Mit einem Verfahren, dass sowohl den Bitflip  $X$ , den Phaseflip  $Z$  als auch die Kombination  $X \cdot Z$  korrigiert, könnten wir die Veränderung durch Dekohärenz auf dem Bit rückgängig machen.

**Aufgabe 9.1:** Machen Sie sich mit diesen Operationen vertraut, indem Sie den Zusammenhang von  $X \cdot Z$  und  $Z \cdot X$  untersuchen.

Aus mathematischer Sicht ist es keine Überraschung, dass sich die Bitfehler auf diese Operationen reduzieren, denn  $I = I_2$ ,  $Z$ ,  $X$  und  $X \cdot Z$  bilden eine Basis des Raums aller  $2 \times 2$ -Matrizen. Es gilt

$$\begin{aligned} \begin{pmatrix} a & b \\ c & d \end{pmatrix} &= \frac{a+d}{2} \cdot I_2 + \frac{b+c}{2} \cdot X + \frac{c-b}{2} \cdot XZ + \frac{a-d}{2} \cdot Z \\ &= \frac{1}{2} \cdot \left[ \begin{pmatrix} a+d & 0 \\ 0 & a+d \end{pmatrix} + \begin{pmatrix} 0 & b+c \\ b+c & 0 \end{pmatrix} \right. \\ &\quad \left. + \begin{pmatrix} 0 & b-c \\ -b+c & 0 \end{pmatrix} + \begin{pmatrix} a-d & 0 \\ 0 & -a+d \end{pmatrix} \right] \end{aligned}$$

Durch scharfes Hinsehen lässt sich leicht feststellen, dass diese Gleichung korrekt ist. Dieses Ergebnis ist für die Korrektur von Quantenbits von fundamentaler Bedeutung.

Jede  $2 \times 2$ -Matrix  $M$  lässt sich als Linearkombination

$$M = a \cdot I_2 + b \cdot X + c \cdot XZ + d \cdot Z$$

darstellen.

Wir lassen die Verschränkung mit der Außenwelt kurz beiseite: Stellen wir uns ein Qubit im Zustand  $\alpha|0\rangle + \beta|1\rangle$  vor, das durch irgendeinen ungewollten Prozess in einen abweichenden Zustand  $\alpha'|0\rangle + \beta'|1\rangle$  übergeht. Wie müssten wir diesen Prozess beschreiben? Nach den Gesetzen der Quantenmechanik durch eine unitäre Transformation  $F = U_{\text{error}}$ . Auch diese ließe sich natürlich wie oben als Linearkombination von  $I_2$ ,  $X$ ,  $Z$  und  $XZ$  beschreiben.

Im Falle der Dekohärenz ist die Situation allgemeiner: In der Dekohärenzgleichung kommt neben dem Qubitzustand die „Außenwelt“ vor. Beides zusammen entwickelt sich unitär, das gilt aber nicht für das isoliert betrachtete Quantenbit. Statt einer mathematisch aufwändigen Analyse eine Veranschaulichung: der Effekt einer unitären Operation ließe sich immer umkehren. Das lässt sich aber auf dem mit der Umgebung verschränkten Qubit im Allgemeinen nicht bewerkstelligen, ohne dass auch die Veränderung der Außenwelt zurückgenommen würde. Möchten Sie sich einen solchen nicht-unitären Einfluss auf einem Qubit vorstellen: ein einfaches Modell wäre die Anwendung des Bitflips  $X$  mit einer Wahrscheinlichkeit  $p$  und sonst  $I_2$ . Realistischere Fehlermodelle findet der Leser zum Beispiel in [48] oder [114]. Behalten wir im Hinterkopf, dass wir es mit Wahrscheinlichkeitsverteilungen über unitäre Transformationen zu tun haben, können wir zunächst guten Gewissens voraussetzen:

Fehler auf einem Quantenbit lassen sich als Linearkombination

$$F = a \cdot I_2 + b \cdot X + c \cdot XZ + d \cdot Z$$

darstellen.

Zusammenfassung

Dekohärenz kann erklären, warum die Welt unserer sinnlichen Erfahrung so klassisch erscheint. Einige Pioniere der Quantenmechanik gingen von einem klaren Schnitt zwischen klassischer Welt und Quantenwelt aus. Die Theorie selbst enthält aber keine Hinweise auf einen solchen und Quanteneffekte wurden tatsächlich auch an komplexen Molekülen beobachtet, den sogenannten Fullerenen. Dekohärenz erklärt, warum allerdings Superpositionen ohne rigorose Abschottung so instabil sind. Sobald es zu einer Interaktion kommt, geht die *kohärente* Superposition dadurch verloren, dass ein verschränkter Zustand entsteht, wie in unserem Ausgangsbeispiel auf Seite 237: der Übergang von (9.1) zu (9.2) beschreibt, wie ein die Außenwelt stellvertretendes Photon auf ein Teilchen trifft.

Quanteneffekte im Alltag?

### Exkurs: Gemischter Zustand

Der Rest dieses Abschnitts soll den Einstieg in weiterführende Literatur erleichtern und kann zunächst übersprungen werden. Es geht um die Frage, was im Zustand  $\frac{1}{\sqrt{2}}(|L\rangle|\downarrow\rangle + |R\rangle|\uparrow\rangle)$  aus (9.2) für das Teilchen allein gilt. Was für ein Zustand entsteht, wenn wir die Außenwelt, also das Photon, ausblenden? In Abschnitt 5.3 haben wir argumentiert, dass von zwei verschränkten Bits keines allein Träger der Information ist. Denn so ein Bit isoliert betrachtet ist nicht länger in einem Quantenzustand, wie wir ihn kennen, sondern in einem *gemischten Zustand*.

Wir nehmen an, zwei Quantenbits befinden sich im Zustand  $|\phi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$ . Einen solchen Zustand eines Quantensystems, der ein Vektor des Zustandsraums ist, bezeichnen wir als *reinen Zustand*. Messen wir bezüglich des ersten Bits, geht das Register mit Wahrscheinlichkeit  $1/2$  in einen der Zustände  $\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle)$  oder  $\frac{1}{\sqrt{2}}(|10\rangle + |11\rangle)$  über. Der Nachfolgezustand ist der eine oder der andere. Wollen wir für weitere Analysen beide Möglichkeiten und die jeweilige Wahrscheinlichkeit berücksichtigen, können wir den Zustand nach der Messung als Wahrscheinlichkeitsverteilung von reinen Quantenzuständen auffassen. In diesem Zusammenhang sagen wir, unser System befinde sich in einem *gemischten Zustand*; man spricht auch von einem *statistischen Gemisch* reiner Zustände. Gemischte Zustände lassen sich durch sogenannte *Dichtematrizen* beschreiben. Und solche Dichtematrizen können verwendet werden, um Teilsysteme auch von verschränkten Zuständen zu untersuchen. Damit lässt sich unsere Ausgangsfrage beantworten: Beschreibt man dazu den reinen Zustand  $\frac{1}{\sqrt{2}}(|L\rangle|\downarrow\rangle + |R\rangle|\uparrow\rangle)$  als Dichtematrix, so ergibt sich, dass das „Teilsystem“ Teilchen isoliert betrachtet in einem gemischten Zustand ist. Das heißt: Beim Übergang von (9.1) zu (9.2) wird die Superposition des Teilchens zerstört; für Details siehe etwa [114] oder [89].

Reine und gemischte Zustände



Damit wirkt die Dekohärenz ähnlich wie eine Messung; allerdings ist es ein zeitabhängiger allmählicher Prozess. Dekohärenz ist der Grund dafür, dass unsere Welt klassisch wirkt und wir unseren Alltag von Quantenphänomenen unbehelligt verbringen.

## 9.2 Klassische Fehlerkorrektur

Um klassische Bits gegen Fehler zu schützen, nutzt man Redundanz produktiv: man verwendet eine größere Anzahl Bits, als mindestens benötigt. Die einfachste Idee hierzu bildet der *Wiederholungscode*. Als Beispiel kodieren wir ein Datenbit in ein Codewort der Länge drei, indem wir das Bit einfach kopieren bzw. wiederholen:

$$0 \mapsto 000, 1 \mapsto 111.$$

Wenn bei Ihnen als Empfänger die Codewörter 000, 001, 011, 111 ankämen, wäre Ihnen sicher klar, wie Sie das Datenbit erschließen sollten: Verwendet man zur Dekodierung das *Mehrheitsvotum*, so kann das Datenbit korrekt ermittelt werden, sofern nur eine oder keine Stelle des Codeworts verändert wurde.

Natürlich gibt es weit raffiniertere und dadurch effizientere Verfahren. Zum Beispiel nutzt der Hamming(7,4)-Code Codewörter der Länge sieben, um vier Datenbits gegen einzelne Bitfehler zu schützen. Als Einführung und Überblick kann [106] dienen.

## 9.3 Herausforderungen bei der Korrektur von Quantenbits

Wie wir gesehen haben, ist Fehlerkorrektur für Quantenbits besonders wichtig. Allerdings führen die Gesetze der Quantenmechanik dazu, dass diese Aufgabe weitaus schwieriger zu sein scheint als im klassischen Fall.

1. Der klassische Wiederholungscode kopiert ein klassisches Bit mehrfach. Das ist nach dem No-Cloning-Theorem für Quantenbits ausgeschlossen.
2. Im Falle des klassischen Wiederholungscode wird zur Fehlerkorrektur das Mehrheitsvotum der Bits des Codeworts verwendet. Klarerweise müssen wir dazu die Bitwerte kennen. Im Quantenfall müssten wir demnach die Quantenbits messen: Zufall kommt ins Spiel und das Codewort wird zerstört.
3. Es gibt unendlich viele mögliche Fehler. Gegen all diese müssen wir ein Qubit schützen. Nicht allein Bitsprünge, sondern auch minimale Veränderungen der Amplituden sind möglich.

Die Entdeckung der ersten fehlerkorrigierenden Codes für Quantenbits durch Peter Shor ([134]) und andere war von fundamentaler Bedeutung für die Möglichkeit, einen praktisch einsetzbaren Quantencomputer zu bauen.

## 9.4 Qubits gegen Fehler sichern

Die im letzten Abschnitt geschilderte Situation könnte hoffnungslos erscheinen. Mit einigen einfachen, in ihrer Kombination aber sehr cleveren Ideen, lassen sich diese Probleme überwinden. Eine wichtige Voraussetzung haben wir am Ende von Abschnitt 9.1 kennen gelernt: Jeder Fehler  $F$  auf einem einzelnen Quantenbit lässt sich als Linearkombination von  $I_2, X, Z$  und  $XZ$  darstellen. Wir beginnen mit dem Bitflip.

### 9.4.1 Bitflip-Code mit Syndrom: Korrektur des Codewords

Unser Ziel ist es, ein Qubit in einem beliebigen Zustand  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  gegen einen Bitflip  $X$  zu schützen. Wie in Abschnitt 9.3 erwähnt, verhindert das No-Cloning-Theorem, dass wir von diesem Qubitzustand wie bei dem klassischen Wiederholungscode einfach Kopien anfertigen. Allerdings können wir den Zustand ausweiten. Dazu verwenden wir ein Dreibit-Register und verfahren gemäß Abbildung 9.2.

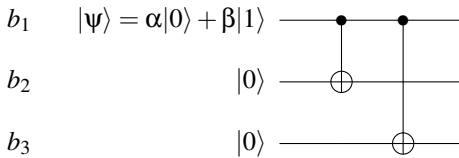


Abbildung 9.2: Ausweiten des Qubits  $b_1$ .

Anfangs ist das Quantenregister  $|b_1 b_2 b_3\rangle$  im Zustand  $(\alpha|0\rangle + \beta|1\rangle)|0\rangle|0\rangle = \alpha|000\rangle + \beta|100\rangle$  und wird durch zweifache Anwendung von CNOT in  $\alpha|000\rangle + \beta|111\rangle$  überführt. Diesen offensichtlich verschränkten Zustand werden wir als Codewort  $|c\rangle$  verwenden. Mit diesem wird es möglich sein, einen einzelnen Bitflip zu korrigieren. Wird zum Beispiel genau das erste Bit gekippt, können wir das abkürzend als Anwendung von  $F = X \otimes I \otimes I$  ausdrücken. Überlegen Sie kurz: Wie sieht das Resultat nach Anwendung auf das Codewort  $|c\rangle$  aus?

Es gilt:  $F|c\rangle = \alpha|100\rangle + \beta|011\rangle$ . Nehmen wir noch den Fall hinzu, dass kein Bitfehler geschieht (Identität auf allen Bits), erhalten wir vier Fälle:

Bitflip	Registerzustand
keiner	$\alpha 000\rangle + \beta 111\rangle$
$b_1$	$\alpha 100\rangle + \beta 011\rangle$
$b_2$	$\alpha 010\rangle + \beta 101\rangle$
$b_3$	$\alpha 001\rangle + \beta 110\rangle$

Können wir ermitteln, welcher der vier Fälle vorliegt, lässt sich das Codewort korrigieren. Messen dürfen wir keines der drei Bits; ansonsten erhalten wir ein zufälliges Ergebnis und alle Bits werden verändert. Wie sollten wir vorgehen? Zunächst ist unmittelbar klar: Nur falls alle drei Bits gleich sind, ist kein Fehler

aufgetreten. Das gilt für beide Komponenten der jeweiligen Superposition. Analog lassen sich alle vier Fälle charakterisieren:

Bitflip	Eigenschaft
keiner	$b_1 = b_2 = b_3$
$b_1$	$b_1 \neq b_2 = b_3$
$b_2$	$b_2 \neq b_1 = b_3$
$b_3$	$b_3 \neq b_1 = b_2$

Um den Fehler zu ermitteln, genügt es also,  $b_1$  mit  $b_2$  und  $b_1$  mit  $b_3$  zu vergleichen. Da es nur zwei Bitwerte gibt, legen zwei Vergleiche den dritten fest. Und diese Vergleiche lassen sich vornehmen, *ohne zu messen*; das erledigt der Schaltkreis in Abbildung 9.3.

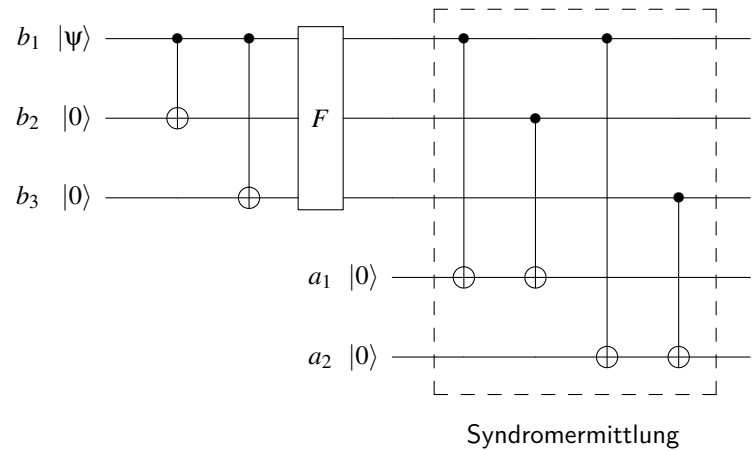


Abbildung 9.3: Ermittlung des fehlerhaften Qubits.

Zunächst wird das Quantenbit  $b_1$  im Zustand  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  wie oben beschrieben zu einem Codewort der Länge drei ausgeweitet, dann kippt der potentielle Bitflip  $F$  möglicherweise eines der Bits. Die zusätzlichen Hilfsbits  $a_1, a_2$  werden als *Syndrombits* bezeichnet. Es gilt  $a_1 = b_1 \oplus b_2$  und also  $a_1 = 1$ , falls  $b_1 \neq b_2$  und genauso  $a_2 = b_1 \oplus b_3 = 1$ , falls  $b_1 \neq b_3$ . Der aufgetretene Fehler wird ermittelt, ohne das Codewort zu messen. Diese Syndrombits sind nicht mit dem Codewort verschränkt und können also gemessen werden. Tatsächlich ist das für die Korrektur gar nicht nötig. Erweitern wir zunächst die Tabelle:

$a_1 a_2$	Bitflip	Eigenschaft
00	keiner	alle Bits gleich
10	$b_2$	$b_2 \neq b_1 = b_3$
01	$b_3$	$b_3 \neq b_1 = b_2$
11	$b_1$	$b_1 \neq b_2 = b_3$

Die Tabelle enthält nun alles, was wir für die Fehlerkorrektur wissen müssen. Beginnen wir mit dem Fall, dass  $b_1$  gekippt wurde. Diesen Fehler können wir mit einem Toffoli-Gatter korrigieren, welches  $b_1$  im Fall  $a_1 = a_2 = 1$  zurückkippt, siehe Abbildung 9.4. Bit  $b_2$  muss genau im Fall  $a_1 = 1, a_2 = 0$  gekippt werden. Dazu dient in demselben Schaltkreis zunächst ein CNOT-Gatter; das operiert aber unabhängig von dem Wert von  $a_2$ : im Fall  $a_1 = a_2 = 1$  wird  $b_2$  fälschlich gekippt. Dieses gleichen wir durch ein weiteres Toffoli-Gatter aus. Dieses kippt  $b_2$  für  $a_1 = a_2 = 1$  zurück. Genauso wird  $b_3$  durch die Kombination von CNOT- und Toffoli-Gatter korrigiert.

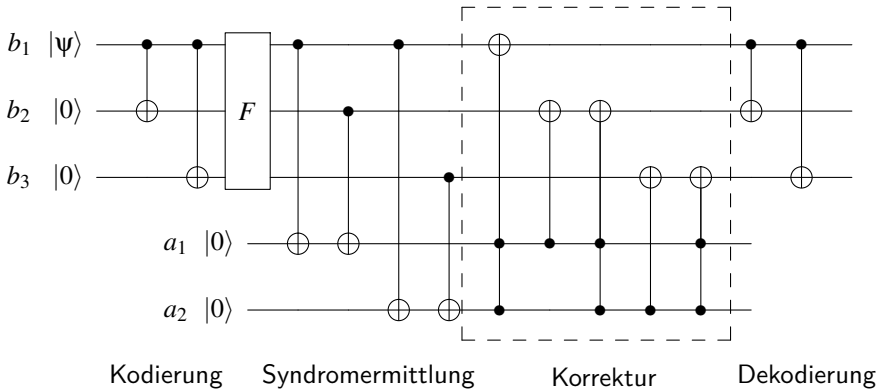


Abbildung 9.4: Bitflip-Korrektur mittels Syndromermittlung

Danach ist das gesamte Codewort wieder hergestellt und kann mit zwei CNOT-Gattern dekodiert werden. Die drei Quantenbits  $|b_1 b_2 b_3\rangle$  sind anschließend unverschränkt und  $b_1$  im Ausgangszustand  $|\psi\rangle$ . Nun haben wir alles beisammen, um uns auf theoretischem Wege leicht von der Korrektheit des Verfahrens zu überzeugen. Zum besseren Verständnis empfiehlt es sich zusätzlich, alle Möglichkeiten durchzurechnen.

**Aufgabe 9.2:** Nehmen Sie für alle vier Fälle schrittweise die Korrektur und Dekodierung gemäß Abbildung 9.4 vor.

### 9.4.2 Der Bitflip-Code: Korrektur des Datenbits

Durch das folgende Verfahren wird nicht das gesamte Codewort korrigiert, sondern nur das Ausgangsqubit wiederhergestellt. Dabei kommen wir ohne zusätzliche Syndrombits aus.

**Aufgabe 9.3:** Untersuchen Sie die Wirkungsweise des Schaltkreises in Abbildung 9.5. Wenden Sie diesen schrittweise auf die vier Fälle aus dem vorhergehenden Abschnitt an.

Nach der Lösung dieser Aufgabe lässt sich die allgemeine Strategie wie folgt beschreiben: Ist kein Fehler aufgetreten, so sind alle drei Bits gleich.  $b_1$  muss

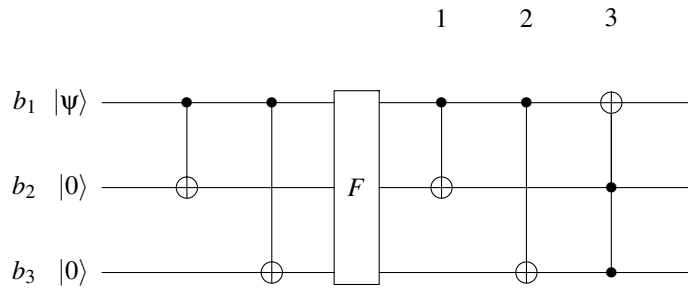


Abbildung 9.5: Bitflip-Code.

korrigiert werden, wenn beide Hilfsbits abweichen (die Fälle 100 oder 011). Das soll das Toffoligatter 3 durch ein Mehrheitsvotum bewerkstelligen: Im Fall 100 kippen die CNOT-Gatter  $b_2$  und  $b_3$ , so dass Toffoli anschließend  $b_1$  ändert. Beachten Sie, dass nur in diesem Fall und für 011 eine Korrektur vorgenommen wird.

Somit wird ein einzelner Bitflip korrigiert. Wie der klassische Wiederholungscode scheitert dieser Ansatz, falls mehrere Bits des Codeworts gekippt werden. Letztlich wird zur Korrektur ebenfalls das Mehrheitsvotum aller Bits verwendet.

### 9.4.3 Korrektur von Phaseflips

Mit den Ideen der vorhergehenden Abschnitte lässt sich ein einzelner Bitflip korrigieren. Das Schöne ist: Mit der gleichen Grundidee lassen sich auch Phaseflips angehen. Die folgenden Übungen leiten Sie dabei an, die Details selbst zu erarbeiten.

**Aufgabe 9.4:** Wenden Sie den Bitflip  $X$  und den Phaseflip  $Z$  auf die folgenden Zustände an:

$$|0\rangle, |1\rangle, |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \text{ und } |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Versuchen Sie, den Zusammenhang zwischen Bitflip und Phaseflip zu beschreiben.

**Aufgabe 9.5:** Sie entwickeln einen 3-Bit-Code für Phaseflips:

- Wie können wir das Ergebnis der vorhergehenden Aufgabe nutzen?
- Konstruieren Sie einen Schaltkreis für Kodierung, Korrektur und Dekodierung.
- Wenden Sie Ihr Verfahren für den Fall an, dass ein Phaseflip auf das erste Bit des Codeworts wirkt.

## 9.5 Shors 9-Qubit-Code

Bisher können wir zwei Arten von Fehlern korrigieren: Bitflips und Phaseflips. In Abschnitt 9.1 haben wir gesehen, wie allgemeine Fehler als Linearkombination  $F = a \cdot I_2 + b \cdot X + c \cdot XZ + d \cdot Z$  dargestellt werden können. Verwenden wir diese entscheidende Erkenntnis, so bleibt immerhin noch das folgende zu tun: Fehler durch Bitflips, Phaseflips und deren Kombination  $X \cdot Z$  müssen von ein und demselben Verfahren korrigiert werden. Das leistet *Shors 9-Qubit-Code*, indem die uns schon bekannten Ideen in zwei Stufen angewendet werden. Zunächst wird das zu schützende Qubit  $|\psi\rangle$  gemäß Abschnitt 9.4.3 mit dem Phaseflip-Code in ein Codewort der Länge drei umgewandelt. Anschließend wird jedes dieser drei Bits mit dem Bitflip-Code aus Abschnitt 9.4.2 kodiert.

Abbildung 9.6 zeigt einen ersten Schritt dazu: Nach Anwendung der Gatter 1-3 sind die drei Qubits der ersten Stufe im Zustand  $\alpha \cdot |++\rangle + \beta \cdot |--\rangle$ , dem Codewort des Phaseflip-Codes. Wir fügen zunächst, wie in genannter Abbildung dargestellt, zwei weitere Bits hinzu, um das erste Bit dieses Codeworts zu schützen. Der Registerzustand vor Anwendung der Gatter 4 und 5 lautet damit:

Grundidee

$$\alpha \cdot |+\rangle|00\rangle + \alpha \cdot |+\rangle + \beta \cdot |-\rangle|00\rangle|-\rangle.$$

Um die Anwendung der CNOT-Gatter 4 und 5 zu untersuchen, beschränken wir uns auf die ersten drei Bits:

$$\begin{aligned} \alpha \cdot |+\rangle|00\rangle + \beta \cdot |-\rangle|00\rangle &= \alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle + |100\rangle) + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle - |100\rangle) \\ &\xrightarrow{4,5} \alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \end{aligned}$$

Das Ergebnis ist die Kodierung der zweiten Stufe eines Bits der ersten Stufe.

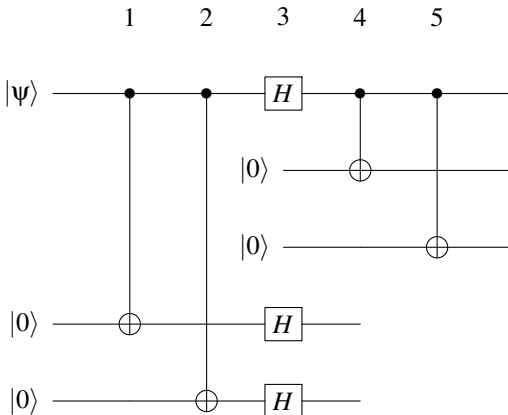


Abbildung 9.6: Shors 9-Qubit-Code: das erste Bit des Codeworts schützen.

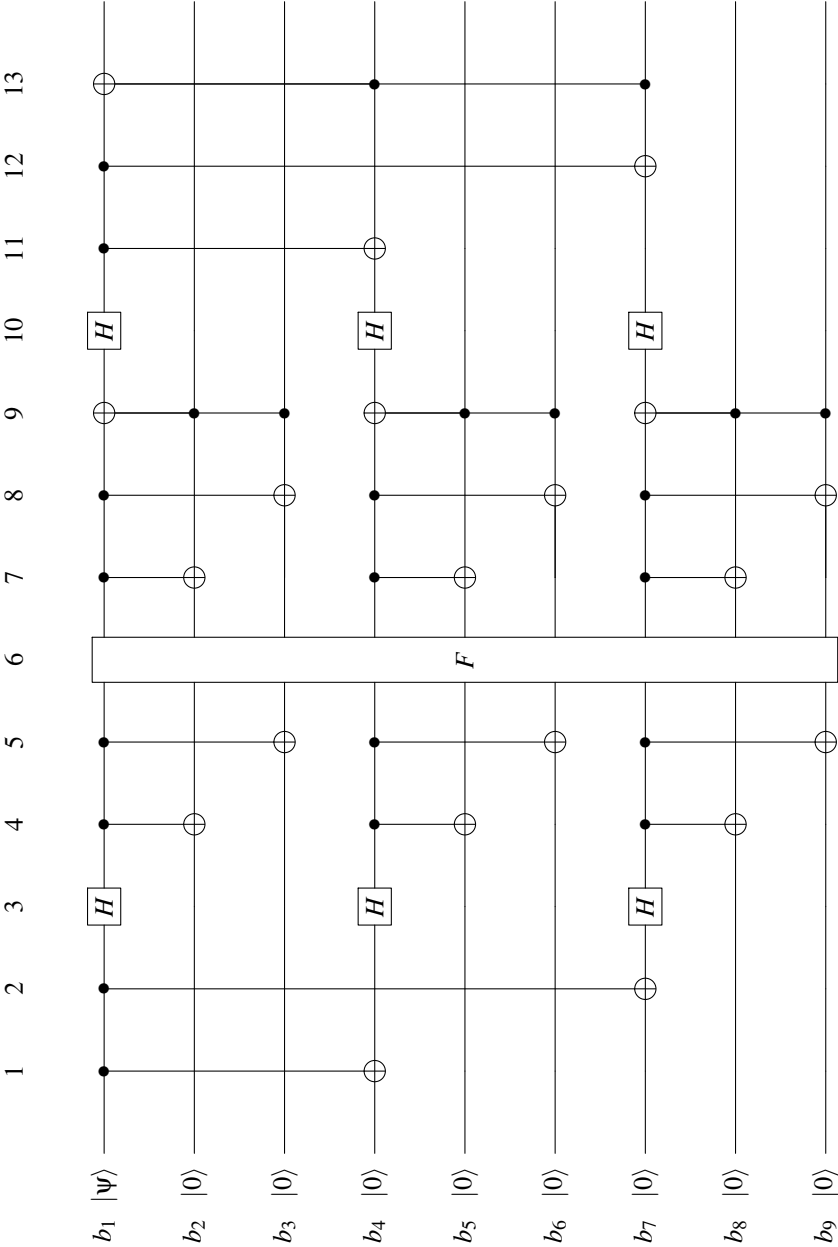


Abbildung 9.7: Shors 9-Qubit-Code

Das gesamte Verfahren ist in Abbildung 9.7 dargestellt. Für die schrittweise Analyse bezeichnen wir mit  $|\phi_i\rangle$  den Zustand des Registers  $|b_1 \dots b_9\rangle$  nach Anwendung von Gatter  $i$ . Wir beginnen mit

Gesamtversion

$$|\phi_0\rangle = (\alpha|0\rangle + \beta|1\rangle)|0\dots 0\rangle = \alpha|000\rangle|000\rangle|000\rangle + \beta|100\rangle|000\rangle|000\rangle.$$

Die Einteilung in Dreiergruppen findet gemäß der Kodierungsstufen statt. Die beiden CNOT-Gatter führen zu

$$|\phi_2\rangle = \alpha \cdot |000\rangle|000\rangle|000\rangle + \beta \cdot |100\rangle|100\rangle|100\rangle$$

und Hadamard anschließend zu

$$|\phi_3\rangle = \alpha \cdot |++0\rangle|++0\rangle|++0\rangle + \beta \cdot |--0\rangle|--0\rangle|--0\rangle.$$

Bits 1,4 und 7 sind verschränkt. Isoliert betrachtet, ergeben diese das Codewort des Phaseflip-Codes  $\alpha \cdot |+++ \rangle + \beta \cdot |-- \rangle$ . Die weiteren Bits mit Wert  $|0\rangle$  sind davon unabhängig und dienen der zweiten Stufe der Kodierung. Wir führen folgende Nebenrechnung auf einem isolierten Dreierblock des Zustands  $|\phi_3\rangle$  aus:

$$\begin{aligned} |++0\rangle &= \frac{1}{\sqrt{2}}(|000\rangle + |100\rangle) \\ &\xrightarrow{4,5} \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) =: |c^+\rangle \end{aligned}$$

Die Abkürzung  $|c^+\rangle$  wird die weitere Darstellung vereinfachen. Genauso:

$$\begin{aligned} |--0\rangle &= \frac{1}{\sqrt{2}}(|000\rangle - |100\rangle) \\ &\xrightarrow{4,5} \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) =: |c^-\rangle \end{aligned}$$

Wenden wir nun diese Schritte, also die CNOT-Gatter 4 und 5, auf den Gesamtzustand  $|\phi_3\rangle$  an, erhalten wir

$$\begin{aligned} |\phi_5\rangle &= \alpha \cdot \frac{1}{\sqrt{8}} \cdot (|000\rangle + |111\rangle) \cdot (|000\rangle + |111\rangle) \cdot (|000\rangle + |111\rangle) \\ &\quad + \beta \cdot \left[ \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \right] \\ &= \alpha \cdot |c^+c^+c^+\rangle + \beta \cdot |c^-c^-c^-\rangle. \end{aligned}$$

Die Bits 1, 4 und 7 repräsentieren also den Phaseflipcode in der Hadamard-Basis und werden jeweils gemäß dem Bitflip-Code erweitert. Der Zustand  $|\phi_5\rangle$  ist das aus  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  entstandene Codewort. Der Leser sei nun aufgefordert, Korrektur und Dekodierung im Detail nachzuvollziehen.



**Aufgabe 9.6:** Betrachten Sie die vier Fälle:

- a) Kein Fehler,
- b) Bitflip  $X$  auf Bit 1,
- c) Phaseflip  $Z$  auf Bit 1 und
- d) die Kombination  $XZ$  auf Bit 1.
- e) Überlegen Sie sich, dass das geschützte Qubit auch bei Fehlern auf einem der anderen Bits korrekt dekodiert wird.

## 9.6 Ausblick

Die letzten Schritte sind technisch aufwändig, einfach hingegen ist die Grundidee, den Phaseflip-Code und den Bitflip-Code in zwei Stufen hintereinander zu schalten. Damit können wir jeden Fehler  $X$ ,  $Z$  oder  $XZ$  auf einem Bit korrigieren. Wie verhält sich Shors 9-Qubit-Code, wenn nun ein beliebiger Fehler  $F$  auf ein einzelnes Bit wirkt? Gemäß Abschnitt 9.1 lässt sich ein solcher als Linearkombination

$$F = a \cdot I_2 + b \cdot X + c \cdot XZ + d \cdot Z$$

darstellen. Lassen wir  $F$  ein Bit unseres Codeworts  $|c\rangle = \alpha \cdot |c^+ c^+ c^+\rangle + \beta \cdot |c^- c^- c^-\rangle$  verändern, erhalten wir (in vereinfachter Schreibweise: wir lassen die Angabe von  $I$  auf den unveränderten Bits weg):

$$F|c\rangle = a \cdot |c\rangle + b \cdot X|c\rangle + c \cdot XZ|c\rangle + d \cdot Z|c\rangle.$$

Nun wird in jeder der vier Komponenten der jeweilige Fehler korrigiert.

Das wirkt allerdings praxisfern: auf einem Codewort aus neun Qubits ist auf einem ein beliebig gravierender Fehler erlaubt, aber alle anderen dürfen sich nicht ändern. Realistischer ist die Situation, dass jedes der Bits einer kleinen Veränderung unterliegt, einem Rauschen nahe der Identität. Ein Beispiel wäre

$$\cos(\epsilon) \cdot I + i \sin(\epsilon) \cdot X,$$

eine unitäre Operation, die für kleines  $\epsilon$  nahe bei der Identität liegt. Wir nähern solche Operationen für die folgende Betrachtung vereinfachend als  $I + \epsilon \cdot F$  an,  $\epsilon \cdot F$  beschreibt also die Abweichung vom Ideal. Auch in dieser Situation hilft Shors 9-Qubit-Code: die Größenordnung der Fehler verringert sich quadratisch. Um das einzusehen, beginnen wir mit zwei Bits  $b_1 \otimes b_2$  und betrachten die Auswirkung von  $(I + \epsilon \cdot F_1)$  auf  $b_1$  und von  $(I + \epsilon \cdot F_2)$  auf  $b_2$ :

$$\begin{aligned} & (I + \epsilon \cdot F_1) \otimes (I + \epsilon \cdot F_2) \\ &= I \otimes I + (I \otimes \epsilon \cdot F_2) + (\epsilon \cdot F_1 \otimes I) + \epsilon^2 (F_1 \otimes F_2) \end{aligned}$$

Wie man sieht, hat die Komponente mit Fehlern auf beiden Bits den Faktor  $\varepsilon^2$ . Bei allen anderen Komponenten der Superposition wird höchstens ein Bit verändert und durch den Code korrigiert. Genauso ergibt sich bei  $n$  Bits:<sup>2</sup>

$$\begin{aligned} & (I + \varepsilon \cdot F_1) \otimes \dots \otimes (I + \varepsilon \cdot F_n) \\ = & I^n + \varepsilon(F_1 \otimes I^{n-1} + I \otimes F_2 \otimes I^{n-2} + \dots + I^{n-1} \otimes F_n) \\ & + \varepsilon^2(\text{Fehler auf zwei Bits}) \\ & + \varepsilon^3(\text{Fehler auf drei Bits}) + \dots + \varepsilon^n(F_1 \otimes \dots \otimes F_n). \end{aligned}$$

Es wird deutlich, dass der von dem Code nicht korrigierte Anteil durch  $O(\varepsilon^2)$  abschätzbar ist. Diese Argumentation gilt nicht nur für Shors 9-Qubit-Code, sondern für all solche, die  $X$ ,  $Z$  und  $XZ$  auf einem Bit korrigieren. Codes, die  $t$  Fehler korrigieren, senken die Ordnung analog auf  $O(\varepsilon^{t+1})$ .

Es gibt effizientere Beispiele als Shors Code: so beschreiben Laflamme, Miquel, Paz und Zurek in [97] einen 5-Qubit-Code. Fünf ist auch die minimale Anzahl an Qubits zur Absicherung gegen einen einzelnen Bitfehler. Daniel Gottesman hat 1997 den *Stabilizer-Formalismus* eingeführt, der durch die geschickte Anwendung von Gruppentheorie eine einheitliche Untersuchung und Darstellung von Codes zur Quantenfehlerkorrektur ermöglicht. So sind Shors 9-Qubit-Code und der erwähnte 5-Qubit-Code Beispiele von *Stabilizer-Codes*; siehe etwa [47] für eine Einführung.

Weitere Codes

Als äußerst vielversprechend hat sich zuletzt das Konzept des *Surface Codes* erwiesen, siehe [60]. Werden dabei alle Qubits in einem zweidimensionalen Raster angeordnet, zeigen sich für die Umsetzung von – zum Beispiel – supraleitenden Quantenbits attraktive Eigenschaften, siehe etwa [65], [90] oder [7]. Surface Codes bilden wiederum einen Spezialfall *topologischer Codes*, [142] bietet einen Überblick.

Ein wichtiger Schritt auf dem Weg zu einem universellen Quantencomputer könnte [91] darstellen. Hier wird für den Sycamore-Prozessor (siehe Abschnitt 12.6) und einen einfachen Wiederholungscode nachgewiesen, dass unter gegebenen Voraussetzungen der Fehler exponentiell mit der Zahl der verwendeten Qubits sinkt.

Stillschweigend haben wir bisher vorausgesetzt, dass ein Qubit auf einem störungsanfälligen Kanal zwischen perfekten Modulen eines Quantencomputers geschützt werden muss. Verwendet man nun einen fehlerkorrigierenden Code und dekodiert das Qubit vor Beginn einer Berechnung, so sind Störungen während dieser möglich. Darum sollten Berechnungen direkt auf den kodierten Zuständen vorgenommen werden, und zwar unter Verwendung von *fehlertoleranten* Gattern. Tatsächlich wird nämlich auch jedes Gatter eine Abweichung vom theoretischen Ideal aufweisen, selbst jene, die zur Fehlerkorrektur dienen. Einführungen in das Thema *Fehlertolerante Berechnungen* finden sich in [48] oder [47].

Fehlertolerante Berechnungen

<sup>2</sup> $I^n$  meint  $I_2^n = I_{2^n}$ .

# 10 Quantenhardware

*Die Computer der Zukunft könnten weniger als anderthalb Tonnen wiegen.*  
Prognose des Magazins *Popular Mechanics*, 1949.

Um den Aufbau von Quantenhardware zu verstehen, sind tiefere Kenntnisse der Physik nötig. Darum bleibt dieses Kapitel eher an der Oberfläche: es werden lediglich einige Grundkonzepte beschrieben. Unser Hauptziel ist es, prinzipiell zu verstehen, wie Operationen auf Quantenbits realisiert werden können und welche Schwierigkeiten dabei überwunden werden müssen. Wir beginnen mit einer allgemeinen Erörterung, welche Anforderungen Quantencomputer zu erfüllen haben. Am ausführlichsten behandeln wir die Informationsverarbeitung mit Hilfe von Photonen. Dabei dient der Abschnitt 10.2.1 über das Mach-Zehnder-Interferometer als allgemeine Einführung. Zum Einstieg in eine tiefergehende Beschäftigung mit diesem Thema ist [27] sehr gut geeignet.

## 10.1 Anforderungen

Was sind die Mindestanforderungen an eine Rechenmaschine? Erinnern wir uns an die Beschreibung einer klassischen Berechnung aus Abschnitt 2.1, können wir folgende Punkte als notwendig angeben:

- Wir können die Maschine in einen Anfangszustand versetzen,
- daraufhin führt diese die nötigen Rechenschritte aus;
- das Resultat der Berechnung ist der Endzustand, den uns der Rechner mitteilt.

Dabei wird vorausgesetzt, dass sich die Berechnung exakt gemäß der Rechenschritte entwickelt, beziehungsweise dass die Maschine Informationen stabil

speichern kann. Das geschieht mit den sogenannten Bits, die technisch auf sehr vielfältige Weise realisiert werden können. Bei sehr frühen Rechenmaschinen bestimmte zum Beispiel die Stellung von Zahnrädern den Zustand der Rechenmaschine und damit das Zwischenergebnis. Diese Maschinen verwendeten meist das Dezimalsystem, aber das Prinzip ähnelt der Umsetzung von Bits. Frühen elektronischen Rechenmaschinen wurde die Aufgabenstellung binär kodiert durch eine Reihe von Schaltern mit zwei Zuständen mitgeteilt, und das Ergebnis war eine Binärzahl, deren einzelne Bits durch Lämpchen visualisiert wurden.

Zusammen mit unseren Kenntnissen über Quantenbits und Quantenberechnungen ergeben sich nun folgende Anforderungen an einen Quantencomputer, die 1985 von David Deutsch auf ähnliche Weise formuliert wurden:

Notwendige  
Eigenschaften  
eines Quanten-  
computers

- Ein Quantencomputer besteht aus einer Reihe von Quantenbits,
1. die in einen Anfangszustand versetzt werden können,
  2. die Information robust speichern,
  3. auf die (universelle) Quantengatter anwendbar sind und
  4. die gemessen werden können.

Für Punkt 1 genügt es, einen festen bekannten Zustand  $|0\rangle$  herstellen zu können – die Eingabe der Quantenberechnung ist in der Regel ein Teil des Schaltkreises, zum Beispiel ein Orakel. Punkt 3 ist die Kurzform der Aussage: Gatter aus einer *universellen Menge* können angewendet werden. Dazu müssen Zwei-Bit-Gatter auf zwei beliebigen Bits einsetzbar sein. Und Messungen – der letzte Punkt – sollten zumindest in der Standardbasis  $|0\rangle, |1\rangle$  und an jeder Teilmenge der Bits ausgeführt werden können.

Am schwierigsten umzusetzen sind die Punkte 2 und 3, die zusammengekommen sicherstellen, dass sich der Zustand des Quantencomputers nach Wunsch unitär entwickelt. Das Stichwort dazu lautet *Dekohärenz*. Wie in Abschnitt 9.1 beschrieben, gerät ein Qubit durch Wechselwirkung mit der Außenwelt in einen mit dieser verschränkten Zustand. Information geht dabei unwiederbringlich verloren.

Dekohärenzzeit

Und in der Praxis lassen sich Quantenbits von der Außenwelt nicht perfekt abschirmen. Das ist schon deshalb nicht möglich, weil mit ihnen gerechnet werden soll. Die Dauer des oben geschilderten Vorgangs nennt man die *Dekohärenzzeit*. Wie die Größenordnung der Quantenwelt erwarten lässt, ist die Dekohärenzzeit bisher realisierter Quantenbits in den Maßstäben unserer Alltagswelt sehr kurz. Aber auch wenn der Zustand eines Quantenbits nur für Sekundenbruchteile stabil ist, könnten ausreichend viele Transformationen samt abschließender Messung vorgenommen werden, wenn sich die Gatter schnell genug anwenden lassen.

Der Weg zu den  
Anwendungen

Wann Quantencomputer praktisch effizient eingesetzt werden können, ist zur Zeit nicht vorhersehbar. Die experimentelle Phase scheint jedoch beendet zu sein. Zum Zeitpunkt der Drucklegung der aktuellen Auflage dieses Buchs haben verschiedene Unternehmen auf stabilen und leistungsfähigen

Quantenbits arbeitende Rechner realisiert, wobei Geräte mit 50 solchen Bits für die nahe Zukunft realistisch erscheinen. Vielleicht können schon in einigen Jahren deutlich größere Rechner realisiert werden. Entscheidend sind jedoch nicht allein Rekorde bei der Anzahl der Bits, sondern die Fortschritte bei Verlässlichkeit (Dekohärenzzeit und Fehlerkorrektur) und der Art, wie die Bits adressiert werden können.

## 10.2 Photonen

Isaac Newton schlug im 17. Jahrhundert vor, Licht als einen Strom von Teilchen anzusehen. Im 19. Jahrhundert durchgeführte Experimente legten es dagegen nahe, Licht als elektromagnetische Welle zu beschreiben. Damit kann jedoch ein Phänomen nicht erklärt werden: Eine geladene Metallplatte wird von ultravioletttem Licht entladen, von für uns sichtbarem Licht jedoch nicht. Einstein löste dieses Rätsel im Jahre 1905, indem er dem Licht wiederum Teilchencharakter zuschrieb. Der eben beschriebene *photoelektrische Effekt* beruht darauf, dass Elektronen aus der geladenen Platte geschlagen werden, und zwar von einzelnen Energiepaketen, in denen die Energie des Lichts konzentriert ist: von den *Photonen*.

### 10.2.1 Mach-Zehnder-Interferometer

Wir beginnen mit einem einfachen Versuchsaufbau, der unsere bisherige abstrakte Sicht von Quantenbits und Quantengattern mit der Praxis verbindet. Es handelt sich um ein Ende des 19. Jahrhunderts entwickeltes Gerät, das in ähnlicher Form zur Navigation von Flugzeugen verwendet wird. Die Grundidee eines Interferometers ist es, Licht zunächst längs zweier Wege zu schicken, um es schließlich wieder zu vereinen. Ziel ist es, die Interferenz (siehe Abschnitt 4.4) zu messen.

Das Mach-Zehnder-Interferometer – unabhängig von den Physikern Ludwig Mach und Ludwig Zehnder entwickelt – ist in Abbildung 10.1 schematisch dargestellt. Links unten trifft das Licht auf einen halbreflektierenden Spiegel oder Strahlteiler  $S_1$ : die Hälfte des Lichts tritt hindurch, die andere Hälfte wird reflektiert und verläuft anschließend senkrecht zu der ursprünglichen Richtung. Nun trifft das Licht auf beiden Wegen auf die vollreflektierenden Spiegel  $S_L$  beziehungsweise  $S_R$  und wird in beiden Fällen auf den Strahlteiler  $S_2$  gelenkt. Zwei Messgeräte  $D_0$  und  $D_1$  messen das Licht, das in der Ursprungsrichtung beziehungsweise in der dazu senkrechten Richtung austritt.

Aufbau

*Schritt 1:* Zunächst halten wir ein Blatt Papier zwischen  $S_R$  und  $S_2$ . Dann kommt in  $S_2$  nur Licht an, das in  $S_1$  reflektiert wurde und den Weg über  $S_L$  genommen hat. Der Strahlteiler  $S_2$  hat die gleichen Eigenschaften wie  $S_1$ :  $S_2$  lässt die Hälfte des ankommenden Lichtes zu  $D_0$  durchtreten, die andere Hälfte wird zu  $D_1$  reflektiert.  $D_0$  und  $D_1$  registrieren also die gleiche Lichtmenge.

Verhalten mit  
Papierstreifen

In *Schritt 2* verwenden wir eine Lichtquelle, die so fein justierbar ist, dass sie nur einzelne Photonen aussendet. Die Empfindlichkeit der Messgeräte passen

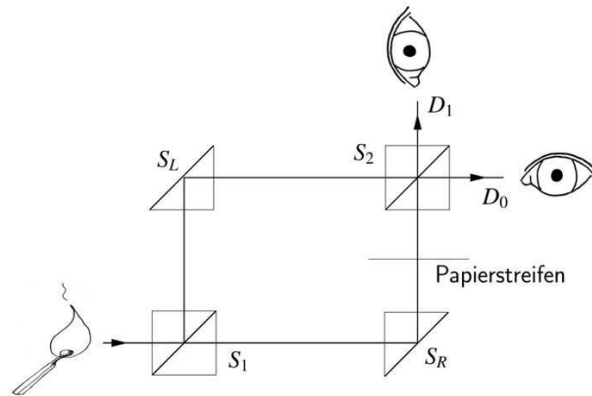


Abbildung 10.1: Schematischer Aufbau des Mach-Zehnder-Interferometers. Der Papierstreifen ist für einen Teil des Experiments nötig.

wir an: Einzelphotonendetektoren zählen die ankommenden Lichtquanten. Nun wird die Hälfte der Photonen von  $D_0$  und die Hälfte von  $D_1$  registriert. Für jedes einzelne Photon sind die Registrierung in  $D_0$  und die in  $D_1$  gleich wahrscheinlich.

Verhalten ohne  
Papierstreifen

Das bedeutet jedoch nicht, dass ein Photon im Strahlteiler mit Wahrscheinlichkeit  $1/2$  den Weg über  $S_L$  geht und mit der gleichen Wahrscheinlichkeit den über  $S_R$  einschlägt. Das lässt sich im nächsten Schritt feststellen.

*Schritt 3:* Wir entfernen den Papierstreifen und beobachten: nur von  $D_0$  werden Photonen registriert! Wenn unsere obige Annahme zuträfe, dass an  $S_1$  ausgewürfelt würde, ob die Reise über  $S_L$  oder über  $S_R$  gehen wird, woher könnte dann ein über  $S_L$  gelenktes Photon von dem Papierstreifen wissen? Also ist die Annahme falsch, dass ein Photon mit jeweils gleicher Wahrscheinlichkeit einen der Wege einschlägt: sie kann die im Experiment beobachtbaren Phänomene nicht erklären.

### Erklärung

Lichtwellen

Zu der Zeit, als das Mach-Zehnder-Interferometer entwickelt wurde, war es nicht möglich, einzelne Photonen zu erzeugen oder zu messen. In dem Versuchsaufbau ohne Papierstreifen (Schritt 3) erhalten wir ein analoges Ergebnis, wenn wir nicht einzelne Photonen, sondern einen kräftigen Lichtstrahl in das Interferometer schicken: Nur bei  $D_0$  tritt Licht aus. Mit der Wellentheorie des Lichts lässt sich dieses Verhalten erklären. Der Grund ist Interferenz. In  $D_0$  ankommendes Licht ist entweder in  $S_1$  reflektiert und in  $S_2$  durchgelassen worden, oder es passierte  $S_1$  und wurde von  $S_2$  gespiegelt. Auf beiden Wegen wurde also einmal gespiegelt und einmal durchgelassen. Wir kürzen diese Fälle mit SD und DS ab. Zu  $D_1$  gelangt Licht wenn es von  $S_1$  und  $S_2$  durchgelassen wurde oder wenn es von beiden Strahlteilern gespiegelt

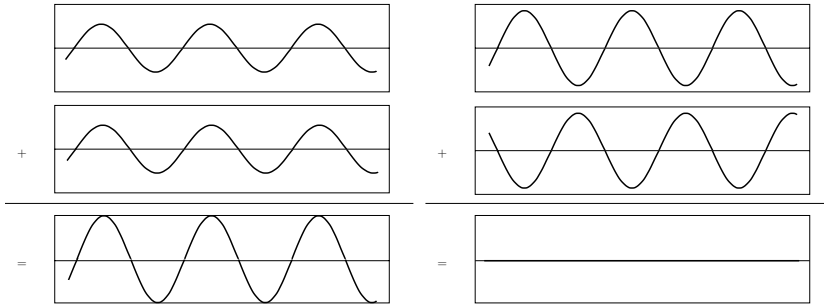


Abbildung 10.2: Konstruktive und destruktive Interferenz zweier Wellen

wurde. Es sind dies die Kombinationen DD und SS. Nun macht es einen Unterschied, ob das Licht gespiegelt oder durchgelassen wurde: das Resultat ist eine *Phasenverschiebung*. Bei einem korrekt justierten Mach-Zehnder-Interferometer ist die Phase zwischen den Fällen DD und SS um gerade die halbe Wellenlänge verschoben. Wie in Abbildung 10.2 rechts gelangt durch destruktive Interferenz kein Licht zu  $D_1$ . Das bei  $D_0$  eintreffende Licht gelangt über die Wege SD und DS dorthin: es macht keinen Unterschied, ob zuerst gespiegelt und dann durchgelassen wurde oder umgekehrt. Die Phasen werden auf beiden Wegen in gleicher Weise verändert. Das Ergebnis ist konstruktive Interferenz, Abbildung 10.2 links.

Doch wie erklären wir das Verhalten einzelner Photonen? Wie erfährt in Schritt 3 ein über  $S_L$  reisendes Lichtteilchen von dem Papierstreifen hinter  $S_R$ ? Diese Frage geht von der mit den Phänomenen nicht zu vereinbarenden Annahme aus, ein Photon gerate entweder über  $S_L$  oder über  $S_R$  an sein Ziel. Dieses Problem verschwindet, wenn wir die Zustände der Photonen durch Superpositionen beschreiben und den Wegen Amplituden beifügen. Dazu ordnen wir einem Photon, das sich in horizontaler Richtung bewegt, den Zustand  $|0\rangle$  zu und einem in vertikaler  $|1\rangle$ . Damit ist ein Photon anfangs, also vor dem Strahlteiler  $S_1$ , im Zustand  $|0\rangle$ . Um eine Erklärung zu finden, nehmen wir an, der halbrelektierende Spiegel erzeuge daraus die Superposition  $1/\sqrt{2}(|0\rangle + |1\rangle)$ : die vertikale sowie die horizontale Richtung erhalten jeweils die Amplitude  $1/\sqrt{2}$ . Die vollreflektierenden Spiegel tauschen die Amplituden von  $|0\rangle$  und  $|1\rangle$ ; der Zustand  $1/\sqrt{2}(|0\rangle + |1\rangle)$  bleibt unverändert. Das am Strahlteiler  $S_2$  eintreffende Licht befindet sich also in einer Superposition aus beiden Richtungen, die Amplitude ist jeweils  $1/\sqrt{2}$ .  $S_2$  verhält sich dabei genauso wie  $S_1$ : der Term  $|0\rangle$  wird zu  $1/\sqrt{2}(|0\rangle + |1\rangle)$ . Wenn nun  $|1\rangle$  seinerseits zu  $1/\sqrt{2}(|0\rangle - |1\rangle)$  wird, können wir das Phänomen erklären! In unserem Erklärungsversuch führen die beiden Strahlteiler zweimal die Hadamard-Transformation aus: durch destruktive Interferenz verschwindet bei  $S_2$  die Amplitude von  $|1\rangle$ , die von  $|0\rangle$  wird durch positive Interferenz vergrößert:

Wahrscheinlichkeitsamplituden

$$\frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right) = |0\rangle.$$

Wie beschreiben wir den Fall mit Papierstreifen? In diesem Fall ist der Weg der  $|1\rangle$ -Komponente blockiert. In  $S_2$  wird gemäß unserer Beschreibung die Hadamard-Transformation auf  $|0\rangle$  angewendet. Das Photon wird mit gleicher Wahrscheinlichkeit von  $D_0$  und  $D_1$  registriert. Die beiden Detektoren ergeben zusammen eine Messung in der Basis  $|0\rangle, |1\rangle$ .

Die Darstellung ist stark vereinfacht und soll einen ersten Hinweis zur Umsetzung von Quantengattern mit einfachen optischen Geräten geben. Strahlteiler können tatsächlich die Hadamard-Transformation ausführen, aber auch viele andere Operationen sind auf ähnliche Weise realisierbar. [2] ist eine vertiefende theoretische Arbeit zu diesem Thema.

### Folgerung für Quantenberechnungen

Ein Mach-Zehnder-Interferometer rechnet nicht, aber seine Komponenten können im Prinzip für Quantenberechnungen verwendet werden. Dafür benötigen wir unitäre Transformationen und Messungen. Bei unserem Interferometer wird mit einer Strahlteiler-Spiegel-Kombination eine unitäre Transformation realisiert. Die Messung wird mit einem weiteren Strahlteiler und zwei Einzelphotonendetektoren vorgenommen. In Abschnitt 10.1 wurde von einem Quantencomputer gefordert, dass dessen Quantenbits Information robust speichern können. In unserem Fall bedeutet das, ein Photon darf auf dem Weg von der Quelle bis zu einem Detektor keinen störenden Einflüssen ausgesetzt sein. In Abschnitt 10.5 wird ein Experiment beschrieben, in dem mit Photonen unter anderem Grovers Algorithmus ausgeführt wurde. In der Folge beschäftigen wir uns zunächst mit Zufallszahlen und Kryptographie.

## 10.2.2 Zufallszahlen

Unsere erste Anwendung in Abschnitt 2.4 war ein Zufallsgenerator. Folgerichtig betrachten wir auch als erste Hardwareumsetzung einen solchen. Abbildung 10.3 zeigt den schematischen Aufbau eines Zufallsgenerators, wie er von der Schweizer Firma ID Quantique hergestellt wird.

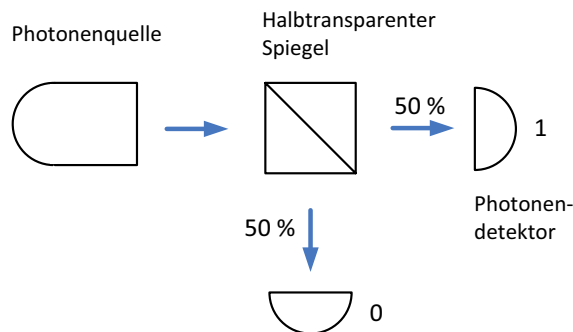


Abbildung 10.3: Zufallszahlen mit Photonen





Abbildung 10.4: Ein schwingendes Seil, vertikal beziehungsweise horizontal ausgerichtet

Eine Einzelphotonenquelle sendet Lichtteilchen auf einen halbdurchsichtigen Spiegel. Mit gleicher Wahrscheinlichkeit passiert ein Photon dieses Bauteil oder wird reflektiert. Diesmal ordnen wir einem Photon, das sich in horizontaler Richtung bewegt, den Zustand  $|1\rangle$  zu und der vertikalen Richtung den Zustand  $|0\rangle$ . Die Detektoren messen das so definierte Qubit. Somit können wir diese Konstruktion als mögliche Umsetzung des Schaltkreises in Abbildung 2.12 ansehen. Natürlich ist es keine 1-zu-1-Umsetzung. So gehen wir hier von dem Anfangszustand  $|1\rangle$  aus. Der Strahlteiler setzt nicht unbedingt die Hadamard-Transformation um, es könnte sich je nach Konstruktion um eine ähnliche Transformation mit einer Phasenverschiebung handeln. Das beschriebene Gerät ist nicht größer als ein Eiswürfel und kann mehrere Millionen Zufallsbits pro Sekunde erzeugen.

### 10.2.3 Kryptographie

Photonen eignen sich gut für die Quanten-Informationsübertragung, da sie sich leicht über große Strecken transportieren lassen. Zur Definition eines Quantenbits verwendet man, dass sich Licht *polarisieren* lässt. Ein wie in Abbildung 10.4 in Bewegung gesetztes Seil schwingt in einer Ebene. Licht verhält sich ähnlich, und wir können uns vorstellen, dass in alltäglichen Lichtquellen wie der Sonne stets die verschiedensten Ausrichtungen vorkommen.

Polarisation

Mit einem Polarisationsfilter – von Fotografen auch Polfilter genannt – lässt sich erreichen, dass Licht in nur einer Ebene schwingt. Folgendermaßen machen wir ein Photon zu einem Quantenbit: vertikaler Polarisation ordnen wir den Wert  $|0\rangle$  zu und horizontaler den Wert  $|1\rangle$ . Um nun solche Quantenbits zu transportieren und zu transformieren, kann auf Techniken und Methoden der modernen Optik zurückgegriffen werden.

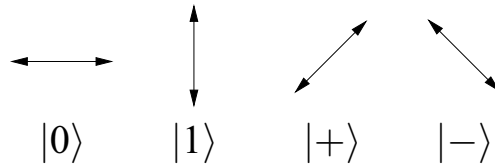


Abbildung 10.5: Quantenkryptographie und polarisierte Photonen

BB84

Wir erinnern uns an den Grundaufbau des BB84-Protokolls (Abschnitt 7.2). Alice sendet Quantenbits an Bob, die bezüglich zweier Basen definiert sind. Den Zuständen  $|0\rangle$  und  $|1\rangle$  ordnen wir horizontale (0 Grad) und vertikale Polarisation (90 Grad) zu,  $|+\rangle$  und  $|-\rangle$  entsprechen die Polarisationsrichtungen 45 Grad und 135 Grad (oder alternativ -45 Grad), siehe Abbildung 10.5.

Nun kann Alice ein für Bob bestimmtes Quantenbit herstellen, indem sie ein Photon mit einem Polarisationsfilter in der gewünschten Weise ausrichtet und zu Bob leitet. Bob leitet das empfangene Photon auf einen Polarisationsstrahlteiler, zum Beispiel einen Kalkspatkristall.

Ist dieses Instrument horizontal ausgerichtet, treten horizontal polarisierte Photonen hindurch, vertikale werden abgelenkt. Mit zwei Detektoren kann Bob diese Fälle unterscheiden. Photonen mit Polarisationsrichtung 45 Grad werden hingegen mit jeweils gleicher Wahrscheinlichkeit durchgelassen und abgelenkt; das gleiche gilt für Richtung 135 Grad. So erlaubt der horizontal ausgerichtete Kristall die Messung in der Basis  $|0\rangle, |1\rangle$ . Ein Quantenbit in einem dieser Basiszustände liefert mit Sicherheit das entsprechende Ergebnis. Quantenbits in den Zuständen  $|+\rangle$  oder  $|-\rangle$  führen zu einem zufälligen Resultat. Drehen wir nun diese Apparatur um 45 Grad, messen wir die Quantenbits entsprechend in der Basis  $|+\rangle, |-\rangle$ . Also können Alice und Bob das BB84-Protokoll ausführen, indem sie in jedem Schritt ihre Apparate gemäß ihrer Zufallsbits einstellen. In der Praxis kann die Polarisationsrichtung mit sogenannten Pockelszellen elektronisch gesteuert und schnell eingestellt werden.

**Aufgabe 10.1:** Warum können wir sichergehen, dass Eve keinen Kristall besitzt, der den Strahl gemäß der vier Richtungen aufspaltet?

Prototyp

Die in dem Abschnitt vor Aufgabe 10.1 beschriebene Situation ist der schematische Aufbau des ersten Prototyps eines Quantenkryptographiesystems. 1991 wurden auf diese Weise Schlüssel über eine Strecke von 30 Zentimetern erfolgreich übertragen [18].

Allerdings weicht dieser Prototyp in einem entscheidenden Punkt von dem BB84-Protokoll ab. Denn Alice' Photonen stammen aus einer Diode, die schwache Lichtblitze aussendet. In jedem Schritt der Schlüsselerzeugung wird nicht ein einzelnes Quantenbit übertragen, sondern eine große Zahl von Photonen. Das stellt ein großes Sicherheitsrisiko dar:

Wenn das Quantenbit mehrfach über den Kanal geschickt wird, ergibt sich für Eve folgende Lauschstrategie. Sie fängt die Photonen ab, die Hälfte

misst sie in der Basis  $B$ , die andere Hälfte in  $B'$ . Wenn bei der Messung in einer Basis zwei verschiedene Ergebnisse entstehen, weiß sie, dass es die falsche ist. Das Ergebnis bezüglich der anderen Basis ist jedoch korrekt – wir vereinfachen die Argumentation hier etwas. Sie erzeugt so viele Photonen, wie sie gemessen hat, präpariert sie in dem ihr bekannten korrekten Zustand und schickt sie an Bob weiter. Ihr Tun wird im weiteren Verlauf nicht aufgedeckt.

Darum hängt die Sicherheit des Verfahrens von der Erzeugung einzelner Photonen ab. Mittlerweile gibt es recht zuverlässige Einzelphotonenquellen, die mit nur sehr geringer Wahrscheinlichkeit mehr als ein Photon erzeugen. Wie dies prinzipiell möglich ist, soll am Beispiel der *parametrischen Fluoreszenz* erläutert werden.

Einzelphotonen-  
quellen

Durchquert ein Photon einen speziellen Kristall, kann dieses in zwei Photonen mit halbiertem Energiegehalt übergehen. Genauer ist dies bei einem optisch nichtlinearen Kristall der Fall – zum Beispiel aus Beta-Bariumborat. Dieses Phänomen tritt mit geringer Wahrscheinlichkeit ein. Dann verlassen die beiden entstehenden Photonen den Kristall in zwei ganz bestimmten Richtungen. Stellt man in einer der Richtungen einen Detektor auf, lässt sich exakt feststellen, wann in der anderen Richtung ein Photon – mit hoher Wahrscheinlichkeit ein einzelnes! – emittiert wird. Die entscheidenden Aspekte: nur ein Bruchteil der eintretenden Photonen spalten sich. Ist dies der Fall, meldet uns das der Detektor.

Parametrische  
Fluoreszenz

Parametrische Fluoreszenz ist aus noch einem anderen Grund interessant. Tritt das oben beschriebene Photonenpaar in einem bestimmten Bereich des Kristalls aus, ist es maximal verschränkt; die Polarisationen sind antikorreliert, der Bell-Zustand  $\Phi^-$  entsteht. Dieser Umstand wird zur Umsetzung von Quantenteleportation, Quantenkryptographie und auch bei Berechnungen mit Clusterzuständen genutzt, siehe Abschnitt 10.5.

Verschränkte  
Bits

Ein einzelnes vollständig isoliertes Atom könnte als Einzelphotonenquelle dienen. Es nimmt nur diskrete Energieniveaus an und kann in einer Weise angeregt werden, dass genau ein Photon emittiert wird. *Quantenpunkte* kann man sich als künstliche Atome vorstellen. Dazu erzeugt man beispielsweise auf einem Halbleiter einen Bereich, in dem die Bewegungsfreiheit eines Elektrons so weit eingeschränkt ist, dass auch dieses nur diskrete Energiewerte annehmen kann.

Quantenpunkte

Schon seit längerem wird auch daran gearbeitet, Quantencomputer auf der Basis von Quantenpunkten zu bauen. Ein Vorteil ist, dass Festkörperphysik und Halbleiter im Computerbau technologisch gewohntes Terrain markieren.

### Praktische Umsetzung

Quantenkryptographiesysteme werden schon seit längerem kommerziell angeboten und es gibt verschiedene Ansätze, dazu polarisierte Photonen einzusetzen. So realisierte Quantenbits können über Glasfaserleitungen übertragen oder bei direkter Sichtverbindung über Teleskope gesendet werden.

Der prinzipielle Aufbau eines solchen Systems lässt sich noch immer gut anhand des folgenden Experiments erläutern: Mit polarisierten Photonen sind im Jahr 2001 Schlüssel über etwa 23 Kilometer übertragen worden. Bei diesem Experiment der Arbeitsgruppe von Harald Weinfurter (LMU

München) kamen kompakte und sehr einfach aufgebaute Geräte zum Einsatz. Den schematischen Aufbau dieses Versuchs zeigt Abbildung 10.6. Auf der linken Seite der Abbildung sieht man Alice' Sendemodul grau hinterlegt. Bobs Empfangsmodul befindet sich auf der rechten Seite, ebenfalls in einer grauen Box. Der Quantenkanal zwischen beiden, verwendet ein astronomisches Teleskop. Die Linsen L2 - L4 und die Spiegel S1 - S4 gehören ebenfalls zu dem Quantenkanal.

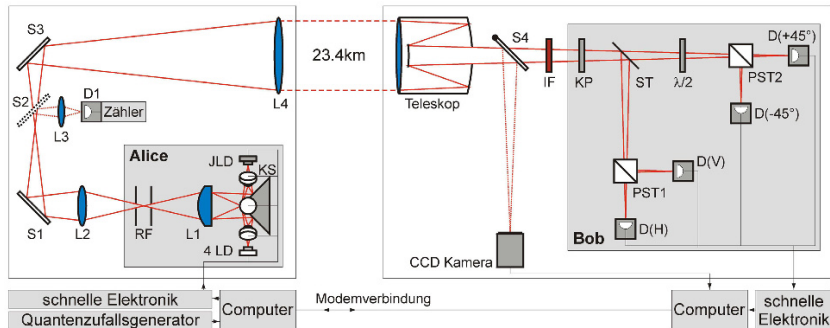


Abbildung 10.6: Übersicht der alpinen Schlüsselübertragung

Alice verwendet vier Laserdioden LD, für jede im BB84-Protokoll verwendete Polarisationsrichtung eine. Dadurch sind keine weiteren polarisierenden Bauteile nötig. Die Steuerung findet über einen Quantenzufallsgenerator statt. Wichtig ist, dass unterschiedlich polarisierte Photonen ansonsten identisch sind, sich also nicht etwa durch die Wellenlänge oder die Emissionsrichtung unterscheiden lassen. Wenn Eve nämlich anhand eines solchen Kennzeichens feststellen könnte, aus welcher der vier Laserdioden ein Photon stammt, könnte sie daraus den Schlüssel ableiten. Der Raumfilter RF verhindert, dass die Richtung eines Photons Informationen über die Quelle enthält. Es wird keine echte Einzelphotonenquelle verwendet. Weniger als fünf Prozent der von Alice erzeugten Quantenbits bestehen aus mehr als einem Photon. Die Information, die Eve aus diesem Umstand gewinnen kann, ist gering. Alice' Aufbau findet in einem Kasten von wenigen Zentimetern Platz.

Ähnlich miniaturisiert ist Bobs Empfangsmodul; Abbildung 10.7 zeigt eine Fotografie seiner Ausrüstung. Gemäß dem BB84-Protokoll wählt Bob zunächst zufällig die Basis, in der er die Quantenbits misst. Das geschieht, indem er die ankommenden Photonen auf einen Strahlteiler ST leitet, wie wir ihn von dem Mach-Zehnder-Interferometer kennen. Der Ausgang, über den ein Photon diesen Strahlteiler verlässt, entscheidet über die Basis, in der gemessen wird. Ein Photon trifft entweder auf den Polarisationsstrahlteiler PST1, der horizontal von vertikal polarisierten Photonen trennt, oder auf PST2, der gemäß den Richtungen 45 Grad/135 Grad teilt. Im ersten Fall vollenden die Detektoren D(H) und D(V) die Messung, im zweiten Fall kommen die Detektoren D(+45), D(-45) zum Einsatz.

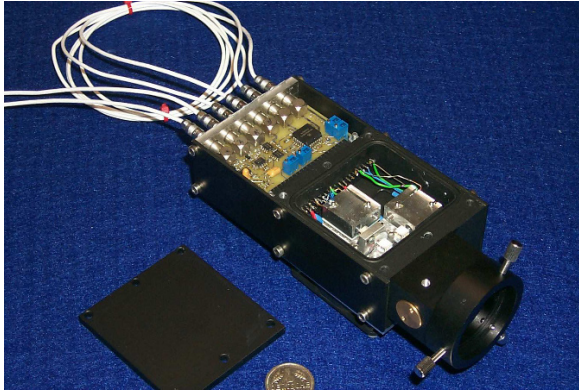


Abbildung 10.7: Bobs Ausrüstung: Miniaturisiertes Empfangsmodul

Seit dem eben beschriebenen Experiment haben sich die Methoden erheblich weiter entwickelt. Die eigentliche Schwierigkeit besteht darin, die Hardware gegen Angriffe zu schützen, welche die Sicherheit des BB84-Protokolls aushebeln.

Angriffe

Ein Beispiel ist die sogenannte *Blendattacke* aus dem Jahr 2010, bei der die spezielle Bauart der Detektoren ausgenutzt wird. Nutzt Bob sogenannte *Lawinenphotodioden*, kann Eve durch Einstrahlen von Licht die Kontrolle über Bobs Empfängermodul übernehmen (siehe [105]). Aufsehen erregte auch der sogenannte After-Gate-Angriff auf ein Gerät der Firma ID Quantique im selben Jahr (siehe [148]). Erwähnt werden sollen hier noch *Phasenverschiebungsangriffe*, siehe [150] und *Zeitverschiebungsangriffe*, siehe [155]. Zur einführenden Lektüre ist auch zu diesem Thema [27] gut geeignet.

Als Zwischenfazit lässt sich sagen: Gegen die bis zur Drucklegung der aktuellen Auflage bekannt gewordenen Angriffe lassen sich die Nachfolgermodelle leicht schützen. Andererseits liegt die Vermutung nahe, dass es noch eine Reihe von bisher nicht veröffentlichten Angriffsmöglichkeiten gibt, welche die eigentliche Gefahr darstellen. Man sollte davon ausgehen, dass die existierenden Quantenkryptographiesysteme Sicherheitslücken aufweisen. Andererseits erfordern diese Angriffe eine anspruchsvolle Ausrüstung und ihre Durchführung ist aufwändig.

## 10.3 Kernspinresonanz

Im Jahr 2001 gelang es der IBM-Forschungsgruppe unter der Leitung von Isaac L. Chuang mit Shors Algorithmus die Zahl 15 in ihre Primfaktoren zu zerlegen. Das Herz des verwendeten Quantencomputers ist ein Fluorkohlenstoff-Molekül, wovon sich eine sehr große Zahl in einer Art Reagenzglas befindet. Aus sieben Quantenbits besteht der damit realisierte Computer.

Diese werden von den einzelnen Atomen des Moleküls realisiert, indem Kernspinresonanz ausgenutzt wird.

Dieses Phänomen (der englische Name lautet *nuclear magnetic resonance*, NMR) entsteht aus der Wechselwirkung von Atomkernen und Magnetfeldern. Befindet sich ein Molekül in einem Magnetfeld, richtet sich der Spin der Atomkerne parallel zu dem Feld aus. Diese Ausrichtung lässt sich durch elektromagnetische Wellen beeinflussen, die senkrecht zu dem ursprünglichen Feld stehen. Nun wählt man einzelne Atome des Moleküls aus und macht sie zu Trägern der Quantenbits. Unterschiedliche chemische Eigenschaften der Umgebung dieser Atome machen die Quantenbits individuell ansprechbar.

Man legt für die Atomkerne, die zu Quantenbits werden sollen, beispielsweise fest: Ausrichtung des Spins entlang dem Hauptmagnetfeld entspricht dem Zustand  $|0\rangle$ , dazu senkrechte Ausrichtung  $|1\rangle$ , wie in Abbildung 10.8 schematisch dargestellt. Mit den die Ausrichtung beeinflussenden oszillierenden Feldern lassen sich nun Gatter realisieren, an denen ein oder zwei Bits beteiligt sind.

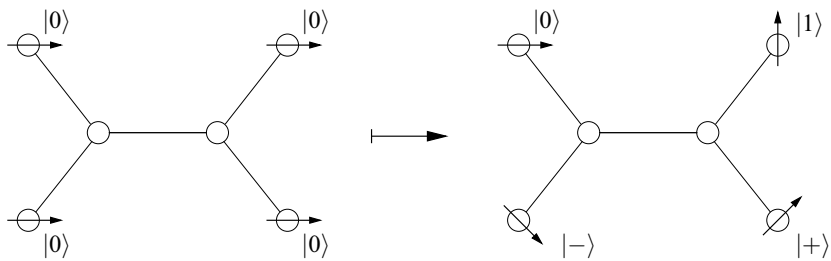


Abbildung 10.8: Ein Molekül mit sechs Atomen, von denen vier als Quantenbits genutzt werden: Die Ausrichtung des Spins bestimmt den Zustand.

Die Zerlegung der Zahl 15 war die bis dahin komplexeste Realisierung eines Quantenverfahrens. Das war auch dadurch möglich, dass die Kernspinresonanz seit ihrer Entdeckung im Jahr 1946 schnell zu einer zuverlässigen Technik mit großem wissenschaftlichen und ökonomischen Nutzen wurde. So ist es ein Standarduntersuchungsverfahren der Chemie, und in der Medizin ist es als Kernspintomographie bekannt. Natürlich gab es zudem viele Vorarbeiten mit einer geringeren Zahl an Quantenbits. So wurden in derselben Forschungsgruppe bereits 1998 zwei Quantenbits an Chloroform-Molekülen erzeugt und für den Algorithmus von Deutsch genutzt.

Trotz der frühen Erfolge wird Kernspinresonanz nach Meinung der Experten nicht zu praktisch verwendbaren Quantencomputern führen, da diese Methode schlecht skalierbar ist, das heißt, die Zahl der Quantenbits lässt sich nur schwer erhöhen. So nimmt die Stärke des Signals mit wachsender Bitzahl exponentiell ab, und es wird immer schwieriger, es vom Rauschen zu unterscheiden.

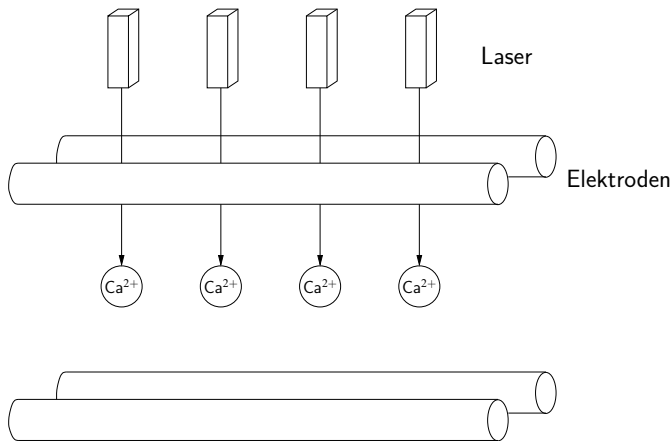


Abbildung 10.9: Vier Kalziumionen in einer elektromagnetischen Falle. Die Laser führen Transformationen aus und lesen den Endzustand.

## 10.4 Ionenfallen

Ionen, also elektrisch geladene Moleküle oder Atome, können mit einem elektromagnetischen Feld an einem bestimmten Ort festgehalten werden: in sogenannten Ionenfallen. Solche Fallen werden zum Beispiel in der Spektrometrie verwendet, mit der sich chemische Substanzen analysieren lassen. Derart gefangene Teilchen zum Bau von Quantencomputern zu verwenden, wurde 1995 von Ignacio Cirac und Peter Zoller von der Universität Innsbruck vorgeschlagen, [38]. So lässt sich dem Grundzustand eines gefangenen Ions der Zustand  $|0\rangle$  zuordnen, und  $|1\rangle$  kann ein energetisch angeregter Zustand sein. Es ist auch möglich, mit einem gefangenen Ion zwei Quantenbits zu realisieren. Der Wert des ersten hängt dann wie beschrieben vom Energiezustand ab, und zusätzlich bilden zwei verschiedene Schwingungszustände die Werte  $|0\rangle$  und  $|1\rangle$  des zweiten Quantenbits. Oft sind die gefangenen Ionen wie in Abbildung 10.9 linear angeordnet und werden mit Laserpulsen adressiert. Dazu sind Temperaturen nahe dem absoluten Nullpunkt nötig, und die Fallen befinden sich in einem Vakuum.

Obwohl die Ionen linear angeordnet sind, lassen sich Zwei-Bit-Gatter auf beliebige Quantenbits anwenden. Geladene Teilchen stoßen sich ab; die Ionen müssen deshalb stark gekühlt werden, damit sie sich in Ruhe befinden. Führt nun der Laser der Kette der gefangenen Ionen Energie zu, gerät diese in Schwingung. Durch Wechselwirkungen kann die Ionenkette als eine Art Daten-BUS benutzt werden, und Interaktionen zwischen je zwei beliebigen Ionen beziehungsweise Quantenbits werden möglich.

Durch frühe Erfolge, wie etwa die Umsetzung des Algorithmus von Deutsch, galt der Ionenfallen-Ansatz als verheißungsvoller Kandidat für den Bau skalierbarer Quantencomputer. Zuletzt wurde mehr über die Erfolge mit

supraleitenden Quantenbits berichtet (Abschnitt 10.6) – Ionenfallen weisen jedoch einige Vorteile auf und stehen weiterhin im Fokus der Forschung. Einen umfangreichen Überblick zu Techniken und Perspektiven bietet [32].

## 10.5 Einwegberechnungen mit Clusterzuständen

Es gibt einen Ansatz zum Bau von Quantencomputern, bei dem der gesamte Quantenschaltkreis – mehr noch: eine große Menge möglicher Schaltkreise – in einer Folge verschränkter Quantenbits kodiert wird. Die Rechenschritte werden durch Messungen an einzelnen Bits durchgeführt. Durch die Wahl der Basen der Messungen wird eine Berechnung auf einer bestimmten Eingabe realisiert. Wir beginnen mit einem Beispiel.

**Beispiel 10.1:** Zwei Quantenbits  $|x\rangle|y\rangle$  befinden sich in dem verschränkten Zustand

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

Wir betrachten zwei verschiedene Messungen des ersten Bits: die erste verwendet die Basis  $B = \{|0\rangle, |1\rangle\}$ , die zweite  $B' = \{|+\rangle, |-\rangle\}$ , mit  $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  und  $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ .

Nach der Messung in der Basis  $B$  befindet sich das Zweibitsystem in einem der Zustände

$$|0\rangle|0\rangle \text{ oder } |1\rangle|1\rangle,$$

das Messergebnis sagt uns, in welchem tatsächlich. Eine Messung in der Basis  $B'$  entspricht einer Messung in  $B$  mit vorheriger Anwendung der Hadamard-Transformation:

$$|\Phi^+\rangle \xrightarrow{H \otimes I} \frac{1}{4}(|00\rangle + |01\rangle + |10\rangle - |11\rangle).$$

Der Zustand nach der Messung des ersten Bits ist

$$\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = |0\rangle|+\rangle$$

oder

$$\frac{1}{\sqrt{2}}(|10\rangle - |11\rangle) = |1\rangle|-\rangle.$$

Durch die Messung am ersten Bit wird das zweite Bit transformiert, die Wahl der Messbasis bestimmt die Transformation. Bit  $|y\rangle$  nimmt zwar einen von zwei möglichen Zuständen zufällig an, wir erfahren diesen jedoch aus der Messung am ersten Bit und kennen damit den Zustand des zweiten Bits.

Dieses äußerst einfache Beispiel verdeutlicht, wie mit Messungen an einem Bit Transformationen an einem weiteren mit diesem verschränkten Bit möglich sind. Von Hans Briegel und Robert Raussendorf stammt das Modell eines Quantencomputers, der sich das zu Nutze macht, siehe etwa [124], [125].



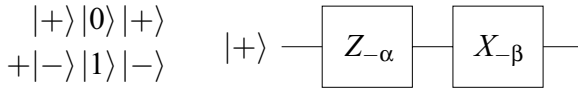


Abbildung 10.10: Ein-Bit-Schaltkreis; realisiert mit drei verschränkten Bits

Man spricht von Einweg-Quantenberechnungen. Durch aufeinanderfolgende Messungen an miteinander verschränkten Quantenbits wird eines nach dem anderen entfernt. Eine solche Berechnung ist nicht umkehrbar.

Der Schaltkreis in Abbildung 10.10 wird durch Messungen an dem verschränkten Zustand dreier Bits auf der linken Seite verwirklicht. Diese *physikalischen* Bits sind von den *kodierten* Bits zu unterscheiden: gerechnet wird auf einem Quantenbit  $|x\rangle$  im Startzustand  $|+\rangle = H|0\rangle$ . Messungen an den beiden ersten physikalischen Bits entsprechen Transformationen des kodierten Bits  $|x\rangle$  mittels

$$Z_{-\alpha} = \begin{pmatrix} e^{i\alpha/2} & 0 \\ 0 & e^{i\alpha/2} \end{pmatrix} \text{ und danach } X_{-\beta} = \begin{pmatrix} \cos \frac{\beta}{2} & \sin \frac{\beta}{2} \\ \sin \frac{\beta}{2} & \cos \frac{\beta}{2} \end{pmatrix}.$$

$X_{\beta}$  und  $Z_{\alpha}$  sind Drehungen im Zustandsraum des Quantenbits. So ist

$$Z_{2\pi} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \text{ und } X_{\pi} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = X.$$

Messen wir das erste physikalische Bit des verschränkten Drei-Bit-Zustands  $(|+\rangle|0\rangle|+\rangle) + (|-\rangle|1\rangle|-\rangle)$  in der Basis

$$B(\alpha) = \left\{ \frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - e^{i\alpha}|1\rangle) \right\},$$

bewirkt dies die Anwendung von  $Z_{-\alpha}$  auf das kodierte Bit im Anfangszustand  $|+\rangle$ . Messen wir danach das zweite Bit bezüglich  $B(\beta)$ , wird anschließend  $X_{-\beta}$  angewendet. Das dritte Bit befindet sich dann im Zustand  $X_{-\beta}Z_{-\alpha}|+\rangle$ . Messen wir dieses, erhalten wir das Ergebnis der Berechnung.

Die vorhergehende Darstellung enthält noch eine Vereinfachung: denn tatsächlich bringt jede Messung ein Zufallselement in die Berechnung, wie wir in Beispiel 10.1 gesehen haben. Die Berechnung wird nur dann wie beschrieben und erwünscht ausgeführt, wenn das Ergebnis beider Messungen dem ersten Basisvektor  $\frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle)$  – beziehungsweise  $\frac{1}{\sqrt{2}}(|0\rangle + e^{i\beta}|1\rangle)$  – entspricht. Im anderen Fall berücksichtigen wir die entstehende Abweichung für den Fortgang der Rechnung. Dabei ist es möglich, die folgenden Messungen an die vorherigen Messergebnisse anzupassen. Dieses Vorgehen nennt man *feed forward*, siehe [123]. Tatsächlich lassen sich auf diese Weise durch Messungen an jeweils einem Quantenbit universelle Quantenberechnungen realisieren. Ausführliche Einführungen findet der Leser in [86] und [41].

Eine Menge von möglichen Zweibit-Schaltkreisen stellt Abbildung 10.11 dar. Messungen an den Bits 1 und 4 führen die im Schaltkreis angegebenen Transformationen auf zwei kodierten Bits im Zustand  $|+\rangle$  aus. Das

Schaltkreise

Feed forward

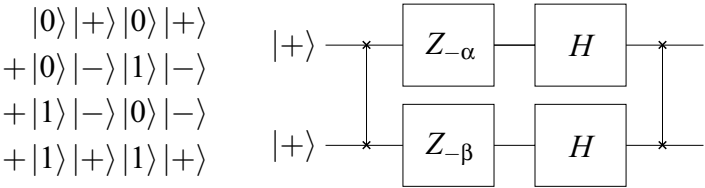


Abbildung 10.11: Zwei-Bit-Schaltkreis aus vier verschränkten Bits

Ergebnis der Transformationen ist der Zustand der Bits 2 und 3. Warum werden gerade die Bits 1 und 4 gemessen? Das resultiert aus der Erzeugung der verschränkten Quantenzustände. Dazu wird auf je zwei benachbarte Bits eine Transformation der Form  $|j\rangle|k\rangle \mapsto (-1)^{jk}|j\rangle|k\rangle$  ausgeführt. Die Nachbarschaftsstruktur der eben beschriebenen vier Bits zeigt Abbildung 10.12.

Realisierung mit Photonen

Im März 2005 berichtet *Nature*, [146], dass in Anton Zeilingers Wiener Arbeitsgruppe Einweg-Quantenberechnungen experimentell realisiert wurden. Grovers Algorithmus wurde mit einer vierelementigen Datenbank als Eingabe ausgeführt. Insbesondere gelang es, eine universelle Menge von Gattern aus jeweils vier verschränkten Quantenbits zu realisieren. Diese bestanden jeweils aus einem Cluster von vier polarisierten Lichtquanten.

Ein solches Cluster wird aus vier gleichzeitig emittierten Photonen hergestellt. Mit einem Puls laser und einem Bariumborat-Kristall werden durch parametrische Fluoreszenz zwei verschränkte Paare erzeugt. Optische Instrumente wie zum Beispiel Polarisationsstrahlteiler werden verwendet, um den gewünschten Clusterzustand herzustellen. Abschließend werden die vier Quantenbits des Clusters auf vier Einzelphotonendetektoren geleitet. Wie oben beschrieben wurde, vollzieht sich die eigentliche Berechnung durch diese Messungen.

Grovers Algorithmus

Bei den meisten in diesem Buch vorgestellten Algorithmen besteht die Eingabe in einem Quantenorakel, das ein Teil der Berechnung, beziehungsweise des Quantenschaltkreises, wird. In diesem Sinne legt die Wahl der Messungen neben der auszuführenden Berechnung auch die Eingabe fest. Betrachten wir noch einmal den Schaltkreis in Abbildung 10.11. Durch korrekte Wahl der Parameter  $\alpha$  und  $\beta - (\alpha, \beta) \in \{0, \pi\}^2$  – können die beiden Z-Gatter das Datenbankorakel realisieren. Die beiden Hadamard-Gatter bilden die linke Hälfte der Spiegelung am Mittelwert, siehe Abschnitt 6.2. Abschließende

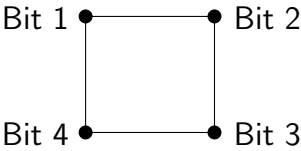


Abbildung 10.12: Nachbarschaftsstruktur zu Abbildung 10.11

unitäre Transformationen werden in die Messung des Ergebnisses integriert. Mit vier entsprechend verschränkten Bits ist Grovers Algorithmus realisierbar, was mit dem Wiener Photonenrechner gelungen ist.

Mit diesem Ansatz wurden außerdem die Algorithmen von Deutsch und von Deutsch-Jozsa umgesetzt. 2013 gelang es, Simons Algorithmus zu implementieren [141]; wie in Kapitel 8 beschrieben, lässt sich dieser als binäre Variante von Shors Algorithmus ansehen. Des Weiteren wurde ein Verfahren aus der Spieltheorie implementiert, das hier bisher nicht erwähnt wurde. Die klassische Spieltheorie hat weitreichende Anwendungen in der Ökonomie und den Biowissenschaften. Das *Gefangendilemma* ist ein Spiel mit zwei Teilnehmern, das unter der ökonomischen Grundannahme der individuellen Nutzenmaximierung zu einem nicht optimalen Ergebnis führt. Sehen beide Spieler nur ihren eigenen Nutzen, ist das Ergebnis für beide schlechter, als wenn sie „uneigennützig“ kooperierten. Quantenversionen von Spielen weisen häufig andere Eigenschaften auf, als die klassischen Gegenstücke. So löst sich in der Quantenversion des eben besagten Spieles das Dilemma. An dieser Stelle ist interessant, dass das Gefangendilemma mit dem oben beschriebenen photonenbasierten Quantenrechner implementiert werden konnte [122]. Weiteres zu Quantenspielen findet der Leser in [69] oder [99].

Weitere  
Algorithmen,  
Quantenspiele

Sie überantworten ein aufwändiges Rechenproblem der Cloud und schicken Ihre Daten und die Aufgabenstellung an einen Server. Sie erhalten die Lösung, wobei diese Antwort, Ihre Daten und auch Ihre Anfrage geheim bleiben. Der Server hat keine Möglichkeit, etwas darüber zu erfahren oder zu speichern. Nach den Gesetzen der Quantenmechanik ist dieses Szenario möglich: das Verfahren nennt sich *Blind Quantum Computing*. Zugrunde liegen Einwegberechnungen: allein durch Messungen an Qubitbits in geeigneten Clusterzuständen können universelle Quantenberechnungen umgesetzt werden. Verschiedene Algorithmen entsprechen verschiedenen Mustern solcher Messungen. Das in [31] präsentierte Protokoll nutzt sogenannte blinde Clusterzustände, die nichts über die verwendeten Messungen verraten. 2012 gelang die Umsetzung dieses Konzepts mit photonenbasierten Clusterzuständen, siehe [11]. Experimentell konnten Anfragen an die Algorithmen von Deutsch und Grover umgesetzt werden.

Blind Quantum  
Computing

## 10.6 Supraleiter

In ringförmigen Supraleitern kann ein Strom verlustlos fließen. Und es treten Quantenphänomene auf, auch wenn solche Ringe – typischerweise einige Mikrometer groß – unserer makroskopischen Welt entstammen. Dies macht Supraleiter zu Kandidaten für die Realisierung von Qubitbits.

Es gibt verschiedene Ansätze, Qubits aus Supraleitern zu konstruieren. Wir beschäftigen uns hier beispielhaft mit dem sogenannten *Flux-Qubit*. Es besteht aus einem durch *Josephson-Kontakte* (siehe unten) unterbrochenen ringförmigen Supraleiter, für den eine Temperatur nahe dem absoluten Nullpunkt erzeugt werden muss. Mit einem von außen angebrachten elektrischen Fluss (engl. *flux*), kann ein zirkulierender Dauerstrom erzeugt werden.

Die beiden Bitzustände 0 und 1 werden durch den Ringstrom realisiert (in Uhrzeigerrichtung bzw. entgegen dieser).

Exkurs:  
Wellenfunktion

Entscheidend ist, dass die Richtung dieses Suprastroms in Superposition gebracht werden kann. Wie ist das möglich? An dieser Stelle ist ein Exkurs nötig: Wir haben uns bisher intensiv mit Superpositionen von Qubit-Zuständen beschäftigt. Wollen wir allerdings in einem physikalischen System Eigenschaften eines Teilchens beschreiben, z.B. dessen Ort, gibt es häufig nicht nur zwei sondern kontinuierlich viele Möglichkeiten. In einer quantenphysikalischen Beschreibung erhält jede dieser Möglichkeiten eine Amplitude: die Eigenschaft wird durch eine Welle beschrieben.<sup>1</sup> Bei einer Messung wird diese Welle zerstört. Das Amplitudenquadrat beschreibt die Wahrscheinlichkeit, einen bestimmten Zustand zu beobachten, z.B. ein Teilchen an einem bestimmten Ort aufzufinden. Die zeitliche Veränderung dieser sogenannten Wellenfunktion wird durch die Schrödingergleichung beschrieben oder durch unitäre Transformationen. Somit verallgemeinert diese Beschreibung unser Qubitmodell.

Tunneleffekt

Damit lässt sich der sogenannte Tunneleffekt beschreiben. Stellen wir uns vor, ein Quantenteilchen befindet sich gegenüber einer Barriere. Diese ist zwar nicht sehr dick, aber in klassischer Beschreibung trotzdem ein unüberwindbares Hindernis für das Teilchen. In der quantenmechanischen Beschreibung ist jedoch die Amplitude auch jenseits der Barriere nicht gleich null: die Barriere kann *durchtunnelt* werden. Mit einer gewissen kleinen Wahrscheinlichkeit kann bei einer Messung das Teilchen dort gefunden werden. Allerdings nimmt diese Wahrscheinlichkeit mit wachsendem Abstand exponentiell ab. Darum sollte die Barriere nicht zu dick sein, möchte man den Tunneleffekt beobachten.

Mit dem eben eingeführten Begriff der Welle lässt sich auch beschreiben, wie ein makroskopischer Supraleiter Quanteneigenschaften annehmen kann. Gemäß der BCS-Theorie, der klassischen Erklärung für Supraleiter nahe dem absoluten Nullpunkt, verbinden sich je zwei Elektronen zu einem sogenannten Cooper-Paar. Zudem bilden alle Cooper-Paare *eine* kohärente – quasi gemeinsame – Wellenfunktion aus. Das heißt, sie verhalten sich wie ein Teilchen, es gibt keine gegenseitigen Beeinflussungen. Solche Cooper-Paare können den oben erwähnten Josephson-Kontakt durchtunneln. Ein Josephson-Kontakt entsteht, indem zwei Supraleiter durch einen sehr dünnen Isolator oder Normalleiter unterbrochen werden. Für das Flux-Qubit wird der Kontakt so entworfen, dass ein Dauerstrom fließen kann, sobald der externe magnetische Fluss wirkt; Cooper-Paare durchtunneln dann den Josephson-Kontakt.

SQUID

Messen lässt sich ein Flux-Qubit mit einem sogenannten SQUID, einem *superconducting quantum interference device*. Die SQUID-Technologie wird seit Jahrzehnten erfolgreich für die Messung äußerst geringer Magnetfelder verwendet, zum Beispiel zur Untersuchung von Hirnströmen. So ein SQUID besteht wiederum aus einem durch Josephson-Kontakte unterbrochenen Supraleiter. Wenn der durch ein Magnetfeld induzierte SQUID-Fluss einen bestimmten Betrag übersteigt, kann an den Kontakten eine Spannung ermittelt werden.

<sup>1</sup>Unsere binären bzw. diskreten Qubit Zustände sind ein Spezialfall davon.

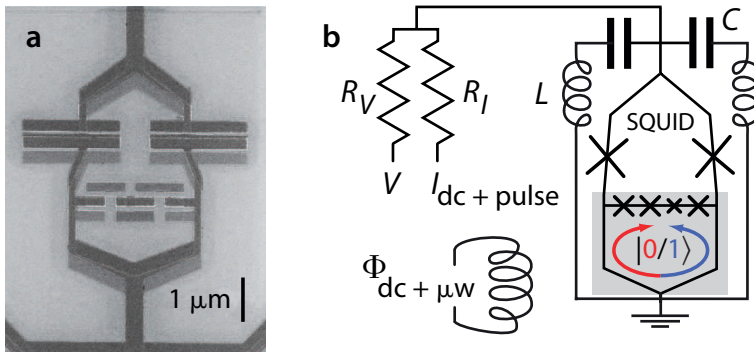


Abbildung 10.13 zeigt schematisch den Aufbau des in [35] beschriebenen Flux-Qubits. Die Kreuze markieren Josephson-Kontakte. Mit Hilfe dieser Kontakte lassen sich die Eigenschaften des Systems von außen einstellen. Man erkennt den Qubit-Ring als Träger der Zustände  $|0\rangle$  und  $|1\rangle$  und das gekoppelte SQUID. Der wenige Mikrometer große Supraleiter besteht aus Aluminium und befindet sich auf einer Halbleiterplatine. Ein Vorteil der Verwendung von Supraleitern für den Bau von Quantencomputern ist, dass man auf bewährte Technologien zur Fertigung zurückgreifen kann. Wie bei klassischen Rechnern lässt sich eine Vielzahl von Qubits auf einer Platine platzieren.

Einen umfangreichen Überblick zur Realisierung von Quantenbits mit Supraleitern bietet [94].

# 11 Ausblick: Optimieren

*Das Optimum stellt sich ein bei einer bestimmten Mischung von Organisation und Chaos.*

Hermann Müller-Thurgau

Die beiden folgenden Abschnitte sollen einen ersten Eindruck von adiabatischen Quantencomputern und Quantum Annealing vermitteln. Im Abschnitt 11.2 zum letztgenannten Verfahren wird vielen einfachen und konkreten Beispielen der Vorzug vor tiefergehenden abstrakten Konzepten gegeben. Einen wichtigen Teil bilden dabei die Übungsaufgaben. Im Idealfall wird dem Leser ohne Vorkenntnisse eine erste Einschätzung der Anwendungsbereiche dieser Technik ermöglicht und der Zugang zu weiterführender Literatur erleichtert.

Es sei darauf hingewiesen, dass es weitere, hier nicht behandelte Ansätze gibt, Optimierungsprobleme mit Quantencomputern zu lösen, beispielhaft sei hier der *Quantum Approximate Optimization Algorithm* (QAOA) genannt [58].

## 11.1 Adiabatische Quantencomputer

In diesem Abschnitt lernen wir ein Verfahren kennen, das unserem Bild des Quantencomputers eine gänzlich neue Facette hinzufügt. Bisher rechnen wir mit Quantenbits, indem wir eine Folge von lokalen unitären Transformationen als Rechenschritte anwenden. Quantenalgorithmen können wir somit als Schaltkreis ausdrücken, wie wir es bisher durchgängig getan haben. Ein solcher *gatterbasierter* Quantencomputer kommt unserem Bild eines klassischen Computers, wie es in Abschnitt 2.1.2 beschrieben wird, sehr nahe. Es gibt einen alternativen Ansatz, der für Nicht-Physiker weniger anschaulich ist, allerdings technisch leichter zu realisieren sein könnte. Vorab eine erste Annäherung: Beim *adiabatischen Quantum Computing* wird die Problemstellung in die energetische Struktur eines Quantensystems übersetzt. Durch

Hamiltonoperator

allmähliche Änderung des Systems wird ein Zielzustand erreicht, aus dem die Lösung ablesbar ist.

Die Zeitentwicklung eines Quantensystems lässt sich mit der Schrödingergleichung

$$\frac{d|\psi\rangle}{dt} = -\frac{i}{\hbar}\mathcal{H}|\psi\rangle$$

beschreiben. Die Änderung des Zustands  $|\psi\rangle$  (linker Teil) ergibt sich im wesentlichen durch die Anwendung des Operators  $\mathcal{H}$  auf  $|\psi\rangle$ . Dabei bezeichnet  $\hbar$  das sogenannte reduzierte Plancksche Wirkungsquantum. Uns interessiert hier  $\mathcal{H}$ , der sogenannte *Hamiltonoperator*. Dieser beschreibt die Gesamtenergie aus kinetischer und potentieller Energie und deren Struktur. Wollen wir ein System aus  $n$  Qubits beschreiben, kann  $\mathcal{H}$  durch eine  $2^n$ -dimensionale Matrix beschrieben werden.<sup>1</sup> Nun entwickelt sich ein Quantensystem gemäß unitärer Transformationen, wo ist hier der Zusammenhang? Die Schrödingergleichung ist eine Differentialgleichung: eine Funktion wird mit Ihrer Ableitung in Beziehung gesetzt. Lösen wir die Gleichung, erhalten wir die uns vertraute unitäre Transformation.

Der Operator  $\mathcal{H}$  selbst beschreibt, wie bereits erwähnt, die Struktur der Gesamtenergie. Diese kann sich natürlich im Laufe der Zeit ändern oder gezielt verändert werden. Bei adiabatischen Berechnungen wird genau dies getan. Man versetzt ein System in seinen *Grundzustand*: das ist der Zustand mit der geringstmöglichen Energie. Nach dem adiabatischen Theorem der Quantenmechanik verbleibt das System in diesem minimalen Grundzustand und wechselt nicht in einen angeregten Zustand, sofern die Veränderung *allmählich* vor sich geht.

Für eine Berechnung beginnen wir mit einem einfachen Hamiltonoperator  $\mathcal{H}_{\text{Start}}$ . Wir kennen dessen Grundzustand und können diesen leicht herstellen. Zum Beispiel könnte  $\mathcal{H}_{\text{Start}}$  so gewählt werden, dass für den Grundzustand jedes Bit den Wert  $1/\sqrt{2}(|0\rangle + |1\rangle)$  erhält. Das Ziel der Berechnung ist ein komplexer Operator  $\mathcal{H}_{\text{Lösung}}$ : Dessen uns unbekannter Grundzustand kodiert die Lösung unserer Problemstellung. Wir beginnen im Grundzustand von  $\mathcal{H}_{\text{Start}}$ , variieren allmählich in Richtung  $\mathcal{H}_{\text{Lösung}}$  ohne jemals den Zustand geringstmöglicher Energie zu verlassen. Am Ende können wir die Lösung auslesen.

3-SAT

An einem Beispiel wollen wir diese Idee konkretisieren. Adiabatisches Quantum Computing wurde zuerst in [57] beschrieben, wo auch die Optimierungsvariante des wichtigen NP-vollständigen Problems 3-SAT behandelt wird. Dabei ist zu entscheiden, ob eine Boolesche Formel, zum Beispiel  $(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3)$  eine *erfüllende Belegung* der Variablen besitzt. Eine Belegung legt für jede der Variablen einen der Werte 0 oder 1 fest. Diese Werte werden in die Formel eingesetzt und nach den Regeln der Logik ausgewertet. Ist das Ergebnis 1, so ist die Belegung erfüllend. In unserem Beispiel führt zum Beispiel die Belegung  $x_1 \leftarrow 1, x_2 \leftarrow 0, x_3 \leftarrow 1$  zu dem Ergebnis  $(1 \vee 0 \vee 0) \wedge (0 \vee 1 \vee 1) \wedge (0 \vee 0 \vee 1) = 1$  und ist also erfüllend. In

<sup>1</sup>Qubits sind ein Spezialfall, da physikalische Eigenschaften in der Regel nicht nur zwei, sondern kontinuierliche Werte annehmen können.

Problemstellungen von 3-SAT werden stets drei Variablen oder deren Negation mittels logischem OR zu einer *Klausel* verbunden. In der Gesamtformel werden diese Klauseln durch AND verknüpft. Damit muss eine erfüllende Belegung jede Klausel erfüllen. Eine einzelne Klausel zu untersuchen ist einfach. Aufwändig ist es im Allgemeinen, eine Lösung zu finden, die alle Klauseln gleichzeitig erfüllt.

Wir untersuchen eine Formel, die auf den Variablen  $x_1, \dots, x_n$  definiert ist und aus den Klauseln  $K_1, \dots, K_m$  besteht. Wir verwenden  $n$  Quantenbits  $b_1, \dots, b_n$ : jedes repräsentiert die Belegung einer Variable. Für jede Klausel definieren wir eine *Energiefunktion*  $h_{K_i}$  auf den Qubits. Diese wird 0, wenn die Qubitbelegung die Klausel erfüllt, und ansonsten 1. Dadurch erhöht jede nichterfüllte Klausel die Energie. Wir summieren die einzelnen Funktionen und erhalten

$$h(b_1, \dots, b_n) = \sum_{i=1}^m h_{K_i}(b_1, \dots, b_n).$$

Also ist  $h$  genau dann 0, wenn alle Klauseln erfüllt werden. Im nächsten Schritt ist ein Hamiltonoperator  $\mathcal{H}_{\text{Lösung}}$  zu konstruieren, dessen Energie der Funktion  $h$  entspricht; wir verweisen auf [57].

Zu Beginn der Berechnung initialisieren wir unsere Qubits mit dem bekannten Grundzustand von  $\mathcal{H}_{\text{Start}}$  und ändern den Hamiltonoperator gleichförmig in Richtung  $\mathcal{H}_{\text{Lösung}}$ :

$$\mathcal{H}(t) = \left(1 - \frac{t}{T}\right) \cdot \mathcal{H}_{\text{Start}} + \frac{t}{T} \cdot \mathcal{H}_{\text{Lösung}}.$$

Zum Zeitpunkt  $t$  wird das System von  $\mathcal{H}(t)$  beschrieben, wir beginnen in  $t = 0$  und erreichen unser Ziel zum Zeitpunkt  $t = T$ . Die Laufzeit  $T$  ist so zu wählen, dass das System permanent im Zustand der geringstmöglichen Energie verbleibt; mehr dazu im folgenden Abschnitt. Denn der unbekannte Grundzustand von  $\mathcal{H}_{\text{Lösung}}$  kodiert unser Ergebnis: Messen wir nach Abschluss der Entwicklung die Quantenbits, erhalten wir eine erfüllende Belegung. Sollte keine existieren, so wird die Anzahl der erfüllten Klauseln maximiert. Welcher Fall vorliegt, lässt sich durch einfaches Einsetzen in die Formel testen. Dieses Vorgehen löst die Optimierungsvariante von 3-SAT. Viele Probleme lassen sich so umformulieren, dass die Lösung als Minimierung einer Energiefunktion ausgedrückt werden kann.

Aber ist dieser Algorithmus auch effizient? Der Begriff *allmählich* könnte uns misstrauisch machen. Verändern wir das System zu schnell, kann dies zu einem Energiezuwachs führen und das System wechselt in einen angeregten Zustand. Die maximale Geschwindigkeit hängt davon ab, wie groß der Unterschied zwischen dem Grundzustand und dem nächstniedrigeren angeregten Zustand ist. Typischerweise gilt: Je schwieriger die Aufgabenstellung und je größer die Eingabe, desto geringer sind die Abstände zwischen den Energiezuständen und desto langsamer müssen wir das System ändern. Bei der Laufzeitanalyse ist zu untersuchen, ob sich die Energieunterschiede polynomial oder exponentiell verringern. So besitzt zum Beispiel Grovers Algorithmus auch als adiabatische Version Laufzeit  $O(\sqrt{N})$ , [129].

Laufzeit



Universalität

In [5] wird gezeigt, dass jeder Quantenalgorithmus im Schaltkreismodell in einen adiabatischen Algorithmus umgeformt werden kann. Der mögliche Zuwachs an Laufzeit ist dabei polynomial beschränkt. Es lässt sich folgern, dass adiabatisches Quantum Computing universell ist.

## 11.2 Quantum Annealing

In diesem Abschnitt geht es nicht mehr um Universalität, sondern darum, diskrete Optimierungsprobleme zu approximieren. Eine intensiv untersuchte Metaheuristik auf Quantencomputern ist *Quantum Annealing*.<sup>2</sup> Kein Zufall ist die Namensähnlichkeit mit dem klassischen *Simulated Annealing* dessen Grundidee wir kurz motivieren wollen.

Uns interessieren Optimierungsprobleme mit einem großen Suchraum und einer effizient zu bestimmenden Zielfunktion. Um den Einstieg intuitiver zu gestalten, erläutern wir einige Begriffe am Beispiel TSP, Seite 111. Beim Problem des Handlungsreisenden suchen wir in der Menge aller möglichen Routen zwischen  $n$  Städten, und die Summe der Abstände auf so einer Rundreise definiert die zu minimierende Zielfunktion.

Lokale Suche

Nun beginnen wir mit einer ausgewählten Rundreise, also einer Anordnung aller  $n$  Städte  $(v_1, \dots, v_n)$ . Diese könnte nach bestimmten Kriterien oder rein zufällig gewählt sein. An dieser nehmen wir eine *lokale Änderung* vor und gehen zu einer *benachbarten Lösung* über. So könnten wir die Position einer Stadt  $v_j$  ändern:  $(v_1, \dots, v_{i-1}, v_j, v_{i+1}, \dots, v_{j-1}, v_{j+1}, \dots, v_n)$ . Wird dadurch die Route kürzer, fahren wir mit dieser neuen Version fort.

Bei der 2-Opt-Heuristik besteht eine lokale Änderung darin, wie in Abbildung 11.1 zwei Kanten der Route  $(v_1, \dots, v_n)$  zu ändern, um die Teilroute zwischen diesen umzukehren:  $(v_1, \dots, v_i, v_j, v_{j-1}, \dots, v_{i+2}, v_{i+1}, v_{j+1}, \dots, v_n)$ .

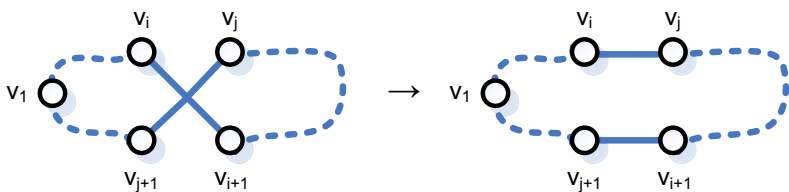


Abbildung 11.1: Zur 2-Opt-Heuristik.

Ähnlich kann man für eine sehr große Klasse von Problemstellungen vorgehen. Dabei ist für eine solche *lokale Suche* eine Nachbarschaftsstruktur nötig: Zu jedem Element des Lösungsraums sind die Nachbarn zu definieren, also die durch lokale Änderungen erreichbaren Elemente. Liegt die Nachbarschaftsstruktur fest, gibt es unterschiedliche Strategien, das Minimum zu finden. So könnten wir wiederholt unter *allen* Elementen in der Nachbarschaft

<sup>2</sup>Eng verbunden ist diese Technik mit dem Hardware-Hersteller D-Wave Systems, siehe Seite 291.

unserer aktuellen Lösung das beste ermitteln. Oder wir folgen immer wieder dem Minimum über eine zufällig gewählte Teilmenge der Nachbarn.

Die Güte leidet nun, wenn wir uns dadurch nicht mehr verbessern können, es aber in unserem Beispiel eine noch kürzere Rundreise gibt. Betrachten wir dazu Abbildung 11.2. Hier soll nicht die Zielfunktion über einem diskreten Suchraum dargestellt werden, sondern ein sofort erkennbares globales Minimum – nach diesem suchen wir – und mehrere lokale Minima. Das Problem unseres obigen Ansatzes ist also die Möglichkeit, in ein solches lokales Minimum zu geraten und durch lokale Operationen nicht besser zu werden, also dort festzustecken.

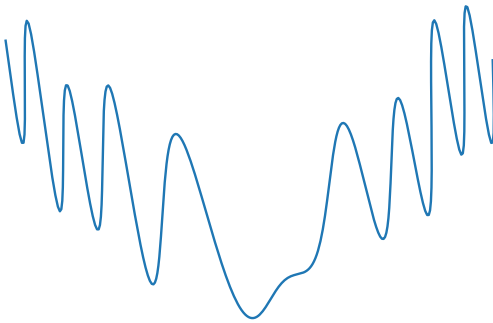


Abbildung 11.2: Finde das globale Minimum!

Gern wird das Bild einer hügeligen Landschaft bemüht. Es gibt Gipfel, Plateaus und Täler. Wir suchen nun das tiefste Tal; dieses entspricht dem globalen Minimum unseres Optimierungsproblems. Lassen wir eine Kugel von einem Hügel hinabrollen, dann verringert sich seine potentielle Energie. Die Kugel landet in einem Tal. Aber in der Regel nicht im tiefsten Tal sondern in einem lokalen Minimum.

Ein naheliegender Ausweg wäre, die lokale Suche häufig mit neuen Eingaben, zum Beispiel Zufallsrouten, zu wiederholen. Bewährt hat sich auch das folgende naturanaloge Vorgehen.

Aus der Metallherstellung ist bekannt: Wird flüssiges Metall schnell abgekühlt, hat das Resultat andere Eigenschaften als beim kontrollierten, langsamen Abkühlen. Jeder Anordnung der Atome in einer Gitterstruktur entspricht eine Energie. Die Gitterstruktur mit dem niedrigsten Energiewert hat oft erstrebenswerte Eigenschaften und ist durch langsames Abkühlen leichter zu erreichen. Mit einer bestimmten Wahrscheinlichkeit, die von der Temperatur abhängt, kann bei diesem Anordnungsprozess nämlich zu einem Zustand mit höherer Energie zurückgekehrt werden. Kurz: Je höher die Temperatur, desto höher das Änderungspotential. In unserer Terminologie: Beim langsamen Abkühlen kann ein lokales Energieminimum wieder verlassen werden. Die Wahrscheinlichkeit dafür verringert sich im Verlauf des Abkühlens.

*Simulated Annealing* nun ahmt diesen Prozess nach ([92]). Wird durch eine Vertauschung wie oben, die Route länger, kann diese mit einer bestimmten Wahrscheinlichkeit trotzdem zu unserer neuen Zwischenlösung

Simulated  
Annealing

werden. Diese Wahrscheinlichkeit sinkt im Laufe des Verfahrens, wobei die Verhältnisse beim kontrollierten Abkühlen nachgeahmt werden. Die Qualität einer Approximation hängt von verschiedenen Faktoren ab. Zum einen davon, wie viel Zeit wir dem Annealing-Prozess einräumen. Aber auch davon wie gut das zu lösende Problem an das Verfahren adaptiert werden konnte.

Beim Quantum Annealing (QA) werden nun die temperaturabhängigen Änderungen durch den Quanteneffekte ersetzt, in erster Linie durch den Tunneleffekt ([87]). Aus einem lokalen Minimum heraustunneln und sich dabei idealerweise verbessern, ohne dafür zuvor die Energie erhöhen zu müssen: das könnte zu einer Beschleunigung führen. Die Rolle der Temperatur im Simulated Annealing übernimmt hier die Stärke eines Magnetfelds, mit dem sich die Wahrscheinlichkeit für Quanteneffekte steuern lässt. Nach bisherigen Erfahrungen ist es von Vorteil, wenn die zu durchquerenden Energiebarrieren eher hoch und schmal sind, siehe dazu [43]. In der Folge wollen wir konkret werden und die ersten Schritte zur Anwendung des QA beschreiben.

Wer mit einem Quanten-Annealer ein Optimierungsproblem lösen will, muss dieses zuvor in eine geeignete Form bringen. Üblich sind das Ising-Modell oder alternativ eine Darstellung als QUBO-Problem. Wir beginnen mit letzterer Darstellungsform: Der Zugang könnte leichter fallen, da solche über die gewohnten Bitwerte definiert sind.

## QUBO

*Quadratic Unconstrained Binary Optimization* (QUBO) minimiert oder maximiert quadratische Gleichungen mit binären Variablen. Im Fall von drei Variablen  $x_1, x_2, x_3 \in \{0, 1\}$  ist die allgemeine Form

$$a_1x_1 + a_2x_2 + a_3x_3 + b_{1,2}x_1x_2 + b_{1,3}x_1x_3 + b_{2,3}x_2x_3. \quad (11.1)$$

Und für  $x_1, \dots, x_n \in \{0, 1\}$  wäre

$$\sum_{i=1}^n a_i x_i + \sum_{i < j \leq n} b_{i,j} x_i x_j \quad (11.2)$$

zu minimieren oder zu maximieren. Da  $x_i^2 = x_i$  wäre  $\sum_{i \leq j \leq n} b_{i,j} x_i x_j$  mit  $b_{i,i} = a_i$  damit äquivalent, aber für unsere Zwecke etwas unhandlich.

Zur Veranschaulichung wollen wir solche Gleichungen aus einem bekannten Graphproblem ableiten. Ein Problem mit NP-vollständiger Entscheidungsvariante, das sich sehr natürlich in ein QUBO-Problem überführen lässt, ist MAX-CUT.

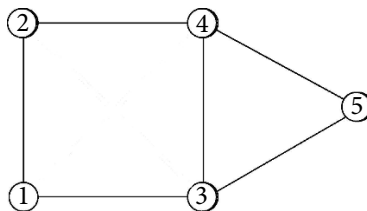


Abbildung 11.3: Finde den maximalen Schnitt!

**Aufgabe 11.1:** Betrachten Sie den Graphen in Abbildung 11.3. Teilen Sie die Knoten  $\{1, 2, 3, 4, 5\}$  so in zwei Mengen  $A$  und  $B$  auf, dass die Zahl der Kanten, die von  $A$  nach  $B$  führen maximiert wird (nach [68], wie auch die weitere Behandlung).

MAX-CUT fragt nun für einen beliebigen Graphen  $G = (V, E)$  nach einem maximalen Schnitt: Nach einer Zerlegung von  $V$  in zwei Mengen, so dass die Zahl der kreuzenden Kanten maximiert wird.<sup>3</sup>

MAX-CUT

Wir führen für jeden Knoten eine binäre Variable ein, also für  $V = \{1, \dots, n\}$  die Variablen  $\{x_1, \dots, x_n\}$ . Wir statten diese mit folgender Bedeutung aus:  $x_i = 1$  heißt, dass Knoten  $i$  in  $A$  liegt, dagegen befindet sich dieser Knoten in  $B$ , falls  $x_i = 0$ . Betrachten wir für eine Kante  $(i, j)$  den Term  $x_i + x_j - 2x_i x_j$ . Dieser ist gleich 0 für  $x_i = x_j$ , also falls  $i$  und  $j$  in der gleichen Menge liegen und Kante  $(i, j)$  den Schnitt nicht kreuzt. Und gleich 1 für den Fall  $x_i \neq x_j$ :  $i$  und  $j$  liegen in verschiedenen Mengen oder die Kante kreuzt den Schnitt.

Nun summieren wir  $x_i + x_j - 2x_i x_j$  über alle Kanten  $(i, j)$ , der entstehende Ausdruck ist über alle Belegungen von  $\{x_1, \dots, x_n\}$  zu maximieren. Die QUBO-Version von MAX-CUT lautet also  $\text{Max } \sum_{(i,j) \in E} x_i + x_j - 2x_i x_j$ .

**Aufgabe 11.2:** Ermitteln Sie die QUBO-Form von MAX-CUT für den Graphen aus Abbildung 11.3.

Manche Tools zur Lösung von QUBO-Problemen erwarten eine Matrix  $Q$ , so dass

$$x^T Q x = (x_1, \dots, x_n) \cdot Q \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

gerade (11.2) ergibt. Für  $n = 3$  gemäß (11.1) könnte dies die Form

$$(x_1, x_2, x_3) \cdot \begin{pmatrix} a_1 & b_{1,2} & b_{1,3} \\ 0 & a_2 & b_{2,3} \\ 0 & 0 & a_3 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

annehmen.

**Aufgabe 11.3:** Ermitteln Sie die Matrix  $Q$  für die Lösung von Aufgabe 11.2.

Tatsächlich ist die Umwandlung von MAX-CUT in ein QUBO-Problem sehr natürlich. Für viele andere Probleme ist dafür ein höherer Aufwand nötig. Ebenfalls strukturell eng verwandt mit QUBO-Problemen ist das *Ising-Modell*: Für  $n$  Variablen  $s_i \in \{-1, +1\}$  ist

$$h(s) = \sum_{i=1}^n h_i s_i + \sum_{i < j \leq n} J_{i,j} s_i s_j \quad (11.3)$$

<sup>3</sup>In einer allgemeineren Version sind die Kanten mit Gewichten versehen.

zu minimieren. Damit modellieren wir  $n$  Positionen, an denen ein Spin den Wert *up* ( $s_i = +1$ ) oder *down* ( $s_i = -1$ ) annehmen kann. Die Kopplungskonstanten  $J_{i,j}$  beschreiben die Wechselwirkung zwischen den Positionen  $i$  und  $j$  und  $h_i$  die Stärke eines externen Magnetfelds.  $h(s)$  ergibt die Energie des modellierten Systems und ist zu minimieren.

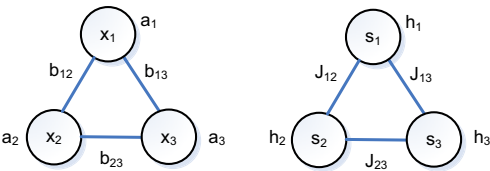


Abbildung 11.4: QUBO-Problem und Ising-Modell mit drei Variablen als Graph

Die strukturelle Ähnlichkeit zwischen (11.2) und (11.3) lässt sich durch Graphen visualisieren. Dazu verwenden wir pro Variable einen Knoten, siehe Abbildung 11.4 für  $n = 3$ . Der linke Graph stammt aus Formel (11.1) nach [153], rechts haben wir das Ising-Analogon. Auf diese Weise lässt sich das auch das allgemeine Ising-Modell über einem Graphen  $G = (V,E)$  definieren:

$$\sum_{i \in V} h_i s_i + \sum_{(i,j) \in E} J_{i,j} s_i s_j.$$

Dazu gehen wir von (11.3) aus und setzen  $V = \{1, \dots, n\}$  und  $(i, j) \notin E$  falls  $J_{i,j} = 0$ . Also sind nur *gekoppelte Positionen* durch eine Kante verbunden. Damit lässt sich folgende Komplexitätsaussage leicht formulieren: Für planare Graphen, also für solche, die in einer Ebene angeordnet werden können, ohne dass Kanten sich schneiden, kann das Ising-Modell in Polynomialzeit minimiert werden.<sup>4</sup> Für allgemeine Graphen hingegen ist dieses Problem NP-schwer.

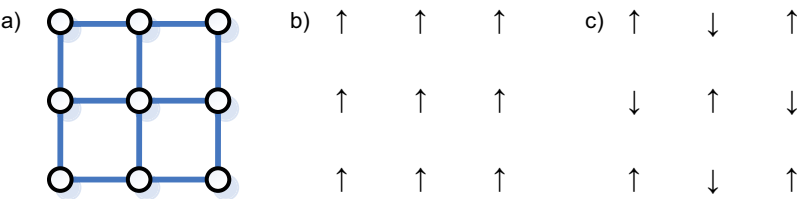


Abbildung 11.5: Ein Gitter mit zwölf Kanten und zwei mögliche Minima

Abbildung 11.5 a) zeigt ein Gitter: In dem Graphen sind nur benachbarte Knoten durch eine Kante verbunden. Setzen wir für jede dieser zwölf Kanten

<sup>4</sup>Analog ist für planare Graphen auch MAX-CUT effizient lösbar.

$(i, j)$  die Kopplungskonstante  $J_{i,j} = -1$ , erhalten wir  $-\sum_{(i,j) \in E} s_i s_j$ . Für  $s_i s_j$  gilt: Genau für  $s_i = s_j$  ist das Produkt  $s_i s_j = 1$  und gerade gleich  $-1$  für  $s_i \neq s_j$ . Also nimmt die Summe den minimalen Wert  $-12$  an, wenn alle  $s_i$  den gleichen Wert haben, jede Abweichung vergrößert diesen Wert. In diesem Modell wird also eine sogenannte *ferromagnetische*, also parallele Ausrichtung der Spins bevorzugt, wie in b) angedeutet. Jedes Paar von benachbarten, also gekoppelten Positionen mit verschiedenen Spins, erhöht die Summe.

Kehren wir das Vorzeichen um, indem wir für alle Kanten  $J_{i,j}$  auf 1 setzen, so nimmt  $\sum_{(i,j) \in E} s_i s_j$  den minimalen Wert  $-12$  an, wenn wir wie in c) eine alternierende Ausrichtungen erhalten: Für jede Kante  $(i, j)$  gilt hier  $s_i \neq s_j$  und liefert den Beitrag  $-1$ . Diesen Fall bezeichnet man als *antiferromagnetisch*.

Das Ising-Modell wurde von Wilhelm Lenz vorgeschlagen und 1925 von seinem Schüler Ernst Ising untersucht. Ursprüngliches Ziel war die Modellierung von Ferromagnetismus.

Die beiden Modelle lassen sich leicht ineinander überführen. Mittels  $x_i \mapsto \frac{s_i+1}{2}$  können wir ein QUBO-Problem als Ising-Modell formulieren und  $s_i \mapsto 2x_i - 1$  erledigt die Gegenrichtung. Im Ising-Modell ist stets das Minimum gesucht (der Grundzustand). Haben wir eine zu maximierende QUBO-Formel übertragen, müssen wir das Resultat gegebenenfalls negieren.

Zusammenhang  
mit QUBO

**Aufgabe 11.4:** Übertragen Sie  $x_i + x_j - 2x_i x_j$  aus der QUBO-Form von MAX-CUT in das Ising-Modell. Bestimmen Sie die Werte für  $s_i = s_j$  und  $s_i \neq s_j$ .

Mit ähnlichen Ideen lässt sich auch das folgende Optimierungsproblem angehen:

**Aufgabe 11.5:** Für das NP-vollständige Mengenzerlegungsproblem PARTITION ist eine Menge von natürlichen Zahlen  $M = \{n_1, \dots, n_N\}$  in zwei Mengen  $A$  und  $B$  zu zerlegen, so dass  $\sum_{n \in A} n = \sum_{n \in B} n$ . Formulieren Sie diese Fragestellung im Ising-Modell.

Die Autoren von [10] haben das Zahlenaufteilungsproblem PARTITION auf dem D-WAVE 2000Q implementiert und getestet. Wenn auch das Ergebnis von Aufgabe 11.5 für uns leicht nachzuvollziehen ist, ist das Ergebnis für einen Quantencomputer eine Herausforderung. Wir haben einen vollständigen Graphen erhalten, alle Positionen des Ising-Modells sind gekoppelt. In real existierenden Quantencomputern sind die Qubits, welche die Spins simulieren, nicht alle paarweise gekoppelt. Die Topologie der Qubit-Kopplungen ist auf Robustheit gegen Dekohärenz ausgelegt. Also muss die Problemstellung noch in die Qubit-Topologie eingebettet werden. Bei den zur Drucklegung realisierten D-WAVE-Rechnern sind die Qubit-Topologien Chimera (2000Q) und Pegasus umgesetzt. Für die nächste Rechnergeneration ist eine Topologie namens Zephyr angekündigt. Zu Details dieses *minor embedding* genannten Vorgehens, siehe etwa [109] oder das schon genannte [10].

Der eigentliche Annealing-Prozess folgt nun dem Grundprinzip aus Abschnitt 11.1 (nach [110]): Aus einer QUBO-Formulierung oder der äquivalenten Ising-Modell-Beschreibung ergibt sich der Hamiltonoperator  $\mathcal{H}_{\text{Lösung}}$ . Die

beteiligten Qubits werden initial in den Grundzustand eines vorab gewählten Hamiltonoperators  $\mathcal{H}_{Start}$  versetzt. Dieser kann zum Beispiel so gewählt werden, dass jedes Qubit den Wert  $1/\sqrt{2}(|0\rangle + |1\rangle)$  erhält. Allmählich wird der Einfluss der Kopplungen  $J_{i,j}$  und der Werte  $h_i$  erhöht, bis das Qubit-System durch  $\mathcal{H}_{Lösung}$  beschrieben wird.

Idealerweise ist das System die ganze Zeit im Grundzustand verblieben und wir erhalten das Minimum. In der Praxis ist dies allerdings nur schwer zu verwirklichen, da die realen Qubits Störungen ausgesetzt sind und sich der Abstand zwischen den Energiezuständen bei steigender Komplexität der Aufgabenstellung verringert. Und selbst unter theoretisch idealen Bedingungen wäre dies vermutlich nicht immer in Polynomialzeit möglich; siehe dazu auch den Absatz *Laufzeit* am Ende von Abschnitt 11.1. Das ist nicht unbedingt ein Problem: Quantum Annealing ist ein heuristisches Verfahren, das weiter darauf zu untersuchen ist, in welchen Fällen es anderen Heuristiken überlegen sein kann.

#### Literatur

Wir haben mit MAX-CUT und PARTITION zwei konkrete, einfache Problemstellungen betrachtet. In der fast schon klassisch zu nennenden Arbeit [104] werden Ising-Formulierungen zahlreicher Optimierungsprobleme dargestellt, ein umfangreiches Tutorium zur QUBO-Erstellung bietet [68]. In [109] wird unter anderem eine konkrete Instanz von max-SAT (hier ist die Zahl der erfüllbaren Klauseln zu maximieren) in QUBO überführt und die weiteren Schritte zur Implementierung in einem D-WAVE System werden beschrieben. Daneben stellt es eine umfangreiche Einführung ins Quantum Annealing dar. In [113] wird eine QUBO zur Verkehrsflussoptimierung hergeleitet, siehe dazu auch [151]. [153] liefert Tools zur QUBO-Erzeugung und -Implementierung.

#### Spin-Glas

In der Literatur wird Ihnen im Zusammenhang mit dem Ising-Modell auch häufiger der Begriff Spin-Glas begegnen. Aber hat Glas denn etwas mit Magnetismus zu tun? Tatsächlich handelt es sich um eine Analogiebildung. Mit *spin glasses* lassen sich keine Brillen oder Fensterscheiben herstellen, sondern es handelt sich um Legierungen mit ungewöhnlichen magnetischen Eigenschaften, es geht um magnetische Phänomene, die durch Unordnung komplex werden. Letzteres ist die Gemeinsamkeit mit unserem Fensterglas: Glas unterscheidet sich dadurch vom Kristall, dass die Positionen von einer symmetrischen Gitterstruktur abweichen. Auf ähnliche Weise unterscheidet sich ein Spin-Glas vom Ferromagneten dadurch, dass die Spins nicht homogen ausgerichtet sind, sondern es Abweichungen von dieser idealen Ausrichtung gibt. Aber sie sind auch nicht alternierend ausgerichtet, wie in Abbildung 11.5 c). Stellen wir uns vor, dass im oben eingeführten Ising-Modell die Vorzeichen der Kopplungskonstanten zufällig gewählt werden. Es ist komplex, in einer solchen Situation zum Zustand der niedrigsten Energie zu gelangen. [138] bietet eine umfangreiche Einführung in das Thema, die sich nicht allein an Physiker wendet.

# 12 Zur Geschichte der Quantenmechanik

*Ich glaube an Geschichte, weil jede Epoche lange nicht so neu, so interessant, so einmalig, so grundsätzlich von allen andern verschieden ist, wie sie sich das einbildet.*

Kurt Tucholsky

Dieses Buch enthält keine Einführung in die Quantenmechanik und das aus gutem Grund. Aber es wäre unbefriedigend, wenn der Leser sich nach der Lektüre mit Quantenbits und kniffligen Quantenalgorithmien auskennen würde, er aber noch nie vom Planckschen Wirkungsquantum oder der Schrödingerschen Wellengleichung gehört hätte. Darum stellt dieses Kapitel Fragen und Konzepte der Quantenphysik mit Hilfe kurzer Portraits auf populärem Niveau dar.

## 12.1 Max Planck: das Quantum der Wirkung

Max Planck markiert den Wendepunkt zwischen klassischer Physik und Quantenmechanik. Geboren wurde er 1858 in Kiel. 1874 begann er Physik zu studieren, obwohl ihm von berufener Seite abgeraten wurde. Nach Auskunft eines Physikprofessors sei das physikalische Wissen so weit fortgeschritten, dass *grundsätzlich Neues darin kaum mehr zu leisten* sei. Was für eine Fehleinschätzung! 1889 übernahm Planck eine Professur in Berlin und machte im Jahr 1900 eine Entdeckung, die die Physik von Grund auf verändern sollte.

Eine eingeschaltete Herdplatte erhitzt sich allmählich. Sie sendet elektromagnetische Strahlung aus, die wir ab einer bestimmten Temperatur als Licht wahrnehmen: die Platte beginnt zu glühen und sendet rotes Licht aus. Noch stärker erhitzte Körper werden gelb und schließlich weiß. Tatsächlich



sendet jeder glühende Körper stets Licht verschiedener Farbe, verschiedener Wellenlänge, aus. Den Zusammenhang zwischen diesem Spektrum des abgestrahlten Lichts und der Temperatur des heißen Körpers zu beschreiben, war Ende des 19. Jahrhunderts ein wichtiges offenes Problem. Es gab bereits Beschreibungsversuche. Diese konnten die experimentellen Ergebnisse jedoch nur in bestimmten Bereichen erklären und widersprachen einander. Planck fand im erwähnten Jahr 1900 eine Formel, die glänzend zu den Daten passte. Diese Formel enthielt jedoch ein Element, das Planck zunächst als vorläufigen mathematischen Kunstgriff ansah, den es so schnell wie möglich zu entfernen galt. Der Grund dafür war, dass gemäß seiner Formel die Energie einer molekularen Schwingung nicht jeden möglichen Wert annehmen kann. Vielmehr waren nur Vielfache des Produkts aus der Frequenz und einer neuen Konstante  $h$  möglich; die Verwendung dieser Konstante ist der angesprochene Kunstgriff. Die Konstante  $h$  wird heute Plancksches Wirkungsquantum genannt, Planck schätzte sie anhand von Messergebnissen an strahlenden Körpern ab. Die Energie kann danach wie auf einer Treppe von Stufe zu Stufe springen, aber die Bereiche dazwischen sind prinzipiell ausgeschlossen, beinahe, als würden sie nicht existieren. Konnte das sein? Plancks Formel entsprach exakt den experimentellen Ergebnissen, aber diese Folgerung mochte er nicht hinnehmen.

Man kann sich heute schwer vorstellen, welche Wirkung diese Entdeckung hatte. Mittlerweile liegen die Anfänge der Quantenmechanik ein Jahrhundert zurück, und wir haben uns an viele ihrer „Zumutungen“ gewöhnt. Es verwundert heutzutage kaum mehr, dass es unteilbare Energieeinheiten gibt. Gab es nicht schon bei den alten Griechen die Auffassung von unteilbaren Grundbestandteilen der Welt, den Atomen?

Das war einmal anders. Von dem Philosophen Gottfried Wilhelm Leibniz (1646-1716), Universalgelehrter der Barockzeit, Erfinder des Binärsystems und der Differentialrechnung, stammt die Aussage: *Nichts geschieht auf einen Schlag; und es ist einer meiner größten und bewährtesten Grundsätze, daß die Natur niemals Sprünge macht. Das nannte ich das Gesetz der Kontinuität.* Dieser Grundsatz – bekannt als *natura non facit saltus* – bestimmte über Jahrhunderte die Naturwissenschaften. Mit Plancks Formel ließ er sich schwerlich vereinen.

Darum dauerte es einige Zeit, bis Planck seine Erklärung des Spektrums so akzeptieren konnte wie sie war. Eine wissenschaftliche Revolution auszulösen, war nicht das Ziel des konservativen Physikers gewesen. Doch das Wirkungsquantum – anfangs ein rechnerisches Hilfsmittel – gewann physikalische Realität. Albert Einstein verwendete es 1905 in seiner Erklärung des photoelektrischen Effekts, und Niels Bohr erklärte 1913 die Stabilität des Atoms mit Hilfe von quantisierten Zuständen.

1918 erhielt Planck den Nobelpreis, und der neuen Physik stand eine stürmische Entwicklung bevor, an der eine Reihe deutscher Physiker beteiligt waren. Nach der politischen Katastrophe, als die bereits viele Zeitgenossen den Beginn der NS-Diktatur ansahen, flohen viele Wissenschaftler aus Deutschland – Albert Einstein ist der berühmteste von ihnen. Planck blieb bis 1937 Vorsitzender der Kaiser-Wilhelm-Gesellschaft. 1945 wurde sein Sohn

Erwin hingerichtet, der zum konservativen Widerstand um Claus Schenk Graf von Stauffenberg gehörte. Zwei Jahre später stirbt Max Planck in Göttingen. 1948 wird die Kaiser-Wilhelm-Gesellschaft umbenannt und trägt seitdem seinen Namen.

## 12.2 Albert Einstein: Spukhafte Fernwirkung

Das Staunen gilt seit der Antike als Anreger neuer Erkenntnisse. *Dies „sich wundern“ scheint dann aufzutreten, wenn ein Erlebnis mit einer in uns hinreichend fixierten Begriffswelt in Konflikt kommt.* Ein Wunder solcher Art erlebte Albert Einstein (1879–1955) als Kind von vier oder fünf Jahren, als ihm sein Vater einen Kompass zeigte. *Dass diese Nadel in so bestimmter Weise sich benahm, passte so gar nicht in die Art des Geschehens hinein, die in der unbewussten Begriffswelt Platz finden konnte: an Berührung geknüpftes Wirken.* Dies machte einen bleibenden Eindruck und führte zu der Überzeugung: *Da musste etwas hinter den Dingen sein, das tief verborgen war.*

Im Jahr 1905 veröffentlicht Einstein innerhalb weniger Monate mehrere bahnbrechende Arbeiten, in dem so bezeichneten Wunderjahr entstanden unter anderem die spezielle Relativitätstheorie und die Erklärung des sogenannten photoelektrischen Effekts, ein entscheidender Schritt für die Entstehung der Quantenmechanik. Zu dieser Zeit ist der 1879 in Ulm geborene als technischer Vorprüfer am Patentamt in Bern angestellt.

1886 beobachtete Heinrich Hertz erstmals, dass geladene Metalle Elektronen abgeben, werden sie mit Licht bestrahlt. Diesen Effekt nutzen Fotozellen aus. Die im 19. Jahrhundert allgemein akzeptierte Wellentheorie des Lichts konnte dieses Phänomen im Prinzip erklären, aber nicht quantitativ. Der klassischen Physik zufolge müsste die Energie der freiwerdenden Elektronen allein von der Intensität des Lichtes abhängen, nicht hingegen von der Wellenlänge. Dem widersprachen die experimentell beobachteten Phänomene. So kann ultraviolettes Licht auch bei niedrigster Intensität Elektronen aus Metallen freisetzen, während das mit langwelligem Licht selbst bei hoher Intensität nicht gelingt.

Einstein fand eine Erklärung, die dem Licht Teilchencharakter zuschrieb, wie es vor ihm bereits Newton getan hatte. Aus der geladenen Platte werden die Elektronen demnach durch diskrete Energiepakete geschlagen, den Lichtquanten oder Photonen. Die Energie eines solchen Photons hängt von der Wellenlänge ab, diese wiederum bestimmt die Frequenz des entweichenden Photons. Diese Energie konnte Einstein durch eine elegante Formel beschreiben: sie ist gleich dem Produkt aus der Frequenz und dem Planckschen Wirkungsquantum.

Für diese Erklärung erhielt Einstein im Jahr 1921 den Nobelpreis für Physik; nicht für die Relativitätstheorie, sondern für eine die Quantenmechanik fundierende Arbeit. Dabei muss man Einsteins Verhältnis zu dieser Theorie als gespalten bezeichnen: er trug Wesentliches zu ihr bei und gehört zu den Mitbegründern, lehnte aber gleichzeitig viele Konzepte und Folgerungen

entschieden ab. Zeit seines Lebens konnte er sich nicht damit abfinden, dass *Unbestimmtheit* und *Verschränkung* Elemente einer vollständigen Beschreibung der Natur sein sollen.

Einsteins berühmte, 1935 mit Boris Podolsky und Nathan Rosen veröffentlichte Arbeit wurde bereits im Abschnitt 2.11 diskutiert. Das Einstein-Podolsky-Rosen-Paradoxon wird dort am Beispiel zweier verschränkter Quantenbits erläutert. Diese sind solcherart gekoppelt, dass durch eine Messung an einem Bit unmittelbar das Ergebnis der Messung an einem anderen Bit festgelegt und bekannt ist. Das Unbehagen der drei Autoren mit dieser *spukhaften Fernwirkung* äußerte sich in dem Vorschlag, das klassisch geordnete Weltbild auf das Fundament verborgener Variablen zu stellen.

Ernst genommen hat diesen Vorschlag John Stewart Bell, der 1964 zeigen konnte, dass die Aussagen einer auf verborgenen Variablen beruhenden lokal-realistischen Theorie und solche der Quantenmechanik einander widersprechen müssen. Die Bedeutung dieses Ergebnisses für die heutige Quantenmechanik kann man nicht hoch genug einschätzen. Der amerikanische Physiker Henry Stapp spricht gar von *der tiefgründigsten Entdeckung seit Kopernikus*.

Als Kind wunderte sich Einstein über einen Kompass, da er *an Berührung geknüpft*es Wirken gewohnt war. *Da musste etwas hinter den Dingen sein*. Magnetische Fernwirkung lässt sich mit einem Medium erklären, das die Kompassnadel mit der Wirkursache verbindet. Aber bei der spukhaften Fernwirkung verschränkter Teilchen gibt es nach heutiger Auffassung nichts dahinter. Es ist ein experimentell nachgewiesenes und von in diesem Buch beschriebenen Verfahren genutztes Phänomen, für das man heute keine anschauliche Erklärung kennt. Das heißt jedoch nicht, dass Einsteins Suche nach einem geordneten Weltbild für gescheitert erklärt werden müsste. Die Physik beschreibt eine verschränkte Welt, in der auch räumlich Getrenntes zusammenhängt.

## 12.3 Niels Bohr: Kopenhagen

*Gott würfelt nicht* ist einer der meist zitierten Aussprüche Albert Einsteins. Diese Kritik an der Rolle von Unbestimmtheit und Zufall in der Quantenmechanik findet sich in einem Brief an den dänischen Physiker Niels Bohr (1885–1962), formuliert als *bange Frage*. Weniger bekannt ist Bohrs Antwort, *dass niemand – und nicht einmal der liebe Gott selber – wissen kann, was ein Wort wie würfeln in diesem Zusammenhang heißen soll*. Hier waren offenbar zwei Skeptiker aufeinander getroffen. Der eine zweifelte an einer Theorie, deren Unanschaulichkeit für ihn ans Widernatürliche grenzte. Der andere an Aussagen, die in ihrem Gegenstandsbereich nicht enthaltsam genug sind, als dass sie sinnvoll sein könnten. Doch bevor wir Bohrs Rolle als Deuter der Quantenmechanik ansprechen, wollen wir sehen, wie er nach Planck und Einstein den dritten großen Schritt auf dem Weg zur Quantenphysik tat.

Der neuseeländische Physiker Ernest Rutherford wurde durch Experimente zu einem Modell des Atoms angeregt, nach dem diese einen positiv geladenen

Kern besitzen, um den negativ geladene Elektronen kreisen. Daraus ergab sich ein Problem: nach den Gesetzen der klassischen Physik kann ein so gebautes Atom unmöglich stabil sein. Die kreisenden Elektronen müssten eigentlich Energie verlieren und von dem Kern angezogen werden, bis sie in diesen hinein stürzen. Aber die Atome waren stabil.

Diesem Problem sah sich 1912 der 27-jährige Niels Bohr gegenüber, der Rutherfords wegen nach Manchester gekommen war. Er erneuerte Rutherfords Modell und verwendete dazu Plancks Theorie der quantisierten Wirkung, die Albert Einstein 1905 zur Erklärung des sogenannten photoelektrischen Effekts genutzt hatte. Bohrs Modell muss als kühn bezeichnet werden. Konzepte der klassischen Physik, wie um ein Zentrum kreisende Satelliten, wurden mit Beschränkungen der Energiewerte im Sinne der entstehenden Quantenphysik zusammengeführt. So sind den Elektronen nur bestimmte Bahnen gestattet, die jeweils verschiedenen quantisierten Energiewerten entsprechen. Er formte ein Bild vom Atom, das einen hybriden Eindruck machen könnte, die zum Beispiel am Wasserstoffatom herrschenden Verhältnisse jedoch qualitativ und quantitativ hervorragend beschrieb. Ergänzt und korrigiert wurde dieses Modell mehr als ein Jahrzehnt später von Heisenberg und Schrödinger.

Zu dieser Zeit wurde auch Bohrs Rolle als Philosoph der Quantenmechanik immer deutlicher. Dem Leser dieses Buches ist die Rolle von Messungen an quantenmechanischen Systemen bekannt. Die Wahl der Basis bestimmt das Resultat eines solchen Vorgangs mit. Messergebnisse können sogar rein zufällig sein, wenn die gemessene Eigenschaft in dieser Basis nicht bestimmt ist. Zur Verdeutlichung kann wieder einmal Schrödingers Katze dienen, deren Gesundheitszustand gegenüber den Begriffen „tot“ und „lebendig“ unbestimmt ist.

Dieser Umstand allein stellte für die wissenschaftliche Öffentlichkeit eine nicht geringe Zumutung dar. Aber was folgt daraus über das gemessene Objekt? Erinnern wir uns an das Mach-Zehnder-Interferometer aus Abschnitt 10.2.1. Dort haben wir das Messergebnis durch Interferenz von Amplituden erklärt, und zwar der Amplituden für zwei Wege, die das Photon einschlagen konnte. Aber wie soll man sich ein Photon vorstellen, das diese Wege geht? Bohrs Antwort wäre: Gar nicht! Man soll sich auf die beobachtbaren Phänomene beschränken. Über diese und nur über diese können wir Aussagen machen. *Es gibt keine Quantenwelt* wird Niels Bohr zitiert. Um diese Aussage zu verstehen, können wir ein anderes Diktum Bohrs heranziehen: *Kein Phänomen ist ein Phänomen, außer es ist beobachtetes Phänomen*. Beobachtbare Phänomene sind klassisch. Das Photon wird vom Detektor  $D_0$  oder vom Detektor  $D_1$  registriert. Darüber, was das Photon vor der Beobachtung macht, können wir nichts aussagen. Gemäß Bohr wäre es sogar sinnlos, etwas darüber aussagen zu wollen. Ein Zustand vor einer Beobachtung ist aus logischen Gründen unbeobachtbar.

Diese Beschränkung auf die Phänomene ist ein kleiner Teil eines Deutungsansatzes der Quantenmechanik, der nach Bohrs wichtigster Wirkungsstätte als *Kopenhagener Interpretation* bezeichnet wird. Fragen, wie die eben geschilderten gehören in gewisser Weise nicht mehr zu der Theorie, die den Namen Quantenmechanik trägt. Eine Interpretation übersetzt Begriffe und

Aussagen eines Systems in ein anderes. Im Falle der Quantenphysik gibt es letztlich drei Systeme. Die formale Theorie mit Vektorräumen und unitären Transformationen, die Experimentalphysik, der wir die beobachtbaren Phänomene zuordnen, und schließlich unsere Alltagssprache, in der physikalische Fragen stets klassisch verhandelt werden.<sup>1</sup> Interpretationen der Quantenmechanik erzeugen in der Regel Sätze der Alltagssprache. Deren Ziel ist jedoch nicht, pädagogische Veranschaulichungen zu finden oder physikalische Erkenntnisse für weltanschauliche Fragen nutzbar zu machen. Interpretationen der Quantenmechanik sagen etwas über die physikalische Realität selbst aus.

Als einen Aspekt von Niels Bohrs Sicht der Quantenphysik haben wir die Beschränkung kennen gelernt, nur über beobachtbare Phänomene zu sprechen. Ein anderer wesentlicher Aspekt der Kopenhagener Interpretation ist die Beschreibung der Messung als Zufallsexperiment. Eine Superposition geht in einen Basiszustand über. Die Wahrscheinlichkeit ergibt sich dabei aus den Amplituden. Entscheidend ist der Aspekt, dass hierbei eine sich ansonsten unitär entwickelnde Superposition von Basiszuständen mit einem Schlag zu einem der Zustände der Basis reduziert wird. Dieser Vorgang ist der Ausgangspunkt verschiedener Deutungsansätze, da hier verschiedene Probleme auftreten, die mit unseren Mitteln leider nicht hinreichend beschrieben werden können.

Die Viele-Welten-Interpretation wurde 1957 von dem amerikanischen Physiker Hugh Everett (1930–1982) vorgeschlagen. Nach diesem Deutungsansatz wird die Superposition bei der Messung nicht reduziert, sondern die verschiedenen Ergebnisse werden in verschiedenen Welten realisiert, es gibt eine gewaltige Zahl von Paralleluniversen. Öffnen wir die Kiste mit Schrödingers Katze, werden wir in einer Welt eine lebendige Katze finden, in einer anderen eine tote. Während der Messung wird zwischen zwei möglichen Welten verzweigt. Alle Alternativen sind jedoch real. Dem Beobachter kommt keine herausgehobene Rolle zu, und mit Blick auf alle Welten wird die Superposition nicht wirklich reduziert. Ein Problem dieses Deutungsansatzes ist, dass wir zu den Paralleluniversen keinen Zugang haben und sich die Grundannahme der Viele-Welten-Theorie nicht experimentell beweisen lässt. Das könnte sich jedoch ändern; denken wir nur an Bells Folgerungen aus den verborgenen Variablen. Anhänger der Kopenhagener Deutung kritisieren an der Viele-Welten-Theorie den ihrer Meinung nach verschwenderischen Umgang mit unbeobachtbaren Elementen. Wir wollen hier Neutralität bewahren und darauf hinweisen, dass Everetts Ansatz eine Erklärung zu einer Frage anbietet, die die Kopenhagener Deutung in eisiges Schweigen hüllt.

<sup>1</sup>Hier und im gesamten Abschnitt bewegen wir uns auf schwankendem Boden. So ist die genannte Trennung in gewisser Weise willkürlich: mathematisch formulierte Theorien sind von der physikalischen Welt der beobachtbaren Phänomene nicht strikt getrennt. Sie sind mit physikalischen Systemen – zum Beispiel Gehirnen – auf das engste verbunden. Auch die Trennung von klassischer Physik und Quantenphysik ist problematisch, da völlig unklar ist, wo die Grenze verlaufen sollte. Dieser knapp formulierte Abschnitt kommt manchmal nicht um Vereinfachungen herum, die sich bei genauer Betrachtung als naiv erweisen. Ziel ist es auch nicht, den vielen tief sinnigen Abhandlungen über die Deutungen der Quantenmechanik eine weitere hinzuzufügen, sondern bei dem Leser einen ersten Eindruck entstehen zu lassen.

Ein bekannter Unterstützer dieses Deutungsansatzes ist David Deutsch, der den Begriff Multiversum verwendet.

Eine weitere interessante Deutung stammt von dem amerikanischen Physiker David Bohm (1917–1992). Dessen Beschreibung der Quantenmechanik ermöglicht es, den Zufall gänzlich zu eliminieren. Das gelingt durch eine zusätzliche Größe, die man Quantenpotential nennen könnte. Damit wird es möglich, zu jeder Zeit den Zustand eines Teilchens als bestimmt anzusehen, auch vor der Messung gibt es einen definierten Zustand. Nach Bohms Deutung ist die Welt letztlich deterministisch, auch wenn seine Deutung die gleichen Voraussagen macht. Um dem augenscheinlichen Zufall in den Experimenten eine deterministische Tiefendimension zu verleihen, muss die Quantenmechanik um experimentell nicht zugängliche Elemente erweitert werden. Insoweit gibt es eine Ähnlichkeit zwischen Bohms Ansatz und Einsteins verborgenen Variablen.

Jede dieser Deutungen hat ihre Vorzüge und Nachteile. Es ist nicht ausgeschlossen, dass eines Tages Experimente zwischen ihnen entscheiden können, denken wir nur an Bells Ungleichung. Bis dahin ist die Wahl des Interpretationsansatzes auch eine Frage des Geschmacks. *Was für eine Philosophie man wähle, hängt sonach davon ab, was für ein Mensch man ist*, so der Philosoph Johann Gottlieb Fichte. Zuletzt kehren wir noch einmal zu Niels Bohr zurück und der Kopenhagener Interpretation, deren Ausprägung von Anhänger zu Anhänger deutlich variieren kann. Sie ist der Deutungsansatz, der unter den lebenden Quantenmechanikern die meisten Befürworter besitzt. In gewisser Weise ist ihr Aufruf zu größtmöglicher Zurückhaltung die philosophischste Alternative, da sie die Grenzen des möglichen Wissens sehr ernst nimmt. Andererseits spielt der Beobachter eine entscheidende Rolle, was Parallelen zu verschiedenen philosophischen Traditionen andeutet. So könnte man an den irischen Bischof George Berkeley (1685–1753) und seinen berühmten Ausspruch *Sein heißt wahrgenommen werden* denken. Es gibt auch einen Bezug zu Kants Ding an sich, den Carl Friedrich von Weizsäcker in seiner Version der Kopenhagener Deutung ausgearbeitet hat. Die philosophische Position, auf die sich die Anhänger der Kopenhagener Interpretation für gewöhnlich beziehen, ist die Maxime von William von Ockham (etwa 1285–1349), die lax ausgedrückt lautet: Diejenige Theorie ist am besten, die möglichst viel erklärt und dabei keine überflüssigen Begriffe verwendet.

## 12.4 Werner Heisenberg: Ein großes Quantenei

Helgoland, Frühjahr 1925. Der Pollenallergiker Werner Heisenberg (1901–1976) war aus Göttingen geflohen, wo er im Vorjahr bei Max Born habilitiert wurde. Sein Betreuer hatte Heisenberg freigestellt und dieser nutzte seinen Aufenthalt auf der Nordseeinsel: er wanderte, las Gedichte und beschäftigte sich mit den offenen Problemen der neuen Physik. Die Leistungsfähigkeit des Bohrschen Atommodells war längst an ihre Grenzen gestoßen. Es fehlte eine solide Basis, die das Verhalten der Elektronen erklären konnte, ein Analogon

zu den Newtonschen Bewegungsgesetzen. Die Quantenphysik war noch keine *Quantenmechanik*.

Diesem Problem widmete sich Heisenberg auf Helgoland mit dem Voratz, sich auf beobachtbare Größen zu beschränken – das hieß, anschauliche Konzepte wie zum Beispiel Elektronenbahnen auszuschließen – und dem Energieerhaltungssatz besondere Aufmerksamkeit zu widmen. Mit der Idee, bestimmte Größen statt durch Zahlen durch Matrizen zu beschreiben, gelang es ihm, alle Elemente in einer mathematischen Beschreibung widerspruchsfrei zusammenzufügen. In seinen Erinnerungen beschreibt Heisenberg diese Entdeckung und das darauf folgende zutiefst erschreckende *Gefühl, durch die Oberfläche der atomaren Erscheinungen hindurch auf einen tief darunter liegenden Grund von merkwürdiger Schönheit zu schauen*. Prosaischer war die Reaktion eines anderen bedeutenden Physikers. *Heisenberg hat ein großes Quantenei gelegt*, schrieb Einstein im September 1925 an seinen Freund Ehrenfest, *in Göttingen glauben sie daran (ich nicht)*.

Zurück in Göttingen, arbeitete Heisenberg seine Entdeckung zusammen mit Max Born und Pascual Jordan zu einem soliden mathematischen Fundament für die Quantenphysik aus. Berühmt geworden ist eine gemeinsame Arbeit von 1926, die sogenannte Drei-Männer-Arbeit, die systematisch darstellt, was noch heute Quantenmechanik genannt wird. Damit war der fünfundzwanzigjährige Heisenberg endgültig zu einer Art Wunderkind der Physik geworden. Ein Jahr später wurde er Professor in Leipzig. Bereits im Jahr 1922 hatte er Niels Bohr kennengelernt, der in Göttingen eine Vortragsreihe über die neue Physik hielt, die unter der Bezeichnung Bohr-Festspiele bekannt geworden ist. Der junge Physiker kritisierte den Nobelpreisträger auf fast ungebührliche Weise. Das Ergebnis war eine intensive Freundschaft und Zusammenarbeit, die allerdings durch die politischen Umstände auf die Probe gestellt wurde. 1932 wurde Heisenberg für seine Arbeiten zur Quantenmechanik mit dem Nobelpreis ausgezeichnet, die Geschehnisse des Folgejahres veränderten jedoch die politischen Verhältnisse radikal. Zu Beginn des Zweiten Weltkriegs wurde er an dem sogenannten „Uranprojekt“, dem deutschen Atombombenprojekt, beteiligt. Ungeschickte Äußerungen Heisenbergs über die Bewertung einer deutschen Vorherrschaft in Europa sollen Bohr sehr irritiert haben. Legendär ist ein Treffen 1941 in Kopenhagen, über dessen Verlauf wenig Konkretes bekannt ist; vieles deutet jedoch darauf hin, dass Heisenberg Bohr dazu überreden wollte, an einer deutschen Atombombe mitzuwirken. Michael Frayns Theaterstück *Kopenhagen* aus dem Jahr 1998 spielt verschiedene Versionen des Gesprächsverlaufs durch.

Wenn Nicht-Physiker den Namen Heisenberg hören, denken sie in der Regel zunächst an die Heisenbergsche Unbestimmtheitsrelation. Von Bohr stammt das Komplementaritätsprinzip, wonach es prinzipiell nicht zu versöhnende Eigenschaften ein und desselben physikalischen Objektes gibt (wie Wellen- und Teilcheneigenschaften), die sich auf der anderen Seite wechselseitig bedingen. Heisenberg drückte diesen Sachverhalt mathematisch in der besagten Unbestimmtheitsrelation aus: Je genauer der Ort eines Teilchens bestimmt ist, desto unschärfer ist sein Impuls (Masse mal Geschwindigkeit) bestimmt – und umgekehrt. Es können also nicht beide Attribute gleichzeitig

genau bestimmt sein. Um dieses Prinzip in der Begriffswelt dieses Buches zu erläutern, greifen wir auf polarisierte Photonen zurück, siehe Abschnitt 10.2.3. Treten Photonen durch einen Polarisationsfilter, sind sie anschließend gemäß einer festen Richtung ausgerichtet. Ist ein Filter gemäß 0 Grad ausgerichtet und wissen wir, dass alle eintretenden Photonen gemäß einer der beiden Richtungen 0 Grad oder 90 Grad ausgerichtet sind, können wir ihn zur Messung der Polarisation der Photonen nutzen: ein Detektor hinter dem Filter registriert in diesem Fall genau dann ein Photon, wenn es gemäß 0 Grad ausgerichtet war. Wird ein durch den Filter geschicktes Photon nicht registriert, war die Polarisationsrichtung 90 Grad.

Nun drehen wir den Filter um 45 Grad. Damit lässt sich eine Messung in der Basis 45 Grad / 135 Grad vornehmen. Was passiert aber, wenn wir ein gemäß 0 Grad oder 90 Grad ausgerichtetes Photon in den gedrehten Versuchsaufbau schicken? Ein solches Photon ist völlig unbestimmt gegenüber der Eigenschaft, gemäß 45 Grad oder 135 Grad ausgerichtet zu sein. In der Hälfte der Fälle wird ein Photon registriert, in der Hälfte der Fälle nicht. Diese Eigenschaft ist es, die sich das BB84-Protokoll zu nutze macht. Ähnlich kann man sich das Verhältnis der Eigenschaften vorstellen, für die Heisenbergs Unbestimmtheitsrelation meistens angegeben wird. Mit Hilfe einer Basis kann man den Ort eines Teilchens bestimmen, mit einer anderen Basis den Impuls, beide Basen gehen durch eine Drehung auseinander hervor. Häufig wird die Unbestimmtheitsrelation auch Unschärferelation genannt. Diese Bezeichnung ist jedoch weniger glücklich, da sie nahelegt, das Verhalten von gemessenen Quantensystemen ließe sich durch die Ungenauigkeit der Messinstrumente erklären.

## 12.5 Erwin Schrödinger: Katzen und Wellen

Unser Streifzug durch die Geschichte der Quantenmechanik muss viele bedeutende Protagonisten unberücksichtigt lassen. Nicht unerwähnt bleiben darf Erwin Schrödinger (1887–1961). Einerseits wird das nach ihm benannte Katzensystem in diesem Buch an mehreren Stellen erwähnt. Zum anderen ist die nach ihm benannte Wellengleichung (siehe Seite 272) für die Quantenmechanik ähnlich zentral wie Newtons Bewegungsgleichungen für die klassische Physik. Richard Feynman sagte über diese: *Woher haben wir diese Gleichung? Nirgendwoher. Es ist unmöglich, sie aus irgend etwas Bekanntem herzuleiten. Sie ist Schrödingers Kopf entsprungen.*

Schrödinger wurde 1887 in Wien geboren. In seiner Heimatstadt habilitierte er in Physik und lehrte an verschiedenen Stationen bis er 1927 in Berlin Max Plancks Nachfolger wurde. 1926 beschäftigte er sich in Zürich mit der Drei-Männer-Arbeit von Born, Heisenberg und Jordan. Die Göttinger Matrizenmechanik war allerdings nicht nach Schrödingers Geschmack, zu unanschaulich und unklassisch erschien sie ihm. *Wenn es doch bei dieser verdammten Quantenspringerei bleiben soll, so bedaure ich, mich überhaupt jemals mit Quantentheorie abgegeben zu haben*, wird Schrödinger zitiert.



Aber er hatte eine Idee, wie sich die klassische Physik rehabilitieren ließe. Louis Victor de Broglie (1892-1987, Nobelpreis 1929), Nachfahr einer alten französischen Adelsfamilie, äußerte 1924 die Vermutung, dass jegliche Materie wellenhaftes Verhalten zeige, so wie das Licht wellenartig und teilchenartig zugleich ist. Schrödinger nahm diesen Gedanken auf und versuchte, das Verhalten der Elektronen in einem Atom durch eine Wellengleichung zu beschreiben. Im Jahr 1926 stieß er auf eine Gleichung, bei der alles stetig und klassisch zuzugehen schien und somit ganz nach Schrödingers Sinn war. Diese Differentialgleichung beschreibt die zeitliche Entwicklung des Zustands eines unbeobachteten Quantensystems; die in diesem Buch zentrale Entwicklung eines solchen Systems mittels unitärer Transformationen lässt sich aus dieser ableiten. Entscheidend ist jedoch der Begriff *unbeobachtet*. Kommen messbare Größen ins Spiel, endet die Stetigkeit der Entwicklung. Schlimmer noch: Schrödingers Ansatz erwies sich bald als mathematisch äquivalent mit Heisenbergs Matrizenmechanik. 1933 wurde Schrödinger mit dem Nobelpreis für Physik geehrt.

## 12.6 Schlaglichter der Geschichte des Quantencomputers

An dieser Stelle macht unsere Darstellung einen inhaltlichen und zeitlichen Sprung in das Jahr 1980, aus dem der erste Entwurf einer Quanten-Turingmaschine stammt. Die Miniaturisierung der Hardwarekomponenten ließ die Frage entstehen, ob sich auch mit Teilchen rechnen lässt, für die die Gesetze der klassischen Physik keine Geltung mehr haben. Der in Chicago tätige Paul Benioff konnte auf Vorarbeiten über umkehrbare Berechnungen aufbauen – eine reversible Turingmaschine wurde 1973 von Charles Bennett entworfen – und zeigte, wie mit einem quantenmechanischen System klassische Berechnungen simuliert werden können. Allerdings ist die dazu von Benioff erdachte Turingmaschine noch nicht allgemein genug, um die in diesem Buch behandelten Verfahren auszuführen. Denn nur zwischen zwei Schritten befindet sich diese Maschine in einem echten Quantenzustand; nach Abschluss eines Schrittes ist ihr Zustand klassisch. Es wurde später gezeigt, dass Benioffs Quanten-Turingmaschine von einem klassischen Rechner simuliert werden kann.

Die Intuition, dass Quantenrechner Fähigkeiten haben könnten, die klassischen Rechnern abgeht, stammt von dem amerikanischen Physiker Richard P. Feynman (1918-1988; Nobelpreis 1965). Im Jahr 1982 veröffentlichte er Überlegungen darüber, wie sich Quantensysteme simulieren lassen, [59]. Er argumentierte, dass dazu jeder klassische Computer exponentiellen Aufwand betreiben müsse. Ein universeller Quantensimulator hingegen könne diese Aufgabe effizient ausführen. Wir erläutern das mit Zuständen von Quantenregistern. Deren Entwicklung kann durch Matrizen beschrieben werden. Die Größe dieser Matrizen ist aber exponentiell in der Anzahl der beteiligten Quantenbits, und klassische Rechner können folglich nicht effizient mit ihnen umgehen. Für die Natur stellen die komplexen Vorgänge der Quantenme-

chanik natürlich kein Problem dar. Also könnte es Aufgaben geben, die für klassische Rechner exponentiellen Aufwand bedeuten, mit einem rechnenden Quantensystem aber effizient lösbar sind. Dazu gehört sicher die Simulation von quantenmechanischen Systemen.

Ein theoretischer Durchbruch datiert von 1985. David Deutsch (geboren 1953) modellierte eine Turingmaschine, die die Gesetze der Quantenmechanik stringent umsetzt, [44]. Während auf dem Band einer klassischen Turingmaschine nur die Einträge 0 oder 1 erlaubt sind, lässt Deutschs Quanten-Turingmaschine Superpositionen zu. Die Rechenschritte folgen den Gesetzen der Quantenmechanik, den Nutzen demonstrierte Deutsch mit dem heute nach ihm benannten Algorithmus und der Erzeugung echter Zufallszahlen, die klassischen Rechnern prinzipiell verwehrt ist.

Ein neues Forschungsgebiet war damit entstanden, das zahlreiche Wissenschaftler mit Pioniergeist anzog. Die Aufmerksamkeit eines größeren Publikums blieb aber zunächst aus. Dafür gibt es mindestens zwei Gründe:

- Es war nicht nur unklar, wie man einen Quantencomputer konkret bauen sollte, es gab auch theoretische Erwägungen, die dessen Existenzmöglichkeiten bestritten. Ein Quantenbit kann unendlich viele verschiedene Zustände annehmen und ist damit sehr störanfällig. Außerdem lässt es sich nicht kopieren, und genau dies tun klassische fehlerkorrigierende Codes auf geschickte Weise.
- Es fehlte eine Anwendung, die die Nützlichkeit des Quantencomputers zeigen könnte, eine *killer application*, die die neue Technologie erstrebenswert machte.

Beide Einwände wurden insbesondere durch Arbeiten des 1959 geborenen Peter W. Shor entkräftet. Sein Verfahren zur effizienten Faktorisierung ganzer Zahlen zeigt, dass Quantencomputer klassischen Rechnern bei sehr natürlichen Fragestellungen und praktisch wichtigen Aufgaben überlegen sein können. Und zudem stammen von ihm die ersten fehlerkorrigierenden Quantencodes, an deren Existenzmöglichkeit es gewisse Zweifel gab.

Wie in Abschnitt 10.3 und auch schon in der Einleitung erwähnt, konnte im Jahr 2001 Shors Algorithmus erstmals auf einem Quantencomputer ausgeführt werden. Die Zahl 15 wurde erfolgreich faktorisiert. Viele weitere Verfahren wurden in der Folge entwickelt und mit sehr unterschiedlichen Hardwareansätzen umgesetzt. Einiges findet sich im Kapitel 10; auch im Rest des Buches finden sich historische Anmerkungen. Dieser Abschnitt soll einen großen Bogen von den ersten Anfängen in die Gegenwart schlagen: Darum wird auf eine kleinteilige Auflistung verzichtet und wir erlauben uns einen zeitlichen und inhaltlichen Sprung.

Die kanadische Firma D-Wave Systems bezeichnet sich selbst als ersten kommerziellen Entwickler von Quantencomputern. Im Jahr 2011 kam der D-Wave One mit 128 Flux-Qubits auf den Markt, siehe Abschnitt 10.6. Der US-Rüstungskonzern Lockheed Martin erwarb ein solches Modell, was für große mediale Aufmerksamkeit sorgte. Dieser wuchs noch, als Google und die NASA das seit 2013 erhältliche Modell D-Wave Two mit 512 Quantenbits

Debatte um  
D-Wave

anschafften. Zur damaligen Zeit konnten die anderen kommerziellen und akademischen Institutionen nur experimentelle Quantencomputer mit einer kleinen zweistelligen Zahl an Quantenbits realisieren. Aus dieser Diskrepanz entstand eine Debatte, die in einer Vielzahl von Publikationen zwischen besagter Firma und akademischen Kritikern kontrovers geführt wurde. Es handelt sich bei diesen Rechnern um reine *Quanten-Annealer*, siehe Abschnitt 11.2. Eine große Schwierigkeit bei der Bewertung ist, dass man für die verwendete Methode die Laufzeit nicht allgemein beschreiben kann. Für manche Problemstellungen oder Eingaben könnte diese sehr gut sein, für andere sehr schlecht. Ende 2021 gab D-WAVE Systems bekannt, zukünftig auch gatterbasierte Quantenrechner bauen zu wollen.

NISQ

Im Jahr 2018 sprach John Preskill in [121] von der *NISQ era*, also dem Zeitraum der *Noisy Intermediate-Scale Quantum*-Technologie. Das beschreibt treffend die Situation zur Drucklegung dieses Buches. Trotz aller Erfolge beim Bau von Quantencomputern ist die Zahl der verfügbaren Qubits und die Verlässlichkeit der darauf ausführbaren Operationen noch zu gering, um zum Beispiel mit Shors Algorithmus das RSA-Verfahren zu brechen. Somit ist der Begriff NISQ mit der Aufforderung verbunden, mit den derzeit existierenden Quantencomputern praxisnahe Aufgaben zu lösen, unter Umständen Teilaufgaben von komplexeren Problemen, und zwar im Idealfall schneller als mit einem klassischen Rechner.

Quantenvorteil

Unter *Quantenvorteil* (oder *quantum advantage*, der ursprüngliche, etwas unglücklich gewählte Begriff lautet *quantum supremacy*) versteht man die Demonstration, dass ein Quantencomputer ein Problem konkret löst, das von einem klassischen Computer nicht in annehmbarer Zeit erledigt werden kann. Hier geht es ausdrücklich nicht unbedingt um ein praktisch relevantes Problem, sondern um den Nachweis von prinzipieller Überlegenheit.

Aufsehen über die Fachwelt hinaus erregte im Jahr 2019 Googles Mitteilung, diesen Quantenvorteil nachgewiesen zu haben, [8]. Der *Sycamore*-Prozessor mit 53 Qubits löste eine zunächst anwendungsferne und auf die Fähigkeiten von NISQ-Rechnern zugeschnittene Aufgabe namens Boson-Sampling. Kritik zog die Einschätzung der Autoren nach sich, dass der weltweit schnellste Computer für die gleiche Aufgabe 10.000 Jahre benötigen würde: Durch Verbesserung des klassischen Algorithmus könne ein Supercomputer die Aufgabe in wenigen Tagen lösen. Unabhängig von dieser Kritik stellt die Veröffentlichung einen wichtigen Meilenstein auf dem Weg zu praktisch einsetzbaren Quantencomputern dar.

### Schluss

Nun ist es an dem Leser, an Ihnen, sich auf weitere Entdeckungsreise zu begeben und zusammen mit dem Autor gespannt zu sein, welche Entwicklungen die nächsten Jahre mit sich bringen werden.

# A Mathematische Grundlagen

*Die Mathematiker sind eine Art Franzosen: redet man zu ihnen, so übersetzen sie es in ihre Sprache, und dann ist es alsobald ganz etwas anderes.*

J. W. v. Goethe

Dieser Anhang sammelt grundlegende mathematische Begriffe und Aussagen. Der Inhalt der ersten drei Abschnitte ist für den Umgang mit Quantenbits wesentlich. Es geht um komplexe Zahlen, Vektorräume und Matrizen. Abschnitt A.4 beschäftigt sich mit Wahrscheinlichkeitstheorie. In diesem Buch werden keine tieferen Einsichten aus diesem Gebiet verwendet. Ein intuitives Verständnis des Begriffs *Wahrscheinlichkeit* ist jedoch hilfreich für den Umgang mit Fehlerwahrscheinlichkeiten. Abschnitt A.5 enthält zahlen-theoretische Grundlagen für das Verständnis von RSA-Kryptographie und Shors Algorithmus.

Die Darstellung ist sehr knapp und soll Bekanntes, aber vielleicht Verschlüsseltetes, wieder zum Vorschein bringen.

## A.1 Komplexe Zahlen

Mit  $\mathbb{C}$  bezeichnen wir die komplexen Zahlen. Diese bilden einen *Körper*, was anschaulich bedeutet: wir können wie von den reellen Zahlen gewohnt rechnen. Sind  $a, b \in \mathbb{C}$ , so auch  $a+b, a-b, a \cdot b$  und  $\frac{a}{b}$ . Zu jeder komplexen Zahl  $a$  gibt es genau eine Zahl  $a'$  mit  $a+a'=0$  und zu jeder Zahl  $a \in \mathbb{C}$  ungleich 0 genau ein  $a'' \in \mathbb{C}$  mit  $a \cdot a'' = 1$ .  $a'$  wird in einem Körper mit  $-a$  und  $a''$  mit  $\frac{1}{a}$  bezeichnet. Außerdem gelten das Assoziativ-, das Distributiv- und das Kommutativgesetz. Wichtig ist, dass diese Eigenschaften, die uns von den reellen Zahlen sehr einleuchtend erscheinen, nicht überall gelten.

**Aufgabe A.1:** Beweisen Sie, dass weder die Menge der natürlichen Zahlen  $\mathbb{N} = \{0, 1, 2, \dots\}$  ein Körper ist noch die Menge der ganzen Zahlen  $\mathbb{Z} =$

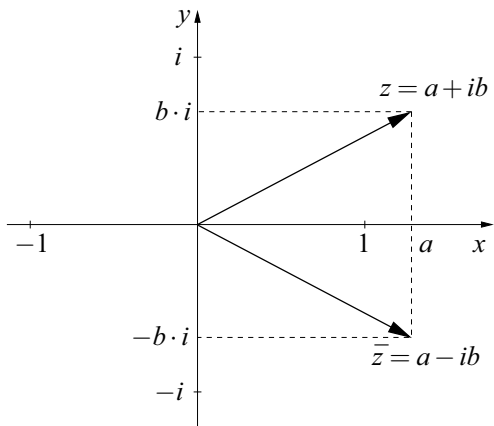


Abbildung A.1: Komplexe Zahlen in der reellen Ebene

$\{\dots, -2, -1, 0, 1, 2, \dots\}$ . Überprüfen Sie, welche der genannten Bedingungen gelten und welche nicht.

**Aufgabe A.2:** Wir betrachten die Menge  $\mathbb{F}_2 = \{0, 1\}$  mit der Addition  $\oplus$  und der Multiplikation  $\cdot$  (siehe auch Seite 16):

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

$\cdot$	0	1
0	0	0
1	0	1

Zeigen Sie, dass  $\mathbb{F}_2 = \{0, 1\}$  mit diesen Operationen ein Körper ist.

Was die komplexen Zahlen von den reellen unterscheidet ist, dass die Gleichung  $x^2 = -1$  dort lösbar ist. Also muss  $\sqrt{-1}$  in  $\mathbb{C}$  enthalten sein! Sei  $i$  eine Lösung von  $x^2 = -1$  und  $-i$  die andere. Man nennt  $i$  die imaginäre Einheit.

Definition  
komplexe Zahl

Eine komplexe Zahl  $z$  ist von der Form  $a + ib$  für reelle Zahlen  $a, b$ . Die Addition komplexer Zahlen ist als

$$(a + ib) + (a' + ib') = (a + a') + i(b + b')$$

definiert. Die Multiplikation als

$$(a + ib) \cdot (a' + ib') = aa' + i(ab' + a'b) + i^2bb' = aa' - bb' + i(ab' + a'b),$$

denn es gilt  $i^2 = -1$ . Wir bezeichnen mit  $z^*$  die zu  $z = a + ib$  komplex konjugierte Zahl  $a - ib$ . In der deutschsprachigen Literatur wird statt  $z^*$  manchmal die Bezeichnung  $\bar{z}$  verwendet.

Wir veranschaulichen uns die Menge der komplexen Zahlen als reelle Ebene wie in Abbildung A.1. Jede Zahl entspricht einem Vektor  $(a, b)$ .  $a$  ist die

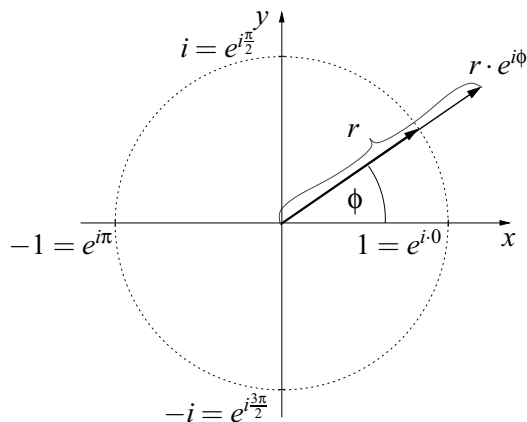


Abbildung A.2: Komplexe Zahlen als Längenangabe und Winkel

reelle Komponente und  $b$  die imaginäre. Der Betrag einer komplexen Zahl  $z$  ist als  $|z| = \sqrt{a^2 + b^2} = \sqrt{zz^*}$  definiert.

Außerdem lässt sich jede komplexe Zahl in der Form  $re^{i\phi}$  darstellen. Dabei ist  $e^{i\phi}$  ein Vektor in der reellen Ebene der Länge 1:  $\phi$  gibt den Winkel zur  $x$ -Achse an und wird als Phase bezeichnet (siehe Abbildung A.2).  $r$  ist die Länge des Vektors  $re^{i\phi}$ . In dieser Schreibweise lassen sich komplexe Zahlen multiplizieren, indem man die gewohnten Potenzrechenregeln anwendet:  $re^{i\phi} \cdot te^{i\chi} = rte^{i(\phi+\chi)}$ . Die Winkel addieren sich, und die Längen multiplizieren sich.

**Beispiel A.1:** Wir betrachten komplexe Zahlen der Länge  $1/\sqrt{2}$ : die reelle Zahl  $-1/\sqrt{2}$  ist als  $1/\sqrt{2} \cdot e^{i\pi}$  darstellbar und hat die Phase  $\pi$ . Auf gleiche Weise hat  $1/\sqrt{2}$  Phase 0,  $1/\sqrt{2}i$  Phase  $\pi/2$  und  $-1/\sqrt{2}i$  die Phase  $3/2\pi$ .

**Aufgabe A.3:** Zeigen Sie, dass  $re^{i\phi}$  und  $r \cos \phi + ir \sin \phi$  dieselben komplexen Zahlen sind. Nutzen Sie die obige Definition des Betrages  $|re^{i\phi}|$ .

## A.2 Vektorräume

### A.2.1 Was sind Vektorräume?

Unser Zugang zu dem Thema Vektorräume ist auf die Bedürfnisse der Quantenalgorithmen zugeschnitten. Wir erwähnen nur, was auch tatsächlich verwendet wird. Unseren Ausgangspunkt bilden Elemente von *komplexen Vektorräumen*:

$$v = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{n-1} \end{pmatrix}.$$

Diese *Vektoren* sind nichts anderes als  $n$  untereinander angeordnete komplexe Zahlen. Wir sprechen auch von  $n$ -*Tupeln*. Jedem solchen Spaltenvektor  $v$  entspricht ein Zeilenvektor  $v^T = (\alpha_0, \dots, \alpha_{n-1})$ , der *transponierte* Vektor. Es gilt:  $(\alpha_0, \dots, \alpha_{n-1})^T = v$ , siehe Abschnitt A.3.

Der Raum  $\mathbb{C}^n$

Das kartesische Produkt  $\mathbb{C}^n$  ist die Menge aller  $n$ -Tupel von komplexen Zahlen  $\alpha_0, \dots, \alpha_{n-1}$ . Addieren wir solche Tupel, gehen wir komponentenweise vor:

$$\begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{n-1} \end{pmatrix} + \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_{n-1} \end{pmatrix} = \begin{pmatrix} \alpha_0 + \beta_0 \\ \vdots \\ \alpha_{n-1} + \beta_{n-1} \end{pmatrix}. \quad (\text{A.1})$$

Außerdem können wir einen solchen Tupel aus  $\mathbb{C}^n$  mit einer komplexen Zahl  $a$  multiplizieren:

$$a \cdot \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{n-1} \end{pmatrix} = \begin{pmatrix} a \cdot \alpha_0 \\ \vdots \\ a \cdot \alpha_{n-1} \end{pmatrix}. \quad (\text{A.2})$$

Diese Definitionen sind sinnvoll, da wir komplexe Zahlen addieren und multiplizieren können. Wir haben diese Operationen von  $\mathbb{C}$  auf  $\mathbb{C}^n$  übertragen. Nun sind für  $\mathbb{C}^n$  die sogenannten *Vektorraumaxiome* erfüllt, die in jedem Buch über lineare Algebra zu finden sind. Ab jetzt ist ein Element aus  $\mathbb{C}^n$  ein *Vektor* und ein Element aus  $\mathbb{C}$  heißt *Skalar*. Die Vektorraumaxiome stellen gerade sicher, dass man Vektoren addieren kann, siehe (A.1), und dass die Skalarmultiplikation, siehe (A.2), sinnvoll definiert ist.

Bezug zum  
Quantenrechnen

Nun nähern wir uns diesen Vektorräumen auf abstraktem Wege. Wir betrachten zunächst *Linearkombinationen* beliebiger Objekte. Diese beschreiben wir als  $n$ -Tupeln beziehungsweise Vektoren. Im *Quantum Computing* spielen die Zustände von Quantenbits eine entscheidende Rolle. Wir beschreiben Superpositionen gerade als Linearkombinationen von gewissen Basiszuständen.

Erzeugenden-  
system

Sei  $M = \{|0\rangle, |1\rangle, \dots, |n-1\rangle\}$  eine Menge mit  $n$  Objekten. Eine Summe der Form

$$\alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle + \dots + \alpha_{n-1} \cdot |n-1\rangle \quad (\text{A.3})$$

mit  $\alpha_i$  aus  $\mathbb{C}$  heißt *Linearkombination über  $\mathbb{C}$* . Als *lineare Hülle* einer Menge  $M$  über  $\mathbb{C}$  bezeichnet man die Menge aller solchen Linearkombinationen.

$$\text{Span}_{\mathbb{C}} M = \{ \alpha_0 |0\rangle + \alpha_1 |1\rangle + \dots + \alpha_{n-1} |n-1\rangle \mid \alpha_0, \dots, \alpha_{n-1} \in \mathbb{C} \}$$

Wir fassen die Koeffizienten  $\alpha_0, \dots, \alpha_{n-1}$  als  $n$ -Tupel

$$\begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{n-1} \end{pmatrix} \quad (\text{A.4})$$

auf. Damit ist die lineare Hülle einer  $n$ -elementigen Menge  $M$  mit dem kartesischen Produkt  $\mathbb{C}^n$  identifizierbar: jeder Linearkombination aus (A.3) entspricht eindeutig der Vektor aus (A.4) aus  $\mathbb{C}^n$  und umgekehrt.

Den Elementen  $|0\rangle, |1\rangle, \dots, |n-1\rangle$  aus  $\text{Span}_{\mathbb{C}} M$  entsprechen die *Einheitsvektoren*

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \dots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \quad (\text{A.5})$$

**Aufgabe A.4:** Alles was wir über komplexe Vektorräume gesagt haben, lässt sich direkt auf Vektorräume über allgemeinen Körpern  $K$  übertragen. Notieren Sie alle Elemente von  $\text{Span}_{\mathbb{F}_2}\{a, b\}$  (siehe Aufgabe A.2).

## A.2.2 Basen und Unterräume

Sei  $V$  der Vektorraum  $\mathbb{C}^n$  den wir wie im vorhergehenden Abschnitt mit  $\text{Span}_{\mathbb{C}}\{|0\rangle, \dots, |n-1\rangle\}$  identifizieren. Die Menge  $\{|0\rangle, \dots, |n-1\rangle\}$ , die nach (A.5) der Menge der Einheitsvektoren entspricht, hat eine besondere Eigenschaft: damit lässt sich der gesamte Vektorraum erzeugen; und zwar mit Hilfe von Linearkombinationen. Außerdem ist diese Menge linear unabhängig.

Lineare  
Unabhängigkeit

Eine Menge von Vektoren  $v_1, \dots, v_m$  heißt *linear unabhängig*, wenn gilt: Eine Linearkombination  $\alpha_1 v_1 + \dots + \alpha_m v_m$  ist nur dann gleich dem Nullvektor, wenn alle Skalare  $\alpha_i = 0$  sind. Anders ausgedrückt: nur die triviale Linearkombination  $0 \cdot v_1 + \dots + 0 \cdot v_m$  ist gleich  $(0, \dots, 0)^T$ .

Das folgende Lemma verdeutlicht diesen Begriff:

**Lemma 1:** Eine Menge  $S$  von Vektoren ist genau dann linear abhängig, wenn gilt: Ein Vektor  $x$  aus  $S$  kann von den anderen Vektoren aus  $S$  erzeugt werden:

$$x \in \text{Span}(S \setminus x).$$

In einer linear abhängigen Menge sind also Vektoren enthalten, die in den Linearkombinationen nicht nötig sind. Linear unabhängige Mengen sind in gewissem Sinne minimal: Entfernt man ein Element, können weniger Vektoren erzeugt werden. Das führt zu der folgenden Definition:

Vektoren  $v_1, \dots, v_m$  eines Vektorraumes  $V$  heißen eine *Basis* von  $V$ , wenn

Basis

- sie linear unabhängig sind und
- ihre lineare Hülle ganz  $V$  ergibt:  $\text{Span}\{v_1, \dots, v_m\} = V$ .

Jeder Vektor von  $V$  kann also als eine Linearkombination von Basisvektoren dargestellt werden. Unsere Überlegungen im letzten Abschnitt ergeben damit unmittelbar, dass die Einheitsvektoren aus Gleichung (A.5) eine Basis von  $\mathbb{C}^n$  ergeben. Wir nennen diese Basis *Standardbasis*. Ebenso fassen wir  $|0\rangle, \dots, |n-1\rangle$  als Basis von  $\mathbb{C}^n$  auf.



## Dimension

Alle Basen eines Vektorraumes haben die gleiche Größe. Diese Größe bezeichnet man als die *Dimension* des Vektorraumes.

**Aufgabe A.5:** Beweisen Sie Lemma 1.

**Aufgabe A.6:** Welche der folgenden Mengen sind Basen von  $\mathbb{C}^2$ ?

- $\{(5,1)^T, (1,0)^T\}$
- $\{(5,0)^T, (1,0)^T\}$
- $\{\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\}$

## Unterräume

Eine Menge  $U$  von Vektoren aus  $V$  heißt ein *Unterraum* von  $V$ , wenn  $U$  alle Linearkombinationen von Vektoren aus  $U$  enthält, wenn also  $\text{Span} U = U$  gilt. Wir können uns leicht Unterräume erzeugen, indem wir aus einer Basis von  $V$  einzelne Elemente auswählen und deren lineare Hülle betrachten. Eindimensionale Unterräume von  $\mathbb{C}^n$  heißen *Geraden*, zweidimensionale *Ebenen* und  $(n-1)$ -dimensionale *Hyperebenen*.

### A.2.3 Winkel und Abstände in einem Vektorraum

Viele Begriffe aus der Geometrie lassen sich auf Vektorräume übertragen. Dazu dient das *Skalarprodukt*.

## Skalarprodukt

Das *innere Produkt* oder *Skalarprodukt* ordnet einem Paar von Vektoren einen Skalar zu. Im Fall von komplexen Vektoren ist es damit eine Abbildung  $\mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C}$ . Ab jetzt (und im Hauptteil des Buches durchgängig) verwenden wir für komplexe Vektoren die Braket-Notation. Beschreibt ein komplexer Vektor  $u$  den Zustand eines Quantensystems, bezeichnen wir ihn mit  $|u\rangle$ . Für  $|u\rangle = (u_1, \dots, u_n)$  und  $|v\rangle = (v_1, \dots, v_n)$  aus  $\mathbb{C}^n$  ist

$$\langle u | v \rangle = u_1^* v_1 + \dots + u_n^* v_n.$$

Für Vektoren  $x, y \in \{0, 1\}^n$  lautet das Skalarprodukt

$$x \cdot y = x_1 y_1 \oplus x_2 y_2 \oplus \dots \oplus x_n y_n$$

und ist 0 oder 1. *Folgende Eigenschaften des Skalarproduktes gelten für alle Arten von Vektorräumen, wenn wir sie auch in der Braket-Schreibweise angeben.*

## Länge

Mit Hilfe des Skalarproduktes definieren wir die Länge eines Vektors:

$$\|v\| = \sqrt{\langle v | v \rangle}$$

$\|v\|$  bezeichnet man als die *Norm* von  $v$ . Eine wichtige Eigenschaft der Norm ist die *Dreiecksungleichung*:

$$\|u + v\| \leq \|u\| + \|v\|$$

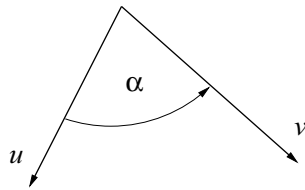


Abbildung A.3: Winkel zwischen zwei Vektoren

**Aufgabe A.7:** Berechnen Sie die Norm des Vektors  $(1, 1)^T$  im  $\mathbb{C}^2$ . Überlegen Sie sich, dass dieser Begriff mit dem gewohnten Längenbegriff zusammenfällt, wenn wir uns in dem Raum  $\mathbb{R}^2$  befinden.

Den *Abstand* zweier Vektoren  $u$  und  $v$  definieren wir als

Abstand

$$\|u - v\|.$$

Mit dem Skalarprodukt kann man den im  $\mathbb{R}^2$  und  $\mathbb{R}^3$  anschaulichen Begriff des Winkels auf abstraktere Vektorräume übertragen: Zwei Vektoren  $u$  und  $v$  heißen *orthogonal*, wenn  $\langle u | v \rangle = 0$  gilt. Wir definieren den Winkel  $\alpha$  zwischen  $u$  und  $v$  mittels

Winkel

$$\cos \alpha = \frac{\langle u | v \rangle}{\|u\| \|v\|}, \quad (\text{A.6})$$

siehe Abbildung A.3.

Für unsere Quantensysteme benötigen wir Basen mit folgenden Eigenschaften:

Orthonormalbasis

- jeder Basisvektor hat die Länge 1, und
- die Basisvektoren stehen senkrecht aufeinander, sind orthogonal.

Man spricht von *Orthonormalbasen*: für so eine Basis  $b_1, \dots, b_n$  gilt also:

- $\|b_i\| = 1$  für alle  $i = 1, \dots, n$ , und
- $\langle b_i | b_j \rangle = 0$  für  $1 \leq i < j \leq n$ .

Die wichtigste Orthonormalbasis ist die Standardbasis aus (A.5).

**Aufgabe A.8:** Zeigen Sie: Hat das Skalarprodukt zweier Einheitsvektoren  $u, v$  den Wert 1,

$$\langle u | v \rangle = 1,$$

so sind die Vektoren gleich:

$$u = v.$$

Berechnen Sie die Norm des Vektors  $(1, 1)^T$  im  $\mathbb{C}^2$ . Überlegen Sie sich, dass dieser Begriff mit dem gewohnten Längenbegriff zusammenfällt, wenn wir uns auf den Raum  $\mathbb{R}^2$  beschränken.

## Hilberträume

Ein Vektorraum mit Skalarprodukt heißt *Hilbertraum*, wenn eine weitere Eigenschaft erfüllt ist, die wir hier nur der Vollständigkeit halber nennen: jede Cauchy-Folge muss konvergieren. Diese Eigenschaft ist für die in diesem Buch verwendeten Räume immer erfüllt, spielt für uns aber keine Rolle. Darum verwenden wir den Begriff Hilbertraum nicht und erwähnen ihn hier nur, um dem Leser den in vielen Publikationen gebrauchten Begriff zu erläutern.

### A.2.4 Projektionen

Mit dem Skalarprodukt können wir die Projektion eines Vektors auf einen Unterraum definieren. Das einfachste Beispiel ist die Projektion auf eine Koordinatenachse. In Abbildung A.1 entspricht der Vektor

$$\begin{pmatrix} a \\ b \end{pmatrix}$$

der komplexen Zahl  $z$ . Die Projektion dieses Vektors auf die  $x$ -Achse ist der Vektor

$$\begin{pmatrix} a \\ 0 \end{pmatrix},$$

entsprechend ist die Projektion auf die  $y$ -Achse gerade

$$\begin{pmatrix} 0 \\ b \end{pmatrix}.$$

Anschaulich gesprochen ist die Projektion eines Vektors auf einen Unterraum dessen Komponente *in dem Unterraum*. Nun zur formalen Definition. Die Komponente eines Vektors in einem Unterraum wird mit dem Skalarprodukt bestimmt.

$$\begin{pmatrix} a \\ b \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = a, \quad \begin{pmatrix} a \\ b \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = b.$$

Für die formale Definition der Projektion verwenden wir die Bracket-Schreibweise.

Definition  
Projektion

Sei  $b_1, \dots, b_m$  Orthonormalbasis eines Unterraums  $E$ . Die Projektion eines Vektors  $v$  auf diesen Unterraum ist der Vektor

$$\langle b_1 | v \rangle \cdot b_1 + \langle b_2 | v \rangle \cdot b_2 + \dots + \langle b_m | v \rangle \cdot b_m.$$

**Aufgabe A.9:** Berechnen Sie die Projektion der Vektoren

$$\begin{pmatrix} 2 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix} \text{ und } \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

in den Raum

$$\text{Span}\left\{\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}\right\}$$

und veranschaulichen Sie das Ergebnis durch eine Zeichnung.

In obiger Definition fassen wir einen Bra-Vektor  $\langle b|$  als Abbildung auf. Ein Ket-Vektor  $|v\rangle$  wird auf das Skalarprodukt  $\langle b|v\rangle$  abgebildet. Formal ist  $|b\rangle$  ein Spaltenvektor

$$|b\rangle = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}$$

und der zugehörige Bra-Vektor  $\langle b|$  der Zeilenvektor  $(a_1^*, \dots, a_n^*)$  mit komplex adjungierten Komponenten;  $\langle b|v\rangle$  ist das Produkt. Mehr dazu im nächsten Abschnitt.

## A.3 Matrizen

Matrizen sind für uns von besonderer Bedeutung, da sich damit Rechenschritte auf Quantenbits beschreiben lassen. Matrizen lassen sich mit Vektoren multiplizieren und ergeben auf diese Weise einen neuen Vektor. Somit beschreiben Matrizen Abbildungen zwischen Vektorräumen.

Eine Matrix ist ein rechteckiges Feld von Einträgen. Zum Beispiel ist

Definition Matrix

$$A = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

eine  $2 \times 3$ -Matrix mit ganzen Zahlen. Die *transponierte* Matrix  $A^T$  erhalten wir, wenn wir die Zeilen mit den Spalten vertauschen. Für unser Beispiel  $A$  erhalten wir

$$A^T = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

Einen Vektor

$$v = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}$$

können wir als  $n \times 1$ -Matrix auffassen. Einem solchen *Spaltenvektor* korrespondiert ein *Zeilenvektor*

$$v^T = (a_1, \dots, a_n).$$

Sprechen wir von der Matrix  $A = (a_{ij})$ , meinen wir die Matrix, die in Zeile  $i$  und Spalte  $j$  den Eintrag  $a_{ij}$  besitzt.

## Multiplikation

Die Multiplikation eines Vektors  $v$  mit einer  $n \times n$ -Matrix  $A = (a_{ij})$  ist wie folgt definiert:

$$Av = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} a_{11}v_1 + a_{12}v_2 + \dots + a_{1n}v_n \\ a_{21}v_1 + a_{22}v_2 + \dots + a_{2n}v_n \\ \vdots \\ a_{n1}v_1 + a_{n2}v_2 + \dots + a_{nn}v_n \end{pmatrix}$$

Die Multiplikation von allgemeinen Matrizen definiert man analog. Ist  $A = (a_{ij})$  eine  $l \times m$ -Matrix und  $B = (b_{jk})$  eine  $m \times n$ -Matrix, so ist das Produkt

$$C = AB$$

die  $l \times n$ -Matrix  $C = (c_{ik})$  mit

$$c_{ik} = \sum_{j=1}^m a_{ij}b_{jk}.$$

## Matrizen als Abbildungen

Für uns sind insbesondere quadratische Matrizen wichtig, deren Einträge komplexe Zahlen sind. Solche Matrizen entsprechen Abbildungen  $f$  von  $n$ -dimensionalen komplexen Vektorräumen in sich selbst. Ein Vektor  $v$  wird mittels der Matrixmultiplikation auf sein *Bild*  $w = Av$  abgebildet:

$$f: \mathbb{C}^n \longrightarrow \mathbb{C}^n, v \mapsto Av.$$

Die Einheitsmatrix

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

entspricht der Identität:  $I \cdot v = v$  für alle Vektoren  $v$ .

## Inverse

Als Inverse einer Matrix  $A$  bezeichnen wir eine Matrix  $A^{-1}$  mit  $AA^{-1} = A^{-1}A = I$ . Eine Matrix  $A$  heißt *invertierbar*, wenn eine solche Inverse existiert. Invertierbare Matrizen entsprechen bijektiven Abbildungen.

Die den Matrizen entsprechenden Abbildungen sind *linear*:  $A(v+w) = Av + Aw$  und  $A(\alpha v) = \alpha(Av)$  für Vektoren  $v, w$  und eine komplexe Zahl  $\alpha$ .

Reelle Matrizen  $A$  heißen *orthogonal*, falls  $A^T = A^{-1}$ . Diese Matrizen entsprechen Abbildungen, die aus Drehungen und Spiegelungen zusammengesetzt sind. Insbesondere werden Längen und Winkel nicht verändert. Der in Abschnitt 2.3 eingeführte Begriff *unitär* überführt diese Idee auf komplexe Matrizen.

## A.4 Kombinatorik und Wahrscheinlichkeit

Eines der einfachsten und wichtigsten kombinatorischen Prinzipien ist das Schubfachprinzip

- Verstauen wir  $n$  Socken in  $m$  Schubfächern, dann gibt es mindestens ein Schubfach, das mindestens  $\lceil n/m \rceil$  Socken enthält (dabei ist  $\lceil x \rceil$  die kleinste ganze Zahl größer oder gleich  $x$ ).

Das ist anschaulich völlig klar; in weniger anschaulichen Situationen kann diese explizite Formulierung hilfreich sein.

Mit den Begriffen der Wahrscheinlichkeitstheorie lassen sich Situationen modellieren, in denen Zufall eine Rolle spielt. Wir werden einige Ideen und Begriffe zusammentragen. Ein Beispiel, auf das wir mehrmals zu sprechen kommen, ist der faire Würfelwurf: Das Ergebnis ist eine der Zahlen  $1, \dots, 6$  – jede wird mit der gleichen Wahrscheinlichkeit geworfen.

Wahrscheinlichkeit

Um eine solche Situation allgemein zu modellieren, betrachten wir eine Menge mit  $m$  Objekten, die wir Grundereignisse nennen:  $M = \{1, \dots, m\}$ . Eine Zufallsvariable ist eine Funktion, deren Bilder aus der Menge  $M$  stammen. Wir interessieren uns für die Wahrscheinlichkeitsverteilung einer Zufallsvariablen, die jedem Grundereignis dessen Wahrscheinlichkeit zuordnet.

Zufallsvariable

$$P(X = 1) = 1/4$$

bedeutet, dass in einem Viertel der Fälle das Grundereignis 1 eintritt ( $P$  kürzt engl. *probability* ab).

**Beispiel A.2:** Im Falle des fairen Würfelwurfes ist  $M = \{1, \dots, 6\}$ ; die Zufallsvariable hat die Verteilung

$$P(X = 1) = \dots = P(X = 6) = 1/6.$$

◇

Die Wahrscheinlichkeit eines Ereignisses ist eine Zahl  $p$  mit  $0 \leq p \leq 1$ . Für  $M = \{1, \dots, m\}$  und  $P(X = i) = p_i$  für  $i = 1, \dots, m$  gilt  $p_1 + \dots + p_m = 1$ .  $p_i$  ist die Wahrscheinlichkeit des Grundereignisses  $i$ ; die Summe dieser Wahrscheinlichkeiten muss 1 ergeben, da ja mit Sicherheit eines davon eintreten wird.

Wir sprechen von *Gleichverteilung*, wenn wie in Beispiel A.2 jedes der Grundereignisse die gleiche Wahrscheinlichkeit hat:  $p_i = 1/m$  für  $i = 1, \dots, m$ .

Ein Ereignis kann nun aus mehreren Grundereignissen bestehen, ist also eine Teilmenge von  $M$ . Wie groß ist die Wahrscheinlichkeit, dass das Ereignis  $\{2, 5, 6\}$  eintritt? Es gilt

Ereignisse

$$P(X \in \{2, 5, 6\}) = P(X = 2) + P(X = 5) + P(X = 6) = 1/2.$$

Als nächstes führen wir zwei (unabhängige, siehe unten) Zufallsexperimente aus. Das erste wird durch eine Zufallsvariable  $X$  und das zweite durch eine

namens  $Y$  beschrieben. Ist nun  $A$  ein Ereignis des ersten Experiments und  $B$  eines des zweiten, gilt für die Wahrscheinlichkeit, dass beide Ereignisse eintreten:

$$P(X \in A \text{ und } Y \in B) = P(X \in A) \cdot P(Y \in B).$$

**Beispiel A.3:** Wir führen nacheinander zwei Experimente aus, denen wir die Zufallsvariablen  $X$  und  $Y$  zuordnen. Zuerst werfen wir einen sechsseitigen Würfel und im zweiten Experiment zwei solche Würfel. Es gilt

$$P(X = i) = 1/6 \text{ für } i = 1, \dots, 6$$

und

$$P(Y = (i, j)) = 1/36 \text{ für } i, j = 1, \dots, 6.$$

Das Ergebnis des zweiten Experiments ist also eines der Zahlenpaare  $(1, 1), (1, 2), \dots, (5, 6), (6, 6)$ .

Wir fragen uns: Wie groß ist die Wahrscheinlichkeit, im ersten Experiment eine gerade Zahl zu erhalten und im zweiten die Summe 3? Das entsprechende Ereignis für den ersten Wurf ist die Menge der geraden Würfelseiten  $A = \{2, 4, 6\}$ . Das der zweiten die Menge  $B = \{(1, 2), (2, 1)\}$ . Es gilt  $P(X \in A) = p_2 + p_4 + p_6 = 1/2$  und  $P(Y \in B) = p_{(1,2)} + p_{(2,1)} = 1/18$ .

Die Wahrscheinlichkeit, dass beide Ereignisse eintreten ist

$$P(X \in A \text{ und } Y \in B) = 1/2 \cdot 1/18 = 1/36.$$

◇

## Unabhängigkeit

Das gilt aber nur, wenn die Experimente voneinander *unabhängig* sind! Wir können davon ausgehen, dass der erste Würfelwurf die weiteren nicht beeinflusst. Aber zum Beispiel sind die Ereignisse *Das erste Ergebnis ist gerade* und *Das zweite Ergebnis ist doppelt so groß, wie das erste* nicht unabhängig!

Zwei Zufallsvariablen  $X, Y$  und die modellierten Experimente sind gerade dann unabhängig, wenn für alle Ereignisse  $A, B$  gilt:  $P(X \in A \text{ und } Y \in B) = P(X \in A) \cdot P(Y \in B)$ .

In diesem Buch haben wir es zumeist mit unabhängigen Zufallsexperimenten zu tun und erwähnen das nicht eigens. Besonders häufig benötigen wir unabhängige gleichverteilte Zufallsvariablen. Verwenden wir in diesem Buch die Sprechweise: Wir wählen *zufällig* ein Element aus  $M = \{1, \dots, m\}$ , dann ist damit eine Aktion mit Gleichverteilung gemeint. Wählen wir auf diese Weise nacheinander, dann unabhängig voneinander.

Interessant ist der Begriff der bedingten Wahrscheinlichkeit: Wie hoch ist die Wahrscheinlichkeit, mit zwei Würfeln eine Summe größer 6 zu erhalten, wenn eines der Ergebnisse 2 ist? Es gilt: die Wahrscheinlichkeit, dass  $A$  unter der Bedingung  $B$  eintritt ist  $P(A \cap B)/P(B)$ .

## A.5 Ganze Zahlen

Dieser Abschnitt versammelt einige Begriffe der Zahlentheorie, also der mathematischen Theorie, die sich mit den ganzen Zahlen  $\mathbb{Z}$  beschäftigt. Diese Begriffe werden in Kapitel 8 verwendet.

### A.5.1 Teiler und Vielfache

Mit  $a$  und  $b$  bezeichnen wir zwei ganze Zahlen.  $a$  *teilt*  $b$ , wenn  $b$  ein Vielfaches von  $a$  ist: wenn es eine ganze Zahl  $k$  gibt, mit

$$b = a \cdot k.$$

Jede Zahl wird von 1 und sich selbst geteilt; 0 wird von jeder Zahl geteilt. Ein Teiler von  $b$ , der weder 1 noch  $b$  selbst ist, heißt *echter Teiler* von  $b$ .

Der *größte gemeinsame Teiler* der Zahlen  $a, b$  ist – wie der Name sagt – die größte Zahl, die sowohl  $a$  als auch  $b$  teilt. Wir bezeichnen diese Zahl als  $\text{ggT}(a, b)$ . Jeder gemeinsame Teiler zweier Zahlen  $a, b$  teilt auch  $\text{ggT}(a, b)$ . Der größte gemeinsame Teiler lässt sich mit Euklids Algorithmus mit  $O(\log a)$  rekursiven Aufrufen berechnen, falls  $a > b$ . In jeder Rekursionsstufe wird eine Division mit Rest ausgeführt, was in Zeit  $O((\log a)^2)$  möglich ist. Damit hat Euklids Algorithmus kubische Laufzeit  $O((\log a)^3)$ .

Euklids  
Algorithmus

Die Zahlen  $a$  und  $b$  heißen *teilerfremd*, wenn 1 der größte gemeinsame Teiler ist:  $\text{ggT}(a, b) = 1$ . Eine Zahl  $p$  heißt *Primzahl*, wenn 1 und  $p$  die einzigen Teiler sind. Für eine Primzahl  $p$  und beliebige ganze Zahlen  $a, b$  gilt:

Aus  $p$  teilt  $a \cdot b$  folgt:  $p$  teilt  $a$  oder  $p$  teilt  $b$ .

### A.5.2 Modulares Rechnen

Multiplizieren wir zwei ganze Zahlen  $a, b$ , so ist das Produkt  $a \cdot b$  wieder eine ganze Zahl. Das ist eine wichtige Eigenschaft der ganzen Zahlen  $\mathbb{Z}$ , denn damit können wir hemmungslos multiplizieren. Computer können jedoch nur Zahlen kleiner einer bestimmten Schranke darstellen: ein 8-Bit-Register kann die Zahlen  $\{0, \dots, 255\}$  darstellen. Die Zahl  $100 \cdot 100$  liegt jedoch nach der gewöhnlichen ganzzahligen Multiplikation außerhalb dieser Menge. Um auf einer beliebigen Menge

$$\{0, \dots, N-1\}$$

rechnen zu können, führt man die Operationen *modulo*  $N$  ein. Wir definieren: Zwei Zahlen  $a$  und  $b$  sind *kongruent modulo*  $N$ ,

$$a \equiv b \pmod{N},$$

falls ihre Differenz ein Vielfaches von  $N$  ist, falls also  $a = b + k \cdot N$  für eine ganze Zahl  $k$  gilt.



**Beispiel A.4:**

1. Alle geraden Zahlen sind kongruent modulo 2, ebenso alle ungeraden.  
Wenn  $a$  gerade ist, gilt nämlich  $a = 0 + 2k$ , für eine ganze Zahl  $k$ , oder

$$a \equiv 0 \pmod{2}.$$

Genauso gilt

$$1 \equiv 3 \equiv 5 \equiv \dots \pmod{2}.$$

2. Für  $N = 10$  gilt

$$-9 \equiv 1 \equiv 11 \equiv 21 \pmod{10},$$

$$-8 \equiv 2 \equiv 12 \equiv 22 \pmod{10},$$

und

$$-7 \equiv 3 \equiv 13 \equiv 23 \pmod{10}.$$

◇

Dieser Begriff ist mit der Addition und der Multiplikation verträglich: Gelten

$$a \equiv a' \pmod{N} \text{ und } b \equiv b' \pmod{N},$$

so ist

$$a + b \equiv a' + b' \pmod{N}$$

und

$$a \cdot b \equiv a' \cdot b' \pmod{N}.$$

Rechnen wir mit nur  $N$  Zahlen, fassen also alle kongruenten Zahlen als *gleichwertig* auf und beschränken uns auf die Menge

$$\mathbb{Z}_N = \{0, \dots, N-1\}.$$

Wir definieren

$$a \bmod N$$

als die Zahl  $b \in \mathbb{Z}_N$  mit

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

Damit können wir Summe und Produkt von Elementen oder „Zahlen“  $a, b$  in  $\mathbb{Z}_N$  definieren, und zwar als

$$a + b \bmod N \text{ und } a \cdot b \bmod N.$$

Wir verwenden also die gewohnten Operationen in  $\mathbb{Z}$  und bilden das Ergebnis auf das kongruente Element in  $\mathbb{Z}_N$  ab.

**Beispiel A.5:**

1. Unsere Addition in  $\{0, 1\}$  ergibt sich auf diese Weise:

$$a \oplus b = a + b \bmod 2.$$

$\oplus$  ist die eben definierte Addition in  $\mathbb{Z}_2$ .

2. In  $\mathbb{Z}_5$  gilt  $4 \cdot 3 = 2$ , denn  $12 \equiv 2 \bmod 5$ .
3. Wir berechnen  $100 \cdot 100$  in  $\mathbb{Z}_{256}$ . Es gilt  $10000 = 39 \cdot 256 + 16$  oder  $100 \cdot 100 \equiv 16 \bmod 256$ . Damit ist 16 das Ergebnis unserer Multiplikation in  $\mathbb{Z}_{256}$ . Dies wäre auch das Ergebnis bei einer Berechnung in einem 8-Bit-Register.

**A.5.3 Zur Division**

Ganze Zahlen können wir problemlos multiplizieren:  $3 \cdot 3$  ergibt 9 und somit wieder eine ganze Zahl. Doch können wir nicht jede Division ausführen. 5 durch 2 ergibt keine ganze Zahl; 2 teilt 5 nicht. Aus diesem Grund spielt der Teiler eine wichtige Rolle in der Theorie der ganzen Zahlen. Wie verhält es sich in den Mengen  $\mathbb{Z}_N$ ?

Folgende Tabelle stellt die Multiplikation in  $\mathbb{Z}_5$  dar:

$\cdot$	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

Betrachten wir die Zeile mit den Vielfachen von 3, sehen wir: jeder Wert  $0, \dots, 4$  kommt vor. Wir können jedes Element durch drei teilen. Dies gilt auch für 1, 2 und 4.

Insbesondere gilt  $3 \cdot 2 = 1 \bmod 5$ . Drei teilt die 1 und damit alle Vielfachen der 1. Wir nennen 2 das *multiplikativ Inverse* von 3: Wir können in  $\mathbb{Z}_5$  durch drei teilen, indem wir mit 2 multiplizieren. Allgemein ist das multiplikativ Inverse einer Zahl  $a$  in  $\mathbb{Z}_N$  die Lösung der Gleichung

$$a \cdot x \equiv 1 \bmod N.$$

Zum Beispiel hat in  $\mathbb{Z}_5$  das Element 4 das multiplikativ Inverse  $x = 4$ . 0 hat kein multiplikativ Inverses; 0 hat auch als ganze Zahl keine Teiler. Wir bezeichnen das multiplikativ Inverse von  $a$  mit  $a^{-1}$ .

**Aufgabe A.10:** Stellen Sie die Multiplikationstabelle für  $\mathbb{Z}_4$  auf.

Die Menge  $\mathbb{Z}_N^*$  enthält alle Elemente aus  $\mathbb{Z}_N$ , die ein multiplikativ Inverses besitzen. Man bezeichnet  $\mathbb{Z}_N^*$  als die *multiplikative Gruppe* von  $\mathbb{Z}_N$ .

**Beispiel A.6:** Es gilt  $\mathbb{Z}_5^* = \{1, 2, 3, 4\}$  und  $\mathbb{Z}_4^* = \{1, 3\}$ .

Für eine Primzahl  $p$  gilt

$$\mathbb{Z}_p^* = \{1, \dots, p-1\} = \mathbb{Z}_p \setminus \{0\}.$$

Ist  $N$  eine zusammengesetzte Zahl, haben neben der Null auch andere Elemente kein multiplikativ Inverses. Wir bezeichnen die Größe der Menge  $\mathbb{Z}_N^*$  mit  $\phi(N)$ ; das ist die als *Eulersche  $\phi$ -Funktion* (nach dem Schweizer Mathematiker Leonard Euler (1707-1783)).

Für eine Primzahl  $p$  gilt

$$\phi(p) = p - 1.$$

Für zwei verschiedene Primzahlen  $p, q$  gilt

$$\phi(pq) = (p-1)(q-1).$$

Also hat die Menge  $\mathbb{Z}_6^*$  gerade  $\phi(2 \cdot 3) = 2$  Elemente; neben 1 noch 5:  $5 \cdot 5 \equiv 1 \pmod{6}$ . Der folgende *Satz von Euler* oder auch *Satz von Euler-Fermat* spielt für uns eine wichtige Rolle:

Satz von Euler

Für jedes Element  $a$  aus  $\mathbb{Z}_N^*$  gilt

$$a^{\phi(N)} \equiv 1 \pmod{N}.$$

Daraus folgt direkt der *kleine Satz von Fermat*:

Kleiner Satz von Fermat

Für jede natürliche Zahl  $a \neq 0$  und eine Primzahl  $p$ , die  $a$  nicht teilt, gilt

$$a^{p-1} \equiv 1 \pmod{p}.$$

# B Lösungen ausgewählter Übungsaufgaben

Hier finden Sie knapp gehaltene Lösungen oder zumindest deutliche Lösungshinweise für wichtige Übungsaufgaben.

**Aufgabe 2.3:** Für jedes der Qubits erhalten wir  $|0\rangle$  mit Wahrscheinlichkeit  $1/4$  und  $|1\rangle$  mit Wahrscheinlichkeit  $3/4$ . Also erwarten wir, dass etwa 25 der Messungen  $|0\rangle$  ergeben und der Rest, also rund 75,  $|1\rangle$ . Kapitel 2

**Aufgabe 2.4:**  $H$  ist symmetrisch und hat nur reelle Einträge, also gilt  $H^* = H$ . Wir müssen zeigen, dass  $H$  zu sich selbst invers ist:  $H \cdot H = I_2$  ist leicht nachzurechnen.

**Aufgabe 2.5:** Die der Transformation entsprechende Matrix muss von der Form

$$A = \begin{pmatrix} \frac{1}{2} & x \\ \frac{\sqrt{3}}{2} & y \end{pmatrix}$$

sein. Aus der Unitarität von  $A$  folgt die Bedingung

$$A^\dagger \cdot A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Daraus entstehen vier Gleichungen mit den Lösungen

$$x = 1/2 \cdot \sqrt{3}e^{it}, y = -1/2 \cdot e^{it} \text{ mit } 0 \leq t < 2\pi.$$

Die reellen Spezialfälle sind durch die Matrizen

$$\frac{1}{2} \begin{pmatrix} 1 & -\sqrt{3} \\ \sqrt{3} & 1 \end{pmatrix} \text{ und } \frac{1}{2} \begin{pmatrix} 1 & \sqrt{3} \\ \sqrt{3} & -1 \end{pmatrix}$$

gegeben.

**Aufgabe 2.6:** Nach Schritt 2 ist das Bit im Zustand  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ . Wir erhalten  $|0\rangle$  mit Wahrscheinlichkeit  $(\frac{1}{\sqrt{2}})^2 = 1/2$  und  $|1\rangle$  mit Wahrscheinlichkeit  $(-\frac{1}{\sqrt{2}})^2 = 1/2$ . Das Ergebnis ist also das gleiche.

**Aufgabe 2.9:** Es genügt, sich folgendes klarzumachen: Wenn  $A$  eine Permutationsmatrix ist, so tauscht  $A^T$  die Elemente gerade wieder zurück.

**Aufgabe 2.10:** Wir geben alle Möglichkeiten an, den Eingaben eine Ausgabe zuzuordnen:

1.  $0 \rightarrow 0, 1 \rightarrow 0,$
2.  $0 \rightarrow 0, 1 \rightarrow 1,$
3.  $0 \rightarrow 1, 1 \rightarrow 0,$
4.  $0 \rightarrow 1, 1 \rightarrow 1$

**Aufgabe 2.11:** wir können die Aufgabe lösen indem wir für jede Wahl von  $f$  die Matrix von  $U_f$  aufschreiben und mit  $U_f^*$  multiplizieren. Eleganter überlegen wir uns, dass  $U_f$  für jede Wahl von  $f$  eine Permutationsmatrix ist. Dann folgt die Aussage mit Übung 2.9.

**Aufgabe 2.12:** Wir vermeiden es, den Vektor  $(-1, -1)^T$  mit der Matrix  $H$  zu multiplizieren und nutzen stattdessen die Linearität.

Es gilt  $H(-(|0\rangle + |1\rangle)) = -(H|0\rangle + H|1\rangle)$ , da unitäre Transformationen linear sind. Wir wenden die Transformationen an und erhalten

$$-\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) = -2/\frac{1}{\sqrt{2}}|0\rangle = -\sqrt{2}|0\rangle.$$

Ein anderer Rechenweg:  $H(-(|0\rangle + |1\rangle)) = -\sqrt{2}H\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = -\sqrt{2}|0\rangle.$

**Aufgabe 2.14:**

$$\begin{aligned} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle) \end{aligned}$$

**Aufgabe 2.15:**

$$H \otimes I_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}, I_2 \otimes H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

**Aufgabe 2.17:** Eine Basis ist orthogonal, wenn für je zwei ihrer Elemente das Skalarprodukt 0 ist, siehe Abschnitt A.2.3 im Anhang. Wir müssen also zeigen, dass  $\langle 0' | 1' \rangle = 0$  gilt. Der Zustandsvektor von  $|+\rangle$  ist

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

der von  $|-\rangle$  ist

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Nun genügt es, das Skalarprodukt zu berechnen.

**Aufgabe 2.20:** Wenden wir  $H$  auf das erste Bit an und anschließend CNOT sehen wir wie auf Seite 53:

$$\begin{aligned} |01\rangle &\xrightarrow{H \otimes I_2} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|1\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |11\rangle) \\ &\xrightarrow{\text{CNOT}} \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \\ |10\rangle &\xrightarrow{H \otimes I_2} \frac{1}{\sqrt{2}}(|00\rangle - |10\rangle) \xrightarrow{\text{CNOT}} \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \\ |11\rangle &\xrightarrow{H \otimes I_2} \frac{1}{\sqrt{2}}(|01\rangle - |11\rangle) \xrightarrow{\text{CNOT}} \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \end{aligned}$$

**Aufgabe 2.21:** Gäbe es eine solche Zerlegung von  $|\Phi^+\rangle$ , so könnten wir die Gleichung

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = (\alpha_0|0\rangle + \alpha_1|1\rangle) \cdot (\beta_0|0\rangle + \beta_1|1\rangle)$$

lösen. Es würde gelten:

$$\alpha_0\beta_0 = \alpha_1\beta_1 = \frac{1}{\sqrt{2}}$$

und

$$\alpha_0\beta_1 = \alpha_1\beta_0 = 0.$$

Das ist nicht möglich.

**Aufgabe 2.22:** Der angegebene Zustand ist in Bitschreibweise gleich  $1/2(|0000\rangle + |0011\rangle + |1100\rangle + |1111\rangle)$  und hat die Zerlegung  $|\Phi^+\rangle \otimes |\Phi^+\rangle$ .

**Aufgabe 2.23:** Es gilt  $z \neq 0$ , wir betrachten

$$\pm \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{x \cdot z} |z\rangle.$$

Seien  $z_{i_1}, \dots, z_{i_v}$  die Komponenten von  $z$  mit Wert 1. Dann gilt für die Hälfte der  $x \in \{0, 1\}^n$ , dass eine *gerade* Anzahl der Komponenten  $x_{i_1}, \dots, x_{i_v}$  den Wert 1 hat; für diese  $x$  gilt  $x \cdot z = 0$ . Für die andere Hälfte gilt  $x \cdot z = 1$ . Also ist

$$\pm \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{x \cdot z} |z\rangle = \pm \frac{1}{2^n} \left( \frac{1}{2^{n-1}} - \frac{1}{2^{n-1}} \right) |z\rangle = 0.$$

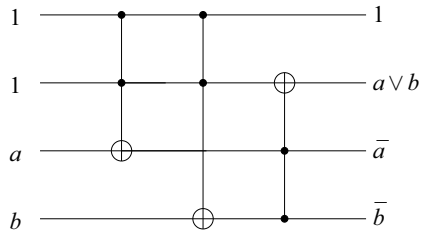


Abbildung B.1: Disjunktion aus drei Toffoligattern

**Aufgabe 2.24:** Nach Schritt 4 befindet sich das Quantenregister in demselben Zustand wie nach Schritt 4 von Deutsch-Jozsa:

$$\left( \frac{1}{2^n} \sum_{z=0}^{2^n-1} \sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot z} |z\rangle \right) \cdot \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).$$

Nun muss nur noch die Amplitude von  $|z\rangle$  für die beiden Fälle  $z = a$  und  $z \neq a$  berechnet werden.

### Kapitel 3

**Aufgabe 3.5:** Um das einzusehen, benötigen wir die Regel von De Morgan:

$$\overline{a \wedge b} = \bar{a} \vee \bar{b}.$$

Damit gilt

$$a \vee b = \overline{\bar{a} \wedge \bar{b}}.$$

Wir können das Oder also durch NAND und zwei Negationen ausdrücken. Das geschieht in Abbildung B.1.

**Aufgabe 3.6:** Wir müssen uns dazu zwei Dinge überlegen: das Fredkin-Gatter führt eine umkehrbare Berechnung aus und ist universal, kann also zum Beispiel AND und NOT berechnen.

Man nennt das Fredkin-Gatter auch *controlled swap*, da die Belegung von  $c$  bestimmt, ob die Werte von  $a$  und  $b$  getauscht werden.

**Aufgabe 3.7:** Da es sich um verschränkte Zustände handelt, betrachten wir das gesamte Register:

$$\begin{aligned} & |a\rangle |b\rangle |10101100\rangle \\ &= \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) |10101100\rangle \\ &= 1/2 (|0010101100\rangle + |0110101100\rangle + |1010101100\rangle + |1110101100\rangle) \\ &\rightarrow 1/2 (|0010101100\rangle + |0110101100\rangle + |1011101100\rangle + |1111101100\rangle) \\ &\rightarrow 1/2 (|0010101100\rangle + |0110111100\rangle + |1011101100\rangle + |1111111100\rangle) \end{aligned}$$

Das ist der Zustand nach den beiden ersten Toffoli-Gattern. Führen wir die Berechnung fort, sehen wir, dass die Ausgabebits korrekt belegt sind. Nennen wir die beiden letzten Bits  $o_1$  und  $o_2$ , ist der Endzustand

$$|a\rangle|b\rangle|o_1\rangle|o_2\rangle = 1/2(|0000\rangle + |0110\rangle + |1010\rangle + |1101\rangle),$$

gemäß der Regel:  $o_1 = a \oplus b, o_2 = a \wedge b$ .

**Aufgabe 3.8:** Das Register aus vier Quantenbits befindet sich im Ausgangszustand

$$|1\rangle|a\rangle|b\rangle|0\rangle = 1/2(|1000\rangle + |1010\rangle + |1100\rangle + |1110\rangle).$$

Die beiden Toffoli-Gatter überführen es in

$$1/2(|1000\rangle + |1010\rangle + |1110\rangle + |1101\rangle).$$

**Aufgabe 3.9:**  $|\phi_3\rangle$  und  $|\psi_3\rangle$  lassen sich mit einer Messung bezüglich der Hadamard-Basis

$$\left\{ \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right\}$$

unterscheiden. Gleichwertig ist,  $H|\phi_3\rangle$  und  $H|\psi_3\rangle$  bezüglich der Standardbasis zu messen.

**Aufgabe 4.1:** Wir müssen zeigen: gilt für eine Eingabe  $G$   $HC(G) = 1$ , so lässt sich das mit Hilfe eines Zertifikats effizient verifizieren. Das Zertifikat ist der Hamilton-Kreis in dem Graphen  $G$ . Wir verifizieren, dass jeder Knoten genau einmal besucht wird, indem wir die Kanten durchlaufen. Kapitel 4

**Aufgabe 6.2:** Im Fall  $N = 2$  können wir direkt nachrechnen, dass  $-HR_2H = D_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$  gilt. Für den allgemeinen Fall bietet es sich an, Multiplikation von Blockmatrizen zu verwenden und die Aussage mit vollständiger Induktion zu zeigen. Hier folgen wichtige Zwischenergebnisse: Kapitel 6

Wegen  $H_{n+1} = H \otimes H_n, N = 2^n$  gilt

$$-H_{n+1}R_{2N}H_{n+1} = \frac{1}{\sqrt{2}} \begin{pmatrix} -H_n & -H_n \\ -H_n & H_n \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} R_N H_n & R_N H_n \\ H_n & -H_n \end{pmatrix}.$$

Dabei wurde verwendet, dass  $R_N$  in einer rechts multiplizierten Matrix die erste Zeile negiert und sonst nichts verändert. Ausmultiplizieren ergibt nun

$$\frac{1}{2} \begin{pmatrix} -H_n R_N H_n - I_N & -H_n R_N H_n + I_N \\ -H_n R_N H_n + I_N & -H_n R_N H_n - I_N \end{pmatrix} = \frac{1}{2} \begin{pmatrix} D_N - I_N & D_N + I_N \\ D_N + I_N & D_N - I_N \end{pmatrix} = D_{2N}.$$

**Aufgabe 6.4:** Wir können die Aussage formal beweisen, wenn wir die vier Basiszustände  $|0\rangle|0\rangle, |0\rangle|1\rangle, |1\rangle|0\rangle, |1\rangle|1\rangle$  betrachten:



$$|0\rangle|0\rangle \xrightarrow{X\otimes X} |1\rangle|1\rangle \xrightarrow{I_2\otimes H} |1\rangle\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \xrightarrow{\text{CNOT}} |1\rangle - \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \xrightarrow{I_2\otimes H} -|1\rangle|1\rangle \xrightarrow{X\otimes X} -|0\rangle|0\rangle.$$

Genauso sieht man, dass der Schaltkreis die Zustände  $|0\rangle|1\rangle, |1\rangle|0\rangle$  und  $|1\rangle|1\rangle$  unverändert lässt.

**Aufgabe 6.6:** Es ist die Operation  $V_f$ ! Sie invertiert die Amplitude jedes von  $f$  auf 1 abgebildeten Elements - rechnen Sie das nach! Nun betrachten wir die Ebene, die vom Startzustand  $|s\rangle$  und von  $|\hat{x}_d\rangle$  aufgespannt wird. In dieser Ebene entspricht das Kippen der Komponenten aller gesuchten Elemente der Spiegelung an der Achse längs  $|\tilde{x}^\perp\rangle$ . Das kann man sich durch eine Skizze veranschaulichen.

**Aufgabe 6.8:** Wir führen den Algorithmus  $k$ -mal aus. Angenommen, wir haben die Elemente  $\hat{x}_1, \dots, \hat{x}_l$  schon gefunden. Dann Kippen wir nach der Anwendung von  $V_f$  die Amplituden von  $\hat{x}_1, \dots, \hat{x}_l$  wieder zurück. Das lässt sich wie auf Seite 143 realisieren. Wir führen die Groveriteration  $G(N, k-l)$ -mal aus und erhalten eine Superposition über  $\hat{x}_{l+1}, \dots, \hat{x}_k$ .

Wenn  $k$  groß ist (Größenordnung  $\Omega(\sqrt{N})$ ), bietet es sich an, für jedes Element  $x$  den Funktionswert zu bestimmen.

## Kapitel 7

**Aufgabe 7.3:** Wir nehmen an, dass  $a' = 0$  und  $e' = 1$  gilt: Messen wir ein Bit im Zustand  $|0\rangle$  oder  $|1\rangle$  in der falschen Basis  $B'$ , befindet sich  $|x\rangle$  anschließend im Zustand  $|+\rangle$  oder  $|-\rangle$ . Misst Bob nun in  $B$ , erhält er nur in der Hälfte der Fälle das korrekte Ergebnis.

Der Fall  $a' = 1$  und  $e' = 0$  ergibt sich analog.

**Aufgabe 7.5:** An der Argumentation ist bis zu dem letzten Satz nichts auszusetzen. Jedoch bietet sich für Eve folgende Möglichkeit an: Alice hat ihr Soll erfüllt, die Quantenbits verschickt und verwahrt  $a_1, \dots, a_m$  in einem Panzerschrank. Diesen kann Eve knacken. Nun wartet sie, bis Bob misst, belauscht den Austausch der Basen und erfährt, welche der von ihr gestohlenen Bits den Schlüssel bilden. Kann sie den Panzerschrank wieder verschließen, bleibt ihr Tun unbemerkt.

Wichtig ist: Theoretisch kann ihr Eindringen in den Panzerschrank unbemerkt bleiben. Ihr Hantieren am Quantenkanal wird aus theoretischen Gründen bemerkt, wenn sie wesentliche Teile des Schlüssels erlauschen will.

## Kapitel 8

**Aufgabe 8.1:** Wäre  $a^{p/2} - 1$  ein Vielfaches von  $n$ , hieße das  $a^{p/2} - 1 \equiv 0 \pmod n$  und also  $a^{p/2} \equiv 1 \pmod n$ . Das ist aber ausgeschlossen, weil in diesem Fall die Periode kleiner gleich  $p/2$  wäre. Diese ist jedoch gleich  $p$ .

**Aufgabe 8.2:** Wenn eine Zahl  $m$  sowohl  $x$  als auch  $n$  teilt, dann teilt sie für jedes  $k$  die Differenz  $x - kn$ . Darauf beruht der Euklidische Algorithmus.

**Aufgabe 8.6:** Der formale Beweis lautet wie folgt:

$$\begin{aligned}\sum_{i=0}^{N-1} (\omega_N^k)^i &= \frac{(\omega_N^k)^N - 1}{\omega_N^k - 1} \\ &= \frac{(\omega_N^N)^k - 1}{\omega_N^k - 1} = 0\end{aligned}$$

Ist  $k$  ein Vielfaches von  $N$ , gilt  $\omega_N^k = 1$  und die Summe ist  $N$ .

**Aufgabe 8.8:**  $R_4 \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$ . Die komplexe Zahl  $-1 = e^{\pi i}$  wurde um den Winkel  $\pi/2$  gedreht: auf  $e^{3/2\pi i} = -i$ .

**Aufgabe 8.10:** Wir betrachten als Beispiel Bit  $y_0$ .  $x_2$  wird von  $H$  in  $|\phi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \omega_2^{x_2}|1\rangle)$  überführt.  $R_4$  multipliziert den  $|1\rangle$ -Anteil in  $|\phi_1\rangle$  mit  $\omega_4$ , wenn  $x_1 = 1$ . Das hat  $|\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \omega_2^{x_2}\omega_4^{x_1}|1\rangle)$  zur Folge. Genauso führt  $R_8$  zu  $|\phi_3\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \omega_2^{x_2}\omega_4^{x_1}\omega_8^{x_0}|1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + \omega_8^x|1\rangle)$ , was das für  $y_0$  gewünschte Ergebnis ist.

**Aufgabe 8.11:** Wenden wir  $\text{QFT}_{16}$  auf  $1/2(|1\rangle + |5\rangle + |9\rangle + |13\rangle)$  an, ergibt sich

$$\begin{aligned}& \frac{1}{8} \left( \sum_{y=0}^{15} (\omega_{16}^y |y\rangle + \omega_{16}^{5y} |y\rangle + \omega_{16}^{9y} |y\rangle + \omega_{16}^{13y} |y\rangle) \right) \\ &= \frac{1}{8} \left( \sum_{y=0}^{15} \omega_{16}^y (\omega_{16}^0 + \omega_{16}^{4y} + \omega_{16}^{8y} + \omega_{16}^{12y}) |y\rangle \right) \\ &= \frac{1}{8} \left( \sum_{y=0}^{15} \omega_4^y (1 + \omega_4^y + \omega_4^{2y} + \omega_4^{3y}) |y\rangle \right).\end{aligned}$$

Wenn nun  $y$  ein Vielfaches von 4 ist, gilt

$$1 + \omega_4^y + \omega_4^{2y} + \omega_4^{3y} = 1 + 1 + 1 + 1.$$

Falls  $y$  kein Vielfaches von vier ist, addiert sich die Amplitude zu 0. Für  $y = 1$  ergibt sich

$$1 + \omega_4^1 + \omega_4^2 + \omega_4^3 = 1 + i - 1 - i$$

(für  $y = 5, 9, 13$  ist das Ergebnis wegen  $\omega_4^4 = 1$  identisch). Für  $y = 2$  (und  $y = 6, 10, 14$ ) ergibt sich

$$1 + \omega_4^2 + \omega_4^4 + \omega_4^6 = 1 + \omega_4^2 + 1 + \omega_4^2 = 1 - 1 + 1 - 1.$$

Für  $y = 3$  (und  $y = 7, 11, 15$ ) ergibt sich

$$1 + \omega_4^3 + \omega_4^6 + \omega_4^9 = 1 + \omega_4^3 + 1 + \omega_4^2 + \omega_4 = 1 - i - 1 + i.$$

Wir folgern

$$\begin{aligned}\text{QFT}_{16}(|1\rangle + |5\rangle + |9\rangle + |13\rangle) &= (|0\rangle + \omega_4|4\rangle + \omega_4^2|8\rangle + \omega_4^3|12\rangle) \\ &= (|0\rangle + i|4\rangle - |8\rangle - i|12\rangle)\end{aligned}$$

und die Aufgabe ist gelöst. Wir können weiter erkennen, dass allgemein für  $j = 0, \dots, 3$  gilt:

$$\begin{aligned}\text{QFT}_{16} \frac{1}{2}(|j\rangle + |j+4\rangle + |j+8\rangle + |j+12\rangle) \\ = \frac{1}{8} \left( \sum_{y=0}^{15} \omega_4^{jy} (1 + \omega_4^y + \omega_4^{2y} + \omega_4^{3y}) |y\rangle \right).\end{aligned}$$

## Kapitel 9

**Aufgabe 9.1:** Durch Anwendung auf den allgemeinen Qubit-Zustand  $\alpha|0\rangle + \beta|1\rangle$  lässt sich erkennen, dass  $X \cdot Z = -Z \cdot X$  gilt.

**Aufgabe 9.3:** Zunächst wird wieder das Codewort  $\alpha|000\rangle + \beta|111\rangle$  erzeugt. Wir betrachten beispielhaft den Fall, dass im Fehlergatter Bit  $b_1$  gekippt wird ( $F = X \otimes I \otimes I$ ).

$$\begin{aligned}\alpha|000\rangle + \beta|111\rangle &\xrightarrow{F} \alpha|100\rangle + \beta|011\rangle \\ &\xrightarrow{1,2} \alpha|111\rangle + \beta|011\rangle \\ &\xrightarrow{3} \alpha|011\rangle + \beta|111\rangle = (\alpha|0\rangle + \beta|1\rangle)|11\rangle.\end{aligned}$$

Auch für die anderen Fälle ergibt sich, dass wir einen unverschränkten Zustand mit wiederhergestelltem Bit  $b_1$  erhalten.

**Aufgabe 9.4:** Die Lösung des ersten Teils lässt sich in folgender Tabelle darstellen:

	$X$	$Z$
$ 0\rangle$	$ 1\rangle$	$ 0\rangle$
$ 1\rangle$	$ 0\rangle$	$- 1\rangle$
$ +\rangle$	$ +\rangle$	$ -\rangle$
$ -\rangle$	$- -\rangle$	$ +\rangle$

Somit verhält sich  $X$  auf  $\{|0\rangle, |1\rangle\}$  wie  $Z$  auf  $\{|+\rangle, |-\rangle\}$  und umgekehrt.

**Aufgabe 9.5:** a) In der Hadamardbasis verhält sich der Phaseflip exakt wie der Bitflip in der klassischen Basis. Also sollten wir als Codewort

$$\alpha \cdot |+\rangle|+\rangle|+\rangle + \beta \cdot |-\rangle|-\rangle|-\rangle$$

verwenden. Ein Phaseflip auf z.B. dem ersten Bit hätte dann das Ergebnis  $\alpha \cdot |-\rangle|+\rangle|+\rangle + \beta \cdot |+\rangle|-\rangle|-\rangle$ . Nach Anwendung der Hadamard-Transformation können wir die Bitflip-Korrektur anwenden.

b) Das ergibt den Schaltkreis in Abbildung B.2:

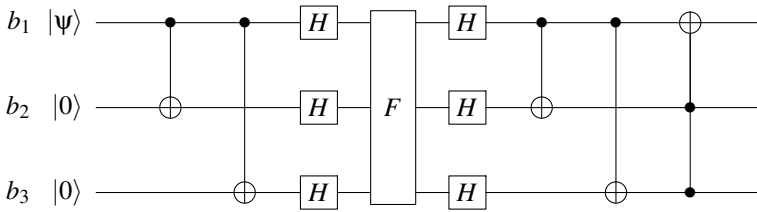


Abbildung B.2: Phaseflip-Code.

**Aufgabe 9.6:** a) Auch wenn kein Fehler auftritt müssen wir sicherstellen, dass korrekt dekodiert wird. Als Vorbereitung betrachten wir zunächst die ersten drei Bits isoliert (wir sollten aber im Hinterkopf behalten, dass alle neun Bits des Codeworts verschränkt sind).

$$\begin{aligned}
 \alpha \cdot |c^+\rangle + \beta \cdot |c^-\rangle &= \alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \\
 &\xrightarrow{7-9} \alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle + |100\rangle) + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle - |100\rangle) \\
 &= \alpha \cdot |+\rangle|00\rangle + \beta \cdot |-\rangle|00\rangle \\
 &\xrightarrow{10} \alpha \cdot |000\rangle + \beta \cdot |100\rangle
 \end{aligned}$$

Nun fällt es leichter, den Überblick über die Dekodierung des gesamten ungestörten Codeworts zu behalten:

$$\begin{aligned}
 &\alpha \cdot |c^+c^+c^+\rangle + \beta \cdot |c^-c^-c^-\rangle \\
 &\xrightarrow{7-10} \alpha \cdot |000\rangle|000\rangle|000\rangle + \beta \cdot |100\rangle|100\rangle|100\rangle \\
 &\xrightarrow{11-13} \alpha \cdot |000\rangle|000\rangle|000\rangle + \beta \cdot |100\rangle|000\rangle|000\rangle \\
 &= (\alpha|0\rangle + \beta|1\rangle)|0\dots 0\rangle
 \end{aligned}$$

d) Wir ermitteln zunächst ausführlich die Wirkung des Fehlers  $XZ$  auf das erste Bit des Codeworts.

$$\begin{aligned}
 &\alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)|c^+\rangle|c^+\rangle + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle)|c^-\rangle|c^-\rangle \\
 &\xrightarrow{Z} \alpha \cdot \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle)|c^+\rangle|c^+\rangle + \beta \cdot \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)|c^-\rangle|c^-\rangle \\
 &\xrightarrow{X} \alpha \cdot \frac{1}{\sqrt{2}}(|100\rangle - |011\rangle)|c^+\rangle|c^+\rangle + \beta \cdot \frac{1}{\sqrt{2}}(|100\rangle + |011\rangle)|c^-\rangle|c^-\rangle
 \end{aligned}$$

Gatter 7-9 führen zu:

$$\begin{aligned} & \alpha \cdot \frac{1}{\sqrt{2}}(|011\rangle - |111\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle + |100\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle + |100\rangle) \\ & + \beta \cdot \frac{1}{\sqrt{2}}(|011\rangle + |111\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle - |100\rangle) \cdot \frac{1}{\sqrt{2}}(|000\rangle - |100\rangle) \\ = & \alpha \cdot |-\rangle|11\rangle|+\rangle|00\rangle|+\rangle|00\rangle + \beta \cdot |+\rangle|11\rangle|-\rangle|00\rangle|-\rangle|00\rangle. \end{aligned}$$

Beachten Sie, dass nur noch die Bits 1,4 und 7 verschränkt sind. Jetzt führt Hadamard in Schritt 10 zu:

$$\begin{aligned} & \alpha|111\rangle|000\rangle|000\rangle + \beta|011\rangle|100\rangle|100\rangle \\ \xrightarrow{11,12} & \alpha|111\rangle|100\rangle|100\rangle + \beta|011\rangle|100\rangle|100\rangle \\ \xrightarrow{13} & \alpha|011\rangle|100\rangle|100\rangle + \beta|111\rangle|100\rangle|100\rangle \\ = & (\alpha|0\rangle + \beta|1\rangle) \cdot |11\rangle|100\rangle|100\rangle. \end{aligned}$$

## Kapitel 10

**Aufgabe 10.1:** Das stünde im Widerspruch zur Quantenmechanik, die nur Messungen bezüglich Basen erlaubt. Besäße Eve einen solchen Kristall, könnte sie mit vier Detektoren die Richtungen bestimmen. Das ist aber nach der quantenmechanischen Beschreibung der Welt (die bisher durch kein Experiment widerlegt werden konnte) ausgeschlossen.

## Kapitel 11

**Aufgabe 11.1:** Für  $A = \{2, 3\}$  und  $B = \{1, 4, 5\}$  gibt es fünf Kanten zwischen den Mengen und ebenso für den Schnitt  $A' = \{1, 4\}$  und  $B' = \{2, 3, 5\}$ .

**Aufgabe 11.2:** Maximiere  $(x_1 + x_2 - 2x_1x_2) + (x_1 + x_3 - 2x_1x_3) + (x_2 + x_4 - 2x_2x_4) + (x_3 + x_4 - 2x_3x_4) + (x_3 + x_5 - 2x_3x_5) + (x_4 + x_5 - 2x_4x_5)$  über  $\{0, 1\}^5$ . Ein Maximum ergibt sich für  $x_2 = x_3 = 1$  und  $x_1 = x_4 = x_5 = 0$  oder  $(0, 1, 1, 0, 0)^T$ . All jene Summanden liefern den Beitrag 1, in denen  $x_2$  oder  $x_3$  mit einer der übrigen Variablen kombiniert sind. Die weiteren Maxima sind  $(0, 1, 1, 0, 1)^T$ ,  $(1, 0, 0, 1, 1)^T$  und  $(1, 0, 0, 1, 0)^T$ . Insgesamt erhalten wir die beiden Aufteilungen aus Aufgabe 11.1 und die Darstellungen, die aus dem Tausch von  $A$  und  $B$  entstehen. Die zu maximierende Formel lässt sich vereinfachen:

$$2x_1 + 2x_2 + 3x_3 + 3x_4 + 2x_5 - 2x_1x_2 - 2x_1x_3 - 2x_2x_4 - 2x_3x_4 - 2x_3x_5 - 2x_4x_5.$$

**Aufgabe 11.3:** Zwei Versionen von  $Q$ : eine obere Dreiecksmatrix  $Q_d$  und die symmetrische Alternative  $Q_s$ .

$$Q_d = \begin{pmatrix} 2 & -2 & -2 & 0 & 0 \\ 0 & 2 & 0 & -2 & 0 \\ 0 & 0 & 3 & -2 & -2 \\ 0 & 0 & 0 & 3 & -2 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix} \quad \text{und} \quad Q_s = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 \\ -1 & 0 & 3 & -1 & -1 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{pmatrix}$$

**Aufgabe 11.4:**  $x_i + x_j - 2x_ix_j \mapsto \frac{s_i+1}{2} + \frac{s_j+1}{2} - 2\frac{s_i+1}{2}\frac{s_j+1}{2} = \frac{1}{2}(1 - s_is_j)$ . Für  $s_i = s_j$  erhalten wir 0. Für  $s_i \neq s_j$  hingegen  $\frac{1}{2}(1 - (-1)) = 1$ . Wir sollten

negieren, da wir am Maximum interessiert sind.  $\frac{1}{2}(s_i s_j - 1)$  liefert nun  $-1$  für jede den Schnitt kreuzende Kante und  $0$  sonst.

Eine vereinfachte Version wäre  $s_i s_j$ : Eine kreuzende Kante liefert  $-1$ , jede andere den Wert  $1$ .

**Aufgabe 11.5:** Wir verwenden den gleichen Ansatz wie bei MAX-CUT:  $s_i = +1$  bedeutet  $n_i \in A$  und  $s_i = -1$ , dass  $n_i \in B$  liegt.

Damit ist  $\sum_{i=1}^N n_i s_i$  genau dann gleich  $0$ , wenn die Partition wie gewünscht zu  $\sum_{n \in A} n - \sum_{n \in B} n = 0$  führt. Für andere Partitionen ist  $\sum_{i=1}^N n_i s_i$  kleiner oder größer  $0$ . Ein Minimierungsproblem erhalten wir durch Quadrieren:  $\min (\sum_{i=1}^N n_i s_i)^2 = 0$  genau dann, wenn  $\sum_{n \in A} n = \sum_{n \in B} n$ .

Weiter gilt:  $(\sum_{i=1}^N n_i s_i)^2 = \sum_{i=1}^N n_i^2 s_i^2 + \sum_{i < j \leq N} 2 \cdot n_i n_j s_i s_j$ .

Hinweis zur Optimierungsvariante: Gibt es keine perfekte Zerlegung, so wird mit dem gleichen Ansatz die Differenz  $|\sum_{n \in A} n - \sum_{n \in B} n|$  minimiert.

Ein QUBO-Problem erhält man daraus über  $s_i \mapsto 2x_i - 1$  oder über folgenden direkten Ansatz; wir legen fest:  $x_i = 1$ , falls  $n_i \in A$  und  $x_i = 0$ , falls  $n_i \in B$ . Dann ist  $\sum_{n \in A} n = \sum_{i=1}^N n_i x_i$  und weiter  $\sum_{n \in B} n = \sum_{i=1}^N n_i - \sum_{i=1}^N n_i x_i$ . Also ist  $(2 \cdot \sum_{i=1}^N n_i x_i - \sum_{i=1}^N n_i)^2$  zu minimieren. Zu weiteren Umformungsschritten siehe [68].

**Aufgabe A.1:** Zum ersten Teil der Aufgabe: Ziehen wir die natürliche Zahl  $2$  von der natürlichen Zahl  $1$  ab, ist das Ergebnis keine natürliche Zahl. Ebenso erhalten wir bei der Division zweier ganzen Zahlen in der Regel keine ganze Zahl. Anhang

**Aufgabe A.7:** Hier soll folgender Hinweis genügen: Beschränken wir uns auf den  $\mathbb{R}^2$ , entspricht die Definition der Norm gerade dem Satz von Pythagoras.

**Aufgabe A.8:** Verwenden Sie die Aussage über den Winkel, Gleichung A.6.

# Literaturverzeichnis

- [1] Aaronson. *Quantum computing since Democritus*. Cambridge University Press, 2013.
- [2] Adami and Cerf. Quantum computation with linear optics. 1998. arXiv:quant-ph/9806048.
- [3] Agrawal, Kayal, and Saxena. PRIMES in P. *Annals of Mathematics*, 160 (2), 2004.
- [4] Aharonov et al. Interactive proofs for quantum computations. *Innovations in Computer Science*, 2010.
- [5] Aharonov and other. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Journal of Computing*, 37, 2007. arXiv:quant-ph/0405098v2.
- [6] Ambainis. Polynomial degree vs. quantum query complexity. In *Proceedings FOCS 03*, pages 230–239, 2003. quant-ph/0305028.
- [7] Andersen and others. Repeated quantum error detection in a surface code. *Nat. Phys*, 16, 2020.
- [8] Arute et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574, 2019.
- [9] Aspect, Dalibard, and Roger. Experimental test of Bell's inequalities using time-varying analyzers. *Physical Review Letters*, 49, 1982.
- [10] Asproni et al. Accuracy and minor embedding in subqubo decomposition with fully connected large problems: a case study about the number partitioning problem. *Quantum Mach. Intell.*, 2020.
- [11] Barz et al. Demonstration of blind quantum computing. *Science*, 335, 2012.
- [12] Beals, Buhrman, Cleve, Mosca, and de Wolf. Quantum lower bounds by polynomials. In *39th FOCS*, pages 352–361, 1998.
- [13] Bell. On the Einstein-Podolski-Rosen paradox. *Physics*, pages 195–200, 1964.
- [14] Ben-Or et al. Quantum multiprover interactive proofs with communicating provers. *SIAM J. Comput.*, 43, 2014.
- [15] Benioff. Quantum mechanical Hamiltonian models of discrete processes that erase their own histories: application to Turing machines. *Int. Journal of Theoretical Physics*, 21:177–202, 1982.
- [16] Bennett. Logical reversibility of computation. *IBM Journal of Research and Dev.*, 17:525–532, 1973.
- [17] Bennett, Bernstein, Brassard, and Vazirani. Strengths and weaknesses of quantum computation. *SIAM Journal of Computing*, 26:5:1510–1523, 1997.
- [18] Bennett, Bessette, Brassard, Salvail, and Smolin. Experimental quantum cryptography. *Journal of Cryptology*, 5(1):3–28, 1992.

- [19] Bennett, Brassard, Crépeau, Jozsa, Peres, and Wootters. Teleporting an unknown quantum state via dual classical and EPR channels. *Phys. Rev. Letters*, 70:1895–1899, 1993.
- [20] Bennett, Brassard, and Robert. Privacy amplification by public discussion. *SIAM Journal of Computing*, 17(2):210–229, 1988.
- [21] Bennett et al. Purification of noisy entanglement and faithful teleportation via noisy channels. *Phys. Rev. Lett.*, 78, 1997.
- [22] Bennett and Wiesner. Communication via one- and two-particle operators on Einstein-Podolski-Rosen channels. *Phys. Rev. Letters*, 69:2881–2884, 1992.
- [23] Born, Jordan, and Heisenberg. Zur Quantenmechanik II. *Zeitschrift für Physik*, 35, 1926.
- [24] Bouwmeester, Ekert, and Zeilinger. *The Physics of Quantum Information*. Springer, 2001.
- [25] Bouwmeester, Pan, Mattle, Eibl, Weinfurter, and Zeilinger. Experimental quantum teleportation. *Nature*, 390(6660):575–579, 1997.
- [26] Boyer, Brassard, Høyer, and Tapp. Tight bounds on quantum searching. *Fourth Workshop on Physics and Computation*, pages 36–43, 1996.
- [27] Brands. *Einführung in die Quanteninformatik*. Springer, 2011.
- [28] Brassard, Høyer, and Tapp. Quantum counting. In *Proceedings, 25th ICALP*, volume 1443 of *Lecture Notes in Computer Science*, pages 820–831. Springer Verlag, 1998. quant-ph/9805082.
- [29] Briegel et al. Quantum repeaters: The role of imperfect local operations in quantum communication. *Phys. Rev. Lett.*, 81, 1998.
- [30] Briegel et al. Quantum repeaters based on entanglement purification. *Phys. Rev. A*, 59, 1999.
- [31] Broadbent, Fitzsimons, and Kashefi. Universal blind quantum computation. *50th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2009)*, 2009. arXiv:quant-ph/0807.4154v3.
- [32] Bruzewicz et al. Trapped-ion quantum computing: Progress and challenges featured. *Applied Physics Reviews*, 6, 2019.
- [33] Buchmann. *Einführung in die Kryptographie*. Springer, 2016.
- [34] Von Burg et al. Quantum computing enhanced computational catalysis. *Phys. Rev. Research*, 3, 2021.
- [35] Bylander et al. Noise spectroscopy through dynamical decoupling with a superconducting flux qubit. *Nature Physics*, 7, 2011.
- [36] Calderbank and Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54:1098–1106, 1996.
- [37] Chen and others. An integrated space-to-ground quantum communication network over 4,600 kilometres. *Nature*, 589, 2021.
- [38] Cirac and Zoller. Quantum computations with cold trapped ions. *Physical Review Letters*, 74:4091–4094, 1995.
- [39] Clay Mathematics Institute (CMI). [www.claymath.org](http://www.claymath.org).
- [40] Cormen, Leiserson, Rivest, and Stein. *Algorithmen - Eine Einführung*. De Gruyter Oldenbourg, 2013.



- [41] Danos et al. The measurement calculus. 2004. arXiv:quant-ph/0412135.
- [42] D'Ariano et al. *Quantum Theory from First Principles: An Informational Approach*. Cambridge University Press, 2017.
- [43] Denchev et al. What is the computational value of finite-range tunneling? *Phys. Rev. X*, 6, 2016.
- [44] Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society*, 400:97–117, 1985.
- [45] Deutsch. Quantum computational networks. *Proc. of the Royal Society, A* 425:73–90, 1989.
- [46] Deutsch and Jozsa. Rapid solution of problems by quantum computations. *Proceedings of the Royal Society*, 439:553–558, 1992.
- [47] Devitt et al. Quantum error correction for beginners. *Physical Review Letters*, 76, 2013. quant-ph/0905.2794.
- [48] Djordjevic. *Quantum Information Processing and Quantum Error Correction: An Engineering Approach*. Elsevier, 2012.
- [49] Dürr, Heiligman, Høyer, and Mhalla. Quantum query complexity of some graph problems. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 481–493, 2004.
- [50] Dürr and Høyer. A quantum algorithm for finding the minimum. 1996. quant-ph/9607014.
- [51] Einstein. Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt. *Annalen der Physik*, 17:132–148, 1905.
- [52] Einstein, Podolsky, and Rosen. Can quantum-mechanical description of physical reality be considered complete? *Physical Review*, 47:777–780, 1935.
- [53] Ekert. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.*, 67(6):661–663, 1991.
- [54] Ekert, Huttner, Palma, and Peres. Eavesdropping on quantum-cryptographical systems. *Phys. Rev. A*, 50(2):1047–1056, 1994.
- [55] Ekert and Mosca. The hidden subgroup problem and eigenvalue estimation on a quantum computer. *Lecture Notes in Computer Science*, 1509:174–188, 1998.
- [56] Embacher. Grundidee der Dekohärenz – Wieso sieht die Welt so klassisch aus? <https://homepage.univie.ac.at/franz.embacher/Quantentheorie/Dekohaerenz/>.
- [57] Farhi et al. Quantum computation by adiabatic evolution. 2000. arXiv:quant-ph/0001106.
- [58] Farhi et al. A quantum approximate optimization algorithm. 2014. arXiv:1411.4028.
- [59] Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488, 1982.
- [60] Fowler et al. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86, 032324, 2012.
- [61] Fredkin and Toffoli. Conservative logic. *Int. Journal Theoretical Physics*, 21:219–253, 1982.

- [62] Freedman and Clauser. Experimental test of local hidden-variable theories. *Physical Review Letters*, 28, 1972.
- [63] Friebe et al. *Philosophie der Quantenphysik*. Springer, 2018.
- [64] Fürnkranz. *Vision Quanteninternet*. Springer, 2019.
- [65] Gambetta et al. Building logical qubits in a superconducting quantum computing system. *npj Quantum Information*, 86, 032324, 2017.
- [66] Garey and Johnson. *Computers and Intractability*. Freeman, 1979.
- [67] Gisin, Ribordy, Tittel, and Zbinden. Quantum cryptography. *Reviews of modern Physics*, 74:145–195, 2002.
- [68] Glover et al. Quantum bridge analytics I: A tutorial on formulating and using QUBO models. *4OR*, 17:335–371, 2019.
- [69] Grabbe. An introduction to quantum game theory. *Science*, 2005. arXiv:quant-ph/0506219.
- [70] Grasselli. *Quantum Cryptography*. Springer, 2021.
- [71] Grover. A fast quantum-mechanical algorithm for database search. *28th STOC*, pages 212–219, 1996. quant-ph/9605043.
- [72] Grover. How fast can a quantum computer search? 1998. quant-ph/9809029.
- [73] Hallgren. Polynomial time quantum algorithms or Pell's equation and the principal ideal problem. *Symp. on the theory of computation STOC*, 2002.
- [74] Handsteiner et al. Cosmic bell test: Measurement settings from milky way stars. *Phys. Rev. Lett.*, 118, 2017.
- [75] Hanson et al. Entanglement distillation between solid-state quantum network nodes. *Science*, 356, 2017.
- [76] Hensen et al. Loophole-free bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526, 2015.
- [77] Hirvensalo. *Quantum computing*. Springer Verlag, 2010.
- [78] Hughes et al. *Quantum Computing for the Quantum Curious*. Springer, 2021.
- [79] Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 2007.
- [80] Ji et al. MIP\*=RE. 2020.
- [81] Johnston et al. *Programming Quantum Computers*. O Reilly, 2019.
- [82] Jordan. <https://quantumalgorithmzoo.org>.
- [83] Jozsa. Quantum algorithms and the Fourier transform. *Proceedings of the Royal Society*, 439:323–337, 1998.
- [84] Jozsa. Searching in Grover's algorithm. 1999. quant-ph/9901021.
- [85] Jozsa. Notes on Hallgren's efficient quantum algorithm for solving Pell's equation. *Proceedings of the Royal Society*, 2003. arXiv:quant-ph/0302134.
- [86] Jozsa. An introduction to measurement based quantum computation. 2005. arXiv:quant-ph/0508124v2.
- [87] Kadowaki and Nishimori. Quantum annealing in the transverse Ising model. *Phys. Rev. E*, 58, 1998.
- [88] Kasirajan. *Fundamentals of Quantum Computing*. Springer, 2021.

- [89] Kaye, Laflamme, and Mosca. *An introduction to Quantum Computing*. Oxford University Press, 2007.
- [90] Kelly et al. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519, 2015.
- [91] Google Quantum AI (Kelly and others). Exponential suppression of bit or phase errors with cyclic error correction. *Nature*, 595, 2021.
- [92] Kirkpatrick et al. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
- [93] Kitaev. Quantum measurements and Abelian stabilizer problem. 1995. arXiv:quant-ph/9511026.
- [94] Kjaergaard et al. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11(1):369–395, 2020.
- [95] Kondacs and Watrous. On the power of quantum finite state automata. In *Proceedings, 36th IEEE FOCS*, pages 66–75, 1997.
- [96] Krengel. *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Vieweg Verlag, 2007.
- [97] Laflamme et al. Perfect quantum error correction code. *Physical Review Letters*, 77:198, 1996.
- [98] Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191, 1961.
- [99] Landsburg. Quantum game theory. *Wiley Encyclopedia of Operations Research and Management Science*, 2011. arXiv:quant-ph/0506219.
- [100] LaPierre. *Introduction to Quantum Computing*. Springer, 2021.
- [101] Lenze. *Mathematik und Quantum Computing*. Logos, 2020.
- [102] Lidar and Brun. *Quantum Error Correction*. Cambridge University Press, 2013.
- [103] Lipton and Regan. *Quantum Algorithms via Linear Algebra: A Primer*. MIT Press, 2014.
- [104] Lucas. Ising formulations of many NP problems. *Front. Phys.*, 2:79, 2014.
- [105] Lydersen et al. Hacking commercial quantum cryptography systems by tailored bright illumination. *Nature Photonics*, 4:686 – 689, 2010.
- [106] Manz. *Fehlerkorrigierende Codes*. Springer-Vieweg, 2017.
- [107] Mayers. Unconditional security in quantum cryptography. 1998. arXiv:quant-ph/9802025.
- [108] McGeoch. Theory versus practice in annealing-based quantum computing. *Theoretical Computer Science*, 816, 2020.
- [109] McGeoch et al. A cross-disciplinary introduction to quantum annealing-based algorithms. *Contemporary Physics*, 59(2):174–197, 2018.
- [110] McGeoch and Wang. Experimental evaluation of an adiabatic quantum system for combinatorial optimization. In *Proceedings of the ACM International Conference on Computing Frontiers*, 2013.
- [111] Miller. Riemann’s hypothesis and tests for primality. *J. Comput. System Sci.*, 13, 1976.

- [112] Mitzenmacher and Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithm and Data Analysis*. Cambridge University Press, 2017.
- [113] Neukart et al. Traffic flow optimization using a quantum annealer. *Frontiers in ICT*, 4, 2017.
- [114] Nielsen and Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2000.
- [115] Petri. Grundsätzliches zur Beschreibung diskreter Prozesse. In *Proceedings, 3. Kolloquium über Automatentheorie (Hannover)*, pages 121–140. Birkhäuser, 1967.
- [116] Piccinini. The physical Church–Turing thesis: Modest or bold? *The British Journal for the Philosophy of Science*, 62:733–769, 10 2011.
- [117] Planck. Zur Theorie des Gesetzes der Energieverteilung im Normalspektrum. *Verh. Deutsche physikalische Gesellschaft*, 237, 1900.
- [118] Pompili and others. Realization of a multinode quantum network of remote solid-state qubits. *Science*, 372, 2021.
- [119] Portugal. *Quantum Walks and Search Algorithms*. Springer, 2018.
- [120] Preskill. *Lecture Notes for Physics 229: Quantum Information and Computation*. California Institute of Technology, 1998. [www.theory.caltech.edu/people/preskill/ph229/](http://www.theory.caltech.edu/people/preskill/ph229/).
- [121] Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.
- [122] Prevedel et al. Experimental realization of a quantum game on a one-way quantum computer. *New Journal of Physics*, 9:205, 2007.
- [123] Prevedel, Walther, Tiefenbacher, Böhi, Kaltenbaek, Jennewein, and Zeilinger. High-speed linear optics quantum computing using active feed-forward. *Nature*, 445:169–176, 2007.
- [124] Raussendorf, Browne, and Briegel. A one-way quantum computer. *Physical Review Letters*, 86:5188–5191, 2001.
- [125] Raussendorf, Browne, and Briegel. Measurement-based quantum computation on cluster states. *Physical Review A*, 68:022312, 2003.
- [126] Reiher et al. Elucidating reaction mechanisms on quantum computers. *PNAS*, 2017.
- [127] Ren et al. Ground-to-satellite quantum teleportation. *Nature*, 549:70–73, 2017.
- [128] Rivest, Shamir, and Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.
- [129] Roland and Cerf. Quantum search by local adiabatic evolution. *Phys. Rev. A.*, 65, 2001.
- [130] Sauerhoff and Sieling. Quantum branching programs and space-bounded nonuniform quantum complexity. *TCS*, 334, 2005.
- [131] Scherer. *Mathematik der Quanteninformatik*. Springer-Verlag, 2016.
- [132] Schönhage and Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.

- [133] Schrödinger. Die gegenwärtige Situation in der Quantenmechanik. *Die Naturwissenschaften*, 23, 1935.
- [134] Shor. Scheme for reducing decoherence in quantum computer memory. *Physical Review A.*, 52:2493–2496, 1995.
- [135] Shor. Polynomial-time algorithms for prime factorization and discrete logarithms. *SIAM Journal on Computing*, 26:1484–1509, 1997. arXiv:quant-ph/9508027.
- [136] Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, 1994.
- [137] Sipser. *Introduction to the Theory of Computation*. Cengage, 2013.
- [138] Stein and Newman. *Spin Glasses and Complexity*. Princeton University Press, 2013.
- [139] Stewart. Mathematical recreations. *Scientific American*, April, 2000.
- [140] Susskind and Friedman. *Quantum Mechanics*. Basic Books, 2014.
- [141] Tame et al. Experimental realization of a one-way quantum computer algorithm solving simon's problem. *Phys. Rev. Lett.*, 113, 2014.
- [142] Terhal. Quantum error correction for quantum memories. *Rev. Mod. Phys.*, 87, 307, 2015. quant-ph/1302.3428.
- [143] Toffoli. Bicontinuous extensions of invertible combinatorial functions. *Mathematical Systems Theory*, 14:13–23, 1981.
- [144] Turing. On computable numbers with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265, 1937.
- [145] Vandersypen et al. Experimental realization of an order-finding algorithm with an NMR quantum computer. *Nature*, 414:883–887, 2001.
- [146] Walther et al. Experimental one-way quantum computing. *Nature*, 434:169–176, 2005.
- [147] Wehner et al. Quantum internet: A vision for the road ahead. *Science*, 362, 2018.
- [148] Wiechers et al. After-gate attack on a quantum cryptosystem. *New J. Phys.*, 13, 2011.
- [149] Wiesner. Conjugate coding. *SIGACT News*, 15:77, 1983.
- [150] Xu, Qi, and Lo. Experimental demonstration of phase-remapping attack in a practical quantum key distribution system. *New J. Phys.*, 12, 2010. arXiv:quant-ph/1005.2376.
- [151] Yarkoni et al. Quantum shuttle: Traffic navigation with quantum computing. In *Proceedings of the 1st ACM SIGSOFT*, 2020.
- [152] Zalka. Grover's quantum searching algorithm is optimal. 1997. quant-ph/9711070.
- [153] Zaman et al. PyQUBO: Python library for mapping combinatorial optimization problems to QUBO form. *IEEE Transactions on Computers*, 2021.
- [154] Zeilinger. *Einstein's Schleier*. C.H.Beck, 2003.
- [155] Zhao et al. Experimental demonstration of time-shift attack against practical quantum key distribution systems. *Physical Review A*, 78, 2008. arXiv:quant-ph/0704.3253v2.

Die Angaben `arXiv:` und `arXiv:quant-ph` beziehen sich auf den unter

`arxiv.org`

erreichbaren Preprint-Server. Wer über den neuesten Stand der Forschung unterrichtet sein will, wird hier fündig.

### **Bildnachweis**

*Den Seitenzahlen folgen die Rechteinhaber.*

**260** und **261** LMU München; **269** Jonas Bylander;

Wir danken allen Rechteinhabern für die freundliche Abdruckgenehmigung!

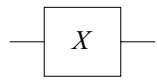
Die Schaltkreise im Kapitel 9 wurden mit dem LaTeX-Makro-Paket *Q-Circuit* von Bryan Eastin und Steven T. Flammia erstellt, Abbildung 11.2 mit Matplotlib, [79].

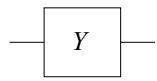
# Symbole und Abkürzungen

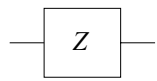
$\oplus$	Addition in der Menge $\{0, 1\}$ ; entspricht dem exklusiven Oder	
$x \cdot y$	bei ganzen oder komplexen Zahlen $x, y$ : gewöhnliche Multiplikation	
$x \cdot y$	bei Vektoren $x, y \in \{0, 1\}^n$ : Skalarprodukt $x_1 y_1 \oplus \dots \oplus x_n y_n$	Seite 16
$A^*$	zu $A$ komplex konjugierte Matrix	Seite 24
$A^\dagger$	zu $A$ adjungierte Matrix	Seite 24
$\langle x   y \rangle$	Skalarprodukt zweier Zustandsvektoren $ x\rangle,  y\rangle \in \mathbb{C}^n$	Seite 32
$ x\rangle \otimes  y\rangle$	Tensorprodukt zweier Zustandsvektoren	Seite 38
$ x\rangle  y\rangle$	Tensorprodukt; verkürzte Schreibweise	
$a = \pm b$	$a = b$ oder $a = -b$	
$\mathbb{N}$	natürliche Zahlen $0, 1, 2, \dots$	
$\mathbb{Z}$	ganze Zahlen $\dots, -2, -1, 0, 1, 2, \dots$	
$\mathbb{C}$	komplexe Zahlen	
$\log$	Logarithmus zur Basis 2	
$\arcsin$	Umkehrfunktion des Sinus	
$O$	asymptotisch kleiner gleich	Seite 70
$\Omega$	asymptotisch größer gleich	Seite 72
$\Theta$	asymptotisch gleich	Seite 72
$\omega_N$	komplexe $n$ -te Einheitswurzel $e^{2\pi i/N}$	Seite 207
$\mathbb{Z}_N^*$	multiplikative Gruppe modulo $N$	Seite 307

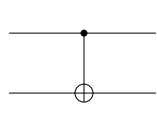
# Quantengatter

	Hadamard-Transformation	$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
---	-------------------------	--

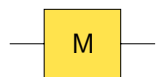
	Pauli-X-Transformation, Bitflip	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
---	---------------------------------	--

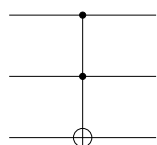
	Pauli-Y-Transformation	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$
---	------------------------	---

	Pauli-Z-Transformation, Phaseflip	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
---	-----------------------------------	---

	gesteuerte Negation (controlled not)	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
---	--------------------------------------	--

	gesteuertes $U$	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & & U \\ 0 & 0 & & \end{pmatrix}$
--	-----------------	--

	Messung	Seite 78
---	---------	----------

	Toffoli-Gatter	Seite 87
---	----------------	----------



# Namen- und Sachwortverzeichnis

- 3-SAT, 272
- abelsche Gruppe, 231
- Adams, Douglas, 123
- Addition ganzer Zahlen, 10, 99
- Additionsschaltkreis, 16
- adiabatische Quantencomputer, 271–274
- Algorithmus, 119
- amplification, *siehe* Probability amplification
- Amplitude, 20
- Amplitudenverstärkung, 138
- Angriff, 169, 260
- asymmetrische Verschlüsselung, 192
- asymptotisches Wachstum, 70
- Authentisierung, 185
- Babbage, Charles, 1
- Basis, 297
- Orthonormal-, 299
- Standard-, 297
- Basistransformation, 45
- Basiswechsel, *siehe* Basistransformation
- BB84-Protokoll, 172–185, 187, 257–259
- bedingter Vorzeichenwechsel, 138, 143
- Bell, John, 54, 58, 284, 286, 287
- Bell-Basis, 55, 127
- Bell-Messung, 127
- Bell-Zustand, 54
- Bells Theorem, 58
- Bells Ungleichung, 59, 186, 287
- Benioff, Paul, 290
- Bennett, Charles, 132, 172, 290
- Berechnung, 11
- deterministische, 103
- gestörte, 95–98, 162
- nichtdeterministische, 103
- randomisierte, 104–111
- umkehrbare, 6, 32, 52, 83–92
- Berechnungsmodell, 74
- Berechnungsproblem, 99
- Bernstein, Ethan, 66
- Bernstein-Vazirani
- Algorithmus von, 65
- bit, 132
- Bitflip, 79, 237
- Black Box, *siehe* Orakel
- Blind Quantum Computing, 267
- Bohr, Niels, 284–287
- Born, Max, 288
- Boyer, Michel, 153
- BPP, 109
- BQP, 117, 116–120, 164–165
- Bra-Vektor, 301
- Brassard, Gilles, 153, 172
- Briegel, Hans, 265
- Broglie, Louis de, 289
- Carmichael-Zahlen, 106
- CHSH-Ungleichung, 59, 186
- Chuang, Isaac, 261
- Church, Alonzo, 119
- Church-Turing-These, *siehe* Churchsche These
- Churchsche These, XII, 119
- Cirac, Ignacio, 263
- CNOT, 30, 78
- D-Wave Systems, 274, 291
- Datenbank, klassische, 158
- Datenbankorakel, 137
- Datenbanksuche, 136, 137, 157, 164
- Dekohärenz, 22, 98, 235–240, 252
- Dekohärenzzeit, 252
- deterministisch, 103
- Deutsch
- Algorithmus von, 9, 33–37, 50–51, 232, 267
- Deutsch, David, 33, 252, 287, 291
- Deutsch-Jozsa
- Algorithmus von, 9, 62–65, 267
- DFT, *siehe* Fouriertransformation, diskrete
- dichte Kodierung, 3, 130–132
- Dichtematrizen, 239

- Diophant, 234
- Dirac, Paul, 20
- Dirac-Notation, 20
- diskrete Exponentialfunktion, 229
- diskreter Logarithmus, 229
- Divide-and-conquer-Algorithmus, 210
- Dreiecksungleichung, 298
- Dürr, Christoph, 154
  
- effizient, 6, 74, 76, 119
- Effiziente Potenzierung, 196
- Eigenvektor, 230
- Eigenwert, 230
- Einheitskostenmaß, 73
- Einheitsmatrix, 302
- Einheitsvektoren, 297
- Einheitswurzeln, 207
  - Halbierungslemma, 209
  - Summationslemma, 211
- Einstein, Albert, 57, 253, 283–285
- Einstein-Podolsky-Rosen-Paradoxon, 57, 284
- Einweg-Quantenberechnungen, 265
- Einzelphotonenquelle, 189, 259
- Ekert, Artur, 185
- Ekert-Protokoll, 185–189
- entanglement, 53
- Entanglement swapping, 129
- Entropie, 92
- Entscheidungsprobleme, 100
- Entscheidungsvariante, 100, 112
- EPR-Paar, 55, 125, 126, 131
- Ersetzungsschiffre, 169
- Euklids Algorithmus, 305
- Euler, Leonard, 308
- Euler, Satz von, 308
- Eulersche  $\phi$ -Funktion, 193, 308
- Everett, Hugh, 286
  
- Faktorisierung, 4, 7, 115, 191, 192, 199–204, 221–229
- Fehler, 95–98, 235–240
- fehlerkorrigierende Codes, 98, 180, 240–249, 252
- Fermat
  - Satz von, 106
- Fermat, kleiner Satz von, 308
- Fernwirkung, 57, 283–284
- Feynman, Richard, 289, 290
- FFT, *siehe* Fouriertransformation, schnelle
- Fichte, Johann Gottlieb, 287
  
- Fluoreszenz, *siehe* parametrische Fluoreszenz
- Flux-Qubit, 267–269
- Fouriertransformation
  - diskrete, 204–211
  - Quanten-, 212–216, 230, 231
  - schnelle, 204, 210–211
- Fredkin, Edward, 88
- Fredkin-Gatter, 88
  
- G-BBHT-Suche, 153, 158
- Gatter
  - klassische, 14
  - Quanten-, 76–81
  - universelles, 86
- gatterbasiert, 5, 271, 292
- Gefangendilemma, 267
- gemischter Zustand, 78, 239
- gestörte Berechnungen, 95–98, 162
- Gleichverteilung, 303
- Google, 291
- Graph, 113, 156
- Grover, Lov, 3–5, 135
- Grover-Iteration, 141
- Grovers Algorithmus, 3, 144, 135–165
  - Grundidee, 141
- Grundzustand, 272
- Gruppe, 231
- Größenordnung, 70
  
- Hadamard-Basis, 45
- Hadamard-Transformation, 40, 42, 60, 59–62, 78, 231, 256
- Halbaddierer, 16, 89, 90
- Hallgren, Sean, 234
- Hallgrens Algorithmus, 234
- Hamiltonkreis, 113
- Hamiltonoperator, 272
- HC, *siehe* Hamiltonkreis
- Hegel, G. W. F., 9
- Heisenberg, Werner, 287–289
- Heisenbergsche Unbestimmtheitsrelation, 288
- Hertz, Heinrich, 283
- hidden subgroup, 232
- Hilbertraum, 300
- Holevo, Alexander, 134
- Holevoschranke, 134
- Høyer, Peter, 153, 154
  
- Informationsverlust, 91

- Informationsmaß, 132
- inneres Produkt, 298
- Interferenz, 3, 37, 118, 253
- Interferometer, 253
- Interpolation, 206
- Ionenfallen, 263
- Ising-Modell, 276, 277
  
- Jozsa, Richard, 63
  
- Kanal
  - klassischer, 124
  - Quanten-, 124, 129
  - verrauschter, 125, 129
- Kant, Immanuel, 287
- Kerckhoffs, Auguste, 169
- Kerckhoffssches Prinzip, 169
- Kernspinresonanz, 261, 262
- Ket-Vektor, 301
- Kettenbrüche, 228
- Klartext, 168
- klassisch, 9
- Koeffizientendarstellung, 205
- komplex konjugiert, 294
- komplexe Zahlen, 293
- Komplexität, 99, 164
  - klassische Suche, 137
  - Quantensuche, 159
- Komplexitätsklasse, 100
- Komplexitätstheorie, 164
- kongruent, 305
- konjugiert, *siehe* komplex konjugiert
- kontrollierte Negation, *siehe* CNOT
- kontrollierte Operationen, 80
- Kopenhagener Interpretation, 285–287
- Kryptographie, 4, 136, 189, 192–197, 230
  - asymmetrische, 192
  - Quanten-, 172–189, 257–261
- Kryptotext, 168
  
- Lambda-Kalkül, 119
- Landauer, Rolf, 67, 92
- Landauers Prinzip, 92
- Laufzeit, 68, 68–74
  - erwartete, 74
  - im schlechtesten Fall, 73
  - lineare, 69
  - polynomiale, 74, 101
  - quadratische, 69
- lauschen, 168
  
- Leibniz, Gottfried Wilhelm, 282
- Lichtenberg, Georg Christoph, 99
- linear
  - lineare Abbildung, 32, 302
  - lineare Laufzeit, 69
  - lineare Transformation, 32, 302
- linear unabhängig, 297
- lineare Hülle, 296
- Linearkombination, 296
- logarithmisches Kostenmaß, 73
- lokal realistisch, 58
- lokale unitäre Transformation, 31, 77
- Lokalitätsprinzip, 57–59
  
- Mach, Ludwig, 253
- Mach-Zehnder-Interferometer, 253–256
- Man-in-the-middle attack, 185
- Matrix
  - adjungierte, 24
  - invertierbare, 302
  - orthogonale, 302
  - Permutations-, 31
  - transponierte, 301
  - unitäre, 24, 32, 302
- MAX-CUT, 276
- maximal verschränkt, 57
- Mehrheitsvotum, 109
- Mergesort, 210
- Messbasis, 44
- Messung, 3, 18, 26, 29, 46, 44–49, 52, 78
  - als Observable, 48
  - als Projektion, 47
  - bezüglich einer Basis, 45
  - einzelner Bits, 46
- Miller-Rabin-Test, 107, 108
- Miniaturisierung, 1, 92
- modulo, 305
- multiplikativ Inverses, 307
- multiplikative Gruppe, 307
- Münzwurf, 26, 42, 105
  
- Newton, Isaac, 253, 283
- nichtdeterministisch, 103
- NISQ, 292
- NMR-Computer, 261
- No-Cloning-Theorem, 6, 81–83, 98, 167, 240
- Norm, 32, 298
- NP, 102, 158, 164
- NP-schwer, 115

- NP-vollständig, 6, 114, 111–116, 158, 164  
 Number partition, *siehe* PARTITION
- O*-Notation, 70  
 OBDD, 92  
 Observable, 48  
 Occam, William von, 287  
 One-Time Pads, 4, 171  
 Optimierungsprobleme, 135, 136, 158, 164, 271–281  
 Optimierungsvariante, 112  
 Orakel, 33, 163  
 Ordnung, 200  
 orthogonal, 299  
 Orthonormalbasis, 299
- P, 101  
 parametrische Fluoreszenz, 259, 266  
 PARTITION, 279  
 Pauli-Matrizen, 79  
 Pellsche Gleichung, 233  
 Periode, 198, 197–204  
 PGP, 193  
 Phase, 22, 295  
 Phaseflip, 79, 237  
 Phasenbestimmung, 230  
 photoelektrischer Effekt, 253, 283  
 Photonen, 253–261, 283  
     polarisierte, 257, 266  
 Planck, Max, 1, 281–285  
 Podolsky, Boris, 57, 284  
 Polarisation, *siehe* Photonen, polarisierte  
 Polynome  
     Koeffizientendarstellung, 205  
     Multiplikation, 205  
     Stützstellendarstellung, 206  
 Polynomialzeit, 74, 101  
 Polynomialzeitreduktion, 113  
 Polynommultiplikation, 205, 205  
 Preskill, John, 292  
 Primfaktorzerlegung, 4, 7, 115, 191, 192, 192, 199–204, 221–229  
 Primzahl, 100, 192, 305  
 Primzahltest, 105  
 Privacy amplification, 179  
 Probability amplification, 108  
 Problem des Handlungsreisenden, 7, 111, 136, 158, 164, 274  
 Projektion, 47, 300–301
- Pseudozufallszahlen, 26, 172  
 PSPACE, 118  
 Public Key-Kryptographie, 192, 230
- QFT, *siehe* Quanten-Fouriertransformation  
 Quanten-Fouriertransformation, 212–216, 230, 231  
 Quanten-Repeater, 129  
 Quantendatenbanken, 158  
 Quantenbit, 2, 20, 52, 251  
     kopieren, 81–83  
 Quantengatter, 76–81  
 Quanteninternet, 129  
 Quantenkanal, 124  
 Quantenkryptographie, *siehe* Kryptographie, Quanten-  
 Quantenparallelismus, 2, 12  
 Quantenpunkte, 189, 259  
 Quantenregister, 29, 51  
 Quantenschaltkreise, 76–81  
 Quantenspiele, 267  
 Quantensuche  
     Komplexität der, 159  
 Quantenteleportation, 3, 123–130, 259  
 Quantenvorteil, 292  
 Quantum Annealing, 274–280, 291–292  
 Quantum supremacy, *siehe* Quantenvorteil  
 QUBO, 276  
 Quicksort, 104, 210
- randomisiert, 105, 104–111  
 Raussendorf, Robert, 265  
 Realismus, lokaler, *siehe* lokal realistisch  
 Rechenschritt, 11, 52  
 Redundanz, 170  
 Register, 28  
 reiner Zustand, 239  
 reversibel, *siehe* Berechnung, umkehrbare  
 Rosen, Nathan, 57, 284  
 RSA-Kryptographie, 4, 191–197, 203  
 Rutherford, Ernest, 284
- Schaltkreise  
     klassische, 14–17, 74–76  
     umkehrbare, 83–92  
     uniforme, 75  
 Schlüsselerzeugung, 175  
 Schlüsselverteilung, 175

- schnelle Fouriertransformation, 210–211
- Schrödinger, Erwin, 18, 20, 53, 289–290
- Schrödingergleichung, 268, 272
- Schrödingers Katze, 17–20, 22, 49, 235, 285, 286
- Schüssel
  - öffentlicher, 192
  - geheimer, 169
- Shor, Peter, 4, 5, 7, 191, 230, 291
- Shors 9-Qubit-Code, 245–248
- Shors Algorithmus, 157, 202, 191–229, 262
  - klassischer Teil, 202
  - Laufzeit, 228
  - Quantenteil, 222
- Simon, Daniel R., 217
- Simons Algorithmus, 216–221, 231, 267
- Simulated Annealing, 274, 275
- Simulation, 75
- Simulation von Quantenprozessen, 5 simulieren, 84
- Single Source Shortest Path, 156
- Skalar, 296
- Skalarprodukt, 32, 43, 298
- skalierbar, 262
- Speicherverbrauch, 118
- Spiegelung am Mittelwert, 139
- Spieltheorie, 105, 267
- Spin-Glas, 280
- SQUID, 268
- Stabilizer-Code, 249
- Standardbasis, 44, 45
- Stapp, Henry, 284
- Strahlteiler, 253
- Stützstellendarstellung, 206
- superdense coding, *siehe* dichte Kodierung
- superdichte Kodierung, 3, *siehe* dichte Kodierung
- Superposition, 2, 12, 17, 20, 52
  - gleichgewichtete, 43
- Supraleiter, 267–269
- Sycamore, 249, 292
- Tapp, Alain, 153
- Teiler, 305
- Teiler, echter, 305
- teilerfremd, 305
- Teleportation, 237, *siehe* Quantenteleportation
- Tensor- und Skalarprodukt, 43
- Tensorprodukt, 37–43
- Toffoli, Tommaso, 87
- Toffoli-Gatter, 87–92
- TSP, *siehe* Problem des Handlungsreisenden
- Tunneleffekt, 268, 276
- Turing, Alan, XI, 13
- Turingmaschine, 13–14
  - universelle, 13
- Überlagerung, *siehe* Superposition
- umkehrbar, *siehe* Berechnung, umkehrbare
- uniform, 75
- unitär, 24, 32
- universell, 86
- untere Schranken, 159–164
  - mit Polynomen, 164
- unterscheidbare Zustände, 92–95
- unverschränkt, 55
- Vandermonde Matrix, 206
- Vazirani, Umesh, 66
- Vektorraum, 295–301
- verborgene Variablen, 58, 284, 286
- Verifikation
  - effiziente, 102
- Vernam, Gilbert, 4, 171
- Verschlüsselung, *siehe* Kryptographie
- Verschränkungsreinigung, 129
- Verschränkung, 3, 53–59, 125, 132–134, 259, 284
  - maximale, 57
  - Maß der, 57
- Verschränkungsdestillation, *siehe* Verschränkungsreinigung
- Volladdierer, 16
- von Neumann, John, XI
- Wahrscheinlichkeit, 21–22
- Wahrscheinlichkeitsverteilung, 303
- Weinfurter, Harald, 128, 189, 260
- Weizsäcker, Carl Friedrich von, 287
- Wiederholungscode, 240
- Wiesner, Stephen, 132
- Zahlentheorie, 305
- Zehnder, Ludwig, 253
- Zeilinger, Anton, 128, 167

Zoller, Peter, 263  
Zufallsgenerator, 26, 42, 256–257  
Zufallsvariable, 303  
Zufallszahlen, 26, 256–257  
zufällig wählen, 304  
Zuse, Konrad, XI, 1  
Zustandsvektor, 22  
Zählen, 157