About Shor's algorithm and quantum computers

Über den Shor-Algorithmus und Quantencomputer

Wissenschaftliche Arbeit
zur Erlangung des akademischen Grades

Bachelor of Science

an der Westfälischen Wilhelms-Universität Münster

Eingereicht von

Jan Mandrysch
jmandrysch@wwu.de

am

13. August 2016

Erstgutachter: Prof. Dr. Raimar Wulkenhaar
Zweitgutachter: Prof. Dr. Gernot Münster

Inhaltsverzeichnis

1 Der Shor-Algorithmus und seine Motivation **3**
1.1 RSA-Verschlüsselung 5

2 Theoretische Grundlagen **7**
2.1 Endliche Gruppen- und Zahlentheorie 8
2.2 Theoretische Informatik 9
2.2.1 Algorithmen und Komplexitätstheorie 9
2.2.2 Rechnermodelle 10
2.2.3 Quantenalgorithmen 12
2.3 Quantenmechanische Zustände und Messprozesse 13
2.4 Quantum Computation - Qubits und Quantenregister 16
2.5 Quantum Computation - Quantengatter 18
2.5.1 Elementare Gatter für ein Qubit 18
2.5.2 Anwendung von 1-Qubit-Gattern auf Multi-Qubits 19
2.5.3 Kontrollierte Gatter 20
2.5.4 Universelle Quantengatter 21

3 Detaillierte Erläuterung des Algorithmus **23**
3.1 Reduktion des Faktorisierungsproblems 24
3.2 Das Ordnungsbestimmungsproblem 26
3.3 Quantenfouriertransformation 26
3.4 Quantenphasenschätzung 27
3.5 Ordnungsbestimmung mittels Quantenphasenschätzung 33
3.6 Kettenbruchentwicklung zur Ordnungsbestimmung 35
3.7 Vorschlag zur Verbesserung des Algorithmus 37
3.8 Zusammenfassung des vollständigen Algorithmus 39
3.9 Beispielhafte Durchführung des Algorithmus für $N = 21$ 40

4 Abschätzung des Aufwandes **42**

5 Schwierigkeiten bei der Umsetzung - Konstruktion von Quantencomputern **49**

Vorwort

Bei diesem Dokument handelt es sich um eine Bachelorarbeit über den Shor-Algorithmus zur Lösung des Problems der Primfaktorzerlegung. Der Fokus der Arbeit liegt auf dem nicht-klassischen Teil des Shor-Algorithmus, das heißt dem Teil, der auf der Berechnung durch einen Quantencomputer basiert. Neben der Erläuterung des Algorithmus soll die Herleitung einer oberen Grenze für den Aufwand dieses Algorithmus gegeben werden. Außerdem soll erläutert werden, warum die Umsetzung des Shor-Algorithmus in die Praxis und die dafür unabdingbare Konstruktion eines Quantencomputers bisher Probleme bereitet. Die Arbeit hat zum Ziel sowohl den Anforderungen des Studiengangs Physik als auch denen des Studiengangs Mathematik gerecht zu werden. Daher werde ich in dieser Arbeit versuchen den Shor-Algorithmus sowohl aus der Perspektive der Physik als auch aus der Perspektive der Mathematik zu beleuchten. Dies bedeutet, dass einerseits auf die dem Algorithmus zugrundeliegende Quantenmechanik und die physikalischen Aspekte der Realisierung eines Quantencomputers eingegangen wird und andererseits auch auf mathematische Untersuchungen nicht verzichtet wird. Mit mathematischen Untersuchungen ist hier in der Regel gemeint, die Gültigkeit bestimmter Schritte oder Aussagen zu beweisen. Da der Umfang, den diese Arbeit einnehmen darf, begrenzt ist, wird an manchen Stellen auf andere Arbeiten und Werke verwiesen, wenn Herleitungen für den Algorithmus weniger relevant sind oder zu viel Platz einnehmen.

Das Dokument ist wie folgt aufgebaut: Im ersten Kapitel wird der Shor-Algorithmus kurz beschrieben und seine Bedeutung herausgestellt. Das nachfolgende Kapitel dient der Einführung in benötigte mathematische und physikalische Inhalte. Sollten diese bereits bekannt sein, kann das Kapitel auch als Festlegung der gewählten Notation dieser Arbeit verstanden werden. Die behandelten Themengebiete umfassen eine kurze Einführung verwendeter Begriffe der endlichen Gruppen- und Zahlentheorie, eine Erläuterung grundlegender Konzepte der theoretischen Informatik wie zum Beispiel dem Algorithmus, der Rechenmaschine und der Laufzeitkomplexität, die Darstellung quantenmechanischer Zustände und Messprozesse und darauf aufbauend die Erläuterung der grundlegenden Modellierung eines Quantencomputers. Im dritten Kapitel wird der Shor-Algorithmus im Detail beschrieben. Es wird gezeigt, dass das Faktorisierungsproblem auf ein Ordnungsbestimmungsproblem reduziert werden und dieses wiederum durch Quantenphasenschätzung und eine Kettenbruchentwicklung gelöst werden kann. Am Ende des Kapitels wird eine mögliche Formulierung des vollständigen Algorithmus zur Lösung des Faktorisierungsproblems angegeben und dieser für den Spezialfall der Faktorisierung der Zahl 21 beispielhaft ausgeführt. Im vierten Kapitel wird eine obere Grenze für den asymptotischen Aufwand des Algorithmus hergeleitet. Im fünften Kapitel wird schließlich erläutert, warum eine Umsetzung des vollständigen Shor-Algorithmus bisher noch nicht möglich ist. Der wesentliche Grund ist, dass der Quantencomputer bisher nur als Zusammenschaltung weniger Qubits realisiert werden konnte, und die Skalierung dieser Systeme große Schwierigkeiten bereitet. In diesem Sinne bleibt es abzuwarten, welche der vielfältigen experimentellen Ansätze schließlich zu einem Erfolg führt und eine Anwendung des Shor-Algorithmus praktikabel werden lässt.

1 Der Shor-Algorithmus und seine Motivation

Seit der Erfindung des Computers zeigt unsere technologische Entwicklung ein unglaublich schnelles Wachstum. Innerhalb mehrerer Jahrzehnte wurden aus einfachen Rechenmaschinen leistungsstarke Supercomputer, die heute Erstaunliches leisten. Über diese Zeitspanne hinweg folgten Größen wie die Speicherkapazität und die Transistorendichte einem exponentiellen Trend. Für die Transistorendichte wurde diese Entwicklung schon 1965 von Gordon E. Moore in Form des heute so genannten Moore'schen Gesetzes formuliert.[1] Nach diesem Gesetz verdoppelt sich die Anzahl von Schaltkreiskomponenten auf einem integrierten Schaltkreis mit minimalen Komponentenkosten etwa alle 18 Monate.[1] Das Mooresche Gesetz zeigte bis heute eine verblüffende Beständigkeit. Doch inzwischen wirken die Herausforderungen zur Fortsetzung des exponentiellen Trends gigantisch. Aktuelle Prozessor-Mikroarchitekturen werden bereits in 14-nm-Fertigungstechnik gebaut.[2] Eine Fortsetzung dieses Trends würde schon in wenigen Jahrzenten zu schwer vorstellbaren Transistorendichten von wenigen Transistoren pro Atom führen. Schon jetzt bei den aktuellen Transistorgrößen im unteren Nanometerbereich werden Schwierigkeiten deutlich. In den letzten Jahren kam es daher zu einem leichten Stagnieren in der Entwicklung der Transistorendichte. Der Grund hierfür sind Probleme, die für die kleiner werdenden Prozessorarchitekturen immer bedeutsamer werden. Denn im unteren Nanometerbereich werden allmählich quantenmechanische Effekte relevant, die die Funktionsweise eines klassischen Computers beeinträchtigen. Eine bisher praktizierte fortwährende Skalierung der bekannten klassischen Architekturen scheint daher innerhalb der nächsten Jahrzehnte an ihre Grenzen zu stoßen. So ist es möglicherweise an der Zeit über Alternativen nachzudenken. Eine dieser Alternativen beschäftigt sich gerade mit den quantenmechanischen Effekten, die den klassischen Computerarchitekturen bald Probleme bereiten werden. Die Quantenmechanik bietet die Möglichkeiten ganz neuer Computerarchitekturen. Das theoretische Konzept einer Rechenmaschine, die auf quantenmechanischen Effekten beruht, ist bereits seit mehreren Jahrzehnten bekannt. Bei der Untersuchung von Quantencomputern durch Benioff konnte gezeigt werden, dass ein Quantencomputer einer Turingmaschine[2]zumindest *ebenbürtig* ist.[3][4] In diesem Fall meint ebenbürtig, dass ein Quantencomputer eine Turingmaschine in polynomialer Laufzeit simulieren kann. Mit anderen Worten: Ein Quantencomputer ist mindestens genauso schnell wie eine Turingmaschine. Diese Aussage erscheint insofern natürlich, als dass die klassische Mechanik als ein spezieller Grenzfall aus der Quantenmechanik hervorgeht und ein Quantencomputer daher auch der Simulation der klassischen Mechanik mächtig sein sollte. Würde zudem eine Turingmaschine ebenso einen Quantencomputer in polynomialer Laufzeit simulieren können, so wären Turingmaschine und Quantencomputer aus Sicht der Komplexitäts- und Berechenbarkeitstheorie äquivalent. Das würde bedeuten, dass jedes Problem von Turingmaschine und Quantencomputer asymptotisch gleich schnell gelöst werden könnte. Bisher sind aber nur Verfahren bekannt, die exponentielle Laufzeit für die Simulation eines Quantencomputers auf einer Turingmaschine benötigen. Dies deutet an, dass Quantencomputer tatsächlich schneller sein können als klassische Rechenmaschinen. Ein Argument, dass diesen Umstand etwas beleuchtet, wäre, dass dem Quantencomputer mit dem komplexen Zah-

[1] Im Originalpaper gab Moore zunächst eine Verdopplungszeit von einem Jahr an. Später wurde diese Zeit aber auf 18 Monate korrigiert.

[2] Die Turingmaschine ist ein einfaches Konzept einer Rechenmaschine in der theoretischen Informatik. In diesem Sinne kann die Turingmaschine hier als Modell eines klassischen Computers verstanden werden. Siehe auch Kapitel 2.2.1.

lenraum ein sehr viel dichterer oder auch mächtigerer Zahlenraum zur Verfügung steht als der klassischen Rechenmaschine mit dem Raum der ganzen Zahlen. Dieses Argument greift aber etwas zu kurz, da es vernachlässigt, dass zur Extraktion einer Information aus dem Quantencomputer ein quantenmechanischer Messvorgang initiiert werden muss, der die Komplexität des Rechenraums wieder auf diskrete Zahlen kollabieren lässt.

Als Peter W. Shor 1997 seinen entdeckten Algorithmus veröffentlichte[5], hatte dies enorme Konsequenzen für die Erforschung von Quantencomputern. Die Veröffentlichung von Shor enthielt die algorithmische Lösung zweier zahlentheoretischer Probleme auf einem Quantencomputer: Die Primfaktorzerlegung und die Berechnung des diskreten Logarithmus. Das Entscheidende dabei war, dass die vorgestellten Algorithmen eine polynomiale Laufzeitkomplexität besaßen, während alle zuvor entdeckten klassischen Algorithmen superpolynomiale Laufzeitkomplexität zeigten. Damit war es gelungen zumindest theoretisch zu zeigen, dass ein Quantencomputer im Stande ist, Probleme *effizient*, das heißt mit polynomialen Aufwand, zu lösen, die - zumindest bis heute - auf klassischen Rechenmaschinen nicht effizient gelöst werden können.[3] Das von Shor vorgestellte Verfahren zur Primfaktorzerlegung wird inzwischen weithin als Shor-Algorithmus bezeichnet und ist, wie schon zuvor festgestellt, das Thema dieser Abhandlung.[4]

Es sollte noch erwähnt sein, dass es noch einige andere Pioniere im Bereich der Quanteninformatik gab, die etwa zurzeit von Shors Publikationen Algorithmen vorstellten, die verschiedene Probleme unter Zuhilfenahme eines Quantencomputers schneller lösten als klassische Algorithmen bislang in der Lage waren. Zu nennen sind vor allem der Deutsch-Jozsa-Algorithmus[6] und der Grover-Algorithmus[7]. Der Grover-Algorithmus löst das Problem der Suche in einer unsortierten Datenbank und ist bis heute der einzige Nachweis dafür, dass Quantencomputer prinzipiell schneller sind als klassische Rechenmaschinen. Denn es ist bewiesen, dass der Algorithmus der linearen Suche mit einer Laufzeitkomplexität von $\mathcal{O}(n)$ der schnellste klassische Algorithmus zur Lösung des Problems der Suche in einer unsortierten Datenbank ist, während der Grover-Algorithmus eine Laufzeitkomplexität von $\mathcal{O}(\sqrt{n})$ besitzt. Da der Geschwindigkeitsvorteil aber nicht exponentiell, sondern nur quadratisch ist, liefert der Grover-Algorithmus noch keine Antwort auf die Äquivalenz oder Nicht-Äquivalenz von Quantencomputer und Turingmaschine.

Die Bedeutung des von Shor gefundenen Algorithmus liegt nun zum einen in der exponentiellen Verbesserung des Laufzeitaufwandes und zum anderen in dem Einsatz der Primfaktorzerlegung im Bereich der Kryptographie. Denn die Sicherheit einer Vielzahl der heute gängigen Verschlüsselungsmethoden stützt sich auf den Umstand, dass die Primfaktorzerlegung einer beliebigen Zahl bis dato nicht effizient durchgeführt werden kann. Eine vollständige Realisierung des Shor-Algorithmus auf einem genügend groß skalierten Quantencomputer würde all diesen Verfahren ihre Grundlage entziehen und sie unsicher machen. Bisher sind Quantencomputer allerdings nur als Zusammenschaltung

[3] Die vorsichtige Formulierung ist dem Umstand geschuldet, dass der Beweis der Unmöglichkeit der effizienten Lösung der beiden Probleme auf einem klassischen Computer bisher nicht erbracht wurde.

[4] Mit der Bezeichnung Shor-Algorithmus wird auch das zweite vorgestellte Verfahren zur Berechnung des diskreten Logarithmus bezeichnet. In der Regel ist aber das Verfahren zur Bestimmung der Primfaktorzerlegung gemeint.

weniger Qubits[5] und nicht in größer skalierten Systemen realisiert. Daher beschränkte sich eine Durchführung des Shor-Algorithmus bisher auf mehrere Spezialfälle kleiner Zahlen in einem kleinen Maßstab. Die erste Durchführung des Shor-Algorithmus gelang 2001 bei der Durchführung der Faktorisierung der Zahl 15 in 3 und 5 unter Verwendung von Kernspinresonanzen.[8] Inzwischen wurden weitere Faktorisierungen der Zahl 15 von verschiedenen unabhängigen Arbeitsgruppen unter Verwendung verschiedener experimenteller Ansätze durchgeführt. (Photonische Systeme [9][10][11], Supraleiter [12]) Die bisher größte mit dem Shor-Algorithmus faktorisierte Zahl ist die 21.[13] Alle diese experimentellen Realisierungen wurden allerdings nur für bestimmte Spezialfälle konstruiert und in einer kompilierten Version ausgeführt.[6] Eine vollständige und allgemeine Realisierung des Shor-Algorithmus ist bisher noch nicht gelungen.

1.1 RSA-Verschlüsselung

Um die Bedeutung der Primfaktorzerlegung für die Kryptographie deutlich zu machen, soll eines der am häufigsten verwendeten Verschlüsselungsverfahren als Beispiel erläutert werden - die RSA-Verschlüsselung. RSA gehört zu den Public-Key-Verfahren. Das grundlegende Prinzip eines Public-Key-Verfahrens kann auf die folgende Weise veranschaulicht werden:

> Nehmen wir an, dass Alice eine geheime Nachricht an Bob übermitteln will. Dann wird von Bob gefordert, dass er sich einen geheimen Schlüssel (Private Key) ausdenkt, der später zum Entschlüsseln der verschlüsselten Botschaft benutzt werden soll. Aus diesem geheimen Schlüssel erzeugt Bob nun einen weiteren Schlüssel, den er öffentlich macht, das heißt an Alice weitergibt. Jeder der diesen öffentlichen Schlüssel (Public Key) kennt - also auch Alice - kann nun eine Nachricht mit Hilfe dieses Schlüssels verschlüsseln und an Bob senden. Auch wenn die verschlüsselte Botschaft von anderen Personen abgefangen wird, können diese Personen die Botschaft nicht entschlüsseln, da ihnen der private Schlüssel zur Entschlüsselung fehlt, den nur Bob kennt.

Wichtig für ein solches Public-Key-Verfahren ist es nun, dass die Erzeugung des öffentlichen Schlüssels idealerweise unumkehrbar ist. Das heißt, dass vom öffentlichen Schlüssel nicht auf den privaten Schlüssel geschlossen werden kann. Da der öffentliche Schlüssel aber in irgendeiner Form vom privaten Schlüssel abhängen muss, ist dies nicht gänzlich zu vermeiden. Hinreichend für die Sicherheit eines Kryptographieverfahrens ist es aber schon, wenn die Entschlüsselung der verschlüsselten Botschaft so viel Zeit benötigt, dass die Relevanz der Botschaft nach der Entschlüsselung verloren ist. Das RSA-Verfahren verwendet als öffentlichen Schlüssel das Zahlenpaar (e, N) und als privaten Schlüssel das Zahlenpaar (d, N).

[5] quantenmechanisches Pendant zum klassischen Bit

[6] Für die gewählten Spezialfälle werden manche Qubits und Gatter überflüssig. Eine kompilierte Ausführung bedeutet dementsprechend redundante und überflüssige Elemente des Algorithmus wegzulassen. Dadurch vereinfacht sich der experimentelle Aufwand und es werden weniger Qubits zur Ausführung benötigt.

Etwas vereinfacht stellt sich der Ablauf des Verfahrens wie folgt dar:

1. Zufällige Wahl zweier Primzahlen $p \neq q$
2. Berechnung von $N = p \cdot q$
3. Berechnung der Eulerschen-φ-Funktion von N: $\varphi(N) = (p-1) \cdot (q-1)$
4. Wahl einer zu $\varphi(N)$ teilerfremden Zahl e (Verschlüsselungsexponent) mit $1 < e < \varphi(N)$
5. Berechnung von d (Entschlüsselungsexponent) mit $e \cdot d \equiv 1 \pmod{\varphi(N)}$

Damit ist also das Paar aus privatem und öffentlichem Schlüssel bestimmt. Wenn m die zu verschlüsselnde Nachricht ist, dann ergibt sich der verschlüsselte Text c durch Berechnung der e-ten Potenz $c \equiv m^e \pmod{N}$. Andersherum wird eine Entschlüsselung erreicht durch die Exponentiation mit dem Entschlüsselungsexponenten. Das heißt es gilt $m \equiv c^d \pmod{N}$, denn es gilt nach Konstruktion der beiden Exponenten e und d: $c^d = (m^e)^d = m^{ed} \equiv m \pmod{N}$. Nun wird der Zusammenhang zum Faktorisierungsproblem ersichtlich. Solange die Faktoren p und q von N für den Angreifer unbekannt sind, muss er die Gleichung $m^e \equiv c \pmod{N}$ nach m hin lösen, das heißt die e-te Wurzel modulo N ziehen. Sobald der Angreifer aber Kenntnis über p und q besitzt, kann er einfach $\varphi(N) = (p-1) \cdot (q-1)$ berechnen. Da ihm e als Teil des öffentlichen Schlüssels bekannt ist, erhält er mit der Ausführung von Schritt 5 den privaten Schlüssel d und kann dementsprechend c entschlüsseln. Die Sicherheit von RSA stützt sich damit darauf, dass sowohl das Problem des Ziehens der e-ten Wurzel modulo N als auch das Faktorisierungsproblem genügend schwer sind. Mit genügend schwer ist hier gemeint, dass die Lösung des Problems genug Aufwand erfordert, sodass jede Relevanz der zu übermittelnden Botschaft innerhalb der verbrauchten Entschlüsselungszeit verlorengeht. Für klassische Algorithmen ist dieser Umstand bisher für beide Probleme gegeben. Denn das Ziehen einer e-ten Wurzel in $\mathbb{Z}_N$, dem Restklassenring der ganzen Zahlen modulo N, benötigt Kenntnis über die Primfaktorzerlegung von N, die bei einer zusammengesetzten Zahl $N = pq$ wieder auf das Faktorisierungsproblem führt. Für das allgemeine Faktorisierungsproblem sind viele verschiedene Lösungsverfahren bekannt, deren Geschwindigkeit häufig von der Art der zu faktorisierenden Zahl abhängt. Daher ist eine Auswahl des schnellsten Faktorisierungsverfahrens nur schwer möglich. Für das Problem der Faktorisierung eines Produktes zweier Primzahlen $N = pq$ ist sicherlich das Zahlenkörpersieb eines der schnellsten Verfahren. Es besitzt eine asymptotische Laufzeit von $e^{\mathcal{O}((\log N)^{\frac{1}{3}}(\log\log N)^{\frac{2}{3}})}$.[14][7] Die Laufzeit des Zahlenkörpersiebs ist damit zumindest subexponentiell, aber immer noch superpolynomial. Der Shor-Algorithmus besitzt dagegen eine nur polynomiale Laufzeitkomplexität. Sie beträgt $\mathcal{O}((\log N)^2 \log\log(N) \log\log\log(N))$.[14] Diese Aufwandsreduktion kann die benötigte Entschlüsselungszeit drastisch reduzieren und damit den Sinn des Verfahrens zunichtemachen.

[7] Der Ausdruck für die Laufzeit ist nicht exakt bewiesen, sondern nur unter einigen Annahmen hergeleitet, die nicht sicher sind, aber zumindest sehr wahrscheinlich gelten (vgl. [15]).

2 Theoretische Grundlagen

Nachdem in den obigen Abschnitten die Bedeutung des Shor-Algorithmus erläutert wurde, stellt sich nun die Frage, wie der genannte Algorithmus funktioniert. Der Algorithmus stützt sich im Wesentlichen auf mehrere Konzepte. Zunächst wird eine zahlentheoretische Erkenntnis dazu eingesetzt, das Problem der Primfaktorzerlegung mit Hilfe effizienter klassischer Verfahren auf die Bestimmung einer Ordnung modulo n zurückzuführen. Die Ordnung einer natürlichen Zahl a modulo n ist die kleinste natürliche Zahl r für die $a^r \equiv 1 \pmod{n}$ gilt. Der Shor-Algorithmus liefert eine Möglichkeit das Bestimmen der Ordnung wesentlich, das heißt exponentiell, schneller als andere klassische Algorithmen zu lösen. Der Aufwand der Ordnungsbestimmung ist für den Aufwand des Gesamtverfahrens ausschlaggebend, da alle anderen Bestandteile des Algorithmus einen geringeren Aufwand zeigen (siehe Kapitel 4). Damit erreicht der Shor-Algorithmus eine exponentiell schnellere asymptotische Laufzeit als alle anderen bisher bekannten Faktorisierungsverfahren. Das Bestimmen der Ordnung geschieht über ein Verfahren, dass sich Quantenphasenschätzung nennt und mit der Quantenfouriertransformation eng verwandt ist. Die Quantenphasenschätzung ist im Wesentlichen ein probabilistisches Verfahren zur Abschätzung eines Eigenwertes eines unitären Operators. Im Falle des Shor-Algorithmus wird ein Operator gewählt, dessen Eigenwert einen Rückschluss auf die Ordnung modulo n zulässt.

Bevor der Algorithmus jedoch in seiner Fülle erläutert werden kann, sollten einige theoretische Grundlagen gelegt werden, die für das Verständnis des Algorithmus wichtig sind. Im Rahmen der endlichen Gruppen- und der Zahlentheorie wird noch einmal definiert, was eine Ordnung modulo n ist. Anschließend findet sich ein Abschnitt aus der theoretischen Informatik, der die wesentlichen in dieser Arbeit auftretenden Begriffe wie zum Beispiel Algorithmus und Laufzeitaufwand beleuchtet. Danach wird eine kurze Einführung in quantenmechanische Messprozesse gegeben. Darauf aufbauend wird das Feld der Quantum Computation eingeführt, bis schließlich so genannte Quantengatter erläutert werden.

Bei den Beschreibungen wird einiges als bekannt vorausgesetzt, so zum Beispiel die Grundlagen linearer Algebra und die Grundlagen der Analysis sowie eine Vertrautheit mit den Konzepten der Quantenmechanik. Dementsprechend gibt dieses Kapitel keine vollständigen Einführungen in die genannten Themengebiete. Vollständigere Einführungen in die mathematischen und quantenmechanischen Grundlagen zur Erschließung der Quanteninformatik und speziell des Shor-Algorithmus liefern zum Beispiel [16] und [17]. Ersteres gibt dabei eine sehr umfangreiche Einführung in Quantenmechanik, Quantum Computation und Quantum Information und liefert sehr viel Hintergrundwissen. Das zweite Werk ist etwas knapper gefasst, legt dafür aber ein großes Augenmerk auf mathematische Exaktheit.

2.1 Endliche Gruppen- und Zahlentheorie

Eine Menge G zusammen mit einer inneren Verknüpfung $\circ$, geschrieben $(G, \circ)$, heißt *Gruppe*, sobald die Verknüpfung assoziativ ist und es sowohl ein neutrales Element e gibt, dass für alle $a \in G$ die Gleichung $a \circ e = e \circ a = a$ erfüllt, als auch zu jedem $a \in G$ ein inverses Element $a^{-1} \in G$ existiert mit $a \circ a^{-1} = a^{-1} \circ a = e$. Ist die Verknüpfung $\circ$ kommutativ, wird $(G, \circ)$ kommutative oder abelsche Gruppe genannt.

Eine Menge R zusammen mit zwei inneren Verknüpfungen $+$ und $\cdot$, geschrieben $(R, +, \cdot)$, heißt *Ring*, genau dann, wenn $(R, +)$ eine kommutative Gruppe ist und $\cdot$ sowohl assoziativ als auch distributiv bezüglich $+$ ist. Ist die Verknüpfung $\cdot$ auch kommutativ, spricht man von einem kommutativen Ring. Enthält R ein neutrales Element bezüglich $\cdot$, Einselement genannt, so spricht man von einem Ring mit Eins. Das neutrale Element bezüglich $+$ wird Nullelement genannt.

Damit bildet die Menge der ganzen Zahlen $\mathbb{Z}$ mit der üblichen Addition und Multiplikation einen kommutativen Ring mit Eins, wobei das Nullelement die Null und das Einselement die Eins ist. Die Einschränkung der Menge der ganzen Zahlen auf die endliche Zahlenmenge $0, 1, ..., n-1$ und die Wahl der Addition modulo n anstelle der üblichen Addition auf $\mathbb{Z}$ liefert einen endlichen kommutativen Ring mit Eins. Im Folgenden wird dieser Ring mit $\mathbb{Z}_n$ bezeichnet. Die Addition modulo n ist wie folgt definiert:

$$a \equiv b \pmod{n} \Leftrightarrow n \text{ teilt } b - a. \tag{1}$$

Man spricht a ist kongruent zu b modulo n. Prinzipiell kann der Ring $\mathbb{Z}_n$ so verstanden werden, als dass die ganzen Zahlen bei null beginnend gezählt werden und sobald n erreicht wird, das Zählen erneut bei null begonnen wird. Mathematisch kann $\mathbb{Z}_n$ auch als Menge der Äquivalenzklassen bezüglich der Kongruenz modulo n betrachtet werden. Das heißt zwei ganze Zahlen a und b werden als gleich betrachtet, sobald sie sich um ein Vielfaches von n unterscheiden.

Für die Erläuterung des Shor-Algorithmus ist der Begriff der *Ordnung* zentral, den wir nun definieren können. Die Ordnung r in einer Gruppe oder einem Ring eines Elements a ist die kleinste Zahl für die gilt $a^r = e$. Mit der *Ordnung von a modulo n* soll im Folgenden die Ordnung r von a im Ring $\mathbb{Z}_n$ bezüglich $\cdot$ bezeichnet werden. Im Allgemeinen ist die Existenz der Ordnung nicht sicher. Unter der Voraussetzung, dass a und n teilerfremd sind, existiert r mit $a^r \equiv 1 \pmod{n}$ aber immer.

Ob zwei Zahlen teilerfremd sind, lässt sich auch über den größten gemeinsamen Teiler (ggT) definieren. Zwei Zahlen a und b sind genau dann teilerfremd, wenn $\text{ggT}(a, b) = 1$ gilt. Der größte gemeinsame Teiler zweier Zahlen lässt sich ohne Kenntnis der Primfaktorzerlegung der beiden Zahlen effizient über den Euklidischen Algorithmus bestimmen, der später noch beschrieben wird.

2.2 Theoretische Informatik

Einige Begriffe der theoretische Informatik sind für die Beschreibung des Shor-Algorithmus zentral; diese sollen hier eingeführt werden. Es soll vor allem geklärt werden, was einen Algorithmus überhaupt ausmacht und wieso der Shor-Algorithmus und Quantencomputer für die theoretische Informatik so interessant sind.

2.2.1 Algorithmen und Komplexitätstheorie

Ein Algorithmus ist im Wesentlichen eine Liste von Instruktionen. Diese Liste von Instruktionen wird schrittweise abgearbeitet und dient schließlich der Lösung einer gezielten Problemstellung. In den meisten Fällen sind für die Lösung eines Problems eine Vielzahl unterschiedlicher Algorithmen bekannt. Dies macht es notwendig über die Qualität verschiedener Algorithmen zur Lösung des gleichen Problems zu urteilen. Ein geeignetes Maß für die Qualität eines Algorithmus ist sicherlich der Bedarf an Ressourcen wie Laufzeit und Speicher. In der Regel ist dieser Bedarf abhängig von der Eingabegröße n des Algorithmus. Die Komplexitätstheorie liefert mit den Komplexitätsklassen ein geeignetes Kriterium zur Bestimmung des ressourcenschonendsten Algorithmus. Im Folgenden wollen wir uns als relevante Ressource auf den Laufzeitaufwand beschränken, das heißt die Zeit, die ein Algorithmus zur Lösung des gestellten Problems benötigt. Aus Gründen der Vergleichbarkeit soll der Laufzeitaufwand nicht von der konkreten Rechenmaschine, die den Algorithmus ausführt, abhängen. Daher wird die Laufzeit in den Einheiten elementarer Rechenoperationen angegeben. In der Komplexitätstheorie wird als Kriterium außerdem nicht die konkrete Laufzeit, sondern die asymptotische Laufzeit - das heißt das Laufzeitverhalten für große n - als Kriterium verwendet. In diesem Kontext findet in der Komplexitätstheorie sehr häufig die Landau-Notation Gebrauch. Diese soll kurz definiert werden: Seien f und g zwei Funktionen $\mathbb{N} \to \mathbb{N}$, dann sind o und $\mathcal{O}$ wie folgt definiert:

$$f \in o(g) \quad \Leftrightarrow \quad \lim_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| = 0, \quad \text{d.h. } f \text{ ist asymptotisch ggü. } g \text{ vernachlässigbar,} \tag{2}$$

$$f \in \mathcal{O}(g) \quad \Leftrightarrow \quad \lim_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| < \infty, \text{ d.h. } g \text{ ist asymptotische obere Schranke von } f. \tag{3}$$

Unter der Angabe von Schrankenfunktionen wie in Definition (3) lassen sich Komplexitätsklassen definieren. Beispiele für typische Komplexitätsklassen sind nach aufsteigendem Aufwand geordnet die Mengen der asymptotisch konstanten sowie der asymptotisch logarithmisch, polynomial und exponentiell wachsenden Funktionen $\mathcal{O}(1)$ sowie $\mathcal{O}(\log n), \mathcal{O}(n^k)$ und $\mathcal{O}(d^n)$ für $k \geq 1$ und $d > 1$.

Die Unterscheidung anhand der asymptotischen Laufzeit mag zunächst verwundern, da einem zum Beispiel eine Halbierung der benötigten Laufzeit als enorme Beschleunigung des Algorithmus erscheinen könnte, während diese Halbierung bei der Unterscheidung nach Komplexitätsklassen keine Veränderung bewirken würde. Die Motivation hinter der Unterscheidung anhand der Asymptotik fußt auf dem Speedup-Theorem. Denn in vereinfachter Form besagt das Speedup-Theorem, dass zu jeder Turingmaschine, die einen Aufwand von $f(n)$ benötigt, eine Turingmaschine gefunden werden kann, die einen Aufwand von

nur $\epsilon f(n)$ benötigt. Dabei kann $\epsilon > 0$ beliebig klein gewählt werden. Dementsprechend würde eine Unterscheidung innerhalb einzelner Komplexitätsklassen irreführend sein, da diese Unterscheidung von der konkreten verwendeten Rechenmaschine abhängig wäre.

In der Praxis ist die Eingabegröße n in der Regel sehr groß. Häufig führt dies dazu, dass Algorithmen mit exponentieller Laufzeitkomplexität aufgrund der langen Rechenzeit praktisch wenig Nutzen haben, während Algorithmen mit polynomialer Laufzeit meist gut zu gebrauchen sind. Daher hat sich die Sprechweise eingebürgert, Algorithmen mit polynomialer oder niedrigerer Laufzeitkomplexität als *effizient* zu bezeichnen, während Algorithmen mit superpolynomialer, häufig exponentieller, Laufzeitkomplexität als *nicht effizient* bezeichnet werden. Dies beinhaltet die Bedeutung des Shor-Algorithmus, der das Problem der Faktorisierung effizient löst, während andere Algorithmen das nicht tun.

2.2.2 Rechnermodelle

Ein Algorithmus, als Rechenvorschrift zur Lösung eines bestimmten Problems, bedarf eines Rechenmodells, das eine Menge möglicher Operationen und Vorschriften sowie einen Speicher, auf dem operiert wird, definiert. Das typische einfache Modell der theoretischen Informatik ist die Turingmaschine. Die Turingmaschine besteht allein aus einem unendlich langem Speicherband und einem Lese-Schreib-Kopf. Das Speicherband besteht aus sequentiell angeordneten Speicherfeldern, die genau ein Zeichen aus einem vorab definierten Alphabet enthalten. Der Lese-Schreib-Kopf wird von einem Programm gesteuert und zeigt immer auf eines der Speicherfelder des Speicherbandes. Eine elementare Operation der Turingmaschine besteht aus dem Lesen des Inhalts des aktuellen Speicherfeldes, der Überschreibung mit einem neuen Wert und einer anschließenden Bewegung um ein oder kein Feld. Durch die Einfachheit des Modells ist es sehr gut mathematisch zu handhaben. Insbesondere stellt die Turingmaschine das Kriterium dar zwischen berechenbaren und nicht-berechenbaren Funktionen zu unterscheiden. Die entscheidende Bedeutung wird in der Church-Turing-These formuliert. Diese besagt

Jede (physikalisch realisierbare) Rechenmaschine kann durch die Turingmaschine simuliert werden.

In anderen Worten besagt die These also, dass aus der Turing-Berechenbarkeit alle anderen Formen von Berechenbarkeit abgeleitet werden können. Insbesondere bedeutet dies, dass es kein Rechenmodell gibt, das prinzipiell mehr berechnen kann als es bereits eine Turingmaschine kann. Es handelt sich bei der Church-Turing-These allerdings nicht um einen mathematisch beweisbaren Satz, da die verwendeten Begriffe - physikalische Realisierbarkeit und Rechenmaschine - nicht ausreichend gut definiert sind. Es lassen sich zwar mathematische Beschreibungen dieser Begriffe finden, eine solche Beschreibung würde aber die Möglichkeit eröffnen ein Rechenmodell zu kreieren, das nicht zu dieser Beschreibung passt, und die These damit abschwächen.[5]

Nicht nur die Berechenbarkeit, sondern auch der benötigte Aufwand eines Algorithmus, hängt vom verwendeten Rechenmodell ab. Mit der erweiterten Churchschen These wird aber folgende quantitative Aussage über den Aufwand von Algorithmen auf unterschiedlichen Rechenmaschinen gemacht:

Jede (physikalisch realisierbare) Rechenmaschine kann durch die Turingmaschine effizient simuliert werden.

Mit effizienter Simulation ist ein polynomialer Ressourcenaufwand gemeint (dies wird im folgenden Kapitel noch genauer erläutert). Bei Gültigkeit der These kann für die Beurteilung der Effizienz eines Algorithmus also ein beliebiges Rechenmodell angenommen werden, da die Turing-Effizienz durch effiziente Simulation des anderen Rechenmodells direkt folgen würde. Die physikalische Realisierbarkeit ist insofern wichtig, als dass Rechenmodelle konstruiert werden können, die die erweiterte Churchsche These verletzen. Diese Modelle können aber in der Regel nicht als physikalisch realisierbar bezeichnet werden.

Die beiden genannten Thesen sind gewissermaßen fundamental in der Berechenbarkeits- und in der Komplexitätstheorie. In beiden Teilgebieten stellen die Thesen die Grundlage dafür da, die Turingmaschine oder auch äquivalente Rechenmodelle, wie zum Beispiel Registermaschinen[18, Kap. 8.6], als universell zu betrachten, wenn es um die Begriffe Berechenbarkeit oder Effizienz von Algorithmen geht. Die Registermaschine ist dabei ein Rechenmodell, das dem Funktionsprinzip heutiger Computer nachempfunden ist. Wegen der Äquivalenz zur Turingmaschine ist ein normaler Computer unter der Voraussetzung eines genügend großen Speichers also in der Lage alles zu berechnen, was prinzipiell berechenbar ist. Wie aber bereits erwähnt, handelt es sich um Thesen deren Gültigkeit nicht sicher ist. So stellt sich die Frage, ob nicht doch Rechenmaschinen konstruiert werden können, die prinzipiell schneller sind oder mehr berechnen können als die bekannten Rechenmodelle. Dies ist der Grund, warum der Quantencomputer und auch der Shor-Algorithmus für die theoretische und schließlich auch für die praktische Informatik so interessant sind. Es gibt mehrere Indizien, die dafür sprechen, dass ein Quantencomputer prinzipiell schneller sein könnte als eine Turingmaschine und eines dieser Indizien ist der Shor-Algorithmus.

2.2.3 Quantenalgorithmen

Ein Quantenalgorithmus ist wie auch ein klassischer Algorithmus eine Rechenvorschrift, das heißt eine Verkettung von Rechenoperationen, die zur Lösung einer gegebenen Problemstellung führt. Nur wird im Falle eines Quantenalgorithmus anstelle einer klassischen Rechenmaschine ein Quantencomputer zur Durchführung der Rechenvorschrift verwendet. Für die Konstruktion eines Algorithmus ändert sich also sowohl die Struktur des verfügbaren Speichers, auf dem der Algorithmus operieren kann, als auch die Menge der verfügbaren Rechenoperationen. Während beim klassischen Computer nur logische Operationen zur Realisierung der Rechenvorschrift verfügbar sind, stehen den Quantenalgorithmen sämtliche unitären Transformationen zur Verfügung (siehe auch die Kapitel 2.3 und 2.5). Außerdem sind Quantenalgorithmen dazu in der Lage bestimmte Berechnungen für mehrere Variablen simultan durchzuführen. Dies wird auch als Quantenparallelismus bezeichnet.

Unter diesen beiden Aspekten betrachtet, scheinen Quantenalgorithmen immense Vorteile gegenüber klassischen Algorithmen zu haben. Allerdings unterliegen diese beiden Vorteile einer großen Einschränkung. Im Gegensatz zum Speicher eines klassischen Computers ist der Speichers eines Quantencomputers nicht ohne weiteres erreichbar. Zur Extraktion von Informationen aus diesem Speicher, der einen quantenmechanischen Zustand darstellt, ist eine quantenmechanische Messung erforderlich. Durch die Messung wird ein Kollaps des Speicherzustands in einen Eigenzustand des Messoperators erzwungen, der zu einem immensen Informationsverlust führt. Dies wird im folgenden Kapitel genauer erläutert.

Damit kommen wir nun zum eigentlichen Sinn dieses Unterkapitels. Gerade wurde geschildert wie sich Vor- und Nachteile eines Quantenalgorithmus durch „plumpe“ Anwendung in etwa aufheben. Das Ziel bei der Konstruktion eines effizienten Quantenalgorithmus muss es nun sein, die vorteilhaften Aspekte überwiegen zu lassen und den Informationsverlust durch die abschließende Messung zu minimieren. Der Informationsverlust ist dadurch bedingt, dass statistisch eines der berechneten Ergebnisse ausgewählt wird und die Informationen über alle anderen Ergebnisse verloren gehen. Das Ziel eines effizienten Verfahrens muss es also sein, die erhaltenen Ergebnisse so zu transformieren, dass sich mit großer Wahrscheinlichkeit ein Ergebnis einstellt, das Rückschluss auf die entscheidenden Informationen zur Lösung der gegebenen Problemstellung zulässt. Dies setzt eine weitreichende Kenntnis über die Problemstellung voraus, sodass die gegebenen Informationen geschickt genutzt werden können, um irrelevante Ergebnisse auszufiltern und das Erhaltenen eines relevanten Ergebnisses wahrscheinlicher zu machen. Genau dies ist es auch, das später im Falle des Algorithmus von Shor zu beobachten ist (siehe auch Kapitel 3.4 über die Quantenphasenschätzung).

2.3 Quantenmechanische Zustände und Messprozesse

Es wird davon ausgegangen, dass die Grundlagen der Quantenmechanik sowohl im mathematischen als auch im physikalischen Sinne bekannt und verstanden sind. Trotzdem wird hier eine kurze Einführung der für das Thema der Arbeit wesentlichen Inhalte gegeben. Dabei sollten auch die später verwendeten Notationen festgelegt werden. Wesentlich sind vor allem die Zeitentwicklung quantenmechanischer Zustände und quantenmechanische Messprozesse.

Beginnen wir zunächst mit den Fundamenten der Quantenmechanik. Diese Fundamente sind als Postulate zu verstehen, auf denen die gesamte Quantentheorie aufbaut. Grundlage für die quantenmechanische Beschreibung eines physikalischen Systems ist der Zustandsraum $\mathcal{H}$ eines Systems. Bei diesem Zustandsraum handelt es sich um einen komplexen Hilbertraum, also einen komplexen Vektorraum mit Skalarprodukt $<,>$, der bezüglich der vom Skalarprodukt induzierten Norm vollständig ist. Der Zustand des Systems wird durch einen Einheitsvektor dieses Zustandsraums eindeutig beschrieben. Dieser Einheitsvektor wird *Zustandsvektor* des Systems genannt. Für Quantencomputer genügt es endlich dimensionale Hilberträume zu betrachten. Da jeder endlich dimensionale Hilbertraum isomorph zu $\mathbb{C}^n$ ist, wobei n der Dimension des Hilbertraumes entspricht, werden in dieser Arbeit nur Hilberträume dieser Form behandelt.

Sei nun $|\Psi\rangle \in \mathcal{H}$ der Zustand des Systems zu einem bestimmten Zeitpunkt t und das betrachtete System sei abgeschlossen, das heißt von der Umgebung isoliert. Der Zustand des Systems $|\Psi'\rangle$ zu einem beliebigen anderen Zeitpunkt t' ist dann gegeben durch eine unitäre Transformation von $|\Psi\rangle$, genauer einen unitären Operator U auf $\mathcal{H}$:

$$|\Psi'\rangle = U\,|\Psi\rangle\,. \tag{4}$$

In der Regel wird dieses Postulat noch genauer spezifiert, nämlich in der Form, dass $U = \exp\left(-\frac{i}{\hbar}H(t'-t)\right)$ gilt. Dabei ist $\hbar$ das Plancksche Wirkungsquantum und H der dem System zugehörige Hamiltonoperator. Der Hamiltonoperator ist eine Observable, die die Energie des Systems misst. Die spezielle Form für U ergibt sich aus der Lösung der zeitabhängigen Schrödingergleichung. In unserem Fall soll aber die allgemeine Form des Postulates genügen. Der Grund hierfür soll genauer erläutert werden: Bei den später zu beschreibenden Systemen handelt es sich um den Speicher unseres Quantencomputers. Dieser soll von der Umgebung genügend gut abgeschirmt sein, sodass ungewollte Wechselwirkungen mit der Umgebung den Zustand des Systems nicht bzw. vernachlässigbar wenig stören. In dieser Hinsicht ist unser System als abgeschlossen zu verstehen und entwickelt sich gemäß dem spezifischeren Postulat. Um mit diesem Speicher aber auch Berechnungen durchführen zu können, müssen gezielte Operationen auf dem Speicher durchführbar sein. Im späteren Verlauf werden wir in diesem Kontext häufig davon sprechen, dass wir einen gewissen Operator auf einen Zustand anwenden. Physikalisch gemeint ist damit, dass wir eine gezielte Wechselwirkung stattfinden lassen, die den Zustand des Speichers verändert.[8] Anstelle der Darstellung dieser Wechselwirkung in Form eines zeitabhängigen Hamilton-Operators und der anschließenden Bestimmung der resultierenden unitären

[8] Das betrachtete System ist also tatsächlich nicht abgeschlossen. Die Wechselwirkung kann aber näherungsweise als eine zeitliche Änderung des Hamiltonoperators aufgefasst werden. In dieser Form kann das System weiterhin als abgeschlossenes System betrachtet werden.

Transformation über die Schrödinger-Gleichung kann auch direkt ein adäquater Operator verwendet werden, der diese unitäre Transformation bewirkt.

In der Quantenmechanik werden physikalisch messbare Größen als Observablen bezeichnet. Im mathematischen Formalismus entspricht das einem hermiteschen Operator auf $\mathcal{H}$. Im Gegensatz zur klassischen Vorstellung stellt die Messung in der Quantenmechanik keine interaktionslose Extraktion von Informationen über das physikalische System dar, sondern bedeutet einen fundamentalen Eingriff in das System.[9] Die Messung verändert den Zustand des Systems im Allgemeinen irreversibel und unstetig. Damit entspricht die zeitliche Entwicklung des Systemzustandes keiner unitären Transformation. Das obige Postulat ist aber nicht verletzt, da es sich, sobald Messungen am System durchgeführt werden, nicht mehr um ein abgeschlossenes System handelt.

Der Messprozess soll nun genauer erläutert werden. Zunächst ist es wichtig zu bemerken, dass die Menge der möglichen Ergebnisse einer Messung einer Observablen ihrem Spektrum, das heißt der Menge ihrer Spektralwerte entspricht.[10] Das Spektrum eines Operators kann im Allgemeinen sowohl kontinuierlich als auch diskret ausfallen. In dieser Arbeit werden aber nur Operatoren behandelt, die diskrete Spektralwerte besitzen. Betrachtungen von Observablen mit diskretem Spektrum sind daher für unsere Zwecke ausreichend. In endlich dimensionalen Vektorräumen stimmen die Begriffe *Spektralwert* und *Eigenwert* überein, weshalb letzterer für folgende Betrachtungen genügt.

Es soll nun mit der Beschreibung der quantenmechanischen Messung begonnen werden. Dazu bezeichne $|\Psi\rangle$ den Zustand des Systems vor der Messung und $\mathcal{A}$ die zu messende physikalische Größe mit einer zugehörigen Observablen A mit diskretem Spektrum. Der Einfachheit halber seien die Eigenwerte der Observablen zunächst nicht entartet. Dann definiert A eine orthonormale Basis aus Eigenvektoren $|\varphi_j\rangle$ zu den einfachen Eigenwerten a_j von A. Der Zustand vor der Messung kann in dieser Basis als $|\Psi\rangle = \sum_j c_j |\varphi_j\rangle$ mit $c_j = \langle\Psi|\varphi_j\rangle$ dargestellt werden. Die Messung der Observablen A liefert als Messwert einen der Eigenwerte von A. Der Zustand des Systems geht von dem reinen Zustand $|\Psi\rangle$ über in ein Gemisch aus den Eigenzuständen $|\varphi_j\rangle$ mit den Wahrscheinlichkeiten $p_j = |c_j|^2 = |\langle\Psi|\varphi_j\rangle|^2$. Ein tatsächliches Abgreifen des Messwertes liefert also $\mathcal{A} = a_j$ mit einer Wahrscheinlichkeit von p_j. Dementsprechend kann die zu messende Größe als eine Zufallsvariable mit der Verteilung $P(\mathcal{A} = a_j) = p_j$ und der Messung als ihre Realisierung aufgefasst werden. Ohne die Einschränkung auf nicht-entartete Observablen sind die Eigenräume zu einem Eigenwert im Allgemeinen mehrdimensional. Zu einem Eigenwert a_j kann aber immer eine orthonormale Basis aus Eigenvektoren $|\varphi_j^i\rangle$ gefunden werden mit $1 \leq i \leq g_j$, wobei g_j die Vielfachheit des Eigenwertes a_j angibt. Ganz analog zur obigen Beschreibung findet man $p_j = \sum_{i=1}^{g_j} |\langle\Psi|\varphi_j^i\rangle|^2$. Der Zustand nach der Messung entspricht der Projektion des vorherigen Zustandes auf den dem Eigenwert zugehörigen Eigenraum.

[9] ..., sofern sich das System nicht bereits in einem Eigenzustand der zu messenden Observablen befindet.

[10] Der Grund hierfür ist die Überlegung: Wird eine Observable eines Systems direkt nach einer erfolgten Messung erneut gemessen, so wird erwartet, das gleiche Messergebnis zu erhalten. Die Menge der Messwerte, die genau diese Bedingung erfüllen, entspricht dem Spektrum der Observablen.

Unter Verwendung der Projektionsoperatoren $P_j = \sum_{i=1}^{g_j} |\varphi_j^i\rangle \langle\varphi_j^i|$ lässt sich der Messprozess also wie folgt formalisieren:

Messung einer Observablen mit diskretem Spektrum
Sei $\mathcal{A}$ die zu messende physikalische Größe mit einer zugehörigen Observablen A, die ein diskretes Spektrum besitzt. Die Eigenwerte des Operators seien a_j mit zugehörigen Eigenzuständen $|\varphi_j^i\rangle$ für $1 \leq i \leq g_j$. Definiere Projektionsoperatoren $P_j = \sum_{i=1}^{g_j} |\varphi_j^i\rangle \langle\varphi_j^i|$.

Präparation: Das System befinde sich vor der Messung in einem Zustand $|\Psi\rangle$. In der Eigenbasis gilt

$$|\Psi\rangle = \sum_j \sum_{i=1}^{g_j} c_j^i |\varphi_j^i\rangle \quad \text{mit} \quad c_j^i = \langle\Psi|\varphi_j^i\rangle . \tag{5}$$

Messung: Die Messung der Observablen liefert den Eigenwert a_j mit einer Wahrscheinlichkeit von

$$p_j = \langle\Psi|P_j|\Psi\rangle = || P_j|\Psi\rangle ||^2 = \sum_{i=1}^{g_j} | \langle\Psi|\varphi_j^i\rangle |^2. \tag{6}$$

Der Zustand nach der Messung ist

$$|\Psi'\rangle = \frac{P_j|\Psi\rangle}{|| P_j|\Psi\rangle ||} = \frac{1}{\sqrt{p_j}} \sum_{i=1}^{g_j} c_j^i |\varphi_j^i\rangle . \tag{7}$$

Dabei ist $|| \cdot ||$ die vom Skalarprodukt induzierte Norm des Hilbertraumes.

Im Folgenden werden wir den Messprozess häufig als Messung bezüglich einer orthonormalen Basis verstehen. Die Messung bzgl. einer Basis verläuft prinzipiell analog zur obigen Definition. Der Unterschied ist, dass wir von einer orthonormalen Basis $\{|\varphi_j\rangle\}$ ausgehen und diese die zu messende Observable A definiert, sodass die Observable die einzelnen Basiszustände bei der Messung unterscheidet. Für die Messung von Qubits und Multi-Qubits bietet es sich an $A = \sum_j j |\varphi_j\rangle \langle\varphi_j|$ zu wählen, sodass jedem Basisvektor $|\varphi_j\rangle$ bei einer Messung der Eigenwert j zugeordnet wird.

Als einfaches Beispiel soll der zweidimensionale Zustandsraum $\mathcal{H} \cong \mathbb{C}^2$ eines Spin-1/2-Teilchens dienen. Dieser wird allein von den zwei Zuständen *Spin-up* ($\uparrow$) und *Spin-down* ($\downarrow$) erzeugt. Da die beiden Zustände eine Orthonormalbasis formen, lassen sich für jedes $|\Psi\rangle \in \mathcal{H}$ zwei Konstanten $\alpha, \beta \in \mathbb{C}$ finden mit:

$$|\Psi\rangle = \alpha |\uparrow\rangle + \beta |\downarrow\rangle \quad \text{und} \quad \langle\Psi|\Psi\rangle = |\alpha|^2 + |\beta|^2 = 1 \text{ (Normierung)}. \tag{8}$$

Das Spin-1/2-Teilchen kann sich also in einer Superposition der beiden Zustände $|\uparrow\rangle$ und $|\downarrow\rangle$ befinden. Dies ist der klassischen Vorstellung eines Systems zuwider, nach der ein System sich allein in einem der Basiszustände befinden kann. Im klassischen Sinne könnte sich das Teilchen also nur entweder im *Spin-up*-Zustand oder im *Spin-down*-Zustand befinden.

Für unser quantenmechanisches System sind aber auch Überlagerungen dieser Entweder-Oder-Zustände möglich. Erst eine quantenmechanische Messung legt fest, ob sich das System im Zustand $|\uparrow\rangle$ oder $|\downarrow\rangle$ befindet. Es soll nun eine quantenmechanische Messung durchgeführt werden. Das Ergebnis dieser Messung soll uns die Spinrichtung liefern, also *Spin-up* oder *Spin-down*. Die Messung des Spins liefert dann $\uparrow$ mit der Wahrscheinlichkeit $|\alpha|^2$ und $\downarrow$ mit der Wahrscheinlichkeit $|\beta|^2$. Dabei ist der Zustand nach Durchführung der Messung dann entweder $|\uparrow\rangle$ oder $|\downarrow\rangle$.

Im Sinne der Informatik lassen sich diese beiden verwendeten Zustände $\uparrow$ und $\downarrow$ auch als die 0 und die 1 eines Bits verstehen. Wir können dann $|\uparrow\rangle = |0\rangle$ und $|\downarrow\rangle = |1\rangle$ definieren und die Messung als Messung bezüglich der Basis $\{|0\rangle, |1\rangle\}$ verstehen. Aus dieser Zuordnung ergibt sich, wie im folgenden Kapitel beschrieben, die Definition des quantenmechanischen Pendants zum klassischen Bit, dem Qubit.

2.4 Quantum Computation - Qubits und Quantenregister

Zur Behandlung des Shor-Algorithmus soll zunächst ein kurzer Einblick in die Idee und die Grundlagen eines Quantencomputers gegeben werden. Wie der klassische Computer besitzt der Quantencomputer einen Speicher. Der Speicher eines klassischen Computers ist zusammengesetzt aus einzelnen Bits. Ein Bit kann die Zustände 0 und 1 annehmen. Bei einer Zusammenschaltung mehrerer Bits zu einem Register lassen sich Zahlen und komplexere Informationen darstellen. So kann ein Register aus acht Bits beispielsweise die Zahlen von 0 bis $2^8 - 1 = 255$ in Binärschreibweise darstellen. Um nun den Computer als Rechenmaschine nutzen zu können, muss der Speicher in irgendeiner Form modifiziert werden können. Dazu dienen elementare Rechenoperationen (beispielsweise logische Gatter wie AND, OR, XOR und NAND). Durch die Hintereinanderschaltung vieler dieser elementaren Rechenoperationen lassen sich schließlich nahezu beliebig komplexe Rechenoperationen realisieren. Solche komplexen Rechenoperationen werden auch Algorithmen genannt. Der grundsätzliche Aufbau des Quantencomputers ist dem des klassischen Computers sehr ähnlich. Auch der Quantencomputer besteht aus einem Speicher, der durch elementare Operationen, sogenannte Quantengatter, modifiziert werden kann. In Analogie zum klassischen Computer wird der Grundbaustein des Quantenspeichers als Qubit bezeichnet. Ein Qubit ist das kleinstmögliche quantenmechanische System. Es hat daher einen zweidimensionalen Zustandsraum $\mathcal{H}$ und verhält sich genauso wie das System eines Spin-1/2-Teilchens. Unter der Voraussetzung einer Orthonormalbasis $\{|0\rangle, |1\rangle\}$ lässt sich der Zustand $|\Psi\rangle$ eines Qubits schreiben als

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad \text{mit} \quad \alpha, \beta \in \mathbb{C} \quad \text{und} \quad |\alpha|^2 + |\beta|^2 = 1 \text{ (Normierung)}. \tag{9}$$

Die Basiszustände $|0\rangle$ und $|1\rangle$ erinnern direkt an die möglichen Zustände des klassischen Bits . Im Unterschied zum klassischen Bit kann das Qubit nicht nur die reinen Zustände $|0\rangle$ oder $|1\rangle$ annehmen, sondern auch Superpositionen der beiden Zustände. Erst eine Messung bzgl. $\{|0\rangle, |1\rangle\}$ führt zu einer Zuordnung im Sinne des klassischen Bits.

Basierend auf der Definition eines einzelnen Qubits kann nun analog zum klassischen Register ein Quantenregister oder auch Multi-Qubit definiert werden. Ein solches Multi-Qubit ist eine Komposition mehrere einzelner Qubits, sodass diese als ein zusammengesetztes physikalisches System behandelt werden können. Der Zustandsraum $\mathcal{H}^{\otimes n} := \otimes_{i=1}^{n} \mathcal{H}$

des Multi-Qubits ist das Tensorprodukt der Zustandsräume der einzelnen Qubits $\mathcal{H}$. Ein beispielhafter Zustand $|\Psi\rangle$ eines Multi-Qubits aus zwei einzelnen Qubits kann der folgende sein:

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|0\rangle_2 + |3\rangle_2). \tag{10}$$

Die letzteren beiden Schreibweisen sind Kurzformen der ersten Form. In der ersten Kurzform werden die Zustände einfach hintereinander geschrieben (also bspw. $|1\rangle \otimes |1\rangle \otimes |0\rangle = |110\rangle$). In der zweiten Kurzform wird eine Dezimalzahl zur Charakterisierung der Binärfolge aus Nullen und Einsen verwendet (also bspw. $|110\rangle = |6\rangle_3$). Um die Eindeutigkeit der Notation beizubehalten muss der zweiten Kurzform ein Index beigefügt werden, der die Stellenzahl des Multi-Qubits angibt. Im weiteren Verlauf werden Zustände hauptsächlich in einer der beiden Kurzformen angegeben.

Ist $\mathcal{H}$ nun wieder der bekannte zweidimensionale Hilbertraum, dann bezeichnen wir die orthonormale Basis $\{|0\rangle_n, |1\rangle_n, ..., |2^n - 1\rangle_n\}$ als die Standardbasis von $\mathcal{H}^{\otimes n}$. Später werden vor allem Messungen bezüglich dieser Basis interessant sein. Diese entspricht einer Messung bezüglich der Observablen $M := \sum_{j=0}^{2^n-1} j \cdot M_j$ mit $M_j := |j\rangle_n \langle j|_n$. Ist das Multi-Qubit im Zustand $|\Psi\rangle = \sum_{j=0}^{2^n-1} c_j |j\rangle_n \in \mathcal{H}^{\otimes n}$ präpariert, so ist das Ergebnis der Messung mit einer Wahrscheinlichkeit von $p_j = |c_j|^2$ gegeben durch j und der Zustand des Systems nach der Messung ist

$$\frac{M_j |\Psi\rangle}{||M_j |\Psi\rangle||} \cong |j\rangle_n . \tag{11}$$

Dabei steht $a \cong b$ dafür, dass sich a und b nur um eine imaginäre Phase unterscheiden. Man kann die Messung von Qubitregistern bezüglich der Standardbasis also als stochastische Zuordnung eines Quantenzustandes auf klassische Bits auffassen.

Auch möglich sind partielle Messungen bezüglich einer Basis. Wir wählen die Standardbasis und präparieren $|\Psi\rangle$ wie zuvor. Sei nun $1 \leq p < n$ und $q := n-p$. Ein partielle Messung der ersten p Qubits bezüglich der Standardbasis entspricht nun einer Messung bezüglich der Observablen $M := \sum_{j=0}^{2^p-1} j \cdot M_j$ mit $M_j := \sum_{k=0}^{2^q-1} \left(|j\rangle_p \otimes |k\rangle_q\right)\left(\langle j|_p \otimes \langle k|_q\right)$. Nach der Messung befindet sich das Multi-Qubit mit einer Wahrscheinlichkeit $p_j = \sum_{k=0}^{2^q-1} |c_{j\cdot 2^q+k}|^2$ im Zustand

$$|j\rangle_p \otimes \frac{1}{\sqrt{p_j}} \sum_{k=0}^{2^q-1} c_{j\cdot 2^q+k} |k\rangle_q . \tag{12}$$

Die partielle Messung bezüglich einer Basis wird vor allem für Quantenspeicher benötigt, die aus mehreren Registern zusammengesetzt sind. In der Regel wird nur auf einem Register gleichzeitig gearbeitet. Eine Messung von einem dieser Register stellt eine partielle Messung dar. Nach Gleichung (12) werden die anderen Register durch diese partielle Messung nicht beeinflusst.

2.5 Quantum Computation - Quantengatter

Um den Speicher für Berechnungen nutzen zu können, werden nun Operationen benötigt, die den Speicher verändern können. Wie eingangs erwähnt, werden diese Operationen in Analogie zu den Gattern des klassischen Computers Quantengatter genannt. Mathematisch betrachtet ist ein Quantengatter nichts anderes als ein unitärer Operator auf $\mathcal{H}^{\otimes n}$. Das heißt für ein Quantengatter U existiert der zu U inverse Operator U^{-1} und es gilt $U^{-1} = U^\dagger$. Dabei bezeichnet $U^\dagger$ den zu U adjungierten Operator. Damit sind jegliche Anwendungen von Quantengattern (Messungen ausgenommen) invertierbar, das heißt reversibel. Dies ist ein wesentlicher Unterschied zu klassischen Computern, die in aller Regel irreversibel arbeiten.

Quantenmechanische Operatoren sind außerdem linear. Damit ergibt sich für klassische Gatter eine natürliche Fortsetzung zu einem Quantengatter. Denn klassische Gatter sind durch ihr Verhalten bei Eingabe der Zustände $|0\rangle$ und $|1\rangle$ definiert. Die lineare Fortsetzung dieser Definition auf $\mathcal{H}$ ergibt direkt einen unitären Operator, der als quantenmechanisches Pendant zum klassischen Gatter angesehen werden kann.[11]

2.5.1 Elementare Gatter für ein Qubit

Die Funktionsweise solcher Quantengatter soll zunächst durch einige Beispiele veranschaulicht werden. Beginnen wir zunächst mit dem einfachsten Fall eines 1-Qubit-Gatters. In der Standardbasis, d.h. $\{|0\rangle, |1\rangle\}$, wird der zugehörige Operator durch eine (komplexe) unitäre 2×2-Matrix dargestellt.

Beispiel 1 (X-Gatter) Das X-Gatter entspricht der Quantenversion des klassischen NOT-Gatters. Das heißt, es gilt

$$X|0\rangle = |1\rangle \quad \text{und} \quad X|1\rangle = |0\rangle \quad \text{bzw.} \quad X(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle. \tag{13}$$

In der Standardbasis kann X also durch

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{14}$$

dargestellt werden.

Beispiel 2 (Phasengatter) Das Phasengatter oder auch S-Gatter führt zu einem Phasenfaktor von i bzw. $\exp(i\pi/2)$ zwischen den Eingaben $|0\rangle$ und $|1\rangle$. Es gilt

$$S|0\rangle = |0\rangle \quad \text{und} \quad S|1\rangle = i|1\rangle. \tag{15}$$

In der Standardbasis kann S durch

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{16}$$

dargestellt werden.

[11] Tatsächlich ist diese Aussage nicht ganz exakt. Generell kann nur ein reversibles klassisches Gatter in dieser Form zu einem Quantengatter fortgesetzt werden, da Quantengatter automatisch reversibel sind. Allerdings lässt sich jedes irreversible klassische Gatter zu einem reversiblen klassischen Gatter erweitern und dieses kann dann durch ein Quantengatter simuliert werden. In dem Sinne hat jedes klassische Gatter ein quantenmechanisches Pendant.

Beispiel 3 ($\pi/8$-Gatter) Ein weiteres Beispiel ist das $\pi/8$- oder auch T-Gatter. Für das $\pi/8$-Gatter gilt

$$P\left|0\right\rangle = \left|0\right\rangle \quad \text{und} \quad P\left|1\right\rangle = \exp\left(i\pi/4\right)\left|1\right\rangle. \tag{17}$$

In der Standardbasis kann P daher durch

$$H = \begin{pmatrix} 1 & 0 \\ 0 & \exp\left(i\pi/4\right) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{\sqrt{2}}(1+i) \end{pmatrix} \tag{18}$$

dargestellt werden. Die Bedeutung des Gatters wird bei der Behandlung universeller Quantengatter in Kapitel 2.5.4 deutlich. Die Bezeichnung des Gatters ist historisch bedingt und wird ersichtlich bei Multiplikation mit einem Phasenfaktor $\exp\left(-i\pi/8\right)$.

Beispiel 4 (Hadamard-Gatter) Als drittes Beispiel soll das Hadamard-Gatter dienen. Dieses Gatter hat eine herausragende Bedeutung im Bereich Quantum Computation und findet auch später Anwendung im Shor-Algorithmus. Doch zunächst die Definition. Sei H der zugehörige Operator, dann gilt

$$H\left|0\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle) \quad \text{und} \quad H\left|1\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle). \tag{19}$$

In der Standardbasis kann H also durch

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{20}$$

dargestellt werden. Die Bedeutung von H wird beim Übergang zu n-Qubit-Gattern und später bei der Behandlung universeller Quantengatter deutlich.

2.5.2 Anwendung von 1-Qubit-Gattern auf Multi-Qubits

Über das Tensorprodukt können 1-Qubit-Gatter auch auf Multi-Qubits angewandt werden. Grundlage für diese Art der Konstruktion ist, dass das Tensorprodukt zweier unitärer Operatoren wieder unitär auf dem Produktraum ist. Es bezeichne I den Identitätsoperator auf einem Qubit. Dann ist

$$H_i := \underbrace{I \otimes ... \otimes I}_{(n-i-1)\text{-fach}} \otimes H \otimes \underbrace{I \otimes ... \otimes I}_{i\text{-fach}} \tag{21}$$

das 1-Qubit-Hadamard-Gatter, welches auf das i-te Qubit angewandt wird. Eine andere Form 1-Qubit-Gatter auf Multi-Qubit-Gatter anzuwenden ist, sie parallel auf jedes Qubit wirken zu lassen. Als wichtiges Beispiel ist mit $H^{\otimes n} := \bigotimes_{k=1}^{n} H$ das n-fache Hadamard-Gatter zu nennen, welches der parallelen Anwendung des einfachen Hadamard-Gatters auf jedes einzelne Qubit des Registers entspricht. Für das mehrfache Hadamard-Gatter gilt

$$H^{\otimes n}\left|0\right\rangle_n = \left(H\left|0\right\rangle\right)^{\otimes n} = \left(\frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle)\right)^{\otimes n} = 2^{-\frac{n}{2}}(\left|0\right\rangle + \left|1\right\rangle)^{\otimes n} = 2^{-\frac{n}{2}}\sum_{x=0}^{2^n-1}\left|x\right\rangle_n. \tag{22}$$

Damit wird einer der Nutzen des Hadamard-Gatters ersichtlich. Ist ein Register zu Beginn im Zustand $|0\rangle_n$ präpariert, so lässt sich durch einfaches Anwenden von $H^{\otimes n}$ eine gleichmäßige Superposition aller Basiszustände von $|0\rangle_n$ bis $|2^n-1\rangle_n$ erreichen. Diese Art der Präparation wird häufig zu Beginn eines Algorithmus durchgeführt. Denn ausgehend von jenem Zustand kann der große Vorteil des Quantencomputers, das parallelisierte Rechnen, ausgespielt werden. Die anschließenden Berechnungen werden dann für jedes $x \in \{0, ..., 2^n - 1\}$ durchgeführt.

2.5.3 Kontrollierte Gatter

Über das Tensorprodukt konnte die Anwendung elementarer Gatter auf Multi-Qubits realisiert werden. Diese Operationen werden jedoch weiterhin für jedes Qubit einzeln durchgeführt. Es fehlen noch Gatter die Interaktionen zwischen einzelnen Qubits zulassen. Der einfachste Fall eines solchen Gatters verwendet zwei Qubits. Dabei wird eines der Qubits als das kontrollierende angesehen. Dieses steuert die Anwendung des Operators auf das andere Qubit. Eine solche kontrollierte Operation ist beispielsweise durch das cNOT-Gatter, eine kontrollierte Not-Operation, gegeben:

Beispiel 5 (cNOT-Gatter für zwei Qubits) Das cNOT-Gatter für zwei Qubits ist für alle $x, y \in \{0, 1\}$ definiert durch

$$\text{cNOT}\,|x\rangle\,|y\rangle = |x\rangle\,|y \oplus x\rangle\,. \tag{23}$$

Dabei stellt $\oplus$ die bitweise Addition modulo 2 bzw. die XOR-Operation dar. Im Fall $x = 0$ bleibt das zweite Qubit unverändert. Im Fall $x = 1$ wird das zweite Qubit invertiert. Damit realisiert cNOT ein bedingtes NOT-Gatter, das nur im Falle von $x = 1$ aktiviert wird. Die Matrixdarstellung des cNOT-Gatters lautet

$$\text{cNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{24}$$

Ist ein beliebiger Operator U gegeben, der auf ein n-Qubit-Register wirkt, dann soll folgende Notation eingeführt werden:

$$\text{c}_j U\,|x_1 x_2 ... x_m\rangle\,|\Psi\rangle_n := \begin{cases} |x_1 x_2 ... x_m\rangle \quad |\Psi\rangle_n & \text{für } x_j = 0, \\ |x_1 x_2 ... x_m\rangle\, U\,|\Psi\rangle_n & \text{für } x_j = 1 \end{cases} \tag{25}$$

$$= |x_1 x_2 ... x_m\rangle\, U^{x_j}\,|\Psi\rangle_n\,. \tag{26}$$

Das Gatter $\text{c}_j U$ ist also definiert als die durch das j-te Qubit des ersten Registers kontrollierte Anwendung von U auf das zweite Register. Ausdrücken lässt sich das durch die Anwendung von U^{x_j}. Ist die Angabe des kontrollierenden Qubits nicht wichtig, kann der Index auch weggelassen werden wie zum Beispiel beim cNOT-Gatter.

2.5.4 Universelle Quantengatter

Im klassischen Sinne wird eine Menge logischer Gatter als *universell* angesehen, wenn jede beliebige logische Funktion durch eine Hintereinanderausführung von Gattern aus jener Menge implementiert werden kann. Dieser Begriff lässt sich auch auf Quantengatter übertragen. So heißt eine Menge von Quantengattern *universell*, wenn jede unitäre Transformation mit beliebiger Genauigkeit[12] durch Hintereinanderausführung von Quantengattern aus jener Menge implementiert werden kann. Anders gesagt, lässt sich dann jede unitäre Transformation approximativ durch ein Produkt universeller Operatoren darstellen. Der Vorteil einer solchen Menge universeller Gatter ist, dass die Realisierung von beliebigen Rechenoperationen auf die Realisierung von einer geringeren Anzahl von Rechenoperationen zurückgeführt werden kann. Im Idealfall werden nur endlich viele Gatter benötigt. Im klassischen Sinne bilden bereits AND und NOT oder sogar nur NAND eine Familie universeller Gatter. Das heißt, dass jede logische Funktion durch eine Zusammenschaltung von NAND-Gattern realisiert werden kann. Es stellt sich die Frage, welche Mengen von Quantengattern universell sind.

Zunächst sollte es klar sein, dass eine Menge von Quantengattern nur dann universell sein kann, wenn sie Quantengatter enthält, die Wechselwirkungen zwischen den Qubits zulassen. Im einfachsten Fall kann ein Zwei-Qubit-Gatter eine solche Wechselwirkung realisieren. Es kann tatsächlich gezeigt werden, dass die Menge der Zwei-Qubit-Gatter universell ist.[19] Darauf aufbauend kann gezeigt werden, dass die Menge der Ein-Qubit-Gatter zusammen mit dem cNOT-Gatter universell ist.[20] Schließlich kann auch gezeigt werden, dass das Hadamard- und das $\pi/8$-Gatter universell für 1-Qubit-Operationen und damit cNOT-, Hadamard- und $\pi/8$-Gatter universell für beliebige Quantengatter sind.[21]

Die Kenntnis einer solchen universellen Familie von Quantengattern kann die experimentelle Realisierung eines Quantencomputers erheblich vereinfachen, indem sie die Konstruktion auf die Implementierung dieser kleinen Menge von Gattern reduziert. Allerdings ist diese Reduktion nur dann sinnvoll, wenn die Approximation der benötigten, am besten beliebiger, unitärer Transformationen durch diese universelle Familie effizient durchgeführt werden kann. Andernfalls geht durch die Beschränkung auf die universelle Familie der Geschwindigkeitsvorteil des Quantencomputers verloren. Effizient meint hier, dass der asmyptotische Aufwand an Gattern für die Approximation einer beliebigen unitären Transformationen mit Genauigkeit ϵ höchstens polynomial in ϵ^{-1} ist. Genau diese fehlende Eigenschaft liefert das Solovay-Kitaev-Theorem (SK-Theorem) oder auch der Solovay-Kitaev-Algorithmus für beliebige 1-Qubit-Gatter.[13] Im Wesentlichen ist die Aussage des Theorems, dass für ein beliebiges $\epsilon > 0$ - eine geeignete[14] Familie universeller Quantengatter $\mathcal{G}$ vorausgesetzt - jedes beliebige 1-Qubit-Gatter mit einer Folge von $\mathcal{O}(\log^c(1/\epsilon))$

[12] Die Genauigkeit wird in der Regel als die Operatornorm der Differenz von Approximation und der zu approximierenden Transformation definiert.

[13] Das Theorem wurde sowohl 1995 von Solovay in einem unveröffentlichtem Manuskript als auch unabhängig davon von Kitaev bewiesen. Eine Beweisskizze von Kitaev in russischer Sprache findet sich in [22]. Außerdem existiert eine Aufzeichnung eines Vortrages, in dem Solovay seinen Beweis erläutert.[23] Eine didaktische Aufbereitung wurde in [24] veröffentlicht.

[14] Hinreichend ist eine Familie universeller Operatoren, die abgeschlossen bezüglich der Inversion ist. Es ist eine offene Fragestellung, ob diese Bedingung für die Gültigkeit eines Theorems in der Art des Solovoy-Kitaev-Theorems notwendig ist.

Gattern aus $\mathcal{G}$ mit einer Genauigkeit von ϵ approximiert werden kann. Bei c handelt es sich um eine Konstante, deren niedrigster (konstruktiv bewiesener) Wert bei $c \approx 2$ liegt. Tatsächlich stellt sich heraus, dass die Approximation beliebiger 1-Qubit-Gatter - erneut geeignete Familien universeller Quantengatter vorausgesetzt - nicht nur effizient und mit beliebiger Genauigkeit stattfinden kann, sondern zusätzlich auch fehlertolerant. Fehlertoleranz meint, dass sich Fehler in einer längeren Sequenz von fehlerbehafteten Gattern nicht aufschaukeln, sondern unter einer gewissen Grenze bleiben. Da in diesem Dokument generell von idealen nicht fehlerbehafteten Operatoren ausgegangen wird, wird hierauf aber nicht weiter eingegangen.

Die in diesem Kapitel dargestellten Resultate, insbesondere das SK-Theorem, bilden somit ein wichtiges Fundament im Bereich der Quantum Computation, da sie sowohl die experimentelle als auch die theoretische Untersuchung von quantenrechnenden Systemen erheblich vereinfachen, indem sie die Untersuchung von der gesamten Menge unitärerer Transformationen auf eine „Handvoll“ von Gattern beschränkt.

3 Detaillierte Erläuterung des Algorithmus

In dieser Sektion soll nun der vollständige Ablauf des Shor-Algorithmus beschrieben und erläutert werden. Der Fokus liegt dabei auf dem Teil des Algorithmus, der nicht-klassisch ist, i.e. einen Quantencomputer zur Berechnung benötigt. Bevor der Algorithmus an sich erläutert wird, sollte zunächst das zu lösende Problem beschrieben bzw. definiert werden. Die Primfaktorzerlegung einer natürlichen Zahl n ist die eindeutige Zerlegung der Zahl in ein Produkt von Potenzen paarweise verschiedener Primzahlen. Es gibt also für jedes $n \in \mathbb{N}$ ein $m \in \mathbb{N}$ sowie natürliche Zahlen $\alpha_1, ..., \alpha_m$ und paarweise verschiedene Primzahlen $p_1, ..., p_m$ mit $n = p_1^{\alpha_1} \cdot ... \cdot p_m^{\alpha_m}$. Das Problem der Primfaktorzerlegung ist es diese Zerlegung zu finden.

Das von Shor vorgeschlagene Verfahren macht sich nun das grundlegende Teile-und-Herrsche-Prinzip der Informatik zunutze und zerlegt das Primfaktorzerlegungsproblem schrittweise in einfacher zu lösende Probleme. Das Faktorisierungsproblem ist dem Problem der Primfaktorzerlegung untergeordnet, denn es beschränkt sich allein darauf, einen nichttrivialen Teiler der Zahl n zu finden. Die Lösung des Faktorisierungsproblems in Verbindung mit einem Primzahltest genügt, um das Problem der Primfaktorzerlegung mit logarithmischem Aufwand (bezogen auf die Anzahl der Ausführungen des Faktorisierungsverfahrens) zu lösen. Denn das Faktorisierungsverfahren liefert eine Faktorisierung $n = ab$. Die beiden Faktoren a und b können wiederum faktorisiert werden. Die entstehenden Faktoren werden solange weiter faktorisiert, bis alle Faktoren prim sind. Der Shor-Algorithmus beschränkt sich daher auf die Lösung des Faktorisierungsproblems.

Auf die erste Reduktion vom Problem der Primfaktorzerlegung auf das Problem der Faktorisierung folgt eine Reduktion des Faktorisierungsproblems auf ein Ordnungsbestimmungsproblem, das heißt das Problem die Ordnung einer natürlichen Zahl a modulo n zu bestimmen. Allerdings ist auch das Ordnungsbestimmungsproblem klassisch nur mit exponentiellem Aufwand lösbar. Das Problem wird daher weiter reduziert, sodass es durch Quantenphasenschätzung gelöst werden kann. Bei der zugrundeliegenden Quantenphasenschätzung handelt es sich nun nicht mehr um ein klassisches Verfahren, sondern um ein Quantenverfahren. Wichtig für die Effizienz des Gesamtverfahrens ist nun einerseits die Effizienz der Reduktionen und andererseits die Effizienz des resultierenden Problems, in unserem Fall der Quantenphasenschätzung. Die genannten Reduktionen können, wie später gezeigt wird, effizient durchgeführt werden, das heißt die Schritte zur Reduktion zeigen einen maximal polynomialen Aufwand. Unter der Voraussetzung eines Quantencomputers kann auch die Quantenphasenschätzung effizient realisiert werden. Damit ergibt sich insgesamt ein Algorithmus mit polynomialen Aufwand. Eine genauere Analyse des Aufwands erfolgt in Kapitel 4, nachdem der Ablauf des Algorithmus erläutert wurde.

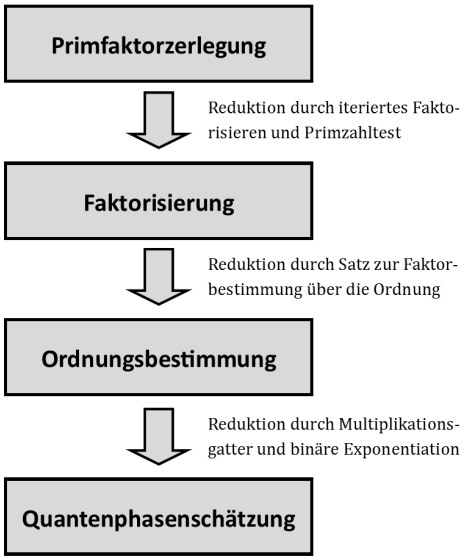

Abbildung 1: Darstellung der vom Shor-Algorithmus durchgeführten Problemreduktionen.

Die Einordnung der einzelnen Probleme, das heißt der Ablauf der genannten Reduktionen ist in Abbildung 3 graphisch dargestellt.

3.1 Reduktion des Faktorisierungsproblems

In diesem Kapitel soll zunächst die Lösung des Faktorisierungsproblems auf die Lösung des Ordnungsbestimmungsproblems, das heißt dem Bestimmen einer Ordnung modulo n, zurückgeführt werden. Diesem Reduktionsschritt liegt ein Satz aus der Zahlentheorie zugrunde. Denn dem Satz zufolge können für bestimmte natürliche Zahlen a mit $1 < a < n$ zwei nichttriviale Teiler von n aus a konstruiert werden. Dafür muss allerdings die Ordnung von a modulo n bekannt sein. Ist ein passendes a gefunden und die Ordnung r bekannt, so erfordert die Konstruktion dieser Teiler kaum Aufwand (siehe Kapitel 4). In diesem Fall wäre das Faktorisierungsproblem also gelöst.

Zunächst das Theorem aus der Zahlentheorie:

Satz 1 (Faktorbestimmung über die Ordnung)
Es seien $a, n \in \mathbb{N}$ mit $1 < a < n$ und $ggT(a, n) = 1$. Falls die Ordnung r von a modulo n gerade ist und falls $a^{\frac{r}{2}} \not\equiv -1 \pmod{n}$ gilt, dann sind

$$1 < ggT(a^{\frac{r}{2}} + 1, n) < n \text{ und } 1 < ggT(a^{\frac{r}{2}} - 1, n) < n$$

zwei nichttriviale Teiler von n.

Beweis (nach [17]) Seien a,n und r wie im Satz angegeben. Da r die Ordnung von a modulo n und gerade ist, gilt:

$$(a^{\frac{r}{2}} + 1)(a^{\frac{r}{2}} - 1) = a^r - 1 \equiv 0 \pmod{n}.$$

Da n ein Teiler von $a^r - 1$ ist, und $a^{\frac{r}{2}} + 1$ und $a^{\frac{r}{2}} - 1$ die Zahl $a^r - 1$ faktorisieren, hat einer der beiden Faktoren einen gemeinsamen Teiler mit n. Also gilt

$$\text{ggT}(a^{\frac{r}{2}} + 1, n) > 1 \quad \text{oder} \quad \text{ggT}(a^{\frac{r}{2}} - 1, n) > 1.$$

Aus der Voraussetzung $a^{\frac{r}{2}} \not\equiv -1 \pmod{n}$ folgt $a^{\frac{r}{2}} + 1 \not\equiv 0 \pmod{n}$. Es gilt außerdem $a^{\frac{r}{2}} \not\equiv 1 \pmod{n}$ und damit $a^{\frac{r}{2}} - 1 \not\equiv 0 \pmod{n}$, da sonst bereits $\frac{r}{2}$ die Ordnung von a modulo n wäre. Damit kann n weder Teiler von $a^{\frac{r}{2}} + 1$ noch Teiler von $a^{\frac{r}{2}} - 1$ sein. Es gilt also

$$\text{ggT}(a^{\frac{r}{2}} + 1, n) < n \quad \text{und} \quad \text{ggT}(a^{\frac{r}{2}} - 1, n) < n.$$

Das heißt, dass zumindest entweder $\text{ggT}(a^{\frac{r}{2}} + 1, n)$ oder $\text{ggT}(a^{\frac{r}{2}} - 1, n)$ nichttrivialer Teiler von n ist, aber damit auch der andere. □

Damit Satz 1 sinnvoll für das Faktorisierungsproblem verwendet werden kann, muss die Wahrscheinlichkeit, dass die Voraussetzungen des Satzes erfüllt sind, genügend groß sein. Denn ansonsten würde die Effizienz des Verfahrens dadurch verloren gehen, dass das Verfahren sehr häufig wiederholt werden müsste, um ein passendes a zu finden. Diese Bedingung liefert der folgende Satz 2.

Satz 2 *Sei $n \in \mathbb{N}$ ungerade und $a \in \mathbb{N}$ mit $1 < a < n$ und $ggT(a, n) = 1$ zufällig gewählt, dann gilt*

$$P(\{r \text{ gerade und } a^{\frac{r}{2}} \not\equiv -1 \pmod{n}\}) \geq 1 - \frac{1}{2^{m-1}}.$$

Dabei bezeichnet $P(E)$ die Wahrscheinlichkeit des Eintreffens von Ereignis E und m die Anzahl paarweise verschiedener Primfaktoren von m.

Beweis Der Beweis wird in [16, Theorem A4.13] gegeben. □

Aus Satz 2 sind zwei Anforderungen an die Zahl n zu erkennen, bevor Satz 1 sinnvoll angewendet werden kann. Zunächst muss n ungerade sein. Es lässt sich aber leicht überprüfen, ob n gerade ist. In dem Fall ist 2 ein nichttrivialer Teiler von n. Die zweite Anforderung an n ist, dass es keine Primpotenz ist, dass n also mehr als einen Primfaktor besitzt. Ansonsten würde wegen $m = 1$ Satz 2 nutzlos werden. Denn für $m = 1$ folgt allein $P \geq 0$.

Aus der Kombination der beiden Sätze ergibt sich die Reduktion. Der allgemeine Ablauf des Algorithmus lässt sich nun wie folgt darstellen:

Eingabe: $n \in \mathbb{N}$, nicht prim
Ausgabe: Nichttrivialer Faktor von n
Ablauf:

1. Ausschluss von n gerade und n Primpotenz:
   Ist n gerade, so ist 2 ein nichttrivialer Teiler von n. (Stop)
   Gibt es $1 < a, b < n$ mit $n = a^b$, so ist a ein nichttrivialer Teiler von n. (Stop)
2. Gleichverteilte Bestimmung eines $a \in \{2, ..., n-1\}$. Prüfung von a:
   Ist $\mathrm{ggT}(a, n) > 1$, so ist $\mathrm{ggT}(a, n)$ ein nichttrivialer Teiler von n. (Stop)
3. Bestimmung der Ordnung r von a modulo n
4. Anwendung von Satz 1:
   Falls r gerade und $a^{\frac{r}{2}} \not\equiv -1$ (mod n), dann sind mit $\mathrm{ggT}(a^{\frac{r}{2}} - 1, n)$ und $\mathrm{ggT}(a^{\frac{r}{2}} + 1, n)$ zwei nichttriviale Teiler von n bekannt. (Stop)
5. Gehe zu Schritt 2.

Mit diesem Algorithmus kann nun das Faktorisierungsproblem gelöst werden. Zu bemerken ist allerdings, dass die obige Vorschrift noch unvollständig ist. Es fehlen Verfahren für die Prüfung, ob n eine Primpotenz ist (und ggf. der Ermittlung der Basis der Potenz), und für die Bestimmung von $\mathrm{ggT}(a, n)$ sowie die Bestimmung der Ordnung r von a modulo n. Die ersten beiden Probleme sind mit Hilfe von klassischen Verfahren effizient lösbar. Das erste Problem lässt sich nach einem Algorithmus von Bernstein, Lenstra Jr. und Pila mit einer geringen asymptotischen Laufzeit von $\log n \, (\log \log n)^{\mathcal{O}(1)}$ lösen.[25] Das zweite Problem, das Finden des größten gemeinsamen Teilers, kann mit Hilfe des Euklidischen Algorithmus mit polynomialen Aufwand in $\log n$ gelöst werden.[26] Durch die Lösung der

beiden ersten Probleme in polynomialer Ordnung wird das Faktorisierungsproblem nun effektiv auf das Bestimmen der Ordnung von a modulo n reduziert. Dieser fehlende Baustein soll in den folgenden Kapiteln ergänzt werden.

Es sollte noch bemerkt werden, dass in vielen Fällen, insbesondere im Falle der RSA-Verschlüsselung, weitere Informationen über n gegeben sind. Diese Informationen können zu einer Vereinfachung des Problems führen. So beschränken sich Faktorisierungsprobleme innerhalb der Kryptographie, insbesondere das Brechen von RSA, meist auf die Faktorisierung eines $n \in \mathbb{N}$, das aus zwei Primzahlen zusammengesetzt ist, das heißt $n = p \cdot q$ mit p, q paarweise verschieden und prim. Üblicherweise sind p und q sehr groß, um eine Faktorisierung zu erschweren. Das heißt wir können $p, q > 2$ annehmen. Mit den zusätzlichen Informationen wird Schritt 1 unnötig, da n weder gerade noch eine Primzahlpotenz sein kann. In diesem Fall kann daher direkt mit Schritt 2 begonnen werden. Außerdem folgt mit dem Finden eines nichttrivialen Teilers direkt die Kenntnis der Primfaktorzerlegung. Daher entfällt dann auch die Notwendigkeit einer Reduktion vom Primfaktorzerlegungsproblem auf das Faktorisierungsproblem.

3.2 Das Ordnungsbestimmungsproblem

Für das Bestimmen der Ordnung modulo n ist bisher kein klassischer Algorithmus mit einer asymptotischen Laufzeit von $\mathcal{O}((\log n)^c)$ bekannt.[16] Daher setzt der Shor-Algorithmus ab diesem Punkt auf einen Quantencomputer. Durch die Verwendung eines Quantencomputers können Rechenprozesse deutlich parallelisierter ablaufen. So kann (bspw. mittels Hadamard-Gatter) eine Ausgangssituation erreicht werden, in der alle natürlichen Zahlen von 0 bis $n-1$ gleichzeitig abgehandelt werden können.[15] Wie bereits in Kapitel 2.2.3 geschildert, ist es dabei aber problematisch, dass die berechneten Ergebnisse nicht direkt einsehbar sind, sondern über eine quantenmechanische Messung abgerufen werden müssen. Würden wir nach der Ausführung der gewollten Berechnung eine Messung durchführen, so erhielten wir allein ein zufälliges Ergebnis einer einzigen Berechnung. Daher würde der Aufwand alle Ergebnisse zu erhalten in n linear (und damit in $\log n$ exponentiell) steigen. Mittels geschickter Umformung unserer Ergebnisse kann dieses Problem gelöst werden. Die Lösung ist es, die Wahrscheinlichkeitsamplituden so zu verändern, dass wir nach der Messung mit hoher Wahrscheinlichkeit ein Ergebnis erhalten, das das Problem löst und nur mit niedriger Wahrscheinlichkeit ein anderes. Das eben Beschriebene wird durch die Quantenphasenschätzung realisiert. Diese beruht stark auf der Quantenfouriertransformation. Da beide Verfahren von essentieller Wichtigkeit für den Quantenteil des Shor-Algorithmus sind, soll in den folgenden beiden Unterkapiteln zunächst eine allgemeine Erläuterung der beiden Verfahren gegeben werden.

3.3 Quantenfouriertransformation

Bevor das Verfahren der Quantenphasenschätzung erläutert wird, sollte die Quantenfouriertransformation (QFT) erklärt werden, denn diese ist ein wesentlicher Bestandteil der Quantenphasenschätzung. Bei dieser Transformation handelt es sich um eine diskrete Fouriertransformation der Wahrscheinlichkeitsamplituden eines quantenmechanischen Zustandes. Die diskrete Fouriertransformation (DFT) ist eine Abbildung $\mathbb{C}^N \to \mathbb{C}^N$ definiert

[15] Gemeint ist eine Superposition der Zustände $|0\rangle, ..., |n-1\rangle$ als Ausgangszustand.

durch

$$x = (x_1, ..., x_N)^T \longmapsto y = (y_1, ..., y_N)^T \quad \text{mit} \quad y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \exp\left(2\pi i \frac{jk}{N}\right). \tag{27}$$

Unter einer gegebenen Orthonormalbasis bspw. der Standardbasis $|0\rangle, ..., |N-1\rangle$, lässt sich der Raum der quantenmechanischen Zustände auch mit $\mathbb{C}^N$ identifizieren. In Analogie zur DFT ist die QFT dann auf den Basiszuständen wie folgt definiert:

$$\mathrm{QFT}_N : |j\rangle \longmapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \exp\left(2\pi i \frac{jk}{N}\right) |k\rangle \quad \text{bzw.} \quad \mathrm{QFT}_N : \sum_{j=0}^{N-1} x_j |j\rangle \longmapsto \sum_{k=0}^{N-1} y_k |k\rangle . \tag{28}$$

Die Definition setzt sich linear auf superponierte Zustände fort. Bei der QFT handelt es sich um eine unitäre Transformation. Dies ist entweder aus der Analogie zur DFT ersichtlich oder kann durch die Existenz der inversen QFT gefolgert werden.[16] Dementsprechend kann die QFT auch als Gatter eines Quantencomputers verstanden werden, dass auf ein Multi-Qubit wirkt. Für einen Multi-Qubitzustand gilt $N = 2^n$ für ein $n \in \mathbb{N}$. Das Gatter wird mit $QFT^{\otimes n}$ bezeichnet. Es gilt dann für alle $x \in \{0, ..., 2^n - 1\}$

$$QFT^{\otimes n} |x\rangle_n = 2^{-\frac{n}{2}} \sum_{y=0}^{2^n-1} \exp\left(2\pi i \frac{xy}{2^n}\right) |y\rangle_n . \tag{29}$$

Es sollte bemerkt sein, dass die Bezeichnung $QFT^{\otimes n}$ der Lesbarkeit halber gewählt worden ist und es sich bei $QFT^{\otimes n}$ nicht um ein Tensorprodukt aus 1-Qubit-Gattern handelt. Trotzdem lässt sich $QFT^{\otimes n}$ als Zusammenschaltung elementarer Quantengatter darstellen wie in Lemma 10 gezeigt. Eine weitere Anmerkung ist, dass das Hadamard-Gatter die Quantenfouriertransformation auf einem Qubit realisiert. In gewisser Weise ist die Quantenfouriertransformation also eine andere Form der Verallgemeinerung des Hadamard-Gatters auf Multi-Qubits als die bisher bekannte.

3.4 Quantenphasenschätzung

Kommen wir nun zur Quantenphasenschätzung. Das Ziel der Quantenphasenschätzung ist es, die Phase eines Eigenwertes eines beliebigen unitären Operators abzuschätzen. Nehmen wir daher an, wir haben einen Eigenzustand $|u\rangle$ zu einem unitären Operator U. Sei φ_u die Phase des Eigenwertes von U zum Eigenzustand $|u\rangle$, das heißt $e^{2\pi i \varphi_u}$ ist der Eigenwert. Die Phase ist reell, da U unitär und damit normerhaltend ist. Das Ziel ist es nun also die unbekannte reelle Phase φ_u, im Folgenden verkürzt zu φ, zu bestimmen.

Voraussetzung für dieses Verfahren ist, dass uns Quantengatter zur Verfügung stehen, die erstens den Zustand $|u\rangle$ präparieren können und zweitens eine kontrollierte Ausführung von U^{2^j} für beliebige $j \in \mathbb{N}$ ermöglichen. Wie diese Prozesse umgesetzt werden, ist zunächst nicht relevant. Daher spricht man häufig von so genannten Black Boxes oder in der Komplexitäts- und Berechenbarkeitstheorie auch von Orakeln. Die Verwendung

[16] Ein solcher Beweis wird beispielsweise in Satz 3.34 von [17] geführt.

solcher Orakel zeigt, dass es sich bei der Quantenphasenschätzung nicht um einen vollständigen Algorithmus handelt, denn für einen solchen fehlen die Abläufe innerhalb der Orakel. Es handelt sich also eher um eine übergeordnete Vorgehensweise, die unter den genannten Voraussetzungen einen Algorithmus konstruieren kann.

Für die Durchführung der Quantenphasenschätzung werden zwei Register benötigt. Das erste Register soll aus m Qubits bestehen. Diese Anzahl begrenzt die Genauigkeit für die Schätzung von φ und die Wahrscheinlichkeit mit der die Quantenphasenschätzung erfolgreich ist. Das zweite Register muss aus genügend Qubits bestehen, um den Zustand $|u\rangle$ darzustellen. In genau diesem Zustand befindet sich das zweite Register zu Beginn. Die Länge des zweiten Registers sei L.

Der Ablauf der Quantenphasenschätzung stellt sich nun wie folgt dar:

Eingabe:	Unitärer Operator U, Eigenzustand $\|u\rangle$ von U
Ausgabe:	Abschätzung $\tilde{\varphi}$ von φ mit $e^{2\pi i\varphi}$ Eigenwert zu $\|u\rangle$ mit Sicherheit $1-\epsilon$
Voraussetzungen:	Orakel für Präparation von $\|u\rangle_L$ und Durchführung von $\mathrm{c}_j U^{2^j}$

Ablauf:

- $|\psi_0\rangle := |0\rangle_m |u\rangle_L$;
- $|\psi_1\rangle := (H^{\otimes m} \otimes I^{\otimes L}) |\psi_0\rangle$;
- $|\psi_2\rangle := \Lambda_m(U) |\psi_1\rangle$;
- $|\psi_3\rangle := ((QFT^{\otimes m})^{-1} \otimes I^{\otimes L}) |\psi_2\rangle$;
- $Y :=$ Messung von $|\psi_3\rangle$ bzgl. Standardbasis

Der Operator $\Lambda_m(U)$ steht dabei für die unitäre Transformation, die für einen n-stelligen Operator U und beliebige $1 \leq x \leq 2^m - 1, 1 \leq y \leq 2^n - 1$

$$\Lambda_m(U) |x\rangle_m |y\rangle_n = |x\rangle_m U^x |y\rangle_n \tag{30}$$

erfüllt. Dementsprechend realisiert $\Lambda_m(U)$ eine Potenz von U, wobei der Exponent durch das erste Register kontrolliert wird. Bevor die genaue Analyse des Verfahrens beginnt, sei auf die schematische Darstellung des Quantenschaltkreises in Abbildung 2 hingewiesen.

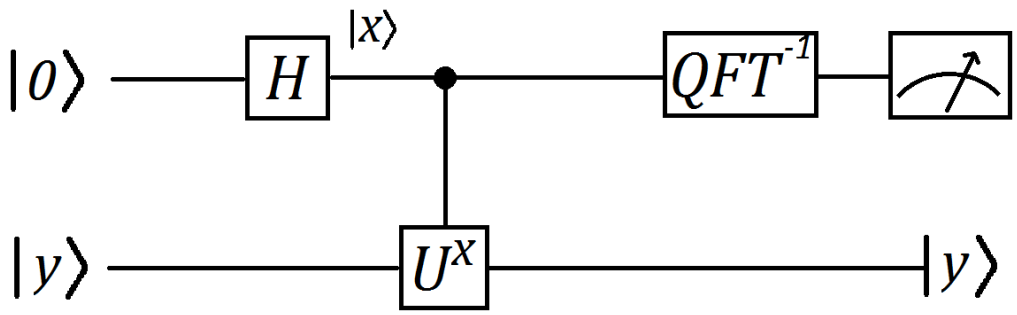

Abbildung 2: *Schematischer Schaltkreis der Quantenphasenschätzung. Die Qubits werden durch die Kabel repräsentiert. Die Kästen entsprechen Anwendungen der in den Kästen gezeigten Operatoren. Die Verbindungen mit ausgefüllten Punkten am Ende repräsentieren die kontrollierenden Qubits. Der Kasten mit Zeiger am Ende des oberen Kabels entspricht dem Messvorgang.*

Es soll nun gezeigt werden, dass der angegebene Algorithmus tatsächlich eine Abschätzung der Phase des Eigenwertes von U liefert. Dazu verfolgen wir die Zustandsänderung Schritt

für Schritt. Der erste Schritt führt zu einer gleichmäßigen Superposition der Zustände $|0\rangle, ..., |2^m - 1\rangle$ im ersten Register (siehe auch Einf. des Hadamard-Gatters, Gl. (22)):

$$|\psi_1\rangle = H^{\otimes m} |0\rangle_m I^{\otimes L} |u\rangle_L = 2^{-\frac{m}{2}} \sum_{x=0}^{2^m-1} |x\rangle_m |u\rangle_L . \tag{31}$$

Im nächsten Schritt wird das Gatter $\Lambda_m(U)$ angewandt, das wie zuvor erwähnt, die kontrollierte Anwendung einer Potenz von U realisiert, wobei der Exponent durch den Wert des ersten Registers festgelegt wird. Im Falle der Quantenphasenschätzung wird also

$$\Lambda_m(U) |x\rangle_m |u\rangle_L = |x\rangle_m U^x |u\rangle_L = e^{2\pi i \varphi x} |x\rangle_m |u\rangle_L \tag{32}$$

und damit

$$\Lambda_m(U) |\psi_1\rangle = 2^{-\frac{m}{2}} \sum_{x=0}^{2^m-1} e^{2\pi i \varphi x} |x\rangle_m |u\rangle_L \tag{33}$$

bewirkt. Die Anforderung ein Gatter, das $\Lambda_m(U)$ realisiert, zu finden, kann darauf reduziert werden, Gatter zu finden, die kontrollierte Anwendungen von U^{2^j} realisieren. Dies kann wie folgt gezeigt werden: Als Hypothese verwende $\Lambda_m(U) = \prod_{j=0}^{m-1} \mathrm{c}_j U^{\otimes 2^j}$. Der Index j unter dem c definiert hier das steuernde Qubit. Die Binärdarstellung von x laute $x_{m-1}x_{m-2}...x_1x_0$. Dann ergibt sich

$$\begin{aligned}
\Lambda_m(U) |x\rangle_m |y\rangle_n = & \prod_{j=0}^{n-1} \mathrm{c}_j U^{2^j} |x_{m-1}...x_0\rangle |y\rangle_n && (34)\\
= & |x_{m-1}...x_0\rangle \prod_{j=0}^{m-1} \left(U^{2^j}\right)^{x_j} |y\rangle_n && (35)\\
= & |x\rangle_m U^{x_{m-1}\cdot 2^{m-1}+...+x_1\cdot 2^1+x_0} |y\rangle_n && (36)\\
= & |x\rangle_m U^x |y\rangle_n . && (37)
\end{aligned}$$

Der dritte Schritt ist die Anwendung der inversen Quantenfouriertransformation auf das erste Register. Das Ergebnis ist ein bisher unbekannter Zustand $|\tilde{\varphi}\rangle_m$ des ersten Registers:

$$|\psi_3\rangle = ((QFT^{\otimes m})^{-1} \otimes I^{\otimes L}) |\psi_2\rangle = |\tilde{\varphi}\rangle_m |u\rangle_L . \tag{38}$$

Zu bemerken ist dabei, dass $|\tilde{\varphi}\rangle$ hier keinen Basiszustand mit Wert '$\tilde{\varphi}$' beschreibt, sondern einen Superpositionszustand, der bei Messung ein statistisch bestimmtes $Y = \tilde{\varphi}$ liefert. Das heißt $\tilde{\varphi}$ ist eine Zufallsvariable und kein fester Wert. Wir werden im Folgenden zeigen, dass $\tilde{\varphi}$ mit hoher Wahrscheinlichkeit eine gute Abschätzung für φ liefert. Im besonderen Fall, dass φ ein ganzzahliges Vielfaches von $\frac{1}{2^m}$ ist, kann φ sogar genau ermittelt werden und es gilt mit Sicherheit $\tilde{\varphi} = \varphi$.

Da die inverse QFT allein auf das erste Register angewandt wird und das zweite Register unverändert bleibt, wird die Darstellung des zweiten Registers in der folgenden Rechnung weggelassen. Die Anwendung von $(QFT^{\otimes m})^{-1}$ gestaltet sich wie folgt

$$|\tilde{\varphi}\rangle = (QFT^{\otimes m})^{-1} \sum_{x=0}^{2^m-1} 2^{-\frac{m}{2}} e^{2\pi i\varphi x} |x\rangle_m \tag{39}$$

$$= 2^{-\frac{m}{2}} \sum_{x=0}^{2^m-1} e^{2\pi i\varphi x} \left(QFT^{\otimes m}\right)^{-1} |x\rangle_m \tag{40}$$

$$\overset{(29)}{=} 2^{-m} \sum_{x=0}^{2^m-1} e^{2\pi i\varphi x} \sum_{y=0}^{2^m-1} e^{-2\pi i\frac{xy}{2^m}} |y\rangle_m \tag{41}$$

$$= \sum_{y=0}^{2^m-1} \underbrace{\frac{1}{2^m} \sum_{x=0}^{2^m-1} e^{2\pi ix\left(\varphi-\frac{y}{2^m}\right)}}_{=:\, a_y} |y\rangle_m . \tag{42}$$

Der Zustand $|\psi_3\rangle$ ist also eine Superposition der Basiszustände mit Amplituden a_y. Es soll nun gezeigt werden, dass diese Amplituden besonders groß sind, wenn $y \approx 2^m\varphi$, das heißt, wenn $\frac{y}{2^m}$ eine gute Abschätzung von φ ist. Dies bedarf einer genaueren Auswertung von a_y.

Die Phase lässt sich oBdA darstellen als $\varphi = \frac{b}{2^m} + \delta$ mit $b \in \mathbb{Z}$ und $0 \leq \delta < 2^{-m}$. Es gilt dann

$$a_y = \frac{1}{2^m} \sum_{x=0}^{2^m-1} \underbrace{\left(e^{2\pi i\left(\frac{b-y}{2^m}+\delta\right)}\right)^x}_{=:\, q}. \tag{43}$$

$$=: q \tag{44}$$

Betrachten wir zunächst den Spezialfall $b = y \wedge \delta = 0$. In diesem Fall gilt $q = 1$ und damit $a_y = 1$. In allen anderen Fällen gilt $q \neq 1$. Daher ergibt sich für diese Fälle gemäß geometrischer Summenformel

$$a_y = \frac{1}{2^m} \frac{1-q^{2^m}}{1-q} = \frac{1}{2^m} \frac{1-\exp\left(2\pi i(b-y+2^m\delta)\right)}{1-\exp\left(2\pi i\left(\frac{(b-y)}{2^m}+\delta\right)\right)}. \tag{45}$$

Für $\delta = 0$ folgt nun $q^{2^m} = 1$ und damit $a_y = 0$ für $y \neq b$. In dem Spezialfall $\delta = 0$ gilt also $a_y = \delta_{by}$. Damit gilt $|\tilde{\varphi}\rangle = |b\rangle$ und eine Messung liefert mit Sicherheit b und damit die korrekte Phase $\varphi = \frac{b}{2^m}$.

Betrachten wir nun den Fall, dass φ nicht durch m Bits exakt dargestellt werden kann, also $\delta \neq 0$. Das Verfahren kann daher nicht mehr exakt φ liefern. Die treffendste Abschätzung von φ ist dann $\frac{b}{2^m}$. Wir betrachten nun die Amplituden $c_l := a_{b+l}$ im Abstand l um die Amplitude der treffendsten Abschätzung b herum. Der Index $b + l$ ist dabei modulo 2^m zu verstehen.

$$c_l = \frac{1}{2^m} \frac{1-\exp\left(2\pi i(l+2^m\delta)\right)}{1-\exp\left(2\pi i\left(\frac{l}{2^m}+\delta\right)\right)}. \tag{46}$$

Wegen $\delta \neq 0$ kann die Phase nicht exakt mit Hilfe von m Bits dargestellt werden. Daher definieren wir eine Mindestgenauigkeit mit der φ bestimmt werden soll. Wir definieren unsere gewünschte Ergebnismenge als $\{l \in \mathbb{Z} : -e < l \leq e\}$, wobei e eine positive ganze Zahl ist. Unsere Ergebnismenge besteht also aus den $2e$ besten Abschätzungen von φ. Für die Genauigkeit der Abschätzung soll gelten $|\tilde{\varphi} - \varphi| = |\frac{l}{2^m} - \delta| \leq \frac{e}{2^m}$. Im Folgenden wird diese Bedingung als Präzisionsbedingung bezeichnet. Eine Abschätzung wird als *präzise* bezeichnet, wenn sie die Präzisionsbedingung erfüllt. Es soll $P(E)$ die Wahrscheinlichkeit des Eintreffens eines Ereignisses E bezeichnen. Dann soll nun eine obere Schranke für die Wahrscheinlichkeit $P(\{\tilde{\varphi}\ \textit{nicht präzise}\})$ die Phase nicht mit der gewünschten Genauigkeit zu messen bestimmt werden. Es gilt (unter Beachtung von $-l \equiv 2^m - l \ (\bmod 2^m)$

$$P(\{\tilde{\varphi}\ \textit{nicht präzise}\}) = \sum_{e \leq -l < 2^{m-1}} |c_l|^2 + \sum_{e < l \leq 2^{m-1}} |c_l|^2. \tag{47}$$

Für die Abschätzung von $|c_l|$ werden zunächst zwei Ungleichungen benötigt. Für beliebiges $\theta \in \mathbb{R}$ gilt

$$|1 - \exp(i\theta)| \leq 2. \tag{48}$$

Für $|\theta| \leq \pi$ gilt ferner

$$|1 - \exp(i\theta)| \geq 2\frac{|\theta|}{\pi}. \tag{49}$$

Denn es gilt $|1-\exp(i\theta)|^2 = (1-\cos\theta)^2+\sin^2\theta = 2(1-\cos\theta) = 4\sin^2\frac{\theta}{2} \geq 4\left(\frac{\theta}{\pi}\right)^2$. Die letzte Ungleichung kann unter der Bedingung $\theta \in [-\pi, \pi]$ bspw. über die Reihenentwicklung von $\sin^2$ gefolgert werden. Mit diesen beiden Ungleichungen kann gefolgert werden

$$|c_l| = \left|\frac{1}{2^m}\frac{1-\exp(2\pi i(2^m\delta - l))}{1-\exp\left(2\pi i\left(\delta - \frac{l}{2^m}\right)\right)}\right| \leq \frac{1}{2^m}\frac{2}{\left|1-\exp\left(2\pi i\left(\delta - \frac{l}{2^m}\right)\right)\right|} \leq \frac{1}{2^{m+1}\left|\delta - \frac{l}{2^m}\right|}. \tag{50}$$

Für die Gültigkeit des letzten Schrittes ist zu beachten, dass wegen $-2^{m-1} < l \leq 2^{m-1}$ die Voraussetzung $|2\pi\left(\delta - \frac{l}{2^m}\right)| \leq \pi$ erfüllt ist. Einsetzen von (50) in (47) liefert dann

$$P(\tilde{\varphi}\ \textit{nicht präzise}) \leq \frac{1}{4}\left(\sum_{l=-2^{m-1}+1}^{-e}\frac{1}{l^2} + \sum_{l=e+1}^{2^{m-1}}\frac{1}{(l-1)^2}\right) \tag{51}$$

$$\leq \frac{1}{2}\sum_{e}^{2^{m-1}-1}\frac{1}{l^2} \tag{52}$$

$$\leq \frac{1}{2}\int_{e-1}^{2^{m-1}-1}\frac{dl}{l^2} \tag{53}$$

$$\leq \frac{1}{2(e-1)}. \tag{54}$$

Die Voraussetzungen an die Quantenphasenschätzung soll es nun sein, dass das Verfahren mit einer Wahrscheinlichkeit von mindestens $1-\epsilon$ eine präzise Abschätzung $\tilde{\varphi}$ mit einer Genauigkeit von 2^{-n}, das heißt auf n Qubits genau, liefert. Dabei ist $0 \leq \epsilon < 1$. Unter

diesen Voraussetzungen sollen e und m optimal, das heißt m möglichst klein gewählt werden. Es muss einerseits

$$\frac{1}{2(e-1)} \leq \epsilon \Leftrightarrow e \geq 1 + \frac{1}{2\epsilon} \tag{55}$$

sowie andererseits

$$|\tilde{\varphi} - \varphi| < \frac{e}{2^m} \leq \frac{1}{2^n} \Leftrightarrow e \leq 2^{m-n} \tag{56}$$

erfüllt werden. Es ergibt sich also insgesamt

$$1 + \frac{1}{2\epsilon} \leq e \leq 2^{m-n}. \tag{57}$$

Das kleinstmögliche m ist durch das kleinstmögliche e beschränkt mit $e_{\min} = 1 + \left\lceil \frac{1}{2\epsilon} \right\rceil \leq 2 + \frac{1}{2\epsilon}$. Damit gilt

$$m_{\min} = n + \left\lceil \log_2 \left(2 + \frac{1}{2\epsilon} \right) \right\rceil. \tag{58}$$

Die Quantenphasenschätzung ist damit ein mächtiges Werkzeug, das es uns ermöglicht die Phase des Eigenwertes eines unitären Operators beliebig genau abzuschätzen. Die einzigen Bedingungen für die Durchführung der Quantenphasenschätzung sind dabei, dass die Präparation des gewünschten Eigenzustandes sowie die kontrollierte Ausführung von U^{2^j}-Operationen möglich sind. Die erste Voraussetzung - also den gewünschten Eigenzustand präparieren zu können - kann sogar auf Kosten einer geringeren Sicherheitswahrscheinlichkeit umgangen werden. Denn: Es sei der Anfangszustand $|\Psi\rangle \in \mathcal{H}^{\otimes L}$ des zweiten Registers beliebig. In der Eigenbasis lässt sich schreiben $|\Psi\rangle = \sum_u c_u |u\rangle$. Damit sollte die Quantenphasenschätzung vor der Messung einen Endzustand von $\sum_u c_u |\tilde{\varphi}_u\rangle |u\rangle$ liefern. Dabei ist $\tilde{\varphi}_u$ die Abschätzung der Phase φ_u des Eigenwertes u. Mit einer Wahl von $m = m_{\min}$ erhalten wir also eine präzise Abschätzung $\tilde{\varphi}_v$ der Phase eines spezifischen Eigenwertes v mit einer Wahrscheinlichkeit von $|c_v|^2(1-\epsilon)$. Die zweite Voraussetzung stellt dagegen eine deutlich schärfere Einschränkung dar. Zwar kann $U^{\otimes 2^j}$ unter Kenntnis von U einfach durch 2^j-fache Hintereinanderausführung realisiert werden, dies bedeutet aber einen exponentiellen Aufwand in j. Ist ein effizienter Algorithmus erwünscht, so muss eine Realisierung von $U^{\otimes 2^j}$ gefunden werden, die nur einen polynomialen Aufwand benötigt. Dies kann eine wesentliche Schwierigkeit bei der Erstellung eines solchen Algorithmus bedeuten.

3.5 Ordnungsbestimmung mittels Quantenphasenschätzung

Im vorherigen Kapitel wurde das Verfahren der Quantenphasenschätzung allgemein erläutert. Jetzt soll dieses Verfahren zur Bestimmung der Ordnung modulo N benutzt werden. Das heißt wir suchen für natürliche, teilerfremde a und N mit $1 < a < N$ den kleinsten natürlichen Exponenten r mit $a^r \equiv 1 \pmod{N}$. Es soll angemerkt sein, dass es sich bei dem hier beschriebenen Algorithmus nicht um die originale Formulierung in Shors Publikation[5] handelt. Stattdessen wird hier eine äquivalente Formulierung getroffen, die zuerst von Kitaev vorgeschlagen wurde[27] und zum Beispiel in [16] zu finden ist. Der

Grund für die Verwendung dieser alternativen Formulierung ist, dass so der Zusammenhang zur Quantenphasenschätzung deutlicher wird.

Um das Verfahren der Quantenphasenschätzung anzuwenden, wird ein unitärer Operator benötigt, dessen Eigenwerte auf r schließen lassen. Die Phasen der Eigenwerte können dann mittels Quantenphasenschätzung abgeschätzt werden. Für den Algorithmus wird der unitäre Operator U_a verwendet, der wie folgt definiert ist:

$$U_a \left|y\right\rangle_L := \begin{cases} \left|ay \bmod N\right\rangle_L & \text{für } 0 \leq y \leq N-1, \\ \left|y\right\rangle_L & \text{für } N \leq y \leq 2^L - 1. \end{cases} \tag{59}$$

Im Folgenden wird zur Verkürzung der Notation die Länge des Registers $L \geq \lceil \log_2(N+1) \rceil$ weggelassen. Die Eigenzustände von U_a sind für $0 \leq s \leq r-1$ gegeben durch

$$\left|u_s\right\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} \left|a^k \bmod N\right\rangle . \tag{60}$$

Dies lässt sich leicht nachrechnen, denn

$$\begin{aligned} U_a \left|u_s\right\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} \left|a^{k+1} \bmod N\right\rangle && (61) \\ &\overset{\text{(Indexverschiebung)}}{=} e^{2\pi i \frac{s}{r}} \frac{1}{\sqrt{r}} \sum_{k=1}^{r} e^{-2\pi i \frac{s}{r} k} \left|a^k \bmod N\right\rangle && (62) \\ &\overset{\text{(0-ter Smd. gleich r-ter Smd.)}}{=} e^{2\pi i \frac{s}{r}} \left|u_s\right\rangle && (63) \end{aligned}$$

ergibt die Eigenwertgleichung mit Eigenwerten $u_s = e^{2\pi i \frac{s}{r}}$. Damit sind die Phasen der Eigenwerte gegeben durch $\varphi_s = \frac{s}{r}$. Da alle Eigenwerte betragsmäßig gleich 1 sind, folgt hiermit auch die Unitarität von U_a.[17]

Bevor wir die Quantenphasenschätzung zur Abschätzung der Phase φ_s benutzen können, müssen allerdings noch zwei Voraussetzungen erfüllt werden. Die erste Voraussetzung beinhaltet, dass ein Eigenzustand $\left|u_s\right\rangle$ oder zumindest eine Superposition dieser Eigenzustände präparierbar ist. Mit folgender Beobachtung ist die Voraussetzung aber leicht erfüllbar:

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left|u_s\right\rangle = \frac{1}{r} \sum_{s=0}^{r-1} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} \left|a^k \bmod N\right\rangle = \frac{1}{r} \sum_{k=0}^{r-1} \underbrace{\left(\sum_{s=0}^{r-1} e^{-2\pi i \frac{s}{r} k} \right)}_{= r \cdot \delta_{k0} \text{ (siehe auch Gl. (43)-(45))}} \left|a^k \bmod N\right\rangle = \left|1\right\rangle . \tag{64}$$

Das heißt es genügt einfach den Zustand $\left|1\right\rangle$ zu präparieren.

[17] Es sollte hier angemerkt sein, dass es sich bei den Eigenzuständen von Gleichung (60) nicht um alle Eigenzustände von U_a handelt. Durch die Präparation des Zustandes in Gleichung (64) wird aber der Raum auf den U_a im Algorithmus wirkt auf den durch die genannten Eigenzustände aufgespannten Raum, die multiplikative Gruppe von a, beschränkt. Daher ist die Unitarität zumindest in diesem Unterraum erfüllt.

Als zweite Voraussetzung müssen kontrollierte Anwendungen von $U_a^{2^j}$ durchführbar sein. Das Wichtige hierbei ist, wie schon vorher erwähnt, dass die einfache 2^j-fache Hintereinanderausführung von U_a nicht genügt, denn dies würde in einem hohen exponentiell mit j steigendem Aufwand resultieren. Es wird eine Methode benötigt, die $U_a^{2^j}$ in polynomialen Aufwand ausführen kann. Die vom Shor-Algorithmus verwendete Methode nennt sich binäre Exponentiation bzw. Repeated Squaring oder auch Square-and-Multiply:

> Wichtig ist zunächst die Beobachtung $U_a^{2^j} = U_{a^{2^j}}$. Die Berechnung von $a^{2^j} = \overbrace{a \cdot ... \cdot a}^{2^j \text{ mal}}$ würde 2^j Multiplikationen benötigen. Das Verfahren der binären Exponentiation benötigt zur Berechnung von a^x nur $\mathcal{O}(\log x)$ Multiplikationen. Für den Spezialfall $x = 2^j$ werden also j Multiplikationen benötigt und das Verfahren lässt sich rekursiv wie folgt darstellen:
>
> $$k_0 = a \text{ und } k_i = k_{i-1}^2 \text{ für } i \in \mathbb{N}. \tag{65}$$
>
> Damit folgt $k_j = a^{2^j}$.

Die Voraussetzungen zur Durchführung der Quantenphasenschätzung sind also erfüllt.[18] Die Abschätzung $\tilde{\varphi}_s$ von $\varphi_s = \frac{s}{r}$ muss uun genau genug sein, damit auf r geschlossen werden kann. Im folgenden Kapitel wird gezeigt, dass die Bedingung $|\tilde{\varphi}_s - \varphi_s| \leq \frac{1}{2r^2}$ dafür hinreichend ist. Um diese Präzision zu erreichen, muss $n = 2\log_2 r + 1$ gewählt werden, denn dann gilt $|\tilde{\varphi}_s - \varphi_s| \leq 2^{-n} = \frac{1}{2r^2}$. Bei Wahl von $L \geq \log_2 N \geq \log_2 r$ kann nun folgendes konstatiert werden: Bei einer Verwendung von $m = 2L + 1 + \lceil \log_2 \left(2 + \frac{1}{2\epsilon}\right) \rceil$ Qubits im ersten Register und L Qubits im zweiten Register lässt sich also mit einer Wahrscheinlichkeit von $1-\epsilon$ eine präzise Abschätzung $\tilde{\varphi}_s$ von $\varphi_s = \frac{s}{r}$ finden. Dabei liegt s gleichverteilt in der Menge $\{0, ..., r-1\}$. Das heißt, die Wahrscheinlichkeit ein spezifisches φ_s zu treffen liegt bei $\frac{1-\epsilon}{r}$.

3.6 Kettenbruchentwicklung zur Ordnungsbestimmung

Als Ergebnis des zuvor geschilderten Verfahrens erhalten wir $\tilde{\varphi}_s$, im Folgenden mit φ bezeichnet, für das mit hoher Wahrscheinlichkeit $1-\epsilon$ gilt $\left|\varphi - \frac{s}{r}\right| \leq \frac{1}{2r^2}$. Es kann nun gezeigt werden, dass sich r aus φ bestimmen lässt. Bei dem verwendeten Verfahren handelt es sich um ein klassisches Verfahren. Zur Erläuterung des Verfahrens werden grundlegende Kenntnisse über Kettenbrüche benötigt.

Für gegebene $a_0 \in \mathbb{Z}$ und $a_1, ..., a_K \in \mathbb{N}$ mit $K \in \mathbb{N}_0$ wird

$$[a_0, ..., a_K] := a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \frac{1}{a_K}}}} \in \mathbb{Q} \tag{66}$$

ein endlicher Kettenbruch genannt. Ein endlicher Kettenbruch ist eindeutig für $K = 0$ oder $a_K > 1$. Für einen eindeutigen endlichen Kettenbruch bezeichnet $[a_0, ..., a_k]$ mit

[18] Streng genommen handelt es sich bei U_x nicht um ein elementares Gatter. In Kapitel 4 wird aber gezeigt, wie U_x in Form elementarer Gatter realisiert werden kann.

$0 \leq k \leq K$ die k-te Konvergente des Kettenbruchs.

Satz 3 *Jede rationale Zahl $\frac{p}{q} \in \mathbb{Q}$ mit $p \in \mathbb{Z}$ und $q \in \mathbb{N}$ lässt sich durch genau einen eindeutigen endlichen Kettenbruch $[a_0, \ldots, a_K] = \frac{p}{q}$ darstellen. Mit $r_1, ..., r_K \in \mathbb{N}$ ergibt sich die Darstellung (Euklidischer Algorithmus)*

$$\begin{aligned}
p &= a_0 q + r_1 && (0 < r_1 < q),\\
q &= a_1 r_1 + r_2 && (0 < r_2 < r_1),\\
r_1 &= a_2 r_2 + r_3 && (0 < r_3 < r_2),\\
&\vdots\\
r_{K-2} &= a_{K-1} r_{K-1} + r_K && (0 < r_K < r_{K-1}),\\
r_{K-1} &= a_K r_K + 0.
\end{aligned}$$

Beweis (nach [17]) Da $(r_k)_k$ eine fallende Folge natürlicher Zahlen ist, gibt es einen Index K mit $r_{K+1} = 0$. Das heißt, dass der euklidische Algorithmus terminiert. Die rekursiven Definitionen für r_k liefern

$$\frac{p}{q} = a_0 + \frac{r_1}{q} = a_0 + \frac{1}{\frac{q}{r_1}} = \frac{1}{a_1 + \frac{r_2}{r_1}} = ... = [a_0, ..., a_K]. \tag{67}$$

Der Kettenbruch ist eindeutig, denn entweder gilt $K = 0$ oder es gilt $K > 0$, aber dann gilt auch $a_K > 1$, denn für $a_K = 1$ folgt $r_{K-1} = r_K$ und damit wäre wegen $r_{K-2} = (a_{K-1} + 1)r_{K-1} + 0$ schon $r_K = 0$. □

Mit der erhaltenen Aussage lässt sich nun die folgende Konvergenteneigenschaft beweisen, mit der wir letztendlich die Bestimmbarkeit von r folgern können.

Satz 4 (Konvergenteneigenschaft) *Es seien $x, \frac{p}{q} \in \mathbb{Q}$ mit $p \in \mathbb{Z}$ und $q \in \mathbb{N}$, so dass*

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

dann ist $\frac{p}{q}$ eine Konvergente in der eindeutigen endlichen Kettenbruchdarstellung von x.

Beweis Der Beweis wird in [16, Theorem A4.16] gegeben. □

Mit der Konvergenteneigenschaft folgt, dass für eine präzise Abschätzung φ der Bruch $\frac{s}{r}$ eine Konvergente der eindeutigen endlichen Kettenbruchdarstellung von φ ist. Denn es gilt $\left|\varphi - \frac{s}{r}\right| \leq \frac{1}{2r^2}$. Die Vorgehensweise zur Bestimmung von r lautet dann wie folgt:

- Bestimme die eindeutige endliche Kettenbruchdarstellung $[a_0, ..., a_K]$ von φ (Euklidischer Algorithmus)
- Für $k = 0, ..., K$
  - Bestimme die k-te Konvergente $[a_0, ..., a_k]$ der Kettenbruchdarstellung von φ

- Bestimme die Bruchdarstellung $\frac{s_k}{r_k} = [a_0, ..., a_k]$ (inverser Euklidischer Algorithmus)
- Prüfe, ob r_k die Ordnung von a modulo N ist. Falls erfolgreich (Stop)

Die Prüfung, ob r_k Ordnung von a modulo N ist, kann durch einfaches Berechnen von $a^{r_k} \pmod N$ über binäre Exponentiation erfolgen. Das Verfahren liefert allerdings noch keine Gewissheit r zu finden, denn es gibt zwei Fälle, in denen das Verfahren scheitert.

- Erster Fall: Mit Wahrscheinlichkeit ϵ ist φ keine präzise Abschätzung von $\frac{s}{r}$. Dementsprechend kann die Konvergenteneigenschaft nicht angewandt werden.
- Zweiter Fall: s und r sind nicht teilerfremd. In diesem Fall ist der Bruch $\frac{s}{r}$ kürzbar und r_k liefert im besten Fall nur einen Faktor von r.

Scheitert also jede Prüfung der Kandidaten r_k muss die Prozedur der Phasenschätzung und Kettenbruchentwicklung (PuK-Prozedur) erneut durchgeführt werden. Für einen effizienten Algorithmus dürfen die beiden Fälle des Scheiterns nur mit einer genügend geringen Wahrscheinlichkeit auftreten. Der erste Fall tritt mit einer konstanten Wahrscheinlichkeit von ϵ auf. Es genügen daher $\mathcal{O}(1)$ Wiederholungen der Prozedur, um eine beliebig kleine Wahrscheinlichkeit des Scheiterns im Sinne des ersten Falles zu erhalten. Hinreichend für den zweiten Fall ist die Feststellung, dass es sehr wahrscheinlich ist, dass ein zufälliges $s \in \{0, ..., r-1\}$ koprim zu r ist. Denn es gilt

$$P(\{s, r \ koprim\}) = \frac{\varphi(r)}{r} > \frac{\delta}{\log\log r} \geq \frac{\delta}{\log\log N} \tag{68}$$

für eine Konstante δ nach [28, Theorem 328]. Dabei ist $\varphi(r)$ die Eulersche Phi-Funktion von r also die Anzahl der zu r koprimen Zahlen kleiner r. Eine $\mathcal{O}(\log\log N)$-fache Wiederholung der PuK-Prozedur sollte also mit hoher Wahrscheinlichkeit ein primes s liefern.

3.7 Vorschlag zur Verbesserung des Algorithmus

Derzeit erscheint es sinnvoll davon auszugehen, dass klassische Berechnungen in der Praxis einen geringeren Aufwand erfordern als auf dem Quantencomputer. Daher ist es lohnenswert über eine weitere Reduktion der Quantenberechnungen auf Kosten von zusätzlichem Post-Processing auf klassischer Seite nachzudenken. In der Publikation von Shor[5] werden daher mehrere Vorschläge unterbreitet, die die Anzahl der Wiederholungen der PuK-Prozedur reduzieren. Die Vorschläge beziehen sich dabei alle auf den zweiten Fall. Hier soll nur der Vorschlag dargestellt werden, der die größte Aufwandsreduktion liefert.

Für diese Aufwandsreduktion wird die PuK-Prozedur direkt zweifach (für gleiches a) ausgeführt. Die beiden Prozeduren liefern zwei endliche Folgen $\left(\frac{s'_{1k}}{r'_{1k}}\right)_k$ und $\left(\frac{s'_{2k}}{r'_{2k}}\right)_k$, die jeweils ein Folgenglied $\frac{s_1}{r_1}$ bzw. $\frac{s_2}{r_2}$ enthalten[19] für das es $s_1, s_2 \in \{0, ..., r-1\}$ gibt mit

$$\frac{s'_1}{r'_1} = \frac{s_1}{r} \quad \text{bzw.} \quad \frac{s'_2}{r'_2} = \frac{s_2}{r}. \tag{69}$$

[19] Die Indizes der beiden Folgenglieder stehen nicht für den Folgenindex, sondern sollen nur das gewählte Folgenglied identifizieren.

Die Idee ist es nun, dass die Ordnung r sich für den Fall zweier teilerfremder s_1 und s_2 aus dem kleinsten gemeinsamen Vielfachen(kgV) von r_1' und r_2' ergibt. Dies kann wie folgt gezeigt werden: Durch Umstellen von (69) nach r_1' und r_2' ergibt sich

$$r_1' = r\frac{s_1'}{s_1} = \frac{r}{\left(\frac{s_1}{s_1'}\right)} \quad \text{und} \quad r_2' = r\frac{s_2'}{s_2} = \frac{r}{\left(\frac{s_2}{s_2'}\right)}. \tag{70}$$

Weil s_1' und r_1' sowie s_2' und r_2' teilerfremd sind, ist s_i ein Vielfaches von s_i' für $i = 1, 2$. Daher sind $\frac{s_1}{s_1'}$ sowie $\frac{s_2}{s_2'}$ natürliche Zahlen und es gilt (vgl. bspw. [29, Satz 17])

$$\mathrm{kgV}(r_1', r_2') = \frac{r}{\mathrm{ggT}(\frac{s_1}{s_1'}, \frac{s_2}{s_2'})}. \tag{71}$$

Wegen $1 \leq \mathrm{ggT}\left(\frac{s_1}{s_1'}, \frac{s_2}{s_2'}\right) \leq \mathrm{ggT}(s_1, s_2) = 1$ folgt $\mathrm{kgV}(r_1', r_2') = r$. Das kleinste gemeinsame Vielfache lässt sich aus dem größten gemeinsamen Teiler bestimmen.[20] Dieser lässt sich wie bereits zuvor erläutert effizient über den Euklidischen Algorithmus bestimmen. Es wird nun noch eine Abschätzung benötigt, die eine untere Schranke für die Wahrscheinlichkeit zwei koprime s_1 und s_2 zu erhalten angibt. Diese Wahrscheinlichkeit lässt sich ausdrücken als

$$P(\{s_1, s_2 \ koprim\}) = 1 - \sum_{p \in \mathbb{P}} P(p|s_1)P(p|s_2). \tag{72}$$

Dabei bezeichnet $\mathbb{P}$ die Menge der Primzahlen und $P(a|b)$ die Wahrscheinlichkeit, dass b durch a geteilt wird. Die Zahlen s_1 und s_2 sind dabei zufällig und gleichverteilt aus der Menge $\{0, ..., r-1\}$ gewählt worden. Da eine Primzahl p von p beginnend jede p-te Zahl teilt, gilt $P(p|s_i) \leq 1/p$ für $i = 1, 2$. Damit folgt

$$\sum_{p \in \mathbb{P}} P(p|s_1)P(p|s_2) \leq \sum_{p \in \mathbb{P}} \frac{1}{p^2}. \tag{73}$$

Eine mögliche Abschätzung wäre zum Beispiel

$$\sum_{p \in \mathbb{P}} \frac{1}{p^2} \leq \sum_{n=1}^{\infty} \frac{1}{n^2} - 1 - \frac{1}{4} = \zeta(2) - \frac{5}{4} = \frac{\pi^2}{6} - \frac{5}{4} \tag{74}$$

und damit

$$P(s_1, s_2 \text{ koprim}) \geq \frac{9}{4} - \frac{\pi^2}{6} \geq 0,6. \tag{75}$$

Damit genügt auch für den zweiten Fall eine Zahl von $\mathcal{O}(1)$ Wiederholungen der PuK-Prozedur, um eine beliebig kleine Wahrscheinlichkeit des Scheiterns zu erzielen.

[20] Für beliebige ganze Zahlen a, b gilt $|a \cdot b| = \mathrm{kgV}(a, b) \cdot \mathrm{ggT}(a, b)$.

3.8 Zusammenfassung des vollständigen Algorithmus

Die zuvor erläuterten Verfahren lassen sich schließlich zu einem vollständigen Algorithmus zusammenfassen. Eine mögliche Realisierung sieht wie folgt aus:

Eingabe: $n \in \mathbb{N}$, nicht prim

Ausgabe: Nichttrivialer Faktor von n (mit endlicher Wahrscheinlichkeit)

qm. Aufwand: $\mathcal{O}(L^2 \log L \log\log L)$

kl. Aufwand: $\mathcal{O}(L^4 \log L \log\log L)$

Ablauf:

1. Ausschluss von n gerade und n Primpotenz:
   Ist n gerade, so ist 2 ein nichttrivialer Teiler von n. (Stop) Ist $n = a^b$, so ist a ein nichttrivialer Teiler von n. (Stop)

2. Gleichverteilte Bestimmung eines $a \in \{2, ..., n-1\}$. Prüfung von a:
   Ist $\mathrm{ggT}(a, n) > 1$, so ist $\mathrm{ggT}(a, n)$ ein nichttrivialer Teiler von n. (Stop)

3. Bestimmung der Ordnung r von a modulo n:
   Wähle $L \geq \lceil \log_2(n+1) \rceil$ und $m \geq 2L + 1 + \lceil \log_2 \left(2 + \frac{1}{2\epsilon}\right) \rceil$.

   $|\psi_0\rangle := |0\rangle_m |1\rangle_L$;
   $|\psi_1\rangle := (H^{\otimes m} \otimes I^{\otimes L}) |\psi_0\rangle$;
   $|\psi_2\rangle := \Lambda^m(U) |\psi_1\rangle$; (Ausführung von U^{2^j} über binäre Exponentiation)
   $|\psi_3\rangle := ((QFT^{-1})^{\otimes m} \otimes I^{\otimes L}) |\psi_2\rangle$;
   $Y :=$ Messung von $|\psi_3\rangle$ bzgl. Standardbasis

   Sei y das Ergebnis der Messung. Dann bestimme zu $\frac{y}{2^m}$ die Konvergenten der Kettenbruchentwicklung und berechne für $k = 0, ..., K$ die Bruchdarstellungen

   $$\frac{s_k}{r_k} := [a_0, ..., a_k].$$

4. Prüfe die Kandidaten r_k. Scheitert jede Prüfung von r_k, schlägt der Algorithmus fehl.(*)

5. Falls r gerade und $a^{\frac{r}{2}} \not\equiv -1 \pmod{n}$, dann sind mit $\mathrm{ggT}(a^{\frac{r}{2}} - 1, n)$ und $\mathrm{ggT}(a^{\frac{r}{2}} + 1, n)$ zwei nichttriviale Teiler von n bekannt. (Stop)
   Andernfalls schlägt der Algorithmus fehl.

An der Stelle (*) lassen sich noch die am Ende des letzten Kapitels genannten Vorschläge zur Verbesserung des Laufzeitverhaltens in der Praxis einbringen. In diesem Falle würde Schritt 3 zwei Mal ausgeführt werden und zwei Folgen von Kandidaten $(r_{1i})_{1 \leq i \leq K}$ und $(r_{2j})_{1 \leq j \leq K'}$ liefern. In Schritt 4 würden dann alle Kandidaten $\mathrm{kgV}(r_{1i}, r_{2j})$ als Ordnung von a modulo n geprüft.

3.9 Beispielhafte Durchführung des Algorithmus für $N = 21$

In diesem Unterkapitel soll der Shor-Algorithmus anhand eines Beispiels, dem Spezialfall $N = 21 = 3 \cdot 7$, veranschaulicht werden. Dazu werden die Bestandteile des Gesamtalgorithmus in der Form des obigen Kapitels Schritt für Schritt durchgeführt. Die Durchführung des Algorithmus beginnt mit Schritt 1. Da N nicht gerade ist, ist 2 kein Teiler von N. Auch ein Test, ob es sich bei N um eine Primpotenz handelt, schlägt fehl. Im zweiten Schritt wird ein zufälliges $a \in \{2, ..., 20\}$ ausgewählt und (durch Anwendung des Euklidischen Algorithmus) der größte gemeinsame Teiler von a und 21 berechnet. Gilt $\mathrm{ggT}(a, 21) > 1$, so ist mit $\mathrm{ggT}(a, 21)$ ein nichttrivialer Teiler gefunden und der Algorithmus terminiert. Andernfalls wird der dritte Schritt des Algorithmus ausgeführt. Wäre zum Beispiel $a = 14$, so ergäbe sich $\mathrm{ggT}(a, 21) = 7$. Damit wäre ein Faktor von 21 gefunden. In unserem Fall sei $a = 11$. Damit gilt $\mathrm{ggT}(a, 21) = 1$ und Schritt 3, das heißt die Bestimmung der Ordnung von 11 modulo 21 per Quantenphasenschätzung, wird ausgeführt. Für den Quantenpart des Algorithmus werden zwei Register benötigt. Für das zweite Register werden $L = \lceil \log_2(N) \rceil = 5$ Qubits verwendet. Die Sicherheitswahrscheinlichkeit des Ordnungsbestimmungsverfahren soll bei $\epsilon = \frac{1}{10}$ liegen. Damit müssen für das erste Register $m = 2L + 1 + \lceil \log_2(7) \rceil = 14$ Qubits verwendet werden. Die Messung des finalen Zustandes liefert im Allgemeinen mit einer Wahrscheinlichkeit von mindestens $1 - \epsilon$ ein $y \in \{0, ..., 2^m - 1\}$ mit

$$\left| \frac{y}{2^m} - \frac{s}{r} \right| \leq \frac{1}{2r^2} \quad \text{für ein } s \in \{0, ..., r-1\}. \tag{76}$$

Die Ordnung von 11 modulo 21 ist 6, denn $11^2 \equiv 16$, $11^3 \equiv 8$, $11^4 \equiv 4$, $11^5 \equiv 2$ und $11^6 \equiv 1$ (hier: $a \equiv b \Leftrightarrow a \equiv b \ (\mathrm{mod}\ 21)$). In dem betrachteten Fall erhalten wir also mit einer Wahrscheinlichkeit von mindestens 90% ein $0 \leq y \leq 16383$ mit

$$\left| \frac{y}{16384} - \frac{s}{6} \right| \leq \frac{1}{72} \quad \text{für ein } s \in \{0, ..., 5\}. \tag{77}$$

In diesem Beispiel soll die Messung $y = 13442$ liefern. Dann ist Gleichung (77) erfüllt, denn $|\frac{13442}{16384} - \frac{5}{6}| \leq 0,013 < \frac{1}{72}$. Nun ist die Kettenbruchentwicklung von $\frac{13442}{16384}$ zu bestimmen. Der Euklidische Algorithmus liefert

$$\begin{aligned}
13442 &= 0 \cdot 16384 + 13442, \\
16384 &= 1 \cdot 13442 + 2942, \\
13442 &= 4 \cdot 2942 + 1674, \\
2942 &= 1 \cdot 1674 + 1268, \\
1674 &= 1 \cdot 1268 + 406, \\
1268 &= 3 \cdot 406 + 50, \\
406 &= 8 \cdot 50 + 6, \\
50 &= 8 \cdot 6 + 2, \\
6 &= 3 \cdot 2 + 0.
\end{aligned}$$

Damit ist die $[0, 1, 4, 1, 1, 3, 8, 8, 3]$ die Kettenbruchdarstellung von $\frac{13442}{16384}$. Nun sind die Nenner der Konvergenten des Kettenbruchs als Kandidaten für die Ordnung 11 modulo 21 zu prüfen.

$$\begin{aligned}
[0,1] &= 1, \\
[0,1,4] &= \frac{1}{1+\frac{1}{4}} = \frac{4}{5}, \\
[0,1,4,1] &= \frac{1}{1+\frac{1}{4+\frac{1}{1}}} = \frac{5}{6}.
\end{aligned}$$

Die Prüfung des Nenners der dritten Konvergente liefert schließlich die korrekte Ordnung $r = 6$.

Es soll hier angemerkt sein, dass die Varianten $s = 2, 3, 4$ hier zu den am Ende von Kapitel 3.6 genannten Problemen führen, da in diesem Fall s und r nicht teilerfremd sind und die Konvergenten einen um den gemeinsamen Faktor gekürzten Nenner liefern. In diesem Fall würde also jede Prüfung der Kandidaten für die Ordnung scheitern und der Algorithmus würde fehlschlagen. In unserem Fall aber, kann Schritt 4 ausgeführt werden. Da die Ordnung $r = 6$ gerade ist und $11^3 \equiv 8 \not\equiv -1 \pmod{21}$, kann Theorem 1 angewandt werden und $\mathrm{ggT}(11^3 - 1, 21) = \mathrm{ggT}(7, 21) = 7$ und $\mathrm{ggT}(11^3 + 1, 21) = \mathrm{ggT}(9, 21) = 3$ sind nichttriviale Faktoren von 21. Damit ist $21 = 3 \cdot 7$ faktorisiert.

4 Abschätzung des Aufwandes

In diesem Kapitel soll der Aufwand des in Kapitel 3.8 angegebenen Algorithmus abgeschätzt werden. Die Vorgehensweise ist dabei nicht in allen Fällen streng formal. Das Ziel soll es eher sein den in 3.8 angegebenen Aufwand zu motivieren. Der abgeschätzte Aufwand wird dabei in Form elementarer Rechenoperationen angegeben. Da zwischen dem Zeitaufwand elementarer klassischer und elementarer quantenmechanischer Rechenoperationen in der Praxis ein großer Unterschied liegen kann, werden beide Komplexitäten getrennt erfasst. Hier werden elementare Rechenoperationen im klassischen Sinne als logische 1- und 2-Bit-Operationen verstanden. Als Menge elementarer Quantengatter dient eine universelle Familie solcher Gatter, die später genauer definiert wird. Damit das Ziel dieses Kapitels klar ist, soll die zu beweisende Abschätzung vorab konstatiert werden.

Satz 5 *Sei $L = \lceil \log_2(N+1) \rceil$ die Anzahl der Bits, um die zu faktorisierende Zahl N darzustellen, dann werden $\mathcal{O}(L^4 \log L \log\log L)$ elementare klassische und $\mathcal{O}(L^2 \log L \log\log L)$ elementare quantenmechanische Rechenoperationen benötigt, um den in Kapitel 3.8 aufgeführten Algorithmus (mit (*)) zu realisieren.*

Bevor wir mit dem Beweis dieser Abschätzung beginnen, benötigen wir zunächst einige Vorüberlegungen, die in den folgenden drei Lemmata bewiesen werden. Für die folgenden Beweise führen wir die Notation $M_X(L)$ ein, die den Aufwand der Ausführung von X mit einer Eingabegröße der Bitlänge L angibt. Die Herleitung des Aufwandes soll zunächst unabhängig von der konkreten Realisierung der Arithmetik geführt werden. Denn die Realisierung der Arithmetik ist letztendlich kein Bestandteil des Algorithmus. Daher wird mit $M(L)$ der maximale Aufwand von einer (klassischen) arithmetischen Operation zweier natürlichen Zahlen mit Bitlänge L bezeichnet. Unter einer arithmetischen Operation verstehen wir die Berechnung von Addition, Subtraktion, Produkt, Division oder Divisionsrest(Modulo) zweier natürlicher Zahlen. Analog wird $M_Q(L)$ für den Aufwand arithmetischer Quantenoperationen eingeführt. Zur Verkürzung der Notation werden wir die Abhängigkeit der M-Größen von L nicht mehr explizit ausschreiben und stattdessen $M := M(L)$ und $M_Q := M_Q(L)$ verwenden.[21] Alle Beweise werden zunächst in dieser M-Zeit ausgedrückt. Es sei vorab gesagt, dass diese Operationen alle effizient mit Laufzeiten in $\mathcal{O}(L^2)$ durchgeführt werden können, dass M aber mindestens linear mit L anwächst, da schon das Einlesen einer L-Bit-Zahl L Schritte erfordert. Beginnen wir mit dem ersten Lemma, dass die Bildung der Potenz effizient realisiert.

Lemma 6 *Zur Berechnung der Potenz $a^b \pmod N$ zweier natürlicher Zahlen a und b mit der Bitlänge K von b und der Bitlänge L von N werden $M_{Pot} \in \mathcal{O}(KM)$ elementare klassische Rechenoperationen benötigt.*

Beweis Die Berechnung der Potenz $a^b \pmod N$ über binäre Exponentiation erfordert $\mathcal{O}(\log b)$ Produkte von Potenzen von a. Denn ist $b_{K-1}b_{K-2}...b_1b_0$ die Binärdarstellung von b, dann kann a^b wie folgt berechnet werden:

$$a^b \;=\; a^{b_{K-1}b_{K-2}...b_1b_0} \;=\; a^{\sum_{i=0}^{K-1} b_i 2^i} \;=\; \prod_{i=0}^{K-1} a^{b_i 2^i} \;=\; \prod_{i=0}^{K-1} b_i \cdot a^{2^i}. \qquad (78)$$

[21] Genauso auch für alle weiteren Größen: $M_X := M_X(L)$.

Damit werden $K-1$ Produkte von Potenzen a^{2^i} benötigt. Die Potenzen $k_i := a^{2^i}$ können über die Rekursion $k_i = k_{i-1}^2$ für $i > 0$ und $k_0 = a$ in $K-1$ Schritten berechnet werden. Die Potenz a^b lässt sich dann nach Gleichung (78) mit mindestens $K-1$ Multiplikationen berechnen. Damit werden $2(K-1) \in \mathcal{O}(K)$ Produkte von Potenzen von a und eine Modulo-Operation benötigt, um $a^b \,(\mathrm{mod}\, N)$ zu berechnen. Die Potenzen von a wachsen allerdings sehr schnell an und der Aufwand des angegebenen Algorithmus liegt deutlich über dem zu zeigenden. Anstatt aber die modulo-Operation am Ende des Algorithmus zu benutzen, können wir diese nach jeder Produktbildung verwenden. Damit bleiben alle Faktoren kleiner als N und wir können nach Voraussetzung eine maximale Bitlänge von L für die Faktoren annehmen. Damit ergibt sich ein Gesamtaufwand von $\mathcal{O}(KM)$. □

Nun können beliebige Potenzen modulo N effizient berechnet werden. Eine weitere Operation, die häufig in unserem Algorithmus verwendet wird, ist die Bildung des größten gemeinsamen Teilers. Diese erfolgt über den Euklidischen Algorithmus.

Lemma 7 *Die Berechnung des Euklidischen Algorithmus zu den natürlichen Zahlen a und b mit maximaler Bitlänge L erfordert $M_{EA} \in \mathcal{O}(LM)$ elementare klassische Operationen.*

Beweis Seien a und b zwei natürliche Zahlen. Dann liefert der Euklidische Algorithmus zwei endliche Folgen natürlicher Zahlen $(r_i)_{1\le i\le K}$ und $(a_i)_{0\le i\le K}$ mit $a = a_0 b + r_1$, $b = a_1 r_1 + r_2$ und $r_i = a_{i+1} r_{i+1} + r_{i+2}$ für ein $K \in \mathbb{N}$ und alle $1 \le i \le K-2$. Dafür erfordert der Algorithmus $K+2$ Divisionen mit Rest, die einen Aufwand $\mathcal{O}(M)$ erfordern. Die Folge (r_i) ist streng monoton fallend. Daher gilt für alle $1 \le i \le K-2$

$$r_i = a_{i+1} r_{i+1} + r_{i+2} > (a_{i+1}+1) r_{i+2} \ge 2 r_{i+2}. \tag{79}$$

Damit halbiert sich (r_i) mindestens alle zwei Folgenglieder. Daher gilt

$$KM \in \mathcal{O}(2\log(\max(a,b))M) \subseteq \mathcal{O}(LM). \tag{80}$$

□

Aus der Laufzeitkomplexität des Euklidischen Algorithmus geht auch direkt die Laufzeitkomplexität der Berechnung des größten gemeinsamen Teilers und des kleinsten gemeinsamen Vielfachen hervor.

Korollar 8 *Zur Berechnung des größten gemeinsamen Teilers zweier natürlicher Zahlen a und b mit maximaler Bitlänge L werden $M_{ggT} \in \mathcal{O}(M_{EA}) \subseteq \mathcal{O}(LM)$ elementare klassische Rechenoperationen benötigt.*

Beweis Eine Anwendung des Euklidischen Algorithmus auf a und b liefert Divisionsreste $(r_i)_{1\le i\le K}$ für ein $K \in \mathbb{N}$ für die es natürliche Zahlen $a_0, ..., a_K$ gibt mit

$$a = a_0 b + r_1, \quad b = a_1 r_1 + r_2, \quad r_1 = a_2 r_2 + r_3, \quad ..., \quad r_{K-1} = a_K r_K + 0. \tag{81}$$

Behauptung: r_K ist der größte gemeinsame Teiler von a und b.

Um dies zu zeigen, sei T_n die Menge der Teiler einer natürlichen Zahl n. Die Menge

der gemeinsamen Teiler von a und b ist dann gegeben durch $T_a \cap T_b$. Für beliebige natürliche Zahlen a, b, c, d mit $a = bc + d$ gilt $T_a \cap T_b = T_b \cap T_d$. Denn ist $x \in T_a \cap T_b$, also $x|a$ und $x|b$, dann teilt x auch $d = a - bc$. Genauso gilt die umgekehrte Richtung. Dann gilt aber dem Euklidischen Algorithmus folgend

$$T_a \cap T_b = T_b \cap T_{r_1} = T_{r_1} \cap T_{r_2} = ... = T_{r_{K-1}} \cap T_{r_K} = T_{r_K} \cap T_0. \tag{82}$$

Da alle natürlichen Zahlen die Null teilen, gilt $T_0 = \mathbb{N}$ und es folgt

$$T_a \cap T_b = T_{r_K} \cap \mathbb{N} = T_{r_K}. \tag{83}$$

Der größte gemeinsame Teiler von a und b ist daher der größte Teiler von r_K also r_K selbst.

□

Korollar 9 *Zur Berechnung des kleinsten gemeinsamen Vielfachen zweier natürlicher Zahlen mit maximaler Bitlänge L werden $M_{kgV} \in \mathcal{O}(M_{EA}) \subseteq \mathcal{O}(LM)$ elementare klassische Rechenoperationen benötigt.*

Beweis Seien a und b zwei natürliche Zahlen mit der Bitlänge L, dann lässt sich das kleinste gemeinsame Vielfache (kgV) über

$$\mathrm{kgV}(a, b) = \frac{ab}{\mathrm{ggT}(a, b)} = \frac{a}{\mathrm{ggT}(a, b)} \cdot b \tag{84}$$

mit Hilfe der Berechnung des größten gemeinsamen Teilers von a und b sowie einer Division und einer Multiplikation berechnen.

□

Neben klassischen Berechnungen werden auch Berechnungen mit Hilfe von Quantengattern benötigt. Als Menge der elementaren Quantengatter wird die universelle Familie bestehend aus Hadamard-, Phasenschift- und kontrollierten Phasenschiftgattern verwendet. Dabei ist das Phasenschiftgatter für ein reelles $\theta \in [0, 1)$ definiert über

$$R(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & \exp(2\pi i\theta) \end{pmatrix}. \tag{85}$$

Die Universalität können wir zeigen, indem wir bemerken, dass die Menge bestehend aus cNOT-, Hadamard- und $\pi/8$-Gatter universell ist (siehe Kapitel 2.5.4) und dass das $\pi/8$-Gatter über $T = R(1/8)$ und das cNOT-Gatter über $\mathrm{cNOT} = H\,\mathrm{c}R(1/2)\,H$ durch unsere gewählte universelle Familie dargestellt werden kann.[22] Der Aufwand der Quantenfouriertransformation(QFT) lässt sich durch eine Zerlegung in Hadamard- und Phasenshift-Gattern abschätzen.

Lemma 10 *Die Implementierung der Quantenfouriertransformation auf einem Register mit L Qubits erfordert $\mathcal{O}(L^2)$ elementare Quantengatter.*

[22] $\mathrm{cNOT} = H\,\mathrm{c}R(1/2)\,H$, denn: Ist das kontrollierende Qubit $|0\rangle$, dann folgt $HIH = H^2 = I$. Ist das kontrollierende Qubit $|1\rangle$, dann folgt $HR(1/2)H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = X$.

Beweis Sei $QFT := QFT^{\otimes L}$ das Quantengatter, dass die QFT des L-Qubit-Registers realisiert. Für einen beliebigen Zustand $|x\rangle_L \in \mathcal{H}^{\otimes L}$ lässt sich der Zustand nach Durchführung der QFT wie folgt zerlegen

$$\begin{aligned}
QFT\,|x\rangle_L &= 2^{-\frac{L}{2}} \sum_{y=0}^{2^L-1} \exp\left(2\pi i \frac{xy}{2^L}\right) |y\rangle_L \\
&= 2^{-\frac{L}{2}} \sum_{y_1=0}^{1} \ldots \sum_{y_L=0}^{1} \exp\left(2\pi i x \sum_{j=1}^{L} \frac{y_j}{2^j}\right) |y_1 \ldots y_L\rangle \\
&= 2^{-\frac{L}{2}} \sum_{y_1=0}^{1} \ldots \sum_{y_L=0}^{1} \bigotimes_{j=1}^{L} \left[\exp\left(2\pi i \frac{x y_j}{2^j}\right) |y_j\rangle\right] \\
&= 2^{-\frac{L}{2}} \bigotimes_{j=1}^{L} \left[\sum_{y_j=0}^{1} \exp\left(2\pi i \frac{x y_j}{2^j}\right) |y_j\rangle\right] \\
&= 2^{-\frac{L}{2}} \bigotimes_{j=1}^{L} \left[|0\rangle + \exp\left(2\pi i \frac{x}{2^j}\right) |1\rangle\right].
\end{aligned}$$

Sei nun $x_1 \ldots x_L$ die Bitdarstellung von x und wir vereinbaren $x_1 \ldots x_j.x_{j+1} \ldots x_L := \sum_{k=1}^{L} x_k 2^{j-k}$. Weil $z \to \exp(iz)$ 2π-periodisch ist, gilt dann

$$\begin{aligned}
\exp\left(2\pi i \frac{x}{2^j}\right) &= \exp\left(2\pi i\, x_1 \ldots x_{L-j}.x_{L-j+1} \ldots x_L\right) \\
&= \exp\left(2\pi i\, 0.x_{L-j+1} \ldots x_L\right).
\end{aligned}$$

Mit der Verwendung von kontrollierten Phasenschiftgattern der Form $R_k := R(1/2^k)$ kann die QFT schließlich in Form des in Abbildung 3 dargestellten Schaltkreises implementiert werden.

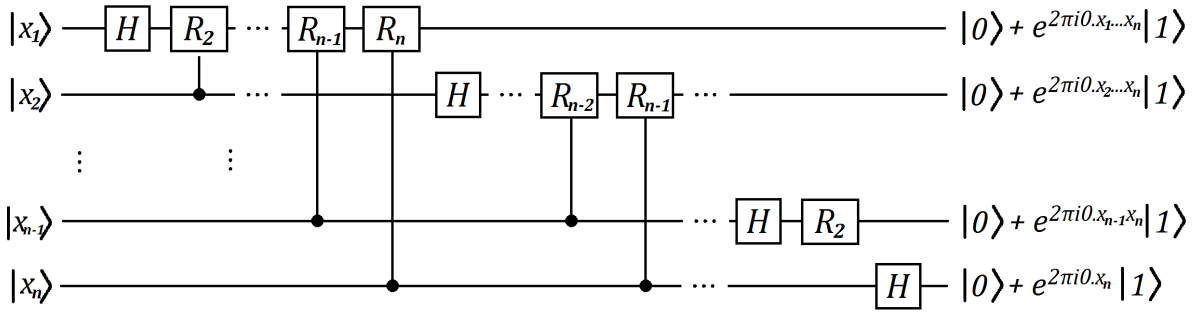

Abbildung 3: *Darstellung des Schaltkreises zur Implementierung der Quantenfouriertransformation. Die Bitdarstellung von x ist wie oben mit $x_1 \ldots x_n$ bezeichnet. Die Kabel repräsentieren die Qubits. Die Kästen repräsentieren die Anwendung des im Kasten gezeigten Operators. Die Verbindungen mit ausgefüllten Punkten • am Ende repräsentieren die Verbindung zum kontrollierenden Qubit.*

Dieser Schaltkreis verwendet n Hadamard- und $\sum_{i=0}^{L-1} i = \frac{L(L-1)}{2} \in \mathcal{O}(L^2)$ Phasenschiftgatter.

□

Abgesehen von der Realisierung der Arithmetik, kann mit den erhaltenen Lemmata nun Satz 5 bewiesen werden. Dabei wird die Variante mit der Verbesserung (*) verwendet.

Lemma 11 *Sei $L = \lceil \log_2(N+1) \rceil$ die Anzahl der Bits, um die zu faktorisierende Zahl N darzustellen, dann werden $\mathcal{O}(L^3 M)$ elementare klassische und $\mathcal{O}(LM_Q)$ elementare quantenmechanische Rechenoperationen benötigt, um den in Kapitel 3.8 aufgeführten Algorithmus (mit (*)) zu realisieren.*

Beweis Wir beweisen zunächst, dass die klassischen Berechnungen in den Schritten 1, 2, 4 und 5 dem Satz genügen. Anschließend wird gezeigt, dass auch die Quantenrechnungen in Schritt 3 dem Satz genügen.
Schritt 1: Die Prüfung, ob N gerade ist, erfordert alleine die Untersuchung des letzten Bits von N, also eine elementare Operation. Die Prüfung, ob N eine perfekte Potenz ist, erfordert nach [25] $(\log L)^{\mathcal{O}(1)}$ oder anders ausgedrückt $\mathcal{O}(poly(\log L))$ Multiplikationen. Dabei ist $poly(x)$ ein Polynom beliebigen Grades in x. Damit ergibt sich für Schritt 1:

$$M_1 \in \mathcal{O}(1 + poly(\log L)M) \subseteq \mathcal{O}(LM). \tag{86}$$

Schritt 2: In Schritt 2 wird der größte gemeinsame Teiler von N und einem beliebigen $a \in \{2, ..., N-1\}$ bestimmt. Da $a < N$, ist die maximale Bitlänge von a und N durch L gegeben und nach Lemma 8 gilt

$$M_2 \in \mathcal{O}(M_{\text{ggT}}) \subseteq \mathcal{O}(LM). \tag{87}$$

Schritt 4: Als Ergebnis der Quantenphasenschätzung erhalten wir eine Zahl $0 \leq y \leq 2^m - 1$ mit $m = 2L + 1 + \lceil \log_2 \left(2 + \frac{1}{2\epsilon}\right) \rceil \in \mathcal{O}(L)$ als Anzahl der Qubits des ersten Registers zur Durchführung der Quantenphasenschätzung. Dementsprechend sind y und 2^m zwei Zahlen mit Bitlänge $\mathcal{O}(L)$. Die Kettenbruchentwicklung von $\frac{y}{2^m}$ kann über den Euklidischen Algorithmus bestimmt werden und erfordert einen Aufwand von $\mathcal{O}(M_{\text{EA}})$. Da wir die Variante (*) betrachten, wird dieser Prozess zweifach ausgeführt. Das heißt wir erhalten zwei Zahlen $1 \leq y, y' \leq 2^m - 1$ mit zwei Kettenbrüchen der Länge K und K'. Nun werden im schlechtesten Fall für alle $k \in \{0, ..., K\}$ und $k' \in \{0, ..., K'\}$ die Bruchdarstellungen der k-ten Konvergenten der Kettenbruchentwicklung bestimmt und das kleinste gemeinsame Vielfache der Nenner dieser Bruchdarstellungen als Ordnung von a modulo N geprüft. Die Bruchdarstellung eines Kettenbruchs lässt sich über den inversen Euklidischen Algorithmus bestimmen. Da die Reihenfolge der Ausführung von Rechenoperationen den Aufwand nicht verändert, haben der Euklidische Algorithmus und seine Inversion den gleichen Aufwand. Die Berechnung des kleinsten gemeinsamen Vielfachen erfordert nach Korollar 9 asymptotisch den gleichen Aufwand wie die Bestimmung des größten gemeinsamen Teilers. Die Prüfung, ob der Kandidat r die Ordnung von a modulo N ist, kann durch einfaches Berechnen der Potenz $a^r \pmod N$ mit Aufwand M_{Pot} erfolgen. Es ist zu beachten, dass die Längen der Kettenbruchentwicklungen K und K' von L abhängen. Nach dem Beweis zum Aufwand des Euklidischen Algorithmus gilt $K, K' \in \mathcal{O}(L)$. Insgesamt ergibt sich daher

$$M_4 \in \mathcal{O}(2M_{\text{EA}} + KM_{\text{EA}} + K'M_{\text{EA}} + KK'(M_{\text{EA}} + M_{\text{Pot}})) \subseteq \mathcal{O}(KK'LM) \subseteq \mathcal{O}(L^3 M). \tag{88}$$

Schritt 5: Die Argumentation in Schritt 5 enthält keine neuen Aussagen. Die Prüfung, ob r gerade ist, benötigt eine elementare klassische Operation. Die Berechnung der Potenzen $a^{\frac{r}{2}} + 1 \pmod N$ und $a^{\frac{r}{2}} - 1 \pmod N$ erfordert nach Lemma 6 einen Aufwand von

M_{Pot}. Denn wegen $r < N$ ist auch die Bitlänge von $\frac{r}{2}$ kleiner als L. Die Konstruktion der Teiler $\text{ggT}(a^{\frac{r}{2}}+1, N)$ und $\text{ggT}(a^{\frac{r}{2}}-1, N)$ erfolgt erneut mittels Euklidischem Algorithmus.

Schritt 3: Es verbleibt noch die Abschätzung des Aufwandes der Quantenphasenschätzung. Die beiden verwendeten Quantenregister bestehen aus $m \in \mathcal{O}(L)$ und L Qubits. Der Ablauf der Quantenphasenschätzung ist die Anwendung des Hadamard-Gatters auf die einzelnen Qubits des ersten Registers (1), die Anwendung der durch das erste Register kontrollierten Potenz von U_a auf das zweite Register (2) und schließlich die Anwendung der inversen Quantenfouriertransformation auf das erste Register (3). (1) erfordert $m \in \mathcal{O}(L)$ elementare Hadamard-Gatter. (3) erfordert nach Lemma 10 $\mathcal{O}(L^2)$ elementare Quantengatter. Für die Vollendung des Beweis fehlt nun allein noch, dass (2) nur $\mathcal{O}(LM_Q)$ elementare Quantengatter benötigt. Der Schritt (2) lässt sich aufteilen in die Anwendung von kontrollierten Potenzen der Form $U_a^{2^j} = U_{a^{2^j}}$ für $0 \leq j \leq m-1$. Die Potenzen a^{2^j} können nach Lemma 6 in $M_{\text{Pot}} \in \mathcal{O}(mM) \subseteq \mathcal{O}(LM)$ klassischen Rechenschritten berechnen. Der Quantenschaltkreis von (2) benötigt nun $\mathcal{O}(m) \subseteq \mathcal{O}(L)$ kontrollierte Anwendungen von U_x. Dabei entsprach die Anwendung von U_x der Multiplikation mit x. Dementsprechend ist die Realisierung von U_x mit einem Aufwand $\mathcal{O}(M_Q)$ verbunden. □

Damit besitzt der klassische Teil des Gesamtalgorithmus einen Aufwand von $\mathcal{O}(L^3M)$ und der quantenmechanische Teil einen Aufwand von $\mathcal{O}(LM_Q)$. Der genaue Aufwand des Gesamtalgorithmus hängt nun von der Realisierung der Arithmetik des Computers ab. Die Verfahren für Addition und Subtraktion sind in der Regel linear in L (bspw. einfaches schriftliches Addieren und Subtrahieren). Die Verfahren für Multiplikation und Division benötigen einen höheren Aufwand.

Für kleine Zahlen: Für kleine Zahlen sind die bekannten Schulbuchverfahren der Arithmetik geeignet. Eine Implementierung der Arithmetik durch einfaches schriftliches Dividieren und Multiplizieren liefert $M \in \mathcal{O}(L^2)$.[30] Für den klassischen Teil ergibt sich damit ein Aufwand von $\mathcal{O}(L^5)$. Diese Verfahren der klassischen Arithmetik lassen sich auf die Arithmetik des Quantencomputers übertragen und liefern den gleichen Aufwand $M_Q \in \mathcal{O}(L^2)$.[31] Für den quantenmechanischen Teil ergibt sich damit ein Aufwand von $\mathcal{O}(L^3)$ für den Gesamtalgorithmus.

Für sehr große Zahlen: Zur Realisierung der Arithmetik gibt es schnelle Verfahren, die annähernd asymptotisch lineare Laufzeiten liefern. Diese Laufzeiten werden allerdings erst bei sehr großen Zahlen erreicht. Das asymptotisch schnellste, die Schönhage-Strassen-Multiplikation, liefert einen Aufwand von $\mathcal{O}(L \log L \log\log L)$.[32] Die Newton-Raphson-Division erfordert $\mathcal{O}(M_P)$ elementare Operationen, wobei M_P den Aufwand der Multiplikation zweier L-Bit-Zahlen bezeichnet.[30, Kapitel 9] Damit folgt ein klassischer Aufwand von $\mathcal{O}(L^4 \log L \log\log L)$. Was die Arithmetik des Quantencomputers anbelangt, so können auch die schnellen klassischen Multiplikationsverfahren wie die Schönhage-Strassen-Multiplikation auf Quantencomputer ohne asymptotische Verluste im Laufzeitaufwand übertragen werden. Denn diese beruhen letztendlich auf elementaren klassischen Gattern, die sich (in reversibler Form) durch eine konstante Anzahl von elementaren Quantengattern simulieren lassen (siehe auch die Originalpublikation von Shor[5]). Der Shor-Algorithmus kann daher mit einem Quanten-Aufwand von $\mathcal{O}(L^2 \log L \log\log L)$ ausgeführt werden. Damit ist Satz 5 bewiesen.

Es ist zu bemerken, dass für den Zwischenbereich mittelgroßer Zahlen eine Vielzahl anderer schneller Verfahren zur Realisierung der (klassischen) Arithmetik existiert. Die genaue Umsetzung der Arithmetik geht aber über den Rahmen dieser Arbeit hinaus. Daher soll die Beschreibung für kleine und sehr große Zahlen genügen.

Es ist außerdem zu bemerken, dass die Realisierung von Satz 5 einen probabilistischen Algorithmus mit einer gewissen Fehlerwahrscheinlichkeit $\epsilon_0 < 1$ liefert. Die Fehlerwahrscheinlichkeit setzt sich zusammen aus der Fehlerwahrscheinlichkeit der Quantenphasenschätzung ϵ und der Wahrscheinlichkeit, dass die Ordnung ungerade ist. Allerdings lässt sich ausgehend von ϵ_0 jede beliebige Fehlerwahrscheinlichkeit ϵ' durch eine konstante Anzahl von Wiederholungen des Algorithmus erreichen. Daher kann der Algorithmus ohne zusätzlichen asymptotischen Aufwand durch Wiederholung mit beliebig hohen Sicherheitswahrscheinlichkeiten ausgeführt werden.

5 Schwierigkeiten bei der Umsetzung - Konstruktion von Quantencomputern

Nachdem der größte Teil dieser Arbeit der rein theoretischen Beschreibung des Shor-Algorithmus diente, soll es nun um die praktische Umsetzung des Algorithmus gehen. Denn es mag sich die Frage stellen, wieso der Shor-Algorithmus trotz seines umfassenden Geschwindigkeitsvorteils und der Publikation vor bereits mehr als 20 Jahren noch keinen Eingang in die Praxis gefunden hat. Es gibt offensichtlich einen wesentlichen Haken: Die Konstruktion des Quantencomputers, der wesentlichen Voraussetzung des Algorithmus, bereitet große Probleme. Diese Probleme sollen hier aufgegriffen und in grundlegender Form erklärt werden. Es ist zunächst zu klären, welcher Voraussetzungen es für die Konzipierung eines Quantencomputers bedarf. Das Konzept einer solchen Rechenmaschine sollte dabei vor allem die Realisierung des Speichers und die Realisierung der auf dem Speicher durchführbaren Operationen erklären. Ein quantenmechanischer Speicher muss also letztendlich folgenden Anforderungen genügen[23]

1. Robuste Darstellung quantenmechanischer Information
2. Präparation eines initialen Zustandes als Referenzpunkt
3. Durchführbarkeit unitärer Transformationen
4. Durchführbarkeit einer quantenmechanischen Messung

Generell wird von dem Konzept eines Quantenrechners für einen praktikablen Einsatz außerdem eine gewisse Skalierbarkeit erwartet. Das heißt, dass die Architektur des Rechners auch für die Zusammenschaltung vieler Qubits funktionsfähig sein muss (Interconnection-Problem). Da der aktuelle Stand der Forschung allerdings noch weit entfernt von der Realisierung groß skalierter Quantenrechner ist, sollten auch wir zunächst klein beginnen und vor allem die Schwierigkeiten behandeln, die bereits auftreten, wenn nur ein einzelnes Qubit den Speicher unseres Quantencomputers konstituiert. Widmen wir uns zunächst der genaueren Spezifizierung der genannten vier Voraussetzungen.

Damit das betrachtete System einen Qubit repräsentieren kann, muss es sich effektiv wie ein Zwei-Zustands-System verhalten. Das soll bedeuten, dass das repräsentierende System im Wesentlichen nur zwei Zustände einnehmen kann, welche mit $|0\rangle$ und $|1\rangle$ identifiziert werden. Alle anderen möglichen Zustände des Systems sollten nicht erreichbar sein bzw. nur mit vernachlässigbar kleiner Wahrscheinlichkeit erreicht werden. In dieser Hinsicht stellt ein Spin-1/2-System (eine genügend gute Isolation vorausgesetzt) ein ideales Qubit dar, während ein harmonischer Oszillator, dessen zwei niedrigste Energiezustände den Zuständen $|0\rangle$ und $|1\rangle$ des Qubits zugeordnet werden, kein gutes Qubit darstellt, da höhere Energiezustände leicht erreicht werden können.[24] Ist eine Repräsentation des Qubits gefunden, so ist es wichtig, dass diese Repräsentation robust ist. Damit ist gemeint, dass der

[23] Die genannten Voraussetzungen entsprechen mit leichten Abwandlungen der Auflistung aus [16, Kapitel 7.2].

[24] Der Energieunterschied zwischen zwei Energiezuständen des harmonischen Oszillators entspricht $\hbar\omega$, wobei ω die Eigenfrequenz des Oszillators ist. Damit kann jede Wechselwirkung, die Übergänge zwischen

gespeicherte Zustand genügend gut isoliert ist von der Umgebung. Ist das System nicht genügend gut isoliert, so können Wechselwirkungen zwischen dem Qubitsystem und der Umgebung stattfinden. Diese Wechselwirkungen verändern den Zustand des Qubitsystems und machen die Kohärenz des Zustandes zunichte. Damit verliert das Qubitsystem die wesentlichen Eigenschaften, die Quantenrechnern ihren Geschwindigkeitsvorteil gegenüber klassischen Rechenmaschinen geben; die Überlagerung, Verschränkung und Interferenz von Zuständen ist nur auf kurzen Zeitskalen möglich. Effekte, die dies hervorrufen, werden allgemein unter dem Begriff der Dekohärenz zusammengefasst. Ein Maß für die Größe der Störung durch Dekohärenzeffekte ist die Kohärenzzeit, das heißt die mittlere Zeit, in der quantenmechanische Zustände des Systems kohärent bleiben. Ziel einer robusten Repräsentierung der quantenmechanischen Information muss es also sein, Dekohärenzeffekte zu minimieren, sodass eine (im Vergleich zur Operationszeit der Quantengatter) lange Dekohärenzzeit erreicht wird.

Zur Durchführung eines Algorithmus wird zunächst ein definierter Anfangszustand benötigt, von dem aus die Berechnungen durchgeführt werden können. Dieser Anfangszustand dient dann gewissermaßen als Referenzpunkt für alle davon ausgehenden Operationen. Die Wahl eines solchen Referenzpunktes ist prinzipiell beliebig, da von diesem Zustand ausgehend alle anderen gewünschten Anfangszustände durch Anwendung unitärer Transformationen erhalten werden können. Als Gütemaß für die Präparation des Anfangszustandes kann die Sicherheitswahrscheinlichkeit gewählt werden.

Die letzten beiden Voraussetzungen betreffen die Durchführbarkeit aller Operationen, die ein Quantenalgorithmus für seine Ausführung womöglich benötigt. Dabei lässt sich die Voraussetzung der Durchführbarkeit beliebiger unitärer Transformationen reduzieren auf die Voraussetzung der Durchführbarkeit einer effizienten universellen Familie unitärer Transformationen (siehe Kapitel 2.5.4). Denn eine effiziente universelle Familie unitärer Transformationen kann jede unitäre Transformation durch Hintereinanderausführung von Elementen der Familie mit beliebiger Genauigkeit effizient approximieren. Als Beispiel genügt die Durchführbarkeit von Hadamard-, Phasen-, cNOT- und $\pi/8$-Gatter.[25] Schließlich muss in irgendeiner Form das Ergebnis einer Berechnung aus dem quantenmechanischen Speicher extrahiert werden können. Dazu ist es notwendig, eine quantenmechanische Messung an dem Speicher durchführen zu können.

Für die physikalische Realisierung eines Qubits gibt es verschiedene experimentelle Ansätze. Zunächst gilt es ein quantenmechanisches System zu finden, dass im Wesentlichen nur zwei Zustände annehmen kann, gut manipulierbar und gleichzeitig stabil, das heißt genügend lang kohärent, ist. Gerade die letzten beiden Voraussetzungen stehen sich im Wege und stellen die große Herausforderung an das System. Es soll einerseits genügend isoliert von der Umgebung sein, um Störungen innerhalb der relevanten Zeitskalen gering zu halten und andererseits für Manipulationen zugänglich sein.

$|0\rangle$ und $|1\rangle$ induziert, auch Übergänge in höhere Energiezustände verursachen.

[25] Im Unterkapitel 2.5.4 ist das Phasengatter nicht mit enthalten. Der Grund für das Hinzufügen ist, dass das Phasengatter für eine fehlertolerante Implementierung beliebiger unitärer Transformationen benötigt wird. Denn in der Praxis bestehen die Quantenschaltkreise aus fehlerbehafteten Bauteilen. Damit die Fehler der einzelnen Komponenten sich nicht aufschaukeln werden diese fehlertolerant implementiert.

Die Ansätze, die zurzeit die größte Verbreitung zeigen, sind vermutlich photonische Systeme, Ionenfallen und magnetische Kernspinresonanz. Photonische Systeme bieten mehrere Möglichkeiten ein Qubit zu realisieren. So kann ein Qubit durch die Polarisation des Lichtes (horizontal: $|0\rangle$, vertikal $|1\rangle$) oder das Vorhandensein eines Photons (Vakuum: $|0\rangle$, Photon: $|1\rangle$) repräsentiert werden. Die Vorteile photonischer Systeme sind, dass Photonen mit der Umgebung in der Regel nur schwach wechselwirken und daher sehr lange Kohärenzzeiten besitzen. Außerdem stellt die angewandte optische Physik eine Vielzahl von Methoden zur Verfügung Photonen gezielt zu manipulieren. Schwierig dagegen, ist es, Wechselwirkungen zwischen mehreren Photonen zu generieren. Dazu benötigt es Effekte der nichtlinearen Optik wie den Kerr-Effekt; allerdings sind diese Wechselwirkungen meist nur sehr schwach auf Kosten hoher Absorptionsraten.

Ionenfallen sind Systeme, in denen eine Anzahl weniger Ionen in einem elektromagnetischen hochfrequenten Wechselfeld gefangen gehalten wird. Die eingefangenen Ionen werden durch verschiedene Verfahren der Laserkühlung bis zu sehr tiefen Temperaturen herabgekühlt. Die Qubits werden durch bestimmte Energiezustände der einzelnen Ionen repräsentiert. Übergänge zwischen den Energiezuständen werden durch gezielte Laserpulse hervorgerufen. Eine wichtige Voraussetzung für gezielte Manipulationen durch Laser ist die Selektierbarkeit der einzelnen Ionen. Dazu muss der Abstand der Ionen größer als die Wellenlänge des Laserlichtes sein. Bei der Erhöhung der Anzahl der Ionen in einer Falle wird dies sehr schnell sehr schwierig.

Der Ansatz der Kernspinresonanz beruht auf der Tatsache, dass Kernspinresonanz durch ihre medizinische Anwendung bereits sehr gut erforscht ist und präzise angewendet werden kann. Die Grundidee ist es, das Qubit durch einen Kernspin zu repräsentieren. Da die Wechselwirkung von Kernspins verschiedener Atomkerne sehr gering ist, lassen sich hohe Kohärenzzeiten realisieren. Leider ist die Isolation von der Umgebung so groß, dass Messungen an einzelnen Atomkernen kaum möglich sind. Aus diesem Grund setzen Quantenberechnungen mit Hilfe von Kernspinresonanzen auf ein Ensemble von Quantencomputern. Die kollektive Auswertung der Berechnungen vieler Quantencomputer, das heißt die Messung eines Gesamtspins des Ensembles, liefert schließlich messbare Größen. Allerdings handelt es sich bei den Messwerten um statistische Mittelwerte der Berechnungen vieler Quantencomputer. Daher sind Quantencomputer, die auf Kernspinresonanzen basieren, nur in der Lage Quantenalgorithmen zu berechnen, deren Ergebnis auch als Mittelwert das betrachtete Problem löst. Im Falle des Shor-Algorithmus wäre das Ergebnis $\langle \frac{s}{r} \rangle$ bedeutungslos ($\langle \cdot \rangle$ repräsentiert die Mittelung über das Ensemble). Der Shor-Algorithmus ließe sich aber trotzdem durchführen, indem das klassische Post-Processing vorgezogen wird. Gemeint ist, dass jeder einzelne Quantencomputer das klassische Post-Processing durchführt und nur bei gefundener Ordnung r ein Ergebnis liefert. Neben den genannten experimentellen Ansätzen gibt es noch eine Vielzahl weiterer Verfahren, die im Hinblick auf die Realisierung eines Quantencomputers erforscht werden. Dies ist darauf zurückzuführen, dass letztendlich jedes quantenmechanische System prinzipiell dazu in der Lage ist einen Quantencomputer zu realisieren. Da aber jedes dieser einzelnen Ansätze noch viele Hürden zu überwinden hat, wird es wohl noch einige Zeit dauern, bis Quantencomputer (im Sinne des Shor-Algorithmus[26]) gewinnbringend eingesetzt werden können.

[26] In anderen Bereichen der Quanteninformatik gibt es Algorithmen, die schon bei der Zusammenschaltung weniger Qubits in der Praxis Vorteile gegenüber klassischen Rechenmaschinen besitzen können. Zu

Literatur

[1] Gordon E. Moore: *Cramming More Components onto Integrated Circuits*, Electronics, 38(8):114-117, 1965.

[2] Intel Corporation, *Intel 14-nm-Technik*, `intel.de/content/www/de/de/silicon-innovations/intel-14nm-technology.html?wapkw=14+nm&_ga=1.186524717.564477432.1470215239`, abgerufen am 3. Aug. 2016, 11:12 Uhr.

[3] Paul A. Benioff: *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines*, J. Statist. Phys., 22(5):563–591, 1980.

[4] Paul A. Benioff: *Quantum mechanical Hamiltonian models of Turing machines*, J. Statist. Phys., 29(3):515–546, 1982.

[5] Peter W. Shor: *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM J. Comput., 26(5):1484-1509, 1997.

[6] David Deutsch, Richard Jozsa: *Rapid solutions of problems by quantum computation*, Proc. R. Soc. Lond. A, 439(1907):553-558, 1992.

[7] Lov K. Grover: *A fast quantum mechanical algorithm for database search*, Proc. 28$^{\text{th}}$ Annual ACM Symposium on the Theory of Computing, S. 212-219, 1996. arXiv:quant-ph/9605043.

[8] Lieven M. K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Mark H. Sherwood, Isaac L. Chuang: *Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance*, Nature, 414:883-887, 2001.

[9] Chao-Yang Lu, Daniel E. Browne, Tao Yang, Jian-Wei Pan: *Demonstration of a Compiled Version of Shor's Quantum Factoring Algorithm Using Photonic Qubits*, Phys. Rev. Lett., 99(25):250504, 2007.

[10] Ben P. Lanyon, Till J. Weinhold, Nathan K. Langford, Marco Barbieri, Daniel F. V. James, Adam Gilchrist, Andrew G. White: *Experimental demonstration of Shor's algorithm with quantum entanglement*, Phys. Rev. Lett., 99(25):250505, 2007.

[11] Alberto Politi, Jonathan C. F. Matthews, Jeremy L. O'Brien: *Shor's quantum factoring algorithm on a photonic chip*, Science, 325(5945):1221, 2009.

[12] Erik Lucero, Rami Barends, Yu Chen, Julian Kelly, Matteo Mariantoni, Anthony Megrant, Peter O'Malley, Daniel Sank, Amit Vainsencher, James Wenner, Ted White, Yi Yin, Andrew N. Cleland, John M. Martinis: *Computing prime factors with a Josephson phase qubit quantum processor*, Nature Physics, 8(10):719-723, 2012.

[13] Enrique Martín-López, Anthony Laing, Thomas Lawson, Roberto Alvarez, Xiao-Qi Zhou, Jeremy L. O'Brien: *Experimental realization of Shor's quantum factoring algorithm using qubit recycling*, Nature Photonics, 6:773-776, 2012.

nennen ist hier vor allem die Quantenkryptographie, die vor allem auf der Verschränkung von Zuständen beruht, die mit klassischen Rechenmaschinen nicht durchführbar ist.

[14] Phillip R. Kaye, Raymond Laflamme, Michele Mosca: *An Introduction to Quantum Computing*, Oxford University Press, New York, 2007.

[15] Arjen K. Lenstra, Hendrik W. Lenstra Jr.: *The development of the number field sieve*, Lecture Notes in Mathematics, Vol. 1554, Springer Verlag, Berlin Heidelberg, 1993.

[16] Michael A. Nielsen, Isaac L. Chuang: *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, 2000.

[17] Thomas F. Sturm, Jörg Schulze: *Quantum Computation aus algorithmischer Sicht*, Oldenbourg Wissenschaftsverlag GmbH, München, 2009.

[18] John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullmann: *Intoduction to Automata Theory, Languages, and Computation*, Addison Wesley, Boston [u.a.], 3. Auflage, 2007.

[19] David P. DiVincenzo: *Two-Bit Gates are Universal for Quantum Computation*, Phys. Rev. A, 51(2):1015, 1995.

[20] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John Smolin, Harald Weinfurter: *Elementary gates for quantum computation*, Phys. Rev. A, 52(5):3457, 1995.

[21] P. Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, Farrokh Vatan: *On Universal and Fault-Tolerant Quantum Computing*, arXiv:quant-ph/9906054, 1999.

[22] Alexey Y. Kitaev: *Quantum computations: algorithms and error correction*, Russ. Math. Surv., 53(6):1191-1249, 1997.

[23] Robert M. Solovay: *Lie Groups and Quantum Circuits*, Vortrag am MSRI, Berkeley, 2000, Aufz. des Vortrages unter `msri.org/realvideo/ln/msri/2000/qcomputing/solovay/1/index.html`, abgerufen am 31. Jul. 2016.

[24] Christopher M. Dawson, Michael A. Nielsen: *The Solovoy-Kitaev Algorithm*, Jour. Quantum Info. Comput., 6(1):81-95, 2006.

[25] Daniel J. Bernstein, Hendrik W. Lenstra Jr., Jonathan Pila: *Detecting Perfect Powers by Factoring into Coprimes*, Math. Comput., 76(257):385-388, 2007.

[26] Donald E. Knuth: *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, 3. Auflage, Addison-Wesley, Boston, 1998.

[27] Alexey Y. Kitaev: *Quantum measurements and the Abelian Stabilizer Problem*, El. Coll. on Comput. Compl., Vol. 3, 1996. arXiv:quant-ph/9511026.

[28] Godfrey H. Hardy, Sir Edward M. Wright: *An Introduction to the Theory of Numbers*, 5. Auflage, The Clarendon Press Oxford University Press, New York, 1979.

[29] Harald Scheid, Andreas Frommer: *Zahlentheorie*, Springer Spektrum, Berlin [u.a.], 4. Auflage, 2013.

[30] Joachim v. zur Gathen, Jürgen Gerhard: *Modern Computer Algebra*, Cambridge University Press, Cambridge, 2003.

[31] Vlatko Vedral, Adriano Barenco, Artur Ekert: *Quantum Networks for Elementary Arithmetic Operations*, Phys. Rev. A, 54(1):147-153, 1996.

[32] Arnold Schönhage, Volker Strassen: *Schnelle Multiplikation großer Zahlen*, Computing, Springer, 7(3):281-292, 1971.