Karlsruher Institut für Technologie

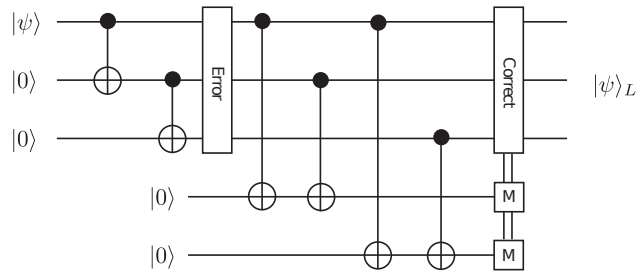
Quanten-Fehler-Korrektur

Hauptseminar *Physik des Quantencomputers*, SS 2013

Martin Koppenhöfer

Institut für Theoretische Festkörperphysik (TFP)

$$\mathcal{G} = \left\{ K^i \in \mathcal{P}_N \;\middle|\; \forall (i,j) : K^i \left|\psi\right\rangle_{\mathrm{L}} = \left|\psi\right\rangle_{\mathrm{L}} \wedge \left[K^i, K^j\right] = 0 \right\}$$



$$K^i E \left|\psi\right\rangle_{\mathrm{L}} = (-1)^m E K^i \left|\psi\right\rangle_{\mathrm{L}} = (-1)^m E \left|\psi\right\rangle_{\mathrm{L}}$$

Übersicht

Karlsruher Institut für Technologie

Wozu Fehlerkorrektur?

Klassische Fehlerkorrektur

- Formalismus
- 3-Bit-Code (klassisch)

Quantenmechanische Fehlerkorrektur

- Grundlagen
- 3-Bit-Code (quantenmechanisch)

Stabilizer-Formalismus

- Grundlagen
- Generatoren der Stabilizer-Gruppe

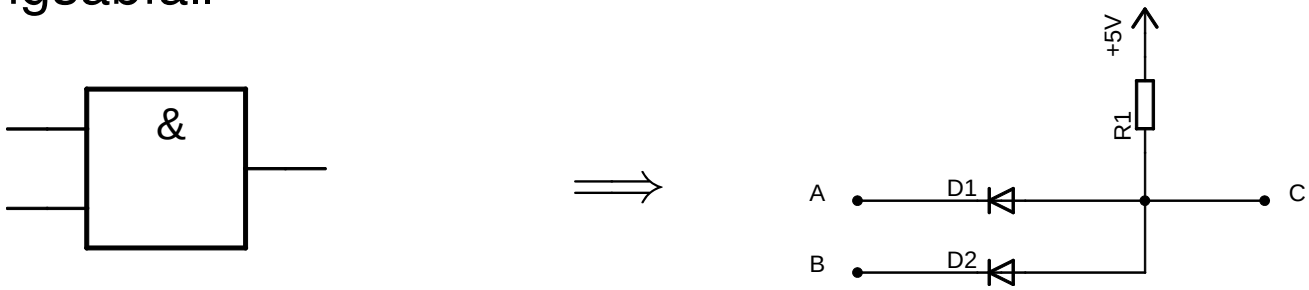
Threshold-Theorem

- Implementierung von Quantenalgorithmen
- Konkatenierte n-Qubit-Codes
- Bestimmung des threshold-value

Wozu Fehlerkorrektur?

Störeffekte in realen (Quanten)Computern:

- Spannungsabfall



Vorbereitungshilfe P1-Versuch Schaltlogik.

- Rauschen

$$S(\omega) = 2\frac{\hbar\omega}{R}\coth\frac{\hbar\omega}{2kT}$$

- Wechselwirkung mit der Umgebung (Relaxierung, Dephasierung)

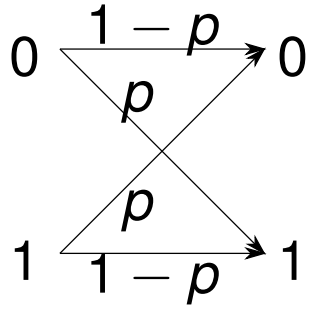
$$H_0 \longrightarrow H = H_0 + H_1$$

$\Rightarrow$ Fehler in der Datenverarbeitung

Klassische Fehlerkorrektur

Formalismus

$$\text{Fehlermodell} \xrightarrow{\text{führt zu}} \text{Codierung} \xrightarrow{\text{erlaubt}} \text{Fehlerkorrektur}$$



nach [KLM07], Fig. 10.1.

- Fehlermodell entwickeln
- Codierung führt Redundanzen ein:

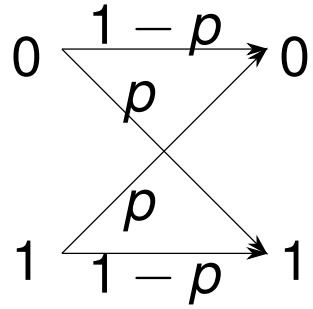
logisches Bit	+	Hilfsbits	=	codiertes Bit
1	+	*hh*	$\mapsto$	111

- Fehlerkorrektur ermittelt *error syndrome* und korrigiert Zustand.

Klassische Fehlerkorrektur

Formalismus

$$\text{Fehlermodell} \xrightarrow{\text{führt zu}} \text{Codierung} \xrightarrow{\text{erlaubt}} \text{Fehlerkorrektur}$$



nach [KLM07], Fig. 10.1.

- Fehlermodell entwickeln
- Codierung führt Redundanzen ein:

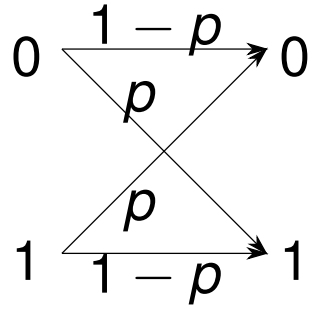
logisches Bit	$+$	Hilfsbits	$=$	codiertes Bit
1	$+$	*hh*	$\mapsto$	111

- Fehlerkorrektur ermittelt *error syndrome* und korrigiert Zustand.

Klassische Fehlerkorrektur

Formalismus

$$\text{Fehlermodell} \xrightarrow{\text{führt zu}} \text{Codierung} \xrightarrow{\text{erlaubt}} \text{Fehlerkorrektur}$$



nach [KLM07], Fig. 10.1.

- Fehlermodell entwickeln
- Codierung führt Redundanzen ein:

logisches Bit	$+$	Hilfsbits	$=$	codiertes Bit
1	$+$	*hh*	$\mapsto$	111

- Fehlerkorrektur ermittelt *error syndrome* und korrigiert Zustand.

Klassische Fehlerkorrektur

3-Bit-Code (klassisch)

- Bitwert in zwei Hilfsbits kopiert

$$0 \mapsto 000 \mapsto 000 \qquad\qquad 1 \mapsto 100 \mapsto 111$$

- korrigiert *einen* Bit-Fehler
- *alle* möglichen Fehler:

$$000 \mapsto \{\underbrace{000}_{(1-p)^3}, \underbrace{100, 010, 001}_{p(1-p)^2}, \underbrace{110, 101, 011}_{p^2(1-p)}, \underbrace{111}_{p^3}\}$$

- *error syndrome*: Vergleiche Bitwerte, abweichendes Bit fehlerhaft
- Wahrscheinlichkeit für nicht erkannten Fehler:

$$3 \cdot p^2(1-p) + p^3 \in \mathcal{O}(p^2) \qquad \text{(mit 3-Bit-Code)}$$

$$p \in \mathcal{O}(p) \qquad \text{(ohne 3-Bit-Code)}$$

Klassische Fehlerkorrektur

3-Bit-Code (klassisch)

- Bitwert in zwei Hilfsbits kopiert

$$0 \mapsto 000 \mapsto 000 \qquad\qquad 1 \mapsto 100 \mapsto 111$$

- korrigiert *einen* Bit-Fehler
- *alle* möglichen Fehler:

$$000 \mapsto \{ \underbrace{000}_{(1-p)^3}, \underbrace{100, 010, 001}_{p(1-p)^2}, \underbrace{110, 101, 011}_{p^2(1-p)}, \underbrace{111}_{p^3} \}$$

- *error syndrome*: Vergleiche Bitwerte, abweichendes Bit fehlerhaft
- Wahrscheinlichkeit für nicht erkannten Fehler:

$$3 \cdot p^2(1-p) + p^3 \in \mathcal{O}(p^2) \qquad \text{(mit 3-Bit-Code)}$$

$$p \in \mathcal{O}(p) \qquad \text{(ohne 3-Bit-Code)}$$

Klassische Fehlerkorrektur

3-Bit-Code (klassisch)

- Bitwert in zwei Hilfsbits kopiert

$$0 \mapsto 000 \mapsto 000 \qquad\qquad 1 \mapsto 100 \mapsto 111$$

- korrigiert *einen* Bit-Fehler
- *alle* möglichen Fehler:

$$000 \mapsto \{ \underbrace{000}_{(1-p)^3}, \underbrace{100, 010, 001}_{p(1-p)^2}, \underbrace{110, 101, 011}_{p^2(1-p)}, \underbrace{111}_{p^3} \}$$

- *error syndrome*: Vergleiche Bitwerte, abweichendes Bit fehlerhaft
- Wahrscheinlichkeit für nicht erkannten Fehler:

$$3 \cdot p^2(1-p) + p^3 \in \mathcal{O}(p^2) \qquad \text{(mit 3-Bit-Code)}$$

$$p \in \mathcal{O}(p) \qquad \text{(ohne 3-Bit-Code)}$$

Klassische Fehlerkorrektur

3-Bit-Code (klassisch)

- Bitwert in zwei Hilfsbits kopiert

$$0 \mapsto 000 \mapsto 000 \qquad\qquad 1 \mapsto 100 \mapsto 111$$

- korrigiert *einen* Bit-Fehler
- *alle* möglichen Fehler:

$$000 \mapsto \{\underbrace{000}_{(1-p)^3}, \underbrace{100, 010, 001}_{p(1-p)^2}, \underbrace{110, 101, 011}_{p^2(1-p)}, \underbrace{111}_{p^3}\}$$

- *error syndrome*: Vergleiche Bitwerte, abweichendes Bit fehlerhaft
- Wahrscheinlichkeit für nicht erkannten Fehler:

$$3 \cdot p^2(1-p) + p^3 \in \mathcal{O}(p^2) \qquad \text{(mit 3-Bit-Code)}$$

$$p \in \mathcal{O}(p) \qquad \text{(ohne 3-Bit-Code)}$$

Klassische Fehlerkorrektur

3-Bit-Code (klassisch)

- Bitwert in zwei Hilfsbits kopiert

$$0 \mapsto 000 \mapsto 000 \qquad\qquad 1 \mapsto 100 \mapsto 111$$

- korrigiert *einen* Bit-Fehler
- *alle* möglichen Fehler:

$$000 \mapsto \{\underbrace{000}_{(1-p)^3}, \underbrace{100, 010, 001}_{p(1-p)^2}, \underbrace{110, 101, 011}_{p^2(1-p)}, \underbrace{111}_{p^3}\}$$

- *error syndrome*: Vergleiche Bitwerte, abweichendes Bit fehlerhaft
- Wahrscheinlichkeit für nicht korrigierbaren Fehler:

$$3 \cdot p^2(1-p) + p^3 = \mathcal{O}(p^2) \qquad \text{(mit 3-Bit-Code)}$$

$$p = \mathcal{O}(p) \qquad \text{(ohne 3-Bit-Code)}$$

Quantenmechanische Fehlerkorrektur

Grundlagen: Unterschiede zur klassischen Fehlerkorrektur

- Messung des codierten Bits nicht erlaubt

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

- *no cloning*-Theorem

$$|\psi\rangle \not\to |\psi\rangle \otimes |\psi\rangle$$

- Bit-Flip *und* Phasenfehler möglich
- Fehler sind kontinuierlich

$$E = \exp\left(i\epsilon\sigma_x\right) = \cos(\epsilon)\,I + i\sin(\epsilon)\sigma_x$$

Quantenmechanische Fehlerkorrektur

Grundlagen: Unterschiede zur klassischen Fehlerkorrektur

- Messung des codierten Bits nicht erlaubt

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

- *no cloning*-Theorem

$$|\psi\rangle \not\to |\psi\rangle \otimes |\psi\rangle$$

- Bit-Flip *und* Phasenfehler möglich
- Fehler sind kontinuierlich

$$E = \exp\left(i\epsilon\sigma_x\right) = \cos(\epsilon)I + i\sin(\epsilon)\sigma_x$$

Quantenmechanische Fehlerkorrektur

Grundlagen: Vereinfachung

Fehlerfunktion wirkt auf *einzelnem* Qubit, d.h.:

- Fehler $= 2 \times 2$-Matrix
- Basis $\{I, \sigma_x, \sigma_y, \sigma_z\}$

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

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

$$iY = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$X|0\rangle = |1\rangle \qquad \text{(Bit-Flip)}$$
$$X|1\rangle = |0\rangle$$
$$Z|0\rangle = |0\rangle \qquad \text{(Phase-Flip)}$$
$$Z|1\rangle = -|1\rangle$$

- *vollständige* Quantencodes: simultane Korrektur von X und Z-Fehlern

Quantenmechanische Fehlerkorrektur

Grundlagen: Vereinfachung

Fehlerfunktion wirkt auf *einzelnem* Qubit, d.h.:

- Fehler $= 2 \times 2$-Matrix
- Basis $\{I, \sigma_x, \sigma_y, \sigma_z\}$

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

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

$$iY = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$X|0\rangle = |1\rangle \qquad \text{(Bit-Flip)}$$
$$X|1\rangle = |0\rangle$$
$$Z|0\rangle = |0\rangle \qquad \text{(Phase-Flip)}$$
$$Z|1\rangle = -|1\rangle$$

- *vollständige* Quantencodes: simultane Korrektur von X und Z-Fehlern

Quantenmechanische Fehlerkorrektur

Grundlagen: Vereinfachung

Fehlerfunktion wirkt auf *einzelnem* Qubit, d.h.:

- Fehler $= 2 \times 2$-Matrix
- Basis $\{I, \sigma_x, \sigma_y, \sigma_z\}$

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

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

$$iY = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$X|0\rangle = |1\rangle \qquad \text{(Bit-Flip)}$$
$$X|1\rangle = |0\rangle$$
$$Z|0\rangle = |0\rangle \qquad \text{(Phase-Flip)}$$
$$Z|1\rangle = -|1\rangle$$

- *vollständige* Quantencodes: simultane Korrektur von X und Z-Fehlern

Quantenmechanische Fehlerkorrektur

Grundlagen: Vereinfachung

Fehlerfunktion wirkt auf *einzelnem* Qubit, d.h.:

- Fehler $= 2 \times 2$-Matrix
- Basis $\{I, \sigma_x, \sigma_y, \sigma_z\}$

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

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

$$iY = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$X|0\rangle = |1\rangle \qquad \text{(Bit-Flip)}$$
$$X|1\rangle = |0\rangle$$
$$Z|0\rangle = |0\rangle \qquad \text{(Phase-Flip)}$$
$$Z|1\rangle = -|1\rangle$$

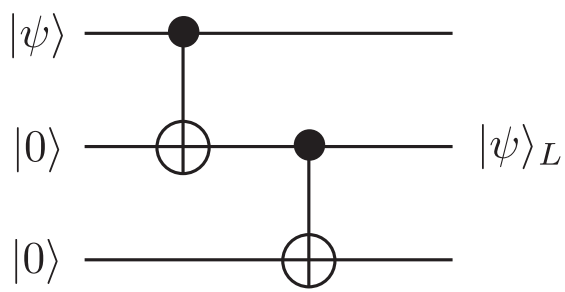
- *vollständige* Quantencodes: simultane Korrektur von X und Z-Fehlern

Quantenmechanische Fehlerkorrektur

3-Bit-Code (quantenmechanisch)

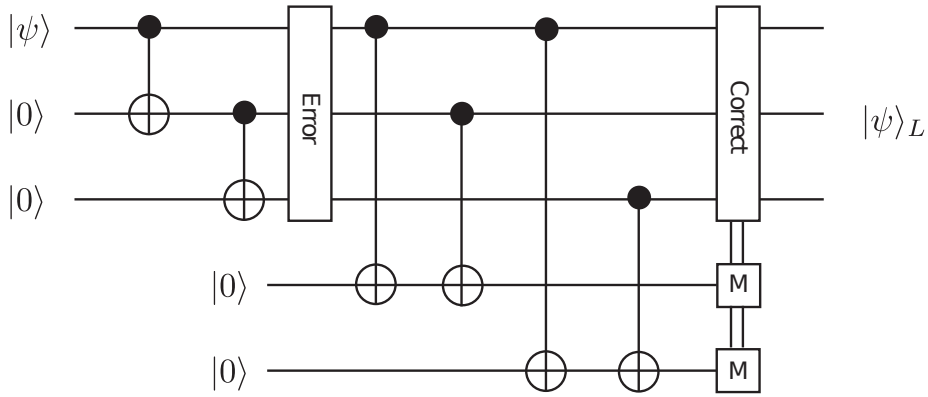
- *kein* vollständiger Quantencode
- Codierung:

$$|0\rangle \mapsto |0\rangle_{\mathrm{L}} = |0\rangle \otimes |0\rangle \otimes |0\rangle \equiv |000\rangle$$
$$|1\rangle \mapsto |1\rangle_{\mathrm{L}} = |1\rangle \otimes |1\rangle \otimes |1\rangle \equiv |111\rangle$$



[DN11], Fig. 2.

- *X error syndrome*: Parität der Bit-Paare vergleichen



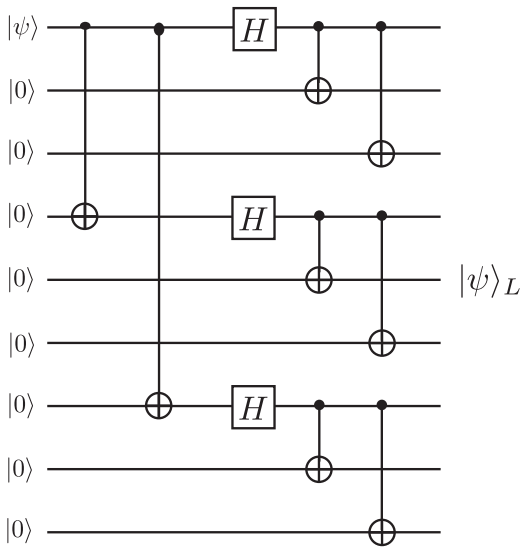
[DN11], Fig. 3.

Quantenmechanische Fehlerkorrektur

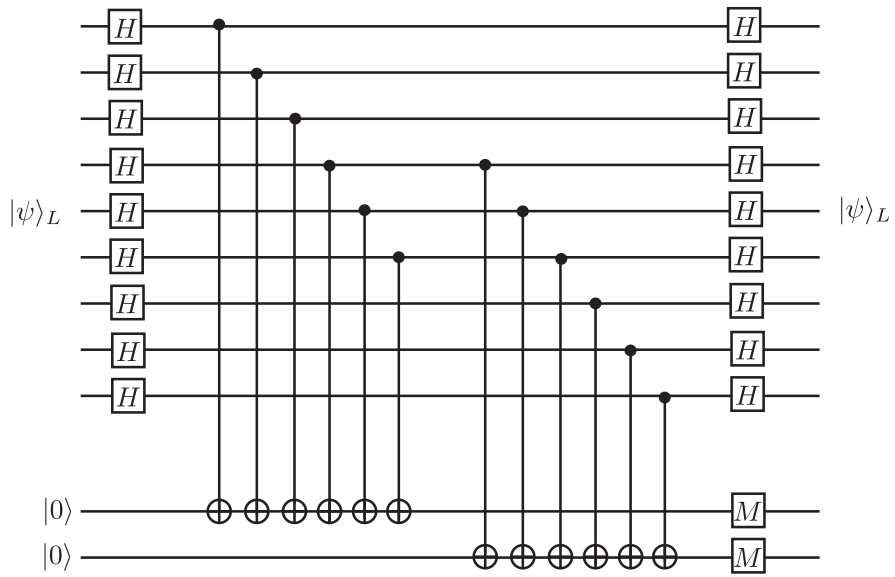
9-Qubit-Code (Shor-Code)

$$|0\rangle_{\mathrm{L}} = \frac{1}{\sqrt{8}}\left(|000\rangle + |111\rangle\right) \otimes \left(|000\rangle + |111\rangle\right) \otimes \left(|000\rangle + |111\rangle\right)$$

$$|1\rangle_{\mathrm{L}} = \frac{1}{\sqrt{8}}\left(|000\rangle - |111\rangle\right) \otimes \left(|000\rangle - |111\rangle\right) \otimes \left(|000\rangle - |111\rangle\right)$$



Codierung. [DN11], Fig. 4.



Korrektur von Z-Fehlern. [DN11], Fig. 5.

Stabilizer-Formalismus

Grundlagen

- Allgemeine Beschreibung der meisten Quanten-Fehler-Codes
- Konstruktionsregeln für Codierungs-, Korrektur- und Rechengatter, z.B:

$$|0\rangle_{\mathrm{L}} = \prod_{i=1}^{N-1} \left(I^{\otimes N} + K^i \right) |0\rangle^{\otimes N}$$

Idee:

- N Qubits $\Rightarrow \dim(\mathcal{H}) = 2^N$
- Wähle 2-dimensionalen Unterraum als logisches Qubit: $|0\rangle_{\mathrm{L}}$, $|1\rangle_{\mathrm{L}}$
- Fehler bringen $|0\rangle_{\mathrm{L}}$, $|1\rangle_{\mathrm{L}}$ in dazu orthogonale Unterräume

Stabilizer-Formalismus

Generatoren der Stabilizer-Gruppe

- Pauli-Gruppe (alle Elemente kommutieren oder antikommutieren):

$$\mathcal{P} = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}$$
$$\mathcal{P}_N = \mathcal{P}^{\otimes N}$$

- Generatoren-Gruppe mit $N-1$ Elementen:

$$\mathcal{G} = \left\{ K^i \in \mathcal{P}_N \;\middle|\; \forall (i,j) : K^i \left|\psi\right\rangle_{\mathrm{L}} = \left|\psi\right\rangle_{\mathrm{L}} \wedge \left[K^i, K^j\right] = 0 \right\}$$

- Fehler $E \in \mathcal{P}_N$ kommutieren oder antikommutieren mit K^i

$$K^i E \left|\psi\right\rangle_{\mathrm{L}} = (-1)^m E K^i \left|\psi\right\rangle_{\mathrm{L}} = (-1)^m E \left|\psi\right\rangle_{\mathrm{L}}$$
$$m = \begin{cases} 0 & \left[E, K^i\right] = 0 \\ 1 & \left\{E, K^i\right\} = 0 \end{cases}$$

Stabilizer-Formalismus

Stabilizer des 5-Bit-Codes

Fehler	$K^1(XZZXI)$	$K^2(IXZZX)$	$K^3(XIXZZ)$	$K^4(ZXIXZ)$
$XIIII$	1	1	1	-1
$IXIII$	-1	1	1	1
$IIXII$	-1	-1	1	1
$IIIXI$	1	-1	-1	1
$IIIIX$	1	1	-1	-1
$ZIIII$	-1	1	-1	1
$IZIII$	1	-1	1	-1
$IIZII$	1	1	-1	1
$IIIZI$	-1	1	1	-1
$IIIIZ$	1	-1	1	1
$IIIII$	1	1	1	1

Threshold-Theorem

Implementierung von Quantenalgorithmen

- Einzelnes Qubit: Fehlerwahrscheinlichkeit ϵ
- Quantenalgorithmus mit N Rechenschritten
- Falsches Ergebnis höchstens mit Wahrscheinlichkeit ϵ_{erg}

$\Rightarrow$ Fehlerwahrscheinlichkeit p pro Rechenschritt:

$$p \leq \frac{\epsilon_{\text{erg}}}{N}$$

Ansatz: n-Qubit-Code, der t Fehler korrigieren kann

$$p \propto \left(t^b \epsilon\right)^{t+1} \qquad (b \approx 3)$$

Threshold-Theorem

Implementierung von Quantenalgorithmen

- Einzelnes Qubit: Fehlerwahrscheinlichkeit ϵ
- Quantenalgorithmus mit N Rechenschritten
- Falsches Ergebnis höchstens mit Wahrscheinlichkeit ϵ_{erg}

$\Rightarrow$ Fehlerwahrscheinlichkeit p pro Rechenschritt:

$$p \leq \frac{\epsilon_{\text{erg}}}{N}$$

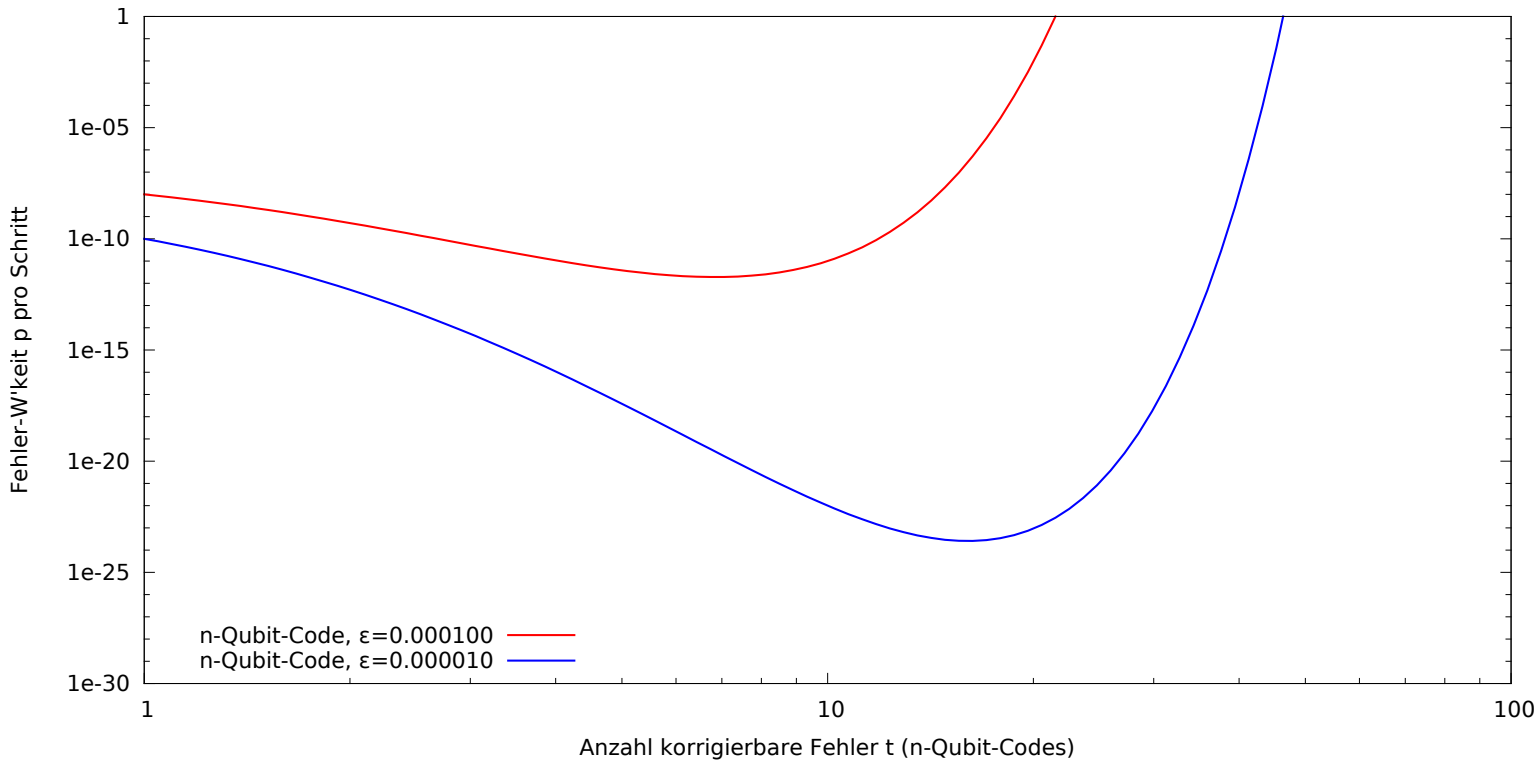
Ansatz: n-Qubit-Code, der t Fehler korrigieren kann

$$p \propto \left(t^b \epsilon\right)^{t+1} \qquad (b \approx 3)$$

Threshold-Theorem

Problem der *n*-Qubit-Codes

$$p \propto \left(t^b \epsilon\right)^{t+1} \quad \overset{t \gg 1}{\Longrightarrow} \quad p_{\min} \approx \exp\left(-\frac{b}{e \cdot \sqrt[b]{\epsilon}}\right)$$



Threshold-Theorem

Konkatenierte *n*-Qubit-Codes

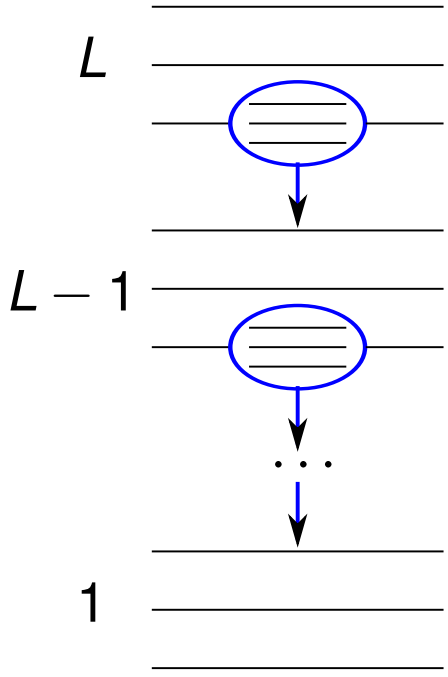
- Stelle jedes Qubit im n-Qubit-Code durch einen n-Qubit-Code dar
- Pro Ebene selbstähnliche Fehlerkorrektur (korrigiert $t = 1$ Fehler)

$$p^{(1)} = c \cdot \epsilon^2$$

$$p^{(2)} = c(p^1)^2 = \frac{(c \cdot \epsilon)^4}{c}$$

$$\vdots$$

$$p^{(L)} = \frac{(c \cdot \epsilon)^{2^L}}{c}$$

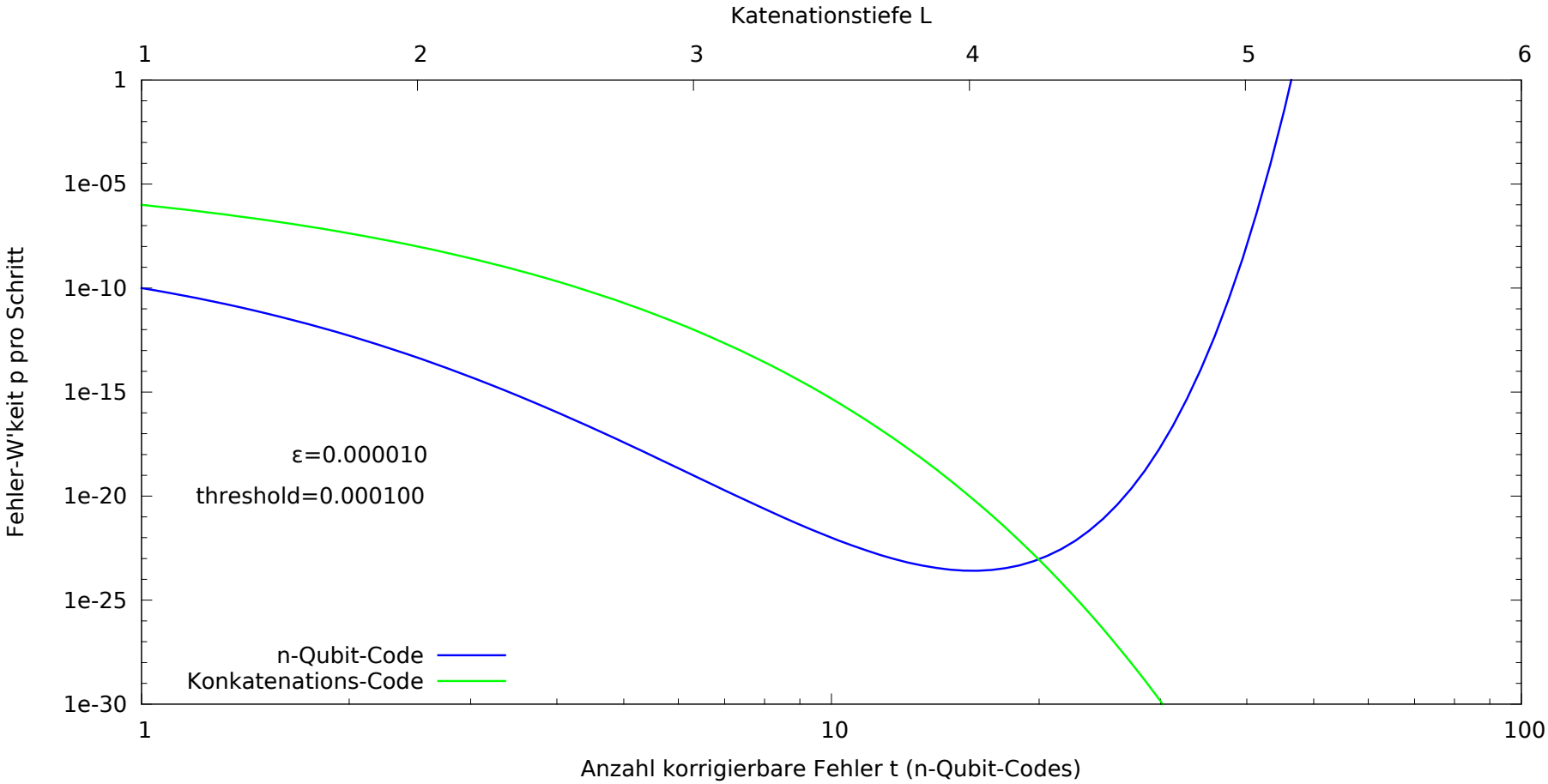


nach [Pre98], Fig. 13.

- Falls $\epsilon < \frac{1}{c} \equiv p_{\text{threshold}}$: $\lim_{L\to\infty} p^{(L)} = 0$

Threshold-Theorem

Konkatenierte *n*-Qubit-Codes



Threshold-Theorem

Bestimmung des threshold-value

Aufstellung von Flussgleichungen basierend auf:

- Algorithmus
- Fehlermodell
- Fehlerfortpflanzung in Gattern
- Quantencode

Näherungsweise Lösung für Shor-Algorithmus:

$$p_{\text{threshold}} \approx \begin{cases} 10^{-4} & \text{ohne Speicherfehler} \\ 10^{-5} & \text{mit Speicherfehlern} \end{cases}$$

Übersicht

Karlsruher Institut für Technologie

Wozu Fehlerkorrektur?

Klassische Fehlerkorrektur

- Formalismus
- 3-Bit-Code (klassisch)

Quantenmechanische Fehlerkorrektur

- Grundlagen
- 3-Bit-Code (quantenmechanisch)

Stabilizer-Formalismus

- Grundlagen
- Generatoren der Stabilizer-Gruppe

Threshold-Theorem

- Implementierung von Quantenalgorithmen
- Konkatenierte n-Qubit-Codes
- Bestimmung des threshold-value

Literatur

Simon Devitt and Kae Nemoto.
Quantum Error Correction for Beginners.
arXiv:quant-ph/09052794v3, 2011.

Phillip Kaye, Raymond Laflamme, and Michele Mosca.
An introduction to quantum computing.
Oxford University Press, Oxford [u.a.], 1. publ. edition, 2007.

John Preskill.
Reliable quantum computers.
Proc.Roy.Soc.Lond., A454:385–410, 1998.