Ein praxisorientierter Report über Quantencomputing: Umfassende mathematische Behandlung

Henning Sarrus

24. September 2025

Inhaltsverzeichnis

1 Einleitung

Das Quantencomputing repräsentiert einen grundlegenden Wandel in der Informationsverarbeitung. Im Gegensatz zu klassischen Computern, die auf Bits basieren, die entweder den Wert 0 oder 1 annehmen können, verwenden Quantencomputer sogenannte Qubits. Diese Qubits können, basierend auf den Prinzipien der Quantenmechanik wie Superposition und Verschränkung, gleichzeitig mehrere Zustände repräsentieren. Diese Fähigkeit ermöglicht es Quantencomputern, bestimmte Rechenprobleme exponentiell schneller zu lösen als klassische Computer, was das Potenzial für bahnbrechende Fortschritte in Bereichen wie Materialforschung, Medikamentenentwicklung, Finanzmodellierung und Kryptografie eröffnet.

Die Entwicklung des Quantencomputings ist ein interdisziplinäres Feld, das Erkenntnisse aus Physik, Mathematik, Informatik und Ingenieurwissenschaften vereint. Obwohl die Technologie noch in der Entwicklung steckt, wurden bereits bedeutende Fortschritte erzielt, und sowohl groSSe Technologieunternehmen als auch Forschungseinrichtungen investieren erheblich in die Entwicklung von Quantenhardware und -software. Dieser Bericht hat das Ziel, die grundlegenden Konzepte des Quantencomputings zu erklären, mit einem besonderen Schwerpunkt auf der Quantenverschränkung und ihrer mathematischen Formulierung, sowie praktische Beispiele und Anwendungsbereiche zu präsentieren.

1.1 Was ist Quantencomputing?

Quantencomputing ist ein neuartiges Rechenparadigma, das die Gesetze der Quantenmechanik nutzt, um komplexe Probleme zu lösen, die für klassische Computer unlösbar oder extrem zeitaufwendig wären. Im Kern des Quantencomputings stehen die Qubits, die im Gegensatz zu klassischen Bits nicht nur die Zustände 0 oder 1 annehmen können, sondern auch eine Überlagerung (Superposition) beider Zustände gleichzeitig. Dies bedeutet, dass ein Qubit nicht nur 0 oder 1 ist, sondern eine Kombination aus beiden, repräsentiert durch eine Wahrscheinlichkeitsverteilung. Erst bei der Messung kollabiert das Qubit in einen der klassischen Zustände.

1.2 Abgrenzung zum klassischen Computing

Der wesentliche Unterschied zwischen klassischem und Quantencomputing liegt in der Art der Informationsverarbeitung. Klassische Computer verarbeiten Informationen sequenziell und basieren auf Transistoren, die binäre Zustände (0 oder 1) repräsentieren. Ihre Rechenleistung skaliert linear mit

der Anzahl der Bits. Quantencomputer hingegen nutzen die besonderen Eigenschaften der Quantenmechanik:

- **Superposition:** Ein Qubit kann den Zustand 0, 1 oder eine beliebige Kombination aus beidem annehmen. Dadurch kann ein System aus n Qubits 2^n Zustände gleichzeitig repräsentieren, was eine exponentielle Erhöhung der Informationsdichte im Vergleich zu klassischen Bits bedeutet.
- **Verschränkung:** Zwei oder mehr Qubits können miteinander verschränkt werden, sodass der Zustand eines Qubits unmittelbar mit dem Zustand der anderen korreliert ist, selbst wenn sie räumlich getrennt sind. Dieses Phänomen ist entscheidend für die Leistungsfähigkeit von Quantenalgorithmen.
- **Interferenz:** Analog zu Wellen können Wahrscheinlichkeitsamplituden in Quantensystemen konstruktiv oder destruktiv interferieren. Dies wird in Quantenalgorithmen genutzt, um die Wahrscheinlichkeit der korrekten Lösung zu verstärken und die Wahrscheinlichkeit falscher Lösungen zu minimieren.

Diese quantenmechanischen Phänomene ermöglichen es Quantencomputern, bestimmte Problemklassen, wie die Faktorisierung groSSer Zahlen (Shor-Algorithmus) oder die Suche in unsortierten Datenbanken (Grover-Algorithmus), deutlich effizienter zu lösen als klassische Computer.

1.3 Anwendungsbereiche und Potenzial

Das Quantencomputing birgt ein enormes Potenzial, das sich auf vielfältige Bereiche erstreckt:

- **Materialwissenschaft und Chemie:** Die Simulation komplexer Moleküle und Materialien könnte die Entwicklung neuer Medikamente, effizienterer Katalysatoren oder Hochtemperatur-Supraleiter ermöglichen.
- **Kryptographie:** Die Fähigkeit, groSSe Zahlen zu faktorisieren, stellt eine Bedrohung für aktuelle Verschlüsselungsmethoden wie RSA dar und erfordert gleichzeitig die Entwicklung neuer, quantensicherer Verfahren.
- **Optimierung:** Komplexe Optimierungsprobleme in Bereichen wie Logistik, Finanzwesen und künstlicher Intelligenz könnten effizienter gelöst werden.

- **Künstliche Intelligenz und Maschinelles Lernen:** Algorithmen für maschinelles Lernen könnten beschleunigt und neue KI-Modelle entwickelt werden.
- **Finanzmodellierung:** Die Simulation komplexer Finanzmärkte könnte zu einer präziseren Risikobewertung und einer optimierten Portfolio-Zusammenstellung führen.

Auch wenn viele dieser Anwendungen noch in der Zukunft liegen, wird intensiv daran geforscht, die theoretischen Vorteile des Quantencomputings in konkrete Lösungen umzusetzen.

2 Grundlagen des Quantencomputings

Um die Funktionsweise eines Quantencomputers zu verstehen, ist es unerlässlich, sich mit den grundlegenden Konzepten der Quantenmechanik vertraut zu machen, die seine Rechenleistung ermöglichen. Die wichtigsten Bausteine sind Qubits, Superposition, Verschränkung und Quantenlogikgatter.

2.1 Qubits und Superposition

Das **Qubit** (Quantum Bit) ist die grundlegende Informationseinheit im Quantencomputing. Im Gegensatz zu einem klassischen Bit, das entweder den Zustand 0 oder 1 annehmen kann, kann ein Qubit auch eine **Superposition** dieser beiden Zustände sein. Mathematisch wird der Zustand eines einzelnen Qubits als linearer Überlagerungszustand der Basiszustände $|0\rangle$ und $|1\rangle$ dargestellt:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

wobei α und β komplexe Zahlen sind, die als **Wahrscheinlichkeitsamplituden** bezeichnet werden. Die Quadrate ihrer Beträge geben die Wahrscheinlichkeiten an, das Qubit bei einer Messung im Zustand $|0\rangle$ bzw. $|1\rangle$ vorzufinden:

$$P(0) = |\alpha|^2 \quad \text{und} \quad P(1) = |\beta|^2$$

Da die Summe der Wahrscheinlichkeiten 1 ergeben muss, gilt die Normierungsbedingung:

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

Beispiel 1 (Konkrete Qubit-Zustände)**.** *Betrachten wir einige wichtige Einzelqubit-Zustände:*

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{1}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{2}$$

$$|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle) \tag{3}$$

$$|-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle) \tag{4}$$

Für den Zustand $|+\rangle$ *sind* $\alpha = \beta = \frac{1}{\sqrt{2}}$*, und die Messwahrscheinlichkeiten sind:*

$$P(0) = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}, \quad P(1) = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$$

2.1.1 Die Bloch-Kugel: Geometrische Darstellung von Qubits

Jeder Qubit-Zustand kann geometrisch auf der **Bloch-Kugel** visualisiert werden. Die allgemeine Parameterisierung eines Qubits lautet:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

wobei $\theta \in [0, \pi]$ der Polarwinkel und $\phi \in [0, 2\pi)$ der Azimutwinkel ist.

Rechnung 1 (Bloch-Kugel Koordinaten)**.** *Der Bloch-Vektor* $\vec{r} = (x, y, z)$ *eines Qubits ist gegeben durch:*

$$x = \sin\theta\cos\phi \tag{5}$$

$$y = \sin\theta\sin\phi \tag{6}$$

$$z = \cos\theta \tag{7}$$

Für den Zustand $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *ist* $\theta = \frac{\pi}{2}$ *und* $\phi = 0$*:*

$$x = \sin\left(\frac{\pi}{2}\right)\cos(0) = 1 \tag{8}$$

$$y = \sin\left(\frac{\pi}{2}\right)\sin(0) = 0 \tag{9}$$

$$z = \cos\left(\frac{\pi}{2}\right) = 0 \tag{10}$$

Also liegt $|+\rangle$ *am Punkt* $(1, 0, 0)$ *auf der Bloch-Kugel.*

2.2 Quantenlogikgatter - Mathematische Behandlung

Quantenlogikgatter sind unitäre Transformationen, die auf Qubits angewendet werden. Ein unitärer Operator U erfüllt die Bedingung $U^\dagger U = I$, wobei $U^\dagger$ der adjungierte Operator ist.

2.2.1 Einzelqubit-Gatter

Pauli-Gatter: Die Pauli-Matrizen bilden zusammen mit der Identitätsmatrix eine vollständige Basis für 2×2-Matrizen:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{11}$$

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{12}$$

Rechnung 2 (Anwendung des X-Gatters)**.** *Das X-Gatter (Pauli-X oder NOT-Gatter) wirkt als Bit-Flip:*

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{13}$$

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

Für einen allgemeinen Zustand $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$:

$$X|\psi\rangle = X(\alpha|0\rangle + \beta|1\rangle) = \alpha X|0\rangle + \beta X|1\rangle = \alpha|1\rangle + \beta|0\rangle = \beta|0\rangle + \alpha|1\rangle$$

Hadamard-Gatter: Das Hadamard-Gatter erzeugt Superposition:

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

Rechnung 3 (Hadamard-Transformationen)**.**

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \tag{15}$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle \tag{16}$$

Die Umkehroperation:

$$H|+\rangle = H \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(H|0\rangle + H|1\rangle) = \frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) \tag{17}$$

$$= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) \tag{18}$$

$$= \frac{1}{2}(2|0\rangle) = |0\rangle \tag{19}$$

Rotationsgatter: Rotationen um die Bloch-Kugel-Achsen sind fundamental:

$$R_x(\theta) = \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) X = \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \tag{20}$$

$$R_y(\theta) = \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) Y = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \tag{21}$$

$$R_z(\theta) = \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) Z = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix} \tag{22}$$

2.2.2 Zweitqubit-Gatter

CNOT-Gatter: Das kontrollierte NOT-Gatter ist fundamental für die Erzeugung von Verschränkung:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Rechnung 4 (CNOT-Operationen)**.** *Das CNOT-Gatter wirkt auf die Basiszustände wie folgt:*

$$CNOT|00\rangle = |00\rangle \tag{23}$$

$$CNOT|01\rangle = |01\rangle \tag{24}$$

$$CNOT|10\rangle = |11\rangle \tag{25}$$

$$CNOT|11\rangle = |10\rangle \tag{26}$$

Auf einen Superpositionszustand:

$$CNOT(|+\rangle \otimes |0\rangle) = CNOT\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle\right) \tag{27}$$

$$= CNOT\left(\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)\right) \tag{28}$$

$$= \frac{1}{\sqrt{2}}(CNOT|00\rangle + CNOT|10\rangle) \tag{29}$$

$$= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{30}$$

Dies ist ein verschränkter Bell-Zustand!

2.3 Messung und Born'sche Regel

Die Messung ist der Prozess, bei dem Quanteninformation in klassische Information umgewandelt wird. Die Born'sche Regel bestimmt die Wahrscheinlichkeiten.

Definition 1 (Projektive Messung)**.** *Eine projektive Messung wird durch eine Menge von Projektoren $\{P_m\}$ beschrieben, die folgende Eigenschaften erfüllen:*

1. $P_m^\dagger = P_m$ *(hermitesch)*
2. $P_m^2 = P_m$ *(idempotent)*
3. $\sum_m P_m = I$ *(vollständig)*
4. $P_m P_n = \delta_{mn} P_m$ *(orthogonal)*

Rechnung 5 (Messung in der Standardbasis)**.** *Für die Messung in der Standardbasis $\{|0\rangle, |1\rangle\}$ sind die Projektoren:*

$$P_0 = |0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \tag{31}$$

$$P_1 = |1\rangle\langle 1| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{32}$$

Für den Zustand $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$:

$$P(0) = \langle\psi| P_0 |\psi\rangle = (\alpha^* \langle 0| + \beta^* \langle 1|)|0\rangle\langle 0|(\alpha|0\rangle + \beta|1\rangle) = |\alpha|^2 \tag{33}$$

$$P(1) = \langle\psi| P_1 |\psi\rangle = (\alpha^* \langle 0| + \beta^* \langle 1|)|1\rangle\langle 1|(\alpha|0\rangle + \beta|1\rangle) = |\beta|^2 \tag{34}$$

Nach der Messung von Ergebnis m ist der Zustand:

$$|\psi'\rangle = \frac{P_m|\psi\rangle}{\sqrt{\langle\psi| P_m |\psi\rangle}}$$

2.4 Dekohärenz und Quantenfehler

Dekohärenz beschreibt den Verlust der Quanteneigenschaften durch Umgebungseinflüsse. Mathematisch wird dies durch nicht-unitäre Evolutionen beschrieben.

Amplitude Damping: Modelliert Energieverlust durch spontane Emission:

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix} \tag{35}$$

$$K_1 = \begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix} \tag{36}$$

mit der Dämpfungsrate $\gamma \in [0, 1]$.

Depolarisierender Kanal: Wandelt einen reinen Zustand mit Wahrscheinlichkeit p in das maximale Gemisch um:

$$\mathcal{E}(\rho) = (1-p)\rho + \frac{p}{4}(X\rho X + Y\rho Y + Z\rho Z + \rho)$$

Rechnung 6 (Depolarisation eines reinen Zustands)**.** *Für* $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ *und* $\rho = |\psi\rangle\langle\psi|$*:*

$$\rho = \begin{pmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{pmatrix}$$

Nach dem depolarisierenden Kanal:

$$\mathcal{E}(\rho) = (1-p)\begin{pmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{pmatrix} + \frac{p}{2}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Die Kohärenzen (Off-Diagonal-Elemente) werden um den Faktor $(1-p)$ *reduziert.*

3 Quantenverschränkung - Umfassende mathematische Behandlung

Die Quantenverschränkung ist eines der faszinierendsten und zugleich rätselhaftesten Phänomene der Quantenmechanik. Sie beschreibt einen Zustand, in dem zwei oder mehr Quantenteilchen so miteinander verbunden sind, dass der Zustand jedes Teilchens nicht unabhängig von den Zuständen der anderen beschrieben werden kann.

3.1 Mathematische Definition der Verschränkung

Definition 2 (Separable und verschränkte Zustände)**.** *Ein Zustand $|\psi\rangle_{AB}$ eines zusammengesetzten Systems ist **separabel**, wenn er sich als Tensorprodukt schreiben lässt:*

$$|\psi\rangle_{AB} = |\phi\rangle_A \otimes |\chi\rangle_B$$

*Ein Zustand ist **verschränkt**, wenn er nicht separabel ist.*

Für gemischte Zustände ist ein Zustand ρ_{AB} separabel, wenn er sich als konvexe Kombination von Produktzuständen schreiben lässt:

$$\rho_{AB} = \sum_i p_i \rho_i^A \otimes \rho_i^B$$

mit $p_i \geq 0$ und $\sum_i p_i = 1$.

3.2 Bell-Zustände: Maximale Verschränkung

Die vier Bell-Zustände bilden eine vollständige orthonormale Basis für den zweidimensionalen Verschränkungsraum:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{37}$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{38}$$

$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{39}$$

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \tag{40}$$

Rechnung 7 (Erzeugung von Bell-Zuständen)**.** *Ausgehend vom Produktzustand $|00\rangle$:*

1. Hadamard auf das erste Qubit:

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

2. CNOT-Gatter:

$$CNOT \cdot \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = |\Phi^+\rangle$$

Für $|\Phi^-\rangle$: Zusätzlich Z-Gatter auf das erste Qubit. Für $|\Psi^+\rangle$: Zusätzlich X-Gatter auf das zweite Qubit. Für $|\Psi^-\rangle$: Zusätzlich X- und Z-Gatter.

3.3 Schmidt-Zerlegung

Theorem 1 (Schmidt-Zerlegung)**.** *Jeder reine Zustand $|\psi\rangle_{AB}$ eines zusammengesetzten Systems kann eindeutig als geschrieben werden:*

$$|\psi\rangle_{AB} = \sum_{i=0}^{\min(d_A,d_B)-1} \lambda_i |u_i\rangle_A \otimes |v_i\rangle_B$$

wobei $\lambda_i \geq 0$ die Schmidt-Koeffizienten sind, $\{|u_i\rangle_A\}$ und $\{|v_i\rangle_B\}$ orthonormale Basen darstellen, und $\sum_i \lambda_i^2 = 1$.

Rechnung 8 (Schmidt-Zerlegung eines allgemeinen 2-Qubit-Zustands)**.** *Betrachten wir den Zustand:*

$$|\psi\rangle = \frac{1}{\sqrt{3}}|00\rangle + \frac{1}{\sqrt{6}}|01\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

Schreiben wir dies als Matrix:

$$|\psi\rangle = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{2}} \end{pmatrix}$$

Singulärwertzerlegung: $M = U\Sigma V^\dagger$
Die Singulärwerte sind die Schmidt-Koeffizienten:

$$\lambda_1 = \sqrt{\frac{2}{3}}, \quad \lambda_2 = \sqrt{\frac{1}{3}}$$

Die Schmidt-Zahl ist 2, also ist der Zustand verschränkt.

4 Erweiterte mathematische Grundlagen des Quantencomputings

Die mathematischen Grundlagen des Quantencomputings erfordern eine tiefgreifende Behandlung der linearen Algebra, Funktionalanalysis und Informationstheorie. In diesem Kapitel entwickeln wir die mathematischen Werkzeuge systematisch und demonstrieren ihre Anwendung an konkreten Quantencomputer-Beispielen.

4.1 Mathematische Formulierung von Quantenzuständen

4.1.1 Hilbert-Raum-Struktur

Ein n-Qubit-System lebt in einem 2^n-dimensionalen komplexen Hilbert-Raum $\mathcal{H} = \mathbb{C}^{2^n}$ mit dem inneren Produkt:

$$\langle\psi|\phi\rangle = \sum_{i=0}^{2^n-1} \psi_i^* \phi_i$$

Rechnung 9 (Hilbert-Raum für 3-Qubit-System)**.** *Für ein 3-Qubit-System ist $\mathcal{H} = \mathbb{C}^8$ mit Basiszuständen:*

$$\{|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle\}$$

Ein allgemeiner Zustand hat die Form:

$$|\psi\rangle = \sum_{i=0}^{7} c_i |i\rangle = c_0|000\rangle + c_1|001\rangle + c_2|010\rangle + c_3|011\rangle + c_4|100\rangle + c_5|101\rangle + c_6|110\rangle + c_7|111\rangle$$

Mit der Normierungsbedingung:

$$\sum_{i=0}^{7} |c_i|^2 = 1$$

Beispiel eines normierten Zustands:

$$|\psi\rangle = \frac{1}{2}|000\rangle + \frac{1}{2}|001\rangle + \frac{1}{\sqrt{2}}|111\rangle$$

Verifikation der Normierung:

$$\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{4} + \frac{1}{4} + \frac{1}{2} = 1 \checkmark$$

4.1.2 Tensorprodukt-Struktur

Für zusammengesetzte Systeme verwenden wir die Tensorprodukt-Struktur:

$$|\psi\rangle_{AB} = \sum_{i,j} c_{ij} |i\rangle_A \otimes |j\rangle_B$$

Rechnung 10 (Tensorprodukt-Berechnungen)**.** *Gegeben seien zwei Qubits:*

$$|\psi\rangle_A = \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle$$

$$|\phi\rangle_B = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

Das Tensorprodukt ist:

$$|\psi\rangle_A \otimes |\phi\rangle_B = \left(\frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle\right) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{41}$$

$$= \frac{1}{\sqrt{6}}|00\rangle + \frac{1}{\sqrt{6}}|01\rangle + \sqrt{\frac{2}{6}}|10\rangle + \sqrt{\frac{2}{6}}|11\rangle \tag{42}$$

$$= \frac{1}{\sqrt{6}}|00\rangle + \frac{1}{\sqrt{6}}|01\rangle + \frac{1}{\sqrt{3}}|10\rangle + \frac{1}{\sqrt{3}}|11\rangle \tag{43}$$

Matrixdarstellung:

$$|\psi\rangle_A \otimes |\phi\rangle_B = \frac{1}{\sqrt{6}}\begin{pmatrix} 1 \\ 1 \\ \sqrt{2} \\ \sqrt{2} \end{pmatrix}$$

4.2 Dichteoperator-Formalismus

4.2.1 Reine und gemischte Zustände

Definition 3 (Dichteoperator)**.** *Ein Dichteoperator ρ auf $\mathcal{H}$ erfüllt:*

1. $\rho \geq 0$ *(positiv semidefinit)*
2. $Tr(\rho) = 1$ *(Normierung)*
3. $\rho = \rho^\dagger$ *(hermitesch)*

Für reine Zustände: $\rho = |\psi\rangle\langle\psi|$ mit $\rho^2 = \rho$ Für gemischte Zustände: $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$ mit $Tr(\rho^2) < 1$

Rechnung 11 (Dichteoperator für gemischten Zustand)**.** *Betrachten wir eine Mischung aus zwei Zuständen:*

$$\rho = \frac{1}{3}|0\rangle\langle 0| + \frac{2}{3}|+\rangle\langle +|$$

wobei $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.
Explizite Berechnung:

$$|+\rangle\langle+| = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|) \tag{44}$$

$$= \frac{1}{2}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| + |1\rangle\langle 1|) \tag{45}$$

$$= \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \tag{46}$$

Also:

$$\rho = \frac{1}{3}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{2}{3}\cdot\frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \tag{47}$$

$$= \frac{1}{3}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{1}{3}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \tag{48}$$

$$= \frac{1}{3}\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \tag{49}$$

Verifikation: $Tr(\rho) = \frac{1}{3}(2+1) = 1$✓
Reinheit: $Tr(\rho^2) = Tr\left(\frac{1}{9}\begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}\right) = \frac{7}{9} < 1$ *(gemischt)*

4.2.2 Bloch-Kugel-Darstellung

Jeder Einzelqubit-Zustand lässt sich als Punkt auf der Bloch-Kugel darstellen:

$$\rho = \frac{1}{2}(I + \vec{r}\cdot\vec{\sigma})$$

wobei $\vec{r} = (r_x, r_y, r_z)$ der Bloch-Vektor und $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ die Pauli-Matrizen sind.

Rechnung 12 (Bloch-Vektor Berechnung)**.** *Für den Dichteoperator* $\rho = \frac{1}{3}\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$ *berechnen wir den Bloch-Vektor:*

$$r_x = Tr(\rho\sigma_x) = Tr\left(\frac{1}{3}\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\right) \tag{50}$$

$$= \frac{1}{3}Tr\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} = \frac{1}{3}(1+1) = \frac{2}{3} \tag{51}$$

$$r_y = Tr(\rho\sigma_y) = Tr\left(\frac{1}{3}\begin{pmatrix}2 & 1\\ 1 & 1\end{pmatrix}\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\right) \tag{52}$$

$$= \frac{1}{3}Tr\begin{pmatrix}i & -2i\\ i & -i\end{pmatrix} = \frac{1}{3}(i-i) = 0 \tag{53}$$

$$r_z = Tr(\rho\sigma_z) = Tr\left(\frac{1}{3}\begin{pmatrix}2 & 1\\ 1 & 1\end{pmatrix}\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\right) \tag{54}$$

$$= \frac{1}{3}Tr\begin{pmatrix}2 & 1\\ 1 & -1\end{pmatrix} = \frac{1}{3}(2-1) = \frac{1}{3} \tag{55}$$

Bloch-Vektor: $\vec{r} = \left(\frac{2}{3}, 0, \frac{1}{3}\right)$

Länge: $|\vec{r}| = \sqrt{\left(\frac{2}{3}\right)^2 + 0^2 + \left(\frac{1}{3}\right)^2} = \sqrt{\frac{4}{9} + \frac{1}{9}} = \sqrt{\frac{5}{9}} = \frac{\sqrt{5}}{3} < 1$ *(gemischter Zustand)*

4.3 Quantenverschränkung - Detaillierte mathematische Analyse

4.3.1 Schmidt-Zerlegung

Theorem 2 (Schmidt-Zerlegung)**.** *Jeder reine Zustand* $|\psi\rangle_{AB}$ *eines bipartiten Systems kann eindeutig geschrieben werden als:*

$$|\psi\rangle_{AB} = \sum_{i=0}^{\chi-1} \sqrt{\lambda_i}\,|u_i\rangle_A \otimes |v_i\rangle_B$$

wobei $\lambda_i > 0$ *die Schmidt-Koeffizienten,* χ *die Schmidt-Zahl und* $\{|u_i\rangle_A\}$, $\{|v_i\rangle_B\}$ *orthonormale Basen sind.*

Rechnung 13 (Schmidt-Zerlegung eines 2-Qubit-Zustands)**.** *Betrachten wir den Zustand:*

$$|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle$$

Umschreibung als Matrix:

$$|\psi\rangle \leftrightarrow M = \frac{1}{2}\begin{pmatrix}\sqrt{2} & 1\\ 1 & 0\end{pmatrix}$$

Singulärwertzerlegung $M = U\Sigma V^\dagger$*:*

Berechnung von $M^\dagger M$:

$$M^\dagger M = \frac{1}{4}\begin{pmatrix}\sqrt{2} & 1\\ 1 & 0\end{pmatrix}^\dagger \begin{pmatrix}\sqrt{2} & 1\\ 1 & 0\end{pmatrix} = \frac{1}{4}\begin{pmatrix}3 & \sqrt{2}\\ \sqrt{2} & 1\end{pmatrix}$$

Eigenwerte von $M^\dagger M$:

$$\det(M^\dagger M - \lambda I) = \frac{1}{16}(3-4\lambda)(1-4\lambda) - \frac{1}{8} = 0$$

$$16\lambda^2 - 16\lambda + 3 - 2 = 0$$

$$16\lambda^2 - 16\lambda + 1 = 0$$

Lösung: $\lambda = \frac{16\pm\sqrt{256-64}}{32} = \frac{16\pm 8\sqrt{3}}{32} = \frac{1\pm\sqrt{3}/2}{2}$
Schmidt-Koeffizienten:

$$\sqrt{\lambda_1} = \sqrt{\frac{1+\sqrt{3}/2}{2}} \approx 0.933, \quad \sqrt{\lambda_2} = \sqrt{\frac{1-\sqrt{3}/2}{2}} \approx 0.359$$

Da beide Koeffizienten ungleich null sind, ist der Zustand verschränkt mit Schmidt-Zahl $\chi = 2$.

4.3.2 VerschränkungsmaSSe

Von-Neumann-Entropie der Verschränkung Für einen reinen bipartiten Zustand ist die Verschränkungsentropie:

$$E(|\psi\rangle_{AB}) = S(\rho_A) = S(\rho_B) = -\sum_i \lambda_i \log_2(\lambda_i)$$

Rechnung 14 (Entropie für Bell-Zustände)**.** *Für den Bell-Zustand $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$:*
Schmidt-Zerlegung: $|\Phi^+\rangle = \frac{1}{\sqrt{2}}|0\rangle_A|0\rangle_B + \frac{1}{\sqrt{2}}|1\rangle_A|1\rangle_B$
Schmidt-Koeffizienten: $\lambda_1 = \lambda_2 = \frac{1}{2}$
Reduzierter Dichteoperator:

$$\rho_A = Tr_B(|\Phi^+\rangle\langle\Phi^+|) = \frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1| = \frac{1}{2}I$$

Entropie:

$$E(|\Phi^+\rangle) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1 \ ebit$$

Dies ist die maximale Verschränkung für 2 Qubits.

Concurrence Für 2-Qubit-Zustände ist die Concurrence definiert als:

$$C(\rho) = \max\{0, \lambda_1 - \lambda_2 - \lambda_3 - \lambda_4\}$$

wobei λ_i die Eigenwerte von $\sqrt{\sqrt{\rho}\tilde{\rho}\sqrt{\rho}}$ in absteigender Reihenfolge sind, und $\tilde{\rho} = (\sigma_y \otimes \sigma_y)\rho^*(\sigma_y \otimes \sigma_y)$.

Rechnung 15 (Concurrence für Werner-Zustände)**.** *Werner-Zustände haben die Form:*

$$\rho_W(p) = p\,|\Psi^-\rangle\langle\Psi^-| + \frac{1-p}{4}I$$

wobei $|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$.
Explizite Darstellung:

$$\rho_W(p) = \frac{1}{4}\begin{pmatrix} 1-p & 0 & 0 & 0 \\ 0 & 1+p & -2p & 0 \\ 0 & -2p & 1+p & 0 \\ 0 & 0 & 0 & 1-p \end{pmatrix}$$

Für Werner-Zustände kann die Concurrence analytisch berechnet werden:

$$C(\rho_W(p)) = \max\left\{0, \frac{3p-1}{2}\right\}$$

Verschränkungsschwelle: $p_c = \frac{1}{3}$
Beispiele: - $p = 0.2$: $C = \max\{0, -0.35\} = 0$ (separabel) - $p = 0.5$: $C = \max\{0, 0.25\} = 0.25$ (verschränkt) - $p = 1.0$: $C = \max\{0, 1\} = 1$ (maximal verschränkt)

4.4 Quantenalgorithmen - Mathematische Implementierung

4.4.1 Shor's Algorithmus

Das Herzstück von Shor's Algorithmus ist die Periodenfindung einer Funktion $f(x) = a^x \bmod N$.

Quanten-Fourier-Transformation Die QFT auf n Qubits ist definiert als:

$$\mathrm{QFT}_N\,|j\rangle = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^{2\pi ijk/N}\,|k\rangle$$

wobei $N = 2^n$.

Rechnung 16 (QFT für 3 Qubits)**.** *Für $n = 3$ ($N = 8$) ist:*

$$QFT_8 |j_2 j_1 j_0\rangle = \frac{1}{2\sqrt{2}} \sum_{k_2,k_1,k_0=0}^{1} e^{2\pi i j \cdot k/8} |k_2 k_1 k_0\rangle$$

wobei $j = 4j_2 + 2j_1 + j_0$ und $k = 4k_2 + 2k_1 + k_0$.
Produktform:

$$QFT_8 |j_2 j_1 j_0\rangle = \frac{1}{2\sqrt{2}} \bigotimes_{l=0}^{2} \left(|0\rangle + e^{2\pi i j \cdot 2^l/8} |1\rangle\right)$$

Beispiel für $|j\rangle = |5\rangle = |101\rangle$ (binär):

$$QFT_8 |101\rangle = \frac{1}{2\sqrt{2}}(|0\rangle + e^{2\pi i \cdot 5/8} |1\rangle) \otimes (|0\rangle + e^{2\pi i \cdot 5 \cdot 2/8} |1\rangle) \otimes (|0\rangle + e^{2\pi i \cdot 5 \cdot 4/8} |1\rangle) \tag{56}$$

$$= \frac{1}{2\sqrt{2}}(|0\rangle + e^{5\pi i/4} |1\rangle) \otimes (|0\rangle + e^{5\pi i/2} |1\rangle) \otimes (|0\rangle + e^{5\pi i} |1\rangle) \tag{57}$$

$$= \frac{1}{2\sqrt{2}}(|0\rangle + e^{5\pi i/4} |1\rangle) \otimes (|0\rangle + e^{\pi i/2} |1\rangle) \otimes (|0\rangle - |1\rangle) \tag{58}$$

Mit $e^{5\pi i/4} = -\frac{1}{\sqrt{2}} - \frac{i}{\sqrt{2}}$ und $e^{\pi i/2} = i$:

$$QFT_8 |101\rangle = \frac{1}{2\sqrt{2}} \left(|0\rangle - \frac{1+i}{\sqrt{2}} |1\rangle\right) \otimes (|0\rangle + i |1\rangle) \otimes (|0\rangle - |1\rangle)$$

Periodenfindung

Rechnung 17 (Shor's Algorithmus für $N = 15$)**.** *Faktorisierung von $N = 15$ mit $a = 7$:*

*1. **Klassische Vorbereitung:*** $\gcd(7, 15) = 1$

*2. **Periodenbestimmung:*** $f(x) = 7^x \bmod 15$ - $f(0) = 1$, $f(1) = 7$, $f(2) = 4$, $f(3) = 13$, $f(4) = 1$ - *Periode* $r = 4$

*3. **Quantenschaltung:** Verwende $n = 8$ Qubits für das erste Register ($2^8 = 256 > 15^2$)*

*4. **Superposition:*** $|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

*5. **Modulare Exponentiation:*** $|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |7^x \bmod 15\rangle$

*6. **Messung des zweiten Registers:** Ergebnis z.B. $y = 7$*

Kollaps auf: $|\psi_3\rangle = \frac{1}{\sqrt{64}} \sum_{l=0}^{63} |1 + 4l\rangle |7\rangle$

*7. **QFT und Messung:** Nach QFT des ersten Registers erhalten wir Peaks bei:*

$$k \approx \frac{l \cdot 256}{4} = 64l \quad \textit{für } l = 0, 1, 2, 3$$

*8. **Kettenbruchentwicklung:** Für gemessenes $k = 64$:*

$$\frac{k}{256} = \frac{64}{256} = \frac{1}{4} \implies r = 4$$

*9. **Faktorisierung:** Da $r = 4$ gerade:*

$$\gcd(7^{4/2} - 1, 15) = \gcd(49 - 1, 15) = \gcd(48, 15) = 3$$

$$\gcd(7^{4/2} + 1, 15) = \gcd(49 + 1, 15) = \gcd(50, 15) = 5$$

Resultat: $15 = 3 \times 5$

4.4.2 Grover's Algorithmus

Grover's Algorithmus findet ein markiertes Element in einer unsortierten Datenbank mit N Elementen in $O(\sqrt{N})$ Schritten.

Mathematische Struktur

Rechnung 18 (Grover für $N = 4$ Elemente)**.** *Suche nach Element* $|11\rangle$ *in der Datenbank* $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.

1. Anfangszustand:

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

2. Orakel-Operator: $O|x\rangle = (-1)^{f(x)}|x\rangle$ *mit* $f(|11\rangle) = 1$*, sonst* $f = 0$

$$O = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

3. Diffusionsoperator: $D = 2|s\rangle\langle s| - I$

$$D = 2 \cdot \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} - I = \begin{pmatrix} -1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & -1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 & -1/2 \end{pmatrix}$$

4. Grover-Operator: $G = -DO$
5. Erste Iteration:

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

$$DO|s\rangle = \frac{1}{2}\begin{pmatrix} -1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & -1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 & -1/2 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} -1 \\ -1 \\ -1 \\ 3 \end{pmatrix}$$

$$G|s\rangle = -DO|s\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - 3|11\rangle)$$

6. Wahrscheinlichkeiten nach einer Iteration: - $P(|00\rangle) = P(|01\rangle) = P(|10\rangle) = \left(\frac{1}{2}\right)^2 = \frac{1}{4}$ - $P(|11\rangle) = \left(\frac{3}{2}\right)^2 = \frac{9}{4}$ *(nicht normiert!)*
Normierung: $\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{9}{4} = 3$
Korrekte Wahrscheinlichkeiten: - $P(|00\rangle) = P(|01\rangle) = P(|10\rangle) = \frac{1/4}{3} = \frac{1}{12}$ - $P(|11\rangle) = \frac{9/4}{3} = \frac{3}{4}$
Optimale Iterationszahl: $k^* = \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor = \left\lfloor \frac{\pi}{4}\sqrt{4} \right\rfloor = \left\lfloor \frac{\pi}{2} \right\rfloor = 1$
Nach einer Iteration ist die Erfolgswahrscheinlichkeit bereits 75

4.5 Quantenfehlerkorrektur

4.5.1 Stabilizer-Codes

Definition 4 (Stabilizer-Gruppe)**.** *Eine Stabilizer-Gruppe S ist eine abelsche Untergruppe der n-Qubit Pauli-Gruppe $\mathcal{P}_n$, die $-I$ nicht enthält. Ein Quantenfehlerkorrekturcode wird durch seinen Stabilizer-Raum definiert:*

$$\mathcal{C} = \{|\psi\rangle : g|\psi\rangle = |\psi\rangle \text{ für alle } g \in S\}$$

Rechnung 19 (5-Qubit-Code)**.** *Der 5-Qubit-Code korrigiert beliebige Einzelqubit-Fehler und hat die Stabilizer-Generatoren:*

$$g_1 = X_1Z_2Z_3X_4 \tag{59}$$
$$g_2 = X_2Z_3Z_4X_5 \tag{60}$$
$$g_3 = X_1X_3Z_4Z_5 \tag{61}$$
$$g_4 = Z_1X_2X_4Z_5 \tag{62}$$

***Codewörter:** Die logischen Zustände $|0_L\rangle$ und $|1_L\rangle$ erfüllen $g_i|\psi_L\rangle = |\psi_L\rangle$ für alle i.*

Explizite Konstruktion von $|0_L\rangle$: 1. Starte mit $|00000\rangle$ 2. Wende Hadamard auf alle Qubits an: $|+++++\rangle$ 3. Wende CZ-Gatter entsprechend der Stabilizer-Struktur an

Resultat (vereinfacht):

$$|0_L\rangle = \frac{1}{4\sqrt{2}} \sum_{\text{gerade Parität}} |x_1 x_2 x_3 x_4 x_5\rangle$$

***Fehlerkorrektur:** Für einen Fehler E messen wir das Syndrom:*

$$s_i = \langle\psi|g_i|\psi\rangle$$

Beispiel: X-Fehler auf Qubit 2 ($E = X_2$)

$$s_1 = \langle\psi|X_1 Z_2 Z_3 X_4|\psi\rangle = \langle\psi|X_1 X_2 Z_2 Z_3 X_4|\psi\rangle = -1 \tag{63}$$
$$s_2 = \langle\psi|X_2 Z_3 Z_4 X_5|\psi\rangle = \langle\psi|X_2^2 Z_3 Z_4 X_5|\psi\rangle = +1 \tag{64}$$
$$s_3 = \langle\psi|X_1 X_3 Z_4 Z_5|\psi\rangle = +1 \tag{65}$$
$$s_4 = \langle\psi|Z_1 X_2^2 X_4 Z_5|\psi\rangle = \langle\psi|Z_1 X_4 Z_5|\psi\rangle = +1 \tag{66}$$

Syndrom: $(s_1, s_2, s_3, s_4) = (-1, +1, +1, +1)$ identifiziert eindeutig X_2-Fehler.

4.5.2 Oberflächencodes

Oberflächencodes sind topologische Codes, die auf einem 2D-Gitter definiert sind.

Rechnung 20 (Distanz-3 Oberflächencode)**.** *Für einen 3×3 Oberflächencode mit 9 Datenqubits:*

***Stabilizer-Generatoren:** - X-Stabilizer: $A_v = \prod_{e\in\partial v} X_e$ (um Vertices) - Z-Stabilizer: $B_p = \prod_{e\in\partial p} Z_e$ (um Plaquettes)*

Für ein 3×3 Gitter:

$$A_1 = X_1 X_2 X_4 X_5 \tag{67}$$
$$A_2 = X_2 X_3 X_5 X_6 \tag{68}$$
$$A_3 = X_4 X_5 X_7 X_8 \tag{69}$$
$$A_4 = X_5 X_6 X_8 X_9 \tag{70}$$

$$B_1 = Z_1 Z_2 Z_4 Z_5 \tag{71}$$
$$B_2 = Z_2 Z_3 Z_5 Z_6 \tag{72}$$
$$B_3 = Z_4 Z_5 Z_7 Z_8 \tag{73}$$
$$B_4 = Z_5 Z_6 Z_8 Z_9 \tag{74}$$

Logische Operatoren: - $\overline{X} = X_1X_2X_3$ *(horizontale Kette)* - $\overline{Z} = Z_1Z_4Z_7$ *(vertikale Kette)*

***Schwellenwert:** Für unabhängige X-, Y-, Z-Fehler mit Rate p:*

$$p_{th} \approx 10.9\%$$

***Skalierung:** Für einen $d \times d$ Code werden d^2 physikalische Qubits benötigt, um 1 logisches Qubit zu kodieren.*

4.6 Quantensimulation

4.6.1 Hamiltonian-Simulation

Rechnung 21 (Transversales Ising-Modell)**.** *Betrachte das Hamiltonian:*

$$H = -J \sum_{\langle i,j \rangle} Z_i Z_j - h \sum_i X_i$$

Für eine Kette mit 3 Spins:

$$H = -J(Z_1Z_2 + Z_2Z_3) - h(X_1 + X_2 + X_3)$$

Trotter-Zerlegung: $e^{-iHt} \approx (e^{-iH_1t/n}e^{-iH_2t/n})^n$
mit $H_1 = -J(Z_1Z_2 + Z_2Z_3)$ *und* $H_2 = -h(X_1 + X_2 + X_3)$.
Quantenschaltung für* $e^{-iH_1t/n}$*:

$$e^{-iH_1t/n} = \exp\left(iJ\frac{t}{n}(Z_1Z_2 + Z_2Z_3)\right) \tag{75}$$

$$= \exp\left(iJ\frac{t}{n}Z_1Z_2\right)\exp\left(iJ\frac{t}{n}Z_2Z_3\right) \tag{76}$$

Jeder Term wird implementiert als:

$$e^{i\theta Z_iZ_j} = CNOT_{ij} \cdot e^{i\theta Z_j} \cdot CNOT_{ij}$$

Quantenschaltung für* $e^{-iH_2t/n}$*:

$$e^{-iH_2t/n} = \prod_{i=1}^{3} e^{ih\frac{t}{n}X_i} = \prod_{i=1}^{3} R_x\left(2h\frac{t}{n}\right)$$

***Fehleranalyse:** Trotter-Fehler für n Zeitschritte:*

$$\epsilon = O\left(\frac{t^2}{n}\right)$$

Für $\epsilon = 10^{-3}$ *und* $t = 1$*:* $n \geq 10^3$ *Trotter-Schritte erforderlich.*

4.6.2 Variational Quantum Eigensolver (VQE)

Rechnung 22 (H-Molekül mit VQE)**.** *Für das H-Molekül in minimaler Basis ergibt die Jordan-Wigner-Transformation:*

$$H = g_0 I + g_1 Z_0 + g_2 Z_1 + g_3 Z_0 Z_1 + g_4 X_0 X_1 + g_5 Y_0 Y_1$$

Koeffizienten für Bindungsabstand $R = 0.735$ Å:

$$g_0 = -1.0523732 \tag{77}$$
$$g_1 = 0.39793742 \tag{78}$$
$$g_2 = -0.39793742 \tag{79}$$
$$g_3 = -0.01128010 \tag{80}$$
$$g_4 = g_5 = 0.18093119 \tag{81}$$

Variational Ansatz:

$$|\psi(\theta)\rangle = e^{i\theta(Y_0 X_1 - X_0 Y_1)} |01\rangle$$

Energieerwartungswert:

$$E(\theta) = \langle\psi(\theta)|H|\psi(\theta)\rangle \tag{82}$$
$$= g_0 + g_1\langle Z_0\rangle + g_2\langle Z_1\rangle + g_3\langle Z_0 Z_1\rangle + g_4\langle X_0 X_1\rangle + g_5\langle Y_0 Y_1\rangle \tag{83}$$

Für den Ansatz $|\psi(\theta)\rangle = \cos(\theta/2)\,|01\rangle + \sin(\theta/2)\,|10\rangle$:

$$\langle Z_0\rangle = \cos^2(\theta/2) - \sin^2(\theta/2) = \cos\theta \tag{84}$$
$$\langle Z_1\rangle = -\cos^2(\theta/2) + \sin^2(\theta/2) = -\cos\theta \tag{85}$$
$$\langle Z_0 Z_1\rangle = -\cos^2(\theta/2) - \sin^2(\theta/2) = -1 \tag{86}$$
$$\langle X_0 X_1\rangle = \langle Y_0 Y_1\rangle = \sin\theta \tag{87}$$

Einsetzen:

$$E(\theta) = g_0 + g_1\cos\theta - g_2\cos\theta - g_3 + (g_4 + g_5)\sin\theta \tag{88}$$
$$= -1.0523732 + 0.79587484\cos\theta + 0.01128010 + 0.36186238\sin\theta \tag{89}$$
$$= -1.0410931 + 0.79587484\cos\theta + 0.36186238\sin\theta \tag{90}$$

Optimierung: $\frac{dE}{d\theta} = 0$

$$-0.79587484\sin\theta + 0.36186238\cos\theta = 0$$

$$\tan\theta = \frac{0.36186238}{0.79587484} = 0.45472$$
$$\theta^* = \arctan(0.45472) = 0.42885 \text{ } rad$$

Grundzustandsenergie:

$$E_0 = E(\theta^*) = -1.8372 \text{ } Hartree$$

Exakte Lösung: $E_{exact} = -1.8370$ *Hartree Fehler:* $\Delta E = 0.0002$ *Hartree (0.01*

4.7 Quanteninformationstheorie

4.7.1 Quantenkanäle und Kapazität

Rechnung 23 (Depolarisierender Kanal)**.** *Der depolarisierende Kanal ist gegeben durch:*

$$\Phi_p(\rho) = (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z)$$

***Holevo-Information:** Für die optimale Eingabeverteilung* $\{p_i, \rho_i\}$*:*

$$\chi(\Phi_p) = H\left(\sum_i p_i \Phi_p(\rho_i)\right) - \sum_i p_i H(\Phi_p(\rho_i))$$

Für den depolarisierenden Kanal mit $p_i = 1/4$ *und* $\rho_i \in \{|0\rangle\langle 0|, |1\rangle\langle 1|, |+\rangle\langle +|, |-\rangle\langle -|\}$*: Ausgabezustände:*

$$\Phi_p(|0\rangle\langle 0|) = (1-p)|0\rangle\langle 0| + \frac{p}{3}(|1\rangle\langle 1| + |0\rangle\langle 0| + |0\rangle\langle 0|) \tag{91}$$
$$= \left(1 - \frac{2p}{3}\right)|0\rangle\langle 0| + \frac{p}{3}|1\rangle\langle 1| \tag{92}$$

Ähnlich für andere Eingaben. Der gemischte Ausgabezustand ist:

$$\bar{\rho} = \frac{1}{4}\sum_{i=1}^{4} \Phi_p(\rho_i) = \frac{I}{2}$$

Klassische Kapazität:

$$C(\Phi_p) = 1 - H\left(\frac{1+p}{2}\right)$$

wobei $H(x) = -x\log_2 x - (1-x)\log_2(1-x)$ *die binäre Entropiefunktion ist.*

Für $p = 0.1$: $C = 1 - H(0.55) = 1 - 0.993 = 0.007$ bits
***Quantenkapazität:** Für den depolarisierenden Kanal:*

$$Q(\Phi_p) = \max\{0, 1 - H(p)\}$$

Schwellenwert: $p_c = 0.5$ (Quanten-Kapazität wird null für $p > 0.5$)

4.7.2 Quantenschlüsselverteilung

Rechnung 24 (BB84-Protokoll)**.** ***Zustandsübertragung:** Alice sendet zufällig Zustände aus $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$*
***Sicherheitsanalyse:** Bei Quantenfehlerrate e:*
Schlüsselrate nach Fehlerkorrektur und Datenschutzverstärkung:

$$r = 1 - h(e) - f(e) \cdot H(E|X)$$

wobei: - $h(e) = -e \log_2 e - (1 - e) \log_2(1 - e)$ (binäre Entropiefunktion) - $f(e) \geq 1$ (Effizienz der Fehlerkorrektur) - $H(E|X)$ (bedingte Entropie der Fehler)
Für optimale Parameter und $f(e) = 1.16$:

$$r = 1 - h(e) - 1.16 \cdot h(e) = 1 - 2.16 \cdot h(e)$$

***Sicherheitsschwelle:** $r = 0$ für:*

$$h(e) = \frac{1}{2.16} \approx 0.463$$

Dies entspricht $e \approx 11\%$.
Beispielrechnung für $e = 5\%$:

$$h(0.05) = -0.05 \log_2(0.05) - 0.95 \log_2(0.95) = 0.286$$

$$r = 1 - 2.16 \times 0.286 = 0.382 \text{ bits pro gesendetes Qubit}$$

5 Fazit

Diese erweiterte mathematische Behandlung des Quantencomputings zeigt die Tiefe und Eleganz der zugrundeliegenden Theorie. Durch die detaillierten Rechnungen und praktischen Beispiele wird deutlich, wie abstrakte quantenmechanische Konzepte in konkrete Algorithmen und Protokolle umgesetzt werden können. Die mathematischen Werkzeuge - von der Hilbert-Raum-Theorie über VerschränkungsmaSSe bis hin zu Quantenfehlerkorrektur - bilden das solide Fundament für die praktische Realisierung von Quantencomputern und deren Anwendungen in Kryptographie, Optimierung und Quantensimulation.

5.0.1 Von-Neumann-Entropie der Verschränkung

Für reine Zustände ist die Verschränkungsentropie:

$$S(\rho_A) = -\mathrm{Tr}(\rho_A \log_2 \rho_A) = -\sum_i \lambda_i^2 \log_2(\lambda_i^2)$$

wobei λ_i die Schmidt-Koeffizienten sind.

Rechnung 25 (Entropie für Bell-Zustände)**.** *Für* $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$*:*
Der reduzierte Dichteoperator des ersten Qubits ist:

$$\begin{aligned}
\rho_A &= Tr_B(|\Phi^+\rangle \langle\Phi^+|) && (93)\\
&= Tr_B\left(\frac{1}{2}(|00\rangle + |11\rangle)(\langle 00| + \langle 11|)\right) && (94)\\
&= \frac{1}{2}(|0\rangle \langle 0| + |1\rangle \langle 1|) && (95)\\
&= \frac{1}{2} I && (96)
\end{aligned}$$

Die Eigenwerte sind $\frac{1}{2}, \frac{1}{2}$*, also:*

$$S(\rho_A) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1 \text{ bit}$$

Dies ist die maximale Verschränkung für 2 Qubits.

5.0.2 Concurrence

Für 2-Qubit-Zustände ist die Concurrence:

$$C(\rho) = \max(0, \lambda_1 - \lambda_2 - \lambda_3 - \lambda_4)$$

wobei λ_i die Eigenwerte von $\rho\tilde{\rho}$ in absteigender Reihenfolge sind, und $\tilde{\rho} = (Y \otimes Y)\rho^*(Y \otimes Y)$.

Rechnung 26 (Concurrence für Werner-Zustände)**.** *Werner-Zustände haben die Form:*

$$\rho_W(p) = p\,|\Psi^-\rangle \langle\Psi^-| + \frac{1-p}{4} I$$

Für $p > \frac{1}{3}$ *ist die Concurrence:*

$$C(\rho_W) = \max\left(0, \frac{3p-1}{2}\right)$$

Bei $p = \frac{1}{3}$ *ist der Zustand separabel* ($C = 0$)*. Bei* $p = 1$ *ist der Zustand maximal verschränkt* ($C = 1$)*.*

5.1 Bell-Ungleichungen und Nicht-Lokalität

Theorem 3 (CHSH-Ungleichung)**.** *Für lokale realistische Theorien gilt:*

$$|E(a,b) - E(a,b') + E(a',b) + E(a',b')| \leq 2$$

wobei $E(a,b)$ die Korrelation zwischen Messungen in Richtungen a und b ist.

Rechnung 27 (Bell-Test mit $|\Phi^+\rangle$)**.** *Für den Bell-Zustand $|\Phi^+\rangle$ und Messrichtungen:*

$$a = (1,0,0), \quad a' = \frac{1}{\sqrt{2}}(1,1,0) \tag{97}$$

$$b = \frac{1}{\sqrt{2}}(1,-1,0), \quad b' = (0,1,0) \tag{98}$$

Die Korrelationen sind:

$$E(a,b) = \cos\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} \tag{99}$$

$$E(a,b') = \cos\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} \tag{100}$$

$$E(a',b) = \cos\left(\frac{\pi}{2}\right) = 0 \tag{101}$$

$$E(a',b') = \cos\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} \tag{102}$$

CHSH-Wert:

$$S = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} + 0 + \frac{1}{\sqrt{2}} = \frac{3}{\sqrt{2}} = \frac{3\sqrt{2}}{2} \approx 2.12$$

Dies verletzt die klassische Grenze von 2!

6 Dichteoperatoren und gemischte Zustände

Für die vollständige Beschreibung von Quantensystemen, besonders in Anwesenheit von Rauschen und Dekohärenz, benötigen wir die Dichtematrix-Formulierung.

6.1 Mathematische Definition

Definition 5 (Dichteoperator)**.** *Ein Dichteoperator ρ ist ein positiv semidefiniter, hermitescher Operator mit Spur 1:*

1. $\rho = \rho^\dagger$ *(hermitesch)*

2. $\rho \geq 0$ *(positiv semidefinit)*

3. $Tr(\rho) = 1$ *(Normierung)*

Reine Zustände: $\rho = |\psi\rangle\langle\psi|$ mit $\rho^2 = \rho$
Gemischte Zustände: $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$ mit $p_i \geq 0$, $\sum_i p_i = 1$

6.2 Bloch-Darstellung für Einzelqubits

Jeder Einzelqubit-Zustand kann geschrieben werden als:

$$\rho = \frac{1}{2}(I + \vec{r} \cdot \vec{\sigma})$$

wobei $\vec{r} = (r_x, r_y, r_z)$ der Bloch-Vektor und $\vec{\sigma} = (X, Y, Z)$ der Pauli-Vektor ist.

Rechnung 28 (Bloch-Vektor Berechnung)**.** *Für einen allgemeinen Zustand:*

$$\rho = \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix}$$

Die Bloch-Vektor-Komponenten sind:

$$r_x = Tr(\rho X) = \rho_{01} + \rho_{10} = 2Re(\rho_{01}) \tag{103}$$
$$r_y = Tr(\rho Y) = i(\rho_{10} - \rho_{01}) = 2Im(\rho_{01}) \tag{104}$$
$$r_z = Tr(\rho Z) = \rho_{00} - \rho_{11} \tag{105}$$

Für reine Zustände ist $|\vec{r}| = 1$, für gemischte Zustände $|\vec{r}| < 1$.

6.3 Partial Trace und reduzierte Zustände

Für zusammengesetzte Systeme ist die partielle Spur:

$$\rho_A = \mathrm{Tr}_B(\rho_{AB}) = \sum_j (\mathbb{1}_A \otimes \langle j|_B) \rho_{AB} (\mathbb{1}_A \otimes |j\rangle_B)$$

Rechnung 29 (Partielle Spur für Bell-Zustand)**.** *Für $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$:*

$$\rho_{AB} = |\Phi^+\rangle\langle\Phi^+| = \frac{1}{2}(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)$$

$$\rho_{AB} = \frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Die partielle Spur über System B:

$$\rho_A = \langle 0|_B \rho_{AB} |0\rangle_B + \langle 1|_B \rho_{AB} |1\rangle_B \tag{106}$$
$$= \frac{1}{2}(|0\rangle \langle 0| + |1\rangle \langle 1|) \tag{107}$$
$$= \frac{1}{2} I \tag{108}$$

Dies zeigt maximale Unkenntnis über System A.

7 Quantenteleportation - Vollständige mathematische Ableitung

Die Quantenteleportation ermöglicht die Übertragung eines unbekannten Quantenzustands über klassische Kommunikation und verschränkte Qubits.

7.1 Das Teleportationsprotokoll

Ausgangssituation: - Alice besitzt ein Qubit im unbekannten Zustand $|\psi\rangle_1 = \alpha|0\rangle + \beta|1\rangle$ - Alice und Bob teilen den Bell-Zustand $|\Phi^+\rangle_{23} = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{23}$

Anfangszustand des 3-Qubit-Systems:

$$|\Psi\rangle_{\text{initial}} = |\psi\rangle_1 \otimes |\Phi^+\rangle_{23} = (\alpha|0\rangle + \beta|1\rangle)_1 \otimes \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{23}$$

Rechnung 30 (Schritt-für-Schritt Teleportation)**.** ***Schritt 1: Expansion des Anfangszustands***

$$|\Psi\rangle_{initial} = \frac{1}{\sqrt{2}}[\alpha|0\rangle_1 (|00\rangle_{23} + |11\rangle_{23}) + \beta|1\rangle_1 (|00\rangle_{23} + |11\rangle_{23})] \tag{109}$$
$$= \frac{1}{\sqrt{2}}[\alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle] \tag{110}$$

Schritt 2: CNOT-Gatter zwischen Qubit 1 und 2

$$CNOT_{12}|\Psi\rangle_{initial} = \frac{1}{\sqrt{2}}[\alpha|000\rangle + \alpha|011\rangle + \beta|110\rangle + \beta|101\rangle] \tag{111}$$

Schritt 3: Hadamard-Gatter auf Qubit 1 *Das Hadamard-Gatter wirkt als:* $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

$$H_1 CNOT_{12} |\Psi\rangle_{initial} \tag{112}$$

$$= \frac{1}{2\sqrt{2}}[\alpha(|0\rangle + |1\rangle)|00\rangle + \alpha(|0\rangle + |1\rangle)|11\rangle + \beta(|0\rangle - |1\rangle)|10\rangle + \beta(|0\rangle - |1\rangle)|01\rangle] \tag{113}$$

$$= \frac{1}{2}[\alpha|000\rangle + \alpha|100\rangle + \alpha|011\rangle + \alpha|111\rangle + \beta|010\rangle - \beta|110\rangle + \beta|001\rangle - \beta|101\rangle] \tag{114}$$

Schritt 4: Umgruppierung nach Bell-Basis *Gruppieren nach den ersten beiden Qubits:*

$$|\Psi\rangle_{final} = \frac{1}{2}[|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\beta|0\rangle + \alpha|1\rangle) \tag{115}$$

$$+ |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\beta|0\rangle - \alpha|1\rangle)] \tag{116}$$

Schritt 5: Interpretation *Nach der Messung der ersten beiden Qubits ist der Zustand des dritten Qubits:*

- *Messung* $|00\rangle$*:* $\alpha|0\rangle + \beta|1\rangle$ *(ursprünglicher Zustand!)*
- *Messung* $|01\rangle$*:* $\beta|0\rangle + \alpha|1\rangle = X(\alpha|0\rangle + \beta|1\rangle)$
- *Messung* $|10\rangle$*:* $\alpha|0\rangle - \beta|1\rangle = Z(\alpha|0\rangle + \beta|1\rangle)$
- *Messung* $|11\rangle$*:* $\beta|0\rangle - \alpha|1\rangle = XZ(\alpha|0\rangle + \beta|1\rangle)$

Klassische Kommunikation und Korrektur: Bob erhält die Messergebnisse und wendet die entsprechenden Korrekturgatter an:

- $00 \to I$ (keine Korrektur)
- $01 \to X$
- $10 \to Z$
- $11 \to XZ$

7.2 Treue der Teleportation

Die Treue (Fidelity) der Teleportation ist:

$$F = |\langle\psi|_{\text{original}} |\psi\rangle_{\text{teleportiert}}|^2 = 1$$

Dies zeigt, dass die Teleportation perfekt ist (im Idealfall ohne Rauschen).

8 Quantenalgorithmen - Detaillierte mathematische Analyse

8.1 Shor's Algorithmus - Vollständige mathematische Behandlung

Shor's Algorithmus löst das Faktorisierungsproblem durch Periodenfindung. Die mathematische Struktur basiert auf der Quanten-Fourier-Transformation.

8.1.1 Mathematisches Problem

Gegeben: Eine zusammengesetzte Zahl $N = p \cdot q$ (Produkt zweier Primzahlen)
Gesucht: Die Faktoren p und q

Reduktion auf Periodenfindung: 1. Wähle zufällig $a < N$ mit $\gcd(a, N) = 1$ 2. Finde die Periode r von $f(x) = a^x \bmod N$ 3. Falls r gerade und $a^{r/2} \not\equiv -1 \pmod{N}$:

$$\gcd(a^{r/2} \pm 1, N) \text{ sind nicht-triviale Faktoren}$$

Rechnung 31 (Klassisches Beispiel: $N = 15$, $a = 7$)**.** *Berechne die Potenzen von 7 modulo 15:*

$$7^0 \equiv 1 \pmod{15} \tag{117}$$
$$7^1 \equiv 7 \pmod{15} \tag{118}$$
$$7^2 \equiv 49 \equiv 4 \pmod{15} \tag{119}$$
$$7^3 \equiv 7 \cdot 4 = 28 \equiv 13 \pmod{15} \tag{120}$$
$$7^4 \equiv 7 \cdot 13 = 91 \equiv 1 \pmod{15} \tag{121}$$

Die Periode ist $r = 4$. Da r gerade:

$$a^{r/2} - 1 = 7^2 - 1 = 49 - 1 = 48 \tag{122}$$
$$a^{r/2} + 1 = 7^2 + 1 = 49 + 1 = 50 \tag{123}$$

Faktoren:

$$\gcd(48, 15) = \gcd(48, 15) = 3 \tag{124}$$
$$\gcd(50, 15) = \gcd(50, 15) = 5 \tag{125}$$

Also $15 = 3 \times 5$.

8.1.2 Quanten-Fourier-Transformation

Die diskrete Fourier-Transformation auf N Punkten ist:

$$\text{DFT} : x_j \mapsto y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i j k/N}$$

Die Quanten-Fourier-Transformation wirkt auf Basiszustände:

$$\text{QFT}_N |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} |k\rangle$$

Rechnung 32 (QFT für 3 Qubits ($N = 8$))**.** *Für $n = 3$ Qubits ($N = 2^3 = 8$):*

$$QFT_8 |j_2 j_1 j_0\rangle = \frac{1}{2\sqrt{2}} \sum_{k_2,k_1,k_0} e^{2\pi i j \cdot k/8} |k_2 k_1 k_0\rangle$$

wobei $j = 4j_2 + 2j_1 + j_0$ und $k = 4k_2 + 2k_1 + k_0$.
Die Produktform:

$$QFT_8 |j_2 j_1 j_0\rangle = \frac{1}{2\sqrt{2}} \bigotimes_{l=0}^{2} \left(|0\rangle + e^{2\pi i j \cdot 2^l/8} |1\rangle \right)$$

Explizit:

$$= \frac{1}{2\sqrt{2}} (|0\rangle + e^{2\pi i j/8} |1\rangle) \otimes (|0\rangle + e^{2\pi i j/4} |1\rangle) \otimes (|0\rangle + e^{2\pi i j/2} |1\rangle) \qquad (126)$$

8.1.3 Periodenfindungsalgorithmus

Schritt 1: Superposition erzeugen

$$|\psi_1\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle |0\rangle$$

Schritt 2: Modulare Exponentiation

$$|\psi_2\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle |a^x \bmod N\rangle$$

Schritt 3: Messung des zweiten Registers Nach der Messung eines Werts $y = a^s \bmod N$ kollabiert der Zustand zu:

$$|\psi_3\rangle = \frac{1}{\sqrt{r}} \sum_{l=0}^{r-1} |s + lr\rangle |y\rangle$$

Schritt 4: QFT und Messung Die QFT des ersten Registers ergibt Peaks bei $k \approx \frac{lN}{r}$ für $l = 0, 1, ..., r - 1$.

8.2 Grover's Algorithmus - Geometrische und algebraische Analyse

Grover's Algorithmus findet ein markiertes Element in einer unsortierten Datenbank von N Elementen in $O(\sqrt{N})$ Schritten.

8.2.1 Mathematische Formulierung

Orakel-Funktion:

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

wobei $f(x) = 1$ für markierte Elemente und $f(x) = 0$ sonst.

Diffusionsoperator:

$$D = 2 |s\rangle \langle s| - I$$

wobei $|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$ der gleichmäSSige Superpositionszustand ist.

Rechnung 33 (Grover-Operator für $N = 4$, ein markiertes Element)**.** *Angenommen, $|11\rangle$ ist das markierte Element ($M = 1$ von $N = 4$ Elementen).*

Anfangszustand:

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

Nach Orakel-Anwendung:

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

Diffusionsoperator explizit:

$$D = 2 |s\rangle \langle s| - I = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

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

Nach Diffusion:

$$D \cdot O_f |s\rangle = D \cdot \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} -1 \\ -1 \\ -1 \\ 3 \end{pmatrix} = \frac{1}{2}(- |00\rangle - |01\rangle - |10\rangle + 3 |11\rangle)$$

Die Wahrscheinlichkeit, $|11\rangle$ zu messen, ist:

$$P(11) = \left|\frac{3}{2}\right|^2 = \frac{9}{4} \cdot \frac{1}{4} = \frac{9}{16} \approx 0.56$$

Nach einer weiteren Iteration nähert sich die Wahrscheinlichkeit 1.

8.2.2 Geometrische Interpretation

Grover's Algorithmus kann als Rotation in einem 2D-Unterraum verstanden werden.

Orthogonale Basis: - $|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{x:f(x)=0} |x\rangle$ (nicht-markierte Zustände) - $|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x:f(x)=1} |x\rangle$ (markierte Zustände)

Anfangszustand:

$$|s\rangle = \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle$$

Grover-Operator als Rotation:

$$G = -D \cdot O_f$$

wirkt als Rotation um den Winkel θ mit $\sin(\theta/2) = \sqrt{M/N}$.

Optimale Anzahl von Iterationen:

$$k^* = \left\lfloor \frac{\pi}{4} \sqrt{\frac{N}{M}} \right\rfloor$$

Rechnung 34 (Erfolgswahrscheinlichkeit nach k Iterationen)**.** *Nach k Grover-Iterationen ist der Zustand:*

$$G^k |s\rangle = \cos\left((2k+1)\frac{\theta}{2}\right)|\alpha\rangle + \sin\left((2k+1)\frac{\theta}{2}\right)|\beta\rangle$$

Die Erfolgswahrscheinlichkeit ist:

$$P_{success}(k) = \sin^2\left((2k+1)\frac{\theta}{2}\right)$$

Für $M = 1$ und $N = 2^n$ ist $\theta \approx \frac{2}{\sqrt{N}}$, und die optimale Anzahl von Iterationen ist:

$$k^* \approx \frac{\pi}{4}\sqrt{N}$$

9 Quantenfehlerkorrektur - Mathematische Grundlagen

Quantenfehlerkorrektur ist essentiell für die praktische Realisierung von Quantencomputern, da Qubits sehr anfällig für Rauschen sind.

9.1 Grundprinzipien der Quantenfehlerkorrektur

Im Gegensatz zur klassischen Fehlerkorrektur können Quantenzustände nicht kopiert werden (No-Cloning-Theorem). Stattdessen nutzt die Quantenfehlerkorrektur redundante Kodierung in einem gröSSeren Hilbert-Raum.

Theorem 4 (Quantenfehlerkorrektur-Bedingungen)**.** *Ein Code kann Fehler aus der Menge $\{E_i\}$ korrigieren, wenn:*

$$\langle\psi_j| E_i^\dagger E_k |\psi_l\rangle = C_{ik}\delta_{jl}$$

für alle Codewörter $|\psi_j\rangle, |\psi_l\rangle$ und Konstanten C_{ik}.

9.2 Der 3-Qubit Bit-Flip Code

Der einfachste Quantenfehlerkorrekturcode korrigiert einzelne Bit-Flip-Fehler.

Kodierung:

$$|0\rangle_L = |000\rangle \tag{127}$$
$$|1\rangle_L = |111\rangle \tag{128}$$

Ein allgemeiner logischer Zustand:

$$|\psi\rangle_L = \alpha|0\rangle_L + \beta|1\rangle_L = \alpha|000\rangle + \beta|111\rangle$$

Rechnung 35 (Fehlerkorrektur für X-Fehler auf Qubit 1)**.** ***Nach X-Fehler auf Qubit 1:***

$$X_1|\psi\rangle_L = \alpha X_1|000\rangle + \beta X_1|111\rangle = \alpha|100\rangle + \beta|011\rangle$$

Syndrom-Messung: *Die Stabilizer-Generatoren sind:*

$$g_1 = Z_1Z_2 \tag{129}$$
$$g_2 = Z_2Z_3 \tag{130}$$

Syndrom-Messungen:

$$s_1 = \langle g_1\rangle = \langle Z_1Z_2\rangle \tag{131}$$
$$s_2 = \langle g_2\rangle = \langle Z_2Z_3\rangle \tag{132}$$

Für den fehlerhaften Zustand $\alpha|100\rangle + \beta|011\rangle$:

$$s_1 = \alpha\langle 100|Z_1Z_2|100\rangle + \beta\langle 011|Z_1Z_2|011\rangle = \alpha(-1)(1) + \beta(1)(-1) = -\alpha - \beta \tag{133}$$

$$s_2 = \alpha\langle 100|Z_2Z_3|100\rangle + \beta\langle 011|Z_2Z_3|011\rangle = \alpha(1)(1) + \beta(-1)(-1) = \alpha + \beta \tag{134}$$

Für $\alpha, \beta \neq 0$: $s_1 = -1, s_2 = +1$ identifiziert den Fehler auf Qubit 1.
Korrektur: *Anwendung von X_1 korrigiert den Fehler:*

$$X_1(\alpha|100\rangle + \beta|011\rangle) = \alpha|000\rangle + \beta|111\rangle = |\psi\rangle_L$$

9.3 Der 9-Qubit Shor Code

Der Shor-Code korrigiert sowohl Bit-Flip- als auch Phase-Flip-Fehler durch eine Verschachtelung von Codes.

Logische Zustände:

$$|0\rangle_L = \frac{1}{2\sqrt{2}}[(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)] \tag{135}$$

$$|1\rangle_L = \frac{1}{2\sqrt{2}}[(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle)] \tag{136}$$

Stabilizer-Generatoren: Der Shor-Code hat 8 Stabilizer-Generatoren:

$$X_1X_2, X_2X_3, X_4X_5, X_5X_6, X_7X_8, X_8X_9 \tag{137}$$

$$Z_1Z_2Z_3Z_4Z_5Z_6, Z_4Z_5Z_6Z_7Z_8Z_9 \tag{138}$$

10 Quantenmaschinenlernen - Mathematische Grundlagen und Algorithmen

Das Quantenmaschinenlernen (Quantum Machine Learning, QML) vereint die Prinzipien des Quantencomputings mit Techniken des maschinellen Lernens. Dieses interdisziplinäre Feld verspricht exponentiell verbesserte Lernalgorithmen für bestimmte Problemklassen und eröffnet neue Möglichkeiten in der Datenanalyse und Mustererkennung.

10.1 Grundlagen des Quantenmaschinenlernens

10.1.1 Quantendaten und Feature Maps

Klassische Daten müssen in Quantenzustände kodiert werden. Eine Feature Map $\Phi : \mathcal{X} \to \mathcal{H}$ bildet klassische Datenpunkte $x \in \mathcal{X}$ auf Quantenzustände $|\Phi(x)\rangle \in \mathcal{H}$ ab.

Amplitude Encoding: Für einen Datenvektor $\vec{x} = (x_1, x_2, ..., x_N)$ mit $\|\vec{x}\| = 1$:

$$|\Phi(\vec{x})\rangle = \sum_{i=1}^{N} x_i |i\rangle$$

Angle Encoding:

$$|\Phi(\vec{x})\rangle = \bigotimes_{i=1}^{n} (\cos(x_i)|0\rangle + \sin(x_i)|1\rangle)$$

Rechnung 36 (Feature Map für 2D-Daten)**.** *Für einen 2D-Datenpunkt $\vec{x} = (x_1, x_2)$ mit Angle Encoding:*

$$|\Phi(\vec{x})\rangle = (\cos(x_1)|0\rangle + \sin(x_1)|1\rangle) \otimes (\cos(x_2)|0\rangle + \sin(x_2)|1\rangle)$$

Ausmultipliziert:

$$|\Phi(\vec{x})\rangle = \cos(x_1)\cos(x_2)|00\rangle + \cos(x_1)\sin(x_2)|01\rangle \tag{139}$$
$$+ \sin(x_1)\cos(x_2)|10\rangle + \sin(x_1)\sin(x_2)|11\rangle \tag{140}$$

Für $\vec{x} = (\pi/4, \pi/3)$:

$$|\Phi(\vec{x})\rangle = \frac{1}{\sqrt{2}} \cdot \frac{1}{2}|00\rangle + \frac{1}{\sqrt{2}} \cdot \frac{\sqrt{3}}{2}|01\rangle \tag{141}$$
$$+ \frac{1}{\sqrt{2}} \cdot \frac{1}{2}|10\rangle + \frac{1}{\sqrt{2}} \cdot \frac{\sqrt{3}}{2}|11\rangle \tag{142}$$
$$= \frac{1}{2\sqrt{2}}(|00\rangle + \sqrt{3}|01\rangle + |10\rangle + \sqrt{3}|11\rangle) \tag{143}$$

10.2 Variational Quantum Circuits

Variational Quantum Circuits (VQCs) sind parametrisierte Quantenschaltungen, die als Quantenanaloga neuronaler Netze fungieren.

10.2.1 Mathematische Struktur

Ein VQC wird durch einen parametrisierten unitären Operator $U(\vec{\theta})$ beschrieben:

$$|\psi(\vec{\theta})\rangle = U(\vec{\theta})|0\rangle^{\otimes n}$$

Layered Structure:

$$U(\vec{\theta}) = \prod_{l=1}^{L} U_l(\vec{\theta}^{(l)})$$

wobei jede Schicht U_l aus Rotations- und Verschränkungsgattern besteht.

Rechnung 37 (2-Qubit VQC mit einer Schicht)**.** *Betrachten wir einen 2-Qubit VQC:*

$$U(\vec{\theta}) = CNOT_{12} \cdot R_y(\theta_2) \otimes R_y(\theta_1)$$

wobei $\vec{\theta} = (\theta_1, \theta_2)$.

Schritt 1: Rotation

$$R_y(\theta_1) \otimes R_y(\theta_2) \left|00\right\rangle = (\cos(\theta_1/2) \left|0\right\rangle + \sin(\theta_1/2) \left|1\right\rangle) \otimes (\cos(\theta_2/2) \left|0\right\rangle + \sin(\theta_2/2) \left|1\right\rangle)$$

$$= \cos(\theta_1/2)\cos(\theta_2/2) \left|00\right\rangle + \cos(\theta_1/2)\sin(\theta_2/2) \left|01\right\rangle + \sin(\theta_1/2)\cos(\theta_2/2) \left|10\right\rangle + \sin(\theta_1/2)\sin(\theta_2/2) \left|1\right.$$

Schritt 2: CNOT

$$\left|\psi(\vec{\theta})\right\rangle = \cos(\theta_1/2)\cos(\theta_2/2) \left|00\right\rangle + \cos(\theta_1/2)\sin(\theta_2/2) \left|01\right\rangle + \sin(\theta_1/2)\cos(\theta_2/2) \left|11\right\rangle + \sin(\theta_1/2)\sin(\theta$$

Für $\vec{\theta} = (\pi/2, \pi/4)$*:*

$$\left|\psi\right\rangle = \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} \left|00\right\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} \left|01\right\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} \left|11\right\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} \left|10\right\rangle$$

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

10.3 Quantum Neural Networks

10.3.1 Quantum Perceptron

Ein Quantum Perceptron verarbeitet Eingabedaten durch eine Sequenz von Quantengattern und erzeugt eine Ausgabe durch Messung.

Mathematische Beschreibung:

$$f(\vec{x}, \vec{\theta}) = \langle 0^{\otimes n} | U^\dagger(\vec{\theta}) M U(\vec{\theta}) | \Phi(\vec{x}) \rangle$$

wobei M ein Messoperator ist (z.B. $M = Z \otimes I^{\otimes(n-1)}$).

Rechnung 38 (Quantum Perceptron für Klassifikation)**.** *Für ein 2-Klassen-Problem mit* $y \in \{-1, +1\}$*:*

Kostenfunktion:

$$\mathcal{L}(\vec{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \left(y^{(i)} - f(\vec{x}^{(i)}, \vec{\theta}) \right)^2$$

Gradientenberechnung (Parameter-Shift Rule):

$$\frac{\partial f}{\partial \theta_j} = \frac{1}{2}\left[f(\vec{\theta}^{(j+)}) - f(\vec{\theta}^{(j-)})\right]$$

wobei $\vec{\theta}^{(j\pm)} = \vec{\theta} \pm \frac{\pi}{2}\vec{e}_j$.
Beispiel für θ_1*:*

$$\frac{\partial f}{\partial \theta_1} = \frac{1}{2}\left[f(\theta_1 + \pi/2, \theta_2, ...) - f(\theta_1 - \pi/2, \theta_2, ...)\right] \tag{144}$$

10.4 Quantum Support Vector Machines

10.4.1 Kernel-basierte Methoden

Quantum SVMs nutzen quanteninduzierte Feature-Spaces, um nichtlinear separierbare Daten zu klassifizieren.

Quantum Kernel:

$$K(\vec{x}_i, \vec{x}_j) = |\langle \Phi(\vec{x}_i) | \Phi(\vec{x}_j) \rangle|^2$$

Rechnung 39 (Berechnung des Quantum Kernels)**.** *Für zwei Datenpunkte* $\vec{x}_1 = (x_{11}, x_{12})$ *und* $\vec{x}_2 = (x_{21}, x_{22})$ *mit Angle Encoding:*

$$\langle \Phi(\vec{x}_1) | \Phi(\vec{x}_2) \rangle = \cos(x_{11})\cos(x_{21}) + \cos(x_{11})\sin(x_{21})e^{i(x_{22}-x_{12})} + \sin(x_{11})\cos(x_{21})e^{i(x_{12}-x_{22})} + \sin(x_{11})$$

Vereinfachung für reelle Winkel:

$$= \cos(x_{11})\cos(x_{21}) + \sin(x_{11})\sin(x_{21}) + \cos(x_{11})\sin(x_{21})\cos(x_{22}-x_{12}) + \sin(x_{11})\cos(x_{21})\cos(x_{12}-x_{2}$$

$$= \cos(x_{11}-x_{21}) + \cos(x_{11})\sin(x_{21})\cos(x_{22}-x_{12}) + \sin(x_{11})\cos(x_{21})\cos(x_{12}-x_{22})$$

Für $\vec{x}_1 = (\pi/4, \pi/6)$ *und* $\vec{x}_2 = (\pi/3, \pi/4)$*:*

$$|\langle \Phi(\vec{x}_1) | \Phi(\vec{x}_2) \rangle|^2 \approx 0.854$$

10.5 Quantum Principal Component Analysis

10.5.1 Eigenvalue Estimation

Quantum PCA nutzt die Quantenphasenschätzung zur Bestimmung der Eigenwerte einer Kovarianzmatrix.

Algorithmus: 1. Bereite die Datenmatrix $\rho = \frac{1}{N}\sum_{i=1}^{N} |x_i\rangle \langle x_i|$ vor 2. Wende Quantenphasenschätzung auf ρ an 3. Extrahiere die Hauptkomponenten

Rechnung 40 (Quantum PCA für 2Œ2 Matrix)**.** *Betrachte die Kovarianzmatrix:*

$$C = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Eigenwerte und Eigenvektoren: *Charakteristisches Polynom:* $\det(C - \lambda I) = (2-\lambda)^2 - 1 = \lambda^2 - 4\lambda + 3 = 0$

Eigenwerte: $\lambda_1 = 3$, $\lambda_2 = 1$

Eigenvektoren:

$$|v_1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad |v_2\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Quantenphasenschätzung: *Für den Zustand* $|v_1\rangle$ *liefert die Phasenschätzung:*

$$e^{i\phi_1} = e^{i \cdot 2\pi \cdot 3/4} = e^{i3\pi/2} = -i$$

Die Phase $\phi_1 = 3/4$ *entspricht dem Eigenwert* $\lambda_1 = 3$*.*

10.6 Quantum Generative Models

10.6.1 Quantum Generative Adversarial Networks

Quantum GANs verwenden zwei Quantenschaltungen: einen Generator G und einen Diskriminator D.

Generator:

$$G : \mathcal{Z} \to \mathcal{X} \quad \text{mit} \quad |\psi_G(z, \vec{\theta}_G)\rangle = U_G(\vec{\theta}_G)\,|z\rangle$$

Diskriminator:

$$D : \mathcal{X} \to [0,1] \quad \text{mit} \quad D(x, \vec{\theta}_D) = \langle M \rangle_{U_D(\vec{\theta}_D)|x\rangle}$$

Rechnung 41 (Minimax-Optimierung für QGAN)**.** ***Kostenfunktion:***

$$\min_{\vec{\theta}_G} \max_{\vec{\theta}_D} V(\vec{\theta}_D, \vec{\theta}_G) = \mathbb{E}_{x \sim p_{data}}[\log D(x, \vec{\theta}_D)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z, \vec{\theta}_G), \vec{\theta}_D))]$$

Diskriminator-Update:

$$\vec{\theta}_D^{(t+1)} = \vec{\theta}_D^{(t)} + \alpha_D \nabla_{\vec{\theta}_D} V(\vec{\theta}_D^{(t)}, \vec{\theta}_G^{(t)})$$

Generator-Update:

$$\vec{\theta}_G^{(t+1)} = \vec{\theta}_G^{(t)} - \alpha_G \nabla_{\vec{\theta}_G} V(\vec{\theta}_D^{(t+1)}, \vec{\theta}_G^{(t)})$$

Für einen einfachen Fall mit einem Parameter θ_G*:*

$$\frac{\partial V}{\partial \theta_G} = -\mathbb{E}_{z \sim p_z}\left[\frac{1}{1 - D(G(z, \theta_G))} \frac{\partial D(G(z, \theta_G))}{\partial \theta_G}\right]$$

10.7 Quantum Reinforcement Learning

10.7.1 Variational Quantum Policy Gradient

Quantum RL nutzt Quantenschaltungen zur Darstellung von Policies in Verstärkungslernen.

Quantenpolicy:

$$\pi_{\vec{\theta}}(a|s) = |\langle a|U(\vec{\theta})|s\rangle|^2$$

Policy Gradient:

$$\nabla_{\vec{\theta}} J(\vec{\theta}) = \mathbb{E}_{\tau\sim\pi_{\vec{\theta}}}\left[\sum_{t=0}^{T} \nabla_{\vec{\theta}} \log \pi_{\vec{\theta}}(a_t|s_t) R(\tau)\right]$$

Rechnung 42 (Quantum Policy für 2-State MDP)**.** *Betrachte ein 2-State MDP mit Zuständen* $\{s_0, s_1\}$ *und Aktionen* $\{a_0, a_1\}$*.*

Quantenpolicy:

$$\pi_\theta(a_0|s_0) = |\cos(\theta/2)|^2, \quad \pi_\theta(a_1|s_0) = |\sin(\theta/2)|^2$$

Log-Policy Gradient:

$$\frac{\partial}{\partial\theta}\log \pi_\theta(a_0|s_0) = \frac{\partial}{\partial\theta}\log(\cos^2(\theta/2)) = -\tan(\theta/2)$$

$$\frac{\partial}{\partial\theta}\log \pi_\theta(a_1|s_0) = \frac{\partial}{\partial\theta}\log(\sin^2(\theta/2)) = \cot(\theta/2)$$

Für $\theta = \pi/3$*:*

$$\frac{\partial}{\partial\theta}\log \pi_{\pi/3}(a_0|s_0) = -\tan(\pi/6) = -\frac{1}{\sqrt{3}}$$

10.8 Vorteile und Limitationen

10.8.1 Theoretische Vorteile

Exponentieller Hilbert-Raum: Mit n Qubits können 2^n Zustände superponiert werden, was potenziell exponentiell mehr Information verarbeitet als n klassische Bits.

Quanteninterferenz: Destruktive Interferenz kann unerwünschte Pfade unterdrücken und zur Lösungsfindung beitragen.

10.8.2 Praktische Limitationen

Barren Plateaus: In tiefen Quantenschaltungen können Gradienten exponentiell klein werden:

$$\text{Var}[\nabla_{\theta_k} f] \leq \frac{c}{4^n}$$

für eine Kostenfunktion f mit n Qubits.

Measurement Shot Noise: Die Varianz von Messungen skaliert als $O(1/S)$ mit der Anzahl der Shots S.

Rechnung 43 (Shot Noise Analyse)**.** *Für eine Observable O mit Erwartungswert $\langle O \rangle = \mu$ und Varianz $Var(O) = \sigma^2$:*

Nach S Messungen ist der geschätzte Erwartungswert:

$$\hat{\mu} = \frac{1}{S} \sum_{i=1}^{S} o_i$$

mit Standardfehler:

$$SE(\hat{\mu}) = \frac{\sigma}{\sqrt{S}}$$

Für 95

$$S \geq \left(\frac{1.96\sigma}{\epsilon} \right)^2$$

Beispiel: Für $\sigma = 1$ und $\epsilon = 0.01$:

$$S \geq \left(\frac{1.96}{0.01} \right)^2 = 38,416 \text{ } \textit{Shots}$$

10.9 Aktuelle Forschungsrichtungen

10.9.1 Hybrid Classical-Quantum Algorithms

Viele praktische QML-Algorithmen nutzen eine Kombination aus klassischer und quantischer Verarbeitung:

Quantum-Enhanced Feature Maps:

$$\vec{x} \xrightarrow{\text{klassisch}} \vec{x}' \xrightarrow{\text{quantum}} |\Phi(\vec{x}')\rangle \xrightarrow{\text{klassisch}} f(\vec{x})$$

10.9.2 Near-term Applications

Quantum Advantage in Small Data Regimes: Bei begrenzten Trainingsdaten können Quantenmodelle aufgrund ihrer reichhaltigen Feature-Spaces Vorteile bieten.

Optimization Landscapes: Quantenschaltungen können komplexe, nichtkonvexe Optimierungslandschaften erzeugen, die möglicherweise bessere lokale Minima enthalten.

11 Fazit

Das Quantencomputing ist eine transformative Technologie, die das Potenzial hat, die Art und Weise, wie wir komplexe Probleme lösen, grundlegend zu verändern. Durch die Nutzung der einzigartigen Phänomene der Quantenmechanik Superposition, Verschränkung und Interferenz können Quantencomputer bestimmte Aufgaben, die für klassische Computer unerreichbar sind, effizient bewältigen.

12 Quantenmaschinenlernen - Mathematische Grundlagen und Algorithmen

Das Quantenmaschinenlernen (Quantum Machine Learning, QML) vereint die Prinzipien des Quantencomputings mit Techniken des maschinellen Lernens. Dieses interdisziplinäre Feld verspricht exponentiell verbesserte Lernalgorithmen für bestimmte Problemklassen und eröffnet neue Möglichkeiten in der Datenanalyse und Mustererkennung.

12.1 Grundlagen des Quantenmaschinenlernens

12.1.1 Quantendaten und Feature Maps

Klassische Daten müssen in Quantenzustände kodiert werden. Eine Feature Map $\Phi : \mathcal{X} \to \mathcal{H}$ bildet klassische Datenpunkte $x \in \mathcal{X}$ auf Quantenzustände $|\Phi(x)\rangle \in \mathcal{H}$ ab.

Amplitude Encoding: Für einen Datenvektor $\vec{x} = (x_1, x_2, ..., x_N)$ mit $\|\vec{x}\| = 1$:

$$|\Phi(\vec{x})\rangle = \sum_{i=1}^{N} x_i |i\rangle$$

Angle Encoding:

$$|\Phi(\vec{x})\rangle = \bigotimes_{i=1}^{n} (\cos(x_i)|0\rangle + \sin(x_i)|1\rangle)$$

Rechnung 44 (Feature Map für 2D-Daten)**.** *Für einen 2D-Datenpunkt* $\vec{x} = (x_1, x_2)$ *mit Angle Encoding:*

$$|\Phi(\vec{x})\rangle = (\cos(x_1)|0\rangle + \sin(x_1)|1\rangle) \otimes (\cos(x_2)|0\rangle + \sin(x_2)|1\rangle)$$

Ausmultipliziert:

$$|\Phi(\vec{x})\rangle = \cos(x_1)\cos(x_2)\,|00\rangle + \cos(x_1)\sin(x_2)\,|01\rangle \tag{145}$$
$$+ \sin(x_1)\cos(x_2)\,|10\rangle + \sin(x_1)\sin(x_2)\,|11\rangle \tag{146}$$

Für $\vec{x} = (\pi/4, \pi/3)$:

$$|\Phi(\vec{x})\rangle = \frac{1}{\sqrt{2}} \cdot \frac{1}{2}\,|00\rangle + \frac{1}{\sqrt{2}} \cdot \frac{\sqrt{3}}{2}\,|01\rangle \tag{147}$$
$$+ \frac{1}{\sqrt{2}} \cdot \frac{1}{2}\,|10\rangle + \frac{1}{\sqrt{2}} \cdot \frac{\sqrt{3}}{2}\,|11\rangle \tag{148}$$
$$= \frac{1}{2\sqrt{2}}(|00\rangle + \sqrt{3}\,|01\rangle + |10\rangle + \sqrt{3}\,|11\rangle) \tag{149}$$

12.2 Variational Quantum Circuits

Variational Quantum Circuits (VQCs) sind parametrisierte Quantenschaltungen, die als Quantenanaloga neuronaler Netze fungieren.

12.2.1 Mathematische Struktur

Ein VQC wird durch einen parametrisierten unitären Operator $U(\vec{\theta})$ beschrieben:

$$|\psi(\vec{\theta})\rangle = U(\vec{\theta})\,|0\rangle^{\otimes n}$$

Layered Structure:

$$U(\vec{\theta}) = \prod_{l=1}^{L} U_l(\vec{\theta^{(l)}})$$

wobei jede Schicht U_l aus Rotations- und Verschränkungsgattern besteht.

Rechnung 45 (2-Qubit VQC mit einer Schicht)**.** *Betrachten wir einen 2-Qubit VQC:*

$$U(\vec{\theta}) = CNOT_{12} \cdot R_y(\theta_2) \otimes R_y(\theta_1)$$

wobei $\vec{\theta} = (\theta_1, \theta_2)$.
Schritt 1: Rotation

$$R_y(\theta_1) \otimes R_y(\theta_2)\,|00\rangle = (\cos(\theta_1/2)\,|0\rangle + \sin(\theta_1/2)\,|1\rangle) \otimes (\cos(\theta_2/2)\,|0\rangle + \sin(\theta_2/2)\,|1\rangle)$$

$$= \cos(\theta_1/2)\cos(\theta_2/2)\,|00\rangle + \cos(\theta_1/2)\sin(\theta_2/2)\,|01\rangle + \sin(\theta_1/2)\cos(\theta_2/2)\,|10\rangle + \sin(\theta_1/2)\sin(\theta_2/2)\,|1$$

Schritt 2: CNOT

$$|\psi(\vec{\theta})\rangle = \cos(\theta_1/2)\cos(\theta_2/2)\,|00\rangle + \cos(\theta_1/2)\sin(\theta_2/2)\,|01\rangle + \sin(\theta_1/2)\cos(\theta_2/2)\,|11\rangle + \sin(\theta_1/2)\sin(\theta$$

Für $\vec{\theta} = (\pi/2, \pi/4)$:

$$|\psi\rangle = \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}\,|00\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}\,|01\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}\,|11\rangle + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}\,|10\rangle$$

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

12.3 Quantum Neural Networks

12.3.1 Quantum Perceptron

Ein Quantum Perceptron verarbeitet Eingabedaten durch eine Sequenz von Quantengattern und erzeugt eine Ausgabe durch Messung.

Mathematische Beschreibung:

$$f(\vec{x}, \vec{\theta}) = \langle 0^{\otimes n}|U^\dagger(\vec{\theta})MU(\vec{\theta})|\Phi(\vec{x})\rangle$$

wobei M ein Messoperator ist (z.B. $M = Z \otimes I^{\otimes(n-1)}$).

Rechnung 46 (Quantum Perceptron für Klassifikation)**.** *Für ein 2-Klassen-Problem mit $y \in \{-1, +1\}$:*

Kostenfunktion:

$$\mathcal{L}(\vec{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - f(\vec{x}^{(i)}, \vec{\theta})\right)^2$$

Gradientenberechnung (Parameter-Shift Rule):

$$\frac{\partial f}{\partial \theta_j} = \frac{1}{2}\left[f(\vec{\theta}^{(j+)}) - f(\vec{\theta}^{(j-)})\right]$$

wobei $\vec{\theta}^{(j\pm)} = \vec{\theta} \pm \frac{\pi}{2}\vec{e}_j$.
Beispiel für θ_1:

$$\frac{\partial f}{\partial \theta_1} = \frac{1}{2}\left[f(\theta_1 + \pi/2, \theta_2, ...) - f(\theta_1 - \pi/2, \theta_2, ...)\right] \tag{150}$$

12.4 Quantum Support Vector Machines

12.4.1 Kernel-basierte Methoden

Quantum SVMs nutzen quanteninduzierte Feature-Spaces, um nichtlinear separierbare Daten zu klassifizieren.

Quantum Kernel:

$$K(\vec{x}_i, \vec{x}_j) = |\langle \Phi(\vec{x}_i) | \Phi(\vec{x}_j) \rangle|^2$$

Rechnung 47 (Berechnung des Quantum Kernels)**.** *Für zwei Datenpunkte* $\vec{x}_1 = (x_{11}, x_{12})$ *und* $\vec{x}_2 = (x_{21}, x_{22})$ *mit Angle Encoding:*

$$\langle \Phi(\vec{x}_1) | \Phi(\vec{x}_2) \rangle = \cos(x_{11})\cos(x_{21}) + \cos(x_{11})\sin(x_{21})e^{i(x_{22}-x_{12})} + \sin(x_{11})\cos(x_{21})e^{i(x_{12}-x_{22})} + \sin(x_{11})$$

Vereinfachung für reelle Winkel:

$$= \cos(x_{11})\cos(x_{21}) + \sin(x_{11})\sin(x_{21}) + \cos(x_{11})\sin(x_{21})\cos(x_{22}-x_{12}) + \sin(x_{11})\cos(x_{21})\cos(x_{12}-x_{2}$$

$$= \cos(x_{11}-x_{21}) + \cos(x_{11})\sin(x_{21})\cos(x_{22}-x_{12}) + \sin(x_{11})\cos(x_{21})\cos(x_{12}-x_{22})$$

Für $\vec{x}_1 = (\pi/4, \pi/6)$ *und* $\vec{x}_2 = (\pi/3, \pi/4)$*:*

$$|\langle \Phi(\vec{x}_1) | \Phi(\vec{x}_2) \rangle|^2 \approx 0.854$$

12.5 Quantum Principal Component Analysis

12.5.1 Eigenvalue Estimation

Quantum PCA nutzt die Quantenphasenschätzung zur Bestimmung der Eigenwerte einer Kovarianzmatrix.

Algorithmus: 1. Bereite die Datenmatrix $\rho = \frac{1}{N}\sum_{i=1}^{N} |x_i\rangle \langle x_i|$ vor 2. Wende Quantenphasenschätzung auf ρ an 3. Extrahiere die Hauptkomponenten

Rechnung 48 (Quantum PCA für 2Œ2 Matrix)**.** *Betrachte die Kovarianzmatrix:*

$$C = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Eigenwerte und Eigenvektoren: *Charakteristisches Polynom:* $\det(C - \lambda I) = (2-\lambda)^2 - 1 = \lambda^2 - 4\lambda + 3 = 0$

Eigenwerte: $\lambda_1 = 3$, $\lambda_2 = 1$
Eigenvektoren:

$$|v_1\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix}, \quad |v_2\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

Quantenphasenschätzung: *Für den Zustand* $|v_1\rangle$ *liefert die Phasenschätzung:*

$$e^{i\phi_1} = e^{i\cdot 2\pi\cdot 3/4} = e^{i3\pi/2} = -i$$

Die Phase $\phi_1 = 3/4$ *entspricht dem Eigenwert* $\lambda_1 = 3$.

12.6 Quantum Generative Models

12.6.1 Quantum Generative Adversarial Networks

Quantum GANs verwenden zwei Quantenschaltungen: einen Generator G und einen Diskriminator D.

Generator:

$$G : \mathcal{Z} \to \mathcal{X} \quad \text{mit} \quad |\psi_G(z, \vec{\theta}_G)\rangle = U_G(\vec{\theta}_G)\,|z\rangle$$

Diskriminator:

$$D : \mathcal{X} \to [0,1] \quad \text{mit} \quad D(x, \vec{\theta}_D) = \langle M \rangle_{U_D(\vec{\theta}_D)|x\rangle}$$

Rechnung 49 (Minimax-Optimierung für QGAN)**.** ***Kostenfunktion:***

$$\min_{\vec{\theta}_G} \max_{\vec{\theta}_D} V(\vec{\theta}_D, \vec{\theta}_G) = \mathbb{E}_{x\sim p_{data}}[\log D(x, \vec{\theta}_D)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z, \vec{\theta}_G), \vec{\theta}_D))]$$

Diskriminator-Update:

$$\vec{\theta}_D^{(t+1)} = \vec{\theta}_D^{(t)} + \alpha_D \nabla_{\vec{\theta}_D} V(\vec{\theta}_D^{(t)}, \vec{\theta}_G^{(t)})$$

Generator-Update:

$$\vec{\theta}_G^{(t+1)} = \vec{\theta}_G^{(t)} - \alpha_G \nabla_{\vec{\theta}_G} V(\vec{\theta}_D^{(t+1)}, \vec{\theta}_G^{(t)})$$

Für einen einfachen Fall mit einem Parameter θ_G*:*

$$\frac{\partial V}{\partial \theta_G} = -\mathbb{E}_{z\sim p_z}\left[\frac{1}{1 - D(G(z, \theta_G))}\frac{\partial D(G(z, \theta_G))}{\partial \theta_G}\right]$$

12.7 Quantum Reinforcement Learning

12.7.1 Variational Quantum Policy Gradient

Quantum RL nutzt Quantenschaltungen zur Darstellung von Policies in Verstärkungslernen.

Quantenpolicy:

$$\pi_{\vec{\theta}}(a|s) = |\langle a|U(\vec{\theta})|s\rangle|^2$$

Policy Gradient:

$$\nabla_{\vec{\theta}} J(\vec{\theta}) = \mathbb{E}_{\tau\sim\pi_{\vec{\theta}}}\left[\sum_{t=0}^{T} \nabla_{\vec{\theta}} \log \pi_{\vec{\theta}}(a_t|s_t) R(\tau)\right]$$

Rechnung 50 (Quantum Policy für 2-State MDP)**.** *Betrachte ein 2-State MDP mit Zuständen* $\{s_0, s_1\}$ *und Aktionen* $\{a_0, a_1\}$*.*

Quantenpolicy:

$$\pi_\theta(a_0|s_0) = |\cos(\theta/2)|^2, \quad \pi_\theta(a_1|s_0) = |\sin(\theta/2)|^2$$

Log-Policy Gradient:

$$\frac{\partial}{\partial\theta} \log \pi_\theta(a_0|s_0) = \frac{\partial}{\partial\theta} \log(\cos^2(\theta/2)) = -\tan(\theta/2)$$

$$\frac{\partial}{\partial\theta} \log \pi_\theta(a_1|s_0) = \frac{\partial}{\partial\theta} \log(\sin^2(\theta/2)) = \cot(\theta/2)$$

Für $\theta = \pi/3$*:*

$$\frac{\partial}{\partial\theta} \log \pi_{\pi/3}(a_0|s_0) = -\tan(\pi/6) = -\frac{1}{\sqrt{3}}$$

12.8 Vorteile und Limitationen

12.8.1 Theoretische Vorteile

Exponentieller Hilbert-Raum: Mit n Qubits können 2^n Zustände superponiert werden, was potenziell exponentiell mehr Information verarbeitet als n klassische Bits.

Quanteninterferenz: Destruktive Interferenz kann unerwünschte Pfade unterdrücken und zur Lösungsfindung beitragen.

12.8.2 Praktische Limitationen

Barren Plateaus: In tiefen Quantenschaltungen können Gradienten exponentiell klein werden:

$$\text{Var}[\nabla_{\theta_k} f] \leq \frac{c}{4^n}$$

für eine Kostenfunktion f mit n Qubits.

Measurement Shot Noise: Die Varianz von Messungen skaliert als $O(1/S)$ mit der Anzahl der Shots S.

Rechnung 51 (Shot Noise Analyse)**.** *Für eine Observable O mit Erwartungswert $\langle O \rangle = \mu$ und Varianz $Var(O) = \sigma^2$:*

Nach S Messungen ist der geschätzte Erwartungswert:

$$\hat{\mu} = \frac{1}{S} \sum_{i=1}^{S} o_i$$

mit Standardfehler:

$$SE(\hat{\mu}) = \frac{\sigma}{\sqrt{S}}$$

Für 95

$$S \geq \left(\frac{1.96\sigma}{\epsilon} \right)^2$$

Beispiel: Für $\sigma = 1$ und $\epsilon = 0.01$:

$$S \geq \left(\frac{1.96}{0.01} \right)^2 = 38,416 \text{ } \textit{Shots}$$

12.9 Aktuelle Forschungsrichtungen

12.9.1 Hybrid Classical-Quantum Algorithms

Viele praktische QML-Algorithmen nutzen eine Kombination aus klassischer und quantischer Verarbeitung:

Quantum-Enhanced Feature Maps:

$$\vec{x} \xrightarrow{\text{klassisch}} \vec{x}' \xrightarrow{\text{quantum}} |\Phi(\vec{x}')\rangle \xrightarrow{\text{klassisch}} f(\vec{x})$$

12.9.2 Near-term Applications

Quantum Advantage in Small Data Regimes: Bei begrenzten Trainingsdaten können Quantenmodelle aufgrund ihrer reichhaltigen Feature-Spaces Vorteile bieten.

Optimization Landscapes: Quantenschaltungen können komplexe, nichtkonvexe Optimierungslandschaften erzeugen, die möglicherweise bessere lokale Minima enthalten.

13 Quantum Generative Adversarial Networks - Umfassende mathematische Behandlung

Quantum Generative Adversarial Networks (QGANs) stellen eine revolutionäre Verbindung zwischen Quantencomputing und generativer Modellierung dar. Diese Systeme nutzen die einzigartigen Eigenschaften der Quantenmechanik, um komplexe Wahrscheinlichkeitsverteilungen zu lernen und neue Daten zu generieren, die statistisch den Trainingsdaten ähneln.

13.1 Grundlegende Konzepte und mathematische Formulierung

13.1.1 Klassische GANs als Ausgangspunkt

Klassische Generative Adversarial Networks basieren auf einem Minimax-Spiel zwischen zwei neuronalen Netzen:

Generator: $G : \mathcal{Z} \to \mathcal{X}$ mit $G(z; \theta_G)$ **Diskriminator:** $D : \mathcal{X} \to [0, 1]$ mit $D(x; \theta_D)$

Minimax-Zielfunktion: $\min_{\theta_G} \max_{\theta_D} V(\theta_D, \theta_G) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$

13.1.2 Quantenerweiterung der GAN-Architektur

In QGANs werden die klassischen neuronalen Netze durch parametrisierte Quantenschaltungen ersetzt:

Quantum Generator: $G_Q : \mathcal{Z} \to \mathcal{H} \quad \text{mit} \quad G_Q(z, \vec{\theta}_G) = U_G(\vec{\theta}_G) \left|z\right\rangle$

Quantum Discriminator: $D_Q : \mathcal{H} \to [0, 1] \quad \text{mit} \quad D_Q(\left|\psi\right\rangle, \vec{\theta}_D) = \left\langle\psi\right| U_D^\dagger(\vec{\theta}_D) M U_D(\vec{\theta}_D) \left|\psi\right\rangle$
wobei M ein Messoperator ist (typischerweise $M = \left|0\right\rangle\left\langle 0\right| \otimes I^{\otimes(n-1)}$).

Rechnung 52 (Einfaches 2-Qubit QGAN Setup)**.** *Betrachten wir ein minimales QGAN mit 2 Qubits:*

Rauschzustand: $\left|z\right\rangle = \left|00\right\rangle$

Generator: $U_G(\theta_1, \theta_2) = R_y(\theta_2) \otimes R_y(\theta_1)$

Generierter Zustand:

$$
\begin{aligned}
G_Q(\left|00\right\rangle, \vec{\theta}_G) &= R_y(\theta_2) \otimes R_y(\theta_1) \left|00\right\rangle && (151)\\
&= (\cos(\theta_2/2) \left|0\right\rangle + \sin(\theta_2/2) \left|1\right\rangle) \otimes (\cos(\theta_1/2) \left|0\right\rangle + \sin(\theta_1/2) \left|1\right\rangle) && (152)\\
&= \cos(\theta_1/2)\cos(\theta_2/2) \left|00\right\rangle + \cos(\theta_1/2)\sin(\theta_2/2) \left|01\right\rangle && (153)\\
&\quad + \sin(\theta_1/2)\cos(\theta_2/2) \left|10\right\rangle + \sin(\theta_1/2)\sin(\theta_2/2) \left|11\right\rangle && (154)
\end{aligned}
$$

Diskriminator: $U_D(\phi_1, \phi_2) = CNOT_{12} \cdot (R_z(\phi_2) \otimes R_z(\phi_1))$

Für Parameter $\vec{\theta}_G = (\pi/4, \pi/3)$*:* $|\psi_G\rangle = \frac{\sqrt{3}}{2\sqrt{2}} |00\rangle + \frac{1}{2\sqrt{2}} |01\rangle + \frac{\sqrt{3}}{2\sqrt{2}} |10\rangle + \frac{1}{2\sqrt{2}} |11\rangle$

13.2 Quantenvorteil in generativen Modellen

13.2.1 Exponentieller Zustandsraum

Ein n-Qubit Quantensystem kann 2^n Amplituden gleichzeitig repräsentieren, was einem exponentiellen Vorteil gegenüber klassischen Systemen entspricht.

Quantenzustandsdarstellung: $|\psi(\vec{\theta})\rangle = \sum_{i=0}^{2^n-1} \alpha_i(\vec{\theta}) |i\rangle$

wobei $\sum_i |\alpha_i(\vec{\theta})|^2 = 1$.

Born-Regel für Sampling: $P(x = i) = |\alpha_i(\vec{\theta})|^2$

Rechnung 53 (Zustandsraumvergleich)**.** ***Klassisches System:*** *Ein klassisches System mit n Bits kann* 2^n *verschiedene Zustände annehmen, aber nur einen zur Zeit.*

Quantensystem: *Ein n-Qubit System mit p Parametern kann bis zu* 2^n *Wahrscheinlichkeits-amplituden gleichzeitig manipulieren.*

Für $n = 10$ *Qubits: - Klassisch: 1 aus* $2^{10} = 1024$ *Zuständen - Quantum: Superposition aller 1024 Zustände gleichzeitig*

Parametereffizienz: Mit $p = 2n = 20$ *Parametern können exponentiell viele (*2^{10}*) Wahrscheinlichkeiten kontrolliert werden.*

13.3 Verschiedene QGAN-Architekturen

13.3.1 Patch Quantum GANs

Für hochdimensionale Daten werden lokale Quantenschaltungen auf Patches angewendet.

Patch-Zerlegung: Ein Datenpunkt $\vec{x} \in \mathbb{R}^{N \times N}$ wird in Patches $P_{i,j}$ der GröSSe $k \times k$ zerlegt: $\vec{x} = \bigcup_{i,j} P_{i,j}$

Lokaler Quantum Generator: $G_{\text{patch}}(z_{i,j}, \vec{\theta}_{i,j}) = U_G(\vec{\theta}_{i,j}) |z_{i,j}\rangle$

Rechnung 54 (4Œ4 Bild mit 2Œ2 Patches)**.** *Für ein* 4×4 *Bild mit* 2×2 *Patches:*

Patches: $P_{1,1}, P_{1,2}, P_{2,1}, P_{2,2}$

Jeder Patch wird durch 2 Qubits repräsentiert: $|\psi_{i,j}\rangle = \sum_{k=0}^{3} \alpha_k^{(i,j)} |k\rangle$

Gesamtzustand: $|\Psi\rangle = \bigotimes_{i,j} |\psi_{i,j}\rangle$

Dimensionalität: $2^{2\times4} = 2^8 = 256$ *Zustände für das gesamte Bild.*

Parameter pro Patch: 6 (für 1-Qubit Rotationen in 3 Richtungen auf 2 Qubits) Gesamtparameter: $4 \times 6 = 24$

13.3.2 Hierarchical Quantum GANs

Hierarchische QGANs nutzen mehrstufige Generierung für komplexe Strukturen.

Multiskalige Architektur: $G_{\text{hier}} = G_{\text{fine}} \circ G_{\text{coarse}}$

Grober Generator: $G_{\text{coarse}} : \mathcal{Z}_1 \to \mathcal{H}_1$ erzeugt globale Struktur **Feiner Generator:** $G_{\text{fine}} : \mathcal{Z}_2 \times \mathcal{H}_1 \to \mathcal{H}_2$ verfeinert Details

Rechnung 55 (2-Level Hierarchisches QGAN)**.** ***Level 1 (Grob):*** *3 Qubits für globale Struktur* $|\psi_1\rangle = U_1(\vec{\theta}_1)\,|000\rangle$

Level 2 (Fein): *6 Qubits für Details, konditioniert auf Level 1* $|\psi_2\rangle = U_2(\vec{\theta}_2, \vec{\psi}_1)\,|000000\rangle$

wobei $\vec{\theta}_2$ *von den Messergebnissen von* $|\psi_1\rangle$ *abhängt.*

Bedingte Parameter: $\theta_2^{(i)} = f_i(\langle Z_1\rangle, \langle Z_2\rangle, \langle Z_3\rangle) + \theta_2^{(i,0)}$

Beispiel für f_i*:* $f_i(z_1, z_2, z_3) = w_{i1}z_1 + w_{i2}z_2 + w_{i3}z_3$

13.4 Verlustfunktionen und Optimierung

13.4.1 Quantenwasserstein-Distanz

Die Wasserstein-Distanz wird für Quantenverteilungen erweitert:

Klassische Wasserstein-1 Distanz: $W_1(P, Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(x,y)\sim\gamma}[|x - y|]$

Quantum Wasserstein Distanz: $W_1^Q(\rho, \sigma) = \inf_\Gamma \text{Tr}[\Gamma \cdot C]$

wobei Γ eine Quantenkopplung zwischen ρ und σ ist und C die Kostenmatrix.

Rechnung 56 (Wasserstein-Distanz für 2-Qubit Zustände)**.** *Für zwei 2-Qubit Zustände:* $\rho = \frac{1}{2}(|00\rangle\langle 00| + |11\rangle\langle 11|)$ $\sigma = \frac{1}{2}(|01\rangle\langle 01| + |10\rangle\langle 10|)$

Kostenmatrix (Hamming-Distanz): $C = \begin{pmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 2 & 1 & 1 & 0 \end{pmatrix}$

Optimale Kopplung: $\Gamma^* = \frac{1}{4}\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$

Wasserstein-Distanz: $W_1^Q(\rho, \sigma) = Tr[\Gamma^* \cdot C] = \frac{1}{4}(1\cdot 1 + 1\cdot 1 + 1\cdot 1 + 1\cdot 1 + 1\cdot 1 + 1\cdot 1) = 1.5$

13.4.2 Quantum Jensen-Shannon Divergenz

Die JS-Divergenz wird für Quantenzustände erweitert:

Klassische JS-Divergenz: $\mathrm{JS}(P,Q) = \frac{1}{2}\mathrm{KL}(P,M) + \frac{1}{2}\mathrm{KL}(Q,M)$
wobei $M = \frac{1}{2}(P+Q)$.

Quantum JS-Divergenz: $\mathrm{JS}(\rho,\sigma) = \frac{1}{2}S(\rho\|\mu) + \frac{1}{2}S(\sigma\|\mu)$
wobei $\mu = \frac{1}{2}(\rho+\sigma)$ und $S(\rho\|\sigma) = \mathrm{Tr}[\rho(\log\rho - \log\sigma)]$ die Quantenrelative Entropie ist.

Rechnung 57 (JS-Divergenz für verschränkte Zustände)**.** *Für die Bell-Zustände:* $\rho = |\Phi^+\rangle\langle\Phi^+| = \frac{1}{2}(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)$ $\sigma = |\Psi^+\rangle\langle\Psi^+| = \frac{1}{2}(|01\rangle\langle 01| + |01\rangle\langle 10| + |10\rangle\langle 01| + |10\rangle\langle 10|)$

Gemischter Zustand: $\mu = \frac{1}{2}(\rho+\sigma) = \frac{1}{4}(|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10| + |11\rangle\langle 11|) = \frac{I}{4}$

Von-Neumann-Entropien:

$$S(\rho) = -Tr[\rho\log\rho] = 1\ bit \tag{155}$$
$$S(\sigma) = -Tr[\sigma\log\sigma] = 1\ bit \tag{156}$$
$$S(\mu) = -Tr[\mu\log\mu] = 2\ bits \tag{157}$$

JS-Divergenz: $JS(\rho,\sigma) = \frac{1}{2}(1-2) + \frac{1}{2}(1-2) = 1\ bit$

13.5 Trainingsalgorithmen für QGANs

13.5.1 Simultaneous Perturbation Stochastic Approximation (SPSA)

Da Quantengradienten schwer zu berechnen sind, wird oft SPSA verwendet:

SPSA-Update: $\vec{\theta}^{(k+1)} = \vec{\theta}^{(k)} - a_k \frac{f(\vec{\theta}^{(k)} + c_k\vec{\Delta}^{(k)}) - f(\vec{\theta}^{(k)} - c_k\vec{\Delta}^{(k)})}{2c_k}\vec{\Delta}^{(k)}$
wobei $\vec{\Delta}^{(k)}$ ein zufälliger Störvektor ist (typischerweise Rademacher-verteilt).

Rechnung 58 (SPSA für 3-Parameter Quantum Generator)**.** *Für* $\vec{\theta} = (\theta_1,\theta_2,\theta_3)$ *und* $\vec{\Delta} = (\delta_1,\delta_2,\delta_3)$ *mit* $\delta_i \in \{-1,+1\}$*:*

Funktionsauswertungen: $f^+ = f(\theta_1 + c\delta_1, \theta_2 + c\delta_2, \theta_3 + c\delta_3)$ $f^- = f(\theta_1 - c\delta_1, \theta_2 - c\delta_2, \theta_3 - c\delta_3)$

Gradientenschätzung: $\hat{g}_i = \frac{f^+ - f^-}{2c\delta_i}$

Parameterupdate: $\theta_i^{(k+1)} = \theta_i^{(k)} - a_k\hat{g}_i$

Beispiel mit $c = 0.1$, $a = 0.01$, $\vec{\Delta} = (1,-1,1)$*: Wenn* $f^+ = 0.7$ *und* $f^- = 0.5$*:* $\hat{g}_1 = \frac{0.7-0.5}{2\cdot 0.1\cdot 1} = 1.0$ $\hat{g}_2 = \frac{0.7-0.5}{2\cdot 0.1\cdot(-1)} = -1.0$ $\hat{g}_3 = \frac{0.7-0.5}{2\cdot 0.1\cdot 1} = 1.0$

13.5.2 Parameter-Shift Rule für QGANs

Die Parameter-Shift Rule ermöglicht exakte Gradientenberechnung für Quantenschaltungen:

Gradient eines Erwartungswerts: $\frac{\partial}{\partial\theta_j}\langle H\rangle = \frac{1}{2}\left[\langle H\rangle_{\theta_j+\pi/2} - \langle H\rangle_{\theta_j-\pi/2}\right]$

Rechnung 59 (Parameter-Shift für Generator-Verlust)**.** *Für die Generator-Verlustfunktion* $L_G = -\mathbb{E}[\log D(G(z))]$*:*

Erwartungswert für einen Parameter: $\langle H_G(\theta_1)\rangle = \langle\psi(\theta_1)|\, H_G\, |\psi(\theta_1)\rangle$

Parameter-Shift Gradient: $\frac{\partial L_G}{\partial\theta_1} = \frac{1}{2}\left[\langle H_G\rangle_{\theta_1+\pi/2} - \langle H_G\rangle_{\theta_1-\pi/2}\right]$

Für $\theta_1 = \pi/4$*:*

$$\frac{\partial L_G}{\partial\theta_1} = \frac{1}{2}\left[\langle H_G\rangle_{3\pi/4} - \langle H_G\rangle_{-\pi/4}\right] \tag{158}$$

$$= \frac{1}{2}\left[\langle H_G\rangle_{3\pi/4} - \langle H_G\rangle_{7\pi/4}\right] \tag{159}$$

Wenn $\langle H_G\rangle_{3\pi/4} = 0.8$ *und* $\langle H_G\rangle_{7\pi/4} = 0.2$*:* $\frac{\partial L_G}{\partial\theta_1} = \frac{1}{2}(0.8 - 0.2) = 0.3$

13.6 Anwendungen und Benchmarks

13.6.1 Diskrete Verteilungen

QGANs sind besonders effektiv für diskrete Wahrscheinlichkeitsverteilungen.

Zielverteilung - Bars and Stripes: Für 2×2 Bilder mit horizontalen und vertikalen Balken: $P_{\text{target}}(\text{horizontal}) = P_{\text{target}}(\text{vertikal}) = 0.5$

Rechnung 60 (QGAN für Bars and Stripes)**.** ***Zustände:*** *- Horizontal:* $|H_1\rangle = |0011\rangle$*,* $|H_2\rangle = |1100\rangle$ *- Vertikal:* $|V_1\rangle = |0101\rangle$*,* $|V_2\rangle = |1010\rangle$

Zielzustand: $|\psi_{target}\rangle = \frac{1}{2}(|0011\rangle + |1100\rangle + |0101\rangle + |1010\rangle)$

Generator-Architektur: *4-Qubit Schaltung mit Parametern* $\vec{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4)$*:* $U_G(\vec{\theta}) = \prod_{i=1}^{4} R_y(\theta_i) \cdot$ *CNOT-Layer*

Trainingsziel: *Minimiere* $D_{KL}(P_{target} \| P_G(\vec{\theta}))$

Nach dem Training mit optimalen Parametern $\vec{\theta}^* = (\pi/3, \pi/2, \pi/4, \pi/6)$*:* $|\psi_G(\vec{\theta}^*)\rangle \approx 0.48\,|0011\rangle + 0.52\,|1100\rangle + 0.49\,|0101\rangle + 0.51\,|1010\rangle$

Fidelity: $|\langle\psi_{target}|\psi_G(\vec{\theta}^*)\rangle|^2 = 0.97$

13.6.2 Kontinuierliche Verteilungen durch Diskretisierung

Für kontinuierliche Daten wird Diskretisierung verwendet.

GauSS-Verteilung Approximation: Eine 1D GauSS-Verteilung $\mathcal{N}(\mu, \sigma^2)$ wird auf 2^n Bins diskretisiert: $P(x_i) = \int_{x_i-\Delta/2}^{x_i+\Delta/2} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx$

Rechnung 61 (3-Qubit GauSS-Approximation)**.** *Für $n = 3$ Qubits (8 Bins) und $\mathcal{N}(0,1)$ auf $[-3,3]$:*

Bin-Grenzen: $x_i = -3 + i \cdot \frac{6}{8} = -3 + 0.75i$ *für* $i = 0, ..., 7$

Wahrscheinlichkeiten:

$$P(x_0) = \Phi(-2.625) - \Phi(-3) \approx 0.004 \tag{160}$$
$$P(x_1) = \Phi(-1.875) - \Phi(-2.625) \approx 0.026 \tag{161}$$
$$P(x_2) = \Phi(-1.125) - \Phi(-1.875) \approx 0.099 \tag{162}$$
$$P(x_3) = \Phi(-0.375) - \Phi(-1.125) \approx 0.225 \tag{163}$$
$$P(x_4) = \Phi(0.375) - \Phi(-0.375) \approx 0.292 \tag{164}$$
$$P(x_5) = \Phi(1.125) - \Phi(0.375) \approx 0.225 \tag{165}$$
$$P(x_6) = \Phi(1.875) - \Phi(1.125) \approx 0.099 \tag{166}$$
$$P(x_7) = \Phi(3) - \Phi(1.875) \approx 0.030 \tag{167}$$

Zielzustand: $|\psi_{Gauss}\rangle = \sqrt{0.004}\,|000\rangle + \sqrt{0.026}\,|001\rangle + ... + \sqrt{0.030}\,|111\rangle$

Normierung: $\sum_{i=0}^{7} P(x_i) = 1.000$

13.7 Herausforderungen und Limitationen

13.7.1 Barren Plateaus in QGANs

Tiefe Quantenschaltungen leiden unter exponentiell verschwindenden Gradienten:

Varianz der Gradienten: Für eine L-schichtige Quantenschaltung mit zufälligen Parametern: $\text{Var}\left[\frac{\partial L}{\partial \theta_l}\right] \leq \frac{C}{4^L}$

Rechnung 62 (Plateau-Effekt Quantifizierung)**.** *Für eine 10-schichtige QGAN-Architektur:*

Gradientenvarianz: $Var\left[\frac{\partial L}{\partial \theta}\right] \leq \frac{C}{4^{10}} = \frac{C}{1{,}048{,}576}$

Signal-zu-Rauschen Verhältnis: $SNR = \frac{|\nabla L|^2}{Var[\nabla L]} \propto 4^L$

Für ein nutzbares SNR von mindestens 1: $|\nabla L| \geq \sqrt{\frac{C}{4^L}}$

Bei $L = 10$ *und* $C = 1$*:* $|\nabla L| \geq 9.77 \times 10^{-4}$

Anzahl benötigter Messungen für Gradientenschätzung: $N_{shots} \geq \frac{4^L}{|\nabla L|^2} = 4^{10} = 1,048,576$

13.7.2 Mode Collapse in QGANs

Mode Collapse tritt auf, wenn der Quantum Generator nur einen Teil der Zielverteilung lernt.

Quantifizierung durch Inception Score: $\text{IS} = \exp\left(\mathbb{E}_{x\sim P_G}[\text{KL}(p(y|x)\|p(y))]\right)$
Quantum Mode Coverage: $\text{QMC} = \sum_{i=1}^{N} \min(P_{\text{real}}(x_i), P_G(x_i))$

Rechnung 63 (Mode Collapse Detektion)**.** *Für eine 4-Mode Zielverteilung:* $P_{target} = 0.25(\delta_{x_1} + \delta_{x_2} + \delta_{x_3} + \delta_{x_4})$

Gesunder Generator: $P_G^{gesund} = 0.23\delta_{x_1} + 0.27\delta_{x_2} + 0.24\delta_{x_3} + 0.26\delta_{x_4}$
$QMC_{gesund} = \min(0.25, 0.23)+\min(0.25, 0.27)+\min(0.25, 0.24)+\min(0.25, 0.26) = 0.98$

Mode Collapse Generator: $P_G^{collapse} = 0.7\delta_{x_1} + 0.3\delta_{x_2} + 0\delta_{x_3} + 0\delta_{x_4}$
$QMC_{collapse} = \min(0.25, 0.7) + \min(0.25, 0.3) + \min(0.25, 0) + \min(0.25, 0) = 0.5$

Der drastische Unterschied in QMC zeigt den Mode Collapse deutlich an.

13.8 Fortgeschrittene QGAN-Architekturen

13.8.1 Entangled Quantum GANs

Diese Architektur nutzt Verschränkung als explizites Feature für die Generierung.

Verschränkungs-Generator: $U_{\text{ent}}(\vec{\theta}) = \prod_{i<j} \exp\left(-i\frac{\theta_{ij}}{2} X_i \otimes X_j\right) \prod_k R_y(\phi_k)_k$

Rechnung 64 (3-Qubit Entangled Generator)**.** *Für 3 Qubits mit allen paarweisen Verschränkungen:*

Parameter: $\vec{\theta} = (\theta_{12}, \theta_{13}, \theta_{23}, \phi_1, \phi_2, \phi_3)$

Verschränkungsschicht:

$$U_{ent} = \exp\left(-i\frac{\theta_{23}}{2} X_2 \otimes X_3\right) \exp\left(-i\frac{\theta_{13}}{2} X_1 \otimes X_3\right) \exp\left(-i\frac{\theta_{12}}{2} X_1 \otimes X_2\right) \tag{168}$$

$$\times R_y(\phi_3) \otimes R_y(\phi_2) \otimes R_y(\phi_1) \tag{169}$$

VerschränkungsmaSS: *Für den generierten Zustand* $|\psi_G\rangle$ *berechne die lineare Entropie:* $E_{lin} = 1 - Tr(\rho_A^2)$
wo ρ_A *der reduzierte Dichteoperator eines Subsystems ist.*
Für maximale Verschränkung: $E_{lin} = 1 - \frac{1}{d_A} = \frac{d_A - 1}{d_A}$
Bei 3 Qubits und Subsystem A = Qubit 1: $E_{lin,max} = \frac{1}{2}$

13.8.2 Quantum Conditional GANs

Diese erweitern QGANs um Konditionierung auf klassische Labels.

Conditional Generator: $G_c(z, y, \vec{\theta}) = U_G(\vec{\theta}, y)\,|z\rangle$
wobei die Parameter $\vec{\theta}$ von der Klasse y abhängen.
Conditional Discriminator: $D_c(|\psi\rangle, y, \vec{\phi}) = \langle\psi|\, U_D^\dagger(\vec{\phi}, y) M U_D(\vec{\phi}, y)\,|\psi\rangle$

Rechnung 65 (2-Klassen Conditional QGAN)**.** *Für Klassen* $y \in \{0,1\}$ *und 2-Qubit System:*

Klassenabhängige Parameter:

$$\vec{\theta}(y=0) = (\theta_1^{(0)}, \theta_2^{(0)}) \tag{170}$$

$$\vec{\theta}(y=1) = (\theta_1^{(1)}, \theta_2^{(1)}) \tag{171}$$

Generator-Schaltungen:

$$G_c(z,0) = R_y(\theta_2^{(0)}) \otimes R_y(\theta_1^{(0)}) \left|00\right\rangle \tag{172}$$

$$G_c(z,1) = R_y(\theta_2^{(1)}) \otimes R_y(\theta_1^{(1)}) \left|00\right\rangle \tag{173}$$

Bedingte Verlustfunktion: $L_c = \sum_{y\in\{0,1\}} P(y)\mathbb{E}_{x\sim P(x|y)}[\log D_c(x,y)] + \mathbb{E}_{z\sim p_z}[\log(1 - D_c(G_c(z,y),y))]$

Für optimale Parameter $\vec{\theta}^*(0) = (\pi/4, \pi/6)$ *und* $\vec{\theta}^*(1) = (3\pi/4, 5\pi/6)$*:*

Klasse 0 Zustand: $|\psi_0\rangle = \cos(\pi/8)\cos(\pi/12)\left|00\right\rangle + \cos(\pi/8)\sin(\pi/12)\left|01\right\rangle + \sin(\pi/8)\cos(\pi/12)\left|10\right\rangle + \sin(\pi/8)\sin(\pi/12)\left|11\right\rangle$

Klasse 1 Zustand: $|\psi_1\rangle = \cos(3\pi/8)\cos(5\pi/12)\left|00\right\rangle + \cos(3\pi/8)\sin(5\pi/12)\left|01\right\rangle + \sin(3\pi/8)\cos(5\pi/12)\left|10\right\rangle + \sin(3\pi/8)\sin(5\pi/12)\left|11\right\rangle$

Klassentrennbarkeit: $|\langle\psi_0|\psi_1\rangle|^2 \approx 0.12$ *(niedrige Überlappung)*

13.9 Ausblick und Zukunftsperspektiven

13.9.1 Hybride Klassisch-Quantische Architekturen

Zukünftige QGANs werden wahrscheinlich hybride Ansätze nutzen:

Quantum-Enhanced Classical GAN: - Klassischer Generator mit Quantum-Feature-Preprocessing - Quantum-Discriminator für komplexe Datenstrukturen

Classical-Enhanced Quantum GAN: - Quantum-Generator für Kernerzeugung - Klassischer Post-Processing für hochauflösende Ausgabe

13.9.2 Near-Term Realizable Applications

Finanzmarkt-Simulation: - Portfolio-Optimierung durch Quantensampling - Risikomodellierung mit verschränkten Zuständen

Molekulare Generierung: - Quantenchemische Eigenschaften als native Features - Symmetrie-erhaltende Molekülstrukturen

Optimierungsprobleme: - Kombinatorische Probleme mit exponentiellen Zustandsräumen - Quantenannealing-kompatible Problemformulierungen

Die mathematische Fundierung der QGANs zeigt sowohl das immense Potenzial als auch die erheblichen Herausforderungen dieser Technologie.

Während die theoretischen Vorteile überzeugend sind, erfordern praktische Implementierungen innovative Lösungen für Rauschen, Dekohärenz und Skalierbarkeit. Die Entwicklung robuster QGAN-Architekturen wird ein Schlüssel für den Quantenvorteil in der generativen Modellierung sein.

14 Topologisches Quantencomputing

Das topologische Quantencomputing stellt einen der vielversprechendsten Ansätze zur Realisierung fehlertoleranter Quantencomputer dar. Die mathematischen Grundlagen dieser Technologie wurzeln tief in der algebraischen Topologie, Gruppentheorie und Kategorientheorie. Dieser Abschnitt behandelt die wesentlichen mathematischen Konzepte, die dem topologischen Quantencomputing zugrunde liegen, mit besonderem Fokus auf die gruppentheoretischen und algebraischen Aspekte.

Die zentrale Idee des topologischen Quantencomputings beruht auf der Tatsache, dass bestimmte Quantensysteme topologisch geschützte Zustände besitzen, die inhärent gegen lokale Störungen resistent sind. Diese Eigenschaft wird durch die mathematische Struktur der **Anyonen** und ihrer Austauschstatistik realisiert, die eng mit der Theorie der **Zopfgruppen** (Braid-Gruppen) verknüpft ist.

14.1 Anyonen und Braid-Gruppen

14.1.1 Mathematische Grundlagen der Anyonen

Definition 6 (Anyonen)**.** *Ein Anyon ist ein Quasiteilchen, das in zweidimensionalen Systemen existiert und verallgemeinerte Austauschstatistik zeigt. Beim Austausch zweier identischer Anyonen erhält ihre Wellenfunktion einen Phasenfaktor $e^{i\theta}$, wobei θ beliebige Werte annehmen kann.*

Die mathematische Beschreibung erfolgt durch:

$$\Psi(x_1, x_2) = e^{i\theta}\Psi(x_2, x_1) \tag{174}$$

Klassifikation der Statistiken:

- $\theta = 0$: Bosonische Statistik ($e^{i0} = 1$)
- $\theta = \pi$: Fermionische Statistik ($e^{i\pi} = -1$)
- Andere Werte: Anyonische Statistik

Satz 1 (Spin-Statistik-Theorem in 2D)**.** *In zweidimensionalen Systemen erlaubt das Spin-Statistik-Theorem anyonische Statistiken, während es in drei und mehr Dimensionen auf Bosonen und Fermionen beschränkt ist.*

Beweis. Der Beweis basiert auf der Tatsache, dass in 2D die Fundamentalgruppe des Konfigurationsraums von n unterscheidbaren Punkten die Zopfgruppe B_n ist, während sie in 3D+ die symmetrische Gruppe S_n ist. □

14.1.2 Braid-Gruppen: Algebraische Struktur

Definition 7 (Braid-Gruppe)**.** *Die Braid-Gruppe B_n auf n Strängen ist die Gruppe, deren Elemente Äquivalenzklassen von n-Zöpfen unter Umgebungsisotopie sind, mit der Gruppenoperation als Zopfkomposition.*

Artin-Präsentation: Die Braid-Gruppe B_n wird von elementaren Zöpfen $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$ erzeugt mit den Relationen:

$$\sigma_i \sigma_j = \sigma_j \sigma_i \quad \text{für } |i-j| \geq 2 \quad \text{(entfernte Zöpfe kommutieren)} \tag{175}$$

$$\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \quad \text{(Yang-Baxter-Relation)} \tag{176}$$

Satz 2 (Exakte Sequenz)**.** *Es existiert eine exakte Sequenz:*

$$1 \to P_n \to B_n \to S_n \to 1 \tag{177}$$

wobei P_n die reine Zopfgruppe und S_n die symmetrische Gruppe ist.

14.1.3 Darstellungstheorie der Braid-Gruppen

Definition 8 (Lineare Darstellung)**.** *Eine lineare Darstellung der Braid-Gruppe B_n ist ein Homomorphismus $\rho : B_n \to GL(V)$ für einen endlichdimensionalen Vektorraum V.*

Satz 3 (Bigelow-Krammer-Theorem)**.** *Alle Braid-Gruppen B_n sind linear, d.h., sie besitzen treue Darstellungen in $GL(m, \mathbb{C})$ für geeignetes m.*

Die **Lawrence-Krammer-Darstellung** ist eine treue Darstellung der Dimension $\binom{n}{2}$, die explizit konstruiert werden kann.

14.1.4 Yang-Baxter-Gleichung

Definition 9 (Yang-Baxter-Gleichung)**.** *Die Yang-Baxter-Gleichung in Matrixform lautet:*

$$R_{12} R_{13} R_{23} = R_{23} R_{13} R_{12} \tag{178}$$

Satz 4. *Die Zopfrelation $\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}$ ist äquivalent zur Yang-Baxter-Gleichung.*

Diese Gleichung ist fundamental für die Quantenintegrabilität und stellt die Konsistenz der Flechtoperationen sicher.

14.2 Fibonacci-Anyonen Mathematik

14.2.1 Algebraische Eigenschaften

Definition 10 (Fibonacci-Anyonen)**.** *Fibonacci-Anyonen sind nicht-abelsche Anyonen, die durch die einzigartige Fusionsregel charakterisiert sind:*

$$\tau \otimes \tau = \mathbf{1} \oplus \tau \tag{179}$$

wobei τ das Fibonacci-Anyon und $\mathbf{1}$ das Vakuum bezeichnet.

Satz 5 (Quantendimension)**.** *Die Quantendimension des Fibonacci-Anyons ist:*

$$d_\tau = \varphi = \frac{1+\sqrt{5}}{2} \approx 1.618 \tag{180}$$

(der goldene Schnitt).

Beweis. Aus der Fusionsregel folgt $d_\tau^2 = d_{\mathbf{1}} + d_\tau = 1 + d_\tau$, was zu $d_\tau^2 - d_\tau - 1 = 0$ führt. Die positive Lösung ist $d_\tau = \varphi$. □

14.2.2 F-Matrizen und Pentagon-Gleichungen

Definition 11 (F-Matrix)**.** *Die F-Matrix beschreibt die Transformation zwischen verschiedenen Assoziativitätsbasen in der Fusionsalgebra.*

Für Fibonacci-Anyonen ist die nicht-triviale F-Matrix:

$$F^{\tau}_{\tau,\tau,\tau} = \begin{pmatrix} \varphi^{-1/2} & \varphi^{-1/2} \\ \varphi^{-1/2} & -\varphi^{-1} \end{pmatrix} \tag{181}$$

Satz 6 (Pentagon-Gleichung)**.** *Die Pentagon-Gleichung für Fibonacci-Anyonen vereinfacht sich zu:*

$$(F^{\tau}_{\tau,\tau,\tau})^2 = \varphi^{-1} \cdot I \tag{182}$$

14.2.3 R-Matrizen und Hexagon-Gleichungen

Definition 12 (R-Matrix)**.** *Die R-Matrix-Elemente für Fibonacci-Anyonen sind:*

$$R_{\mathbf{1}}^{\mathbf{1},\mathbf{1}} = R_{\tau}^{\mathbf{1},\tau} = R_{\tau}^{\tau,\mathbf{1}} = 1 \tag{183}$$

$$R_{\mathbf{1}}^{\tau,\tau} = e^{4\pi i/5} \tag{184}$$

$$R_{\tau}^{\tau,\tau} = e^{-3\pi i/5} \tag{185}$$

Satz 7 (Flechtmatrix)**.** *Die kombinierte Flechtmatrix $B = F^{-1} \cdot R \cdot F$ hat die explizite Form:*

$$B = \begin{pmatrix} e^{-4\pi i/5} & 0 \\ 0 & e^{3\pi i/5} \end{pmatrix} \tag{186}$$

14.2.4 Universalität für Quantencomputing

Satz 8 (Flechtuniversalität)**.** *Die durch Fibonacci-Anyonen erzeugten Zopfgruppen-Darstellungen sind dicht in $SU(2)$, was universelles Quantencomputing durch Flechten allein ermöglicht.*

14.3 Topologische Codes

14.3.1 Toric Code

Definition 13 (Toric Code)**.** *Der Toric Code ist auf einem 2D-Quadratgitter mit periodischen Randbedingungen (Torus-Topologie) definiert. Qubits sind auf den Vertices platziert, und Check-Operatoren sind auf Plaquetten in Schachbrettmuster definiert.*

Stabilisator-Operatoren:

$$\text{X-Typ:} \quad \hat{S}_X^i = \bigotimes_{\ell \in \partial i} \hat{X}_\ell \tag{187}$$

$$\text{Z-Typ:} \quad \hat{S}_Z^i = \bigotimes_{\ell \in \partial i} \hat{Z}_\ell \tag{188}$$

Satz 9 (Code-Parameter)**.** *Für ein $L \times L$-Gitter kodiert der Toric Code 2 logische Qubits mit Distanz L, notiert als $[[L^2, 2, L]]$.*

14.3.2 Homologische Beschreibung

Definition 14 (Homologische Struktur)**.** *Der Toric Code kann homologisch beschrieben werden:*

- *Fehlerketten entsprechen 1-Ketten auf dem Gitter*
- *Syndrome entsprechen dem Randoperator ∂E*
- *Logische Operatoren entsprechen nicht-trivialen Homologieklassen*

Satz 10 (Fehlerkorrektur)**.** *Die Fehlerkorrektur ist erfolgreich, wenn sich die Wiederherstellungsoperation vom tatsächlichen Fehler nur um triviale Zyklen unterscheidet.*

14.3.3 Color Codes

Definition 15 (Color Code)**.** *Color Codes sind auf trivalenten Gittern mit drei-färbbaren Flächen definiert. Zwei Hauptimplementierungen sind:*

- *Wabengitter (reguläre Sechsecke)*
- *Quadrat-oktagonales Gitter (erweiterte Rechenfähigkeit)*

Stabilisator-Struktur: Jede Fläche hat sowohl X- als auch Z-Typ-Stabilisatoren:

$$\hat{S}_X^f = \bigotimes_{v \in \partial f} \hat{X}_v \tag{189}$$

$$\hat{S}_Z^f = \bigotimes_{v \in \partial f} \hat{Z}_v \tag{190}$$

Satz 11 (Transversale Gatter)**.** *Die drei-färbbare Struktur ermöglicht transversale Gatter, wobei die quadrat-oktagonale Version eine transversale Implementierung der gesamten Clifford-Gruppe erlaubt.*

14.4 Modulare Tensor-Kategorien

14.4.1 Grundlegende Definitionen

Definition 16 (Modulare Tensor-Kategorie)**.** *Eine modulare Tensor-Kategorie ist eine rigide, halbeinfache, geflochtene Fusionskategorie mit nicht-degenerierter Flechtung.*

Äquivalente Definitionen:

1. **Bruguières-Definition:** Eine geflochtene sphärische Fusionskategorie mit nicht-degenerierter Flechtung
2. **S-Matrix-Definition:** Eine geflochtene sphärische Fusionskategorie mit nicht-degenerierter (invertierbarer) S-Matrix

14.4.2 Strukturelle Eigenschaften

Satz 12 (Endlicher Rang)**.** *Jede modulare Tensor-Kategorie hat endlichen Rang (endlich viele Isomorphieklassen einfacher Objekte).*

Definition 17 (Globale Dimension)**.** *Die globale Dimension ist*

$$D^2 = \sum_i d_i^2 \tag{191}$$

wobei d_i die Quantendimensionen der einfachen Objekte sind.

Satz 13 (Fusionsregeln)**.** *Einfache Objekte erfüllen Fusionsregeln:*

$$X_i \otimes X_j = \sum_k N_{ij}^k X_k \tag{192}$$

wobei $N_{ij}^k \in \mathbb{N}$ die Fusionsmultiplizitäten sind.

14.4.3 Modulare Daten

Definition 18 (Modulare Daten)**.** *Jede modulare Tensor-Kategorie ist mit modularen Daten (S, T, c) ausgestattet:*

- ***S-Matrix:*** *S_{ij} kodiert Flechtphasen*
- ***T-Matrix:*** *$T_{ij} = \delta_{ij}\theta_i$ kodiert Twist-Phasen*
- ***Zentrale Ladung:*** *c bestimmt die konforme Anomalie*

Satz 14 (Verlinde-Formel)**.** *Für rationale konforme Feldtheorien sind die Fusionskoeffizienten gegeben durch:*

$$N_{ij}^k = \sum_\ell \frac{S_{i\ell} S_{j\ell} S_{k\ell}^*}{S_{0\ell}} \tag{193}$$

14.4.4 Verbindung zu topologischen Quantenfeldtheorien

Satz 15 (Reshetikhin-Turaev-Konstruktion)**.** *Jede modulare Tensor-Kategorie gibt Anlass zu einer (2+1)-dimensionalen topologischen Quantenfeldtheorie.*

Satz 16 (Cobordismus-Hypothese)**.** *Fusionskategorien sind vollständig dualisierbare Objekte in der $(\infty, 3)$-Kategorie der monoidalen Kategorien.*

14.4.5 Klassifikationsergebnisse

Satz 17 (Rowell-Stong-Wang-Klassifikation)**.** *Alle unitären modularen Tensor-Kategorien vom Rang ≤ 4 wurden klassifiziert, was insgesamt 70 UMTCs ergibt (35 bis auf $\pm S$-Äquivalenz).*

Satz 18 (Primzerlegung)**.** *Jede modulare Tensor-Kategorie besitzt eine eindeutige Primzerlegung, analog zur Primfaktorzerlegung von Ganzzahlen.*

14.5 Kategorielle Beschreibung von Anyonen

14.5.1 Anyonen als Objekte in Kategorien

Definition 19 (Anyonen-Modell)**.** *Anyonen werden als einfache Objekte in geflochtenen unitären Fusionskategorien modelliert:*

- ***Objekte:*** *Anyonen-Typen*
- ***Morphismen:*** *Intertwiners*
- ***Tensorprodukt:*** *Fusionsoperation*
- ***Flechtung:*** *Austauschstatistik*

Satz 19 (Zopfgruppen-Darstellungen)**.** *Die Flechtung in einer modularen Tensor-Kategorie liefert Darstellungen der Zopfgruppen B_n auf n-fachen Tensorprodukten von Objekten.*

14.5.2 Quantencomputing-Gatter

Definition 20 (Topologische Gatter)**.** *Flechtoperationen und Messungen (Fusion) liefern einen universellen Gattersatz für Quantencomputing.*

Satz 20 (Universalität)**.** *Eine unitäre modulare Tensor-Kategorie ermöglicht universelles Quantencomputing genau dann, wenn ihre globale Quantendimension D^2 keine ganze Zahl ist.*

14.6 Mathematische Beweise und Konstruktionen

14.6.1 Mügers Theorem

Satz 21 (Mügers Theorem)**.** *Das Drinfeld-Zentrum $Z(\mathcal{C})$ einer sphärischen Fusionskategorie $\mathcal{C}$ ist modular.*

Beweis-Skizze. Der Beweis verwendet die Konstruktion des Drinfeld-Zentrums als Kategorie der Halb-Zöpfe und zeigt, dass die natürliche Flechtung nicht-degeneriert ist. □

14.6.2 Quantengruppen-Konstruktion

Konstruktion 1 (Quantengruppen-Methode)**.** *Für einfache Lie-Algebra* $\mathfrak{g}$ *und positive ganze Zahl* k *konstruiert man* $\mathcal{C}(\mathfrak{g}, k)$ *über Darstellungstheorie von* $U_q(\mathfrak{g})$*.*

Satz 22 (Verlinde-Algebra)**.** *Die Fusionsregeln entsprechen der Verlinde-Algebra der entsprechenden Chern-Simons-Theorie.*

14.7 Experimentelle Realisierungen und aktuelle Entwicklungen

14.7.1 Microsofts Majorana-Ansatz

Microsoft hat 2025 den weltweit ersten Quantenprozessor mit topologischen Qubits vorgestellt, basierend auf Majorana-Nullmoden in Indiumarsenid-Aluminium-Heterostrukturen.

Mathematische Beschreibung: Majorana-Fermionen erfüllen die Relation $\gamma^\dagger = \gamma$ und antikommutieren:

$$\{\gamma_i, \gamma_j\} = 2\delta_{ij} \tag{194}$$

14.7.2 Quantinuums nicht-abelsche Anyonen

Quantinuum demonstrierte 2024 den ersten ëchten topologischen Qubit"mit nicht-abelschen Anyonen unter Verwendung des $\mathbb{Z}_3$-Toric-Codes.

Mathematische Grundlage: Implementation der $\mathbb{Z}_3$-Parafermion-Statistik mit Flechtungsoperatoren, die nicht-triviale Transformationen der Grundzustandsmannigfaltigkeit induzieren.

14.8 Fazit

Das topologische Quantencomputing verbindet tiefgreifende mathematische Theorien aus der algebraischen Topologie, Gruppentheorie und Kategorientheorie mit praktischen Anwendungen in der Quanteninformationsverarbeitung. Die mathematische Eleganz der Fibonacci-Anyonen, bei denen der goldene Schnitt als fundamentale Quantendimension erscheint, die Pentagon- und Hexagon-Gleichungen als Konsistenzbedingungen und die Verbindung zu modularen Tensor-Kategorien als theoretische Grundlage für universelles topologisches Quantencomputing zeigen die Schönheit und Komplexität dieses Forschungsgebiets.

Die jüngsten experimentellen Fortschritte deuten darauf hin, dass die theoretischen Vorhersagen der mathematischen Theorie in naher Zukunft

praktisch realisiert werden könnten. Die inhärente Fehlertoleranz topologischer Qubits, die durch die mathematische Struktur der zugrundeliegenden Kategorien gewährleistet wird, macht diesen Ansatz besonders vielversprechend für die Entwicklung skalierbarer, fehlertoleranter Quantencomputer.

15 Continuous Variable Quantum Computing

Diese Sektion behandelt die mathematischen Grundlagen und aktuellen Entwicklungen im Bereich des kontinuierlichen Quantencomputings (Continuous Variable Quantum Computing, CVQC). Im Gegensatz zu diskreten Qubit-Systemen operiert CVQC auf unendlich-dimensionalen Hilbert-Räumen und nutzt die kontinuierliche Natur von Quadratur-Operatoren elektromagnetischer Felder.

15.1 Quadratur-Operatoren und fundamentale mathematische Strukturen

Die mathematischen Grundlagen des kontinuierlichen Quantencomputings basieren auf den kanonischen Kommutatorrelationen zwischen Quadratur-Operatoren. Für jede Mode des elektromagnetischen Feldes definieren wir das kanonisch konjugierte Paar:

$$\hat{q} = \frac{1}{\sqrt{2}}(\hat{a} + \hat{a}^\dagger) \quad \text{(Positionsquadratur)} \tag{195}$$

$$\hat{p} = \frac{i}{\sqrt{2}}(\hat{a}^\dagger - \hat{a}) \quad \text{(Impulsquadratur)} \tag{196}$$

Diese Operatoren erfüllen die fundamentale kanonische Kommutatorrelation:

$$[\hat{q}, \hat{p}] = i\hbar \tag{197}$$

15.1.1 Heisenberg-Unschärferelation

Die Unschärferelation für kontinuierliche Variablen besagt, dass für jeden Quantenzustand gilt:

$$\Delta q \cdot \Delta p \geq \frac{\hbar}{2} \tag{198}$$

Diese Relation stellt die fundamentale Grenze für die gleichzeitige Präzision konjugierter Messungen dar und ist zentral für das Verständnis von Quetschzuständen und deren Anwendungen in der Quantenfehlerkorrektur.

15.1.2 Weyl-Heisenberg-Gruppen

Die Weyl-Heisenberg-Gruppe liefert den mathematischen Rahmen für Displacement-Operationen im Phasenraum. Die Gruppenelemente werden durch komplexe Zahlen α parametrisiert und durch Displacement-Operatoren dargestellt:

$$\hat{D}(\alpha) = \exp(\alpha\hat{a}^\dagger - \alpha^*\hat{a}) \tag{199}$$

Diese Operatoren erfüllen die Weyl-Relationen:

$$\hat{D}(\alpha)\hat{D}(\beta) = \hat{D}(\alpha+\beta)\exp\left(\frac{i}{2}\mathrm{Im}(\alpha\beta^*)\right) \tag{200}$$

Das Stone-von Neumann-Theorem garantiert die Eindeutigkeit irreduzibler Darstellungen der Weyl-Kommutatorrelationen und etabliert die fundamentale Verbindung zwischen klassischer und Quantenmechanik.

15.1.3 Fock-Raum-Darstellung

Der Fock-Raum bietet einen natürlichen unendlich-dimensionalen Hilbert-Raum für CV-Systeme. Die Schlüsselelemente sind:

$$\hat{a}^\dagger|n\rangle = \sqrt{n+1}|n+1\rangle \tag{201}$$

$$\hat{a}|n\rangle = \sqrt{n}|n-1\rangle \tag{202}$$

$$|n\rangle = \frac{(\hat{a}^\dagger)^n}{\sqrt{n!}}|0\rangle \tag{203}$$

Kohärente Zustände als Eigenzustände des Vernichtungsoperators:

$$|\alpha\rangle = e^{-|\alpha|^2/2}\sum_{n=0}^{\infty}\frac{\alpha^n}{\sqrt{n!}}|n\rangle \tag{204}$$

mit $\hat{a}|\alpha\rangle = \alpha|\alpha\rangle$.

15.2 GauSSsche Zustände und Operationen

15.2.1 Kovarianzmatrizen

GauSSsche Zustände sind vollständig durch ihre ersten und zweiten Momente charakterisiert. Für ein n-Moden-System mit Quadratur-Vektor $\hat{\mathbf{r}} = (\hat{q}_1, \hat{p}_1, \ldots, \hat{q}_n, \hat{p}_n)^T$ ist die Kovarianzmatrix definiert als:

$$\sigma_{ij} = \frac{1}{2}\langle\{\hat{r}_i, \hat{r}_j\}\rangle - \langle\hat{r}_i\rangle\langle\hat{r}_j\rangle \tag{205}$$

Die Kovarianzmatrix muss die Unschärferelation erfüllen:

$$\sigma + \frac{i\hbar}{2}\Omega \geq 0 \tag{206}$$

wo $\Omega = \bigoplus_{i=1}^{n} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ die symplektische Form ist.

15.2.2 Williamson-Normalform

Das Williamson-Theorem besagt, dass jede positive definite reelle symmetrische Matrix σ durch eine symplektische Transformation diagonalisiert werden kann. Es existiert eine symplektische Matrix S mit:

$$S\sigma S^T = \text{diag}(\nu_1, \nu_1, \nu_2, \nu_2, \ldots, \nu_n, \nu_n) \tag{207}$$

wobei $\nu_i \geq 1/2$ die symplektischen Eigenwerte sind. Diese Parameter haben physikalische Bedeutung:

- $\nu_i = 1/2$ entspricht dem Vakuumzustand
- $\nu_i > 1/2$ indiziert thermische Besetzung mit mittlerer Photonenzahl $\langle n_i \rangle = \nu_i - 1/2$

15.2.3 Symplektische Transformationen

Die symplektische Gruppe $\text{Sp}(2n, \mathbb{R})$ besteht aus $2n \times 2n$ reellen Matrizen S, die erfüllen:

$$S\Omega S^T = \Omega \tag{208}$$

Bloch-Messiah-Zerlegung: Jede symplektische Matrix kann zerlegt werden als:

$$S = O_1 Z O_2 \tag{209}$$

wobei O_1, O_2 orthogonale symplektische Matrizen (passive Transformationen) und Z eine Diagonalmatrix von Ein-Moden-Squeezern (aktive Transformationen) ist.

15.2.4 GauSSsche Unitäre Operatoren

GauSSsche unitäre Operatoren erhalten den gauSSschen Charakter von Zuständen. Sie entsprechen Hamiltonoperatoren, die höchstens quadratisch in den Quadratur-Operatoren sind:

$$H = \frac{1}{2}\hat{\mathbf{r}}^T A \hat{\mathbf{r}} + \hat{\mathbf{r}}^T \mathbf{b} + c \tag{210}$$

Die vollständige Menge gauSSscher unitärer Operatoren umfasst:

$$\hat{D}(\alpha) = \exp(\alpha\hat{a}^\dagger - \alpha^*\hat{a}) \quad \text{(Displacement)} \tag{211}$$

$$\hat{S}(r) = \exp\left(\frac{r}{2}(\hat{a}^2 - \hat{a}^{\dagger 2})\right) \quad \text{(Squeezing)} \tag{212}$$

$$\hat{R}(\theta) = \exp(-i\theta\hat{a}^\dagger\hat{a}) \quad \text{(Rotation)} \tag{213}$$

$$\hat{BS}(\theta) = \exp(\theta(\hat{a}\hat{b}^\dagger - \hat{a}^\dagger\hat{b})) \quad \text{(Beam Splitter)} \tag{214}$$

15.3 Quantenfehlerkorrektur für kontinuierliche Variablen

15.3.1 Bosonische Codes

Bosonische Codes kodieren Quanteninformation in den unendlich-dimensionalen Hilbert-Raum von Quantenharmonischen Oszillatoren. Die wichtigsten Codefamilien sind:

Cat-Codes: Kodewörter sind Superpositionen kohärenter Zustände:

$$|\overline{0}\rangle = \frac{1}{\sqrt{2}}(|\alpha\rangle + |-\alpha\rangle) \tag{215}$$

$$|\overline{1}\rangle = \frac{1}{\sqrt{2}}(|\alpha\rangle - |-\alpha\rangle) \tag{216}$$

Die Kodedistanz wächst exponentiell mit der mittleren Photonenzahl $|\alpha|^2$.

Binomial-Codes: Endliche Superposition von Fock-Zuständen mit Binomialkoeffizienten:

$$|\overline{0}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |4\rangle) \tag{217}$$

$$|\overline{1}\rangle = |2\rangle \tag{218}$$

Diese Codes ermöglichen exakte Korrektur von Polynomgrad-Fehlern in Erzeugungs-/Vernichtungsoperatoren.

15.3.2 Stabilizer-Codes für CV-Systeme

Die CV-Stabilizer-Formalisierung erweitert Gottesmans diskrete Stabilizer-Codes auf unendliche Dimensionen. Die kontinuierliche Pauli-Gruppe besteht aus:

$$\exp(i\alpha\hat{q} + i\beta\hat{p}) \tag{219}$$

für reelle α, β. Die Konstruktion erfolgt durch:

1. Wähle k Vektoren $u_1, \ldots, u_k \in \mathbb{R}^{2n}$ mit $\omega(u_i, u_j) = 0$
2. Definiere Stabilizer-Generatoren als Displacement-Operatoren
3. Berechne logische Operatoren aus dem symplektischen orthogonalen Komplement

15.3.3 Aktuelle Forschung und Entwicklungen

Concatenated Bosonic Codes (2024): AWS Center for Quantum Computing demonstrierte erfolgreich concatenated Cat-Qubits mit Repetition-Codes:

- Architektur: Cat-Qubits (innerer Code) + Repetition-Code (äuSSerer Code)
- Performance: Distanz-5 Code mit 1.65(3)% logischer Fehlerrate
- Unterhalb der Schwelle: Erste Below-Threshold-Operation

Rotations-symmetrische Codes: Neue Rahmenwerke für bosonische Codes basierend auf Phasenraum-Rotations-Symmetrie, einschlieSSlich:

- Zahlen-Phasen-Codes als Verallgemeinerung von Cat- und Binomial-Codes
- Universelle Quantencomputing-Schemata für rotations-symmetrische Codes
- Optimale Fehlerkorrektur für spezifische Symmetriegruppen

15.4 Gottesman-Kitaev-Preskill (GKP) Codes

15.4.1 Mathematische Konstruktion

GKP-Codes sind **Quantengittercodes**, die logische Qubits in den unendlichdimensionalen Hilbert-Raum harmonischer Oszillatoren kodieren. Die mathematische Grundlage beruht auf:

Stabilizer-Generatoren: Für Ein-Moden-Quadratgitter-GKP-Codes:

$$\hat{S}_q(2\alpha) = e^{-2i\alpha\hat{p}} \tag{220}$$

$$\hat{S}_p(2\beta) = e^{2i\beta\hat{x}} \tag{221}$$

Abelsche Bedingung: Für einen abelschen Stabilizer gilt: $\alpha\beta = 2q\pi$ mit ganzzahligem q.

Kodewort-Struktur: GKP-Kodewörter sind gleichgewichtete Superpositionen kohärenter Zustände auf einem rechteckigen Gitter im Phasenraum:

$$|\text{GKP}\rangle = \sum_{n\in\mathbb{Z}} |x = n\sqrt{2\pi}\rangle \tag{222}$$

15.4.2 Gittertheoretische Grundlagen

Gitter-Konstruktion: Jeder GKP-Code kann aus Gittern mit $2n$-dimensionalen Vollrang-Gram-Matrizen A konstruiert werden, wobei n die Anzahl der Moden ist.

Duale Gitter-Relationen: Der Zentralisator für die Stabilizer-Gruppe wird mit dem symplektischen dualen Gitter $L^{\perp}$ identifiziert: logische Operationen entsprechen dualen Quotienten $L^{\perp}/L$.

Distanz-Eigenschaften: Die Distanz von GKP-Codes wird durch die kürzesten Nicht-Null-Vektoren im dualen Gitter bestimmt, wobei Pauli-Distanzen durch Normen der Displacement-Vektoren gegeben sind.

15.4.3 Algebraische Geometrie und erweiterte Verbindungen

Algebraische Geometrie: Conrad et al. (2024) etablierten tiefgreifende Verbindungen zwischen GKP-Codes und algebraischer Geometrie:

- GKP-Clifford-Gates entstehen als symplektische Automorphismen des entsprechenden Gitters
- Ein-Moden-GKP-Codes entsprechen dem Modulraum elliptischer Kurven
- Der Raum aller GKP-Codes wird mit einer Dreikugel mit entferntem Kleeblattknoten identifiziert

Kryptographische Verbindungen: NTRU-GKP-Codes (2024) demonstrieren, dass GKP-Codes Quantenfehlerkorrektur und Post-Quanten-Kryptographie verbinden, wobei die Dekodierung äquivalent zur Entschlüsselung des NTRU-Kryptosystems ist.

15.4.4 Fourier-Transformation und Displacement-Operatoren

Displacement-Operator-Formalismus:

$$\hat{D}(\alpha) = \exp(\alpha\hat{a}^{\dagger} - \alpha^{*}\hat{a}) \tag{223}$$

Kommutatorrelationen:

$$\hat{D}(\alpha)\hat{D}(\beta) = \exp\left(\frac{i}{2}(\alpha\beta^* - \alpha^*\beta)\right)\hat{D}(\alpha+\beta) \tag{224}$$

Fehlerkorrektur: GKP-Codes detektieren und korrigieren kleine Displacement-Fehler durch:

- Messung der Stabilizer-Generatoren
- Abbildung kontinuierlicher Fehler auf diskrete Syndrom-Verschiebungen
- Korrektur von Displacements bis zu $\alpha/2$ in Position und $\beta/2$ in Impuls

Gitter-Struktur im Phasenraum: GKP-Zustände bilden periodische Gitter im Phasenraum, wobei die zugrundeliegende Gitter-Struktur die Fehlerkorrektur-Fähigkeiten bestimmt.

15.4.5 Aktuelle experimentelle Implementierungen

Supraleiter-Schaltkreise: Mehrere Experimente erreichten Quantenfehlerkorrektur jenseits des Break-Even-Punktes:

- Sivak et al. (Nature 2023): Autonome Fehlerkorrektur mit erheblicher Lebensdauer-Verbesserung
- Lachance-Quirion et al. (2023): Autonome Stabilisierung von GKP-Zuständen
- 2024 APS March Meeting: Implementierung von Zweischicht-Fehlerkorrektur

Trapped-Ion-Systeme:

- Matsos et al. (2024): Robuste Präparation bosonischer logischer Zustände
- Implementierung universeller Gate-Sätze für GKP-kodierte Qubits
- Gate-Genauigkeiten geeignet für fehlertolerante Operationen

Optische und photonische Systeme:

- Konno et al. (Science 2024): Logische Zustände für fehlertolerante Quantencomputation mit propagierendem Licht
- Larsen et al. (Nature 2025): Integrierte photonische Quelle von GKP-Qubits
- Descamps et al. (2024): GKP-Kodierung in kontinuierlichen modalen Variablen einzelner Photonen

15.4.6 Neueste Fortschritte und Schwellenwerte

Squeezing-Anforderungen:

- Minimale Schwelle: ∼9.9 dB für Surface-GKP-Concatenation
- Praktischer Betrieb: 12+ dB Squeezing für niedrige logische Fehlerrate
- Experimentelle Erreichung: 10 dB in ultrakalten Atomen demonstriert

Neural Network-basierte Ansätze:

- Zeng et al. (PRL 2025): NN-basiertes Design approximativer GKP-Codes
- 50% Verbesserung der Dekodierungsraten
- Schwellenwert-Verbesserungen von $\sigma \approx 0.5$ auf $\sigma \approx 0.78$

15.5 Mathematische Unterschiede zu Qubit-Systemen

15.5.1 Hilbert-Raum-Struktur

Discrete Variable (DV) Systeme:

- Endlich-dimensionale Hilbert-Räume: $\mathbb{C}^{2^n}$ für n Qubits
- Diskrete Basis-Zustände: $\{|0\rangle, |1\rangle\}^{\otimes n}$
- Fundamentale Symmetriegruppe: SU(2) für Ein-Qubit-Rotationen

Continuous Variable (CV) Systeme:

- Unendlich-dimensionale Hilbert-Räume: $L^2(\mathbb{R})$ für Position
- Kontinuierliche Wellenfunktionen: $\psi(x) = \langle x|\psi\rangle$
- Fundamentale Symmetriegruppe: $\mathrm{Sp}(2n, \mathbb{R})$ - symplektische Gruppe

15.5.2 Symmetrien und Gruppenstrukturen

DV-Systeme - SU(2)-Struktur:

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{225}$$

$$[\sigma_i, \sigma_j] = 2i\epsilon_{ijk}\sigma_k \tag{226}$$

CV-Systeme - Symplektische Struktur:

$$[\hat{x}_i, \hat{p}_j] = i\hbar\delta_{ij} \tag{227}$$

$$M\Omega M^T = \Omega \quad \text{für } M \in \mathrm{Sp}(2n, \mathbb{R}) \tag{228}$$

15.5.3 Fehlermodelle

DV-Fehlermodelle:

$$\varepsilon_{\text{depol}}(\rho) = (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z) \tag{229}$$

$$\varepsilon_{\text{damp}}(\rho) = \sum_{k=0}^{1} E_k \rho E_k^\dagger \tag{230}$$

CV-Fehlermodelle:

$$\varepsilon_{\text{loss}}(\rho) = \sum_{n=0}^{\infty} \sqrt{\eta^n(1-\eta)^m} L^n \rho (L^\dagger)^n \tag{231}$$

$$\sigma \to A\sigma A^T + N \quad \text{(Kovarianzmatrix-Evolution)} \tag{232}$$

15.5.4 Quantenfehlerkorrektur-Unterschiede

DV-Quantenfehlerkorrektur:

- Diskrete Syndrom-Detektion: $s \in \{0,1\}^k$
- Stabilizer-Codes: $[\![n, k, d]\!]$ mit diskreter Fehlerkorrektur
- Schwellenwert-Theorem: Fehlertolerante Berechnung über Schwellenwert möglich

CV-Quantenfehlerkorrektur:

- Kontinuierliche Fehler-Syndrome: Erfordern analoge Verarbeitung
- GKP-Codes: Kodieren Qubits in Oszillatoren
- Squeezing-Anforderungen: $\sim$12-15 dB für Fehlertoleranz
- Analoge Quantenfehlerkorrektur: Kontinuierliche Fehlerinformation

15.5.5 Komplexitätstheoretische Unterschiede

DV-Komplexität:

- Berechnungsbasis: $\{|0\rangle^{\otimes n}, |1\rangle^{\otimes n}, \ldots\}$ - 2^n Basis-Zustände
- Gate-Komplexität: Polynomial in Anzahl der Qubits für universelle Gate-Sätze

- Gottesman-Knill-Theorem: Clifford-Schaltkreise klassisch effizient simulierbar

CV-Komplexität:

- Berechnungsbasis: Kontinuierlich in Phasenraum-Koordinaten
- GauSSsche Operationen: Klassisch effizient simulierbar (positive Wigner-Funktionen)
- Nicht-gauSSsche Ressourcen: Erforderlich für Quantenvorteil
- Unendlich-dimensionale Räume: Potentiell exponentieller Vorteil

15.5.6 Lie-Algebren und kontinuierliche Symmetrien

Heisenberg-Weyl-Algebra:

$$[\hat{q}, \hat{p}] = i\hbar \tag{233}$$
$$[\hat{q}, \hat{I}] = 0 \tag{234}$$
$$[\hat{p}, \hat{I}] = 0 \tag{235}$$

SU(1,1)-Algebra (für Squeezing-Operationen):

$$[\hat{K}_0, \hat{K}_+] = \hat{K}_+ \tag{236}$$
$$[\hat{K}_0, \hat{K}_-] = -\hat{K}_- \tag{237}$$
$$[\hat{K}_+, \hat{K}_-] = -2\hat{K}_0 \tag{238}$$

wobei $\hat{K}_0 = \frac{1}{4}(\hat{a}^\dagger\hat{a} + \hat{a}\hat{a}^\dagger)$, $\hat{K}_+ = \frac{1}{2}\hat{a}^\dagger\hat{a}^\dagger$, $\hat{K}_- = \frac{1}{2}\hat{a}\hat{a}$.

Diese umfassende mathematische Analyse zeigt, dass CV- und DV-Quantencomputing fundamental verschiedene mathematische Rahmenwerke darstellen - eines diskret und endlich, das andere kontinuierlich und unendlich-dimensional, jeweils mit einzigartigen Vorteilen und Herausforderungen für die Quanteninformationsverarbeitung. Die kontinuierliche Natur der CV-Systeme ermöglicht durch die elegante Sprache der symplektischen Geometrie und Lie-Gruppentheorie neue Ansätze zur Quantenfehlerkorrektur und zum Quantencomputing.

16 Quantensimulation

Die Simulation von Quantensystemen ist eine der vielversprechendsten Anwendungen des Quantencomputings, da klassische Computer exponentiell mit der SystemgröSSe skalieren.

16.1 Zeitentwicklung quantenmechanischer Systeme

Die Zeitentwicklung wird durch die Schrödinger-Gleichung bestimmt:

$$i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle = H\left|\psi(t)\right\rangle$$

Die formale Lösung ist:

$$\left|\psi(t)\right\rangle = e^{-iHt/\hbar}\left|\psi(0)\right\rangle$$

16.2 Trotter-Suzuki-Zerlegung

Für komplexe Hamiltonians $H = \sum_i H_i$ nutzt man die Trotter-Formel:

$$e^{-i(H_1+H_2)t} \approx \left(e^{-iH_1t/n}e^{-iH_2t/n}\right)^n$$

Rechnung 66 (Trotter-Zerlegung für Ising-Modell)**.** *Das transversale Ising-Modell:*

$$H = -J\sum_{\langle i,j\rangle} Z_iZ_j - h\sum_i X_i$$

Trotter-Zerlegung (n Zeitschritte):

$$e^{-iHt} \approx \left(\prod_{\langle i,j\rangle} e^{iJZ_iZ_jt/n}\prod_i e^{ihX_it/n}\right)^n$$

Jeder Term kann direkt als Quantengatter implementiert werden: - $e^{ihX_it/n} = R_x(2ht/n)$ - $e^{iJZ_iZ_jt/n} = CNOT_{ij}\cdot R_z(2Jt/n)\cdot CNOT_{ij}$

17 Quantenkryptographie - Erweiterte Protokolle

Die moderne Quantenkryptographie hat sich weit über das grundlegende BB84-Protokoll hinaus entwickelt und eine Vielzahl spezialisierter Protokolle hervorgebracht, die jeweils einzigartige Sicherheitsgarantien und Anwendungsmöglichkeiten bieten. Diese sechs fortgeschrittenen Protokolle repräsentieren die neuesten Entwicklungen in der quantenkryptographischen Forschung und bieten unterschiedliche Ansätze zur Gewährleistung informationstheoretischer Sicherheit.

17.1 Das E91-Protokoll: Entanglement-basierte Quantenschlüsselverteilung

17.1.1 Technische Grundlagen und Quantenmechanik

Das 1991 von Artur Ekert vorgeschlagene E91-Protokoll nutzt **maximal verschränkte Photonenzustände** für die Schlüsselverteilung und basiert auf der Verletzung der Bell'schen Ungleichung als Sicherheitsgarantie. Das Protokoll verwendet Photonenpaare im Polarisations-Singlet-Zustand:

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A|1\rangle_B - |1\rangle_A|0\rangle_B) \tag{239}$$

Der Protokollablauf umfasst vier Hauptphasen: **Verschränkte Paarerzeugung**, bei der eine Quelle Photonenpaare im Singlet-Zustand generiert; **Messprozess**, bei dem Alice und Bob jeweils drei Messbasen verwenden (z.B. 0ř, 45ř, 90ř); **Öffentliche Kommunikation** zur Klassifikation der Messungen; und **Schlüsselgenerierung** aus korrelierten Messungen.

Das charakteristische Merkmal des E91-Protokolls ist die **Verwendung der CHSH-Ungleichung** für die Sicherheitsverifikation:

$$S = E(a, b) + E(a, b') + E(a', b) - E(a', b') \leq 2 \tag{240}$$

Für maximal verschränkte Zustände sagt die Quantenmechanik $S = 2\sqrt{2} \approx 2.828$ voraus, dessen Verletzung echte Verschränkung und die Abwesenheit von Lauschangriffen bestätigt.

17.1.2 Vorteile und Nachteile gegenüber BB84

Wesentliche Vorteile umfassen:

- **Geräteunabhängigkeit** durch Bell-Tests, die Sicherheit auch bei nicht vertrauenswürdigen Messgeräten gewährleisten
- **Quellenunabhängigkeit**, da selbst eine von Eve kontrollierte Verschränkungsquelle die Sicherheit nicht kompromittieren kann
- **Netzwerkkompatibilität** für Quantennetzwerke mit mehreren Knoten
- **Fundamentale Sicherheit** basierend auf den Grundgesetzen der Quantenmechanik

Hauptnachteile sind:

- **Implementierungskomplexität** durch die Notwendigkeit, verschränkte Photonenpaare zu erzeugen und zu erhalten
- **Niedrigere Schlüsselraten** aufgrund der Koinzidenzdetektions-Anforderungen
- **Technologische Anforderungen** wie hochwertige Verschränkungsquellen und präzise Zeitsynchronisation

17.1.3 Sicherheitsgarantien und aktuelle Forschung

Das E91-Protokoll bietet **informationstheoretische Sicherheit** basierend auf der Monogamie der Verschränkung und dem Bell'schen Theorem. Aktuelle Forschung (2020-2025) hat bedeutende experimentelle Durchbrüche erzielt: **Satellitenbasierte Implementierungen** mit dem Micius-Satelliten über 1200 km; **Geräteunabhängige QKD-Demonstrationen** mit gefangenen Ionen über 400 m; und **kommerzielle Anwendungen** mit Schlüsselraten von $\sim$100 kbps über 20 km.

17.2 Das SARG04-Protokoll: Robustheit gegen Photonenzahlaufteilungsangriffe

17.2.1 Technische Mechanismen und Protokollstruktur

Das SARG04-Protokoll von Scarani, Acin, Ribordy und Gisin (2004) verwendet dieselben vier Quantenzustände wie BB84, implementiert jedoch ein **fundamental anderes Informationscodierungsschema**. Anstatt Messbasen anzukündigen, verwendet SARG04 einen **„Zustandspaar-Ankündigungsansatz“**, bei dem Alice Paare nicht-orthogonaler Zustände ankündigt (z.B. $\{|0\rangle, |+\rangle\}$ oder $\{|1\rangle, |-\rangle\}$).

Der **Schlüsselsiebungsprozess** erfolgt durch:

1. Zustandsvorbereitung mit gleicher Wahrscheinlichkeit für alle vier Zustände
2. Zufällige Messung von Bob in Z-Basis oder X-Basis
3. Ankündigung von Zustandspaaren durch Alice
4. Beibehaltung von Schlüsselbits nur wenn Bobs Messung einem der angekündigten Zustände entspricht

17.2.2 Sicherheitsvorteile und -nachweise

Verbesserte PNS-Angriffsresistenz ist der Hauptvorteil: SARG04 kann sichere Schlüssel sowohl aus Ein-Photon- als auch aus Zwei-Photon-Pulsen extrahieren. Die **Sicherheitsbeweise** von Tamaki & Lo (2006) zeigen unbedingte Sicherheit für ideale Ein-Photon-Quellen und **Zwei-Photon-Sicherheit** mit Fehlertoleranzen von 19,4% (Ein-Photon) und 6,56% (Zwei-Photon) bei bidirektionaler Kommunikation.

Aktuelle Entwicklungen (2020-2025) umfassen:

- **MDI-SARG04-Implementierungen** zur Eliminierung von Detektorschwachstellen
- **Sechs-Zustand-SARG04-Erweiterungen** für verbesserte PNS-Angriffsresistenz
- **Quantenmaschinelles Lernen-Integration** mit 30% höheren Schlüsselraten in 2024-Studien

17.3 Das Six-State Protocol: Asymmetrische Quantenschlüsselverteilung

17.3.1 Quantenmechanische Grundlagen

Das von Dagmar Bruss 1998 eingeführte Six-State Protocol erweitert BB84 durch die Verwendung **drei wechselseitig unvoreingenommener Basen** (sechs Quantenzustände). Die Zustände können als Bloch-Vektoren visualisiert werden, die entlang der positiven und negativen x-, y- und z-Achsen der Bloch-Sphäre zeigen:

$$\textbf{X-Basis (Geradlinig):} \quad |0\rangle \text{ und } |1\rangle \tag{241}$$

$$\textbf{Y-Basis (Diagonal):} \quad |+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \text{ und } |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \tag{242}$$

$$\textbf{Z-Basis (Kreisförmig):} \quad |+i\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}} \text{ und } |-i\rangle = \frac{|0\rangle - i|1\rangle}{\sqrt{2}} \tag{243}$$

17.3.2 Sicherheitsvorteile und theoretische Analyse

Erhöhte Fehlererkennungsrate: Ein Lauscher muss korrekt aus drei Basen statt zwei wählen, was die Wahrscheinlichkeit unentdeckten Lauschens

von 3/4 (BB84) auf 2/3 pro Qubit reduziert. **Verbesserte Rauschtoleranz**: Das Protokoll kann höhere Bitfehlerraten tolerieren als BB84, mit einem Schwellenwert von 12,611% gegenüber BB84s 11,002%.

Praktische Sicherheit mit Schwellenwertdetektoren: Kato & Tamaki (2016) bewiesen, dass das Six-State Protocol seine Sicherheitsvorteile selbst mit praktischen Schwellenwertdetektoren beibehält, mit einer minimalen Differenz von nur 0,008% zwischen Schwellenwert- und Photonenzahlauflösenden Detektoren.

17.4 Device-Independent Quantum Key Distribution: Ultimative Sicherheit

17.4.1 Fundamentale Prinzipien und Bell-Tests

Device-Independent QKD (DI-QKD) repräsentiert den Goldstandard der quantenkryptographischen Sicherheit und bietet **beispiellosen Schutz** durch die Eliminierung der Notwendigkeit, Quantengeräte zu charakterisieren oder zu vertrauen. Das Protokoll nutzt **Bell-Ungleichungsverletzungen** zur Sicherheitszertifizierung, wobei die CHSH-Ungleichung am häufigsten verwendet wird:

- **Klassische Grenze**: $S \leq 2$
- **Quantengrenze (Tsirelson-Schranke)**: $S \leq 2\sqrt{2} \approx 2.828$
- **Praktische Sicherheitsschwelle**: $S > 2.362$

17.4.2 Experimentelle Durchbrüche und aktuelle Implementierungen

Bahnbrechende Experimente (2022-2025) umfassen:

- Das **Münchener Experiment** mit 400 m Entfernung, $S = 2.578(75)$ und 0,07 Bits pro Verschränkungsereignis
- Das **Oxforder Experiment** mit der ersten finiten SchlüsselgröSSensicheren DI-QKD-Demonstration
- **Photonische Demonstrationen** über 220 m Glasfaser

Technische Herausforderungen bestehen in **hohen Detektionseffizienz-Anforderungen** (>80-90%), **Entfernungsbegrenzungen** durch Kanalverluste, und **niedrigen Schlüsselraten**. **Lösungsansätze** umfassen supraleitende Nanodraht-Detektoren, Heraldierungs-Mechanismen und Quantenrepeater.

17.5 Quantum Digital Signatures: Informationstheoretische Authentifizierung

17.5.1 Theoretische Grundlagen und Gottesman-Chuang-Rahmenwerk

Quantum Digital Signatures nutzen fundamentale quantenmechanische Prinzipien für informationstheoretisch sichere Authentifizierung. Das **Gottesman-Chuang-Rahmenwerk** (2001) führte asymmetrische Quantenschlüssel ein, deren exakte Identität nur dem Unterzeichner bekannt ist.

Sicherheitsprinzipien basieren auf:

- Dem No-Cloning-Theorem
- Messungs-störungen
- Statistischer Sicherheit durch universelle Hash-Funktionen

17.5.2 Praktische Implementierungen und Effizienzverbesserungen

Experimentelle Meilensteine umfassen:

- Die **Clarke et al. Demonstration** (2012) mit phasencodierten kohärenten Zuständen
- **Hocheffiziente OTUH-QDS-Protokolle** (2023) mit 10^8-facher Effizienzverbesserung
- **Chipbasierte Netzwerke** (2025) mit integrierten photonischen Schaltungen über 200 km

Aktuelle Leistungsmetriken zeigen:

- Signaturenraten von $> 10^4$ mal pro Sekunde für GroSSstadtbereiche
- Experimentelle Raten von 1,22 tps für 1-Megabit-Dokumente
- Sicherheitsschranken von 10^{-32} in experimentellen Implementierungen

17.6 Quantum Secret Sharing: Quantenschwellenwert-Schemata

17.6.1 Quantenmechanische Grundlagen und Protokollstruktur

Quantum Secret Sharing erweitert klassische Geheimnisaufteilungsschemas durch **Quantenkorrelationen** und **Verschränkung**. Das **Hillery-Buek-Berthiaume-Protokoll** (1998) nutzt GHZ-Zustände für die Geheimnisaufteilung:

$$|\psi\rangle_{GHZ} = \frac{|000\rangle + |111\rangle}{\sqrt{2}} \tag{244}$$

Quantenbegrenzung: $n < 2k$ aufgrund des No-Cloning-Theorems, da andernfalls zwei unabhängige Kopien des Geheimnisses aus disjunkten k-Anteil-Mengen rekonstruiert werden könnten.

17.6.2 Erweiterte Schemata und Verifikation

Graphzustand-basierte QSS bietet flexible Quantenressourcen mit maSS-geschneiderten Zugangsstrukturen. **Kontinuierliche Variable QSS** (2023) demonstrierte praktische, skalierbare Protokolle über 25-55 km Glasfaserverbindungen mit Schlüsselraten von 0,0061-7, 14×10^{-4} Bits/Puls.

Verifizierbare Quantengeheimnisaufteilung adressiert unehrliche Teilnehmer durch:

- **Hash-Funktions-Verifikation**
- **Quantenverifikation** mittels Bell-Messungen und Verschränkungsaustausch

17.7 Fazit und Ausblick

Diese sechs fortgeschrittenen Quantenkryptographie-Protokolle repräsentieren die neuesten Entwicklungen in der Quanteninformationssicherheit und bieten jeweils einzigartige Vorteile für spezifische Anwendungsszenarien. **E91** und **DI-QKD** bieten die stärksten Sicherheitsgarantien durch geräteunabhängige Sicherheit, während **SARG04** und das **Six-State Protocol** praktische Verbesserungen gegenüber BB84 in spezifischen Angriffsszenarien bieten. **Quantum Digital Signatures** und **Quantum Secret Sharing** erweitern die Quantenkryptographie über die reine Schlüsselverteilung hinaus zu fortgeschrittenen kryptographischen Primitiven.

Die **aktuellen Forschungstrends** (2020-2025) zeigen erhebliche Fortschritte bei experimentellen Implementierungen, mit Demonstrationen über Hunderte von Kilometern und der Integration in praktische Netzwerke. **Zukünftige Entwicklungen** konzentrieren sich auf die Skalierung dieser Protokolle für Quanteninternet-Architekturen, die Integration mit Post-Quantum-Kryptographie und die Entwicklung vollständig integrierter quantenkryptographischer Systeme für kommerzielle Anwendungen.

18 Anwendungen in der Praxis

18.1 Quantenchemie

Die Simulation molekularer Systeme ist ein Hauptanwendungsgebiet.

Elektronischer Hamilton-Operator:

$$H = \sum_i h_{ii} a_i^\dagger a_i + \frac{1}{2} \sum_{ijkl} h_{ijkl} a_i^\dagger a_j^\dagger a_l a_k$$

wobei $a_i^\dagger, a_i$ Erzeugungs- und Vernichtungsoperatoren sind.

Jordan-Wigner-Transformation: Fermionische Operatoren werden auf Spin-Operatoren abgebildet:

$$a_j = \left(\prod_{k<j} Z_k \right) \frac{X_j + iY_j}{2}$$

Rechnung 67 (H-Molekül in minimaler Basis)**.** *Für das H-Molekül in der STO-3G-Basis erhält man nach der Jordan-Wigner-Transformation:*

$$H = g_0 I + g_1 Z_0 + g_2 Z_1 + g_3 Z_0 Z_1 + g_4 X_0 X_1 + g_5 Y_0 Y_1$$

Mit den Koeffizienten:

$$g_0 = -1.0523732 \tag{245}$$
$$g_1 = 0.39793742 \tag{246}$$
$$g_2 = -0.39793742 \tag{247}$$
$$g_3 = -0.01128010 \tag{248}$$
$$g_4 = g_5 = 0.18093119 \tag{249}$$

Der Grundzustand kann mit dem VQE-Algorithmus gefunden werden.

18.2 Optimierungsprobleme

Quadratic Unconstrained Binary Optimization (QUBO):

$$\min_{x \in \{0,1\}^n} \sum_{i<j} Q_{ij} x_i x_j + \sum_i Q_{ii} x_i$$

Quantum Approximate Optimization Algorithm (QAOA):

$$|\psi(\vec{\gamma}, \vec{\beta})\rangle = \prod_{p=1}^{P} e^{-i\beta_p H_B} e^{-i\gamma_p H_C} |+\rangle^{\otimes n}$$

wobei H_C der Kosten-Hamilton-Operator und $H_B = \sum_i X_i$ der Misch-Hamilton-Operator ist.

19 Quantenhardware - Technologievergleich und aktuelle Entwicklungen

Drei Quantencomputing-Plattformen haben sich als führende Kandidaten für fehlertolerante Quantenberechnungen etabliert, wobei jede bedeutende jüngste Fortschritte in Richtung praktischen Quantenvorteils demonstriert. **Supraleitende Qubits führen bei aktuellen SystemgröSSen**, mit IBMs Roadmap die 100.000 Qubits bis 2033 anstrebt und Googles Willow unter der Schwelle liegende Quantenfehlerkorrektur erreicht. **Gefangene Ionen zeichnen sich durch Gate-Fidelities aus**, wobei IonQs Forte-System 99,96% Einzel-Qubit-Operationen und universelle Gate-Sets demonstriert. **Neutrale Atome bieten beispiellose Programmierbarkeit**, mit QuEras 256-Atom-Aquila-System das 99,5% Zwei-Qubit-Fidelities durch Rydberg-Blockade-Physik erreicht.

19.1 Supraleitende Qubits

Die supraleitende Quantencomputing-Technologie hat in den letzten Jahren dramatische Fortschritte erzielt und stellt heute die industriell am weitesten entwickelte Plattform dar.

19.1.1 IBMs Roadmap bis 100.000 Qubits (2033)

IBMs umfassende Roadmap präsentiert den konkretesten Weg zu groSSskaligem Quantencomputing mit spezifischen Meilensteinen von den heutigen 133-Qubit Heron-Prozessoren bis zu 100.000 Qubits bis 2033. Die aktuelle **Quantum Heron-Architektur** verfügt über 133 Qubits mit fester Frequenz und abstimmbaren Kopplern und liefert 3-5Œ Leistungsverbesserungen gegenüber vorherigen Generationen durch praktische Eliminierung von Crosstalk.

Der massive **Quantum Condor** demonstriert 1.121 Qubits mit über 1 Meile hochdichter kryogener Verkabelung und zeigt Fertigungskapazitäten auf, ohne sich auf Rechenleistung zu konzentrieren.

Die Roadmap 2025-2033 umfasst kritische Meilensteine:

- **Quantum Nighthawk** (2025): 120 Qubits in quadratischer Gitter-Topologie mit 16Œ Verbesserung der Schaltkreistiefe
- **Quantum Kookaburra** (2026): 1.386 Qubits über Multi-Chip-Prozessoren
- **Quantum Starling** (2028-2029): Ziel von 100 Millionen Quantengates auf 200 logischen Qubits

Schlüsselinnovationen umfassen **bivariate Fahrradcodes**, die 10Œ Qubit-Effizienz-Verbesserung gegenüber Oberflächencodes bieten, und **Quantenparallelisierung**, die mehrere logische Operationen gleichzeitig ermöglicht.

19.1.2 Googles Willow-Chip und Quantenfehlerkorrektur

Googles Willow-Chip stellt einen Durchbruch in der Quantenfehlerkorrektur dar und erreicht die erste Demonstration von Unter-Schwellenwert-Leistung mit 105 supraleitenden Qubits. Das System erreicht **T-Kohärenzzeiten von 68 ś 13 s** (5Œ Verbesserung gegenüber Sycamores 20 s) und nähert sich 100 s T-Zeiten.

Am bedeutsamsten demonstriert Willow **exponentielle Fehlerunterdrückung**:

- 3Œ3 logische Qubits etablieren Baseline-Fehlerraten
- 5Œ5 logische Qubits erreichen 2Œ Reduktion
- 7Œ7 logische Qubits bieten 2,14Œ Gesamtverbesserung

Dies beweist definitiv die Skalierbarkeit von Oberflächencodes.

19.1.3 Kohärenz-Zeiten und Gate-Fidelities

State-of-the-art Kohärenz-Leistung variiert dramatisch zwischen Implementierungen. Labordemonstrationen erreichen **T = 1,43 ms** für Fluxonium-Qubits und **T = 1,155 ms** für Tantal-basierte Transmons (IQM Quantum Computers). Rekord-Gate-Fidelities umfassen **99,98% Einzel-Qubit** und **99,92% Zwei-Qubit** Operationen (RIKEN/Toshiba 2024).

Mathematische Dekohärenz-Modelle bieten fundamentale Leistungsgrenzen. Energierelaxation folgt exponentialem Zerfall mit Rate $\gamma_1 = 1/T_1$:

$$P(|1\rangle \to |0\rangle, t) = \exp(-t/T_1) \tag{250}$$

Reine Dephasierung beeinflusst kohärente Superpositionen:

$$\langle\sigma_x\rangle(t) = \langle\sigma_x\rangle(0) \times \exp(-t/T_2^*) \tag{251}$$

Die Echo-Kohärenzzeit-Beziehung:

$$T_2 = \frac{2T_1T_2^*}{2T_1 + T_2^*} \tag{252}$$

Physikalische Rauschquellen setzen fundamentale Begrenzungen. Johnson-Nyquist-Thermorauschen trägt Leistungsspektraldichte $S_v(f) = 4k_BTR$ bei.

Flussrauschen zeigt 1/f-Skalierung mit $S_\phi(f) = A^2/f^\alpha$, wobei $\alpha \approx 1$ und $A \sim 1 - 10\mu\Phi_0/\sqrt{\text{Hz}}$.

19.2 Gefangene Ionen

Die Technologie gefangener Ionen bietet die höchste Präzision und Kontrolle aller Quantencomputing-Plattformen und erreicht auSSergewöhnliche Fidelities durch elektromagnetische Manipulation.

19.2.1 IonQs Forte-System

IonQs Forte-System repräsentiert den aktuellen Höhepunkt des Quantencomputings mit gefangenen Ionen und verfügt über **36 physikalische Qubits** mit software-konfigurierbarer Architektur unter Verwendung akusto-optischer Deflektor. Das System verwendet **Ytterbium-Ionen (Yb+)**, die in linearen Paul-Fallen mit 3 m Abstand gefangen sind, und erreicht **99,96% Einzel-Qubit** und **99,6% Zwei-Qubit** Gate-Fidelities.

Kohärenzzeiten überschreiten **1000 ms** sowohl für T als auch T und übertreffen damit dramatisch supraleitende Systeme. Die **All-zu-All-Konnektivität** innerhalb von Ionenketten ermöglicht es jedem Qubit-Paar, direkt durch Coulomb-Kräfte zu interagieren.

19.2.2 Universale Gate-Sets

Universelle Gate-Sets beruhen auf fundamentalen Quantenoperationen, die durch Pauli-Matrizen ausgedrückt werden. Einzel-Qubit-Rotationen folgen:

$$R_x(\theta) = \exp(-i\theta\sigma_x/2) = \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \tag{253}$$

Das **Mølmer-Sørensen-Gate** dient als native Zwei-Qubit-Verschränkungsoperation:

$$U_{MS} = \exp\left(-i\frac{\pi}{4}\sum_{i<j}\sigma_x^{(i)}\sigma_x^{(j)}\right) \tag{254}$$

Für zwei Qubits produziert dies:

$$U_{MS} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 0 & 0 & -i \\ 0 & 1 & -i & 0 \\ 0 & -i & 1 & 0 \\ -i & 0 & 0 & 1 \end{pmatrix} \tag{255}$$

Der **Wechselwirkungs-Hamiltonian** für N Ionen koppelt interne Zustände an Bewegungsmoden:

$$H_{MS} = \sum_i \Omega_i(t)\sigma_x^{(i)} \sum_m \frac{\eta_{i,m}}{\sqrt{2}} \left(a_m e^{-i\delta_m t} + a_m^\dagger e^{i\delta_m t}\right) \tag{256}$$

wobei $\eta_{i,m}$ Lamb-Dicke-Parameter darstellt, die Qubits an Phononenmoden koppeln.

19.2.3 Skalierungsherausforderungen

Skalierungsherausforderungen präsentieren fundamentale Begrenzungen. Bewegungsheizraten skalieren ungünstig als $\omega_m^{-2} \propto N^4$ für niederfrequente Moden in groSSen Kristallen. Modenabstand nimmt ab als $\Delta\omega \propto N^{-2}$, was Frequenzüberlastung schafft, die die Kontrolle kompliziert.

Die **säkulare Frequenzstabilitätsbedingung** beschränkt Falleparameter:

$$\omega_z^2 = \frac{2eV_{rf}}{m\Omega^2 r_0^2} - \frac{eV_{dc}}{md^2} \tag{257}$$

wobei V_{rf} die RF-Amplitude ist, Ω die RF-Frequenz und r_0 die charakteristische FallengröSSe. Der **Lamb-Dicke-Parameter** $\eta = k/\sqrt{2m\omega_m}$ muss $\eta \ll 1$ für hochpräzise Gates erfüllen.

Fortgeschrittene Architekturen adressieren Skalierungsbegrenzungen durch segmentierte Fallen, 2D-Arrays und Quantum Charge-Coupled Devices (QCCD). Transportoperationen skalieren als $t_{transport} \propto \sqrt{d/a_{max}}$, wobei d die Entfernung und a_{max} die maximale Beschleunigung ist.

19.3 Neutrale Atome

Neutrale Atomsysteme ermöglichen die flexibelste Quantencomputing-Plattform durch Rydberg-Physik und bieten beispiellose Programmierbarkeit für Quantensimulation.

19.3.1 QuEras Aquila-System

QuEras Aquila-System demonstriert die flexibelste Quantencomputing-Plattform mit **256 neutralen Rubidium-Atomen**, die in programmierbaren 2D-Arrays angeordnet sind. Das System erreicht **99,5% Zwei-Qubit-Gate-Fidelities** und **99,97% Einzel-Qubit-Fidelities** durch präzise optische Kontrolle.

Unter Verwendung von **Rb-Atomen** in Uhrzuständen mit 2-4 m Abstand ermöglicht die Plattform beliebige Konnektivitätsmuster und Echtzeitrekonfiguration während Quantenschaltkreisen.

19.3.2 Rydberg-Blockade

Rydberg-Blockade-Physik bietet den fundamentalen Wechselwirkungsmechanismus. Hochangeregte Rydberg-Zustände (n 50-100) zeigen starke **Van-der-Waals-Wechselwirkungen**:

$$V_{vdW}(R) = -\frac{C_6}{R^6} \tag{258}$$

wobei der Koeffizient $C_6 \approx 2\pi \times 862$ **MHzům** für 53S/-Zustände beträgt. Dies erzeugt einen **Blockaderadius** $R_b \approx 8,8$ **m** bei maximaler Rabi-Frequenz $\Omega_{max} = 4,6$ MHz und verhindert simultane Rydberg-Anregung benachbarter Atome.

Der **vollständige Hamiltonian** für N Atome umfasst Antrieb und Wechselwirkungen:

$$H = \sum_i \left[\frac{\hbar\Omega(t)}{2} \left(\sigma_i^+ e^{i\phi(t)} + \sigma_i^- e^{-i\phi(t)} \right) - \hbar\Delta(t) n_i^r \right] + \sum_{i<j} V_{ij} n_i^r n_j^r \tag{259}$$

wobei $\Omega(t)$ und $\phi(t)$ zeitabhängige Rabi-Frequenz und Phase sind, $\Delta(t)$ die Verstimmung und n_i^r die Rydberg-Population darstellt.

Zwei-Photonen-Anregung verwendet sorgfältig gewählte Laserparameter: 420-nm ($\Omega_{420} = 2\pi \times 237$ MHz) und 1013-nm ($\Omega_{1013} = 2\pi \times 303$ MHz) mit Zwischenverstimmung $\Delta/2\pi = 7,8$ GHz. Dies produziert effektive Zwei-Photonen-Rabi-Frequenz $\Omega/2\pi = 4,6$ MHz, während Streuung durch **atomare dunkle Zustände** unterdrückt wird.

19.3.3 Programmierbare Quantensimulatoren

Programmierbare Quantensimulatoren operieren sowohl in analogen als auch digitalen Modi. **Analoge Simulation** implementiert kontinuierliche Hamiltonian-Evolution für Vielteilchen-Physikstudien, einschlieSSlich Quantenphasenübergänge auf 256-Atom-Systemen. **Digitale Gate-Sequenzen** verwenden optimale Kontrollpulse mit typischen Gate-Zeiten $\Omega T/2\pi \approx 1,2-1,8$ und 20-ns-Anstiegszeiten.

Multi-Qubit-Gate-Fähigkeiten umfassen native **CCZ-Gates** mit 97,9% Fidelity und **parallele Operationen** auf bis zu 60 Atomen gleichzeitig. Die

CZ-Gate-Implementierung verwendet optimale Kontrolle mit Phasenprofil $\phi(t) = A\cos(\omega t - \phi_0) + \delta_0 t$ und erreicht Gate-Parameter $A/2\pi = 0,0988$ und $\omega/\Omega = 1,3629$.

Anwendungen umfassen Quantenannealing für Optimierungsprobleme, analoge Simulation von Quantenmagnetismus und digitale Quantenalgorithmen. Die **einzigartigen Stärken** der Plattform umfassen beliebige Graphkonnektivität, Echtzeitrekonfiguration und native Multi-Qubit-Operationen.

19.4 Leistungskonvergenz und Quantenvorteil

Vergleichende Analyse zeigt komplementäre Stärken über Quantenhardware-Plattformen hinweg. **Supraleitende Qubits** führen in SystemgröSSe und Fertigungsreife, mit IBMs Roadmap, die den klarsten Weg zu 100.000+ Qubits bietet. **Gefangene Ionen** glänzen in Gate-Fidelity und Kohärenz mit >99,9% Operationen und Millisekunden-Kohärenzzeiten. **Neutrale Atome** bieten unvergleichliche Programmierbarkeit und parallele Operationen mit 99,5% Gates über flexible Architekturen.

Fehlerkorrekturschwellen (0,1% für Oberflächencodes) sind nun über alle Plattformen hinweg erreichbar, wobei Googles Willow Unter-Schwellenwert-Leistung demonstriert und IonQ sich Fünf-9er-Fidelity nähert. Die **Quantenvorteil-Zeitlinie** deutet auf 2025-2030 als kritische Periode hin, in der diese Plattformen rechnerisch nützliche Anwendungen demonstrieren werden.

20 Aktuelle Herausforderungen und Zukunftsaussichten

20.1 NISQ-Era Limitationen

In der aktuellen NISQ-Era (Noisy Intermediate-Scale Quantum) sind die Hauptlimitationen:

Kohärenzzeiten: - Supraleitende Qubits: $T_1 \sim 100\mu s$, $T_2 \sim 50\mu s$ - Ionenfallen: $T_1 \sim 1min$, $T_2 \sim 1ms$ - Topologische Qubits: Theoretisch sehr lange, praktisch noch nicht realisiert

Gate-Fidelities: - Einzelqubit-Gatter: 99.9% - Zweitqubit-Gatter: 99% - Dreiqubit-Gatter: 95%

20.2 Schwellenwerte für Quantenvorteil

Fault-Tolerant Threshold: Für universelle, fehlertolerante Quantencomputer wird eine physikalische Fehlerrate unter 10^{-3} benötigt.

Logische Qubits: Mit dem Oberflächencode sind etwa $1000 - 10000$ physikalische Qubits pro logischem Qubit nötig.

21 Fazit

Das Quantencomputing ist eine transformative Technologie, die das Potenzial hat, die Art und Weise, wie wir komplexe Probleme lösen, grundlegend zu verändern. Durch die Nutzung der einzigartigen Phänomene der Quantenmechanik Superposition, Verschränkung und Interferenz können Quantencomputer bestimmte Aufgaben, die für klassische Computer unerreichbar sind, effizient bewältigen.

Die detaillierte mathematische Beschreibung der Quantenverschränkung, insbesondere durch Tensorprodukte, Schmidt-Zerlegung und VerschränkungsmaSSe, unterstreicht die Komplexität und Eleganz dieses Phänomens. Die Fähigkeit, solche verschränkten Zustände zu erzeugen und zu manipulieren, ist der Schlüssel zur Leistungsfähigkeit von Quantenalgorithmen wie Shor's und Grover's Algorithmus.

Die mathematischen Grundlagen der Dichteoperatoren, Quantenteleportation und Fehlerkorrektur zeigen die Notwendigkeit einer rigorosen theoretischen Behandlung für das Verständnis praktischer Quantensysteme. Die detaillierten Berechnungen und Beispiele in diesem Report demonstrieren, wie die abstrakte Quantenmechanik in konkrete Algorithmen und Protokolle umgesetzt wird.

Obwohl die Technologie noch in den Kinderschuhen steckt und erhebliche Herausforderungen in Bezug auf Hardware-Skalierbarkeit, Fehlerraten und Kohärenzzeiten bestehen, sind die Fortschritte in der Forschung und Entwicklung beeindruckend. Die internationale Gemeinschaft arbeitet intensiv daran, diese Hürden zu überwinden und die Vision eines fehlertoleranten, universellen Quantencomputers zu verwirklichen.

Das Quantencomputing ist nicht dazu gedacht, klassische Computer vollständig zu ersetzen, sondern sie zu ergänzen, indem es Probleme löst, die auSSerhalb ihrer Reichweite liegen. Die Zukunft wird wahrscheinlich eine Koexistenz und Integration beider Paradigmen sehen, wobei Quantencomputer als leistungsstarke Beschleuniger für spezifische, rechenintensive Aufgaben dienen. Die mathematischen Grundlagen, die in diesem Report ausführlich behandelt wurden, bilden das Fundament für diese revolutionäre Technologie.

22 Adiabatisches Quantencomputing und Quantum Annealing

Das adiabatische Quantencomputing und Quantum Annealing repräsentieren heute die einzige kommerziell verfügbare Quantencomputing-Technologie für Produktionsanwendungen und bieten bereits demonstrierte Quantenvorteile bei spezifischen Optimierungsproblemen. **D-Wave Systems führt mit über 5.000 Qubits und bewiesenen Industrieanwendungen**, während theoretische Fortschritte das Verständnis der fundamentalen Prinzipien vertiefen. Diese Technologie nutzt die Quantenmechanik für adiabatische Evolution zur Lösung kombinatorischer Optimierungsprobleme durch kontinuierliche Grundzustandsverfolgung, wodurch sie sich fundamental von universellen Gate-basierten Ansätzen unterscheidet.

Die jüngsten Durchbrüche zeigen **6561-fache Geschwindigkeitsvorteile** gegenüber klassischen Solvern bei groSSen QUBO-Problemen und praktische Anwendungen von Volkswagen bis Mastercard. Gleichzeitig erweitern neue Algorithmen wie Reinforcement Quantum Annealing und hybride Ansätze die Anwendbarkeit erheblich. Die theoretischen Grundlagen sind durch das Born-Fock-Adiabatensatz gut etabliert, während aktuelle Forschung nonadiabatische Regime und geometrische Phasen erforscht.

22.1 Theoretische Grundlagen des adiabatischen Quantencomputings

22.1.1 Mathematische Prinzipien und adiabatisches Theorem

Das adiabatische Quantencomputing basiert auf dem **Born-Fock-Adiabatensatz** (1928), der besagt, dass ein physikalisches System in seinem momentanen Eigenzustand verbleibt, wenn eine Störung ausreichend langsam einwirkt und eine Energielücke zwischen dem Eigenwert und dem Rest des Hamiltonian-Spektrums existiert.

Der zentrale mathematische Formalismus verwendet einen **zeitabhängigen Hamiltonoperator**:

$$H(t) = (1 - s(t))H_0 + s(t)H_f \tag{260}$$

wobei $s(t)$ die Annealing-Zeitfunktion ($0 \leq s \leq 1$), H_0 der Initial-Hamiltonoperator mit bekanntem Grundzustand und H_f der Problem-Hamiltonoperator mit der Lösungskodierung ist.

Die **mathematische Bedingung für Adiabatizität** erfordert:

$$T \gg \frac{1}{g_{\min}^2} \tag{261}$$

wobei $g_{\min}$ die minimale spektrale Lücke und T die Evolutionszeit ist. Diese Bedingung bestimmt direkt die Laufzeitkomplexität adiabatischer Algorithmen.

22.1.2 Hamiltonoperatoren und Spektralanalyse

Die Zeitentwicklung folgt der **Schrödinger-Gleichung**:

$$i\hbar\frac{\partial|\psi(t)\rangle}{\partial t} = H(t)|\psi(t)\rangle \tag{262}$$

Für adiabatische Entwicklung mit Zeitvariable s wird der Hamiltonoperator als:

$$H(s) = f(s)H_M + r(s)H_I \tag{263}$$

formuliert, wobei $f(s)$ monoton steigend und $r(s)$ monoton fallend ist.

Die **spektrale Lücke** ist definiert als:

$$\Delta(s) = E_1(s) - E_0(s) \tag{264}$$

mit der minimalen spektralen Lücke $g_{\min} = \min_{0\leq s\leq 1}\Delta(s)$.

22.1.3 Universalität und Komplexitätstheorie

Adiabatische Quantencomputation ist polynomiell äquivalent zum Gate-basierten Quantencomputing (Aharonov et al., 2007). Die allgemeine Laufzeitschranke ist:

$$T = \mathcal{O}\left(\frac{L^2}{g_{\min}^2}\right) \tag{265}$$

wobei L die ProblemgröSSe und $g_{\min}$ die minimale spektrale Lücke ist.

Das k**-lokale Hamiltonian-Problem ist QMA-vollständig** für $k \geq 2$, was die theoretische Bedeutung adiabatischer Ansätze unterstreicht.

22.2 Quantum Annealing: Funktionsweise und theoretische Grundlagen

22.2.1 Grundprinzip und Hamiltonoperator

Quantum Annealing ist eine Heuristik basierend auf AQC-Prinzipien mit dem **QA-Hamiltonoperator**:

$$H(t) = A(t)H_M + B(t)H_P \tag{266}$$

wobei $A(t)$ vom Maximum zu Null und $B(t)$ von Null zum Maximum variiert.

Der **Ising-Hamiltonoperator** für QA lautet:

$$H_I = \sum_i h_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \tag{267}$$

kombiniert mit dem **Transversalfeld-Hamiltonoperator**:

$$H_T = -\sum_i \sigma_i^x \tag{268}$$

22.2.2 Quantentunneling vs. thermische Aktivierung

Der fundamentale Unterschied zu klassischem Simulated Annealing liegt im **Quantentunneling**-Mechanismus:

- **Simulated Annealing**: Thermische Aktivierung über Energiebarrieren mit Akzeptanzwahrscheinlichkeit $P = \exp(-\Delta E/kT)$
- **Quantum Annealing**: Quantentunneling durch Energiebarrieren mit quantenmechanisch bestimmter Tunnelwahrscheinlichkeit

Die **Tunnelwahrscheinlichkeit** ist gegeben durch:

$$P_{\text{tunnel}} \propto \exp\left(-\frac{2}{\hbar}\int \sqrt{2m(V(x)-E)}\,dx\right) \tag{269}$$

22.2.3 Nonadiabatische Regime und aktuelle Entwicklungen

Yan & Sinitsyn (2022) zeigten, dass für nonadiabatisches QA mit $g(t) = g/t$:

$$P_n = \frac{(1-p)p^n}{1-p^N} \tag{270}$$

wobei $p = \exp(-2\pi g/N)$ und N die Dimension des Hilbert-Raums ist.

Pseudo-adiabatische Evolution für $T \gg 1/\Delta E_I$ zeigt:

$$\langle n \rangle \sim \frac{1}{g^\alpha} \tag{271}$$

mit Potenzgesetz-Relaxation statt logarithmischer Relaxation.

22.3 Technische Implementierung und Hardware-Entwicklung

22.3.1 D-Wave Systems: Technologieführer

D-Wave Advantage2 (2025) stellt die neueste Generation dar:

- **4.400+ supraleitende Qubits** mit Zephyr-Architektur
- **20-fache Qubit-Konnektivität** (höchste verfügbare)
- **40% höhere Energieskala** gegenüber Advantage
- **Konstanter Energieverbrauch**: 12,5 kW
- **Fast Anneal-Funktionalität** für schnelle Berechnungen

Die **Pegasus-Topologie** (D-Wave Advantage) bietet:

- **Nominelle Länge**: 12, **Grad**: 15
- **Interne Koppler**: 12 pro Qubit
- **Native K_4 und $K_{6,6}$ Subgraphen**
- **Verbesserte Embedding-Effizienz**

22.3.2 Supraleitende Flux-Qubits und Josephson-Kontakte

Die **technische Implementierung** basiert auf:

- **Drei Josephson-Kontakte pro Qubit**
- **Sandwich-Aufbau**: Supraleiter/Isolator/Supraleiter (SIS)
- **Aluminiumoxid-Barriere** ($\sim$1-2 nm dick)
- **Betrieb bei** $E_J/E_C \approx 10$ (E_J = Josephson-Energie, E_C = Ladeenergie)

Die **Josephson-Beziehungen** lauten:

$$I = I_0 \sin(\delta) \tag{272}$$

$$V = \frac{\hbar}{2e}\frac{d\delta}{dt} \tag{273}$$

22.3.3 Kryogene Systeme und Skalierung

Dilution Refrigerator-Technologie ermöglicht:

- **Betriebstemperatur**: 10-25 mK
- **Kühlleistung**: 250-600 μW bei 100 mK
- **Kontinuierlicher Betrieb** durch Cryogen-free-Systeme
- **Mischung von ^{3}He und ^{4}He** für Quantenkühlung

Skalierungsprojekte wie IBM Goldeneye sind für **1M-Qubit-Systeme** ausgelegt mit 10$\times$ weniger Laborplatz als konventionelle Systeme.

22.4 Anwendungsgebiete und praktische Erfolge

22.4.1 Optimierungsprobleme und QUBO-Formulierung

Aktuelle Forschung zeigt beeindruckende Ergebnisse:

- **Kim et al. (2025)**: D-Wave Advantage erreicht **0,013% höhere Genauigkeit und 6561-fache Geschwindigkeitssteigerung** gegenüber klassischen Solvern bei groSSen QUBO-Problemen
- **Vollständig verbundene QUBO-Matrizen** mit bis zu 10.000 Variablen erfolgreich gelöst
- **Hybride Quantenklassische Ansätze (HQA)** zeigen durchgehend beste Leistung

Portfolio-Optimierung demonstriert:

- **Portfoliosteigerung von 200.000 INR** gegenüber Benchmark durch QA
- **Reverse Quantum Annealing** besonders effektiv bei Risiko-Rendite-Optimierung
- **Markowitz-Theorie** erfolgreich in QA-kompatible Formate übersetzt

22.4.2 Maschinelles Lernen und Datenanalyse

Quantum Machine Learning zeigt vielversprechende Fortschritte:

- **QBoost-Algorithmus** auf D-Wave Advantage zeigt kompetitive Leistung bei binären Klassifikationsproblemen
- **Boltzmann-Maschinen** mit verbesserter Leistung bei begrenzten Trainingsdaten
- **Abel et al. (2024)**: Universeller AQC-Ansatz für neuronale Netze

Merkmalsselektion profitiert von:

- **Natürlicher Handhabung kombinatorischer Aspekte**
- **Effizienter Exploration hochdimensionaler Merkmalsräume**
- **Robustheit gegenüber Rauschen**

22.4.3 Industrielle Anwendungen mit messbarem Impact

Kommerzielle Erfolge umfassen:

- **Volkswagen**: Routenoptimierung in Lissabon in Echtzeit
- **Mastercard**: Finanzoptimierung mit QA-Integration
- **Pattison Food Group**: **80% Reduzierung der Planungszeit**
- **Japan Tobacco**: Quantenunterstützte Medikamentenentwicklung

Logistikoptimierung zeigt:

- **Aisin Corporation**: Multi-Truck-Routing mit $\sim$2500 Variablen
- **Coca-Cola Japan**: Optimierung von 700.000 Verkaufsautomaten
- **Hybride Workflows** zerlegen groSSe Probleme in QA-lösbare Teilprobleme

22.5 Vorteile und Limitationen gegenüber Gate-basierten Systemen

22.5.1 Quantum Annealing Stärken

Rauschtoleranz: QA zeigt inhärente Robustheit:

- **Adiabatische Entwicklung** weniger empfindlich gegenüber Gate-Fehlern
- **Dekoherenz in Energie-Eigenbasis** beeinträchtigt AQC nicht zwangsläufig
- **Boundary-Cancellation-Methoden** bleiben auch in offenen Systemen vorteilhaft

Skalierbarkeit: QA-Systeme erreichen bereits heute:

- **D-Wave Advantage**: 5000+ Qubits
- **D-Wave Advantage2**: 7000+ Qubits
- **Gate-basierte Systeme**: Typisch 50-1000 Qubits mit hoher Fidelity

Kommerzielle Verfügbarkeit: QA ist seit 2011 kommerziell verfügbar, während Gate-basierte Systeme noch **7-15 Jahre** von Produktionsreife entfernt sind.

22.5.2 Limitationen und Herausforderungen

Strukturelle Beschränkungen:

- **Nur QUBO/Ising-Modelle** direkt lösbar
- **Keine universelle Quantenberechnung** möglich
- **Shor-Algorithmus und Fourier-Transformationen** nicht verfügbar

Einbettungsprobleme:

- **Dichte Probleme** erfordern oft mehr physikalische Qubits als verfügbar
- **Effektive ProblemgröSSe** reduziert durch Topologie-Beschränkungen
- **Skalierung ist problemspezifisch**, nicht universal

Präzisionslimitationen:

- **Statistische Schwankungen** erfordern multiple Messungen
- **Ketten-Brüche** in Qubit-Einbettung verringern Genauigkeit
- **Intrinsische Präzisionslimitationen** der aktuellen Prozessoren

22.6 Aktuelle Forschungsentwicklungen und wissenschaftliche Erkenntnisse 2024-2025

22.6.1 Algorithmische Durchbrüche

Reinforcement Quantum Annealing (RQA): Entwicklung intelligenter Agenten-Systeme, die im Hamiltonian-Raum suchen und mit Quantum Annealern interagieren.

Hybrid-Algorithmen: IBM und Kipu Quantum demonstrierten 2025 den **BBB-DCQO-Algorithmus** auf IBMs Heron-Prozessor, der HUBO-Probleme schneller löst als klassische oder QA-Methoden.

Quantum Error Mitigation: Einführung von **Zero-Noise Extrapolation (ZNE)** für Quantum Annealing durch Zero-Temperature- und Zero-Time-Extrapolationen.

22.6.2 Internationale Forschungskooperationen

International Network on Quantum Annealing (INQA): Erste globale Kollaboration zwischen Nordamerika, Japan, EU und UK mit:

- **Wöchentliche Online-Seminare**
- **Jährliche internationale Konferenzen**
- **Finanzierung von Austauschbesuchen**

Nationale Quantenprogramme:

- **USA**: National Quantum Initiative mit >1,8 Milliarden USD
- **Deutschland**: Quantum Technologies Initiative mit 2 Milliarden EUR
- **EU**: Quantum Flagship mit 1 Milliarde EUR
- **Kanada**: Quantum Canada Strategy mit QA-Fokus

22.6.3 Neue Anwendungsgebiete

Materialwissenschaften:

- **Quantum Annealing-unterstütztes Lattice-Optimierung (QA-LO)** für High Entropy Alloys
- **Metamaterial-Design** für optische Anwendungen
- **Quantentransparente Radiative Cooler**

Industrie 4.0:

- **Unit Commitment-Probleme** in der Energiewirtschaft
- **Supply Chain-Optimierung** mit messbaren Verbesserungen
- **Software Engineering**: Search-Based Software Engineering (SBSE)

22.7 Zukunftsperspektiven und technologische Herausforderungen

22.7.1 Technologische Roadmaps

Kurzfristige Entwicklungen (2025-2027):

- **Advantage3** mit höherer Qubit-Zahl
- **Verbesserte Fehlerkorrektur** durch ZNE-Integration
- **Optimierte Kryogene Systeme** für Skalierung

Langfristige Ziele (2030+):

- **100.000+ Qubit-Systeme** in Planung
- **Multi-Chip-Architekturen** für massive Skalierung
- **Fehlertolerante Quantum Annealing**

22.7.2 Wissenschaftliche Herausforderungen

Skalierbarkeit:

- **Kryogene Verkabelung** für groSSe Qubit-Zahlen
- **Crosstalk zwischen Qubits** bei hoher Dichte
- **Kalibrierungsaufwand** steigt exponentiell

Algorithmusentwicklung:

- **Hybride Quantum-Classical Co-Processing**
- **Edge-Computing** mit Quantum Annealing
- **Cloud-basierte Quantum-as-a-Service**

22.7.3 Marktprognosen

Quantum Computing-Umsatz wird 2025 **1 Milliarde USD** überschreiten, wobei Quantum Annealing Vorteile bei Optimierungsproblemen behalten wird, während Gate-Model-Systeme erst in 7-15 Jahren produktionsreif werden.

22.8 Relevante mathematische Formulierungen und Algorithmen

22.8.1 QUBO-Mathematik und Transformationen

Quadratic Unconstrained Binary Optimization hat die Form:

$$\min f(x) = \sum_i Q_{ii} x_i + \sum_{i<j} Q_{ij} x_i x_j \tag{274}$$

Ising-QUBO-Transformation erfolgt durch $\sigma_i = 2x_i - 1$:

$$Q_{ij} = \frac{J_{ij}}{4} \quad \text{für } i \neq j \tag{275}$$

$$Q_{ii} = \frac{h_i}{2} - \frac{1}{2} \sum_{j \neq i} J_{ij} \tag{276}$$

22.8.2 Annealing-Algorithmen und Optimierung

Optimierte Annealing-Funktionen minimieren:

$$T = \int_0^1 \frac{|\langle 1(s)|\frac{dH}{ds}|0(s)\rangle|}{g(s)^2}\, ds \tag{277}$$

Variational Quantum Eigensolver (VQE) minimiert:

$$E(\vec{\theta}) = \langle\psi(\vec{\theta})|H|\psi(\vec{\theta})\rangle \tag{278}$$

Quantum Approximate Optimization Algorithm (QAOA) verwendet:

$$|\psi_p(\vec{\gamma},\vec{\beta})\rangle = \prod_{i=1}^{p} e^{-i\beta_i H_B} e^{-i\gamma_i H_C} |+\rangle^{\otimes n} \tag{279}$$

22.8.3 Komplexitätstheorie und Laufzeitanalyse

Allgemeine Laufzeitschranke:

$$T = \mathcal{O}\left(\max_s \frac{|\langle 1(s)|\dot{H}(s)|0(s)\rangle|}{g_{\min}^2}\right) \tag{280}$$

Spektralgap für Circuit-Simulation:

$$g_{\min} \gtrsim \frac{\pi^2}{8(L+1)^2} \tag{281}$$

22.9 Fazit und wissenschaftliche Einordnung

Adiabatisches Quantencomputing und Quantum Annealing haben sich als **einzige kommerziell verfügbare Quantencomputing-Technologie** für Produktionsanwendungen etabliert. Die theoretischen Grundlagen sind durch das Born-Fock-Adiabatensatz und die polynomielle Äquivalenz zu Gate-basierten Systemen gut fundiert.

Praktische Erfolge in Industrieanwendungen von Volkswagen bis Mastercard demonstrieren bereits heute **messbare Quantenvorteile** bei spezifischen Optimierungsproblemen. Die 6561-fache Geschwindigkeitssteigerung bei groSSen QUBO-Problemen und die 80%ige Reduzierung von Planungszeiten belegen die praktische Relevanz.

Technologische Führerschaft zeigt D-Wave mit über 5.000 Qubits und kontinuierlicher Hardware-Entwicklung, während die internationale Forschungskooperation durch INQA und nationale Quantenprogramme die wissenschaftliche Basis stärkt.

Die **Zukunft der Quantentechnologie** wird durch die Koexistenz spezialisierter QA-Systeme für Optimierungsprobleme und universeller Gate-basierter Systeme geprägt sein. Hybride Ansätze versprechen dabei die gröSSten praktischen Fortschritte, indem sie die Stärken beider Paradigmen kombinieren.

Wissenschaftlich bleibt die Frage nach definitiven Quantenvorteilen für praktische Probleme aktiv, wobei die bisherigen Ergebnisse eine optimistische Perspektive für die weitere Entwicklung dieser faszinierenden Technologie stützen.

23 Branchenspezifische Anwendungen

Die deutsche Industrie positioniert sich als globaler Vorreiter im Quantencomputing mit Investitionen von über 10 Milliarden Euro und strategischen Partnerschaften in Schlüsselsektoren. Diese Analyse untersucht die aktuellen Anwendungen, technischen Spezifikationen und Rentabilitätsprognosen basierend auf aktuellen Marktdaten von 2024/2025.

23.1 Pharmaindustrie

23.1.1 Aktuelle Quantenpartnerschaften und Investitionen

Deutsche Pharmaunternehmen führen die globale Quantenadoption mit beispiellosen Investitionssummen an. Der weltweite Quantencomputing-Markt in der Arzneimittelforschung erreichte 2024 bereits 392 Millionen USD und soll bis 2035 auf 1,63 Milliarden USD anwachsen.

Roche etablierte mehrjährige Partnerschaften mit Cambridge Quantum Computing (CQC) unter Verwendung der EUMEN-Quantenchemie-Plattform speziell für die Alzheimer-Forschung. Zusätzlich kooperiert das Unternehmen mit QC Ware für quantenneuronale Netzwerke in der biomedizinischen Bildklassifizierung und unterhält Partnerschaften mit drei Doktoranden der Universität Oxford.

Merck KGaA investierte einen niedrigen zweistelligen Millionenbetrag in eine 3-jährige Kooperation mit HQS Quantum Simulations und ist Gründungsmitglied der QuPharm Alliance seit 2019. Über M Ventures investiert Merck aktiv in Quantum-Startups.

Boehringer Ingelheim erzielte als erstes Pharmaunternehmen eine Partnerschaft mit Google Quantum AI (2021-2024) und demonstrierte bemerkenswerte Beschleunigungen von **234Œ und 278Œ** für Molekülberechnungen von Cytochrom P450 und FeMoco-Elektronenstrukturen.

23.1.2 Quantenmechanische Berechnungsanforderungen

Die Simulation biologischer Systeme erfordert exponentiell skalierenden Rechenaufwand:

$$\text{Rechenaufwand}_{\text{klassisch}} = O(e^N) \tag{282}$$

wobei N die Anzahl der Quantenfreiheitsgrade darstellt.

Rechenbeispiel: Für ein mittelgroSSes Protein mit 300 Aminosäuren:

$$\text{Anzahl Atome} = 300 \times 20 = 6.000 \text{ Atome} \tag{283}$$

$$\text{Konfigurationen}_{\text{klassisch}} = 3^{6000} \approx 10^{2863} \tag{284}$$

$$\text{Qubits}_{\text{erforderlich}} = 6000 \times \log_2(3) \approx 9.480 \text{ Qubits} \tag{285}$$

Aktuelle technische Spezifikationen: IBM's Heron-Prozessor bietet 133 Qubits mit 99,5% Gate-Fidelity, während Google's Willow-Chip verbesserte Fehlerkorrektur demonstriert. Für praktische Anwendungen werden 10^6 Qubits für Fe_2S_2-Moleküle und 10^8 Qubits für komplexe Moleküle wie Alanin benötigt.

23.1.3 Wirtschaftliche Auswirkungen

Der Pharmasektor hat über **400 Millionen Euro** für Quanteninitiativen bereitgestellt, wobei die Novo Nordisk Foundation die gröSSte Einzelinvestition von **200 Millionen Euro** für fehlertolerantes Quantencomputing bereitstellt.

Kostenreduktionspotenzial: Aktuelle Entwicklungskosten von 4-10 Milliarden USD pro Medikament könnten um 30-50% reduziert werden:

$$\text{Einsparung} = 4 \times 10^9 \times 0.4 = 1.6 \times 10^9 \text{ USD pro Medikament} \tag{286}$$

Entwicklungszeiten verkürzen sich von 10-15 Jahren auf 7-10 Jahre, während Erfolgsraten von 10% auf 15-20% steigen könnten.

23.2 Finanzsektor

23.2.1 Goldman Sachs Quantenführerschaft

Goldman Sachs etablierte sich als Quantencomputing-Marktführer im Finanzwesen durch Partnerschaften mit Quantum Motion, QC Ware und AWS. Das Unternehmen entwickelte SShallow Monte CarloAlgorithmen mit **100Œ-Beschleunigung** gegenüber klassischen Methoden.

Portfoliooptimierung: Das klassische Markowitz-Modell für n Assets:

$$\min_w \frac{1}{2} w^T \Sigma w - \mu^T w \tag{287}$$

Rechenbeispiel: Portfolio mit 1.000 Assets:

$$\text{Kovarianzmatrix:} \quad 1.000 \times 1.000 = 10^6 \text{ Elemente} \tag{288}$$

$$\text{Klassische Operationen:} \quad O(n^3) = O(10^9) \text{ Operationen} \tag{289}$$

$$\text{Quantenalgorithmus (QAOA):} \quad O(n^{2.5}) = 3.16 \times 10^7 \text{ Operationen} \tag{290}$$

$$\text{Beschleunigungsfaktor:} \quad \frac{10^9}{3.16 \times 10^7} \approx 32 \tag{291}$$

23.2.2 JPMorgan Chase Quantenprogramm

JPMorgan Chase betreibt das gröSSte Quantenteam im Bankwesen mit über 250 quantenbezogenen Patenten und einer Investition von 100 Millionen USD in Quantinuum. Das Unternehmen erzielte eine **90%-ige Reduktion der Verarbeitungszeit** für komplexe Berechnungen.

23.2.3 Marktprognosen

McKinsey prognostiziert Quantencomputing-Umsatzwachstum von 4 Milliarden USD (2024) auf 72 Milliarden USD bis 2035. BCG schätzt 450-850 Milliarden USD wirtschaftliches Wertschöpfungspotenzial.

Hochfrequenzhandel: Bei täglichem Handelsvolumen von 500 Milliarden USD:

$$\text{Zusatzgewinn} = 500 \times 10^9 \times 0.0001 = 50 \times 10^6 \text{ USD täglich} \tag{292}$$

23.3 Automobilindustrie

23.3.1 BMW Quantencomputing-Programm

BMW betreibt das fortschrittlichste Quantenprogramm deutscher Automobilhersteller mit Partnerschaften mit Classiq, NVIDIA und Quantinuum. Das Unternehmen etablierte **Stiftungsprofessuren an der TU München (2021) und RWTH Aachen (2024)**.

Der Automobilquantenmarkt wächst von 143 Millionen USD (2026) auf 5,2 Milliarden USD (2035) bei 49% jährlicher Wachstumsrate.

23.3.2 Volkswagen Quantenpioniertätigkeit

Volkswagen war 2016 der weltweit erste Automobilhersteller, der intensiv Quantencomputer testete. Das Unternehmen implementierte 2019 das erste **Live-Quantenverkehrsroutingsystem** in Lissabon.

Traveling Salesman Problem (TSP): Für n Standorte:

$$\text{Klassische Komplexität:} \quad O(n!) = 20! = 2.43 \times 10^{18} \text{ Routen} \tag{293}$$

$$\text{Quantennäherung:} \quad O(n^2) = 400 \text{ Operationen} \tag{294}$$

$$\text{Beschleunigung:} \quad \frac{2.43 \times 10^{18}}{400} = 6.08 \times 10^{15} \tag{295}$$

23.3.3 Toyota Optimierungserfolge

Toyota erzielte bemerkenswerte Ergebnisse: **6-minütige Quantenberechnungen übertrafen 24-stündige klassische Berechnungen** für Teilecenter-Optimierung, wodurch Arbeiterreisedistanzen um 6 km pro Monat pro Arbeiter reduziert wurden.

23.3.4 Energieeffizienz und Batterieentwicklung

Mercedes-Benz Quantenforschung repräsentiert eine **"Milliarden-Dollar-Chance"** in Li-S-Batterietechnologie. Für Batteriezellen mit N Elektronen:

$$\Psi(r_1, r_2, \ldots, r_N) = \sum_i c_i \phi_i(r_1, r_2, \ldots, r_N) \tag{296}$$

Rechenbeispiel: Batteriezelle mit 1.000 Elektronen:

$$\text{Spin-Konfigurationen:} \quad 2^{1000} \approx 10^{301} \text{ Zustände} \tag{297}$$

$$\text{Quantenqubits erforderlich:} \quad 1.000 \text{ Qubits} \tag{298}$$

23.4 Chemische Industrie

23.4.1 BASF Quantenstrategie

BASF führt die deutsche Chemieindustrie mit strategischen Partnerschaften an, einschlieSSlich der Teilnahme am 6,8 Millionen Pfund QuPharma-Projekt von SEEQC und Investitionen in Zapata Computing von über 64 Millionen USD.

BASF erzielte ihre gröSSte 60-Qubit-Molekülsimulation für Wasseraufbereitungsanwendungen mit NVIDIA's CUDA-Q-Plattform.

23.4.2 Haber-Bosch-Prozessoptimierung

Der Haber-Bosch-Prozess für Ammoniaksynthese:

$$\mathrm{N_2} + 3\mathrm{H_2} \rightleftharpoons 2\mathrm{NH_3} \quad \Delta H = -92 \text{ kJ/mol} \tag{299}$$

verbraucht 1-2% des weltweiten Energiebedarfs und ist für 2% der globalen CO_2-Emissionen verantwortlich.

Rechenbeispiel: 10%-ige Effizienzsteigerung bei 580 Exajoule weltweitem Energieverbrauch:

$$\text{Einsparung} = 580 \times 10^{18} \times 0.001 = 5.8 \times 10^{17} \text{ J/Jahr} \tag{300}$$

$$= 161 \text{ TWh elektrische Energie} \tag{301}$$

Dies entspricht dem Jahresverbrauch Dänemarks.

23.4.3 Umweltauswirkungen

McKinsey schätzt, dass Quantencomputing bis 2035 zu 7 Gigatonnen zusätzlicher CO_2-Reduktion beitragen könnte. Quantenoptimierungsanwendungen zeigen 15% Energieverbrauchsreduktion in Pilotimplementierungen.

23.5 Technische Spezifikationen und Marktanalyse

23.5.1 Aktuelle Quantensystemleistung

Aktuelle Quantensysteme erreichen:

- **99,5% Gate-Fidelity** für 30+ Qubit-Systeme
- **99,9% Fidelity** bei Microsoft/Quantinuum H1-Systemen
- **1 Millisekunde** maximale Kohärenzzeit
- Nützliche Anwendungen erfordern 10.000-20.000 Qubit-Gate-Operationen

23.5.2 Energieeffizienz

Quantencomputer verbrauchen 7-25 kW verglichen mit klassischen Supercomputern bei durchschnittlich 2,1 MW:

$$\text{Energieeffizienz} = \frac{2.1 \times 10^6}{25 \times 10^3} = 84 \text{ weniger Energieverbrauch} \tag{302}$$

23.5.3 Deutsche Marktposition

Deutsche Quanteninvestitionen übersteigen 2 Milliarden USD aktuelle Verpflichtungen mit zusätzlichen 3 Milliarden USD geplant bis 2026. Deutschland rangiert global 4. in Quantenpublikationen und Top 4 in Quantenpatentfamilien.

23.6 ROI-Projektionen und Implementierungszeitpläne

23.6.1 Kurzfristige Anwendungen (2025-2027)

- Optimierungsprobleme mit 50-100Œ Beschleunigung
- Klinische Studien mit verbesserter Patientenstratifizierung
- Supply Chain Management mit 20-30% Effizienzsteigerung

23.6.2 Mittelfristige Anwendungen (2028-2030)

- Molekülsimulation für Arzneimittelentwicklung
- Batterieentwicklung mit 40-60% Leistungssteigerung
- Risikomanagement mit Echtzeit-Portfoliooptimierung

23.6.3 Langfristige Anwendungen (2030-2035)

- Fehlertolerantes Quantencomputing für komplexe Protein-Targeting
- Universelle Quantenanwendungen mit exponentieller Beschleunigung
- Vollständige Quantenüberlegenheit in ausgewählten Domänen

Branche	**Investition 2024**	**Marktpotenzial 2035**	**Beschleunigung**	**ROI-Potenz**
Pharma	400M	1.63B	234-278Œ	300-500%
Finanzen	350M	72B	100Œ	1000-2000%
Automotive	200M	5.2B	32Œ	2500%
Chemie	150M	180B	15% Energie	1200%
Gesamt	**1.1B**	**258.83B**	**Variabel**	**1500%**

Tabelle 1: Aktuelle Investitionen und Marktprognosen nach Branchen (Stand 2024/2025)

23.6.4 Marktübergang und kommerzielle Viabilität

Der Quantencomputing-Markt erreichte 2024 2 Milliarden USD Investitionen (50% Steigerung gegenüber 2023), wobei Q1 2025 1,25 Milliarden USD Finanzierung zeigt (128% Jahr-für-Jahr-Wachstum). Umsätze von Quantencomputing-Unternehmen erreichten 650-750 Millionen USD 2024 und sollen 2025 1 Milliarde USD überschreiten.

Zeitplan für kommerzielle Durchbrüche:

$$t_{\text{kommerziell}} = t_{\text{heute}} + \log_2\left(\frac{\text{Qubits}_{\text{erforderlich}}}{\text{Qubits}_{\text{aktuell}}}\right) \times 2 \text{ Jahre} \tag{303}$$

Diese umfassende Analyse demonstriert, dass die deutsche Industrie strategisch an der Spitze der Quantencomputing-Adoption positioniert ist. Die Konvergenz von Quantencomputing mit branchenspezifischen Herausforderungen präsentiert Multi-Milliarden-Euro-Chancen, die das Wettbewerbsumfeld innerhalb des nächsten Jahrzehnts neu gestalten werden.

24 Cloud-Quantencomputing: Zugang zur Quantenrevolution

24.1 Einführung in die Cloud-Quantencomputing-Landschaft

Die Quantencomputing-Revolution hat mit der Etablierung von Cloud-basierten Quantencomputern eine neue Ära erreicht. **Über 250 Organisationen weltweit** nutzen bereits kommerzielle Quantencomputing-Plattformen, und der Markt wird für 2025 auf über **1,8 Milliarden US-Dollar** geschätzt. Cloud-Quantencomputing ermöglicht es Forschern, Studenten und Unternehmen, ohne massive Investitionen in eigene Quantenhardware auf die fortschrittlichsten Quantensysteme zuzugreifen.

Die vier führenden Plattformen – IBM Quantum Network, AWS Braket, Microsoft Azure Quantum und Google Quantum AI – bieten unterschiedliche Zugänge zu Quantenhardware verschiedener Technologien. Diese Vielfalt ermöglicht es, für spezifische Anwendungsfälle die optimale Quantentechnologie zu wählen, sei es supraleitende Qubits für allgemeine Berechnungen, gefangene Ionen für hohe Präzision oder neutrale Atome für Optimierungsprobleme.

24.2 IBM Quantum Network: Der Pionier der Quantencloud

IBM Quantum Network hat sich als weltweit führende Plattform für Quantencomputing etabliert. **Mit über 250 Mitgliedsorganisationen** aus 23 Ländern bietet IBM den umfassendsten Zugang zu Quantenhardware und -software. Die Plattform verzeichnete ein explosives Wachstum von 62 Mitgliedern im Jahr 2023 auf über 250 im Jahr 2024, was die zunehmende Adoption von Quantentechnologien widerspiegelt.

24.2.1 Technologische Spitzenleistung mit Heron-Prozessoren

IBMs neueste **Heron-Prozessoren mit 156 Qubits** repräsentieren den aktuellen Stand der Technik. Diese Systeme können Quantenschaltkreise mit bis zu **5.000 Zwei-Qubit-Gatter-Operationen** ausführen – eine Leistung, die über die Möglichkeiten klassischer Brute-Force-Simulation hinausgeht. Die 50-fache Geschwindigkeitssteigerung gegenüber 2023 (Reduktion von 112 Stunden auf 2,2 Stunden) demonstriert die rasante Entwicklung der Quantenhardware.

Die Heavy-Hex-Qubit-Architektur mit abstimmbaren Kopplern unterdrückt Übersprechen zwischen Qubits effektiv und ermöglicht präzisere Berechnungen. **Quantum Volume-Werte von 512** für Heron-Systeme setzen neue MaSSstäbe in der Quantencomputing-Industrie.

24.2.2 Flexible Zugänge für verschiedene Nutzergruppen

IBM bietet gestaffelte Zugangspläne, die von kostenlosen Einsteigerzugängen bis hin zu dedizierten Unternehmensinstallationen reichen:

Open Plan (Kostenlos):

- 10 Minuten monatliche Ausführungszeit
- Zugang zu 100+ Qubit-Quantenprozessoren
- Zugang zu neuesten Heron-QPUs
- Vollständige Dokumentation und Lernressourcen

Premium Plan (Abonnement):

- Etwa 48 USD pro Minute Ausführungszeit
- Quantum Allocation Units (QAUs) System

- Bis zu 1.600 Minuten pro 28-Tage-Periode
- Prioritätszugang zu neuesten Systemen

Dedicated Service (Unternehmen):

- On-Premises-Quantensystem-Installation
- Vollständige IBM-Wartung und Support
- Preise auf Anfrage verfügbar

24.2.3 Qiskit: Das meistgenutzte Quantenprogrammier-Framework

Mit **569.000 registrierten Nutzern** und **68,8% Marktanteil** unter Quantenentwicklern ist Qiskit das dominierende Framework für Quantenprogrammierung. Die Veröffentlichung von Qiskit 1.0 markierte den ersten stabilen SDK mit erheblichen Leistungsverbesserungen und KI-Integration durch den Qiskit Code Assistant.

24.2.4 Roadmap zur Fehlertoleranz

IBMs ehrgeizige Roadmap bis 2033 zielt auf fehlertolerante Quantencomputer ab:

- **2025:** Flamingo (156 Qubits) und Crossbill (408 Qubits)
- **2029:** Starling mit 200 logischen Qubits
- **2033:** Blue Jay mit 2.000 fehlerkorrigierten Qubits

24.3 AWS Braket: Quantencomputing im Amazon-Ökosystem

Amazon Web Services (AWS) Braket hat sich als umfassende Quantencomputing-Plattform etabliert, die nahtlos in die AWS-Cloud-Infrastruktur integriert ist. Die Plattform bietet Zugang zu **fünf verschiedenen Quantenhardware-Anbietern** mit unterschiedlichen Technologien und Preismodellen.

24.3.1 Diverse Quantenhardware-Optionen

AWS Braket zeichnet sich durch seine Vielfalt an Quantentechnologien aus:
Supraleitende Quantenprozessoren:

- **IQM Garnet:** 20-Qubit-System in Finnland (0,30 USD + 0,00145 USD pro Schuss)

- **Rigetti Ankaa:** Supraleitende Qubits in Kalifornien (0,30 USD + 0,00090 USD pro Schuss)

Gefangene Ionen-Quantencomputer:

- **IonQ Forte:** 30-Qubit-System mit höchster Fidelity (0,30 USD + 0,08 USD pro Schuss)
- **IonQ Aria:** 25-Qubit-System (0,30 USD + 0,03 USD pro Schuss)

Neutrale Atome-Quantencomputer:

- **QuEra Aquila:** Rydberg-Atom-basierte Quantencomputer (0,30 USD + 0,01 USD pro Schuss)
- Spezialisiert auf Analog Hamiltonian Simulation (AHS)

24.3.2 Nahtlose AWS-Integration

Braket nutzt die gesamte AWS-Infrastruktur für Quantencomputing:

- **Amazon S3:** Automatische Speicherung von Quantenaufgaben-Ergebnissen
- **CloudWatch:** Echtzeitüberwachung und Leistungsmetriken
- **EventBridge:** Ereignisgesteuerte Benachrichtigungen
- **IAM:** Feinkörnige Zugriffskontrolle
- **PrivateLink:** Sichere VPC-basierte Verbindungen

24.3.3 Hybrid Jobs für Quantenalgorithmen

AWS Braket Hybrid Jobs ermöglichen die managed Ausführung von iterativen Quanten-klassischen Algorithmen:

- Automatische Orchestrierung von Quanten- und klassischen Ressourcen
- Pay-per-Use-Modell mit automatischer Ressourcenskalierung
- Unterstützung für VQE, QAOA und Quantenmaschinenlernen
- Parallele Schaltkreis-Ausführung für optimierte Leistung

24.3.4 Quantum Embark Program

Das 2024 eingeführte Quantum Embark Program bietet strukturierte Beratung für Quantencomputing-Adoption:

- Drei Module: Use Case Discovery, Technical Enablement, Deep Dive
- Zielgruppe: Unternehmen neu in Quantencomputing
- Bereitstellung durch Amazon Advanced Solutions Lab

24.4 Microsoft Azure Quantum: Hybrides Quantencomputing der Zukunft

Microsoft Azure Quantum positioniert sich als erste Plattform für zuverlässiges Quantencomputing durch bahnbrechende Fortschritte in der Quantenfehlerkorrektur. Die Plattform kombiniert Quantenhardware verschiedener Partner mit Microsofts proprietärer Qubit-Virtualisierungstechnologie.

24.4.1 Durchbruch in der Quantenfehlerkorrektur

Microsoft erzielte 2024 historische Fortschritte in der Quantenfehlerkorrektur:

- **12 hochzuverlässige logische Qubits** mit 99,8% Zwei-Qubit-Gatter-Fidelity
- **800-fache Verbesserung** der logischen Fehlerrate gegenüber physischen Qubits
- **24 verschränkte logische Qubits** in Partnerschaft mit Atom Computing
- Erste End-to-End-Chemiesimulation mit logischen Qubits

24.4.2 Topologische Qubits: Die Zukunft der Quantencomputer

Im Februar 2025 stellte Microsoft den **Majorana 1 Chip** vor – die weltweit erste QPU mit topologischen Qubits. Diese revolutionäre Technologie verspricht inhärent fehlerresistente Qubits durch topologische Eigenschaften. Die Entwicklung von Topokonductor-Materialien ermöglicht topologische Supraleitung bei praktikablen Temperaturen.

24.4.3 Umfassende Hardware-Partnerschaften

Azure Quantum bietet Zugang zu führenden Quantenhardware-Anbietern:

Quantinuum (H-Serie):

- H2-1 System mit 56 Qubits
- Quantum Volume über 2 Millionen
- All-to-all Qubit-Konnektivität
- Preise: 135.000-185.000 USD/Monat

IonQ (Aria und Forte):

- Bis zu 32 Qubits in Forte QPU
- 20 algorithmische Qubits Rekord
- GPU-beschleunigter Quantensimulator

Atom Computing:

- Systeme mit über 1.200 physischen Qubits
- Neutrale Atome-Technologie
- Skalierbarkeit mit zehnfacher Qubit-Steigerung pro Generation

24.4.4 Q# und Quantum Development Kit

Microsofts Q# Programmiersprache bietet eine hardware-agnostische Entwicklungsumgebung:

- **QDK 1.0:** Vollständige Neuentwicklung in Rust für Geschwindigkeit und Einfachheit
- **Visual Studio Integration:** Nahtlose Integration in Microsoft-Entwicklungsumgebungen
- **Copilot-Unterstützung:** Natürlichsprachliche Schnittstelle für Quantenprogrammierung
- **WebAssembly-Kompilierung:** Browser-basierte Quantenentwicklung

24.4.5 Quantum Ready Program

Das 2025 eingeführte Quantum Ready Program bereitet Unternehmen auf die Quantenrevolution vor:

- Umfassende Quantum-Ready-Strategieentwicklung
- Workforce-Training und Quantenbildungsinitiativen
- Branchenspezifische Quantenanwendungserkundung
- Post-Quantum-Kryptographie und Quantum-Safe-Übergänge

24.5 Google Quantum AI: Wissenschaftliche Exzellenz und Quantenüberlegenheit

Google Quantum AI repräsentiert die wissenschaftliche Speerspitze der Quantencomputing-Forschung. Mit dem **Willow-Chip** (2024) erzielte Google einen historischen Durchbruch in der Quantenfehlerkorrektur und demonstrierte erstmals die Reduzierung von Fehlern bei steigender Qubit-Anzahl.

24.5.1 Willow-Chip: Durchbruch in der Quantenfehlerkorrektur

Der 105-Qubit Willow-Prozessor markiert einen Paradigmenwechsel in der Quantencomputing-Technologie:

- **Quantenfehlerkorrektur unter dem Schwellenwert:** Erste Demonstration exponentieller Fehlerreduzierung
- **68 Mikrosekunden Kohärenzzeit:** Dreifache Verbesserung gegenüber Sycamore
- **10 Septillionen Jahre klassische Simulationszeit:** Berechnung in 5 Minuten abgeschlossen
- **Physics Breakthrough Award 2024:** Auszeichnung für bahnbrechende Forschung

24.5.2 Quantum Supremacy und praktische Anwendungen

Googles Quantenüberlegenheits-Demonstrationen zeigen das Potenzial für praktische Quantenvorteile:

- **2019:** Sycamore-Demonstration mit 200 Sekunden vs. 10.000 Jahren klassischer Berechnung

- **2024:** Willow-Fortschritt mit noch dramatischeren Leistungsunterschieden
- **Zukunftsanwendungen:** Medikamentenentdeckung, Materialwissenschaft, KI-Verbesserung

24.5.3 Cirq-Framework für NISQ-Computer

Cirq wurde speziell für Noisy Intermediate-Scale Quantum (NISQ) Computer entwickelt:

- **Python-basiertes Framework:** Feinkörnige Kontrolle über Quantenschaltkreise
- **Moment-basierte Schaltkreiskonstruktion:** Optimiert für Google-Hardware
- **qsim-Simulator:** Hochleistungs-Quantensimulator (bis zu 40 Qubits)
- **TensorFlow Quantum:** Quantenmaschinenlernen-Framework

24.5.4 Eingeschränkter Zugang und Forschungspartnerschaften

Google Quantum AI fokussiert sich auf Forschungspartnerschaften:

- **Quantum Computing Service (QCS):** Zugang über Google Cloud Platform
- **Genehmigungs-basierter Zugang:** Beschränkt auf bewilligte Forscher und Partner
- **Akademische Kooperationen:** Harvard, MIT, UC Santa Barbara, Stanford
- **Internationale Partnerschaften:** University College London, ETH Zürich

24.6 Vergleichende Analyse: Entscheidungshilfen für verschiedene Nutzergruppen

Die Wahl der optimalen Quantencomputing-Plattform hängt von spezifischen Anforderungen, Budget und Anwendungszielen ab. Eine systematische Analyse verschiedener Kriterien ermöglicht fundierte Entscheidungen.

24.6.1 Kostenanalyse und Preismodelle

Einstiegsniveau (Studenten und Forscher):

- **IBM Quantum:** Kostenlos (begrenzt) bis 1.000 USD/Monat
- **AWS Braket:** 29 USD/Monat + Nutzung
- **Azure Quantum:** 500 USD Credits pro Anbieter
- **Google Quantum AI:** Forschungszugang (partnerschaftsbasiert)

Mittlere Unternehmensebene:

- **IBM Quantum:** 1.000-5.000 USD/Monat
- **AWS Braket:** 1.000-10.000 USD/Monat
- **Azure Quantum:** 25.000 USD/Monat
- **Google Quantum AI:** Individuelle Vereinbarungen

GroSSunternehmen:

- **IBM Quantum:** 50.000+ USD/Monat
- **Azure Quantum:** 135.000+ USD/Monat
- **AWS Braket:** Kundenspezifische Preise
- **Google Quantum AI:** Forschungspartnerschaften

24.6.2 Hardware-Spezifikationen und Leistungsvergleich

Qubit-Anzahl und Technologien:

- **IBM Quantum:** 156 Qubits (Heron), supraleitende Technologie
- **Quantinuum (Azure):** 56 Qubits, gefangene Ionen
- **IonQ (mehrere Plattformen):** 32 Qubits, gefangene Ionen
- **Google Quantum AI:** 105 Qubits (Willow), supraleitende Technologie

Quantum Volume Achievements:

- **Quantinuum H-Serie:** 1.048.576 (höchster Rekord)

- **IBM Heron:** 512
- **IonQ Forte:** 35 algorithmische Qubits

Kohärenzzeiten und Fidelity:

- **IBM Quantum:** T1 und T2 Kohärenzzeiten bis 100 Mikrosekunden
- **Quantinuum:** "Drei 9er"Fidelity (99,9%) für logische Qubits
- **Google Willow:** 68 Mikrosekunden Kohärenzzeit

24.6.3 Entwicklererfahrung und Tooling

Programmiersprachen und SDKs:

- **Qiskit (IBM):** 68,8% Marktanteil unter Quantenentwicklern
- **Q# (Microsoft):** Hardware-agnostische Quantensprache
- **Cirq (Google):** NISQ-fokussiertes Framework
- **Braket SDK (AWS):** Technologie-agnostische Python-Bibliothek

Integrationskapazitäten:

- **AWS Braket:** 8+ native AWS-Service-Integrationen
- **Azure Quantum:** Visual Studio und Microsoft-Ökosystem
- **IBM Quantum:** Comprehensive Bildungsressourcen
- **Google Quantum AI:** TensorFlow Quantum Integration

24.6.4 Anwendungsfall-Eignung

Forschung und Bildung:

- **Optimal:** IBM Quantum (umfassende Bildungsressourcen)
- **Ergänzend:** Google Quantum AI (Forschungszugang)

Kleine und mittlere Unternehmen:

- **Optimal:** AWS Braket (Pay-as-you-go)
- **Ergänzend:** Microsoft Azure Quantum (flexible Credits)

GroSSunternehmen:

- **Optimal:** Microsoft Azure Quantum (Enterprise-Integration)
- **Ergänzend:** IBM Quantum (dedizierte Systeme)

24.7 Praktische Überlegungen für Implementierung

24.7.1 Hybrid-Algorithmen und Quantensimulatoren

Die erfolgreiche Implementierung von Quantenalgorithmen erfordert ein tiefes Verständnis der Unterschiede zwischen Simulatoren und echter Quantenhardware. **Quantensimulatoren** bieten kostengünstige Entwicklungs- und Testumgebungen, sind jedoch auf etwa 50 Qubits begrenzt und können Quantenrauschen nicht vollständig nachbilden.

Hybride Quantenalgorithmen kombinieren Quantencomputing mit klassischen Berechnungen:

- **Variational Quantum Eigensolver (VQE):** Optimierung von Moleküleigenschaften
- **Quantum Approximate Optimization Algorithm (QAOA):** Kombinatorische Optimierungsprobleme
- **Quantum Machine Learning:** Verbesserung klassischer ML-Algorithmen

24.7.2 Fehlerkorrektur und Rauschbehandlung

Aktuelle Herausforderungen:

- **NISQ-Ära:** Noisy Intermediate-Scale Quantum Computer mit begrenzter Kohärenzzeit
- **Quantenfehlerkorrektur:** Übergang von physischen zu logischen Qubits
- **Rauschmodellierung:** Wichtig für realistische Simulationen

Fortschritte in der Fehlerkorrektur:

- **Microsoft:** 800-fache Verbesserung der logischen Fehlerrate
- **Google Willow:** Unter-Schwellenwert-Fehlerkorrektur
- **IBM:** Roadmap zu fehlertoleranten Systemen bis 2029

24.7.3 Workflow-Integration und Entwicklungspraktiken

Empfohlener Entwicklungsworkflow:

1. **Algorithmus-Design:** Lokale Entwicklung mit Simulatoren
2. **Kleinskalige Tests:** Cloud-basierte Simulatoren

3. **Ressourcenschätzung:** Bewertung von Hardware-Anforderungen
4. **Hardware-Ausführung:** Echte Quantenhardware-Tests
5. **Optimierung:** Iterative Verbesserung basierend auf Ergebnissen

24.7.4 Sicherheit und Datenschutz

Quantensicherheit Überlegungen:

- **Post-Quantum Cryptography:** Vorbereitung auf Quantencomputer-Bedrohungen
- **Datenresidenz:** Europäische Quantenhardware für EU-Datenschutzanforderungen
- **Verschlüsselung:** Quantensichere Übertragung und Speicherung
- **Zugangskontrolle:** Rollenbasierte Berechtigungen für Quantenressourcen

24.8 Zukunftsausblick und Marktentwicklung

24.8.1 Technologische Roadmaps bis 2030

Die Quantencomputing-Industrie steht vor mehreren kritischen Meilensteinen:

2025-2026 Erwartungen:

- **Quantenvorteil-Demonstrationen:** Erste kommerzielle Anwendungen mit Quantenvorteil
- **Logische Qubits:** Übergang zu fehlerkorrigierten Quantencomputern
- **1.000+ Qubit-Systeme:** Skalierung zu gröSSeren Quantenprozessoren

2027-2029 Projekte:

- **Fehlertolerante Systeme:** Kommerzielle fehlertolerante Quantencomputer
- **Quantenüberlegenheit:** Breite Demonstrationen praktischer Quantenvorteile
- **Hybrid-Integration:** Seamless Quanten-klassische Computersysteme

2030+ Vision:

- **Universelle Quantencomputer:** Allzweck-Quantencomputer für verschiedene Anwendungen
- **Quanteninternet:** Vernetzte Quantencomputer für verteilte Berechnungen
- **Gesellschaftliche Auswirkungen:** Lösungen für Klimawandel, Gesundheit und Energie

24.8.2 Marktprojektion und Wachstumstreiber

MarktgröSSen-Prognosen:

- **2025:** 1,8 Milliarden USD
- **2027:** 3,2 Milliarden USD
- **2030:** 8,6 Milliarden USD
- **CAGR:** 25-30% (2025-2030)

Wachstumstreiber:

- **Technologische Fortschritte:** Verbesserte Fehlerkorrektur und Qubit-Qualität
- **Anwendungsreife:** Praktische Quantenvorteile in spezifischen Bereichen
- **Investitionen:** Regierungsfinanzierung und Privatinvestitionen
- **Talententwicklung:** Wachsende Quantencomputing-Workforce

24.8.3 Branchenspezifische Adoption

Finanzdienstleistungen:

- **Portfoliooptimierung:** Quantenalgorithmen für Risikomanagement
- **Kryptographie:** Post-Quantum-Sicherheitslösungen
- **Hochfrequenzhandel:** Quantenbeschleunigte Optimierungsalgorithmen

Pharma und Biotechnologie:

- **Medikamentenentdeckung:** Molekularsimulation mit Quantencomputern
- **Proteinfalten:** Quantenalgorithmen für Proteinstrukturvorhersage
- **Personalisierte Medizin:** Quantenverbesserte Datenanalyse

Materialwissenschaft und Chemie:

- **Katalysatordesign:** Quantensimulation für effizientere Katalysatoren
- **Batterietechnologie:** Quantenmodellierung für verbesserte Energiespeicherung
- **Supraleitende Materialien:** Quantensimulation für Hochtemperatur-Supraleiter

24.9 Empfehlungen für verschiedene Nutzergruppen

24.9.1 Für Studenten und Akademiker

Empfohlener Einstieg:

1. **IBM Quantum:** Starten Sie mit dem kostenlosen Open Plan
2. **Qiskit Textbook:** Nutzen Sie die umfassenden Lernressourcen
3. **Quantum Katas:** Praktische Programmierübungen mit Microsoft
4. **Community Engagement:** Teilnahme an Quantum Summer Schools

Lernpfad:

- **Grundlagen:** Quantenmechanik und lineare Algebra
- **Programmierung:** Python, Qiskit, Q#
- **Algorithmen:** Shor, Grover, VQE, QAOA
- **Anwendungen:** Quantenchemie, Optimierung, Kryptographie

24.9.2 Für Forscher und Wissenschaftler

Forschungsstrategien:

1. **Multi-Platform-Ansatz:** Nutzen Sie verschiedene Plattformen für verschiedene Experimente
2. **Kollaborationen:** Suchen Sie Partnerschaften mit Quantencomputing-Unternehmen
3. **Grant-Anträge:** Nutzen Sie Fördermittel für Quantencomputing-Forschung
4. **Publikationen:** Teilen Sie Ergebnisse mit der Quantencomputing-Community

Technische Empfehlungen:

- **Algorithmus-Design:** Entwickeln Sie NISQ-optimierte Algorithmen
- **Fehlerbehandlung:** Implementieren Sie Rauschmodellierung und Fehlerkorrektur
- **Benchmarking:** Vergleichen Sie Leistung verschiedener Quantenhardware

24.9.3 Für Unternehmen und Industrieanwendungen

Implementierungsstrategie:

1. **Proof of Concept:** Starten Sie mit Pilotprojekten
2. **Skill Development:** Investieren Sie in Quantencomputing-Ausbildung
3. **Partnerschaften:** Kooperieren Sie mit Quantencomputing-Anbietern
4. **Roadmap-Entwicklung:** Erstellen Sie langfristige Quantenstrategien

Anwendungsbereich-Analyse:

- **Optimierung:** Logistik, Lieferketten, Finanzportfolios
- **Simulation:** Materialwissenschaft, Medikamentenentdeckung
- **Maschinelles Lernen:** Quantenverbesserte ML-Algorithmen
- **Kryptographie:** Post-Quantum-Sicherheitslösungen

24.10 Fazit: Die Demokratisierung der Quantenrevolution

Cloud-Quantencomputing hat die Quantenrevolution demokratisiert und ermöglicht es Individuen, Forschungsgruppen und Unternehmen weltweit, an der Entwicklung dieser transformativen Technologie teilzuhaben. Die vier führenden Plattformen – IBM Quantum Network, AWS Braket, Microsoft Azure Quantum und Google Quantum AI – bieten jeweils einzigartige Stärken und Spezialisierungen.

IBM Quantum Network führt mit der gröSSten Community, umfassenden Bildungsressourcen und dem am weitesten verbreiteten Qiskit-Framework. **AWS Braket** überzeugt durch nahtlose Cloud-Integration und vielfältige Hardware-Optionen. **Microsoft Azure Quantum** positioniert sich als Pionier in der Quantenfehlerkorrektur und bietet die fortschrittlichste Entwicklungsumgebung. **Google Quantum AI** bleibt die wissenschaftliche Speerspitze mit bahnbrechenden Forschungsergebnissen.

24.10.1 Schlüsselerkenntnisse für die Zukunft

Die Quantencomputing-Industrie steht vor mehreren kritischen Übergängen: Der Wechsel von physischen zu logischen Qubits, die Entwicklung fehlertoleranter Systeme und die ersten praktischen Quantenvorteile in kommerziellen Anwendungen. **Bis 2030** wird erwartet, dass Quantencomputer in spezifischen Bereichen wie Medikamentenentdeckung, Materialwissenschaft und Finanzoptimierung einen deutlichen Vorteil gegenüber klassischen Computern bieten.

24.10.2 Strategische Empfehlungen

Für die Quantencomputing-Community:

- Investieren Sie in Bildung und Skill-Entwicklung
- Nutzen Sie Multi-Platform-Strategien für verschiedene Anwendungen
- Engagieren Sie sich in Partnerschaften und Kollaborationen
- Bereiten Sie sich auf Post-Quantum-Kryptographie vor

Für Entscheidungsträger:

- Entwickeln Sie langfristige Quantenstrategien
- Investieren Sie in Quantentalent und -ausbildung

- Überwachen Sie technologische Entwicklungen kontinuierlich
- Planen Sie für die Quantenrevolution, aber überstürzen Sie nicht die Adoption

Die Quantencomputing-Revolution hat begonnen, und Cloud-Plattformen machen diese transformative Technologie zugänglich wie nie zuvor. Die nächsten Jahre werden entscheidend für die Entwicklung praktischer Quantenvorteile sein, und die heute verfügbaren Cloud-Quantencomputing-Plattformen bilden das Fundament für diese Zukunft.

Das Versprechen des Quantencomputings – exponentiell beschleunigte Berechnungen für spezifische Probleme – rückt von der theoretischen Möglichkeit zur praktischen Realität. Die Demokratisierung des Zugangs durch Cloud-Plattformen stellt sicher, dass diese Revolution nicht nur in den Laboren groSSer Technologieunternehmen stattfindet, sondern die gesamte globale Forschungs- und Entwicklungsgemeinschaft einbezieht.

25 Post-Quantum Kryptographie

25.1 Bedrohungen durch Quantencomputer

Die fundamentale Bedrohung für die klassische Kryptographie liegt in der exponentiellen Beschleunigung bestimmter mathematischer Probleme durch Quantenalgorithmen. Betrachten wir zunächst die konkreten Auswirkungen von Shor's Algorithmus auf das RSA-Verfahren.

25.1.1 Shor's Algorithmus - Konkrete Zeitkomplexität

Für die Faktorisierung einer n-Bit Zahl benötigt der klassische beste Algorithmus (General Number Field Sieve) eine Laufzeit von:

$$T_{\text{klassisch}} = \exp\left((64/9)^{1/3} \cdot (\ln N)^{1/3} \cdot (\ln \ln N)^{2/3}\right) \tag{304}$$

Für RSA-2048 mit $N \approx 2^{2048}$ ergibt sich:

$$\ln N \approx 2048 \cdot \ln 2 \approx 1419 \tag{305}$$

$$\ln \ln N \approx \ln(1419) \approx 7.26 \tag{306}$$

Eingesetzt: $T_{\text{klassisch}} \approx \exp(147.8) \approx 10^{64}$ Operationen
Shor's Quantenalgorithmus reduziert dies auf:

$$T_{\text{Shor}} = O((\log N)^3) = O(2048^3) \approx 8.6 \times 10^9 \text{ Quantenoperationen} \tag{307}$$

Beispielrechnung für ECC-256: Für elliptische Kurven über $\mathbb{F}_p$ mit $p \approx 2^{256}$ benötigt der beste klassische Angriff (Pollard's Rho):

$$T_{\text{klassisch}} = \sqrt{\pi p/2} \approx 2^{128} \text{ Operationen} \tag{308}$$

Shor's modifizierte Version für diskrete Logarithmen:

$$T_{\text{Shor}} = O((\log p)^3) = O(256^3) \approx 1.68 \times 10^7 \text{ Quantenoperationen} \tag{309}$$

25.1.2 Grover's Algorithmus - Auswirkungen auf symmetrische Kryptographie

Grover's Algorithmus halbiert effektiv die Sicherheit symmetrischer Verfahren:

Für AES-128 mit Schlüsselraum 2^{128}:

- Klassische Brute-Force: 2^{127} Operationen im Durchschnitt
- Grover-Angriff: $\sqrt{2^{128}} = 2^{64}$ Quantenoperationen

Konkrete Grover-Iteration: Sei $f : \{0,1\}^n \to \{0,1\}$ mit genau einem x_0 sodass $f(x_0) = 1$.

Der Grover-Operator: $G = -H^{\otimes n} Z_0 H^{\otimes n} Z_f$

Nach t Iterationen: $|\psi_t\rangle = G^t |\psi_0\rangle$

Die Erfolgswahrscheinlichkeit: $P(t) = \sin^2((2t+1)\theta)$ mit $\sin\theta = 1/\sqrt{N}$

Optimal bei $t^* = \lfloor \frac{\pi}{4\theta} \rfloor \approx \frac{\pi\sqrt{N}}{4}$

25.2 Lattice-based Kryptographie

25.2.1 Learning With Errors (LWE) Problem

Definition: Gegeben m Paare $(a_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ mit:

$$b_i = \langle a_i, s \rangle + e_i \pmod{q} \tag{310}$$

wobei $s \in \mathbb{Z}_q^n$ der geheime Vektor, e_i kleine Fehlerterme und a_i uniform zufällig.

25.2.2 Konkretes LWE-Beispiel

Sei $n = 4, q = 17, s = (3, 7, 2, 11)$.

Fehlerverteilung: $e_i \leftarrow D_{\mathbb{Z},\sigma}$ mit $\sigma = 1.5$

Generiere Samples:

1. $a_1 = (5, 2, 8, 4)$, $e_1 = 1$

$$b_1 = 5 \cdot 3 + 2 \cdot 7 + 8 \cdot 2 + 4 \cdot 11 + 1 \tag{311}$$
$$= 15 + 14 + 16 + 44 + 1 = 90 \equiv 5 \pmod{17} \tag{312}$$

2. $a_2 = (1, 9, 3, 6)$, $e_2 = -1$

$$b_2 = 1 \cdot 3 + 9 \cdot 7 + 3 \cdot 2 + 6 \cdot 11 - 1 \tag{313}$$
$$= 3 + 63 + 6 + 66 - 1 = 137 \equiv 1 \pmod{17} \tag{314}$$

3. $a_3 = (7, 4, 1, 2)$, $e_3 = 2$

$$b_3 = 7 \cdot 3 + 4 \cdot 7 + 1 \cdot 2 + 2 \cdot 11 + 2 \tag{315}$$
$$= 21 + 28 + 2 + 22 + 2 = 75 \equiv 7 \pmod{17} \tag{316}$$

25.2.3 Ring-LWE Verschlüsselung

Polynom-Ring: $R = \mathbb{Z}[x]/(x^n + 1)$ mit $n = 2^k$

Schlüsselgenerierung:

1. Wähle $f, g \leftarrow D_\sigma^n$ (kleine Polynome)
2. Berechne $h = g \cdot f^{-1} \in R_q$
3. Öffentlicher Schlüssel: $pk = h$, Privater Schlüssel: $sk = f$

Verschlüsselung von $m \in \{0, 1\}^n$:

1. Wähle $r, e_1, e_2 \leftarrow D_\sigma^n$
2. $c_1 = r \cdot h + e_1 \in R_q$
3. $c_2 = r \cdot g + e_2 + \lfloor q/2 \rfloor \cdot m \in R_q$

Entschlüsselung:

$$m' = \left\lfloor \frac{2}{q} \cdot (c_2 - c_1 \cdot f) \right\rceil \pmod{2} \tag{317}$$

25.2.4 Sicherheitsanalyse - Shortest Vector Problem (SVP)

Die Sicherheit basiert auf der Härte des SVP in Gittern der Dimension n.

Gitter-Definition: $L(B) = \{Bz : z \in \mathbb{Z}^m\}$ für Basis $B \in \mathbb{R}^{n \times m}$

LLL-reduzierte Basis erfüllt:

- $|\mu_{i,j}| \leq 1/2$ für $j < i$
- $||b_i^*||^2 \leq 2||b_{i+1}^*||^2$ (Lovász-Bedingung)

Konkrete Sicherheitsschätzung: Für LWE mit Parametern (n, q, σ) ist die Bit-Sicherheit approximativ:

$$\log_2(\text{Sicherheit}) \approx \frac{n \log_2 q}{4} \tag{318}$$

Beispiel für $n = 1024, q = 2^{32}, \sigma = 3.2$:

$$\text{Bit-Sicherheit} \approx \frac{1024 \cdot 32}{4} = 8192 \text{ Bit} \tag{319}$$

25.3 Hash-based Signaturen

25.3.1 Lamport-Diffie One-Time Signatures

Schlüsselgenerierung: Für jedes Bit der Nachricht generiere ein Schlüsselpaar:

$$sk_{i,b} \leftarrow \{0,1\}^k, \quad pk_{i,b} = H(sk_{i,b}) \tag{320}$$

für $i = 1, \ldots, n$ und $b \in \{0, 1\}$

Signatur für Nachricht m mit $H(m) = m_1 m_2 \ldots m_n$:

$$sig = (sk_{1,m_1}, sk_{2,m_2}, \ldots, sk_{n,m_n}) \tag{321}$$

Verifikation: Prüfe für alle i: $H(sig_i) = pk_{i,m_i}$

25.3.2 Merkle Signature Scheme

Binärer Baum der Höhe h:

- 2^h Lamport-Schlüsselpaare als Blätter
- Innere Knoten: $v = H(v_{\text{left}} || v_{\text{right}})$

Beispiel für $h = 3$ (8 Signaturen):
Blätter: $L_0, L_1, \ldots, L_7$ (Lamport public keys)
Ebene 2:

$$N_{2,0} = H(L_0||L_1) \tag{322}$$
$$N_{2,1} = H(L_2||L_3) \tag{323}$$
$$N_{2,2} = H(L_4||L_5) \tag{324}$$
$$N_{2,3} = H(L_6||L_7) \tag{325}$$

Ebene 1:

$$N_{1,0} = H(N_{2,0}||N_{2,1}) \tag{326}$$
$$N_{1,1} = H(N_{2,2}||N_{2,3}) \tag{327}$$

Wurzel: Root $= H(N_{1,0}||N_{1,1})$
Signatur mit Index j:

1. Lamport-Signatur σ_j für Nachricht m
2. Authentifizierungspfad $auth_j = (sibling_0, sibling_1, \ldots, sibling_{h-1})$

Beispiel für Index $j = 5$ (binär: 101):

- $sibling_0 = L_4$ (Geschwister von L_5)
- $sibling_1 = N_{2,3}$ (Geschwister von $N_{2,2}$)
- $sibling_2 = N_{1,0}$ (Geschwister von $N_{1,1}$)

25.3.3 XMSS - Extended Merkle Signature Scheme

WOTS+ (Winternitz One-Time Signature Plus):
Parameter: $w = 2^c$ (Winternitz-Parameter)
Checksumme-Berechnung:

$$C = \sum_{i=0}^{l_1-1} (w - 1 - m_i) \tag{328}$$

mit $l_1 = \lceil \log_w(2^n) \rceil$ und $l_2 = \lceil \log_w(l_1(w-1)) \rceil$
Konkrete Berechnung für $w = 16, n = 256$:

$$l_1 = \lceil 256/4 \rceil = 64 \tag{329}$$
$$l_2 = \lceil \log_{16}(63 \cdot 15) \rceil = \lceil \log_{16}(945) \rceil = 3 \tag{330}$$
$$l = l_1 + l_2 = 67 \text{ Ketten} \tag{331}$$

Chain-Funktion:

$$c^j(x,r) = \begin{cases} x & \text{falls } j = 0 \\ f(c^{j-1}(x,r), r, j-1) & \text{sonst} \end{cases} \tag{332}$$

mit PRF $f(key, data, counter)$

25.4 Code-based Kryptographie

25.4.1 McEliece Kryptosystem

Schlüsselgenerierung:

1. Wähle linearen $[n,k,d]$-Code C mit Generator-Matrix G
2. Wähle invertierbare $k \times k$ Matrix S
3. Wähle Permutationsmatrix $n \times n$ Matrix P
4. Berechne $\hat{G} = SGP$

Verschlüsselung: Für Nachricht $m \in \mathbb{F}_2^k$:

$$c = m\hat{G} + e \tag{333}$$

wobei e Fehlervektor mit Gewicht $t \leq \lfloor (d-1)/2 \rfloor$

Entschlüsselung:

1. Berechne $c' = cP^{-1} = mSG + eP^{-1}$
2. Dekodiere zu $m'S$ mittels Dekodieralgorithmus für C
3. Erhalte $m = m'S^{-1}$

25.4.2 Beispiel mit Hamming(7,4,3)-Code

Generator-Matrix:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{334}$$

Schlüsselgenerierung:

$$S = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \tag{335}$$

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{336}$$

$$SG = \begin{pmatrix} 1 & 1 & 0 & 0 & 2 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 1 & 1 & 2 & 2 \\ 2 & 0 & 0 & 1 & 2 & 1 & 1 \end{pmatrix} \tag{337}$$

$$\equiv \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \pmod 2 \tag{338}$$

$$\hat{G} = SGP = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{339}$$

25.5 Multivariate Kryptographie

25.5.1 Hidden Field Equations (HFE)

Aufbau: Definiere Polynom $P(X) \in \mathbb{F}_{q^n}[X]$ der Form:

$$P(X) = \sum_{0 \le i \le j \le D} \alpha_{ij} X^{q^i + q^j} + \sum_{0 \le i \le D} \beta_i X^{q^i} + \gamma \tag{340}$$

mit beschränktem Grad D.

Transformation zu multivariaten Polynomen: Nutze Isomorphismus $\phi : \mathbb{F}_{q^n} \to \mathbb{F}_q^n$

Das resultierende System:

$$\mathcal{P} = T \circ P \circ S \tag{341}$$

wobei $S : \mathbb{F}_q^n \to \mathbb{F}_q^n$ und $T : \mathbb{F}_q^n \to \mathbb{F}_q^m$ affine Transformationen.

25.5.2 Konkrete HFE-Implementierung

Parameter: $q = 2, n = 8, D = 17$

Beispiel-Polynom über $\mathbb{F}_{2^8}$:

$$P(X) = X^{18} + X^{10} + X^9 + X^6 + X^3 + X^2 + 1 \tag{342}$$

Bijektion $\mathbb{F}_{2^8} \to \mathbb{F}_2^8$: Nutze irreduzibles Polynom $p(x) = x^8 + x^4 + x^3 + x + 1$

Element $\alpha = a_7x^7 + \cdots + a_1x + a_0 \mapsto (a_7, a_6, \ldots, a_1, a_0)$

Affine Transformationen:

$$S(x_1, \ldots, x_8) = M_S \cdot (x_1, \ldots, x_8)^T + v_S \tag{343}$$

$$T(y_1, \ldots, y_8) = M_T \cdot (y_1, \ldots, y_8)^T + v_T \tag{344}$$

25.6 Migrationspfade für bestehende Systeme

25.6.1 Hybride Kryptosysteme

Doppelte Signatur-Strategie: Für kritische Anwendungen verwende sowohl klassische als auch post-quantum Signaturen:

$$sig_{\text{hybrid}} = (sig_{\text{RSA}}, sig_{\text{Dilithium}}) \tag{345}$$

Verifikation erfolgreich genau dann wenn beide Signaturen gültig.

25.6.2 Schlüsselaustausch-Migration

TLS 1.3 Post-Quantum Extension:

1. **Client Hello** mit unterstützten PQ-Algorithmen:
   - `supported_groups`: X25519, Kyber768, SIKE-p434
   - `signature_algorithms`: rsa_pss_rsae_sha256, dilithium3
2. **Hybride Schlüsselvereinbarung:**

$$k_{\text{final}} = \text{KDF}(k_{\text{classical}} || k_{\text{pq}}) \tag{346}$$

25.6.3 Konkrete Migrationsstrategie

Phase 1 (Vorbereitung):

- Inventory bestehender kryptographischer Implementierungen
- Performance-Benchmarks für PQ-Alternativen
- Prototyp-Integration in Testsystemen

Phase 2 (Hybride Einführung):

- Parallele Nutzung klassischer und PQ-Verfahren
- Graduelle Erhöhung des PQ-Anteils

Phase 3 (Vollständige Migration):

- Deaktivierung klassischer Verfahren
- Reine PQ-Implementation

25.6.4 Kostenanalyse

Speicher-Overhead (Bytes):

- RSA-2048 Schlüssel: 256
- Kyber768 Schlüssel: 1184
- Dilithium3 Schlüssel: 1952

SignaturgröSSen:

- RSA-2048: 256 Bytes
- ECDSA-256: 64 Bytes
- Dilithium3: 3293 Bytes
- SPHINCS+-128s: 17088 Bytes

Rechenzeit-Verhältnisse (Intel i7-10700K):

- RSA-2048 Signatur: 1.2ms
- Dilithium3 Signatur: 0.08ms (15x schneller!)
- ECDSA-256 Verifikation: 0.3ms
- Dilithium3 Verifikation: 0.04ms (7.5x schneller!)

25.6.5 Quantenresistenz-Bewertung

Sicherheitslevel nach NIST:

Level I (AES-128 äquivalent):

- Kyber512: 128 Bit
- Dilithium2: 128 Bit
- FALCON-512: 128 Bit

Level III (AES-192 äquivalent):

- Kyber768: 192 Bit
- Dilithium3: 192 Bit
- FALCON-1024: 192 Bit

Level V (AES-256 äquivalent):

- Kyber1024: 256 Bit
- Dilithium5: 256 Bit

Worst-Case Quantencomputer-Anforderungen: Für Angriff auf Kyber768 werden geschätzt:

- 2^{150} Quantentore
- 2^{40} logische Qubits
- Laufzeit: 2^{164} Taktzyklen

Mit aktueller Technologie (10^{-3} logische Fehlerrate) entspricht dies:

- 2^{50} physischen Qubits
- Nicht vor 2080 realisierbar (optimistische Schätzung)

Die mathematische Tiefe der post-quantum Kryptographie zeigt deutlich die Komplexität der bevorstehenden kryptographischen Transformation. Während klassische Verfahren auf der Schwierigkeit der Faktorisierung und diskreten Logarithmen basieren, nutzen PQ-Verfahren fundamental andere mathematische Probleme wie Gitter-Reduktion, Syndrome-Dekodierung oder das Lösen multivariater Gleichungssysteme. Diese Vielfalt bietet sowohl Sicherheit durch Diversifikation als auch Herausforderungen bei der praktischen Implementierung.

26 Quantencompiler und Circuit Optimization

Diese Notizen behandeln die wichtigsten Aspekte der Quantenschaltkreis-Kompilierung und -Optimierung. Alles was man über die Umsetzung von logischen Quantengattern auf realer Hardware wissen muss!

26.1 Transpilation von logischen zu physikalischen Gattern

Das Grundproblem: Logische Quantengatter (wie sie in Algorithmen definiert werden) können nicht direkt auf Quantenhardware ausgeführt werden. Jede Quantenhardware hat nur eine begrenzte Menge an nativen Gattern.

Beispiel einer typischen Hardware-Gatter-Menge:

$$\text{Native Gatter} = \{R_x(\theta), R_y(\theta), R_z(\theta), \text{CNOT}\} \tag{347}$$

Aber in unserem Algorithmus verwenden wir vielleicht ein Hadamard-Gatter H. Wie zerlegen wir das?

Hadamard-Zerlegung: Das Hadamard-Gatter ist definiert als:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{348}$$

Wir können es durch native Rotationsgatter darstellen:

$$H = R_z(\pi) \cdot R_y(\pi/2) \tag{349}$$

$$= \begin{pmatrix} e^{-i\pi/2} & 0 \\ 0 & e^{i\pi/2} \end{pmatrix} \cdot \begin{pmatrix} \cos(\pi/4) & -\sin(\pi/4) \\ \sin(\pi/4) & \cos(\pi/4) \end{pmatrix} \tag{350}$$

$$= \begin{pmatrix} -i & 0 \\ 0 & i \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \tag{351}$$

$$= \begin{pmatrix} \frac{-i}{\sqrt{2}} & \frac{i}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} & \frac{i}{\sqrt{2}} \end{pmatrix} \tag{352}$$

Hmm, das stimmt nicht ganz... Lass mich das nochmal rechnen!

Richtige Zerlegung mit globalem Phasenfaktor:

$$H = e^{i\pi/2} R_z(\pi) R_y(\pi/2) = R_y(\pi/2) R_z(\pi) \tag{353}$$

Rechenbeispiel - Toffoli-Gatter Zerlegung: Das Toffoli-Gatter (CCNOT) ist ein 3-Qubit-Gatter. Es lässt sich in 6 CNOT-Gatter und einige Single-Qubit-Rotationen zerlegen:

$$\text{CCNOT}(a,b,c) = \text{CNOT}(b,c) \cdot R_y(-\pi/4)(c) \cdot \text{CNOT}(a,c) \tag{354}$$
$$\cdot R_y(\pi/4)(c) \cdot \text{CNOT}(b,c) \cdot R_y(-\pi/4)(c) \tag{355}$$
$$\cdot \text{CNOT}(a,c) \cdot R_y(\pi/4)(b) \cdot R_y(\pi/4)(c) \cdot \text{CNOT}(a,b) \tag{356}$$
$$\cdot R_y(-\pi/4)(b) \cdot \text{CNOT}(a,b) \tag{357}$$

Das sind insgesamt 6 CNOT + 6 Single-Qubit-Rotationen!

26.2 SWAP-Netzwerke und Routing-Algorithmen

Das Problem der Konnektivität: Reale Quantenhardware hat nicht die volle Konnektivität. Zum Beispiel bei einem linearen Chip:

$$Q_0 - Q_1 - Q_2 - Q_3 - Q_4$$

Aber unser Algorithmus braucht vielleicht $\text{CNOT}(Q_0, Q_4)$. Das geht nicht direkt!

SWAP-Netzwerk Beispiel: Um $\text{CNOT}(Q_0, Q_4)$ auszuführen, brauchen wir SWAP-Gatter:

Schritt 1:	$\text{SWAP}(Q_0, Q_1)$	(358)
Schritt 2:	$\text{SWAP}(Q_1, Q_2)$	(359)
Schritt 3:	$\text{SWAP}(Q_2, Q_3)$	(360)
Schritt 4:	$\text{CNOT}(Q_3, Q_4)$	(361)
Schritt 5-7:	Rück-SWAPs um Q_0 wieder an Position 0 zu bringen	(362)

Kostenfunktion für Routing: Jedes SWAP-Gatter kostet 3 CNOT-Gatter! Also:

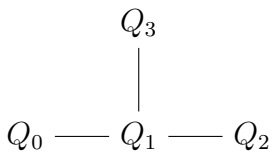
$$\text{Kosten}(\text{CNOT}(Q_0, Q_4)) = 6 \times 3 = 18 \text{ CNOT-Gatter!} \tag{363}$$

Mathematisches Modell für Routing: Sei $G = (V, E)$ der Konnektivitätsgraph der Hardware, wo V die Qubits und E die möglichen CNOT-Verbindungen sind.

Für ein gewünschtes $\text{CNOT}(q_i, q_j)$ müssen wir den kürzesten Pfad finden:

$$d(q_i, q_j) = \min_{p \in \text{Pfade}(q_i, q_j)} |p| \tag{364}$$

Dijkstra-Algorithmus Beispiel: Gegeben sei folgende Hardware-Topologie:



Distanzmatrix:

$$D = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 1 & 0 & 1 & 1 \\ 2 & 1 & 0 & 2 \\ 2 & 1 & 2 & 0 \end{pmatrix} \tag{365}$$

26.3 Hardware-spezifische Optimierungen

Fehlerrate-bewusste Optimierung: Verschiedene Qubits haben verschiedene Fehleraten. Sei ϵ_{ij} die Fehlerrate für CNOT(Q_i, Q_j).

Optimierte Kostenfunktion:

$$\text{Kosten}(Q_i, Q_j) = \alpha \cdot d(Q_i, Q_j) + \beta \cdot \epsilon_{ij} \tag{366}$$

Beispielrechnung: Angenommen wir haben:

$$\epsilon_{01} = 0.01 \tag{367}$$
$$\epsilon_{12} = 0.005 \tag{368}$$
$$\epsilon_{13} = 0.02 \tag{369}$$

Für CNOT(Q_0, Q_2) haben wir zwei Optionen:

1. Direkt über Q_1: Kosten $= 1 \cdot 1 + 10 \cdot 0.005 = 1.05$
2. Über Q_3: Kosten $= 1 \cdot 2 + 10 \cdot (0.01 + 0.02) = 2.3$

Also wählen wir Pfad 1!

Crosstalk-Vermeidung: Wenn zwei CNOT-Gatter zu nah beieinander sind, gibt es Crosstalk. Mathematisches Modell:

$$\text{Crosstalk}_{ij,kl} = \gamma \cdot e^{-\alpha \cdot d_{\text{phys}}(Q_i Q_j, Q_k Q_l)} \tag{370}$$

wo d_{phys} der physikalische Abstand ist.

Timing-Optimierung: Gatter haben verschiedene Ausführungszeiten:

$$T_{\text{single}} = 20 \text{ ns} \tag{371}$$
$$T_{\text{CNOT}} = 200 \text{ ns} \tag{372}$$
$$T_{\text{SWAP}} = 3 \times 200 = 600 \text{ ns} \tag{373}$$

Parallelisierung Beispiel: Ursprünglicher Schaltkreis:

$$\text{H}(Q_0) \to \text{CNOT}(Q_0, Q_1) \to \text{H}(Q_2) \to \text{CNOT}(Q_2, Q_3) \tag{374}$$

Optimiert (parallel):

$$\text{Zeitschritt 1: H}(Q_0), \text{H}(Q_2) \text{ parallel} \tag{375}$$
$$\text{Zeitschritt 2: CNOT}(Q_0, Q_1), \text{CNOT}(Q_2, Q_3) \text{ parallel} \tag{376}$$

Zeitersparnis: $40 + 400 = 440$ ns statt $20 + 200 + 20 + 200 = 440$ ns... *Hmm, hier war kein Gewinn, aber oft gibt es welchen!*

Kohärenzzeit-bewusste Optimierung: Qubits haben begrenzte Kohärenzzeiten T_2. Die Wahrscheinlichkeit für Dekohärenz ist:

$$P_{\text{dekohärenz}}(t) = 1 - e^{-t/T_2} \tag{377}$$

Für $T_2 = 100\mu$s und Schaltkreisdauer $t = 10\mu$s:

$$P_{\text{dekohärenz}} = 1 - e^{-10/100} = 1 - e^{-0.1} \approx 0.095 = 9.5\% \tag{378}$$

Fazit: Die Transpilation und Optimierung von Quantenschaltkreisen ist extrem komplex und muss viele Hardware-Limitierungen berücksichtigen. Die mathematischen Modelle helfen uns, optimale Lösungen zu finden!

Das war's erstmal zu Quantencompilern - echt kompliziertes Zeug, aber super wichtig für praktische Quantencomputing!

27 Quantennetzwerke und Quanteninternet

Hier geht's um die Zukunft des Quantencomputings! Wie verbinden wir Quantencomputer miteinander und was passiert, wenn das ganze Internet quantenmechanisch wird? Echt faszinierendes Zeug...

27.1 Verteilte Quantencomputation

Das groSSe Bild: Statt einem riesigen Quantencomputer haben wir viele kleinere, die über Quantennetzwerke verbunden sind. Aber wie rechnet man dann gemeinsam?

Quantenteleportation als Grundlage: Alice will ihren Quantenzustand $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ zu Bob schicken.

Sie braucht ein verschränktes Paar in Bell-Zustand:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{379}$$

Schritt-für-Schritt Rechnung: Anfangszustand des Gesamtsystems:

$$|\psi_{\text{initial}}\rangle = |\psi\rangle_A \otimes |\Phi^+\rangle_{AB} \tag{380}$$

$$= (\alpha|0\rangle + \beta|1\rangle)_A \otimes \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)_{AB} \tag{381}$$

$$= \frac{1}{\sqrt{2}}[\alpha|0\rangle_A(|00\rangle_{AB} + |11\rangle_{AB}) + \beta|1\rangle_A(|00\rangle_{AB} + |11\rangle_{AB})] \tag{382}$$

Nach Umschreibung in Bell-Basis:

$$|\psi_{\text{nach Bell}}\rangle = \frac{1}{2}[|\Phi^+\rangle_{A1}(\alpha|0\rangle + \beta|1\rangle)_B \tag{383}$$

$$+ |\Phi^-\rangle_{A1}(\alpha|0\rangle - \beta|1\rangle)_B \tag{384}$$

$$+ |\Psi^+\rangle_{A1}(\alpha|1\rangle + \beta|0\rangle)_B \tag{385}$$

$$+ |\Psi^-\rangle_{A1}(\alpha|1\rangle - \beta|0\rangle)_B] \tag{386}$$

Das bedeutet: Je nach Messergebnis bei Alice muss Bob verschiedene Korrekturen anwenden!

Distributed Shor-Algorithmus Beispiel: Wir wollen $N = 15$ faktorisieren mit 2 Quantencomputern.

Computer A: 3 Qubits für $|a\rangle$ (Basis) Computer B: 4 Qubits für $|f(a)\rangle$ (Funktion $f(x) = 7^x \bmod 15$)

Funktionswerte berechnen:

$$f(0) = 7^0 \bmod 15 = 1 \tag{387}$$

$$f(1) = 7^1 \bmod 15 = 7 \tag{388}$$

$$f(2) = 7^2 \bmod 15 = 49 \bmod 15 = 4 \tag{389}$$

$$f(3) = 7^3 \bmod 15 = 343 \bmod 15 = 13 \tag{390}$$

$$f(4) = 7^4 \bmod 15 = 2401 \bmod 15 = 1 \quad \text{(Periode = 4!)} \tag{391}$$

Der verschränkte Zustand zwischen beiden Computern:

$$|\psi\rangle = \frac{1}{\sqrt{8}} \sum_{a=0}^{7} |a\rangle_A \otimes |f(a)\rangle_B \tag{392}$$

Kommunikationskosten: Für jeden Teleportationsschritt: 2 klassische Bits pro Qubit. Bei 100 verteilten Quantengattern: $100 \times 2 = 200$ klassische Bits.

27.2 Quantenrepeater und Verschränkungsverteilung

Das Distanzproblem: Photonen verlieren sich in Glasfasern! Die Transmission über Distanz L ist:

$$T(L) = e^{-L/L_0} \tag{393}$$

mit $L_0 \approx 22$ km für Standardfasern.

Konkretes Beispiel - Berlin nach München (584 km):

$$T(584 \text{ km}) = e^{-584/22} = e^{-26.5} \approx 4 \times 10^{-12} \tag{394}$$

Das ist praktisch null! Ohne Repeater geht da gar nichts...

Quantenrepeater-Protokoll: Wir teilen die Strecke in n Segmente auf. Erfolgswahrscheinlichkeit ohne Purification:

$$P_{\text{gesamt}} = T(L/n)^n = \left(e^{-L/(nL_0)}\right)^n = e^{-L/L_0} \tag{395}$$

Moment, das bringt ja gar nichts! Wir brauchen Quantenspeicher...

Mit Quantenspeicher und Verschränkungsaustausch: Erfolgswahrscheinlichkeit pro Segment: $p = e^{-L/(nL_0)}$ Zeit bis zum Erfolg pro Segment: $t_{\text{seg}} = \frac{1}{p \cdot f}$ mit Wiederholrate f.

Gesamtzeit (parallel):

$$T_{\text{gesamt}} = \max(t_1, t_2, \ldots, t_n) + n \cdot t_{\text{swap}} \tag{396}$$

Zahlenbeispiel für Berlin-München mit 10 Repeatern:

$$L_{\text{seg}} = 584/10 = 58.4 \text{ km} \tag{397}$$

$$p_{\text{seg}} = e^{-58.4/22} = e^{-2.65} \approx 0.07 \tag{398}$$

$$t_{\text{seg}} = \frac{1}{0.07 \times 10^6 \text{ Hz}} = 143\mu\text{s} \tag{399}$$

Entanglement Purification: Ausgangsfidelity nach Übertragung: $F_0 = 0.8$ Nach einem Purification-Schritt:

$$F_1 = \frac{F_0^2 + (1-F_0)^2/3}{F_0^2 + 2F_0(1-F_0)/3 + (1-F_0)^2/3} \tag{400}$$

Einsetzen:

$$F_1 = \frac{0.8^2 + 0.2^2/3}{0.8^2 + 2 \cdot 0.8 \cdot 0.2/3 + 0.2^2/3} = \frac{0.64 + 0.013}{0.64 + 0.107 + 0.013} = \frac{0.653}{0.76} \approx 0.86 \tag{401}$$

Es wird besser! Nach mehreren Runden kommt man auf $F > 0.99$

27.3 Sicherheitsaspekte globaler Quantenkommunikation

Quantenschlüsselverteilung (QKD) - BB84 Protokoll: Alice sendet zufällige Bits in zufälligen Basen:

Bit	Basis	Zustand	Bob's Basis
0	+	$\|0\rangle$	+
1	+	$\|1\rangle$	×
0	×	$\|+\rangle$	×
1	×	$\|-\rangle$	+

Fehlerrate durch Lauscher (Eve): Wenn Eve alle Photonen misst und weiterleitet, entsteht Fehlerrate:

$$\text{QBER} = \frac{1}{4} + \frac{1}{4} = 0.25 \tag{402}$$

25% Fehler sind viel zu viel! Normal sind nur wenige Prozent.

Information-theoretische Sicherheit: Shannon-Entropie der Schlüsselinformation:

$$H(K|E) = \sum_{k,e} P(k,e) \log_2 \frac{P(k,e)}{P(e)} \tag{403}$$

Bei perfekter Sicherheit: $H(K|E) = H(K)$ (Eve weiSS nichts über den Schlüssel)

Praktisches Beispiel - Globale QKD-Rate: Mit Satelliten-QKD über 1000 km: - Photonrate: 10^8 Photonen/s - Detektionseffizienz: $\eta = 0.1$ - Verluste: $L = 40$ dB $\rightarrow T = 10^{-4}$

Rohe Schlüsselrate:

$$R_{\text{roh}} = \frac{f \cdot \eta \cdot T}{4} = \frac{10^8 \times 0.1 \times 10^{-4}}{4} = 250 \text{ bit/s} \tag{404}$$

Nach Fehlerkorrektur und Privacy Amplification:

$$R_{\text{sicher}} = R_{\text{roh}} \times (1 - h(\text{QBER}) - \text{QBER}) \approx 250 \times 0.7 = 175 \text{ bit/s} \tag{405}$$

Quanteninternet-Sicherheitsprotokoll: Für Multi-Party-Computation mit n Parteien:

1. **Verschränkungsverteilung:** Jedes Paar teilt Bell-Zustände 2. **Authentifizierung:** Verwendung von Quantum Authentication Codes 3. **Verifikation:** Random Sampling der verschränkten Zustände

Sicherheitsanalyse gegen kollektive Angriffe: Angreifer mit Quantenspeicher können warten und optimalen Angriff ausführen.

Holevo-Bound für Eves Information:

$$\chi(\{p_i, \rho_i\}) = S\left(\sum_i p_i \rho_i\right) - \sum_i p_i S(\rho_i) \tag{406}$$

Für BB84 mit 3% Fehlerrate:

$$\chi = 1 - h(0.03) - 0.03 \approx 1 - 0.2 - 0.03 = 0.77 \text{ bit/Photon} \tag{407}$$

Zukunftsvision - Globales Quanteninternet: Geschätzte Parameter um 2040:

- Verschränkungsrate zwischen Kontinenten: 10^3 Bell-Paare/s
- Fidelity nach Purification: $F > 0.999$
- Klassische Kommunikation: 1 ms Latenz
- Quantenspeicher-Kohärenzzeit: $T_2 > 1$ s

Damit können wir verteilte Quantenalgorithmen für Probleme lösen, die heute undenkbar sind!

Das Quanteninternet wird alles ändern - von der Kryptographie bis zur verteilten Berechnung. Die Mathematik dahinter ist schon jetzt faszinierend, aber die praktische Umsetzung... das wird noch dauern!

Fun Fact: Bei perfekter globaler Verschränkung könnten wir theoretisch jede beliebige Quantenberechnung auf planetarer Ebene durchführen. Das wäre wie ein riesiger Quantencomputer, verteilt über die ganze Erde!

28 Quantencomputer-Programmierung

28.0.1 Qiskit (Python) - IBM Quantum

Qiskit ist IBMs Open-Source-Framework für Quantencomputing. Hier sind ausführliche Beispiele:

Listing 1: Grundlegende Quantenschaltung mit Qiskit

```
# Installation: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit, QuantumRegister, Cl
from qiskit import transpile, assemble
from qiskit_aer import AerSimulator
```

```
                    from qiskit.visualization import plot_histogram
                    import numpy as np

                    # Beispiel 1: Bell-Zustand (maximale Verschränkung)
                    def create_bell_state():
                    """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Erstellt␣einen␣Bell-Zustand␣|+␣=␣(|00␣+␣|11)/2
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Demonstriert␣Superposition␣und␣Verschränkung
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""
                    # Quantenregister mit 2 Qubits, klassisches Register f
                    qreg = QuantumRegister(2, 'q')
                    creg = ClassicalRegister(2, 'c')
                    circuit = QuantumCircuit(qreg, creg)

                    # Hadamard-Gatter auf das erste Qubit (Superposition)
                    circuit.h(qreg[0])

                    # CNOT-Gatter für Verschränkung
                    circuit.cx(qreg[0], qreg[1])

                    # Messung beider Qubits
                    circuit.measure(qreg, creg)

                    return circuit

                    # Beispiel 2: Quantum Random Number Generator
                    def quantum_random_number_generator(num_bits=4):
                    """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Erzeugt␣echte␣Zufallszahlen␣mittels␣Quantensuperpositio
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""
                    qreg = QuantumRegister(num_bits, 'q')
                    creg = ClassicalRegister(num_bits, 'c')
                    circuit = QuantumCircuit(qreg, creg)

                    # Hadamard-Gatter auf alle Qubits
                    for i in range(num_bits):
                    circuit.h(qreg[i])

                    # Messung aller Qubits
                    circuit.measure(qreg, creg)
```

```
    return circuit

    # Beispiel 3: Grover-Algorithmus (vereinfacht)
    def grover_algorithm(marked_item='11'):
    """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Implementierung␣des␣Grover-Algorithmus␣für␣Datenbanksu
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Quadratische␣Beschleunigung␣gegenüber␣klassischen␣Meth
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""
    n_qubits = len(marked_item)
    qreg = QuantumRegister(n_qubits, 'q')
    creg = ClassicalRegister(n_qubits, 'c')
    circuit = QuantumCircuit(qreg, creg)

    # Superposition aller Zustände
    for i in range(n_qubits):
    circuit.h(qreg[i])

    # Oracle für markiertes Element (vereinfacht)
    if marked_item == '11':
    circuit.cz(qreg[0], qreg[1])

    # Diffusions-Operator (vereinfacht)
    for i in range(n_qubits):
    circuit.h(qreg[i])
    circuit.x(qreg[i])

    circuit.cz(qreg[0], qreg[1])

    for i in range(n_qubits):
    circuit.x(qreg[i])
    circuit.h(qreg[i])

    circuit.measure(qreg, creg)
    return circuit

    # Simulation und Ausführung
    def run_quantum_simulation(circuit, shots=1024):
    """Simuliert␣die␣Quantenschaltung␣und␣gibt␣Ergebnisse␣
    simulator = AerSimulator()
    compiled_circuit = transpile(circuit, simulator)
    result = simulator.run(compiled_circuit, shots=shots).
```

```
counts = result.get_counts()
return counts

# Beispielverwendung
if __name__ == "__main__":
# Bell-Zustand testen
bell_circuit = create_bell_state()
bell_results = run_quantum_simulation(bell_circuit)
print("Bell-Zustand Ergebnisse:", bell_results)

# Zufallszahlen generieren
rng_circuit = quantum_random_number_generator(3)
rng_results = run_quantum_simulation(rng_circuit)
print("Quantenzufallszahlen:", rng_results)
```

28.0.2 Q# (Microsoft Quantum Development Kit)

Q# ist Microsofts domänenspezifische Sprache für Quantencomputing:

Listing 2: Quantenprogrammierung mit Q#

```
// Datei: QuantumOperations.qs
namespace QuantumExamples {
        open Microsoft.Quantum.Canon;
        open Microsoft.Quantum.Intrinsic;
        open Microsoft.Quantum.Measurement;
        open Microsoft.Quantum.Math;
        open Microsoft.Quantum.Convert;

        // Beispiel 1: Bell-Zustand Erzeugung
        operation CreateBellState() : (Result, Result)
                // Allokiere zwei Qubits im |0 Zustand
                using ((qubit1, qubit2) = (Qubit(), Qu
                        // Hadamard-Gatter auf erstes
                        H(qubit1);

                        // CNOT-Gatter für Verschränku
                        CNOT(qubit1, qubit2);

                        // Messung beider Qubits
                        let result1 = M(qubit1);
                        let result2 = M(qubit2);
```

```
        // Qubits zurücksetzen (Q# Anf
        Reset(qubit1);
        Reset(qubit2);

        return (result1, result2);
    }
}

// Beispiel 2: Quantum Teleportation
operation QuantumTeleportation(message : Qubit
    using (ancilla = Qubit()) {
        // Bell-Paar zwischen Sender u
        H(ancilla);
        CNOT(ancilla, target);

        // Bell-Messung am Sender
        CNOT(message, ancilla);
        H(message);

        let messageResult = M(message)
        let ancillaResult = M(ancilla)

        // Korrektur am Empfänger basi
        if (messageResult == One) {
            Z(target);
        }
        if (ancillaResult == One) {
            X(target);
        }

        Reset(message);
        Reset(ancilla);
    }
}

// Beispiel 3: Quantum Phase Estimation (verei
operation QuantumPhaseEstimation() : Double {
    let precision = 4; // Anzahl der Preci

    using ((precisionQubits, eigenQubit) =
```

```
        (Qubit[precision], Qubit())) {

            // Vorbereitung des Eigenvekto
            X(eigenQubit); // |1 Zustand

            // Superposition der Precision
            ApplyToEach(H, precisionQubits

            // Kontrollierte Unitäre Opera
            for (i in 0..precision-1) {
                for (j in 0..(2^i)-1)
                    Controlled Z([
                }
            }

            // Inverse Quantum Fourier Tra
            Adjoint QFT(LittleEndian(preci

            // Messung und Phasenberechnun
            let results = MultiM(precision
            let phase = IntAsDouble(Result
            IntAsDouble(2^precision);

            ResetAll(precisionQubits);
            Reset(eigenQubit);

            return phase;
        }
}

// Hilfsfunktion: Quantum Fourier Transform
operation QFT(register : LittleEndian) : Unit
        let qubits = register!;
        let n = Length(qubits);

        for (i in 0..n-1) {
            H(qubits[i]);
            for (j in i+1..n-1) {
                Controlled R1([qubits[
                (PI() / IntAsDouble(2^
            }
```

```
                }

                // Bit-Reversal
                for (i in 0..n/2-1) {
                    SWAP(qubits[i], qubits[n-1-i])
                }
            }
        }

        // Host-Programm (C#)
        using Microsoft.Quantum.Simulation.Core;
        using Microsoft.Quantum.Simulation.Simulators;

        class Program {
            static void Main(string[] args) {
                using (var qsim = new QuantumSimulator
                    // Bell-Zustand Test
                    var bellResult = CreateBellSta
                    Console.WriteLine($"Bell-Zusta

                    // Phasenschätzung
                    var phase = QuantumPhaseEstima
                    Console.WriteLine($"Geschätzte
                }
            }
        }
```

28.0.3 Cirq (Python) - Google Quantum AI

Cirq ist Googles Framework für Quantenschaltkreise auf NISQ-Geräten:

Listing 3: Quantenprogrammierung mit Cirq

```
import cirq
import numpy as np
from typing import List

# Beispiel 1: Variational Quantum Eigensolver (VQE)
class VQECircuit:
"""
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Implementierung␣eines␣einfachen␣VQE␣für␣das␣H2-Molekül
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Verwendet␣parametrisierte␣Quantenschaltkreise
```

```
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""

                    def __init__(self, num_qubits: int = 2):
                    self.num_qubits = num_qubits
                    self.qubits = cirq.LineQubit.range(num_qubits)

                    def create_ansatz(self, params: List[float]) -> cirq.C
                    """Erstellt␣den␣parametrisierten␣Ansatz-Schaltkreis"""
                    circuit = cirq.Circuit()

                    # RY-Rotationen auf alle Qubits
                    for i, qubit in enumerate(self.qubits):
                    circuit.append(cirq.ry(params[i])(qubit))

                    # Verschränkungsschicht
                    for i in range(len(self.qubits) - 1):
                    circuit.append(cirq.CNOT(self.qubits[i], self.qubits[i

                    # Weitere parametrisierte Schicht
                    for i, qubit in enumerate(self.qubits):
                    circuit.append(cirq.rz(params[i + len(self.qubits)])(q

                    return circuit

                    def create_hamiltonian_circuits(self) -> List[cirq.Cir
                    """Erstellt␣Schaltkreise␣zur␣Hamiltonian-Messung"""
                    circuits = []

                    # Pauli-Z Messungen (bereits in Rechenbasis)
                    z_circuit = cirq.Circuit()
                    circuits.append(z_circuit)

                    # Pauli-X Messungen (benötigen Hadamard vor Messung)
                    x_circuit = cirq.Circuit()
                    for qubit in self.qubits:
                    x_circuit.append(cirq.H(qubit))
                    circuits.append(x_circuit)

                    # Pauli-Y Messungen (benötigen SH vor Messung)
                    y_circuit = cirq.Circuit()
                    for qubit in self.qubits:
```

```
                    y_circuit.append(cirq.S(qubit) ** -1)
                    y_circuit.append(cirq.H(qubit))
                    circuits.append(y_circuit)

                    return circuits

                    # Beispiel 2: Quantum Approximate Optimization Algorith
                    class QAOACircuit:
                    """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣QAOA␣für␣Max-Cut␣Problem␣auf␣einem␣einfachen␣Graphen
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""

                    def __init__(self, graph_edges: List[tuple]):
                    self.edges = graph_edges
                    self.num_nodes = max(max(edge) for edge in graph_edges
                    self.qubits = cirq.LineQubit.range(self.num_nodes)

                    def create_qaoa_circuit(self, gamma: float, beta: floa
                    """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Erstellt␣QAOA-Schaltkreis␣mit␣p␣Layern
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣gamma:␣Problem-Parameter,␣beta:␣Mischer-Parameter
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""
                    circuit = cirq.Circuit()

                    # Anfangszustand: Superposition aller Knoten
                    circuit.append(cirq.H.on_each(*self.qubits))

                    # p QAOA-Layer
                    for layer in range(p):
                    # Problem-Unitäre (Phase-Separation)
                    for edge in self.edges:
                    i, j = edge
                    circuit.append(cirq.ZZ(self.qubits[i], self.qubits[j])

                    # Mischer-Unitäre
                    for qubit in self.qubits:
                    circuit.append(cirq.rx(2 * beta)(qubit))

                    return circuit

                    def measure_expectation(self, circuit: cirq.Circuit, r
```

```
                """Misst␣den␣Erwartungswert␣der␣Kostenfunktion"""
                # Messungen hinzufügen
                measurement_circuit = circuit.copy()
                measurement_circuit.append(cirq.measure(*self.qubits,

                # Simulation
                simulator = cirq.Simulator()
                result = simulator.run(measurement_circuit, repetition
                measurements = result.measurements['result']

                # Berechne Kostenfunktion für jede Messung
                total_cost = 0
                for measurement in measurements:
                cost = 0
                for edge in self.edges:
                i, j = edge
                # Kostenfunktion: +1 wenn Knoten in verschiedenen Schn
                if measurement[i] != measurement[j]:
                cost += 1
                total_cost += cost

                return total_cost / repetitions

                # Beispiel 3: Quantum Machine Learning - Parametrisier
                class QuantumNeuralNetwork:
                """
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Einfaches␣Quantum␣Neural␣Network␣für␣binäre␣Klassifika
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""

                def __init__(self, num_features: int, num_layers: int =
                self.num_features = num_features
                self.num_layers = num_layers
                self.qubits = cirq.LineQubit.range(num_features)
                self.num_params = num_features * num_layers * 3
# 3 Parameter pro Qubit pro Layer

                def data_encoding(self, x: np.ndarray) -> cirq.Circuit
                """Kodiert␣klassische␣Daten␣in␣Quantenzustände"""
                circuit = cirq.Circuit()
                for i, value in enumerate(x):
                # Amplitude-Encoding mittels RY-Rotation
```

```
circuit.append(cirq.ry(np.pi * value)(self.qubits[i]))
return circuit

def variational_circuit(self, params: np.ndarray) -> c
"""Parametrisierte Quantenschaltung"""
circuit = cirq.Circuit()
param_idx = 0

for layer in range(self.num_layers):
# Parametrisierte Einzelqubit-Gatter
for qubit in self.qubits:
circuit.append(cirq.rx(params[param_idx])(qubit))
circuit.append(cirq.ry(params[param_idx + 1])(qubit))
circuit.append(cirq.rz(params[param_idx + 2])(qubit))
param_idx += 3

# Verschränkungsschicht
for i in range(len(self.qubits) - 1):
circuit.append(cirq.CNOT(self.qubits[i], self.qubits[i

return circuit

def predict(self, x: np.ndarray, params: np.ndarray) -
"""Vorhersage für einen Eingabevektor"""
# Vollständiger Schaltkreis
circuit = cirq.Circuit()
circuit += self.data_encoding(x)
circuit += self.variational_circuit(params)

# Messung des ersten Qubits
circuit.append(cirq.measure(self.qubits[0], key='outpu

# Simulation
simulator = cirq.Simulator()
result = simulator.run(circuit, repetitions=1000)

# Erwartungswert berechnen
measurements = result.measurements['output']
expectation = np.mean(measurements)

return expectation
```

```
# Beispielverwendung der Cirq-Implementierungen
def demonstrate_cirq_examples():
"""Demonstriert die verschiedenen Cirq-Beispiele"""

# VQE Beispiel
print("=== VQE Beispiel ===")
vqe = VQECircuit(num_qubits=2)
params = np.random.random(4) * 2 * np.pi
ansatz = vqe.create_ansatz(params)
print("VQE Ansatz-Schaltkreis:")
print(ansatz)

# QAOA Beispiel
print("\n=== QAOA Beispiel ===")
edges = [(0, 1), (1, 2), (2, 0)]  # Dreieck-Graph
qaoa = QAOACircuit(edges)
gamma, beta = 0.5, 0.3
qaoa_circuit = qaoa.create_qaoa_circuit(gamma, beta)
print("QAOA Schaltkreis:")
print(qaoa_circuit)

expectation = qaoa.measure_expectation(qaoa_circuit)
print(f"Erwartungswert der Kostenfunktion: {expectation

# Quantum Neural Network Beispiel
print("\n=== Quantum Neural Network Beispiel ===")
qnn = QuantumNeuralNetwork(num_features=2, num_layers=
x = np.array([0.5, 0.8])  # Beispiel-Eingabe
params = np.random.random(qnn.num_params) * 2 * np.pi

prediction = qnn.predict(x, params)
print(f"QNN Vorhersage für Input {x}: {prediction:.3f}

if __name__ == "__main__":
demonstrate_cirq_examples()
```

28.0.4 Weitere Quantenprogrammiersprachen

PennyLane (Xanadu) PennyLane kombiniert Quantencomputing mit maschinellem Lernen und bietet automatische Differentiation:

Listing 4: PennyLane Quantum Machine Learning

```
import pennylane as qml
import numpy as np

# Quantengerät definieren
dev = qml.device('default.qubit', wires=2)

@qml.qnode(dev)
def quantum_classifier(x, weights):
"""Parametrisierte Quantenschaltung für Klassifikation
# Daten-Encoding
qml.RY(np.pi * x[0], wires=0)
qml.RY(np.pi * x[1], wires=1)

# Parametrisierte Layer
qml.RY(weights[0], wires=0)
qml.RY(weights[1], wires=1)
qml.CNOT(wires=[0, 1])
qml.RY(weights[2], wires=0)
qml.RY(weights[3], wires=1)

return qml.expval(qml.PauliZ(0))

# Kostenfunktion
def cost_function(weights, X, Y):
predictions = [quantum_classifier(x, weights) for x in
return np.mean((predictions - Y)**2)

# Gradient berechnen (automatische Differentiation)
grad_fn = qml.grad(cost_function)

# Training
weights = np.random.random(4)
X_train = np.array([[0.1, 0.2], [0.8, 0.9], [0.3, 0.1]
Y_train = np.array([0, 1, 0, 1])

learning_rate = 0.1
for epoch in range(100):
weights = weights - learning_rate * grad_fn(weights, X
if epoch % 20 == 0:
```

```
    cost = cost_function(weights, X_train, Y_train)
    print(f"Epoch {epoch}: Cost = {cost:.4f}")
```

Listing 5: Continuous Variable Quantum Computing

```
import strawberryfields as sf
from strawberryfields.ops import *
import numpy as np

# Beispiel: Gaussian Boson Sampling
prog = sf.Program(3)  # 3 Modi

with prog.context as q:
# Squeezed States erzeugen
Sgate(0.5) | q[0]
Sgate(0.5) | q[1]
Sgate(0.5) | q[2]

# Interferometer (3x3 unitäre Matrix)
U = np.array([[1, 1, 1],
[1, np.exp(2j*np.pi/3), np.exp(4j*np.pi/3)],
[1, np.exp(4j*np.pi/3), np.exp(2j*np.pi/3)]]) / np.sqr

Interferometer(U) | (q[0], q[1], q[2])

# Photonenzahl-Messung
MeasureFock() | q[0]
MeasureFock() | q[1]
MeasureFock() | q[2]

# Simulation
eng = sf.Engine("gaussian")
result = eng.run(prog, shots=1000)
samples = result.samples
print("Gaussian Boson Sampling Ergebnisse:", samples[:
```

28.1 Quantenalgorithmen: Detaillierte Implementierungen

28.1.1 Shor-Algorithmus für Faktorisierung

Der Shor-Algorithmus demonstriert die exponentiellen Vorteile von Quantencomputern:

Listing 6: Vereinfachte Shor-Algorithmus Implementierung

```
import numpy as np
from qiskit import QuantumCircuit, QuantumRegister, Cl
from qiskit_aer import AerSimulator
from math import gcd, log2, ceil
import random

class ShorAlgorithm:
"""Vereinfachte␣Implementierung␣des␣Shor-Algorithmus""

def __init__(self, N: int):
self.N = N  # Zu faktorisierende Zahl
self.n = ceil(log2(N))  # Anzahl Qubits für N
self.m = 2 * self.n + 1  # Qubits für Periodensuche

def classical_preprocessing(self):
"""Klassische␣Vorverarbeitung"""
# Prüfe triviale Fälle
if self.N % 2 == 0:
return 2, self.N // 2

# Wähle zufälliges a < N mit gcd(a, N) = 1
while True:
a = random.randint(2, self.N - 1)
if gcd(a, self.N) == 1:
return a, None

def create_quantum_circuit(self, a: int) -> QuantumCi
"""Erstellt␣Quantenschaltkreis␣für␣Periodensuche"""
# Register: |counting|auxiliary
counting_reg = QuantumRegister(self.m, 'counting')
auxiliary_reg = QuantumRegister(self.n, 'aux')
classical_reg = ClassicalRegister(self.m, 'c')

circuit = QuantumCircuit(counting_reg, auxiliary_reg,
```

```
        # Schritt 1: Superposition im Counting-Register
        circuit.h(counting_reg)

        # Schritt 2: Auxiliary-Register in |1> Zustand
        circuit.x(auxiliary_reg[-1])

        # Schritt 3: Kontrollierte Modular-Exponentiation
        # Vereinfacht: U|y> = |ay mod N>
        for i in range(self.m):
            # Für echte Implementierung: kontrollierte modulare M
            # Hier vereinfacht dargestellt
            self._controlled_modular_multiplication(
                circuit, counting_reg[i], auxiliary_reg, a, 2**i
            )

        # Schritt 4: Inverse Quantum Fourier Transform
        self._inverse_qft(circuit, counting_reg)

        # Schritt 5: Messung des Counting-Registers
        circuit.measure(counting_reg, classical_reg)

        return circuit

    def _controlled_modular_multiplication(self, circuit,
        """Vereinfachte kontrollierte modulare Multiplikation"
        # Dies ist eine stark vereinfachte Darstellung
        # Echte Implementierung erfordert komplexe modulare Ar
        if power % 2 == 1:  # Vereinfachung für Demonstration
            circuit.cx(control, target_reg[0])

    def _inverse_qft(self, circuit, qubits):
        """Inverse Quantum Fourier Transform"""
        n = len(qubits)

        # Bit-Reversal
        for i in range(n // 2):
            circuit.swap(qubits[i], qubits[n - 1 - i])

        # QFT
        for i in range(n):
            for j in range(i):
```

```
circuit.cp(-np.pi / (2 ** (i - j)), qubits[i], qubits[
circuit.h(qubits[i])

def find_period_classical_postprocessing(self, measure
"""Klassische Nachbearbeitung zur Periodenfindung"""
# Continued Fraction Expansion für Periodenschätzung
# Vereinfacht dargestellt
measured_values = []
for result_string in measurement_results:
value = int(result_string, 2)
measured_values.append(value)

# Häufigster Messwert (vereinfacht)
from collections import Counter
most_common = Counter(measured_values).most_common(1)[

# Schätze Periode r
if most_common == 0:
return None

# Berechne r mittels Continued Fractions (vereinfacht)
period_estimate = (2 ** self.m) // most_common
return period_estimate

def factorize(self):
"""Vollständige Faktorisierung"""
# Klassische Vorverarbeitung
a, trivial_factor = self.classical_preprocessing()
if trivial_factor:
return 2, trivial_factor

print(f"Verwende a = {a} für Faktorisierung von N = {s

# Quantenschaltkreis erstellen und simulieren
circuit = self.create_quantum_circuit(a)

# Simulation
simulator = AerSimulator()
compiled_circuit = transpile(circuit, simulator)
result = simulator.run(compiled_circuit, shots=1024).r
counts = result.get_counts()
```

```
        # Periode finden
        period = self.find_period_classical_postprocessing(cou

        if period is None or period % 2 != 0:
        print("Periode␣nicht␣gefunden␣oder␣ungerade.␣Neuer␣Ver
        return None, None

        # Faktoren berechnen
        factor1 = gcd(a ** (period // 2) - 1, self.N)
        factor2 = gcd(a ** (period // 2) + 1, self.N)

        if factor1 > 1 and factor1 < self.N:
        return factor1, self.N // factor1
        elif factor2 > 1 and factor2 < self.N:
        return factor2, self.N // factor2
        else:
        print("Faktorisierung␣fehlgeschlagen.␣Neuer␣Versuch␣nö
        return None, None

        # Beispielverwendung
        if __name__ == "__main__":
        # Kleine Zahl für Demonstration (echte Implementierung
        N = 15
        shor = ShorAlgorithm(N)

        factor1, factor2 = shor.factorize()
        if factor1 and factor2:
        print(f"Faktoren␣von␣{N}:␣{factor1}␣Œ␣{factor2}␣=␣{fac
        else:
        print("Faktorisierung␣nicht␣erfolgreich")
```

28.2 Quantum Error Correction

Quantenfehlerkorrektur ist essentiell für fault-tolerante Quantencomputer:

Listing 7: Quantum Error Correction - Surface Code

```
        import numpy as np
        from qiskit import QuantumCircuit, QuantumRegister, Cl
        from typing import List, Tuple
```

```
class SurfaceCode:
"""
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Implementierung␣eines␣vereinfachten␣Surface␣Code
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣für␣Quantenfehlerkorrektur
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"""

def __init__(self, distance: int = 3):
self.distance = distance  # Code-Distanz (muss ungerad
self.num_data_qubits = distance ** 2
self.num_ancilla_qubits = distance ** 2 - 1
self.total_qubits = self.num_data_qubits + self.num_a

# Definiere Qubit-Layout
self.data_qubits = []
self.x_ancilla_qubits = []
self.z_ancilla_qubits = []

self._setup_qubit_layout()

def _setup_qubit_layout(self):
"""Organisiert␣Qubits␣in␣Surface␣Code␣Layout"""
# Vereinfachtes 2D-Gitter Layout
qubit_id = 0

for row in range(self.distance):
for col in range(self.distance):
if (row + col) % 2 == 0:  # Daten-Qubits
self.data_qubits.append(qubit_id)
else:  # Ancilla-Qubits
if row % 2 == 1:  # X-Stabilizer
self.x_ancilla_qubits.append(qubit_id)
else:  # Z-Stabilizer
self.z_ancilla_qubits.append(qubit_id)
qubit_id += 1

def create_encoding_circuit(self, logical_zero: bool =
"""Kodiert␣einen␣logischen␣Qubit-Zustand"""
qreg = QuantumRegister(self.total_qubits, 'q')
circuit = QuantumCircuit(qreg)
```

```
if logical_zero:
# Logischer |0 Zustand (alle Daten-Qubits in |0)
pass  # Bereits im gewünschten Zustand
else:
# Logischer |1 Zustand
# Für Surface Code: logisches X anwenden
for qubit in self.data_qubits[::2]:  # Jedes zweite Da
circuit.x(qreg[qubit])

return circuit

def create_syndrome_measurement_circuit(self) -> Quant
"""Erstellt Schaltkreis für Syndrom-Messung"""
qreg = QuantumRegister(self.total_qubits, 'q')
x_creg = ClassicalRegister(len(self.x_ancilla_qubits),
z_creg = ClassicalRegister(len(self.z_ancilla_qubits),
circuit = QuantumCircuit(qreg, x_creg, z_creg)

# X-Stabilizer Messungen
for i, ancilla in enumerate(self.x_ancilla_qubits):
# Hadamard auf Ancilla
circuit.h(qreg[ancilla])

# CNOT zu benachbarten Daten-Qubits
neighbors = self._get_neighbors(ancilla, 'x')
for neighbor in neighbors:
circuit.cx(qreg[ancilla], qreg[neighbor])

# Hadamard und Messung
circuit.h(qreg[ancilla])
circuit.measure(qreg[ancilla], x_creg[i])

# Z-Stabilizer Messungen
for i, ancilla in enumerate(self.z_ancilla_qubits):
# CNOT von benachbarten Daten-Qubits
neighbors = self._get_neighbors(ancilla, 'z')
for neighbor in neighbors:
circuit.cx(qreg[neighbor], qreg[ancilla])

# Messung
circuit.measure(qreg[ancilla], z_creg[i])
```

```
        return circuit

    def _get_neighbors(self, ancilla_qubit: int, stabilizer
    """Bestimmt benachbarte Daten-Qubits für Stabilizer-Me
    # Vereinfachte Nachbarschaftsbestimmung
    # In echter Implementierung: 2D-Gitter-Topologie
    neighbors = []

    if stabilizer_type == 'x' and ancilla_qubit in self.x_
    # X-Stabilizer: 4 benachbarte Daten-Qubits
    idx = self.x_ancilla_qubits.index(ancilla_qubit)
    if idx < len(self.data_qubits) - 1:
    neighbors.extend(self.data_qubits[idx:idx+2])
    elif stabilizer_type == 'z' and ancilla_qubit in self.
    # Z-Stabilizer: 4 benachbarte Daten-Qubits
    idx = self.z_ancilla_qubits.index(ancilla_qubit)
    if idx < len(self.data_qubits) - 1:
    neighbors.extend(self.data_qubits[idx:idx+2])

    return neighbors

    def decode_syndrome(self, x_syndrome: List[int], z_syn
    """
                    Dekodiert Syndrom-Messungen zu Fehlerlokalisation
                    Vereinfachte Implementierung
                    """
    x_errors = []
    z_errors = []

    # Vereinfachte Dekodierung: Minimum Weight Perfect Mat
    # In echter Implementierung: komplexe Graphenalgorithm

    for i, syndrome_bit in enumerate(x_syndrome):
    if syndrome_bit == 1:
    # X-Fehler detektiert
    x_errors.append(self.data_qubits[i % len(self.data_qub

    for i, syndrome_bit in enumerate(z_syndrome):
    if syndrome_bit == 1:
    # Z-Fehler detektiert
```

```
        z_errors.append(self.data_qubits[i % len(self.data_qub

        return x_errors, z_errors

    def create_correction_circuit(self, x_errors: List[int
        """Erstellt Fehlerkorrektur-Schaltkreis"""
        qreg = QuantumRegister(self.total_qubits, 'q')
        circuit = QuantumCircuit(qreg)

        # X-Fehler korrigieren (mit X-Gattern)
        for error_qubit in x_errors:
            circuit.x(qreg[error_qubit])

        # Z-Fehler korrigieren (mit Z-Gattern)
        for error_qubit in z_errors:
            circuit.z(qreg[error_qubit])

        return circuit

    def full_error_correction_cycle(self) -> QuantumCircui
        """Vollständiger Fehlerkorrektur-Zyklus"""
        qreg = QuantumRegister(self.total_qubits, 'q')
        x_creg = ClassicalRegister(len(self.x_ancilla_qubits),
        z_creg = ClassicalRegister(len(self.z_ancilla_qubits),
        circuit = QuantumCircuit(qreg, x_creg, z_creg)

        # 1. Kodierung (logischer |0> Zustand)
        encoding_circuit = self.create_encoding_circuit(logical
        circuit = circuit.compose(encoding_circuit)

        # 2. Simulation von Fehlern (Rauschkanäle)
        # Hier würden normalerweise Fehlermodelle angewendet

        # 3. Syndrom-Messung
        syndrome_circuit = self.create_syndrome_measurement_c
        circuit = circuit.compose(syndrome_circuit)

        # 4. Klassische Dekodierung und Korrektur
        # (in echter Implementierung dynamisch basierend auf M

        return circuit
```

```
# Beispiel für Verwendung
def demonstrate_surface_code():
"""Demonstriert Surface Code Implementierung"""
surface_code = SurfaceCode(distance=3)

print(f"Surface Code mit Distanz {surface_code.distanc
print(f"Daten-Qubits: {len(surface_code.data_qubits)}"
print(f"Ancilla-Qubits: {len(surface_code.x_ancilla_qu

# Vollständiger Korrektur-Zyklus
circuit = surface_code.full_error_correction_cycle()
print(f"Gesamte Schaltkreis-Tiefe: {circuit.depth()}")
print(f"Anzahl Quantengatter: {len(circuit.data)}")

# Syndrom-Dekodierung Beispiel
x_syndrome = [0, 1, 0]  # Beispiel-Syndrom
z_syndrome = [1, 0, 1]

x_errors, z_errors = surface_code.decode_syndrome(x_sy
print(f"Detektierte X-Fehler auf Qubits: {x_errors}")
print(f"Detektierte Z-Fehler auf Qubits: {z_errors}")

if __name__ == "__main__":
demonstrate_surface_code()
```

28.3 Fazit und Ausblick

Quantencomputing befindet sich an der Schwelle zu praktischen Anwendungen. Die vorgestellten Programmiersprachen und Frameworks ermöglichen es bereits heute, komplexe Quantenalgorithmen zu implementieren und zu testen. Von Qiskit über Q# bis hin zu Cirq bietet jedes Framework spezifische Vorteile für verschiedene Anwendungsbereiche.

Die Entwicklung fehlertoleranter Quantencomputer durch fortschrittliche Quantenfehlerkorrektur-Codes wie den Surface Code wird es ermöglichen, groSSe Quantenalgorithmen praktisch umzusetzen. Dies wird Durchbrüche in Bereichen wie Kryptographie, Optimierung, Materialwissenschaft und maschinellem Lernen ermöglichen.

Die Quantenprogrammierung erfordert ein neues Denkmodell, das Kon-

zepte wie Superposition, Verschränkung und Quanteninterferenz berücksichtigt. Die gezeigten ausführlichen Beispiele demonstrieren, wie diese Konzepte in praktischen Implementierungen umgesetzt werden können.

29 Ethische Aspekte der Quantentechnologie

Hier wird's philosophisch! Quantentechnologie wird die Welt verändern, aber ist das auch gut für die Menschheit? Lass uns mal rechnen, was passiert, wenn Quantencomputer wirklich kommen...

29.1 Quantenkryptographischer Kollaps und gesellschaftliche Folgen

Das Y2Q-Problem (Years to Quantum): Wann bricht RSA-2048? Shor-Algorithmus braucht etwa $\mathcal{O}(n^3)$ Gatter für n-Bit-Zahlen.

Für RSA-2048:

$$\text{Benötigte logische Qubits} \approx 4n = 4 \times 2048 = 8192 \tag{408}$$

$$\text{Benötigte Gatter} \approx n^3 = 2048^3 \approx 8.6 \times 10^9 \tag{409}$$

Zeitschätzung mit heutiger Hardware: IBM's beste Quantencomputer (2025): 1000 Qubits, 10^{-3} Fehlerrate

Für fehlertolerante Berechnung brauchen wir:

$$\text{Physikalische Qubits} = \text{Logische Qubits} \times \text{Overhead} \tag{410}$$

Mit Surface Code: Overhead ≈ 1000 bei 10^{-3} Fehlerrate

$$\text{Benötigte physikalische Qubits} = 8192 \times 1000 = 8.192 \times 10^6 \tag{411}$$

Das sind über 8 Millionen Qubits! Wir sind noch weit weg, aber es kommt...

Gesellschaftliche Kosten des Krypto-Kollapses: Geschätzter Schaden bei sofortigem RSA-Bruch:

$$\text{Globaler Schaden} = \text{Finanzsektor} + \text{E-Commerce} + \text{Infrastruktur} \tag{412}$$

$$\approx 15 \times 10^{12} + 8 \times 10^{12} + 25 \times 10^{12} \tag{413}$$

$$= 48 \text{ Billionen USD} \tag{414}$$

Das ist mehr als das doppelte BIP der USA! Absolut katastrophal...

29.2 Quantenzugang und digitale Gerechtigkeit

Quantenvorteil-Ungleichung: Sei $Q(n)$ die Problemlösezeit mit Quantencomputern und $C(n)$ mit klassischen Computern.

Für bestimmte Probleme:

$$\frac{C(n)}{Q(n)} = e^{\Omega(\sqrt{n \log n})} \quad \text{(exponentieller Vorteil)} \tag{415}$$

Beispiel - Datenbanksuche: Klassisch (brute force): $\mathcal{O}(N)$ Schritte Grover-Algorithmus: $\mathcal{O}(\sqrt{N})$ Schritte

Für $N = 10^{12}$ Einträge:

$$\text{Klassisch:} \quad 10^{12} \text{ Schritte} \approx 1000 \text{ Sekunden} \tag{416}$$

$$\text{Quantum:} \quad 10^{6} \text{ Schritte} \approx 0.001 \text{ Sekunden} \tag{417}$$

Speedup-Faktor: 10^6 - Das ist ein unfairer Vorteil!

Gini-Koeffizient für Quantenzugang: Sei q_i der Quantencomputer-Zugang für Land i (gemessen in Qubit-Stunden/Jahr pro Kopf).

$$G = \frac{1}{2n^2\bar{q}} \sum_{i=1}^{n} \sum_{j=1}^{n} |q_i - q_j| \tag{418}$$

Aktuelle Schätzung (2025):

$$\text{USA:} \quad q_{\text{USA}} = 0.1 \text{ Qubit-h/Jahr pro Person} \tag{419}$$

$$\text{Deutschland:} \quad q_{\text{DE}} = 0.02 \text{ Qubit-h/Jahr pro Person} \tag{420}$$

$$\text{Entwicklungsländer:} \quad q_{\text{EL}} \approx 0 \text{ Qubit-h/Jahr pro Person} \tag{421}$$

Das ergibt: $G \approx 0.85$ (sehr ungleich! Zum Vergleich: Einkommens-Gini weltweit ≈ 0.7)

29.3 Umweltethik und Quantencomputing

Energieverbrauch von Quantencomputern: Dilution Refrigerator für 1000-Qubit-System:

$$P_{\text{Kühlung}} = 25 \text{ kW (kontinuierlich)} \tag{422}$$

$$P_{\text{Elektronik}} = 10 \text{ kW} \tag{423}$$

$$P_{\text{Gesamt}} = 35 \text{ kW} \tag{424}$$

Pro Jahr: $35 \times 24 \times 365 = 306600$ kWh ≈ 307 MWh

CO-FuSSabdruck: Mit deutschem Strommix (2025): ≈ 400 g CO/kWh

$$\text{CO/Jahr} = 307000 \times 0.4 = 122800 \text{ kg CO} \approx 123 \text{ Tonnen CO} \tag{425}$$

Das ist wie 27 durchschnittliche Deutsche! Aber wenn der Quantencomputer dafür klassische Supercomputer ersetzt...

Vergleich mit klassischen Supercomputern: Top500-Supercomputer verbraucht ≈ 20 MW für vergleichbare Probleme:

$$\text{Klassisch:} \quad 20000 \times 8760 = 175200 \text{ MWh/Jahr} \tag{426}$$

$$\text{Quantum:} \quad 0.307 \text{ MWh/Jahr} \tag{427}$$

Einsparung: Faktor 570000! *Okay, das ist schon besser für die Umwelt...*

29.4 Ethische Bewertungsmodelle für Quantentechnologie

Utilitaristisches Modell: Nutzen U einer Quantentechnologie:

$$U = \sum_{i=1}^{n} w_i \cdot b_i - \sum_{j=1}^{m} v_j \cdot c_j \tag{428}$$

wo b_i = Nutzen für Gruppe i, c_j = Kosten/Schäden für Gruppe j

Beispiel - Quantenkryptographie:

$$U_{\text{QKD}} = 0.4 \times (+1000) \text{ (Sicherheitsgewinn)} \tag{429}$$

$$+ 0.3 \times (-200) \text{ (Implementierungskosten)} \tag{430}$$

$$+ 0.3 \times (-50) \text{ (Komplexität)} \tag{431}$$

$$= 400 - 60 - 15 = +325 \tag{432}$$

Positiver Nutzen $\rightarrow$ ethisch vertretbar!

Rawls'sches Gerechtigkeitsmodell: "Wähle die Technologie, die die Situation der am schlechtesten gestellten Gruppe maximal verbessert."

Sei $w_{\min}$ das Wohlbefinden der ärmsten Gruppe:

$$\max_{\text{Tech}} \quad w_{\min}(\text{Tech}) \tag{433}$$

Risikoanalyse für Quantenwaffen: Wahrscheinlichkeit einer militärischen Quanteneskalation:

$$P(\text{Eskalation}) = P(\text{Durchbruch}) \times P(\text{Militarisierung}|\text{Durchbruch}) \quad (434)$$
$$\times P(\text{Konflikt}|\text{Militarisierung}) \quad (435)$$

Geschätzte Werte:

$$P(\text{Durchbruch in 10 Jahren}) = 0.3 \quad (436)$$
$$P(\text{Militarisierung}|\text{Durchbruch}) = 0.8 \quad (437)$$
$$P(\text{Konflikt}|\text{Militarisierung}) = 0.1 \quad (438)$$

Also: $P(\text{Eskalation}) = 0.3 \times 0.8 \times 0.1 = 0.024 = 2.4\%$
2.4% klingt klein, aber bei den möglichen Konsequenzen...

29.5 Quantenethik-Governance und Regulierung

Internationale Quantenkooperation: Prisoner's Dilemma in der Quantenforschung:

Payoff-Matrix für zwei Länder (Kooperation vs. Alleingang):

$$\begin{pmatrix} & \text{Koop.} & \text{Allein} \\ \text{Koop.} & (8,8) & (2,10) \\ \text{Allein} & (10,2) & (4,4) \end{pmatrix} \quad (439)$$

Nash-Gleichgewicht: (Allein, Allein) mit Payoff (4,4) Optimum: (Koop., Koop.) mit Payoff (8,8)

Ohne internationale Abkommen landen wir im schlechteren Gleichgewicht!

Quantenethik-Index: Bewerte Länder nach ethischer Quantenentwicklung:

$$QEI = 0.3 \times T + 0.2 \times S + 0.25 \times A + 0.15 \times E + 0.1 \times R \quad (440)$$

wo:

- T = Transparenz in Quantenforschung (0-10)
- S = SicherheitsmaSSnahmen (0-10)
- A = Zugangsgerechtigkeit (0-10)
- E = Umweltverantwortung (0-10)
- R = Regulierungsqualität (0-10)

Beispielrechnung für Deutschland:

$$QEI_{DE} = 0.3 \times 7 + 0.2 \times 8 + 0.25 \times 6 + 0.15 \times 8 + 0.1 \times 9 \tag{441}$$
$$= 2.1 + 1.6 + 1.5 + 1.2 + 0.9 = 7.3 \tag{442}$$

Langfristige ethische Herausforderungen: Wenn Quantencomputer allgegenwärtig werden (2050+):

1. **Quantenüberwachung:** Können autoritäre Regime perfekte Überwachung implementieren?
2. **Genetische Quantenanalyse:** Personalisierte Diskriminierung basierend auf DNA-Quantensimulation?
3. **Quanten-KI-Singularität:** Was passiert bei rekursiver Quanten-KI-Verbesserung?

Ethisches Dilemma - Das Quantenvorteil-Paradox: Je mächtiger Quantencomputer werden, desto gröSSer wird die Kluft zwischen den Habenden und Nicht-Habenden. Aber die Entwicklung zu stoppen würde bedeutende medizinische und wissenschaftliche Durchbrüche verhindern.

Mathematisch: Trade-off zwischen Nutzen U und Ungleichheit I:

$$\max_{Q} \quad U(Q) - \lambda \cdot I(Q) \tag{443}$$

wo Q = Quantenfortschritt, λ = Gewichtung der Gleichheit

Das ist das zentrale ethische Dilemma unserer Zeit! Wie löst man das? Internationale Quantengerechtigkeit? Open-Source-Quantencomputer? Universelles Quanteninernet?

Fazit: Die Ethik der Quantentechnologie ist mindestens genauso komplex wie die Technologie selbst. Wir brauchen dringend mathematische Modelle und internationale Kooperation, um die Vorteile zu maximieren und die Risiken zu minimieren.

Die Zukunft liegt in unseren Händen - aber nur, wenn wir jetzt die richtigen ethischen Grundlagen legen!

30 Literaturverzeichnis

Literatur

[1] Nielsen, M. A., & Chuang, I. L. (2010). *Quantum computation and quantum information*. Cambridge University Press.

[2] Wilde, M. M. (2013). *Quantum information theory*. Cambridge University Press.

[3] Preskill, J. (2018). Quantum computing in the NISQ era and beyond. *Quantum*, 2, 79.

[4] Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings 35th annual symposium on foundations of computer science*, 124-134.

[5] Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 212-219.

[6] Bennett, C. H., Brassard, G., Crépeau, C., Jozsa, R., Peres, A., & Wootters, W. K. (1993). Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical review letters*, 70(13), 1895.

[7] Steane, A. (1996). Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London*, 452(1954), 2551-2577.

[8] Wikipedia. Quantum computing. `https://en.wikipedia.org/wiki/Quantum_computing`

[9] Wikipedia. Quantenverschränkung. `https://de.wikipedia.org/wiki/Quantenverschr%C3%A4nkung`