Ein Vergleich verschiedener Varianten von endlichen Quantenautomaten

Stephan Sigg

Diplomarbeit
am Fachbereich Informatik
der Universität Dortmund

August 2004

Gutachter

PD Dr. Detlef Sieling
PD Dr. Martin Sauerhoff

Stephan Sigg
Hessische Straße 165
44339 Dortmund

Matrikel-Nummer: 78302
Fachbereich Informatik
Universität Dortmund

Ein Vergleich verschiedener Varianten von endlichen Quantenautomaten

Stephan Sigg

Inhaltsverzeichnis

Einleitung

If computers that you build are quantum,
Then spies everywhere will all want 'em.
Our codes will all fail,
And they'll read our email,
Till we get crypto that's quantum, and daunt 'em.
- Jennifer und Peter Shor

Nachdem zu Beginn des zwanzigsten Jahrhunderts vermehrt Widersprüche und Fehlaussagen in den Vorhersagen der klassischen Physik für Ergebnisse der aktuellen Forschung auftraten, wurde in den folgenden Jahren die Quantenphysik als die akuratere Beschreibung der Wirklichkeit entwickelt. Nicht wenige Physiker konnten sich in den ersten Jahren nicht mit den teilweise revolutionären Vorhersagen der Quantenphysik anfreunden. Viele Vorhersagen konnten jedoch mittlerweile experimentell bestätigt werden, sodass wir aus heutiger Sicht keinen Grund haben, die Quantenphysik nicht als die korrekte Beschreibung unserer Umwelt anzusehen. Eine Anwendung in der Informatik ist bis in die Achtziger Jahre nicht vorgeschlagen worden. Erst 1982 wies der Physiker Richard Feynman darauf hin, dass klassische Rechner vermutlich große Schwierigkeiten haben, physikalische Systeme zu simulieren, die durch die Quantenphysik beschrieben werden. Im Jahr 1985 entwickelte David Deutsch ein formales Modell für einen Rechner, mit dem beliebige physikalische Systeme effizient simuliert werden können ([Deu85]). Seither wurden rasante Fortschritte in dem Bereich der Quantenschaltkreise und der Quantencomputer erzielt. Als prominenteste Beispiele seien hier der Algorithmus zum Faktorisieren großer Zahlen in Polynomialzeit von Peter Shor [Sho97] sowie der Algorithmus von Lov K. Grover [Gro96] zum Suchen in großen Datenbanken genannt. Beide Algorithmen erreichen eine erhebliche Beschleunigung im Vergleich zu bekannten Algorithmen für die angegebenen Probleme.

Derzeit ist die Theorie der Praxis in diesem Bereich um Längen voraus. Quantencomputer, die mit mehr als nur einer Handvoll Qubits rechnen, können praktisch nicht realisiert werden. Es ist zu erwarten, dass die ersten Quantenrechner nur mit wenigen Qubits rechnen, auf die sie bei Bedarf zurückgreifen. Wir suchen also zunächst ein einfaches Rechnermodell, das wir mit den Gesetzmäßigkeiten der Quantentheorie beschreiben können. Wir werden dem Modell ein kleines Register aus Qubits spendieren, mit dem dann die Berechnungen durchgeführt werden können. Es stellt sich die Frage, ob die Vorteile, die der Gebrauch von Qubits bei Quantenschaltkreisen bringt, auch schon in diesem einfachen Modell erkennbar sind oder ob das Modell durch die Verwendung von Qubits behindert wird. Ein einfa-

ches Rechnermodell ist das Modell der endlichen Automaten. Das Modell eines endlichen Quantenautomaten wurde 1997 von Attila Kondacs und John Watrous [KW97] vorgestellt sowie unabhängig davon in [MC00]. Die beiden Definitionen unterscheiden sich leicht und es stellt sich heraus, dass die Automaten aus [MC00] von Automaten nach [KW97] simuliert werden können. In [KW97] werden Automaten vorgestellt, die den Kopf in eine Richtung bewegen können (*1-QFA*s) und Automaten, die den Kopf in verschiedene Richtungen bewegen können (*2-QFA*s). Während *2-QFA*s mehr Sprachen erkennen können, als alle klassischen Automaten, stellt sich heraus, dass *1-QFA*s nur eine echte Teilmenge der regulären Sprachen erkennen können. Der entscheidende Nachteil bei dem Modell der *2-QFA*s ist, dass abhängig von der Wortlänge unterschiedlich viele Qubits für die Berechnung benötigt werden. Aufbauend auf der Arbeit von Kondacs und Watrous gibt es verschiedene Arbeiten in denen *1-QFA*s untersucht werden. In [AF98] wird festgestellt, dass die Menge der von *1-QFA*s erkannten Sprachen mit der verwendeten Akzeptanzwahrscheinlichkeit des Automaten variiert. In [ABFK99] wird dazu eine Hierarchie der von *1-QFA*s erkannten Sprachen in Abhängigkeit von der verwendeten Akzeptanzwahrscheinlichkeit vorgestellt. Außerdem werden in [AF98] einige Modifikationen vorgeschlagen, wie das Modell des *1-QFA*s so verändert werden kann, dass mehr als nur eine Teilmenge der regulären Sprachen erkannt werden kann. Wir werden diese vorgeschlagenen Automaten in Kapitel 4 untersuchen. Dort greifen wir zusätzlich eine Idee aus [Hir01] auf, bei der die Messungen des *1-QFA*s verändert werden, wodurch wir größere Freiheiten bei der Konstruktion von *QFA*s erhalten. Wir stellen Sprachen vor, die von diesen Automaten erkannt werden können und geben, wenn möglich, die Menge der Sprachen an, die von den Automaten erkannt werden können. Dabei versuchen wir ein Modell zu finden, das mehr Sprachen erkennen kann als alle klassischen Automaten, das jedoch anders als der *2-QFA* nur konstant viele Qubits benötigt. Im Einzelnen betrachten wir in Kapitel 4.1 einen *1-QFA* mit verallgemeinerten Messungen, untersuchen in den Kapiteln 4.1.1 und 4.2 wie sich die Verwendung eines zyklischen Eingabebandes auswirkt und analysieren in Kapitel 4.3 ob eine Kombination von *1-QFA*s mit klassischen Automatentypen dazu führt, dass mehr Sprachen erkannt werden können.

Wir werden uns zunächst in Kapitel 1 einen kleinen Überblick über die Möglichkeiten und auch die Schwierigkeiten der Gesetzmäßigkeiten der Quantentheorie verschaffen. Wir werden feststellen, dass das mathematische Berechnungsmodell der Quantentheorie über einige überraschende und manchmal hinderliche Berechnungsvorschriften verfügt. Außerdem werden wir uns in Kapitel 1 eine kleine Sammlung an probabilistischen Methoden aneignen, um sie in späteren Kapiteln bei der Analyse von verschiedenen Automatentypen zu verwenden.

In Kapitel 2 werden das einfache Modell endlicher Automaten und einige klassische Automatentypen, die wir in späteren Kapiteln den Quantenautomaten gegenüberstellen werden, vorgestellt.

Kapitel 3 führt dann *1-QFA*s und *2-QFA*s zusammen mit den für uns interessanten Ergebnissen ein. Die in Kapitel 4 vorgestellten Automatentypen werden durch kleine Modifikationen an der Definition des *1-QFA*s aus diesem Modell hervorgehen.

Bezeichnungen und Symbole

Die auf der folgenden Seite eingeführten Bezeichnungen werden, sofern möglich, in der gesamten Arbeit verwendet. An einigen Stellen werden zusätzlich Indizes benutzt. Es wird in der Regel die in der Literatur gängige Notation verwendet, wenn dies für die Lesbarkeit sinnvoll erscheint. Die knappe Auflistung der Notationen auf diesen beiden Seiten kann nur eine stichwortartige Beschreibung sein und sollte primär zum Nachschlagen während der Lektüre dienen. Sonstige Definitionen werden an geeigneter Stelle definiert und sind über das Stichwortverzeichnis am Ende der Arbeit zu finden.

Ergebnisse aus anderen Arbeiten werden ohne Beweis zitiert. Der zugehörige Beweis kann unter der aufgeführten Quelle nachgelesen werden. Wenn Mengen, Matrizen und verschiedene Automaten betrachtet werden, sind die Bezeichner in der Regel so gewählt, dass eine einfache einheitliche Identifizierung durch Großbuchstaben möglich ist. Matrizen werden die Buchstaben M, I, O bzw. B zugeordnet und Mengen die Bezeichner U, V, E, K. Bei Automaten unterscheiden wir zwischen dem Automatentyp. Probabilistische Automaten (*PFA*s) heißen P, deterministische Automaten (*DFA*s) dagegen D, reversible Automaten (*RFA*s) R und Quantenautomaten (*QFA*s) schließlich A. Sprachen werden druch den Bezeichner L angesprochen. Eine Sprache, die in verschiedenen Kapitlen behandelt wird, ist mit einem eindeutigen Index versehen. Für zwei Buchstaben a und b sind dies die Sprachen

$$\begin{aligned} L_{ab} &= \{a^*b^*\}, \\ L_{eq} &= \{a^n b^n | n \in \mathbb{N}\}, \\ L_{pal} &= \{w | w \in \{a,b\}^* \text{ mit } w = w^{-1}\} \text{ und} \\ L_{a=b} &= \{w | w \in \{a,b\}^* \text{ mit } |w|_a = |w|_b\}. \end{aligned}$$

$\mathbb{R}^+$	Die Menge der positiven reellen Zahlen inklusive der Null.
$G = (V, E)$	Ein Graph mit der Knotenmenge V und der Kantenmenge E.
$v \in V$	Ein Knoten aus der Knotenmenge V eines Graphen.
$d(v)$	Der Grad bzw. Ausgangsgrad von einem Knoten v in einem Graphen.
$\|l\rangle$	Der Spaltenvektor l in Dirac-Notation.
$\langle l\|$	Der Zeilenvektor l in Dirac-Notation.
$\|r\rangle$	Ein Vektor aus den reellen Zahlen.
$\|p\rangle$	Ein Vektor aus den den positiven reellen Zahlen.
$\|z\rangle$	Ein Vektor aus den ganzen Zahlen.
$\|q\rangle$	Ein Vektor aus den komplexen Zahlen.
$\|i\rangle$	Der i-te Einheitsvektor.
$\langle i\|j\rangle$	Das Skalarprodukt zwischen $\|i\rangle$ und $\|j\rangle$.
$\|i\rangle\langle j\|$	Das äußere Produkt zwischen $\|i\rangle$ und $\|j\rangle$.
$\text{span}\{U\}$	Der von den Vektoren aus U aufgespannte Vektorraum.
σ	Ein Buchstabe.
w	Ein Wort. Ein Wort ist eine Konkatenation von $k \geq 0$ Buchstaben.
w_i	Für $i \in \{1, \ldots, \|w\|\}$ der i-te Buchstabe des Wortes w.
$\|w\|$	Die Länge einer Eingabe w. In der Regel ist $\|w\| = n$.
w^{-1}	Sei $w = w_1 \ldots w_n$. Dann ist $w^{-1} = w_n \ldots w_1$.
ε	Das leere Wort bzw. die Fehlerwahrscheinlichkeit eines Automaten.
y	Eine Ausgabe eines Automaten.
Σ	Ein Alphabet. Ein Alphabet ist eine Menge von Buchstaben.
Σ^*	Die Menge aller Wörter über dem Alphabet Σ.
Σ^n	Die Menge aller Wörter w mit $\|w\| = n$ über dem Alphabet Σ.
L	Eine Sprache. Eine Sprache ist eine Menge von Wörtern über einem Alphabet Σ.
Γ	Das Bandalphabet eines Automaten.
Q	Die Zustandsmenge eines Automaten.
q_0	Der Startzustand eines Automaten.
q	Ein Zustand eines Automaten.
q_{acc}	Ein akzeptierender Zustand.
q_{rej}	Ein verwerfender Zustand.
δ	Die Überführungsfunktion eines Automaten.
M_σ	Die Überführungsmatrix eines Automaten zu dem Zeichen σ.
I	Die Einheitsmatrix. In der Regel von der Dimension $n \times n$.
$\|\|\|q\rangle\|\|_k$	Die L_k-Norm des Spaltenvektors q.
$\|U\|$	Die Kardinalität einer Menge U.
$\mathcal{P}[R]$	Die Wahrscheinlichkeit für ein Ereignis R.
$(d)^+$	Die positiven Teile aus dem Term d.
$(d)^-$	Die negativen Teile aus dem Term d.
d^*	Das komplex Konjugierte zu d.
$M^\dagger$	Die zu M adjungierte Matrix.

Kapitel 1

Grundlagen und Methoden

Die in diesem Kapitel vorgestellten quantentheoretischen Grundlagen können, wenn nicht anders angegeben, in [NC02] nachgeschlagen werden. Zunächst werden wir die verwendete Notation erläutern.

1.1 Schreibweise

Ein Vektorraum wird von einer Menge von Vektoren $\psi_1, \ldots, \psi_n$ aufgespannt. Wir schreiben auch $\text{span}\{\psi|\psi \in U\}$, wenn wir den von den Vektoren aus der Menge U aufgespannten Vektorraum meinen. Als Vektorschreibweise wählen wir in Anlehnung an die gebräuchliche Notation in der Literatur die Dirac-Schreibweise. Dabei beschreibt $|\psi\rangle \in \mathbb{C}^n$ den n-dimensionalen Spaltenvektor $(\psi_1, \ldots, \psi_n)^T$ mit $\psi_1, \ldots, \psi_n \in \mathbb{C}$. Den zugehörigen Zeilenvektor bezeichnen wir mit $\langle\psi| = |\psi\rangle^\dagger$, wobei jeder Eintrag von $\langle\psi|$ zu dem entsprechenden Eintrag aus $|\psi\rangle$ komplex Konjugiert ist. Wir schreiben das Skalarprodukt zweier Vektoren $|\psi\rangle, |\psi'\rangle$ als $\langle\psi|\psi'\rangle$. Zwei Vektoren $|\psi\rangle, |\psi'\rangle$ sind zueinander orthogonal, wenn $\langle\psi|\psi'\rangle = 0$ gilt. In einem n-dimensionalen Vektorraum definieren wir für $i, j \in \{1, \ldots, n\}$ den Spaltenvektor $|i\rangle$ als den Einheitsvektor mit

$$\langle i|j\rangle = \begin{cases} 1, & \text{falls } i = j, \\ 0, & \text{sonst.} \end{cases}$$

Wenn wir auf einzelne Einträge in einem Spaltenvektor $|\psi\rangle \in \mathbb{C}^n$ zugreifen möchten, können wir also $\langle i|\psi\rangle$ mit $i \in \{1, \ldots, n\}$ schreiben. Damit wird die i-te Zeile in $|\psi\rangle$ angesprochen. Es ist $\sum_{i=1}^{n} |i\rangle\langle i|\psi\rangle = |\psi\rangle$.

Definition 1.1-1
Seien $|\psi_1\rangle \in \mathbb{C}^m$ *und* $|\psi_2\rangle, |\psi_3\rangle \in \mathbb{C}^n$ *beliebige Vektoren. Dann bezeichnen wir die Abbildung* $|\psi_1\rangle\langle\psi_2| : \mathbb{C}^n \to \mathbb{C}^m$ *mit*

$$(|\psi_1\rangle\langle\psi_2|)\,|\psi_3\rangle := |\psi_1\rangle\langle\psi_2|\psi_3\rangle$$

als das äußere Produkt von $|\psi_1\rangle$ *und* $|\psi_2\rangle$.

Mit der Norm eines Vektors meinen wir in der Regel die L_k-Norm. Sei $|\psi\rangle$ ein Vektor der Dimension n. Wir schreiben die L_k-Norm von $|\psi\rangle$ als $|||\psi\rangle||_k$ und meinen damit

$$\sqrt[k]{\sum_{i=1}^{n} |\langle i|\psi\rangle|^k}.$$

Eine Basis eines Vektorraumes C^n ist eine Menge U von Vektoren mit $C^n = \text{span}\{|\psi\rangle||\psi\rangle \in U\}$. Jeder Punkt in dem Vektorraum C^n kann dann als eine Linearkombination aller Vektoren $|\psi\rangle$ mit $|\psi\rangle \in U$ beschrieben werden. Es gibt verschiedene Basen zu einem Vektorraum. Eine Menge von Vektoren $|\psi_1\rangle, \ldots, |\psi_n\rangle \in \mathbb{C}^n$ bilden eine orthonormale Basis (ON-Basis) eines Vektorraumes, falls die L_2-Norm von allen Vektoren den Wert 1 hat und paarweise verschiedene Vektoren orthogonal zueinander sind. Wenn nicht anders erwähnt, setzen wir die ON-Basis $|1\rangle, \ldots, |n\rangle$ voraus.

1.1.1 Einige nützliche Werkzeuge

Wir werden in diesem Abschnitt kurz einige elementare Methoden vorstellen, die sich in späteren Kapiteln als nützlich erweisen werden. Die beiden folgenden Abschätzungen werden Chernoff-Schranken genannt. Wir werden Chernoff-Schranken benutzen, um für Ereignisse, von denen wir nur den Erwartungswert kennen, zu zeigen, dass sie mit großer Wahrscheinlichkeit nur um einen kleinen Faktor von ihrem Erwartungswert abweichen.

Satz 1.1-2 *(Theorem* 4.1 *in [MR00])*
Seien $X_1, \ldots, X_n$ *unabhängige 0-1 Zufallsvariablen, sodass für* $i \in [1, n], \mathcal{P}[X_i = 1] = p_i$, $i \in \{1, \ldots, n\}$, *gilt. Dabei ist* $p_i \in]0, 1[$ *für alle* i. *Dann ist für* $X = \sum_{i=1}^{n} X_i$, $\mu = E[X] = \sum_{i=1}^{n} p_i$ *und* $\delta > 0$,

$$\mathcal{P}[X > (1+\delta)\mu] < \left[\frac{e^\delta}{(1+\delta)^{1+\delta}}\right]^\mu.$$

Satz 1.1-3 *(Theorem* 4.2 *in [MR00])*
Seien $X_1, \ldots, X_n$ *unabhängige 0-1 Zufallsvariablen, sodass für* $i \in [1, n], \mathcal{P}[X_i = 1] = p_i$ *mit* $p_i \in]0, 1[$ *gilt. Dann ist für* $X = \sum_{i=1}^{n} X_i, \mu = E[X] = \sum_{i=1}^{n} p_i$ *und* $\delta \in]0, 1]$,

$$\mathcal{P}[X < (1-\delta)\mu] < e^{-\frac{\mu\delta^2}{2}}.$$

Das folgende Ergebnis aus der Wahrscheinlichkeitstheorie kann zum Beispiel in [Fel68] nachgelesen werden. Es wird in der Literatur häufig als das *Gamblers-Ruin Problem* bezeichnet. Wir betrachten zwei Spieler S_1 und S_2, die wiederholt gegeneinander antreten. Beide verfügen über ein Startkapital von a bzw. $n-a$ Geldeinheiten. Spieler S_1 gewinnt in einem Spiel mit Wahrscheinlichkeit $\mathcal{P}_1$ und Spieler S_2 mit Wahrscheinlichkeit $\mathcal{P}_2 = 1 - \mathcal{P}_1$. Der verlierende Spieler gibt eine seiner Geldeinheiten an den gewinnenden Spieler ab. Danach wird erneut gespielt, bis schließlich einer der Spieler all sein Geld verloren hat. Er ist damit ruiniert. Es stellt sich die Frage, mit welcher Wahrscheinlichkeit Spieler S_1 am Ende ruiniert ist. Diese Wahrscheinlichkeit nennen wir q_a, wenn Spieler S_1 zu Beginn der Betrachtung über a Geldeinheiten verfügt. Wenn wir $q_0 = 1$ und $q_n = 0$ setzen, können wir allgemein schreiben $q_a = \mathcal{P}_1 q_{a+1} + \mathcal{P}_2 q_{a-1}$. Außerdem sind wir an der erwarteten Dauer interessiert, bis einer der Spieler ruiniert ist, wenn a das Startkapital von Spieler S_1 ist. Wir bezeichnen diese Größe mit D_a und können allgemein schreiben $D_a = \mathcal{P}_1 D_{a+1} + \mathcal{P}_2 D_{a-1} + 1$. Für $\mathcal{P}_1 = \mathcal{P}_2 = \frac{1}{2}$ kann man zeigen, dass

$$q_a = 1 - \frac{a}{n} \text{ und} \tag{1.1}$$

$$D_a = a(n-a) \tag{1.2}$$

gilt (siehe dazu [Fel68]). Andere Werte für $\mathcal{P}_1$ und $\mathcal{P}_2$ sind für uns nicht interessant. Um angeben zu können, wie lange ein Spiel dauert, in dem ein bestimmter Spieler gewinnt, machen wir einen kleinen Ausflug in die Graphentheorie.

Definition 1.1-4
Sei $G = (V, E)$ ein beliebiger ungerichteter Graph und $v \in V$ ein Knoten in dem Graphen. Dann bezeichnet $d(v)$ den Grad des Knotens v.

Definition 1.1-5
Sei $G = (V, E)$ ein beliebiger gerichteter Graph und $v \in V$ ein Knoten in dem Graphen. Dann bezeichnet $d(v)$ den Ausgangsgrad des Knotens v.

Definition 1.1-6
Sei $G = (V, E)$ ein zusammenhängender, ungerichteter Graph. Ein Random-Walk auf G wird durch ein Teilchen beschrieben, das zufällig auf den Knoten des Graphen umherwandert. Den Übergang des Teilchens von einem Knoten zu einem Nachbarknoten nennen wir einen Schritt. Für $v \in V$ erreicht das Teilchen ausgehend von v einen der Nachbarknoten von v mit Wahrscheinlichkeit $\frac{1}{d(v)}$.

Abbildung 1.1: Darstellung des *Random-Walks* auf der Linie der Länge $n+1$ durch einen ungerichteten Graphen $G' = (V', E')$.

Definition 1.1-7
Sei $G = (V, E)$ ein zusammenhängender, ungerichteter Graph und $v \in V$ ein Knoten in diesem Graphen. $C_v(G)$ bezeichnet die erwartete Dauer, die ein Random-Walk mit Start- und Endknoten v benötigt, bis jeder Knoten von G mindestens einmal erreicht ist. Es ist $C(G) = \max\{C_v(G) | v \in V\}$.

Satz 1.1-8 *(Theorem 6.8 in [MR00])*
Sei $G = (V, E)$ ein zusammenhängender, ungerichteter Graph, dann ist die maximale erwartete Dauer eines Random Walks auf dem Graphen $C(G) \leq 2|E|(|V| - 1)$.

Definition 1.1-9
Sei $G = (V, E)$ mit $|V| = n + 1, |E| = n$ der ungerichtete Graph aus Abbildung 1.1. Den Prozess eines Teilchens, das in dem Knoten 1 startet und in jedem Schritt mit Wahrscheinlichkeit $\frac{1}{d(v)}$ von einem Knoten v zu einem der Nachbarknoten von v wandert, bezeichnen wir als Random-Walk auf der Linie der Länge $n + 1$.

Da wir einen *Random-Walk* in diskreten Zeitschritten betrachten können, bezeichnen wir die Anzahl der Schritte, die in einem *Random-Walk* auf der Linie der Länge n startend in Knoten 1 absolviert werden, bis der Knoten n erreicht wird, als die Dauer des Prozesses.

Lemma 1.1-10
Der Erwartungswert für die Dauer eines Random-Walks auf der Linie der Länge n ist höchstens $2n(n-1) = O(n^2)$.

Beweis. Wir erhalten die Behauptung direkt aus Satz 1.1-8. □

Bemerkung 1.1-11
Für den Prozess des Random-Walks auf der Linie der Länge n kann sogar gezeigt werden, dass die erwartete Dauer des Prozesses $\Theta(n^2)$ ist (siehe [MR00]).

Satz 1.1-12
Seien S_1 und S_2 zwei Spieler bei dem Gamblers-Ruin Problem mit Startkapital 1 bzw. $n-1$

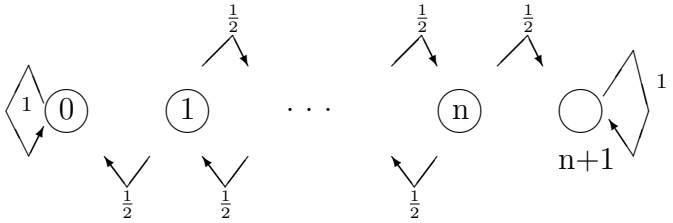

Abbildung 1.2: Darstellung des Prozesses des *Gamblers-Ruin Problems* als die Bewegung eines Teilchens auf einem Graphen $G = (V, E)$.

sowie $\mathcal{P}_1 = \mathcal{P}_2 = \frac{1}{2}$ *die Wahrscheinlichkeiten, mit denen die Spieler gegeneinander gewinnen. Die erwartete Dauer eines Spiels, in dem der Spieler* S_2 *ruiniert wird, ist* $D_{S_2} \leq 2n^2$.

Beweis. Wir stellen den Prozess, der dem *Gamblers-Ruin Problem* zu Grunde liegt, durch die zufällige Bewegung eines Teilchens auf einem gerichteten, kantengewichteten Graphen $G = (V, E)$ dar. Es ist $|V| = n + 2$ und $|E| = 2(n + 2)$ (vgl. Abbildung 1.2). Wir stellen uns ein Teilchen vor, das auf diesem Graphen entlang der Kanten, jedoch nie entgegengesetzt der Kantenrichtung, auf den Knoten umherwandert. Die Position des Teilchens auf den Knoten spiegelt die Vermögenssituation von Spieler S_1 wider. Wenn sich das Teilchen an Knoten i befindet, verfügt der Spieler S_1 über i Geldeinheiten. Die Beschriftung der Kanten entspricht den jeweiligen Übergangswahrscheinlichkeiten zwischen den Knoten. Für $v, w \in V$ mit $(v, w) \in E$ ist die Übergangswahrscheinlichkeit, um über eine Kante vom Knoten v zu dem Knoten w zu gelangen $\frac{1}{d(v)}$. Die Übergangswahrscheinlichkeit ist fast überall $\frac{1}{2}$. Lediglich die Endknoten, wenn also einer der beiden Spieler ruiniert ist, können von dem Teilchen nicht verlassen werden.

Wenn wir ausschließen, dass der Spieler S_1 ruiniert wird, können wir dies durch einen ähnlichen Graphen $G^{S_2} = (V^{S_2}, E^{S_2})$ mit $|V^{S_2}| = n + 1, |E^{S_2}| = 2n + 1$ beschreiben. Der Graph G^{S_2} unterscheidet sich von G dadurch, dass der Knoten mit der Beschriftung 0 aus der Knotenmenge und die Kante $(1, 0)$ aus der Kantenmenge entfernt wurden. Außerdem ist die Beschriftung der Kante $(1, 2)$ anstatt $\frac{1}{2}$ in dem Graphen G^{S_2} nun 1. Die Wahrscheinlichkeit, dass das Teilchen den Knoten mit der Beschriftung 0 erreicht, ist also gleich Null, und die Wahrscheinlichkeit, von Knoten 1 aus den Knoten 2 zu erreichen, ist gleich Eins. Die sonstigen Kanten und Kantenbeschriftungen sind identisch zu den entsprechenden Kanten und Beschriftungen in dem Graphen G. Der Graph ist in Abbildung 1.3 dargestellt. Parallel dazu betrachten wir den Prozess eines *Random-Walks* auf der Linie. Wieder interessiert uns die Bewegung eines Teilchens auf den Knoten eines Graphen $G' = (V', E')$ mit $|V'| = n+1$ und $|E'| = n$ (vgl. Abbildung 1.1). Für $v \in V$ ist die Übergangswahrscheinlichkeit des Teilchens von v zu einem der Nachbarknoten von v stets $\frac{1}{d(v)}$. Wir wissen bereits aus Lemma 1.1-10, dass die erwartete Zeit, die das Teilchen, startend bei dem Knoten 1, benötigt, bis es den Knoten $n + 1$ erreicht, kleiner als $2n^2$ ist. Das gilt auch dann, wenn wir die ungerichteten Kanten durch je zwei entgegengesetzt gerichtete Kanten ersetzen

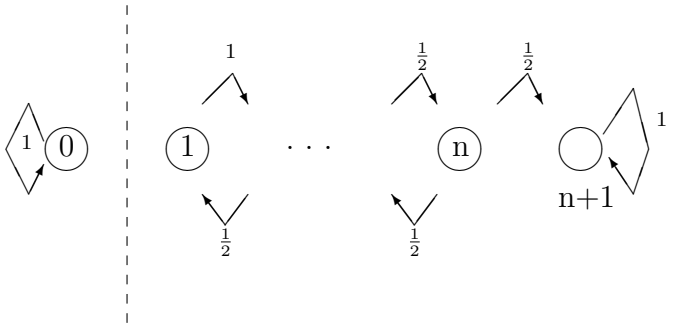

Abbildung 1.3: Wir nennen den zusammenhängenden Graphen rechts von der gestrichelten Linie $G^{S_2} = (V^{S_2}, E^{S_2})$. Der Graph G^{S_2} stellt den Prozess des *Gamblers-Ruin Problems* unter der Voraussetzung, dass Spieler S_2 ruiniert wird, dar.

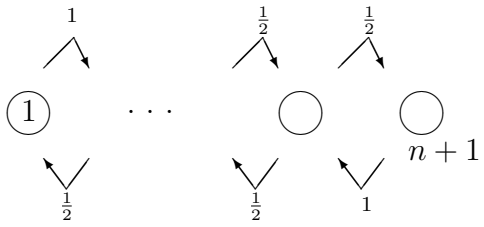

Abbildung 1.4: Darstellung des *Random-Walks* auf einer Linie durch einen gerichteten, kantengewichteten Graphen $G'' = (V'', E'')$.

und die Übergangswahrscheinlichkeiten beibehalten (siehe Abbildung 1.4). Wir nennen diesen Graphen dann $G'' = (V'', E'')$. Die Kantenbeschriftung beschreibt die Übergangswahrscheinlichkeit des Teilchens. Kanten können nicht entgegen ihrer Richtung traversiert werden. Der Prozess, bis ein in Knoten 1 startendes Teilchen zum ersten Mal den Knoten mit der Beschriftung $n+1$ erreicht, ist auf den Graphen G bzw. G'' identisch, weil sich die Graphen nur in den ausgehenden Kanten des Knotens $n+1$ unterscheiden. Was passiert, nachdem der Knoten $n+1$ das erste Mal besucht wurde, interessiert uns bei dem Prozess in dem Graphen G^{S_2}, in dem das *Gamblers-Ruin Problem* unter der Voraussetzung simuliert wird, dass Spieler S_2 ruiniert wird, nicht. Wenn der Knoten $n+1$ erreicht wird, ist der Spieler S_2 ruiniert. Die Dauer dieses Prozesses kann also auch für den Graphen G^{S_2} durch $2n^2$ nach oben beschränkt werden. Wir erhalten aus Bemerkung 1.1-11 sogar, dass die erwartete Wartezeit $\Theta(n^2)$ ist. □

Wir werden die Ergebnisse über das *Gamblers-Ruin Problem* für die Laufzeitabschätzung eines Quantenautomaten in Kapitel 4.3 verwenden.

1.2 Qubits

Unter einem Qubit verstehen wir in der Quantentheorie die Erweiterung eines klassischen Bits. Ähnlich zu einem klassischen Bit besitzt ein Qubit zwei Basiszustände, die in der Regel in Dirac-Notation mit $|0\rangle$ bzw. $|1\rangle$ bezeichnet werden. Üblicherweise ist

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad ; \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Anders als ein klassisches Bit kann ein Qubit aber mehr als nur einen dieser beiden Basiszustände annehmen und sich dann in einer Linearkombination $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ der Basiszustände befinden. Eine solche Linearkombination von Basiszuständen werden wir im Folgenden in der Regel Überlagerung oder Zustandsvektor nennen. Wir nennen $\alpha, \beta \in \mathbb{C}$ die Amplituden der Überlagerung. Die Amplituden erfüllen die Bedingung $|||\psi\rangle||_2^2 = 1$. Eine Überlagerung eines Qubits kann durch einen Punkt in einem komplexen Vektorraum beschrieben werden. Es sind überabzählbar viele verschiedene Überlagerungen für ein Qubit möglich. Dies scheint eine sehr mächtige Eigenschaft von Qubits zu sein. Wir werden jedoch in Kapitel 1.2.2 sehen, dass es eine entscheidende Schwierigkeit bei der Behandlung von Qubits gibt, die die Handhabung stark verkompliziert und den Nutzen dieser Eigenschaft relativiert.

Mit einem einzigen Qubit alleine sind noch keine allzu komplexen Anwendungen möglich. Dazu benötigen wir eine Menge von Qubits. Wir sprechen dann von einem System aus Qubits. Das kleinste System aus Qubits kennen wir schon. Es besteht aus genau einem Qubit. Jedes zu einem bestehenden System hinzugenommene Qubit verdoppelt die Anzahl der Basiszustände des Systems. Ein System aus zwei Qubits besitzt die Basiszustände

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

Ein System aus Qubits befindet sich zu jedem Zeitpunkt in einer Überlagerung $|\psi\rangle = \alpha_1|00\rangle + \alpha_2|01\rangle + \alpha_3|10\rangle + \alpha_4|11\rangle$ aller Basiszustände. Die von einem einzelnen Qubit bereits bekannten Bedingungen an die Amplituden bleiben erhalten. Es ist also $\alpha_1, \ldots, \alpha_4 \in \mathbb{C}$ und $|||\psi\rangle||_2^2 = 1$. Die Überlagerung, in der sich das System zum gegenwärtigen Zeitpunkt befindet, kann wieder durch einen Punkt in einem komplexen Vektorraum beschrieben werden.

Bemerkung 1.2-1
Wir werden stets unter der Annahme arbeiten, dass sich die Überlagerung eines Systems aus Qubits zu diskreten Zeitpunkten nicht ohne unser Zutun verändet. Wir können von einem Zeitpunkt zum nächsten Zeitpunkt eine Veränderung herbeiführen, sodass sich das System dann in einer anderen Überlagerung befindet (siehe dazu Kapitel 1.2.1). Diese

Annahme stellt eine häufig getroffene Abstraktion der Wirklichkeit dar, die das Rechnen mit Qubits vereinfacht.

Für $k \in \mathbb{N}$ betrachten wir im Allgemeinen ein System aus k Qubits, das sich für $n = 2^k$ in einer Überlagerung $|\psi\rangle = \sum_{i=1}^{n} \alpha_i |q_i\rangle, \alpha_i \in \mathbb{C}$, mit $|||\psi\rangle||_2^2 = 1$ befindet. Für $i \in \{1, \ldots, n\}$ sei q_i ein Basiszustand des Systems aus Qubits. Wir assoziieren die Basiszustände $q_1, \ldots, q_n$ mit der ON-Basis $|1\rangle, \ldots, |n\rangle$.

1.2.1 Transformationen

Um Berechnungen mit einem System aus Qubits durchzuführen, müssen wir die Überlagerung, in der sich das System befindet, manipulieren können. Wir sagen, dass wir eine Transformation auf einem gegebenen System durchführen. Jede Transformation kann durch eine quadratische Matrix beschrieben werden. Wir werden im Folgenden auch von Überführungsmatrizen sprechen. Die Dimension einer Überführungsmatrix M entspricht der Dimension des Vektorraumes, in dem sich die Überlagerung des Systems beschreiben lässt. Für die weitere Diskussion nehmen wir an, dass n die Dimension des betrachteten Vektorraumes ist. Jede Zeile bzw. Spalte der Matrix ist einem der Basiszustände des Systems zugeordnet. Wir assoziieren den i-ten Basiszustand des Systems mit dem Einheitsvektor $|i\rangle$. Sei $M \in \mathbb{C}^{n \times n}$ eine Überführungsmatrix auf dem System und $|\psi\rangle = \sum_{i=1}^{n} \alpha_i |i\rangle$ die gegenwärtige Überlagerung. Wenn wir die zu M gehörige Transformation auf die Überlagerung $|\psi\rangle$ anwenden, befindet sich das System danach in der Überlagerung $|\psi_1\rangle = M|\psi\rangle$. Jede Überführungsmatrix M ist unitär. Es ist also $M^\dagger M = I$. Dabei bezeichnet $M^\dagger$ die zu M adjungierte Matrix und I die entsprechende Einheitsmatrix. Für eine beliebige Überlagerung $|\psi\rangle$ eines gegebenen Systems aus Qubits sowie eine unitäre Überführungsmatrix M gilt wegen $M^\dagger M = I$ und $|||\psi\rangle||_2^2 = 1$ auch

$$1 = |||\psi\rangle||_2^2 = \langle\psi|\psi\rangle = \langle\psi|M^\dagger M|\psi\rangle = ||M|\psi\rangle||_2^2.$$

Eine Überlagerung $|\psi\rangle$ eines Systems aus Qubits kann als Punkt in einem komplexen Vektorraum beschrieben werden. Wenn das System aus nur einem Qubit besteht, befindet sich der Punkt $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in einem zweidimensionalen Vektorraum (vgl. Abbildung 1.5). Wegen der Bedingung $|||\psi\rangle||_2^2 = 1$ beträgt der Abstand des Punktes $|\psi\rangle$ zum Nullpunkt stets Eins. Jede Transformation in diesem einfachen System entspricht einer Drehung des Punktes $|\psi\rangle$ um den Nullpunkt. Soll der Winkel δ in der Abbildung zum Beispiel im Uhrzeigersinn um einen Winkel γ vergrößert werden, so gelingt dies durch die Transformation

$$M_\gamma = \begin{bmatrix} \cos(\gamma) & \sin(\gamma) \\ \sin(-\gamma) & \cos(\gamma) \end{bmatrix}.$$

(vgl. Abbildung 1.6). Wir werden in Kapitel 4.1.1 eine Spezielle Drehung betrachten.

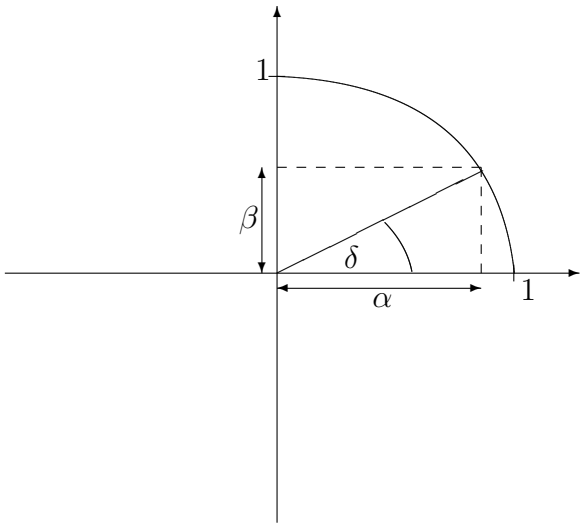

Abbildung 1.5: Darstellung einer Überlagerung $\alpha|0\rangle + \beta|1\rangle$ als Punkt in einem komplexen Vektorraum.

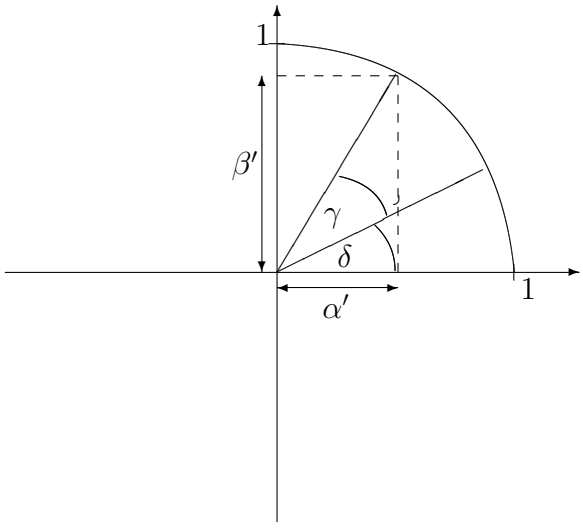

Abbildung 1.6: Drehung der Überlagerung um einen Winkel γ.

Spezielle Transformationen

Wir betrachten vier Matrizen $M_1, M_2, B_1, B_2 \in \mathbb{Z}^{3\times 3}$ mit

$$M_1 = \begin{bmatrix} 4 & 3 & 0 \\ -3 & 4 & 0 \\ 0 & 0 & 5 \end{bmatrix}, \tag{1.3}$$

$$M_2 = \begin{bmatrix} 4 & 0 & 3 \\ 0 & 5 & 0 \\ -3 & 0 & 4 \end{bmatrix}, \tag{1.4}$$

$$B_1 = \frac{1}{5}M_1, \tag{1.5}$$

$$B_2 = \frac{1}{5}M_2. \tag{1.6}$$

Wir werden nun einige Eigenschaften dieser Matrizen untersuchen. Dabei wird der Nutzen der Matrizen eventuell nicht sofort klar werden. Der Leser sollte die auch für sich schon interessanten Ergebnisse zur Kenntnis nehmen und vorerst im Hinterkopf behalten. Die Matrizen werden sich in Kapitel 4.1 im geeigneten Kontext als äußerst nützlich erweisen.

Bemerkung 1.2-2
Die Matrizen $B_1, B_2, B_1^{-1}, B_2^{-1}$ sind unitär.

Wir definieren für einen Vektor $|z\rangle \in \mathbb{Z}^3$ eine Funktion $f : \mathbb{Z}^3 \to \mathbb{Z}$ durch $f(|z\rangle) = 4\langle 1|z\rangle + 3\langle 2|z\rangle + 3\langle 3|z\rangle$ sowie eine Menge $K \subseteq \mathbb{Z}^3$ mit

$$K = \{|z\rangle \in \mathbb{Z}^3 : \langle 1|z\rangle \not\equiv 0 \mod 5, f(|z\rangle) \not\equiv 0 \mod 5 \text{ und } \langle 2|z\rangle\langle 3|z\rangle \equiv 0 \mod 5\}$$

Lemma 1.2-3 *(Lemma 2 in [AW02])*
Sei $|z\rangle \in \mathbb{Z}^3$. Falls $|z\rangle \in K$ ist, gilt $M_1|z\rangle \in K$ und $M_2|z\rangle \in K$.

Lemma 1.2-4 *(Lemma 3 in [AW02])*
Seien $|z\rangle, |v\rangle, |w\rangle \in \mathbb{Z}^3$ mit $|z\rangle = M_1|v\rangle = M_2|w\rangle$. Dann ist $|z\rangle \notin K$.

Lemma 1.2-5 *(Lemma 4 in [AW02])*
Sei $|z\rangle = Y_1^{-1} \cdots Y_n^{-1} X_n \cdots X_1 |1\rangle$ mit $|1\rangle = (1,0,0)^T$ und $X_j, Y_j \in \{M_1, M_2\}, j \in \{1, \ldots, n\}$. Dann ist

$$\langle 2|z\rangle^2 + \langle 3|z\rangle^2 \begin{cases} = 0, & \text{falls } X_j = Y_j \text{ für alle } j, \\ > \frac{1}{25^n}, & \text{sonst.} \end{cases}$$

Bemerkung 1.2-6
Aus Bemerkung 1.2-2 wissen wir, dass die Matrizen $B_1, B_2, {B_1}^{-1}, {B_2}^{-1}$ unitär sind. Für $|1\rangle \in \mathbb{Z}^3$ sei $|z\rangle = 5{Y_1}^{-1} \cdots 5{Y_n}^{-1} \frac{1}{5} X_n \cdots \frac{1}{5} X_1 |1\rangle$. Dann ist $|||z\rangle||_2^2 = 1$. Wegen

$$5{Y_1}^{-1} \cdots 5{Y_n}^{-1} \frac{1}{5} X_n \cdots \frac{1}{5} X_1 = {Y_1}^{-1} \cdots {Y_n}^{-1} X_n \cdots X_1$$

gilt also auch für $|z'\rangle = {Y_1}^{-1} \cdots {Y_n}^{-1} X_n \cdots X_1 |1\rangle$ sofort $|||z'\rangle||_2^2 = 1$.

1.2.2 Messungen

Bisher wissen wir, wie sich ein System aus Qubits mathematisch darstellen lässt und wie wir die Überlagerung des Systems manipulieren können. Wir sind gewohnt, mit Systemen zu arbeiten, die uns ihre Informationen gewissermaßen kostenlos zur Verfügung stellen. Bei einem System aus Qubits ist dies nicht so. Wir wissen in der Regel nicht, in welcher Überlagerung sich das System befindet. Um eine Information über die gegenwärtige Überlagerung des Systems zu bekommen, müssen wir eine Messung der Überlagerung vornehmen. Wir erinnern uns an unsere vereinfachende Annahme aus Bemerkung 1.2-1, dass das System seine erreichte Überlagerung nicht verändert, solange wir nicht in das System eingreifen. Wir haben es dann mit einem diskreten System zu tun. Durch unser Eingreifen wechselt das System in den Folgezeitpunkt. Dies geschieht beim Anwenden einer Transformation oder beim Messen der gegenwärtigen Überlagerung des Systems. Messungen eines Systems aus Qubits können durch eine Sammlung von speziellen Matrizen O_j beschrieben werden. Der Index j bezeichnet die j-te dieser Matrizen. Bei insgesamt k solcher Matrizen ist also $j \in \{1, \ldots, k\}$. Für die Matrizen gilt

$$\sum_{j=1}^{k} O_j^\dagger O_j = I.$$

Wir werden die Matrizen O_j sogar noch weiter einschränken, um die Betrachtung zu vereinfachen. Alle von uns benötigten Messungen lassen sich durch diese weiter eingeschränkten Matrizen beschreiben. In einem System aus $\log_2 n$ Qubits hat der Eintrag o_{ii}^j in O_j für einige $i \in \{1, \ldots, n\}$ den Wert Eins. Alle anderen Einträge in O_j sind mit dem Wert Null belegt. Sei $|\psi\rangle = \sum_{i=1}^{n} \alpha_i |i\rangle$ die aktuelle Überlagerung des Systems. Die Wahrscheinlichkeit, dass das Ergebnis von $O_j, j \in \{1, \ldots, k\}$, gemessen wird, ist $\mathcal{P}_j[|\psi\rangle] = \langle\psi|O_j^\dagger O_j|\psi\rangle$. Nach der Messung befindet sich das System aus Qubits dann in der normierten Überlagerung

$$|\psi'\rangle = \frac{O_j|\psi\rangle}{\sqrt{\langle\psi|O_j^\dagger O_j|\psi\rangle}}.$$

Nach dem Normieren ist wieder $|||\psi'\rangle||_2^2 = 1$. Jede weitere Messung des Systems liefert mit Sicherheit dasselbe Ergebnis wie die Erste Messung. Erst nach dem Anwenden einer Transformation auf die normierte Überlagerung ist ein anderes Ergebnis der Messung möglich.

1.3 Reellwertige Zustandsvektoren

Wir können Überlagerungen eines Systems aus Qubits auch durch reellwertige Vektoren beschreiben. Wir betrachten ein System mit n Basiszuständen. Jede Überlagerung $|\psi\rangle \in \mathbb{C}^n$ von Zuständen kann durch einen Vektor $|r\rangle \in \mathbb{R}^{2n}$ eindeutig beschrieben werden, indem zum Beispiel die Dimension des Vektors $|\psi\rangle$ verdoppelt wird. Wir beschreiben dies durch eine Funktion $s_1 : \mathbb{C}^n \to \mathbb{R}^{2n}$. Für $|\psi\rangle \in \mathbb{C}^n$ ist

$$s_1(|\psi\rangle) = \sum_{i=1}^{n} \left(\mathrm{Re}(\langle i|\psi\rangle)|i\rangle + \mathrm{Im}(\langle i|\psi\rangle)|n+i\rangle\right). \tag{1.7}$$

Die Zeilen $1, \ldots, n$ von $|r\rangle$ repräsentieren den Realteil des Ursprungsvektors $|\psi\rangle$, die restlichen Zeilen den Imaginärteil. Diese Darstellung wird auch in [BV93] eingeführt.

Bemerkung 1.3-1
Sei $|\psi\rangle \in \mathbb{C}^n$ und $\alpha \in \mathbb{R}$. Dann ist $s_1(\alpha|\psi\rangle) = \alpha s_1(|\psi\rangle)$.

Eine Zustandsüberlagerung $|\psi\rangle \in \mathbb{C}^n$ eines Systems aus Qubits lässt sich durch einen positiven reellwertigen Vektor beschreiben, wenn die Dimension durch eine Funktion $s_2 : \mathbb{R}^n \to \mathbb{R}^{2n}$ ein zweites Mal verdoppelt wird. Sei dazu $|r\rangle \in \mathbb{R}^n$. Es ist dann

$$s_2(|r\rangle) = \sum_{i=1}^{n} \left(\frac{|\langle i|r\rangle| + \langle i|r\rangle}{2}|i\rangle + \frac{|\langle i|r\rangle| - \langle i|r\rangle}{2}|n+i\rangle\right) \tag{1.8}$$

Bemerkung 1.3-2
Sei $|r\rangle \in \mathbb{R}^n$ und $\alpha \in \mathbb{R}$. Dann ist $s_2(\alpha|r\rangle) = \alpha s_2(|r\rangle)$.

Es ist $|r_2\rangle = s_2(s_1(|\psi\rangle))$ mit $|r_2\rangle \in (\mathbb{R}^+)^{4n}$. Die obere Hälfte des resultierenden Vektors entspricht dem Wert der positiven Einträge von $s_1(|\psi\rangle)$, die untere Hälfte dem Betragswert der negativen Einträge. Die Darstellung ist eindeutig und umkehrbar, sodass aus einem Vektor $|r_2\rangle \in (\mathbb{R}^+)^{4n}$ wieder der ursprüngliche Vektor $|\psi\rangle \in \mathbb{C}^n$ erhalten werden kann.

1.3.1 Wahrscheinlichkeitsvektoren

Definition 1.3-3
Ein Vektor $|p\rangle \in \mathbb{R}^n$ ist ein Wahrscheinlichkeitsvektor, falls $|p\rangle \in (\mathbb{R}^+)^n$ und $\sum_{i=1}^{n}\langle i|p\rangle = 1$ ist. Dabei ist $|1\rangle, \ldots, |n\rangle$ die ON-Basis.

Durch Normieren eines positiven reellwertigen Vektors erhalten wir einen Wahrscheinlichkeitsvektor. Ein Vektor $|r\rangle \in \mathbb{R}^{4n}$ wird hier durch $\frac{|r\rangle}{|||r\rangle||_1}$ normiert. Für $|r\rangle \in (\mathbb{R}^+)^n$ definieren wir die Funktion $s_3 : (\mathbb{R}^+)^n \to (\mathbb{R}^+)^n$ durch

$$s_3(|r\rangle) = \frac{|r\rangle}{|||r\rangle||_1} \tag{1.9}$$

Lemma 1.3-4
Seien $|\psi\rangle, |\psi'\rangle \in \mathbb{C}^n$, $|||\psi\rangle||_2^2 = |||\psi'\rangle||_2^2 = 1$ *sowie die Vektoren* $|r\rangle = s_3(s_2(s_1(|\psi\rangle)))$ *und* $|r'\rangle = s_3(s_2(s_3(|\psi'\rangle)))$ *die zugehörigen Wahrscheinlichkeitsvektoren (vgl. 1.7, 1.8 bzw. 1.9). Dann ist*

$$|\psi\rangle = |\psi'\rangle \Leftrightarrow \frac{|r\rangle}{|||r\rangle||_1} = \frac{|r'\rangle}{|||r'\rangle||_1}.$$

Beweis. „⇒“ Sei $|\psi\rangle = |\psi'\rangle$. Nach dem Anwenden von $s_2(s_1(|\psi\rangle))$, bzw. $s_2(s_1(|\psi'\rangle))$ (vgl. 1.7 und 1.8) gilt $|\psi\rangle = |\psi'\rangle \Leftrightarrow |r\rangle = |r'\rangle$. Daraus folgt sofort $|||r\rangle||_1^{-1}|r\rangle = |||r'\rangle||_1^{-1}|r'\rangle$.
„⇐“ Sei $|||r\rangle||_1^{-1}|r\rangle = |||r'\rangle||_1^{-1}|r'\rangle$. Es gibt dann ein $c \in \mathbb{R}$ mit $|r\rangle = c|r'\rangle$. Wegen $|r\rangle, |r'\rangle \in (\mathbb{R}^+)^{4n}$ ist $c \geq 0$. Es ist $|r\rangle = c|r'\rangle \Leftrightarrow |\psi\rangle = c|\psi'\rangle$ (vgl. Bemerkung 1.3-1 und Bemerkung 1.3-2). Wegen $|||\psi\rangle||_2^2 = |||\psi'\rangle||_2^2 = 1$ ist $c = 1$. □

Auch die Darstellung der Zustandsvektoren $|\psi\rangle \in \mathbb{C}^n$ als Wahrscheinlichkeitsvektoren $|p\rangle \in (\mathbb{R}^+)^{4n}$ ist also eindeutig und umkehrbar, sodass wieder die ursprünglichen Zustandsvektoren erreicht werden können.

Definition 1.3-5
Für beliebige $d \in \mathbb{R}$ *definieren wir*

$$(d)^+ = \begin{cases} d, & \textit{falls } d > 0, \\ 0, & \textit{sonst;} \end{cases} \tag{1.10}$$

$$(d)^- = \begin{cases} d, & \textit{falls } d < 0, \\ 0, & \textit{sonst.} \end{cases} \tag{1.11}$$

Lemma 1.3-6
Seien $a_i, b_i \in \mathbb{R}^+, i \in \{1, \ldots, n\}$, *mit* $\sum_{i=1}^n a_i = \sum_{i=1}^n b_i = 1$. *Dann gilt die Gleichung* $\sum_{i=1}^n (a_i - b_i)^+ = \frac{1}{2}\sum_{i=1}^n |a_i - b_i|$.

Beweis.

$$\begin{aligned} & \textstyle\sum_{i=1}^n (a_i - b_i)^+ \quad + \sum_{i=1}^n (a_i - b_i)^- \\ = \; & \textstyle\sum_{i=1}^n (a_i - b_i) \\ = \; & 1 - 1 = 0 \\ \Rightarrow \quad & \textstyle\sum_{i=1}^n (a_i - b_i)^+ \quad = -\sum_{i=1}^n (a_i - b_i)^- \\ \Rightarrow \quad & \textstyle\sum_{i=1}^n (a_i - b_i)^+ \quad = \frac{1}{2}\sum_{i=1}^n |(a_i - b_i)^+| + \frac{1}{2}\sum_{i=1}^n |(a_i - b_i)^-| \\ & \qquad\qquad\qquad\qquad = \textstyle\frac{1}{2}\sum_{i=1}^n |a_i - b_i| \end{aligned}$$

□

1.3.2 Reellwertige Überführungsmatrizen

Jede Überlagerung eines Systems aus Qubits kann also durch reellwertige Vektoren beschrieben werden. Gibt es auch für die Transformationen in einem solchen System entsprechende reellwertige Matrizen? Es stellt sich heraus, dass beliebige Transformationen eines Systems aus Qubits auch durch reellwertige Matrizen beschrieben werden können.

Lemma 1.3-7
Sei $k \in \mathbb{N}$ und $n = 2^k$ sowie M eine beliebige $n \times n$-Überführungsmatrix auf einem System aus $\log n$ Qubits. Zu M kann eine reellwertige $2n \times 2n$ Matrix M' angegeben werden, sodass für beliebige $|\psi\rangle \in \mathbb{C}^n$ mit $|\psi_2\rangle = M|\psi\rangle$ und $|r\rangle = s_1(|\psi\rangle), |r_2\rangle = s_1(|\psi_2\rangle)$ (vgl. 1.7) die Gleichung $|r_2\rangle = M'|r\rangle$ gilt.

Beweis. Wir definieren zu M eine Matrix M' mit

$$M' = \begin{bmatrix} [\mathrm{Re}(M)] & [-\mathrm{Im}(M)] \\ [\mathrm{Im}(M)] & [\mathrm{Re}(M)] \end{bmatrix}. \tag{1.12}$$

Es ist dann

$$\begin{array}{rcll} |\psi_2\rangle = & M|\psi\rangle & & \\ = & \sum_{j=1}^{n} |j\rangle\langle j|M|\psi\rangle & & \\ = & \sum_{j=1}^{n}\sum_{l=1}^{n} & (|j\rangle \mathrm{Re}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) & -|j\rangle \mathrm{Im}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) \\ & + & |j\rangle \mathrm{Re}(m_{jl}) i\mathrm{Im}(\langle l|\psi\rangle) & +|j\rangle i\mathrm{Im}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle)) \end{array}$$

und

$$\begin{array}{rcll} |r_2\rangle = & M'|r\rangle & & \\ = & \sum_{j=1}^{2n} |j\rangle\langle j|M'|r\rangle & & \\ = & \sum_{j=1}^{2n}\sum_{l=1}^{2n} |j\rangle m'_{jl}\langle l|r\rangle & & \\ = & \sum_{j=1}^{n}\sum_{l=1}^{n} & (|j\rangle \mathrm{Re}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) & -|j\rangle \mathrm{Im}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) \\ & + & |n+j\rangle \mathrm{Im}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) & + |n+j\rangle \mathrm{Re}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle)) \end{array}$$

nach der speziellen Konstruktion von $|r\rangle = s_1(|\psi\rangle)$ (vgl. 1.7) und M' aus M. Wenn wir die Summanden in $M|\psi\rangle$ nach Realteil und Imaginärteil ordnen, erhalten wir

$$\mathrm{Re}(M|\psi\rangle) = \sum_{j=1}^{n}\sum_{l=1}^{n} |j\rangle \mathrm{Re}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) - |j\rangle \mathrm{Im}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) \tag{1.13}$$

$$\mathrm{Im}(M|\psi\rangle) = \sum_{j=1}^{n}\sum_{l=1}^{n} |j\rangle \mathrm{Re}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) + |j\rangle \mathrm{Im}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) \tag{1.14}$$

In $|r_2\rangle$ gibt es weder einen Realteil noch einen Imaginärteil, aber nach der Transformation $s_1(|\psi_2\rangle)$ (vgl. 1.7) werden die ersten n Einträge in $|r_2\rangle$ dem Realteil von $|\psi_2\rangle$ zugeordnet und die restlichen n Einträge dem Imaginärteil. Durch Aufsplitten der Summe $M'|r\rangle$ bekommen wir

$$\sum_{i=1}^{n} |i\rangle\langle i|r_2\rangle = \sum_{j=1}^{n}\sum_{l=1}^{n} |j\rangle \mathrm{Re}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) - |j\rangle \mathrm{Im}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) \quad (1.15)$$

$$\sum_{i=1}^{n} |n+i\rangle\langle n+i|p_2\rangle = \sum_{j=1}^{n}\sum_{l=1}^{n} |n+j\rangle \mathrm{Re}(m_{jl})\mathrm{Im}(\langle l|\psi\rangle) + |n+j\rangle \mathrm{Im}(m_{jl})\mathrm{Re}(\langle l|\psi\rangle) \quad (1.16)$$

Wir sehen durch Vergleichen der Summen 1.13 mit 1.15 sowie 1.14 mit 1.16, dass $|r_2\rangle = s_1(|\psi_2\rangle)$ gilt. □

Kapitel 2

Klassische Automatenmodelle

Ein einfaches Rechnermodell ist das Modell endlicher Automaten. Es gibt eine ganze Reihe verschiedener Varianten endlicher Automaten, von denen wir einige im Folgenden vorstellen möchten. In diesem Kapitel beschränken wir uns auf Automaten, die ihre Berechnung durchführen, ohne auf ein System aus Qubits zurückzugreifen. Wir nennen diese Automaten klassische Automaten. Es wird keine umfassende Beschreibung aller klassischen Automatenmodelle gegeben, sondern es werden nur die für die weitere Betrachtung von Quantenautomaten interessanten Modelle und Ergebnisse vorgestellt.

Mit Σ bezeichnen wir ein Alphabet von Symbolen σ. Wir werden die Symbole Buchstaben nennen und eine Konkatenation w von Buchstaben ein Wort. Das leere Wort bezeichnen wir mit ε. Dabei ist Σ^* die Menge aller Wörter über dem Alphabet Σ.

Definition 2-1
Sei $w \in \Sigma^*$. Eine Menge $L \subseteq \Sigma^*$ nennen wir eine Sprache.

Alle hier vorgestellten Automatenmodelle verfügen über ein Eingabeband unbeschränkter Länge, auf das sie lesend zugreifen können. Auf dem Band stehen Buchstaben $\sigma \in \Sigma$. Ein Automat besitzt eine endliche Zustandsmenge Q. Zu jedem Zeitpunkt befindet sich der Automat in einem der Zustände aus Q. Die Reihenfolge und die Art der gelesenen Zeichen haben Einfluss darauf, in welchen Zuständen sich der Automat befindet. Dieser Einfluss wird durch eine Zustandsüberführungsfunktion δ ausgedrückt. Wir werden die klassischen Automaten in zwei Gruppen von deterministischen bzw. probabilistischen Automaten teilen. Ein deterministischer Automat wird bei wiederholter Eingabe desselben Wortes $w \in \Sigma^*$ immer genau dieselbe Folge von Zuständen durchlaufen. Das Verhalten eines probabilistischen Automaten auf einer Eingabe w können wir durch eine Wahrscheinlichkeitsverteilung auf einer Menge von Zustandsfolgen beschreiben.

2.1 Deterministische Automaten

Wir teilen die vorzustellenden deterministischen Automaten grob in zwei Gruppen ein. Dadurch erhalten wir auf der einen Seite Automaten, die keine Ausgabe erzeugen und

auf der anderen Seite Automaten mit Ausgabe. Im Wesentlichen besitzen Automaten mit Ausgabe ein zusätzliches Ausgabeband, auf das sie eine Ausgabe schreiben. Dafür führen sie im Gegensatz zu Automaten ohne Ausgabe keine weitere Bewertung der Eingabe durch. Ein Akzeptieren oder Verwerfen ist für Automaten mit Ausgabe nicht vorgesehen.

2.1.1 Automaten mit Ausgabe

Definition 2.1-1
Ein 2-DFA $D = (Q, \Sigma_{ein}, \Sigma_{aus}, q_0, \delta, Q_{stop})$ mit Ausgabe verfügt über ein Eingabeband unbeschränkter Länge, das nur gelesen werden darf sowie über ein ebenfalls in der Länge unbeschränktes Ausgabeband. Der Automat besitzt eine endliche Zustandsmenge Q, ein endlichen Eingabealphabet Σ_{ein}, einen Startzustand q_0, ein endliches Ausgabealphabet Σ_{aus}, eine Überführungsfunktion δ sowie eine Menge $Q_{stop} \subset Q$ von haltenden Zuständen. Es ist $Q_{non} = Q \setminus Q_{stop}$ und $q_0 \in Q_{non}$. Das Ausgabeband wird von links nach rechts beschrieben. Der Lesekopf auf dem Eingabeband darf in einem Schritt eine Rechtsbewegung oder eine Linksbewegung durchführen. Für zwei Endmarkierungen $\dashv, \vdash \notin \Sigma_{ein}$ ist $\Gamma = \Sigma_{ein} \cup \{\dashv, \vdash\}$ das Bandalphabet des Eingabebandes. Die Überführungsfunktion ist

$$\delta : Q_{non} \times \Gamma \times Q \times \{-1, 1\} \times (\Sigma_{aus} \cup \{\varepsilon\}) \to \{0, 1\}$$

und wird folgendermaßen interpretiert. Für $q_1, q_2 \in Q, \sigma_1 \in \Gamma, d \in \{-1, 1\}, \sigma_2 \in \Sigma_{aus}$ beschreibt $\delta(q_1, \sigma_1, q_2, d, \sigma_2)$ die Wahrscheinlichkeit, mit der bei gelesenem Zeichen σ_1 aus dem Zustand q_1 in den Zustand q_2 gewechselt, das Ausgabezeichen σ_2 geschrieben und der Kopf in die Richtung d bewegt wird. Dabei entspricht $d = -1$ einer Linksbewegung und $d = 1$ einer Rechtsbewegung. Eine Rechtsbewegung (Linksbewegung) über $\vdash$ ($\dashv$) hinaus ist nicht möglich. Die Berechnung ist deterministisch, weil die Wahrscheinlichkeit stets Null oder Eins ist. Zu Beginn der Berechnung steht die Eingabe in der Form $\dashv w \vdash$ auf dem Band, die Maschine befindet sich im Zustand q_0 und der Lesekopf liest das Zeichen $\dashv$. Die Berechnung des Automaten wird entsprechend der Überführungsfunktion δ durchgeführt. Die Berechnung stoppt, wenn ein Zustand $q \in Q_{stop}$ erreicht wird, oder sich der Automat in einer Endlosschleife befindet.

Bemerkung 2.1-2
Der 2-DFA mit Ausgabe schreibt in jedem Schritt maximal ein Ausgabezeichen. Den Fall, dass der Automat für $k \geq 2$ in einem Schritt die Zeichen $\sigma_1 \dots \sigma_k$ schreibt, können wir dadurch simulieren, dass ein neues Zeichen $\sigma_{1\dots k}$ in das Ausgabealphabet aufgenommen wird, das die Zeichenfolge $\sigma_1 \dots \sigma_k$ repräsentiert. Wenn die Ausgabe anschließend von einem weiteren Automaten B gelesen werden soll, ist es notwendig, die Überführungsfunktion von B entsprechend anzupassen.

2.1.2 Automaten ohne Ausgabe

Definition 2.1-3
Ein 1-DFA $D = (Q, \Sigma, q_0, \delta, Q_{acc})$ verfügt über ein Eingabeband unbeschränkter Länge, eine endliche Zustandsmenge Q, ein endliches Eingabealphabet Σ, einen eindeutigen Startzustand $q_0 \in Q$, eine Überführungsfunktion δ sowie eine Menge $Q_{acc} \subset Q$ von akzeptierenden Zuständen. Die Überführungsfunktion ist $\delta : Q \times \Sigma \to Q$ und beschreibt, in welchen Zustand der Automat in Abhängigkeit von dem gelesenen Zeichen und dem Ausgangszustand wechselt. Zu Beginn der Berechnung steht das Eingabewort $w = w_1 \dots w_n$ auf dem Band. Der Lesekopf liest das erste Zeichen w_1 der Eingabe. Der Automat befindet sich im Zustand q_0. Entsprechend ist $\delta(q_0, w_1)$ der Nachfolgezustand des Automaten. Der Lesekopf bewegt sich dann einen Buchstaben nach rechts und dieselbe Prozedur beginnt von vorne. Wenn das Wort einmal gelesen wurde und sich der Automat dann in einem akzeptierenden Zustand $q \in Q_{acc}$ befindet, wird die Eingabe akzeptiert, ansonsten wird sie verworfen. Der Automat D erkennt eine Sprache L, wenn

1. $\forall\ w \in\ L:$ *D akzeptiert w,*
2. $\forall\ w \notin\ L:$ *D verwirft w*

gilt.

Definition 2.1-4
Die Menge der Sprachen, die von 1-DFAs erkannt werden können, nennen wir die Menge der regulären Sprachen.

Wir werden nicht weiter auf die Eigenschaften von *1-DFA*s eingehen. Eine weitergehende Betrachtung liefert zum Beispiel [Weg99]. Mit der Definition eines neuen Automatentyps sollte die Hoffnung verbunden sein, mehr als nur die Menge der regulären Sprachen erkennen zu können, oder zumindest weniger Zustände zu benötigen.

Definition 2.1-5
Ein 2-DFA $D = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ verfügt über ein Eingabeband unbeschränkter Länge, eine endliche Zustandsmenge Q, ein endliches Eingabealphabet Σ, einen eindeutigen Startzustand q_0, eine Überführungsfunktion δ sowie Mengen Q_{acc} und Q_{rej} von akzeptierenden bzw. verwerfenden Zuständen. Wir definieren $Q_{non} = Q \setminus (Q_{acc} \cup Q_{rej})$. Es ist $Q_{acc} \cap Q_{rej} = \emptyset, q_0 \in Q_{non}$ und $(Q_{acc} \cup Q_{rej}) \subset Q$. Für zwei Endmarkierungen $\dashv, \vdash \notin \Sigma$ ist $\Gamma = \Sigma \cup \{\dashv, \vdash\}$ das Bandalphabet des Automaten. Die Überführungsfunktion ist $\delta : Q_{non} \times \Gamma \times Q \times \{-1, 1\} \to \{0, 1\}$. Zu Beginn der Berechnung steht die Eingabe w zwischen den beiden Endmarkierungen in der Form $\dashv w \vdash$ auf dem Band. Der Automat befindet sich im Zustand q_0 und der Lesekopf liest die linke Endmarkierung. Der 2-DFA kann den Lesekopf in einem Schritt nach rechts oder nach links bewegen. Eine Kopfbewegung über eine der beiden Endmarkierungen hinaus ist jedoch nicht möglich. Die Berechnung stoppt akzeptierend, wenn ein Zustand $q \in Q_{acc}$ erreicht wird und verwerfend

beim Erreichen eines Zustandes $q \in Q_{rej}$, oder wenn der Automat in eine Endlosschleife gerät. Der Automat D erkennt eine Sprache L, wenn

1. $\forall\, w \in L:$ *D akzeptiert w,*

2. $\forall\, w \notin L:$ *D verwirft w*

gilt.

Mit dieser Definition weichen wir in mehrfacher Hinsicht von der Definition von *2-DFA*s in [Weg99] ab. Dort wird die Berechnung nicht abgebrochen, wenn ein akzeptierender Zustand vor dem Erreichen des letzten Buchstabens in der Eingabe erreicht wird. Dieses Verhalten können wir mit dem Modell nach Definition 2.1-5 simulieren, wenn wir den Automaten in eine Endlosschleife laufen lassen, anstatt in einen akzeptierenden Zustand, was in [Weg99] eine Möglichkeit zum Verwerfen ist. Eine andere Möglichkeit ist, in einen Zustand zu wechseln, der eine Rechtsbewegung des Lesekopfes herbeiführt, bis die rechte Endmarkierung erreicht wird, um dann in einen akzeptierenden Zustand zu wechseln.

Ein weiterer Unterschied sind die Endmarkierungen. Das Modell in [Weg99] besitzt keine Endmarkierungen. Die Eingabe wird verworfen, wenn der Lesekopf über den linken Rand der Eingabe hinausbewegt wird und kann nur akzeptiert werden, wenn die Eingabe nach rechts verlassen wird. Es ist eine leichte Übung, dieses Verhalten mit dem Modell nach Definition 2.1-5 zu simulieren. Wir wählen diese leicht abgewandelte Definition eines *2-DFA*s, um eine bessere Vergleichbarkeit zu den ab Kapitel 3 vorzustellenden Quantenautomaten zu gewährleisten.

Bemerkung 2.1-6
Wir fordern für alle klassischen Automaten mit Ausnahme des 1-QFA, dass $q_0 \in Q_{non}$ gilt. Diese Konvention ist nicht üblich. Es muss so zusätzlicher Aufwand betrieben werden, um das leere Wort ε zu akzeptieren oder zu verwerfen. Man kann sich leicht überlegen, dass das leere Wort ε trotzdem von den hier definierten klassischen Automaten akzeptiert bzw. verworfen werden kann. Dazu kann der Automat direkt beim beim Lesen der linken Endmarkierung $\dashv$ in einen Zustand q mit $q \in Q_{acc}$ oder $q \in Q_{rej}$ wechseln.

Satz 2.1-7 *(Satz 4.5.6 in [Weg99])*
Die von 2-DFAs erkannten Sprachen sind regulär.

Die Fähigkeit, den Lesekopf in verschiedene Richtungen auf dem Eingabeband zu bewegen, führt nicht dazu, dass ein *2-DFA* mehr Sprachen erkennen kann, als ein *1-DFA*. Für bestimmte Sprachen kann ein *2-DFA* jedoch bezogen auf die Größe der Zustandsmenge exponentiell kleiner sein, als ein *1-DFA* [Weg99]. Ein spezieller deterministischer Automat, bei dem die Berechnung umkehrbar ist, ist ein reversibler Automat (*RFA*).

Definition 2.1-8
Ein 2-RFA ist ein spezieller 2-DFA, bei dem für die Überführungsfunktion δ für beliebige Zustände $q_1, q_2, q_3 \in Q,\ d_1, d_2 \in \{-1, 1\}$ und $\sigma \in \Gamma$ gilt

$$1 = \delta(q_1, \sigma, q_3, d_1) = \delta(q_2, \sigma, q_3, d_2) \Rightarrow (q_1 = q_2) \wedge (d_1 = d_2).$$

Eine detailliertere Betrachtung von *2-RFA*s wird zum Beispiel in [Pin97] gegeben. Wir benötigen hier lediglich das folgende Lemma. Die Kernaussage dazu liefert *Proposition* 4 in [KW97].

Lemma 2.1-9
Sei $D = (Q, \Sigma, q_0, \delta, Q_{acc})$ ein beliebiger 1-DFA. Dann kann zu D ein 2-RFA R konstruiert werden, sodass für beliebige $w \in \Sigma^*$ gilt:

$$D \text{ akzeptiert } w \Leftrightarrow R \text{ akzeptiert } w.$$

Die Laufzeit von R ist $O(|w|)$.

Ein *2-RFA* erkennt damit genau die Menge der regulären Sprachen. Wir werden in dem folgenden Abschnitt untersuchen, ob die Hinzunahme von Probabilismus dazu führt, dass mehr Sprachen als im deterministischen Fall erkannt werden können.

2.2 Probabilistische Automaten

Wir werden die grobe Einteilung aus Kapitel 2.1 hier beibehalten und die probabilistischen Automaten in zwei Gruppen einteilen. Auf der einen Seite Automaten mit Ausgabe, die zwar über ein zusätzliches Ausgabeband verfügen, aber die Eingabe nicht bewerten sowie auf der anderen Seite probabilistische Automaten ohne Ausgabe.

2.2.1 Automaten mit Ausgabe

Definition 2.2-1
Ein 2-PFA $P = (Q, \Sigma_{ein}, \Sigma_{aus}, q_0, \delta, Q_{stop})$ mit Ausgabe verfügt über ein Eingabeband, ein Ausgabeband, eine endliche Zustandsmenge Q, ein endliches Eingabealphabet Σ_{ein}, ein endliches Ausgabealphabet Σ_{aus}, einen eindeutigen Startzustand q_0, eine Überführungsfunktion δ sowie über eine Menge $Q_{stop} \subset Q$ von haltenden Zuständen. Die Ein- und Ausgabebänder sind nicht in der Länge beschränkt. Das Ausgabeband wird von links nach rechts beschrieben. Der Lesekopf auf dem Eingabeband darf mehrfach die Leserichtung wechseln oder stehenbleiben. Es ist $Q_{non} = Q \setminus Q_{stop}$ und $q_0 \in Q_{non}$. Für zwei Endmarkierungen $\dashv, \vdash \notin \Sigma_{ein}$ ist $\Gamma = \Sigma_{ein} \cup \{\dashv, \vdash\}$ das Bandalphabet des Eingabebandes. Die Überführungsfunktion ist

$$\delta : Q_{non} \times \Gamma \times Q \times \{-1, 0, 1\} \times \Sigma_{aus} \cup \{\varepsilon\} \to [0, 1],$$

wobei für alle $q \in Q_{non}, \sigma \in \Sigma_{ein}$ *gilt*

$$\sum_{\substack{q' \in Q,\\ d \in \{-1,0,1\},\\ \sigma' \in \Sigma_{aus} \cup \{\varepsilon\}}} \delta(q, \sigma, q', d, \sigma') = 1.$$

Dabei beschreibt d *die Richtung, in der sich der Lesekopf des Automaten bewegt. Es entspricht* $d = 1$ *einer Rechtsbewegung,* $d = -1$ *einer Linksbewegung und* $d = 0$ *der Situation, dass der Kopf nicht bewegt wird. Zu Beginn der Berechnung steht das Eingabewort* w *zwischen den Endmarkierungen in der Form* $\dashv w \vdash$ *auf dem Eingabeband. Der Kopf liest das Zeichen* $\dashv$*, der Startzustand ist* q_0*. Mit Wahrscheinlichkeit* $\delta(q, \sigma, q', d, \sigma')$ *wird* P *bei gelesenem Zeichen* σ *aus dem Zustand* q *in den Zustand* q' *wechseln, dabei den Kopf einen Schritt in die Richtung* d *bewegen und das Zeichen* σ' *schreiben. Eine Rechtsbewegung (Linksbewegung) nach dem Lesen von* $\vdash$ *(*$\dashv$*) ist nicht möglich. Die Berechnung wird beendet, sobald ein Zustand* $q \in Q_{stop}$ *erreicht wird, oder der Automat in eine Endlosschleife gerät.*

Wir setzen wieder voraus, dass der Automat in jedem Schritt maximal ein Ausgabezeichen schreibt. Das schreiben mehrerer Ausgabezeichen kann wie in Bemerkung 2.1-2 beschrieben simuliert werden.

Im Gegensatz zu den *2-DFA*-Varianten darf der *2-PFA* mit Ausgabe den Lesekopf in einem Rechenschritt auch nicht bewegen. Bei den deterministischen Automaten kann man sich leicht überlegen, dass eine Nullbewegung in einer Endlosschleife resultiert und deswegen überflüssig ist. Ein *2-PFA* mit Ausgabe kann hingegen so definiert sein, dass er mit Wahrscheinlichkeit kleiner als Eins stehenbleibt und dabei eine Ausgabe σ schreibt. Diese „Schleife“ wird mit Wahrscheinlichkeit Eins wieder verlassen. Wie häufig die Ausgabe σ an dieser Stelle geschrieben wird, kann nur mit dem Erwartungswert angegeben werden. Das Modell ändert sich also, wenn die Kopfbewegung „0“ zugelassen wird.

2.2.2 Automaten ohne Ausgabe

Wir werden im Folgenden wieder nur die für uns relevanten Ergebnisse zu probabilistischen Automaten vorstellen. Eine weitergehende Betrachtung ist zum Beispiel in [Paz71] zu finden.

Definition 2.2-2
Ein 2-PFA $P = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ *verfügt über ein Eingabeband unbeschränkter Länge, eine endliche Zustandsmenge* Q*, ein endliches Eingabealphabet* Σ*, einen eindeutigen Startzustand* q_0*, eine Überführungsfunktion* δ *sowie über Mengen* Q_{acc} *und* Q_{rej} *von akzeptierenden bzw. verwerfenden Zuständen. Es ist* $Q_{acc} \cap Q_{rej} = \emptyset$*,* $(Q_{acc} \cup Q_{rej}) \subset Q$ *und* $Q_{non} = Q \setminus (Q_{rej} \cup Q_{acc})$ *sowie* $q_0 \in Q_{non}$*. Für zwei Endmarkierungen* $\dashv, \vdash \notin \Sigma$ *ist* $\Gamma = \Sigma \cup \{\dashv, \vdash\}$ *das Bandalphabet des Automaten. Die Überführungsfunktion ist*

$$\delta : Q_{non} \times \Gamma \times Q \times \{-1, 1\} \to [0, 1],$$

wobei für alle $q \in Q_{non}, \sigma \in \Sigma$ *gilt*

$$\sum_{q' \in Q, d \in \{-1,1\}} \delta(q, \sigma, q', d) = 1.$$

Dabei beschreibt d *die Richtung, in der sich der Lesekopf bewegt, wobei* $d = 1$ *eine Rechtsbewegung und* $d = -1$ *entsprechend eine Linksbewegung des Lesekopfes bewirkt. Zu Beginn der Berechnung steht das Eingabewort* w *zwischen den Endmarkierungen in der Form* $\dashv w \vdash$ *auf dem Band. Der Kopf liest das Zeichen* $\dashv$*, der Startzustand ist* q_0*. Bei einem gelesenen Zeichen* $\sigma \in \Gamma$ *und dem gegenwärtigen Zustand* $q \in Q_{non}$ *wird* P *mit Wahrscheinlichkeit* $\delta(q, \sigma, q', d)$ *in den Zustand* q' *wechseln und den Kopf einen Schritt in die Richtung* d *bewegen. Eine Rechtsbewegung (Linksbewegung) nach dem Lesen von* $\vdash$ *(*$\dashv$*) ist nicht möglich.*

Die Berechnung wird akzeptierend beendet, sobald ein Zustand $q \in Q_{acc}$ *erreicht wird, bzw. verwerfend beim Erreichen eines Zustandes* $q \in Q_{rej}$ *oder wenn der Automat in eine Endlosschleife gerät. Sei* $\varepsilon \in [0, \frac{1}{2}[$*. Der Automat* P *erkennt eine Sprache* L *mit einem Fehler von* ε*, wenn*

1. $\forall\ w\ \in L : \mathcal{P}[P \text{ akzeptiert } w] \geq 1 - \varepsilon$,

2. $\forall\ w\ \notin L : \mathcal{P}[P \text{ akzeptiert } w] < \varepsilon$

gilt.

Die Bewegung $d = 0$ ist für den *2-PFA* nicht notwendig. Weil der Automat keine Ausgabe erzeugen kann, kann nur eine Verzögerung der Berechnung herbeigeführt werden. Man kann leicht einsehen, dass durch eine Modifikation der Überführungsmatrizen alle aus einer Nullbewegung resultierenden Wahrscheinlichkeiten für die Nachfolgezustände und Kopfbewegungen auch ausschließlich mit Links- oder Rechtsbewegungen erreicht werden können. Wir können eine Sprache finden, die von einem *2-PFA* erkannt werden kann, aber von keinem deterministischen Automaten.

Satz 2.2-3 *(Theorem* 1 *in [Fre81])*
Für beliebige $\varepsilon > 0$ *gibt es einen 2-PFA, der die Sprache* $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ *mindestens mit Wahrscheinlichkeit* $1 - \varepsilon$ *akzeptiert.*

Die Sprache $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ ist nicht regulär und kann deswegen nicht von einem *1-DFA* erkannt werden [Weg99]. Wir können mit einem *2-PFA* also mehr Sprachen erkennen, als mit allen bisher vorgestellten Automaten. Der in dem Beweis zu diesem Satz definierte
2-PFA hat eine exponentielle Rechenzeit. Schöner wäre es, wenn wir ein ähnliches Ergebnis für einen *2-PFA* mit polynomieller Rechenzeit bekommen könnten. In [DS90] wird jedoch gezeigt, dass wir uns hier keine Hoffnungen zu machen brauchen.

Satz 2.2-4 *(Theorem 4.4 in [DS90])*
Sei M ein 2-PFA, der eine nicht-reguläre Sprache L in einer erwarteten Rechenzeit $\mathcal{T}(|w|)$ erkennt, wobei $w \in \Sigma^*$ die Eingabe bezeichnet. Dann gibt es eine Konstante $b > 0$, sodass $\mathcal{T}(|w|) \geq 2^{|w|^b}$ für unendlich viele Eingaben w gilt.

Wir können also einen kleinen Fortschritt verzeichnen. Zumindest wenn wir die Rechenzeit nicht beschränken, haben wir ein Modell gefunden, das mehr Sprachen erkennen kann als unser Referenzmodell des *1-DFA*s. Da ein deterministischer Automat ein spezieller probabilistischer Automat ist, bei dem jede Zustandsüberführung mit Wahrscheinlichkeit Eins bzw. Null eintritt, kann der minimale deterministische Automat für eine Sprache L nicht kleiner sein, als der kleinste probabilistische Automat. Die Größe eines Automaten wird dabei durch die Anzahl seiner Zustände bestimmt. In [Amb96] wurde sogar gezeigt, dass es Sprachen gibt, für die die Anzahl der Zustände des kleinsten deterministischen Automaten exponentiell in der Anzahl der Zustände eines probabilistischen Automaten ist.

Definition 2.2-5
Ein 1-PFA ist ein Spezialfall eines 2-PFA, bei dem für alle $q, q' \in Q, \sigma \in \Gamma$ für die Überführungsfunktion $\delta(q, \sigma, q', -1) = 0$ gilt. Wir definieren deswegen die Überführungsfunktion eines 1-PFA durch $\delta : Q \times \Gamma \times Q \to [0, 1]$. Für alle $q \in Q, \sigma \in \Gamma$, ist $\sum_{q' \in Q} \delta(q, \sigma, q') = 1$.

Bemerkung 2.2-6
Ein 1-DFA ist ein spezieller 1-PFA, bei dem $\delta : Q \times \Gamma \times Q \to \{0, 1\}$ gilt.

Eventuell ist es schon ausreichend, verschiedene klassische Automaten zu kombinieren, um ein Automatenmodell zu erhalten, das mehr als nur die Menge der regulären Sprachen erkennen kann.

Definition 2.2-7
Ein Automat $B^{(A)}$ ist eine Kombination von zwei Automaten A und B. Dabei ist A ein Automat mit Ausgabe. Der Automat B hat keine Ausgabe. Eine Eingabe w wird zunächst von A bearbeitet. Wenn A stoppt, bearbeitet B die Ausgabe y von A. Sei $\varepsilon \in [0, \frac{1}{2}[$ und sei P ein B^A. Dann erkennt P eine Sprache L mit einem Fehler von ε, wenn

1. $\forall\, w \in L : \mathcal{P}[P \text{ akzeptiert } w] \geq 1 - \varepsilon$,
2. $\forall\, w \notin L : \mathcal{P}[P \text{ akzeptiert } w] < \varepsilon$

gilt.

Bemerkung 2.2-8
Ein Automat $B^{(A)}$ kann jede Sprache erkennen, die der Automat B erkennen kann, weil

der Automat A die Eingabe unverändert wieder ausgeben kann. Außerdem kann der Automat $B^{(A)}$ jede Sprache erkennen, die ein Automat mit Ausgabe vom Typ A erkennen kann. Dazu muss A zunächst die Eingabe bearbeiten. A besitzt keine akzeptierenden oder verwerfenden Zustände. Statt in einem solchen Zustand zu halten, schreibt A am Ende der Berechnung eine von zwei speziellen Ausgaben y_{acc} bzw. y_{rej} und stoppt danach. Der nachgeschaltete Automat B braucht dann selbst keine Entscheidung zu treffen, sondern kann akzeptieren, wenn er y_{acc} liest und andernfalls verwerfen.

Lemma 2.2-9
Sei $\varepsilon \in [0, \frac{1}{2}[$ und sei P ein 1-PFA$^{(2\text{-}PFA)}$ der eine Sprache L in erwarteter polynomieller Zeit mit einem Fehler von ε erkennt. Seien $P_1 = (Q^{P_1}, \Sigma^{P_1}, q_0^{P_1}, \delta^{P_1}, Q_{acc}^{P_1}, Q_{rej}^{P_1})$ der 1-PFA und $P_2 = (Q^{P_2}, \Sigma^{P_2}, q_0^{P_2}, \delta^{P_2}, Q_{stop}^{P_2})$ der 2-PFA mit Ausgabe in dem Automaten P. Es gibt einen 2-PFA für L mit höchstens $|Q^{P_1}| \cdot |Q^{P_2}|$ Zuständen, der L in erwarteter polynomieller Zeit mit einem Fehler von ε erkennt.

Beweis. Sei L_{ein} die von dem 1-PFA$^{(2\text{-}PFA)}$ P in erwarteter polynomieller Zeit mit einem Fehler von ε erkannte Sprache. Um die Aussage zu zeigen, werden wir einen 2-PFA P_3 entwerfen, der jede Eingabe $w \in L_{ein}$ mindestens mit Wahrscheinlichkeit $1 - \varepsilon$ akzeptiert und jede Eingabe $w \notin L_{ein}$ nur mit einer Wahrscheinlichkeit kleiner als ε. Die Rechenzeit von P_3 ist identisch zu der Rechenzeit von P_2 und polynomiell in der Eingabelänge. Der 2-PFA $P_3 = (Q^{P_3}, \Sigma_{ein}^{P_2}, q_0^{P_2}, \delta^{P_3}, Q_{acc}^{P_3}, Q_{rej}^{P_3})$ setzt sich wie folgt zusammen. Seien $Q_0^{P_2}, \ldots, Q_{|Q^{P_1}|-1}^{P_2}$ mit $Q_i^{P_2} = \{q_{(0,i)}, \ldots, q_{(|Q^{P_2}|-1,i)}\}$ Kopien der Zustandsmenge Q^{P_2} von P_2. Es ist

$$\begin{aligned} Q_{acc}^{P_3} &= \{q_{(j,i)} | q_i^{P_1} \in Q_{acc}^{P_1} \text{ und } j \in \{0, \ldots, |Q^{P_2}| - 1\}\}, \\ Q_{rej}^{P_3} &= \{q_{(j,i)} | q_i^{P_1} \notin Q_{acc}^{P_1} \text{ und } q_j^{P_2} \in Q_{stop}^{P_2} \text{ oder } q_i^{P_1} \in Q_{rej}\} \end{aligned}$$

und $Q^{P_3} = \bigcup_{i=0}^{|Q^{P_1}|-1} Q_i^{P_2}$.

Die Überführungsfunktion ist für $i, j \in \{0, \ldots, |Q^{P_2}| - 1\}; l, m \in \{0, \ldots, |Q^{P_1}| - 1\}$, durch

$$\delta^{P_3}(q_{(i,l)}, \sigma, q_{(j,m)}, d) = \begin{cases} \delta^{P_2}(q_i, \sigma, q_j, d, \varepsilon), & \text{falls } m = l, \\ \sum_{y \in \Sigma_{aus}} \delta^{P_2}(q_i, \sigma, q_j, d, y) \delta^{P_1}(q_l, y, q_m), & \text{sonst} \end{cases}$$

definiert. Anschaulich können wir uns vorstellen, dass in jedem Zustand des Automaten P_1 ein kleiner Automat P_2 gebaut wird (vgl. Abbildung 2.1 und Abbildung 2.2). In Abbildung 2.1 gibt es für alle i aus $\{0, \ldots, |Q^{P_1}|\}$ ein j aus $\{0, \ldots, |Q^{P_1}|\}$ mit $\delta^{P_1}(q_i, \sigma, q_j) = 1$. Außerdem hängt der Ausgabebuchstabe eindeutig von dem erreichten Zustand und dem gelesenen Zeichen ab. Allgemein gilt dies so nicht.

Das Verhalten der Automaten P_2 und P_1 wird von P_3 parallel simuliert. Der Automat beginnt die Berechnung im Zustand $q_{(0,0)}$ in der Kopie $Q_0^{P_2}$ der Zustandsmen-

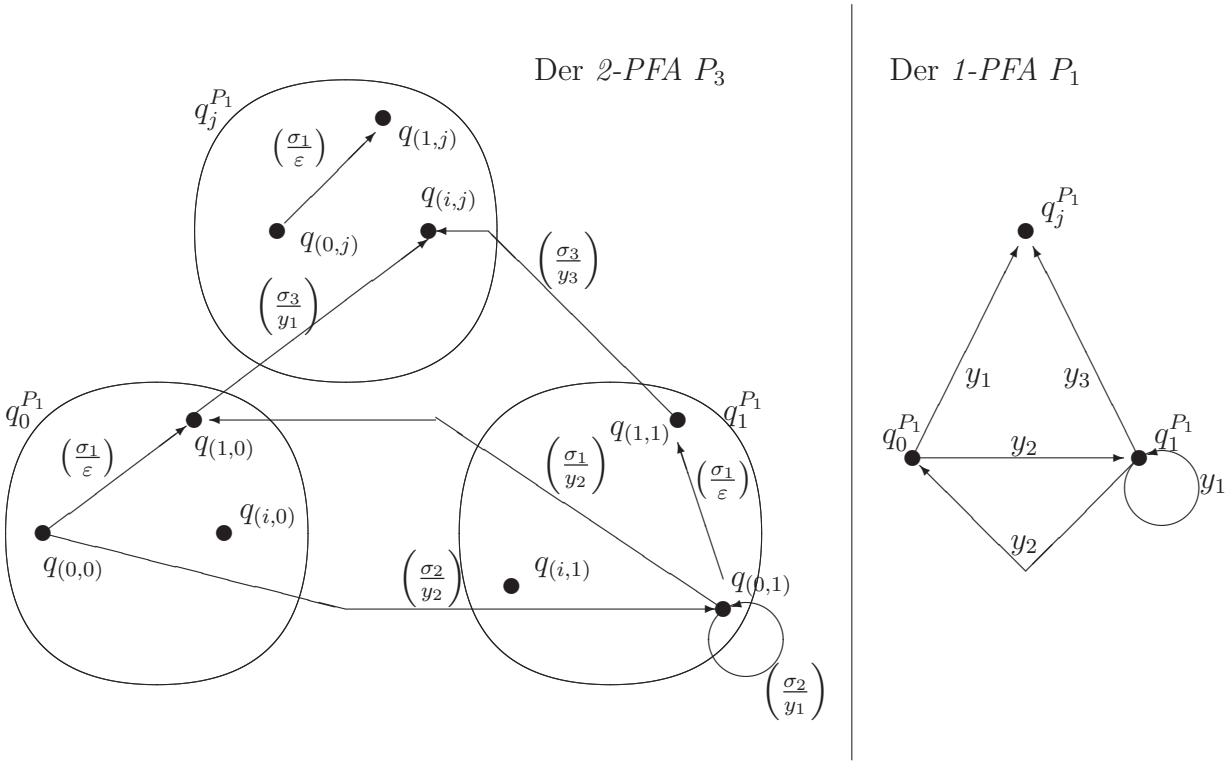

Abbildung 2.1: **Links:** Ausschnitt aus dem Automaten C. Kleine Schwarze Punkte stellen Zustände des Automaten P_2 dar. Große Kreise symbolisieren die Zustände von P_1. In jedem Zustand von P_1 ist eine Kopie aller Zustände von P_2 enthalten. Die Beschriftungen an den Überführungspfeilen folgen der Notation $\left(\frac{\text{Eingabebuchstabe}}{\text{Ausgabebuchstabe von } P_2}\right)$. **Rechts:** Ausschnitt aus dem Automaten P_1.

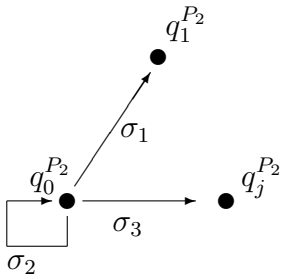

Abbildung 2.2: Ausschnitt aus dem *2-PFA* P_2. Die Abbildung berücksichtigt nur die Zustandsübergänge. Die Kopfbewegung ist aus der Abbildung nicht abzulesen.

ge von P_2, die dem Startzustand q_0 von P_1 entspricht. In dieser Kopie $Q_0^{P_2}$ wird solange die Berechnung des Automaten P_2 simuliert, bis P_2 zum ersten Mal eine Ausgabe schreibt. Eventuell sind für $k \in \mathbb{N}$ verschiedene Ausgaben $y_1, \ldots, y_k$ möglich, wobei für $i \in \{1, \ldots, k\}$ die Ausgabe y_i in dem Zustand q bei dem gelesenem Zeichen σ mit Wahrscheinlichkeit $\sum_{d \in \{-1,0,1\}, q' \in Q} \delta^A(q, \sigma, q', d, y_i)$ geschrieben wird. Der Automat erreicht dann mit Wahrscheinlichkeit $\sum_{i \in \{1,\ldots,k\}, d \in \{-1,0,1\}} \delta^{P_2}(q, \sigma, q_j, d, y_i)\delta^{P_1}(q_0, y_i, q_m)$ die Zustandsmenge $Q_m^{P_2}$, die dem Zustand q_m von P_1 entspricht. Der Nachfolgezustand ist nämlich $q_{(j,m)}$. Wenn eine Kopie von P_2 erreicht wird, die einem akzeptierenden Zustand von P_1 entspricht, akzeptiert P_3. Dies entspricht der Situation, dass P_1 beim Lesen der Ausgabe von P_2 akzeptiert. Die Ausgabe muss dazu nicht komplett gelesen werden. Die Eingabe wird verworfen, wenn ein haltender Zustand von P_2 in einer Kopie von P_2 erreicht wird, die keinem akzeptierenden Zustand von P_1 entspricht, oder wenn ein verwerfender Zustand von P_1 erreicht wird. Das entspricht der Situation, dass P_1 den letzten Buchstaben der Eingabe gelesen hat, ohne zu akzeptieren oder, dass P_1 beim Lesen der Ausgabe von P_2 verwirft.

Jeder Schritt von P_3 entspricht einem Schritt von P_2. Der *2-PFA* P_3 wechselt lediglich noch zwischen den verschiedenen Kopien von P_2. Der Automat P_3 hat also dieselbe Laufzeit wie P_2 und akzeptiert wie P_1 alle Eingaben $w \in L_{ein}$ mindestens mit Wahrscheinlichkeit $1 - \varepsilon$, Eingaben $w' \notin L_{ein}$ dagegen nur mit Wahrscheinlichkeit kleiner als ε, weil die Berechnung von P_2 und P_1 parallel simuliert wird. Es ist $|Q^{P_3}| = |Q^{P_1}| \cdot |Q^{P_2}|$. □

Satz 2.2-10
Jede von einem 1-PFA$^{(2\text{-}PFA)}$ in erwarteter polynomieller Zeit erkannte Sprache ist regulär.

Beweis. Die Behauptung folgt aus Lemma 2.2-9 und Satz 2.2-4. □

Es bleibt zu untersuchen, welche Mengen von Sprachen von einem *2-PFA*$^{(2-PFA)}$ erkannt werden kann. Eine einfache Simulation wie in Lemma 2.2-9 ist nicht möglich, weil nicht klar ist, wie die Kopfbewegung beider Automaten berücksichtigt werden kann.

Kapitel 3

Bekannte Quantenautomaten

Wir werden in diesem Kapitel einige Automaten definieren, die auf die Gesetzmäßigkeiten aus der Quantentheorie (vgl. Kapitel 1.2) zurückgreifen. Wir können uns vorstellen, dass ein Quantenautomat über ein Register aus Qubits verfügt, auf das er die notwendigen Transformationen anwendet. Wir werden untersuchen, ob dadurch ein Automatenmodell entsteht, das mehr als nur die Menge der regulären Sprachen erkennen kann. Ein Quantenautomat besitzt eine Zustandsmenge Q. Jeder Zustand $q \in Q$ wird durch einen Vektor $|i\rangle \in \mathbb{C}^{|Q|}, i \in \{1, \ldots, |Q|\}$ repräsentiert. Es sind $\lceil \log_2 |Q| \rceil$ Qubits notwendig, um alle Zustände in Q beschreiben zu können. Ein bemerkenswerter Unterschied von Quantenautomaten zu klassischen Automaten ist, dass sich ein Quantenautomat in einer Überlagerung aus allen Zuständen befinden kann. Sei $|\psi\rangle \in \mathbb{C}^{|Q|}$ eine solche Überlagerung. Dann ist $|||\psi\rangle||_2^2 = 1$. Die Überführungsfunktion δ der Quantenautomaten kann durch Überführungsmatrizen beschrieben werden (vgl. Kapitel 1.2.1). Sei $|\psi\rangle$ die gegebene Überlagerung eines Quantenautomaten und M eine unitäre Überführungsmatrix. Dann ist $|\psi'\rangle = M|\psi\rangle$ die Überlagerung des Automaten nach dem Anwenden der Transformation M.

3.1 2-Wege Automaten

Der Automatentyp des *2-QFA*s wird in [KW97] definiert. Wir übernehmen diese Definition in den wesentlichen Punkten.

Definition 3.1-1
Ein 2-QFA $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ verfügt über ein Eingabeband unbeschränkter Länge, eine endliche Zustandsmenge Q, ein endliches Eingabealphabet Σ, einen eindeutigen Startzustand q_0, eine Überführungsfunktion δ sowie Mengen Q_{acc} bzw. Q_{rej} von akzeptierenden und verwerfenden Zuständen. Wir nennen $Q_{non} = Q \setminus (Q_{acc} \cup Q_{rej})$ die Menge der nicht-haltenden Zustände. Es ist $Q_{acc} \cap Q_{rej} = \emptyset$ und $q_0 \in Q_{non}$. Seien $\dashv, \vdash \notin \Sigma$ zwei Endmarkierungen, dann ist $\Gamma = \Sigma \cup \{\dashv, \vdash\}$ das Bandalphabet des Automaten. Zu Beginn der Berechnung steht die Eingabe in der Form $\dashv w \vdash$ auf dem Band. Die Überführungsfunktion ist $\delta : Q_{non} \times \Gamma \times Q \times \{-1, 0, 1\} \to \mathbb{C}$.

Wir beschreiben die Berechnung des Automaten durch Konfigurationen. Eine Konfiguration ist ein Paar aus Zustand und Bandposition. Wenn eine Eingabe die Länge n hat, gibt es $n \cdot |Q|$ Konfigurationen, in denen sich der Automat befinden kann. Sei C_n die Menge aller Konfigurationen bei einer Eingabelänge von n mit $C_n = Q \times \mathbb{Z}_n$. Der Automat befindet sich zu jedem Zeitpunkt in einer Linearkombination von möglichen Konfigurationen. In Anlehnung an Kapitel 1.2 sprechen wir dann von einer Überlagerung des Automaten.

Eine Überlagerung des Automaten ist ein Element eines komplexen Vektorraumes mit innerem Produkt, dessen L_2-Norm den Wert 1 hat. Der komplexe Vektorraum wird durch $H_n = span\{|i\rangle | i \in \{1, \ldots, |Q| \cdot n\}\}$ aufgespannt. Zu Beginn der Berechnung befindet sich der Automat in der Überlagerung $|q_0, 0\rangle$. Das heißt, der Automat befindet sich im Zustand q_0 und der Lesekopf liest die linke Endmarkierung an Position 0 des Eingabebandes. Für jeden Eingabebuchstaben $\sigma \in \Gamma$ bekommen wir aus δ einen Operator M_σ auf H_n, mit dem die Berechnung des Automaten beschrieben werden kann. Sei $w = w_1, \ldots, w_n$ eine Eingabe des Automaten mit $\sigma \in \{w_1, \ldots, w_n\}$. Dann ist für $q \in Q, k \in \{1, \ldots, |w|\}$,

$$M_\sigma |q, k\rangle = \sum_{q' \in Q, d \in \{-1,0,1\}} \delta(q, \sigma, q', d) |q', (k + d) mod (|w| + 2)\rangle,$$

wobei $d = -1$ eine Linksbewegung des Lesekopfes repräsentiert, $d = 1$ eine Rechtsbewegung und $d = 0$ ausdrückt, dass der Lesekopf in diesem Rechenschritt nicht bewegt wird. In jedem Rechenschritt wird zunächst die gegenwärtige Überlagerung gemessen und danach abhängig von dem gelesenen Zeichen der zugehörige Operator angewandt. Sei der Vektor $|\psi\rangle = \sum_{q \in Q, k \in \{1,\ldots,n\}} \alpha_{q,k} |q, k\rangle$ die gegenwärtige Überlagerung des Automaten. Dann wird die Konfiguration $|q, k\rangle$ mit Wahrscheinlichkeit $|\alpha_{q,k}|^2$ gemessen. Wenn für ein $k \in \{1, \ldots, n\}$ eine Konfiguration $|q, k\rangle$ mit $q \in Q_{acc}$ oder $q \in Q_{rej}$ gemessen wird, stoppt der Automat und akzeptiert bzw. verwirft. Andernfalls ist die Überlagerung nach der Messung $\frac{1}{||\psi_{non}\rangle||_2} |\psi\rangle$ mit $|\psi_{non}\rangle = \sum_{q \in Q_{non}, k \in \{1,\ldots,n\}} \alpha_{q,k} |q, k\rangle$. Für $\varepsilon \in [0, \frac{1}{2}[$ akzeptiert A eine Sprache L mit einem Fehler ε, falls

1. $\forall\ w \in\ L:\ \mathcal{P}[A \text{ akzeptiert } w] \geq 1 - \varepsilon$,
2. $\forall\ w \notin\ L:\ \mathcal{P}[A \text{ verwirft } w] < \varepsilon$

gilt.

Bemerkung 3.1-2
Der 2-QFA kann sich in mehreren Zuständen gleichzeitig befinden und den Lesekopf zur selben Zeit an verschiedenen Positionen des Eingabebandes haben. Das Eingabeband des Automaten ist zyklisch. Wenn der Automat den Lesekopf über eines der Zeichen $\dashv$ oder $\vdash$ hinausbewegt, liest er danach am anderen Ende der Eingabe die entsprechende andere Endmarkierung.

In [MC00] werden *QFA*s etwas anders definiert. Der Unterschied ist, dass eine Messung von einem Automaten aus [MC00] erst nach dem Lesen des gesamten Eingabewortes

durchgeführt wird und nicht schon nach jedem Rechenschritt. Weil damit in der gesamten Berechnung stets genau eine Messung durchgeführt wird, werden diese Automaten in der Literatur häufig als *MO-QFA*s bezeichnet. Dabei steht *MO* für *measurement once*. In dem Modell aus Definition 3.1-1 wird in jedem Schritt eine Messung durchgeführt. In der Literatur wird deswegen häufig die Bezeichnung *MM-QFA* für *many measurements* verwendet. Ein *MO-QFA* ist ein spezieller *MM-QFA* der erst beim Lesen der Rechten Endmarkierung einen haltenden Zustand erreichen kann. Deswegen kann jede von einem *MO-QFA* erkannte Sprache auch von einem *MM-QFA* mit derselben Akzeptanzwahrscheinlichkeit erkannt werden. Die umgekehrte Richtung stimmt jedoch nicht [AF98, BP02]. Wir werden deswegen nicht weiter auf *MO-QFA*s eingehen. Die im weiteren behandelten Quantenautomaten sind stets *MM-QFA*s. Wir verzichten deswegen auf den Präfix „*MM-*“.

Definition 3.1-3
Einen QFA, bei dem sich aus jeder gültigen Überlagerung stets eine gültige Überlagerung entwickelt, nennen wir abgeschlossen.

Lemma 3.1-4 *(Proposition 1 in [KW97])*
Ein 2-QFA $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ *ist genau dann abgeschlossen, wenn für jede Wahl von* $\sigma, \sigma_1, \sigma_2 \in \Gamma$ *und* $q_1, q_2 \in Q$ *gilt:*

1. $\sum_{q',d} \delta^*(q_1, \sigma, q', d)\delta(q_2, \sigma, q', d) = \begin{cases} 1, & \text{falls } q_1 = q_2, \\ 0, & \text{sonst,} \end{cases}$
2. $\sum_{q'} (\delta^*(q_1, \sigma_1, q', 1)\delta(q_2, \sigma_2, q', 0) + \delta^*(q_1, \sigma_1, q', 0)\delta(q_2, \sigma_2, q', -1)) = 0,$
3. $\sum_{q'} \delta^*(q_1, \sigma_1, q', 1)\delta(q_2, \sigma_2, q', -1) = 0.$

Wir wissen bereits aus Lemma 2.1-9 (Seite 27), dass jeder *1-DFA* von einem *2-RFA* simuliert werden kann. Ein *2-RFA* ist ein spezieller *2-QFA*, bei dem für alle $q_1, q_2 \in Q$, $d \in \{-1, 1\}, \sigma \in \Gamma$ gilt $\delta(q_1, \sigma, q_2, d) \in \{0, 1\}$. Wir erhalten deswegen direkt das folgende Lemma.

Lemma 3.1-5 *(Proposition 4 in [KW97])*
Sei $D = (Q, \Sigma, q_0, \delta, Q_{acc})$ *ein beliebiger 1-DFA. Dann kann zu D ein 2-QFA A definiert werden, sodass für beliebige* $w \in \Sigma^*$ *gilt* (D *akzeptiert* w) $\Leftrightarrow$ (A *akzeptiert* w). *Die Laufzeit von A ist* $O(|w|)$.

Lemma 3.1-6 *(Proposition 2 aus [KW97])*
Sei $w \in \{a, b\}^*$ *und* $N \in \mathbb{N}$ *beliebig. Es gibt einen 2-QFA A, der bei Eingabe w akzeptiert, falls* $w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ *ist und der andernfalls höchstens mit Wahrscheinlichkeit* $\frac{1}{N}$ *akzeptiert. Der Automat A hält nach* $O(N \cdot |w|)$ *Schritten.*

Wir haben also ein Modell gefunden, das mächtiger als das Modell des *1-DFA* ist. Das gilt sogar, wenn die Rechenzeit polynomiell ist. Allerdings scheint das Modell des *2-QFA*

aus einem anderen Grund unhandlich. Weil wir zulassen, dass auch die Kopfposition des Automaten durch Qubits kodiert wird, wächst die Anzahl der von dem Automaten benötigten Qubits, wenn die Länge des Eingabewortes wächst. Wir sind scheinbar etwas über das Ziel hinausgeschossen und werden im Folgenden QFAs betrachten, die einen Quantenteil mit konstanter Größe besitzen.

3.2 1-Wege Automaten

Definition 3.2-1
Ein 1-QFA $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ verfügt über ein Eingabeband unbeschränkter Länge, eine endliche Zustandsmenge Q, ein endliches Eingabealphabet Σ, einen eindeutigen Startzustand q_0, eine Zustandsüberführungsfunktion δ sowie zwei Mengen Q_{acc} und Q_{rej} von akzeptierenden bzw. verwerfenden Zuständen. Die Menge der nicht-haltenden Zustände schreiben wir als $Q_{non} = Q \setminus (Q_{acc} \cup Q_{rej})$. Es ist $Q_{acc} \cap Q_{rej} = \emptyset, q_0 \in Q_{non}$ und $(Q_{acc} \cup Q_{rej}) \subseteq Q$. Für zwei Endmarkierungen $\dashv, \vdash \notin \Sigma$ bezeichnen wir mit $\Gamma = \Sigma \cup \{\dashv, \vdash\}$ das Bandalphabet des Automaten. Die Überführungsfunktion ist

$$\delta : Q \times \Gamma \times Q \to \mathbb{C}.$$

Der Automat befindet sich stets in einer Überlagerung von Zuständen aus Q. Wir können jede Überlagerung als Vektor $|\psi\rangle \in \mathbb{C}^{|Q|}$ schreiben. Jeder Zeile $\langle i|\psi\rangle, i \in \{1, \ldots, |Q|\}$ von $|\psi\rangle$ ist dabei ein Zustand $q \in Q$ zugeordnet. Wir assoziieren also die Zustände $|q_0\rangle, \ldots, |q_{|Q|-1}\rangle$ mit der ON-Basis $|1\rangle, \ldots, ||Q|\rangle$. Für jede zulässige Überlagerung $|\psi\rangle$ ist $|||\psi\rangle||_2^2 = 1$.

Wir können die Überführungsfunktion δ durch eine Menge von Überführungsmatrizen beschreiben. Zu einem Zeichen $\sigma \in \Gamma$ sei M_σ die zugehörige Überführungsmatrix (vgl. Kapitel 1.2.1). Dann ist M_σ eine $|Q| \times |Q|$-Matrix. Für $i, j \in \{1, \ldots, |Q|\}$ seien q_i bzw. q_j der i-te und j-te Zustand in Q. Dann hat der Eintrag m_{ij}^σ in M_σ den Wert $\delta(q_i, \sigma, q_j)$. Für alle $\sigma \in \Gamma$ ist M_σ eine unitäre Matrix. Das Anwenden einer Transformation M_σ auf eine Überlagerung $|\psi\rangle \in \mathbb{C}^{|Q|}$ kann beschrieben werden als $|\psi'\rangle = M_\sigma|\psi\rangle$.

Zu Beginn der Berechnung steht die Eingabe w in der Form $\dashv w \vdash$ auf dem Band. Der Automat befindet sich im Zustand q_0. Die Überlagerung des Automaten zu Beginn der Berechnung ist also $|1\rangle$. Beginnend mit der linken Endmarkierung $\dashv$ wird die Eingabe von links nach rechts gelesen. Sei allgemein $|\psi_1\rangle$ die zu Beginn eines Rechenschritts erreichte Überlagerung. In einem Rechenschritt wird zunächst $|\psi_1\rangle$ gemessen. Sei O_{acc} eine $|Q| \times |Q|$ Matrix, mit

$$o_{ij} = \begin{cases} 1, & \text{falls } i = j \text{ und } q_{i-1} \in Q_{acc}, \\ 0, & \text{sonst.} \end{cases}$$

Analog seien zwei Matrizen O_{rej} und O_{non} definiert. Sei $|\psi_{acc}\rangle = O_{acc}|\psi_1\rangle$, $|\psi_{rej}\rangle = O_{rej}|\psi_1\rangle$ und $|\psi_{non}\rangle = O_{non}|\psi_1\rangle$. Dann führt das Messen von $|\psi_1\rangle$ mit Wahrscheinlichkeit $|||\psi_{acc}\rangle||_2^2$ bzw. $|||\psi_{rej}\rangle||_2^2$ dazu, dass der Automat die Berechnung akzeptierend oder verwerfend abbricht. Mit Wahrscheinlichkeit $|||\psi_{non}\rangle||_2^2$ wird die Berechnung in der Überla-

gerung $|\psi_1'\rangle = \frac{|\psi_{non}\rangle}{|||\psi_{non}\rangle||_2}$ *fortgesetzt. Dazu wird bei gelesenem Zeichen* σ *die Transformation* $|\psi_2\rangle = M_\sigma|\psi_1'\rangle$ *durchgeführt und der Lesekopf einen Schritt nach rechts bewegt. Der Rechenschritt ist damit beendet und es beginnt ein neuer Rechenschritt. Spätestens beim Erreichen der rechten Endmarkierung* $\vdash$ *akzeptiert oder verwirft der Automat. Eine Rechtsbewegung über* $\vdash$ *hinaus ist nicht möglich. Sei* $\varepsilon \in [0, \frac{1}{2}[$*. Dann erkennt* A *eine Sprache* L *mit einem Fehler von* ε*, wenn*

1. $\forall\ w \in L : \mathcal{P}[A \text{ akzeptiert } w] \geq 1 - \varepsilon$,
2. $\forall\ w \notin L : \mathcal{P}[A \text{ akzeptiert } w] < \varepsilon$

gilt.

Bemerkung 3.2-2
Ein 1-QFA ist abgeschlossen, da für die Überführungsfunktion δ *gilt*

$$\delta(q, \sigma, q') = \langle q'|M_\sigma|q\rangle.$$

Diese Behauptung erhalten wir aus Lemma 3.1-4, da die zweite und dritte Bedingung von dort zwangsläufig erfüllt sind, weil ein 1-QFA den Kopf nur nach rechts bewegen kann.

Wir können feststellen, dass ein *1-QFA* nicht mehr Sprachen erkennen kann, als ein *1-DFA*.

Lemma 3.2-3 *(Proposition* 6 *aus [KW97])*
Sei L *eine beliebige von einem 1-QFA erkannte Sprache. Dann ist* L *regulär.*

In [KW97] gibt es eine Beweisskizze zu diesem Lemma.

Lemma 3.2-4 *(Proposition* 7 *aus [KW97])*
Die Sprache $L = \{a, b\}^* a$ *kann nicht von einem 1-QFA erkannt werden.*

Dieses Ergebnis ist zunächst ernüchternd. Quantenautomaten, die auf ein konstant großes Register aus Qubits zugreifen, können weniger Sprachen erkennen, als *1-DFA*s. Es kann jedoch beobachtet werden, dass die Menge der von *1-QFA*s erkannten Sprachen mit der verwendeten Akzeptanzwahrscheinlichkeit des Automaten variiert [AF98]. Wird eine kleinere Akzeptanzwahrscheinlichkeit verwendet, können mehr Sprachen erkannt werden. In [ABFK99] wird abhängig von der Akzeptanzwahrscheinlichkeit eine Hierarchie von erkannten Sprachen angegeben. Außerdem wird in [AF98] gezeigt, dass es einige Sprachen gibt, für die sowohl ein *1-PFA* als auch ein *1-DFA* exponentiell mehr Zustände benötigt, als ein *1-QFA*. Dabei muss aber beachtet werden, dass *1-QFA*s für eine Sprache L nicht immer kleiner als der minimale *1-DFA* bzw. *1-PFA* für dieselbe Sprache sind. In [AAN99] (Theorem 1.3) wird eine Familie von Sprachen vorgestellt, für deren Erkennen ein *1-QFA* exponentiell mehr Zustände benötigt, als ein *1-DFA*.

Kapitel 4

Erweiterungen von Quantenautomaten

Es gibt einige Überlegungen, wie *1-QFA*s oder *2-QFA*s modifiziert werden können, sodass das Modell eine konstante Größe hat, aber dennoch mehr als nur die Menge der regulären Sprachen erkennen kann. Häufig ist dabei die Richtung, in die der Lesekopf des Automaten bewegt werden darf, nicht eindeutig (siehe zum Beispiel [ABFG99, AI99]), oder der Automat wird um ein Modul erweitert, das ihm das Zählen ermöglicht (siehe zum Beispiel [BFK01, Kra99]). Wir werden diese Modelle hier nicht betrachten, sondern uns auf weniger einschneidende Modifikationen von *1-QFA*s konzentrieren. Die in diesem Kapitel vorgestellten Automaten haben eine eindeutige Leserichtung und werden nicht durch zusätzliche Module erweitert. Stattdessen werden wir untersuchen, wie sich eine alternative Überführungsfunktion, ein zyklisches Leseband sowie Kombinationen mit verschiedenen klassischen Automaten auswirken.

4.1 Verallgemeinerte Messungen

Wir werden nun ein Modell vorstellen, bei dem wir im Vergleich zu *1-QFA*s Änderungen an der Überführungsfunktion und den Messungen durchführen. Die Überführungsfunktion des Automaten ist im Vergleich zu der Überführungsfunktion von *1-QFA*s relaxiert und die Messungen sind an eine feinere Aufteilung der Zustandsmenge angepasst. Das Modell wird im Folgenden *1-QFA* mit verallgemeinerten Messungen (*1-QFA*^{vm}) genannt.

Definition 4.1-1
Ein 1-QFA^{vm} $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ *verfügt über eine endliche Zustandsmenge* Q*, ein endliches Eingabealphabet* Σ*, einen eindeutigen Startzustand* q_0 *sowie disjunkte Mengen* Q_{acc} *und* Q_{rej} *von akzeptierenden bzw. verwerfenden Zuständen. Wir nennen* $Q_{non} = Q \setminus (Q_{acc} \cup Q_{rej})$ *die Menge der nicht-haltenden Zustände. Die Menge* Q_{non} *wird für* $k \in \mathbb{N}$ *weiter unterteilt in* $k - 2 \geq 1$ *disjunkte Teilmengen* $\{Q_0, \ldots, Q_{k-3}\}$ *mit* $Q_{non} = \bigcup_{i=0}^{k-3} Q_i$*. Insgesamt ist* Q *also in* k *disjunkte Zustandsmengen unterteilt. Es ist* $q_0 \in Q_0$*. Seien* $\dashv, \vdash \notin \Sigma$ *zwei Endmarkierungen. Dann ist* $\Gamma = \Sigma \cup \{\dashv, \vdash\}$ *das Bandalphabet des Automaten.*

Für alle Paare $q_1, q_2 \in Q_i$, $i \in \{0, \ldots, k-3\}$, ist

$$\forall\ \sigma \in \Gamma : \sum_{q \in Q} \delta^*(q_1, \sigma, q)\delta(q_2, \sigma, q) = \begin{cases} 1, & \textit{falls } q_1 = q_2, \\ 0, & \textit{sonst.} \end{cases} \tag{4.1}$$

In jedem Rechenschritt wird zunächst die gegenwärtige Überlagerung gemessen und danach die zu dem gelesenen Zeichen gehörende Zustandsüberführung auf der aktuellen Überlagerung durchgeführt.

Wenn bei einer Messung ein Zustand aus Q_{acc} oder Q_{rej} gemessen wird, bricht die Berechnung akzeptierend bzw. verwerfend ab. Wenn ein Zustand $q \in Q_i, i \in \{0, \ldots, k-3\}$, gemessen wird, setzt der Automat die Berechnung in dem Untervektorraum $E_i = span\{|q\rangle | q \in Q_i\}$ fort. Sei $|\psi\rangle$ die Überlagerung des Automaten, bevor eine Messung durchgeführt wird und sei $q \in Q_i, i \in \{1, \ldots, k-3\}$, der gemessene Zustand. Dann ist

$$|\psi_i\rangle = \frac{\sum_{q_j \in Q_i} \langle j+1|\psi\rangle}{||\sum_{q_j \in Q_i} \langle j+1|\psi\rangle||_2}$$

die Überlagerung des Automaten nach der Messung. Der Automat akzeptiert eine Sprache L mit einem Fehler von $\varepsilon \in [0, \frac{1}{2}[$, falls

1. $\forall\ w \in\ L : \mathcal{P}[A \textit{ akzeptiert }] \geq 1 - \varepsilon$,
2. $\forall\ w \notin\ L : \mathcal{P}[A \textit{ akzeptiert }] < \varepsilon$.

gilt.

Wir verwenden wieder Überführungsmatrizen für die Beschreibung der Zustandsüberführungsfunktion δ. Ein Matrixeintrag für $\sigma \in \Gamma$ für alle $i, j \in \{1, \ldots, n\}$ ist $m_{ij} = \delta(q_j, \sigma, q_i)$.

Bemerkung 4.1-2
Wegen 4.1 ist für $\sigma \in \Gamma$ eine Überführungsmatrix M_σ eines 1-QFA^{vm} nicht zwingend unitär.

Wir können recht einfach einsehen, dass ein *1-QFA^{vm}* mindestens so viele Sprachen wie ein *1-QFA* erkennen kann, da ein *1-QFA* ein Spezialfall eines *1-QFA^{vm}* ist, bei dem die Menge Q_{non} in genau eine Menge $Q_0 = Q_{non}$ unterteilt ist und bei dem die Bedingung 4.1 zusätzlich auf die Mengen Q_{acc} und Q_{rej} ausgeweitet ist. Es können sogar alle regulären Sprachen erkannt werden.

Satz 4.1-3
Sei $D = (Q, \Sigma, q_0, \delta, Q_{acc})$ ein 1-DFA für eine Sprache L. Dann existiert ein 1-QFA^{vm} A für L. Die Größe des 1-QFA^{vm} ist $|Q| + 2$, die Rechenzeit des 1-QFA^{vm} ist nicht größer als die Rechenzeit des 1-DFA für L.

Beweis. Wir zeigen, dass ein *1-QFA^{vm}* einen *1-DFA* D simulieren kann. D hat $|Q| = r$ Zustände, von denen einige aus der Menge Q_{acc} der akzeptierenden Zustände sind. Wir übernehmen alle r Zustände von D in die Zustandsmenge des *1-QFA^{vm}* und nennen diesen Automaten $A = (Q^A, \Sigma, q_0, \delta^A, Q^A_{acc}, Q^A_{rej})$. Die Zustandsmenge Q^A teilt sich in $r+2$ Untermengen Q_i mit $i \in \{acc, rej, 0, \ldots, r-1\}$, $q_i \in Q^A_i$, auf. Für alle i ist also $|Q^A_i| = 1$. Das Bandalphabet wird um die beiden Endmarkierungen $\dashv$ und $\vdash$ erweitert. Die Überführungsfunktion δ^A ergibt sich folgendermaßen aus der Überführungsfunktion δ des *1-DFA*. Für $q, q' \in Q$, $\sigma \in \Gamma$, ist

$$\delta^A(q, \sigma, q') = \begin{cases} 1, & \text{falls } \delta(q, \sigma) = q', \\ 0, & \text{sonst.} \end{cases}$$

Außerdem ist $\delta^A(q, \vdash, q_{acc}) = 1$ für $q \in Q^A_{acc}$ und $\delta^A(q, \vdash, q_{rej}) = 1$ falls $q \notin Q^A_{acc}$ sowie $\delta^A(q_0, \dashv, q_0) = 1$. Wegen $|Q^A_i| = 1$ gilt für den Automaten A die Gleichung 4.1 aus Definition 4.1-1. □

Bemerkung 4.1-4
Es gibt einen 1-QFA^{vm} für eine reguläre Sprache L mit maximal k Zuständen. Dabei entspricht k dem Minimum der Größe eines minimalen 1-QFA, bzw. eines minimalen 1-DFA für die Sprache L, weil beide Automatentypen von dem 1-QFA^{vm} mit maximal zwei zusätzlichen Zuständen simuliert werden können.

Ein ähnliches Modell wird in [Hir01] vorgestellt. Es wird in [Hir01] behauptet, dass jede von dem Modell erkannte Sprache regulär ist. Wir können leider keine vergleichbare Aussage für den *1-QFA^{vm}* zeigen.Die Hinzunahme der verallgemeinerten Messungen hat das Modell des *1-QFA* zwar mächtiger gemacht, allerdings können wir nicht zeigen, ob mehr als die Menge der regulären Sprachen erkannt werden können. Wir werden deswegen noch weitere Veränderungen an diesem Modell vornehmen.

4.1.1 Zyklisches Eingabeband und ein spezieller Akzeptanzmodus

Wir betrachten hier einen *1-QFA^{vm}*, der über ein zyklisches Eingabeband verfügt (*1-QXFA^{vm}*).

Definition 4.1-5
Ein 1-QXFA^{vm} $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ ist eine Verallgemeinerung eines 1-QFA^{vm}. Der Automat A ist ein 1-QFA^{vm}, der über ein zyklisches Eingabeband verfügt. Wenn A beim Lesen der rechten Endmarkierung nicht in einen haltenden Zustand fällt, bewegt sich der Lesekopf ein Zeichen nach rechts und liest dann die linke Endmarkierung.

Bemerkung 4.1-6
Da ein 1-QXFA^{vm} eine Verallgemeinerung eines 1-QFA^{vm} ist, können alle von einem 1-QFA^{vm} erkannten Sprachen auch von einem 1-QXFA^{vm} erkannt werden. Die Laufzeit

des 1-QXFA^{vm} ist für die betrachtete Sprache nicht größer als für den 1-QFA^{vm}. Der 1-QXFA^{vm} hat maximal so viele Zustände, wie der 1-QFA^{vm}.

Um die Diskussion im Folgenden zu vereinfachen, definieren wir zunächst, was ein Lesezyklus ist.

Definition 4.1-7
Unter einem Lesezyklus eines Automaten mit zyklischem Band verstehen wir das einmalige Lesen des Eingabewortes inklusive der beiden Endmarkierungen. Das erste in einem Lesezyklus von dem Automaten gelesene Zeichen ist die linke Endmarkierung ⊣.

Wir stellen uns zunächst die Frage, ob ein *1-QXFA* die Sprache aller Pallindrome aus zwei verschiedenen Buchstaben ($L_{pal} = \{w|w \in \{a,b\}^* \text{ und } w = w^{-1}\}$) erkennen kann, wenn ein Nicht-Halten des Automaten als Akzeptieren interpretiert wird.

Bemerkung 4.1-8
Die Sprache $L_{pal} = \{w|w \in \{a,b\}^*$ und $w = w^{-1}\}$ kann nicht von einem 2-PFA erkannt werden (siehe [DS92]).

Damit kann L_{pal} von keinem der in Kapitel 2 vorgestellten Automatentypen erkannt werden. Viele Ideen für den vorzustellenden *1-QXFA^{vm}* für die Sprache L_{pal} sind stark von einer Diskussion in [AW02] inspiriert. Mit einigen Ergebnissen aus Kapitel 1 erhalten wir leicht das folgende Ergebnis.

Satz 4.1-9
Sei $w \in \{a,b\}^*$. Es gibt einen 1-QXFA^{vm}, der bei der Eingabe w nie hält, falls $w \in L_{pal} = \{w|w \in \{a,b\}^*$ und $w = w^{-1}\}$ ist und andernfalls mit Wahrscheinlichkeit Eins verwirft. Für $|w| = n$ und $w \notin L_{pal}$ ist die erwartete Rechenzeit des Automaten $O(25^n)$.

Beweis. Wir definieren den *1-QXFA^{vm}* als $A = (Q, \Sigma, q_0, \delta, Q_{rej})$. Die Zustandsmenge des Automaten ist $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q^1_{rej}, q^2_{rej}\}$ mit $q^1_{rej}, q^2_{rej} \in Q_{rej}$ und dem Startzustand q_0. Die Überführungsfunktion δ wird durch die folgenden 8×8 Überführungsmatrizen

definiert.

$$M_a = \begin{bmatrix} B_1 & & & \\ & B_1^{-1} & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \qquad M_b = \begin{bmatrix} B_2 & & & \\ & B_2^{-1} & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

$$M_\dashv = I \qquad M_\vdash = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & & & & & & 1 & \\ 0 & & & & & & & 1 \\ 1 & & & & & & & \\ 0 & 1 & & & & & & \\ 0 & & 1 & & & & & \\ 0 & & & & 1 & & & \\ 0 & & & & & 1 & & \end{bmatrix}$$

Um eine bessere Lesbarkeit zu gewährleisen, sind die meisten Felder mit Nullen weggelassen. Außerdem sind B_1 und B_2 jeweils 3×3 Matrizen wie in 1.5 und 1.6 definiert. Weil $B_1, B_2, B_1^{-1}, B_2^{-1}$ unitär sind (vgl. Bemerkung 1.2-2 auf Seite 16), sind auch alle Überführungsmatrizen $M_\sigma, \sigma \in \{a, b, \vdash, \dashv\}$ unitär. Bei einer Eingabe w ist die Berechnung des Automaten gewissermaßen in zwei Phasen geteilt, wobei in jeder Phase ein Lesezyklus durchgeführt wird. Die Eingabe w wird also in einer Phase einmal komplett gelesen. Wir assoziieren die Zustände $q_0, q_1, q_2, q_3, q_4, q_5, q_{rej^1}$ und q^2_{rej} mit der ON-Basis $|1\rangle, \ldots, |8\rangle$. In der ersten Phase werden nur die Amplituden der Vektoren $|1\rangle, |2\rangle$ und $|3\rangle$ modifiziert, in der zweiten Phase die Amplituden der Vektoren $|4\rangle, |5\rangle$ und $|6\rangle$. Nur am Ende der zweiten Phase ist es möglich, dass die Amplituden von $|7\rangle$ oder $|8\rangle$ einen anderen Wert als Null annehmen. Die Vektoren $|7\rangle$ und $|8\rangle$ sind mit den Zuständen q^1_{rej} und q^2_{rej} assoziiert. In der ersten Phase wird für jedes Vorkommen von a die Transformation B_1 auf die gegenwärtige Überlagerung angewandt und für jedes Zeichen b die Transformation B_2. Wir betrachten dabei nur die drei in Phase 1 relevanten Vektoren $|1\rangle, |2\rangle$ und $|3\rangle$. Beim Lesen der rechten Endmarkierung $\vdash$ werden die Amplituden von $|1\rangle, |2\rangle$ und $|3\rangle$ durch die Transformation $M_\vdash$ gewissermaßen zu den Vektoren $|4\rangle, |5\rangle$ und $|6\rangle$ „verschoben“. Mit dem erneuten Lesen der linken Endmarkierung beginnt für den Automaten die zweite Phase. Wir betrachten wieder nur die Amplituden der relevanten Vektoren. Das sind in der zweiten Phase $|4\rangle, |5\rangle$ und $|6\rangle$. Für jedes gelesene Zeichen a wird B_1^{-1} und für jedes Zeichen b die Transformation B_2^{-1} angewandt. Der Startzustand ist mit dem Vektor $|1\rangle$ assoziiert. Wir wissen aus Lemma 1.2-5 und Bemerkung 1.2-6 (Seite 16), dass für $|1\rangle = (1, 0, 0)^T$ und $|\psi\rangle = Y_1^{-1} \cdots Y_n^{-1} X_n \cdots X_1 |1\rangle$ mit $|\psi\rangle = \alpha\langle 1|\psi\rangle + \beta\langle 2|\psi\rangle + \gamma\langle 3|\psi\rangle$ gilt

1. $|\alpha|^2 = 1 \Leftrightarrow X_j = Y_j, j \in \{1, \ldots, n\}$,
2. $|\alpha|^2 < 1 - \frac{1}{25^n}$ sonst.

Sei $|\psi'\rangle = \sum_{i=1}^{8} \alpha_i |i\rangle$ die nach dem Lesen des Zeichens w_n in der zweiten Phase erreichte Überlagerung. Es ist dann $|\alpha_4|^2 = 1 \Leftrightarrow w \in L_{pal}$ und $|\alpha_5|^2 + |\alpha_6|^2 > \frac{1}{25^n} \Leftrightarrow w \notin L_{pal}$.

Das nächste gelesene Zeichen ist die rechte Endmarkierung $\vdash$. Es wird also die Transformation $M_{\vdash}$ angewandt. Die folgende Überlagerung ist $|\psi''\rangle = \alpha_4|1\rangle + \alpha_5|7\rangle + \alpha_6|8\rangle$. Genau für $w' \in L_{pal}$ ist die Wahrscheinlichkeit des Automaten zu verwerfen wegen $|\alpha_5|^2 + |\alpha_6|^2 = 0$ gleich Null. Für $w \notin L_{pal}$ verwirft der Automat an dieser Stelle mit Wahrscheinlichkeit größer als $\frac{1}{25^n}$. Die erwartete Laufzeit des Automaten ist also kleiner als 25^n. Sollte in einem Lesezyklus kein verwerfender Zustand gemessen werden, beginnt der Automat die Berechnung im Zustand q_0 in Phase 1 erneut mit dem Kopf über der linken Endmarkierung $\dashv$. □

Bemerkung 4.1-10
Der in Satz 4.1-9 beschriebene Automat macht keinen Gebrauch von den verallgemeinerten Messungen. Alle Überführungsmatrizen sind unitär.

Der in Satz 4.1-9 vorgestellte Automat hat für $w \notin L_{pal}$ eine erwartete exponentielle Laufzeit. Es gibt aber auch einen *1-QXFA^{vm}* für eine nicht-reguläre Sprache, der mit demselben Akzeptanzmodus Wörter, die nicht zu der Sprache gehören in, erwarteter polynomieller Laufzeit verwirft. Die Sprache $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ kann von einem *1-QXFA^{vm}* erkannt werden, wenn ein Nicht-Halten des Automaten als Akzeptieren interpretiert wird.

Lemma 4.1-11
Sei $w \in \Sigma^*$. Es gibt einen 1-QXFA^{vm}, der bei einer Eingabe $w \in \{a, b\}^*$ nie hält, falls $w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ ist und für $w \notin L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ mit Wahrscheinlichkeit Eins verwirft.

Beweis. Wir nennen den vorzustellenden Automaten $A = (Q, \Sigma, q_0, \delta, Q_{rej})$ und lassen in der Definition die leere Menge Q_{acc} bereits weg. Wir teilen die Berechnung von A gedanklich in zwei Phasen ein, wobei jede Phase genau einem Lesezyklus des Automaten entspricht (vgl. Definition 4.1-7). In der ersten Phase testet A, ob die Eingabe das Format a^*b^* hat. In der zweiten Phase wird sichergestellt, dass ebensoviele Zeichen a wie b vorkommen. Die beiden Phasen werden von dem Automaten alternierend durchlaufen. Weil $L_{ab} = \{a^*b^*\}$ eine reguläre Sprache ist, kann in der ersten Phase ein entsprechender *1-DFA* simuliert werden (vgl. Satz 4.1-3). Für die zweite Phase übernehmen wir eine Idee aus [AW02]. Das heißt, bei jedem gelesenen Zeichen a wird eine Transformation M_l auf der Menge der vorkommenden Zustände durchgeführt und bei jedem Zeichen b eine Transformation M_r mit $M_r M_l = I$. Es ist $|\psi\rangle = M_r^h M_l^h |\psi\rangle$ für alle $h \in \mathbb{N}$. Außerdem soll für alle $h, l \in \mathbb{N}$ mit $h \neq l$ gelten $|\psi\rangle \neq M_r^h M_l^l |\psi\rangle$. Als Transformation wird deswegen eine Drehung um ein irrationales Vielfaches von π verwendet. Wir wählen eine Drehung um den Winkel $\rho = \pi r$ mit

$$r = \sum_{i=1}^{\infty} 10^{-2^i} = 0,\overbrace{0101}^{1,2,3,4}\ldots\overbrace{010}^{7,8,9}\ldots\overbrace{0100}^{15,16,\ldots}\ldots.$$

Die i-te Eins in den Nachkommastellen von r steht an Position 2^i. Die $(i+1)$-te Eins folgt 2^i Positionen später. Es ist dann wie im Kapitel 1.2.1 beschrieben

$$M_l = \begin{bmatrix} \cos(\rho) & \sin(-\rho) \\ \sin(\rho) & \cos(\rho) \end{bmatrix}; \quad M_r = \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ \sin(-\rho) & \cos(\rho) \end{bmatrix}.$$

Die Zustände der Maschine sind $Q = \{q_0, q_1, q_2, q_3, q_{rej}\}$. Der Startzustand von A ist q_0. Der einzige verwerfende Zustand ist $q_{rej} \in Q_{rej}$, einen akzeptierenden Zustand gibt es nicht. Die Menge der nicht-haltenden Zustände ist $Q_{non} = \bigcup_{i=0}^{2} Q_i$ mit $Q_0 = \{q_0\}$, $Q_1 = \{q_1\}$ und $Q_2 = \{q_2, q_3\}$. Wir assoziieren die Zustände $q_0, q_1, q_2, q_3, q_{rej}$ mit der ON-Basis $|1\rangle, \ldots, |5\rangle$. Die Überführungsfunktion wird durch die folgenden Matrizen definiert.

$$M_a = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos(\rho) & \sin(-\rho) & 0 \\ 0 & 0 & \sin(\rho) & \cos(\rho) & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$M_b = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & \cos(\rho) & \sin(\rho) & 0 \\ 0 & 0 & \sin(-\rho) & \cos(\rho) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{\vdash} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{\dashv} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Wir analysieren das Verhalten des Automaten bei einer Eingabe $w \in \Sigma^*$.

Fall 1: $w \in L_{eq}$.
Der Automat A startet im Zustand $|1\rangle$ und liest das Zeichen $\dashv$. Entsprechend bleibt er in der Überlagerung $|1\rangle$. Der Kopf wandert ein Zeichen nach rechts. Der Automat A bleibt solange im Zustand $|1\rangle$, bis eines der Zeichen b oder $\vdash$ gelesen wird. Für $\vdash$ erreicht der Automat den Zustand $|3\rangle$ und beginnt danach die zweite Phase mit einem erneuten Lesen der Eingabe. Bei dem Zeichen b wechselt der Automat A nach $|2\rangle$ und bleibt dort, solange weitere Zeichen b folgen. Bei dem Lesen von $\vdash$ wechselt der Automat schließlich nach $|3\rangle$ und beginnt die zweite Phase. Die Eingabe ist dann einmal bearbeitet und wir wissen nun,

dass sie das Format a^*b^* hat. In der zweiten Phase wird für jedes Zeichen a die Drehung M_l und für jedes Zeichen b die Drehung M_r durchgeführt. Wegen $w \in L_{eq}$ findet die Maschine in der zweiten Phase gleich viele Buchstaben a wie b und wechselt nach nochmaligem Lesen des Wortes in den Zustand $|\psi_3\rangle = M_r^n M_l^n |3\rangle = I^n|3\rangle = |3\rangle$. Das Zeichen $\vdash$ ändert die Überlagerung nach $|1\rangle$ und die Berechnung beginnt wieder von vorne. Es gibt keine Möglichkeit, Eingaben w mit $w \in L_{eq}$ zu verwerfen.

Fall 2: $w \notin L_{eq}$.
Dann ist $w' = a^*b^+a\{a,b\}^*$ oder $w'' = a^hb^l,\ h \neq l$. Eine Eingabe $w' \notin L_{ab} = \{a^*b^*\}$ wird schon in der ersten Phase, in der ein *1-DFA* simuliert wird, mit Wahrscheinlichkeit Eins verworfen. Für $w'' = a^hb^l, h \neq l$, verläuft die Berechnung in der ersten Phase analog zu $w \in L_{eq}$. Zu Beginn der zweiten Phase befindet sich der Automat in der Überlagerung $|3\rangle$ mit dem Kopf über dem Zeichen $\dashv$. Danach wird w'' noch einmal gelesen und dabei für jedes Zeichen a eine Drehung um ρ bzw. für b um $-\rho$ durchgeführt. Anschließend erreicht A die Überlagerung $\alpha|3\rangle + \beta|4\rangle$. Wegen $l \neq h$ und weil ρ ein irrationales Vielfaches von π ist, gilt $\beta \neq 0$. Der Automat liest nun die Zeichen $\vdash$ und $\dashv$ und befindet sich danach in der Überlagerung $\alpha|1\rangle + \beta|5\rangle$. Der Vektor $|5\rangle$ ist mit dem verwerfenden Zustand q_{rej} assoziiert. Der Automat hat also eine positive Wahrscheinlichkeit zu verwerfen. Sollte nicht q_{rej} gemessen werden, beginnt die Berechnung im Zustand $|1\rangle$ wieder von vorne. □

Wir werden die folgende Feststellung in Kapitel 4.2 benötigen.

Bemerkung 4.1-12
Für die Überführungsfunktion des Automaten aus Lemma 4.1-11 ist $\delta(q_2, \vdash, q_0) = 1$.

Wir interessieren uns nun für die Laufzeit des Automaten. Dazu beschäftigen wir uns zunächst mit der Drehung, die der Automat in der zweiten Phase auf der Überlagerung $\alpha|3\rangle + \beta|4\rangle$ durchführt.

Lemma 4.1-13
Sei $r = \sum_{i=1}^{\infty} 10^{-2^i}$. *Für ein beliebiges* $n \in \mathbb{N}$ *hat die erste von Null verschiedene Nachkommastelle* z_v *in* nr *einen Index* $v < 2^{\lfloor \log_2(1+\log_{10} n)\rfloor+1}$.

Beweis. Sei $10^{k-1} \leq n < 10^k,\ k \in \mathbb{N}$. Damit ist n eine k-stellige Dezimalzahl und $k \leq 1 + \log_{10} n$. Um eine bessere Vorstellung zu bekommen, schreiben wir die einzelnen Summanden der Summe $\sum_{i=1}^{\infty} n10^{-2^i} = nr$ untereinander. Dabei lassen wir die Zeilen der Stellen von r, die Null sind, weg. Wie in Abbildung 4.1 angedeutet, teilen wir die Dezimalzahl $n = n_1 \dots n_k$ gedanklich in zwei Abschnitte $a_1 \dots a_o = n_1 \dots n_o$ und $b_1 \dots b_p = n_{o+1} \dots n_k$. Diese Aufteilung richtet sich nach der Lage von $n_1 \dots n_k$ in den Zeilen $\lfloor \log_2 k \rfloor$ und $\lfloor \log_2 k \rfloor + 1$. In Zeile $\lfloor \log_2 k \rfloor + 1$ beginnt $b_1 \dots b_p$ an Position $2^{\lfloor \log_2 k \rfloor} + 1$. In Zeile $\lfloor \log_2 k \rfloor$ sind $b_1 \dots b_p$ genau die Nachkommastellen des Summanden. In der Abbildung ist $m_1 = 2^{\lfloor \log_2 k \rfloor},\ m_2 = 2^{\lfloor \log_2 k \rfloor + 1} - k + 1$ und $m_3 = 2^{\lfloor \log_2 k \rfloor + 1}$.

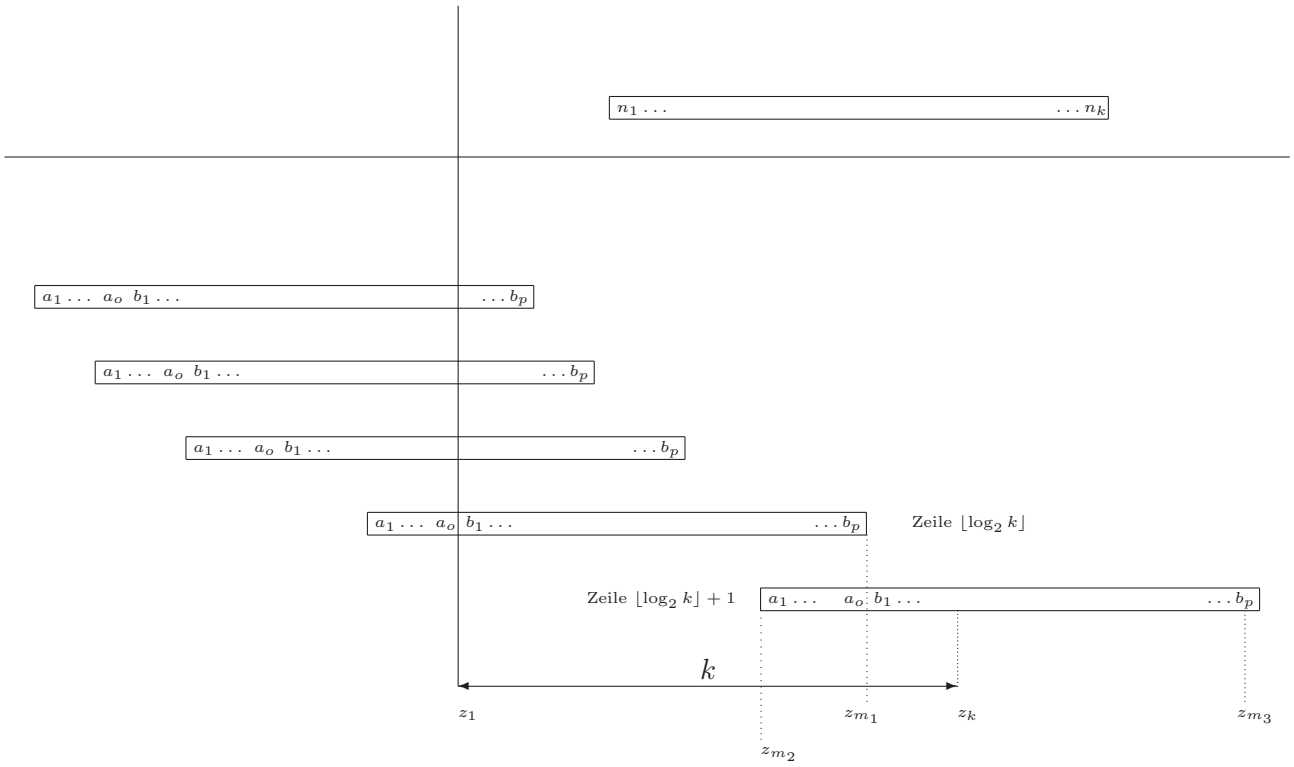

Abbildung 4.1: Schematische Darstellung der Addition $\sum_{i=1}^{\infty} n10^{-2^i}$. Es ist $m_1 = 2^{\lfloor \log_2 k \rfloor}$ sowie $m_2 = 2^{\lfloor \log_2 k \rfloor + 1} - k + 1$ und $m_3 = 2^{\lfloor \log_2 k \rfloor + 1}$.

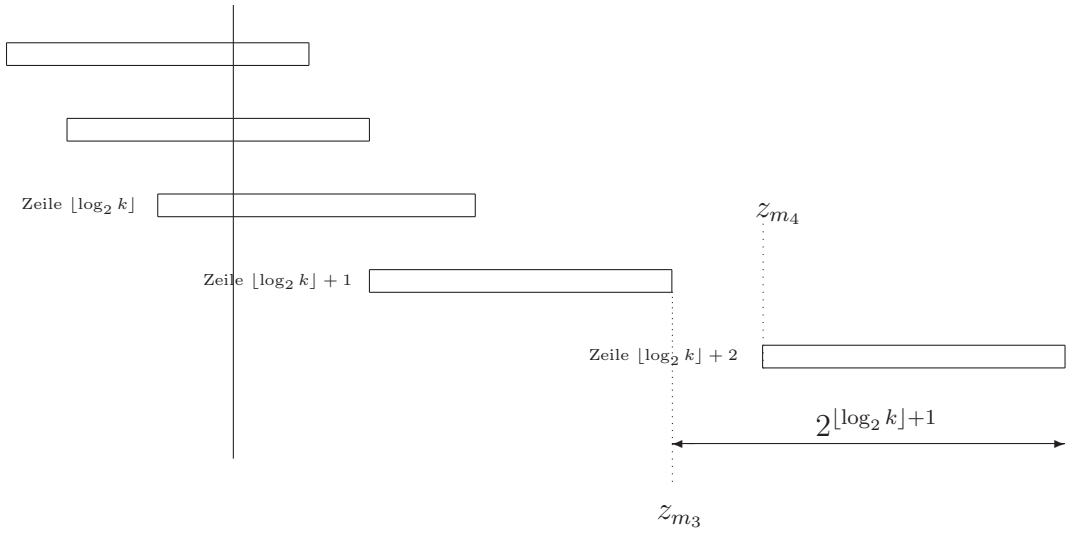

Abbildung 4.2: Die erste Null in den Nachkommastellen der Dezimalschreibweise von nr liegt spätestens an Position $z_{2^{\lfloor \log_2 k \rfloor + 1} + 1}$. Es ist $m_3 = 2^{\lfloor \log_2 k \rfloor + 1}$ und $m_4 = 2^{\lfloor \log_2 k \rfloor + 2} - k + 1$.

Im Widerspruch zur Behauptung nehmen wir an, dass eine Zahl n mit $10^{k-1} \leq n < 10^k$ existiert, sodass die ersten $2^{\lfloor \log_2 k \rfloor + 1}$ Nachkommastellen $z_1 \dots z_{2^{\lfloor \log_2 k \rfloor + 1}}$ in nr alle den Wert Null haben.

Wegen $n < 10^k$ wird zwischen den Zeilen $\lfloor \log_2 k \rfloor + 1$ und $\lfloor \log_2 k \rfloor + 2$ kein Übertrag erzeugt (vgl. Abbildung 4.2). Damit $z_1 = \dots = z_{2^{\lfloor \log_2 k \rfloor + 1}} = 0$ gilt, muss also $b_1 = \dots = b_p = 0$ gelten, da $b_1 \dots b_p$ in Zeile $\lfloor \log_2 k \rfloor + 1$ an den Positionen $z_{2^{\lfloor \log_2 k \rfloor + 1}} \dots z_{2^{\lfloor \log_2 k \rfloor + 1}}$ liegt. Wegen $10^{k-1} \leq n$ ist $a_1 = n_1 \neq 0$. Weil die Nachkommastellen in den Zeilen $1, \dots, \lfloor \log_2 k \rfloor$ ausschließlich aus dem hinteren Teil $b_1 \dots b_p$ der Dezimalzahl n stammen, tragen sie jeweils mit dem Wert Null zu der Addition in den Nachkommastellen bei. Deswegen muss die Nachkommastelle $z_{m_2} = z_{2^{\lfloor \log_2 k \rfloor + 1} - k + 1}$ denselben Wert wie a_1 haben. Das steht im Widerspruch zur Annahme. □

Lemma 4.1-14
Sei $r = \sum_{i=1}^{\infty} 10^{-2^i}$. Für beliebiges $n \in \mathbb{N}$ hat der Index v der ersten Nachkommastelle z_v in nr mit $z_v = 0$ einen Wert $v \leq \lceil 3 + 2 \log_{10} n \rceil$.

Beweis. Sei $n \in \mathbb{N}$ mit $10^{k-1} \leq n < 10^k, k \in \mathbb{N}$. Dann ist n eine k-stellige Dezimalzahl. Wie in Abbildung 4.2 zu sehen ist, steht die letzte Ziffer n_k von n in Zeile $\lfloor \log_2 k \rfloor + 1$ an Position $z_{m_3} = z_{2^{\lfloor \log_2 k \rfloor + 1}}$, die erste Ziffer n_1 in Zeile $\lfloor \log_2 k \rfloor + 2$ jedoch erst an Position

$z_{m_4} = z_{2^{\lfloor \log_2 k \rfloor + 2} - k + 1}$. Es ist

$$\begin{aligned} & 2^{\lfloor \log_2 k \rfloor + 2} - k + 1 - 2^{\lfloor \log_2 k \rfloor + 1} \\ = \; & 2^{\lfloor \log_2 k \rfloor + 1} - 2^{\log_2 k} + 1 \\ > \; & 1 \end{aligned}$$

die Anzahl der Nachkommastellen zwischen z_{m_3} und z_{m_4}. Alle diese Nachkommastellen haben in nr den Wert Null. Die erste Null in den Nachkommastellen liegt also spätestens an Position $z_{m_3+1} = z_{2^{\lfloor \log_2 k \rfloor + 1} + 1}$. Es ist $2^{\lfloor \log_2 k \rfloor + 1} + 1 \leq 2^{1 + \log_2 k} + 1$ und wegen $k \leq 1 + \log_{10} n$ ist

$$2^{1+\log_2 k} + 1 \leq 3 + 2\log_{10} n \leq \lceil 3 + 2\log_{10} n \rceil.$$

□

Wir fügen nun die Ergebnisse aus den letzten beiden Lemmata zusammen. Die Schreibweise in dem folgenden Lemma mag auf den ersten Blick etwas verwundern. Wir benötigen das folgende Lemma jedoch nur als Zwischenergebnis und wählen schon an dieser Stelle die Schreibweise, die wir in Satz 4.1-16 verwenden, weil wir hoffen, dass die Diskussion in dem Satz dadurch besser verfolgt werden kann.

Lemma 4.1-15
Seien $n \in \mathbb{N}$, $r = \sum_{i=1}^{\infty} 10^{-2^i}$, $d = \lceil 3 + 2\log_{10} n \rceil$ *und* $\frac{\kappa}{\pi} = 10^{-(d+1)}$. *Dann ist* $\frac{\kappa}{\pi} \mod 1 < nr \mod 1 < 1 - (\frac{\kappa}{\pi} \mod 1)$.

Beweis. Wir werden zunächst die Ungleichung $\frac{\kappa}{\pi} \mod 1 < nr \mod 1$ zeigen und anschließend $nr \mod 1 < 1 - \left(\frac{\kappa}{\pi} \mod 1\right)$.

$\frac{\kappa}{\pi} \mod 1 < nr \mod 1$:
Uns interessieren nun die Nachkommastellen von nr. Sei z_1 die erste und allgemein z_i die i-te Nachkommastelle von nr. Wegen $\frac{\kappa}{\pi} = 10^{-(d+1)}$ reicht es zu zeigen, dass die erste von Null verschiedene Nachkommastelle in nr die Nachkommastelle z_v mit $v \leq d$ ist. Wir bekommen sogar ein minimal besseres Ergebnis aus Lemma 4.1-13.

Wir werden sehen, dass ein noch besseres Ergebnis für unsere Wahl von r nicht möglich ist. Wie in Abbildung 4.3 zu sehen ist, hat $n' = 990100000099$ bei der Multiplikation mit r erst an Position $z_{16-1} = z_{15}$ einen von Null verschiedenen Wert. Es ist $n' \in]10^{11}, 10^{12}[$ und $2^{\lfloor \log_2 12 \rfloor + 1} = 16$. Der Term $n'r$ hat also erst an der $(2^{\lfloor \log_2 k \rfloor + 1} - 1)$-ten Nachkommastelle einen von Null verschiedenen Wert, wenn k die Anzahl der Stellen in der Dezimalschreibweise von n' angibt. Jetzt muss noch $nr \mod 1 < 1 - 10^{-(d+1)} = 1 - \left(\frac{\kappa}{\pi} \mod 1\right) = 1 - \frac{\kappa}{\pi}$ gezeigt werden. Interessant ist also der Fall

$$nr \mod 1 = 0,\overbrace{99\ldots99}^{v-1}0\ldots.$$

Die Sequenz von großen Ziffern in den Nachkommastellen von nr *mod* 1 muss rechtzeitig von mindestens einer kleinen Ziffer unterbrochen werden. Auf jeden Fall passiert dies bei

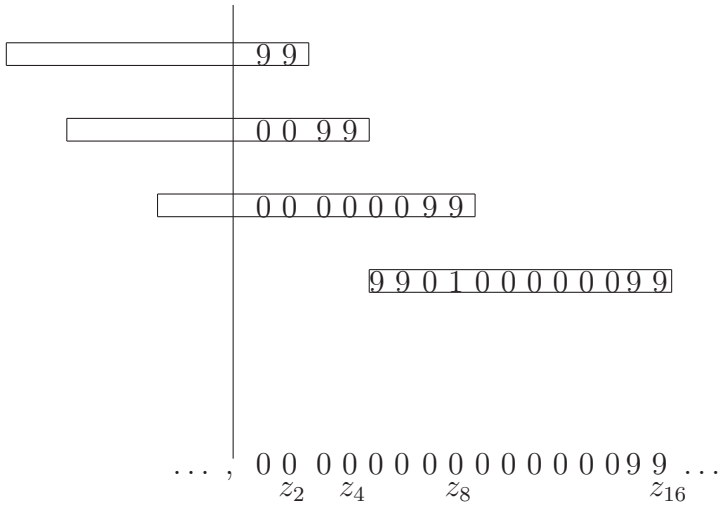

Abbildung 4.3: Beispiel für eine Zahl n' mit möglichst vielen führenden Nullen in den Nachkommastellen von rn' in Abhängigkeit von der Länge k der Dezimalschreibweise von n'. In dem Beispiel ist $n' = 990100000099$. Erst die $(2^{\lfloor \log_2 k \rfloor + 1} - 2)$-te Nachkommastelle ist von Null verschieden.

der ersten Ziffer 0. Wir erhalten aus Lemma 4.1-14, dass für die erste Nachkommastelle z_v mit $z_v = 0$ die Ungleichung $v < d$ gilt.

Mit diesen beiden Ergebnissen zusammen ist die Behauptung gezeigt. □

Satz 4.1-16
Sei $\varepsilon = \sin(\pi 10^{-\lceil 3+2\log_{10} n \rceil})$ *und* $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$. *Die erwartete Laufzeit des 1-QXFA*^{vm} *aus Lemma 4.1-11 bei einer Eingabe* $w \in (\Sigma^n \setminus L_{eq})$ *ist* $O(\frac{n}{\varepsilon^2}) = O(n^5)$. *Mit Wahrscheinlichkeit kleiner als* $\left(\frac{e}{4}\right)^{O(n^4)}$ *ist die Rechenzeit des Automaten dann nicht größer als* $\frac{2}{\varepsilon^2} \cdot O(n)$.

Beweis. Um die erwartete Rechenzeit anzugeben, bis der Automat eine Eingabe $w \notin L_{eq}$ verwirft, interessiert uns besonders der Fall $w = a^h b^l; \quad h \neq l$. Andernfalls können wir für $|w| = m$ von einer Laufzeit in der Größenordnung $O(m)$ ausgehen, da der simulierte *1-DFA* den Fehler nach einmaligem Lesen der Eingabe erkennt.

Für den Fall $w = a^h b^l,\ h \neq l$, möchten wir eine untere Schranke für die Wahrscheinlichkeit, in der zweiten Phase den Zustand q_{rej} zu messen, angeben. Die Maschine erreicht nach dem Ende der zweiten Phase eine Überlagerung $\alpha|1\rangle + \beta|5\rangle$ mit $\beta \neq 0$. Der Vektor $|5\rangle$ wird mit dem verwerfenden Zustand q_{rej} assoziiert. An keiner anderen Stelle in der Berechnung hat die Amplitude von $|5\rangle$ einen Wert $\beta \neq 0$. Damit hängt die erwartete Laufzeit nur von der Amplitude β nach dem Lesen von $\dashv$ ab. Für ein Wort w mit $|w| = n$ und $\varepsilon = \sin(\kappa)$

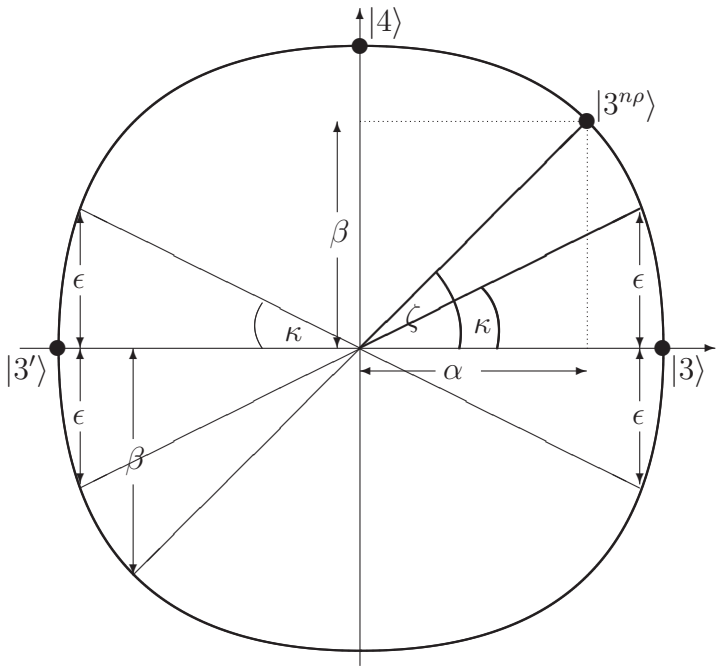

Abbildung 4.4: Drehungen der Überlagerung $|3\rangle$ um die Winkel κ und ζ.

für eine geeignet gewählte Konstante κ zeigen wir $\beta > \varepsilon$. Der Automat verwirft dann in der zweiten Phase mindestens mit Wahrscheinlichkeit $|\varepsilon|^2 = \varepsilon^2$.

Zunächst werden wir versuchen, eine anschauliche Vorstellung von der angewandten Rotation zu bekommen. Der Automat beginnt die Berechnung in der zweiten Phase im Zustand $|3\rangle$. Danach werden Drehungen um ρ bzw. $-\rho$ durchgeführt. Es soll gezeigt werden, dass mehr als n Drehungen in eine Richtung notwendig sind, damit $\beta < \varepsilon$ gilt. Wenn wir voraussetzen, dass alle Drehungen das gleiche Vorzeichen haben, betrachten wir den für uns ungünstigsten Fall. Weil sich Drehungen mit verschiedenem Vorzeichen gegenseitig aufheben, brauchen keine weiteren Fälle betrachtet werden. Nach einmaligem Lesen der Eingabe hat der *1-QXFA*^{vm} A den Ausgangszustand um $n\rho$ gedreht. Der aktuell erreichte Zustand wird bei diesen Drehungen eventuell mehrfach an dem Ausgangszustand $|3\rangle$ 'vorbeigedreht'. Der erreichte Zustand sei $|3^{n\rho}\rangle$ und der Winkel zwischen $|3\rangle$ und $|3^{n\rho}\rangle$ sei $\zeta = n\rho \mod 2\pi$ (vgl. Abbildung 4.4). Weil ρ ein irrationales Vielfaches von π ist, gilt $|3\rangle \neq |3^{n\rho}\rangle$.

Es ist $|3^{n\rho}\rangle = \alpha|3\rangle + \beta|4\rangle$ mit $\beta = \sin(\zeta)$. Der Punkt $|3^{n\rho}\rangle$ kann in jedem der vier von den Koordinatenachsen aufgespannten Quadranten liegen. In dem ersten Quadranten ist ζ der Winkel, um den $|3^{n\rho}\rangle$ von $|3\rangle$ abweicht. Der Winkel $\zeta = n\rho \ mod \ 2\pi$ soll dann größer als κ sein. Wir müssen deswegen $\beta = \sin(n\rho \mod 2\pi) > \sin(\kappa \mod 2\pi) = \varepsilon$ zeigen, falls $|3^{n\rho}\rangle$ im ersten Quadranten liegt. Auch wenn $|3^{n\rho}\rangle$ im vierten Quadranten liegt, darf der Winkel zu $|3\rangle$ nicht kleiner als κ sein. Deswegen können diese beiden Fälle zusammengefasst werden zu $|\sin(n\rho \mod 2\pi)| > \sin(\kappa \mod 2\pi)$. Alle Punkte, die in den dritten oder vier-

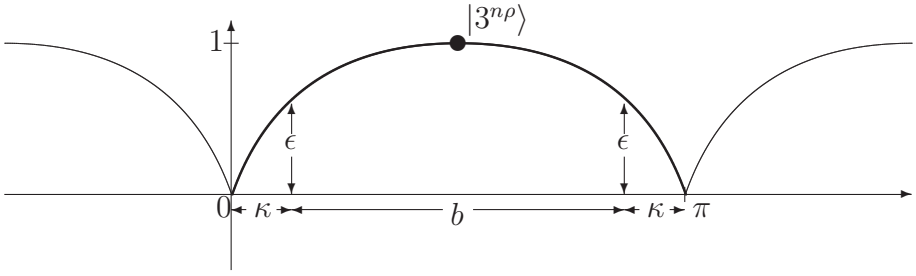

Abbildung 4.5: Der Bereich b, in dem $|3^{n\rho}\rangle$ liegen darf.

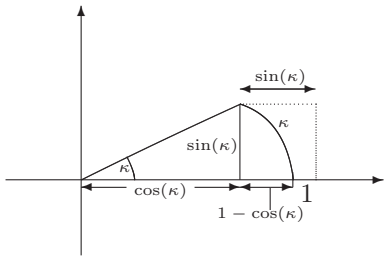

Abbildung 4.6: Es ist $\sin(\kappa) > \frac{\kappa}{2}$.

ten Quadranten fallen, werden dadurch auf den ersten bzw. zweiten Quadranten gespiegelt. Sollte $|3^{n\rho}\rangle$ im zweiten Quadranten liegen, muss der Winkel zu $|3'\rangle = -|3\rangle$ größer als κ sein. Wie in Abbildung 4.5 veranschaulicht, muss $|3^{n\rho}\rangle$ in dem Bereich b liegen, um ausreichend weit von $|3\rangle$ entfernt zu sein.

Es muss also $\kappa \mod \pi < n\rho \mod \pi < \pi - (\kappa \mod \pi)$ bzw. wegen $\rho = \pi r$ dann $\frac{\kappa}{\pi} \mod 1 < nr \mod 1 < 1 - (\frac{\kappa}{\pi} \mod 1)$ gelten. Für $d = \lceil 3 + 2\log_{10} n \rceil$ definieren wir κ durch

$$\frac{\kappa}{\pi} = 0,\overbrace{000\ldots000}^{d}1 = 10^{-(d+1)}.$$

Wir erhalten $\frac{\kappa}{\pi} \mod 1 < nr \mod 1 < 1 - (\frac{\kappa}{\pi} \mod 1)$ mit Lemma 4.1-15. Insgesamt gilt also $\beta > \varepsilon$. Der Automat verwirft in einem Lesezyklus mit Wahrscheinlichkeit größer als $|\varepsilon|^2 = \varepsilon^2$. Der Erwartungswert für die Anzahl der Lesezyklen des Automaten bis der verwerfende Zustand erreicht wird, ist also ε^{-2} Ein Lesezyklus für $w \in \Sigma^n$ verursacht Kosten in Höhe von $O(n)$. Wir erhalten damit die erwartete Rechenzeit $O(n\varepsilon^{-2})$.

Nun ist $\varepsilon = \sin(\kappa)$ eine recht unhandliche Größe. Weil ε klein gewählt werden soll, können wir κ auf nicht zu große Werte einschränken. Wir werden zeigen, dass für $0 < \kappa \leq \frac{\pi}{2}$ eine Konstante c gefunden werden kann, sodass $\sin(\kappa) \geq c\kappa$ gilt. Wegen $\varepsilon = O(n^{-2})$ erhalten wir die erwartete Laufzeit $O(n\varepsilon^{-2}) = O(n^5)$. In der folgenden Betrachtung gehen wir davon aus, dass $\kappa \in]0, \frac{\pi}{2}]$ ist. Abbildung 4.6 verdeutlicht die Diskussion. Es ist $\sin(\kappa) = \sqrt{1 - \cos^2(\kappa)}$ und wegen $0 < \kappa \leq \frac{\pi}{2}$ auch $\sin(\kappa) = \sqrt{1 - \cos^2(\kappa)} \geq 1 - \cos(\kappa)$. In Abbildung 4.6 haben die beiden gestrichelten Linien oberhalb des Kreisabschnitts der Länge κ jeweils die Länge $\sin(\kappa)$ und liegen deswegen stets oberhalb des Kreisbogens. Es ist also $2\sin(\kappa) \geq \kappa$ und mit $c = \frac{1}{2}$ gilt $\sin(\kappa) \geq c\kappa$. Der Erwartungswert für die Häufig-

keit, mit der der Automat die beiden Phasen durchläuft, bis die Eingabe verworfen wird, ist bei einer Eingabe $w = a^k b^l, k \neq l$, also $\frac{1}{\varepsilon^2}$. Mit Chernoff-Schranken (vgl. Satz 1.1-2) erhalten wir, dass der Automat die beiden Phasen mit einer Wahrscheinlichkeit kleiner als $\left(\frac{e}{4}\right)^{\varepsilon^{-2}} = \left(\frac{e}{4}\right)^{O(n^4)}$ mehr als $\frac{2}{\varepsilon^2}$-mal durchläuft. □

Durch die Hinzunahme eines speziellen Akzeptanzmodus kann der *1-QFA*^{vm} also nichtreguläre Sprachen erkennen. Wir werden in Kapitel 4.2 sehen, dass die verallgemeinerten Messungen dafür nicht notwendig sind. Ein zyklisches Band und der spezielle Akzeptanzmodus sind bereits ausreichend.

4.2 Zyklisches Eingabeband

Wir werden nun einen *1-QFA* mit zyklischem Band vorstellen (*1-QXFA*).

Definition 4.2-1
Ein 1-QXFA $A = (Q, \Sigma, q_0, \delta, Q_{acc}, Q_{rej})$ ist eine Verallgemeinerung eines 1-QFA (vgl. Definition 3.2-1). Ein 1-QXFA ist ein 1-QFA, bei dem die Beschränkung, dass der Lesekopf nicht über die rechte Endmarkierung $\vdash$ hinaus bewegt werden darf, aufgehoben ist. Der Automat verfügt über ein zyklisches Eingabeband. Wenn A nach dem Lesen von $\vdash$ nicht in einen haltenden Zustand fällt, bewegt sich der Lesekopf ein Feld nach rechts und liest dann die linke Endmarkierung $\dashv$.

Bemerkung 4.2-2
Da ein 1-QXFA eine Verallgemeinerung eines 1-QFA ist, können alle von einem 1-QFA erkannten Sprachen auch von einem 1-QXFA erkannt werden. Die Laufzeit des Automaten ist für die betrachtete Sprache nicht größer als für den 1-QFA. Der 1-QXFA hat maximal so viele Zustände, wie der 1-QFA.

Der Automatentyp des *1-QXFA*s wurde in [AF98] vorgeschlagen, weil der Beweis 3.2-4 (siehe Seite 39) der Simulation eines *1-QFA* durch einen *1-DFA* nicht einfach auf das Modell des *1-QXFA* ausgeweitet werden kann. Das liegt an dem zyklischen Band des *1-QXFA*.

Satz 4.2-3
Es gibt einen 1-QXFA, der bei einer Eingabe $w \in \{a, b\}^*$ nie einen haltenden Zustand erreicht, wenn $w \in L_{pal} = \{w | w \in \{a, b\}^* \text{ und } w = w^{-1}\}$ ist und der andernfalls mit Wahrscheinlichkeit Eins verwirft. Für $w \notin L_{pal}$ ist die erwartete Rechenzeit des Automaten $O(25^{|w|})$.

Beweis. Die Behauptung folgt aus Bemerkung 4.1-10. □

Lemma 4.2-4
Die Menge der von 1-QXFAs erkannten Sprachen ist gegen die Operation Durchschnitt abgeschlossen, wenn für beide Automaten

1. *ein Zustand q aus der Zustandsmenge mit $\delta(q, \vdash, q_0) = 1$ existiert,*
2. *ein Halten nicht zum Akzeptieren notwendig ist und*
3. *Wörter, die nicht aus der Sprache sind, mit Wahrscheinlichkeit Eins verworfen werden.*

Für zwei Sprachen L_1 und L_2 über demselben Alphabet Σ mit eben solchen Automaten A_1 bzw. A_2 gibt es für die Schnittsprache $L_{12} = L_1 \cap L_2$ einen 1-QXFA A_{12}, der Wörter $w \notin L_{12}$ auf jeden Fall verwirft und ansonsten nicht hält. Seien Q_1 und Q_2 die Zustandsmengen von A_1 bzw. A_2. Dann enthält die Zustandsmenge Q_{12} von A_{12} genau $|Q_1|+|Q_2|$ Zustände. Seien $\mathcal{T}_1$ und $\mathcal{T}_2$ die erwarteten Rechenzeiten von A_1 bzw. A_2. Dann ist $\mathcal{T}_{12} = O(\max(\mathcal{T}_1, \mathcal{T}_2))$ die erwartete Rechenzeit von A_{12}.

Beweis. Seien L_1 und L_2 Sprachen über demselben Alphabet Σ und $A_1 = \{Q_1, \Sigma, q_0^1, \delta^1, Q_{rej}^1\}$ bzw. $A_2 = \{Q_2, \Sigma, q_0^2, \delta^2, Q_{rej}^2\}$ zwei *1-QXFA*s für L_1 bzw. L_2. Die Automaten A_1 und A_2 besitzen keine akzeptierenden Zustände und brechen die Berechnung bei einem Eingabewort, das aus der Sprache ist, nie ab. Andernfalls verwerfen A_1 und A_2 mit Wahrscheinlichkeit Eins. Wir entwerfen aus A_1 und A_2 einen Automaten A_{12}, der Wörter $w \notin L_{12} = L_1 \cap L_2$ mit Wahrscheinlichkeit 1 verwirft und für Eingaben $w \in L_{12}$ nie hält. Der Automat A_{12} liest die Eingabe wiederholt und simuliert dabei abwechselnd die Automaten A_1 bzw. A_2. Weil beide Automaten für ein Wort w, das aus der Sprache ist, nie halten, erreichen sie in einem Lesezyklus die rechte Endmarkierung $\vdash$ mit Wahrscheinlichkeit Eins. An dieser Stelle gibt es nach Voraussetzung einen Zustand q^1 bzw. q^2 aus der Zustandsmenge des entsprechenden Automaten mit $\delta^1(q^1, \vdash, q_0^1) = 1$ bzw. $\delta^2(q^2, \vdash, q_0^2) = 1$. Wir werden diesen Teil der Überführungsfunktion neu definieren, um den Wechsel zwischen der Simulation der beiden Automaten zu realisieren. Der Automat wird definiert durch $A_{12} = \{Q_{12}, \Sigma, q_0^{12}, \delta^{12}, Q_{rej}^{12}\}$ mit $Q_{12} = Q_1 \cup Q_2, q_0^{12} = q_0^1, Q_{rej}^{12} = Q_{rej}^1 \cup Q_{rej}^2$ und

$$\delta^{12}(q, \sigma, q') = \begin{cases} \delta^1(q, \sigma, q'), & \text{falls } q, q' \text{ in } Q^1 \text{ und } q' \notin q_0^1, \\ \delta^2(q, \sigma, q'), & \text{falls } q, q' \text{ in } Q^2 \text{ und } q' \notin q_0^2, \\ 1, & \text{falls } q' = q_0^2 \text{ und } \sigma = \vdash \text{ und } \delta^1(q, \sigma, q_0^1) = 1, \\ 1, & \text{falls } q' = q_0^1 \text{ und } \sigma = \vdash \text{ und } \delta^2(q, \sigma, q_0^2) = 1, \\ 0, & \text{sonst.} \end{cases}$$

Weil die Überführungsmatrizen der Automaten A_1 und A_2 unitär sind, bleiben auch die Überführungsmatrizen von A_{12} unitär, wie in Abbildung 4.7 zu sehen ist. Während bei allen Zeichen $\sigma \neq \vdash$ die Überführungsmatrizen M_σ^1 und M_σ^2 zu einer Überführungsmatrix M_σ^{12} zusammengefasst werden können, findet im Fall $\sigma = \vdash$ zusätzlich ein Zeilentausch statt. Die Matrix bleibt dabei unitär.

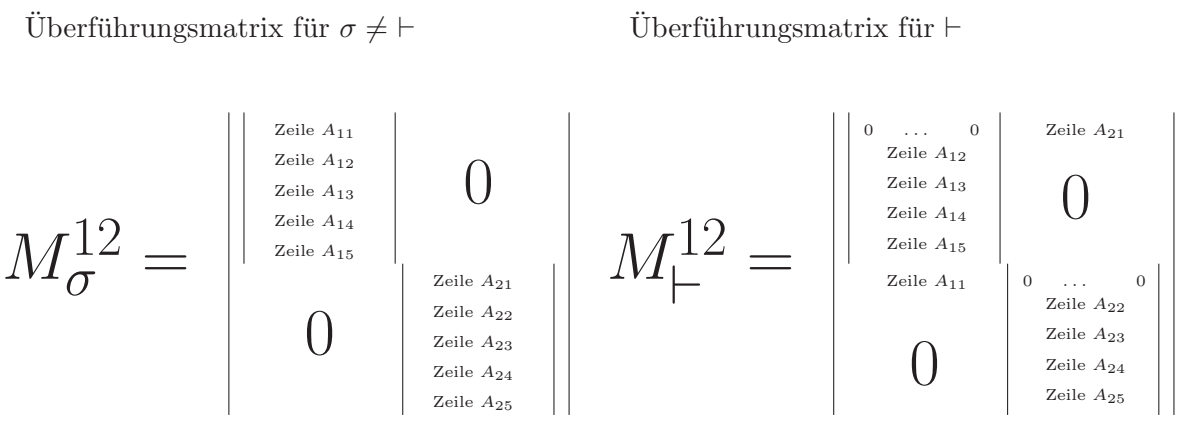

Abbildung 4.7: Schematische Darstellung der Überführungsmatrizen für $\sigma \neq \vdash$ und $\sigma = \vdash$ sowie $|Q^{12}| = 10$. Links oben in M_σ^{12} ist die Überführungsmatrix von V_σ^1, rechts unten die Matrix für V_σ^2. Für $\vdash$ müssen die ersten Zeilen von $V_\vdash^1$ und $V_\vdash^2$ vertauscht werden.

Bei einer Eingabe $w \in L_{12}$ startet A_{12} im Zustand q_0^1 und liest das Zeichen $\dashv$. Solange noch nicht das Zeichen $\vdash$ gelesen wird, wird der Automat A_1 simuliert. Wegen $w \in L_{12} \Rightarrow w \in L_1$ verwirft A_{12} nicht und liest schließlich das Zeichen $\vdash$. Der Folgezustand ist q_0^2, der Startzustand von A_2. Dies gilt nach Voraussetzung, weil es für jeden der beiden Automaten A_1 und A_2 einen speziellen Zustand q aus der Zustandsmenge gibt, aus dem der Automat beim Lesen von $\vdash$ mit Wahrscheinlichkeit Eins in den Startzustand wechselt. Weil alle Überführungsmatrizen unitär sind, ist die Wahrscheinlichkeit, beim Lesen von $\vdash$ zwischen zwei beliebigen anderen Zuständen zu wechseln, gleich Null. Beide Automaten erreichen also am Ende der Eingabe den speziellen Zustand q, wenn sie nicht vorher verworfen haben. Wir kehren nun wieder zurück zu der Berechnung des Automaten A_{12}. Beginnend mit dem Zeichen $\dashv$ wird nun der Automat A_2 simuliert. Wieder wird wegen $w \in L_{12} \Rightarrow w \in L_2$ kein verwerfender Zustand erreicht. Nach nochmaligem Lesen der Eingabe wechselt A_{12} beim Lesen von $\vdash$ wieder nach $q_0^1 = q_0^{12}$ und beginnt erneut mit der Simulation von A_1 von vorne.

Für $w \neq L_{12}$ ist zumindest eine der Aussagen $w \notin L_1$ oder $w \notin L_2$ richtig. Wie oben bereits gesehen, werden abwechselnd die Automaten A_1 und A_2 simuliert. Mindestens einer der Automaten A_1 bzw. A_2 hat eine positive Wahrscheinlichkeit, w zu verwerfen. Weil die Berechnung andernfalls kontinuierlich fortgesetzt wird, wird $w \in L_{12}$ mit Wahrscheinlichkeit Eins verworfen. Sei $\mathcal{P}_1$ die Wahrscheinlichkeit, dass A_1 eine Eingabe $w \notin L_{12}$ bei einmaligem Lesen verwirft und $\mathcal{P}_2$ die entsprechende Wahrscheinlichkeit für A_2. Nachdem A_1 und A_2 je einmal simuliert wurden, verwirft A_{12} eine Eingabe $w \notin L_{12}$ dann mindestens mit Wahrscheinlichkeit $\mathcal{P}_{12} = \min\{\mathcal{P}_1, \mathcal{P}_2\}$. Die erwartete Rechenzeit von A_{12} ist entsprechend höchstens $\mathcal{T}_{12} = 2\max\{\mathcal{T}_1, \mathcal{T}_2\}$. □

Wir suchen nun zwei *1-QXFA*s für Sprachen L_1 und L_2, aus denen wir mit Hilfe von Lemma 4.2-4 einen *1-QXFA* für die Sprache $L_{12} = L_1 \cap L_2$ entwerfen können. Die Spra-

che L_{12} wird eine nicht reguläre Sprache sein.

Lemma 4.2-5
Es gibt einen 1-QXFA, der bei einer Eingabe $w \in \Sigma^*$ genau dann nie in einen haltenden Zustand gerät, wenn $w \in L_{ab} = \{a^*b^*\}$ ist und der andernfalls mit Wahrscheinlichkeit Eins verwirft.

Beweis. Die Zustände des *1-QXFA* A sind $Q = \{q_0, q_1, q_2, q_3, q_{rej}\}$ mit $Q_{non} = \{q_0, q_1, q_2, q_3\}$ und $Q_{rej} = \{q_{rej}\}$. Der Startzustand von A ist q_0. Mit den Zuständen $q_0, q_1, q_2, q_3, q_{rej}$ assoziieren wir die ON-Basis $|1\rangle, \ldots, |5\rangle$. Die Überführungsfunktion wird durch die folgenden Überführungsmatrizen definiert. Sei $\rho = \pi r$ mit $r = \sum_{i=1}^{\infty} 10^{-2^i}$.

$$M_a = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$M_b = \begin{bmatrix} \cos(\rho) & \sin(\rho) & 0 & 0 & 0 \\ \sin(\rho) & \cos(\rho) & 0 & 0 & 0 \\ 0 & 0 & \cos(\rho) & \sin(\rho) & 0 \\ 0 & 0 & \sin(-\rho) & \cos(\rho) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{\vdash} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$M_{\dashv} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bei einer Eingabe $w \in L_{ab} = \{a^*b^*\}$ wird der Automat zunächst im Zustand q_0 in der Überlagerung $|1\rangle$ starten und das Zeichen $\dashv$ lesen. Solange nur die Zeichen $\dashv$ oder a vorkommen, bleibt A in der Überlagerung $|1\rangle$. Bei jedem Zeichen b wird eine Transformation

$$B = \begin{bmatrix} \cos(\rho) & \sin(-\rho) \\ \sin(\rho) & \cos(\rho) \end{bmatrix}$$

auf der erreichten Überlagerung durchgeführt. Wir sehen, dass B eine Drehung um ein irrationales Vielfaches von π ist (vgl. Kapitel 1.2.1). Wegen $w \in L_{ab}$ liest der Automat danach die rechte Endmarkierung $\vdash$. Die erreichte Überlagerung $v|1\rangle + u|2\rangle$ wird

zu $|\psi\rangle = v|3\rangle + u|4\rangle$ verändert und die Eingabe erneut gelesen. Solange $\dashv$ oder a gelesen wird, bleibt die Überlagerung unverändert. Bei jedem Zeichen b wird die Transformation B^{-1} durchgeführt. Es ist

$$B^{-1} = \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ \sin(-\rho) & \cos(\rho) \end{bmatrix}.$$

Wegen $B^{-1}B = I$ ist die Überlagerung, in der sich A befindet, nachdem alle Zeichen b gelesen wurden, $|\psi_2\rangle = |3\rangle$. Durch das Lesen von $\vdash$ wird die Überlagerung wieder zu $|1\rangle$ geändert und die Berechnung beginnt erneut mit dem Lesekopf über dem Zeichen $\dashv$. Es gibt keine Möglichkeit, Eingaben $w \in L_{ab}$ zu verwerfen.

Ein Wort $w \notin L_{ab}$ enthält mindestens einmal die Sequenz ba. Bis zu dem ersten Auftreten dieser Sequenz verläuft die Berechnung analog zu dem oben beschriebenen Fall. Weil ρ ein irrationales Vielfaches von π ist, befindet sich A dann in der Überlagerung $|\psi'\rangle = v'|1\rangle + u'|2\rangle$ mit $u' \neq 0$. Das Lesen von a verändert die Überlagerung zu $v'|1\rangle + u'|5\rangle$. Der Vektor $|5\rangle$ ist mit dem verwerfenden Zustand q_{rej} assoziiert. Der Automat verwirft deswegen an dieser Stelle mit Wahrscheinlichkeit $|u'|^2$. Dieselbe Situation wiederholt sich bei jeder folgenden Sequenz ba. A verwirft also mit positiver Wahrscheinlichkeit. Nachdem die Zeichen $\vdash$ und $\dashv$ gelesen wurden, befindet sich A in der Überlagerung $|\psi''\rangle = v''|3\rangle + u''|4\rangle$. Die Eingabe wird dann ein weiteres Mal von links nach rechts gelesen. Zeichen a lassen die Überlagerung unverändert. Bei b wird die Transformation B^{-1} auf die erreichte Überlagerung angewandt. Weil im ersten Durchlauf q_{rej} gemessen werden konnte, wird nach nochmaligem Lesen der Eingabe eine Überlagerung $v'|3\rangle + u'|4\rangle$ mit $u' \neq 0$ erreicht. Beim Lesen von $\vdash$ verwirft die Maschine nochmals mit positiver Wahrscheinlichkeit und beginnt ansonsten im Zustand $|1\rangle$ erneut von vorne. Die Berechnung setzt sich solange analog fort, bis ein verwerfender Zustand erreicht wird. □

Bemerkung 4.2-6
Aus den Überführungsmatrizen des Automaten in dem Beweis zu Lemma 4.2-5 geht hervor, dass $\delta(q_2, \vdash, q_0) = 1$ gilt. Der Automat erfüllt damit alle Bedingungen aus Lemma 4.2-4.

Lemma 4.2-7
Sei $w \in \Sigma^n$. Die erwartete Laufzeit des 1-QXFA aus Lemma 4.2-5 bei einer Eingabe $w \notin L_{ab} = \{a^*b^*\}$ ist $O(n^5)$. Mit Wahrscheinlichkeit kleiner als $\left(\frac{e}{4}\right)^{O(n^4)}$ ist die Rechenzeit dann nicht größer als $\frac{2}{\varepsilon^2} \cdot O(n)$.

Beweis. $w \notin L_{ab}$ enthält die Sequenz ba. Wenn die erste Sequenz ba von dem *1-QXFA* gelesen wird, befindet sich der Automat in einer Überlagerung $v'|1\rangle + u'|2\rangle$ mit $u' \neq 0$. Diese Überlagerung wird dann zu $v'|1\rangle + u'|5\rangle$ verändert. Analog zu der Laufzeitabschätzung des *1-QXFA*^{vm} für $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ können wir zeigen, dass $u' > \varepsilon = \sin(\kappa)$ mit $\frac{\kappa}{\pi} = 10^{\lceil -(3+2\log_{10} n)\rceil}$ gilt. Dazu überlegen wir uns, dass u' erst dann kleiner als ε wird, wenn ausreichend Drehungen B oder B^{-1} auf der Ausgangsüberlagerung $|1\rangle$ durchgeführt

wurden. Wegen $B^{-1}B = I$ ist es irrelevant, welche Drehung wir betrachten. Wir betrachten deswegen die für uns ungünstigste Eingabe $w = b^n a$. Nachdem n-mal die Drehung B auf der Überlagerung $|1\rangle$ durchgeführt wurde, ist die Amplitude u' von $|2\rangle$ bzw. $|5\rangle$ noch größer als ε und damit die Wahrscheinlichkeit zu verwerfen größer als ε^2, weil der Vektor $|5\rangle$ mit dem verwerfenden Zustand q_{rej} assoziiert ist. Die erwartete Anzahl an Lesezyklen ist also $\frac{1}{\varepsilon^2}$. Als erwartete Laufzeit erhalten wir $O(n\varepsilon^{-2}) = O(n^5)$. Mit Chernoff-Schranken (vgl. Satz 1.1-2) können wir abschätzen, dass der Automat mit Wahrscheinlichkeit kleiner als $\left(\frac{e}{4}\right)^{\varepsilon^{-2}} = \left(\frac{e}{4}\right)^{O(n^4)}$ mehr als $\frac{2}{\varepsilon^2}$ Lesezyklen benötigt, bis ein verwerfender Zustand gemessen wird. □

Die Sprache L_{ab} ist regulär. Ein *1-DFA* kann die Sprache bei einer Eingabe $w \in \Sigma^*$ sogar in Zeit $O(|w|)$ einfacher erkennen, als der eben beschriebene *1-QXFA*. Wir können jedoch mit diesem Ergebnis zeigen, dass auch eine nicht-reguläre Sprache von einem *1-QXFA* erkannt werden kann. In [AF98] finden wir ohne Beweis den folgenden Satz.

Satz 4.2-8 *(Theorem* 13 *aus [AF98])*
Sei $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$. Es gibt einen 1-QXFA A mit

1. *Für $w \notin L_{eq}$ hält A mit Wahrscheinlichkeit* 1*, nachdem das Eingabeband $O(|w|)$-mal gelesen wurde,*
2. *Für $w \in L_{eq}$ hält A nie.*

Wir können ein etwas schwächeres Ergebnis zeigen.

Satz 4.2-9
Sei $w \in \Sigma^m$. Es gibt einen 1-QXFA, der bei Eingabe w genau dann nie in einen haltenden Zustand gerät, wenn $w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ ist und der andernfalls mit Wahrscheinlichkeit Eins verwirft. Die erwartete Laufzeit des Automaten bei einer Eingabe $w \notin L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ ist $O(m^5)$. Mit Wahrscheinlichkeit $1 - \left(\frac{e}{4}\right)^{O(m^4)}$ verwirft der 1-QXFA nach $\frac{4}{\varepsilon^2} \cdot O(m)$ Rechenschritten.

Beweis. Wir wissen aus Satz 4.1-16, dass es einen *1-QXFA^{vm}* gibt, für den bei einer Eingabe $w \in \Sigma^m$ die erwartete Laufzeit $O(m^5)$ beträgt, falls $w \notin L_{eq}$ ist und der ansonsten nie hält. Eingaben $w \notin L_{eq}$ werden stets verworfen. Die Berechnung des Automaten ist in zwei Phasen aufgeteilt (vgl. Lemma 4.1-11), wobei in der ersten Phase ein *1-DFA* simuliert wird, um L_{ab} zu erkennen. In der zweiten Phase muss dann noch $L_{a=b} = \{w \in \{a,b\}^* | \ |w|_a = |w|_b\}$ entschieden werden. Dazu führt der *1-QXFA^{vm}* je nach Eingabebuchstabe Drehungen auf der erreichten Überlagerung durch. Genau dann, wenn dieselbe Anzahl a-Drehungen wie b-Drehungen durchgeführt werden, wird eindeutig eine Überlagerung $|\psi\rangle$ erreicht. Die verallgemeinerten Messungen sind dazu nicht notwendig. Die zweite Phase kann also ebenso von einem *1-QXFA* berechnet werden. In der ersten

Phase muss dann sichergestellt werden, dass in der Eingabe w nie ein Buchstabe a auf einen Buchstaben b folgt. Wir kennen aus Lemma 4.2-5 schon einen Automaten, der dies testet. Die erwartete Rechenzeit des Automaten ist für eine Eingabe $w \in \Sigma^m \setminus L_{ab}$ wieder $O(m^5)$ (vgl. Lemma 4.2-7). Beide Automaten erfüllen die Voraussetzungen zu Lemma 4.2-4 (vgl. Bemerkung 4.1-12 und Bemerkung 4.2-6). Wir können durch das Kombinieren beider Automaten einen *1-QXFA* für $L_{eq} = L_{a=b} \cap L_{ab}$ konstruieren. Die Laufzeit dieses neuen Automaten ist nach Lemma 4.2-4 wieder $O(m^5)$. Wir wissen aus Satz 4.1-16 und Lemma 4.2-7, dass beide Automaten mit Wahrscheinlichkeit $\left(\frac{e}{4}\right)^{O(m^4)}$ mehr als $\frac{2}{\varepsilon^2}$ Lesezyklen benötigen, um ein Wort, das nicht aus der Sprache ist, zu verwerfen. Nach der Verkettung gemäß Lemma 4.2-4 verwirft der resultierende Automat dann mit Wahrscheinlichkeit $\left(\frac{e}{4}\right)^{O(m^4)}$ erst nach $\frac{4}{\varepsilon^2}$ Lesezyklen. □

Als Vorbereitung auf ein Ergebnis in Kapitel 4.3 Fassen wir hier noch mal einige Eigenschaften des Automaten aus Satz 4.2-9 in der folgenden Bemerkung zusammen.

Bemerkung 4.2-10
Der 1-QXFA aus Satz 4.2-9 ist aus zwei 1-QXFAs zusammengesetzt, die die Sprachen L_{ab} und $L_{a=b}$ entscheiden. Der 1-QXFA für L_{ab} benötigt zwei Lesezyklen. Je nachdem, für welche Sprache die Zugehörigkeit der Eingabe $w \in \Sigma^*$ gegenwärtig getestet wird, unterscheiden sich die Überlagerungen, in denen sich der 1-QXFA für L_{eq} nach dem Lesen des letzten Buchstaben w_m der Eingabe w mit $|w| = m$ befinden kann. Seien dies die Überlagerungen

$$\begin{array}{ll} \alpha^1|q^1\rangle + \beta^1|q^2\rangle, & \textit{falls } w \in L_{a=b} \textit{ getestet wird,} \\ \alpha^2|q^3\rangle + \beta^2|q^4\rangle, & \textit{falls der erste Teil von } w \in L_{ab} \textit{ getestet wird,} \\ \alpha^3|q^5\rangle + \beta^3|q^6\rangle, & \textit{falls der zweite Teil von } w \in L_{ab} \textit{ getestet wird} \end{array}$$

(vgl. Lemma 4.1-11 und Lemma 4.2-5). Die beiden Endmarkierungen werden genutzt, um zwischen den verschiedenen Phasen des Automaten zu wechseln, bzw. in eine Überlagerung zu wechseln, die zum Verwerfen führen kann. Dabei ist

$$w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\} \Leftrightarrow \beta^1 = \beta^3 = 0. \tag{4.2}$$

Ein *1-QFA* mit zyklischem Band und einem speziellen Akzeptanzmodus ist in der Lage, nicht-reguläre Sprachen zu erkennen. Wir kennen jedoch keine nicht-reguläre Sprache, die von einem *1-QXFA* nach Definition 4.2-1 ohne den speziellen Akzeptanzmodus erkannt wird. Es ist nicht klar, ob alle regulären Sprachen erkannt werden können bzw. ob schon das zyklische Band alleine dafür ausreicht.

4.3 Kombination verschiedener Automatentypen

In Anlehnung an Definition 2.2-7 (siehe Seite 30) definieren wir die Automatentypen *1-QFA*$^{(2\text{-}DFA)}$ und *1-QFA*$^{(2\text{-}PFA)}$. Dabei kombinieren wir jeweils einen *1-QFA* (vgl. Definition 3.2-1, S. 39) mit einem weiteren Automaten. In dem Fall des *1-QFA*$^{(2\text{-}DFA)}$ ist dies

ein *2-DFA* mit Ausgabe (vgl. Definition 2.1-1, S. 24), im Fall des *1-QFA*$^{(2\text{-}PFA)}$ entsprechend ein *2-PFA* mit Ausgabe (vgl. Definition 2.2-1, S. 27).

Bemerkung 4.3-1
Die Menge der regulären Sprachen ist eine Teilmenge der von 1-QFA$^{(2\text{-}PFA)}$*s erkannten Sprachen und eine Teilmenge der von 1-QFA*$^{(2\text{-}DFA)}$*s erkannten Sprachen (vgl. Bemerkung 2.2-8, S. 30).*

Wir betrachten zunächst einen *1-QFA*$^{(2\text{-}DFA)}$ und interessieren uns besonders für die Ausgabe, die der *2-DFA* erzeugt.

Lemma 4.3-2
Gegeben sei ein 1-QFA$^{(2\text{-}DFA)}$ *für eine Sprache* L_{ein}*. Für alle* $w \in \Sigma^*$ *sei* y_w *die Ausgabe des 2-DFA bei Eingabe* w*. Die Sprache* $L_{aus} = \{y_w | w \in L_{ein}\}$ *ist regulär.*

Beweis. Weil wir bei dem *2-DFA* einen deterministischen Automaten betrachten, produziert dieser zu jeder möglichen Eingabe deterministisch eine konkrete Ausgabe. Zu jedem $w \in L_{ein}$ gibt es also genau ein $y_w \in \Sigma^*_{aus}$. Es können genau dann verschiedene Eingabewörter auf dasselbe Ausgabewort abgebildet werden, wenn beide Eingaben aus L_{ein} oder beide nicht aus L_{ein} sind. Andernfalls können wir einen Widerspruch erzeugen.

Seien $w \in L_{ein}$, $w' \notin L_{ein}$ und y_w bzw. $y_{w'}$ die zugehörigen Ausgabewörter mit $y_w = y_{w'}$. Weil der *1-QFA*$^{(2\text{-}DFA)}$ jede Eingabe $w \in L_{ein}$ für $\varepsilon \in [0, \frac{1}{2}[$ mindestens mit Wahrscheinlichkeit $1 - \varepsilon$ akzeptiert, muss der *1-QFA* die zugehörige Ausgabe y_w mindestens mit Wahrscheinlichkeit $1 - \varepsilon$ akzeptieren. Allerdings akzeptiert der *1-QFA*$^{(2\text{-}DFA)}$ jede Eingabe $w' \notin L_{ein}$ mit Wahrscheinlichkeit kleiner als ε. Der *1-QFA* muss die Ausgabe $y_{w'} = y_w$ also mit einer Wahrscheinlichkeit kleiner als ε akzeptieren, was ein Widerspruch ist.

Wir fassen alle y_w mit $w \in L_{ein}$ zu der Menge L_{aus} zusammen. Weil die Wahrscheinlichkeit des *1-QFA*$^{(2\text{-}DFA)}$, eine Eingabe $w \in L_{ein}$ zu akzeptieren, mindestens $1 - \varepsilon$ ist und es keine Ausgabe y_w des *2-DFA* mit $y_w \notin L_{aus}$ und $w \in L_{ein}$ gibt, muss der nachgeschaltete *1-QFA* jede Eingabe $y_w \in L_{aus}$ mindestens mit Wahrscheinlichkeit $1 - \varepsilon$ akzeptieren. Mit derselben Begründung akzeptiert der *1-QFA* eine Eingabe y_w mit $w \notin L$ nur mit einer Wahrscheinlichkeit kleiner als ε. Der *1-QFA* erkennt also die Sprache L_{aus}. Die Menge der von *1-QFA*s erkannten Sprachen ist eine echte Teilmenge der regulären Sprachen (siehe [KW97]). Demnach ist L_{aus} eine reguläre Sprache. □

Wir können den *1-QFA* in dem *1-QFA*$^{(2\text{-}DFA)}$ durch einen *1-DFA* ersetzen, der die reguläre Sprache L_{aus} entscheidet und erhalten so einen *1-DFA*$^{(2\text{-}DFA)}$ für eine Sprache L_{ein}. Analog zu dem Vorgehen in Lemma 2.2-9 (siehe Seite 31) können wir dann sogar einen *2-DFA* entwerfen, der die Sprache L_{ein} erkennt. Zusammen mit Bemerkung 4.3-1 erhalten wir den folgenden Satz.

Satz 4.3-3
Sei $\varepsilon \in [0, \frac{1}{2}[$. Jede von einem 1-QFA$^{(2\text{-}DFA)}$ mit einem Fehler von ε erkannte Sprache ist regulär.

Wir konnten bei einem *1-QFA*$^{(2\text{-}DFA)}$ die beiden beteiligten Automaten getrennt betrachten und den einen Automaten durch den anderen Automaten simulieren, weil die Ausgabe des *2-DFA*s eine reguläre Sprache ist. Was können wir erreichen, wenn wir für die Vorabberechnung einen probabilistischen Automaten verwenden? In [AF98] wird ohne Beweis behauptet, dass es einen *1-QFA*$^{(2\text{-}PFA)}$ gibt, der sogar mehr Sprachen erkennen kann, als ein *1-QFA* bzw. ein *2-PFA* alleine.

Lemma 4.3-4 *(Theorem 13 in [AF98])*
Für beliebiges $\varepsilon > 0$ gibt es einen 2-PFA P und einen 1-QFA A, sodass für jede Eingabe $w \in \Sigma^*$ mindestens mit Wahrscheinlichkeit $1 - \varepsilon$ gilt

1. *P hält nach $O(|w|^2)$ Rechenschritten,*
2. *A akzeptiert die Ausgabe von P genau für $w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$.*

Wir können ein etwas schwächeres Ergebnis zeigen. Dazu benötigen wir zunächst die folgenden Definitionen.

Definition 4.3-5
Sei $P = (Q, \Sigma, q_0, \delta, Q_{stop})$ ein 2-PFA mit Ausgabe und $\Gamma = \Sigma_{ein} \cup \{\dashv, \vdash\}$ das Bandalphabet des Eingabebandes. P führt ein Turnier aus, wenn er bei Eingabe $w \in \Sigma^n$

1. *den Lesekopf an die Stelle w_n des letzten Buchstabens der Eingabe bewegt und*
2. *ihn solange mit Wahrscheinlichkeit $\frac{1}{2}$ einen Buchstaben nach links bzw. rechts bewegt, bis eine der Endmarkierungen $\dashv$ bzw. $\vdash$ erreicht wird.*

Das Turnier ist beim Erreichen einer der beiden Endmarkierungen beendet.

Definition 4.3-6
Sei $l \in \mathbb{N} \setminus \{0\}$. Ein Turnier heißt rechtsseitiges, l-faches Turnier, wenn der 2-PFA mit Ausgabe das Turnier nur beim Erreichen der rechten Endmarkierung beendet, beim Lesen der linken Endmarkierung den Kopf wieder zurück auf den letzten Buchstaben der Eingabe bewegt und ihn erneut mit Wahrscheinlichkeit $\frac{1}{2}$ einen Schritt nach links bzw. rechts bewegt. Erst nachdem das Turnier $(l-1)$-mal auf diese Weise fortgesetzt wurde, wird es beim l-ten Erreichen der linken Endmarkierung $\dashv$ beendet.

Definition 4.3-7
Ein rechtsseitiges, l-faches Turnier heißt erfolgreich, wenn es durch das Erreichen der linken Endmarkierung beendet wird. Sobald in einem rechtsseitigen, l-fachen Turnier die rechte Endmarkierung erreicht wird, nennen wir das Turnier nicht-erfolgreich. Das Turnier ist dann sofort beendet.

Definition 4.3-8
Sei $P = (Q, \Sigma, q_0, \delta, Q_{stop})$ ein 2-PFA mit Ausgabe und $l \in \mathbb{N} \setminus \{0\}$. P führe eine Menge von rechtsseitigen, l-fachen Turnieren durch. Eine Menge von Turnieren, beginnend mit dem ersten nicht erfolgreichen Turnier bis einschließlich dem ersten erfolgreichen Turnier, nennen wir eine Runde.

Wir möchten einen *1-QFA*$^{(2\text{-}PFA)}$ für L_{eq} angeben. Wir suchen also einen *2-PFA* mit Ausgabe, der eine Eingabe $w \in \Sigma^*$ durch seine Bearbeitung in eine Form y bringt, in der ein geeigneter *1-QFA* eine Entscheidung auf y treffen kann. Bei der Konstruktion des nachgeschalteten *1-QFA* A wird uns der *1-QXFA* für L_{eq} (vgl. Satz 4.2-4) als Vorlage dienen. Ein *1-QFA* liest die Eingabe nur einmal. Weil der Automat A einem *2-PFA* mit Ausgabe nachgeschaltet sein soll, ist es möglich, dass in der Ausgabe y des *2-PFA*s die Eingabe w mehrfach hintereinander geschrieben wird. So kann der *1-QFA* A die Ausgabe y nur einmal, das Wort w jedoch mehrfach lesen.

Wir werden einen *2-PFA* vorstellen, der bei Eingabe w eine Ausgabe $y = \lhd w \rhd \ldots \lhd w \rhd$ erzeugt. Die Zeichen $\lhd$ und $\rhd$ stellen Trennsymbole dar, die nicht in dem Bandalphabet des Eingabebandes von dem *2-PFA* vorkommen. Die Länge der Ausgabe des Automaten hängt von einer Konstanten $z \in \mathbb{N}$ ab. Wir werden den Automaten deswegen P_z nennen. Die Berechnung des *2-PFA*s P_z beschränkt sich darauf, dass er eine Menge von rechtsseitigen, 4-fachen Turnieren durchführt und vor jedem Turnier einmal die Ausgabe $\lhd w \rhd$ schreibt. Nach dem ersten Turnier hat die Ausgabe die Gestalt $\lhd w \rhd$ und nach dem k-ten Turnier die Gestalt

$$(\lhd w \rhd)^k = \underbrace{\lhd w \rhd \lhd w \rhd \ldots \lhd w \rhd}_{k-mal\ \lhd w \rhd}.$$

Die Folge von Turnieren lässt sich gemäß Definition 4.3-8 in Runden einteilen. Nach jedem erfolgreichen Turnier beginnt für P_z eine neue Runde. Diese Arbeitsweise setzt der *2-PFA* unverändert fort, bis schließlich z Runden abgeschlossen sind. Danach ist die Berechnung beendet. Die Zustandsmenge Q^{P_z} des *2-PFA* $P_z = \{Q^{P_z}, \Sigma^{P_z}_{ein}, \Sigma^{P_z}_{aus}, q_0^{P_z}, \delta^{P_z}, Q^{P_z}_{stop}\}$ mit Ausgabe lässt sich weiter unterteilen in disjunkte Teilmengen $Q_i^{P_z},\ i \in \{1, \ldots, z\}$ mit $\bigcup_{i=1}^{z} Q_i^{P_z} = Q^{P_z}$ und $q_0^{P_z} \in Q_1^{P_z}$ sowie $Q^{P_z}_{stop} \subset Q_z^{P_z}$. Für $\lhd, \rhd \notin \Gamma$ ist das Ausgabealphabet $\Sigma^{P_z}_{aus} = \Sigma^{P_z}_{ein} \cup \{\lhd, \rhd\}$. Für jede Runde reservieren wir also eine eigene Zustandsmenge. Die Zustandsmengen können nur in aufsteigender Reihenfolge durchlaufen werden. Für $j \in \{1, \ldots, z\}$ führt das Verlassen der Zustandsmenge $Q_j^{P_z}$ mit Wahrscheinlichkeit Eins in die Zustandsmenge $Q_{j+1}^{P_z}$. Die Zustandsüberführungsfunktion δ^{P_z} gewährleistet, dass der Automat für $i \in \{1, \ldots, z-1\}$ in jeder Zustandsmenge $Q_i^{P_z}$ eine Runde gemäß

Definition 4.3-8 durchführt und nach Abschluss der Runde in die nächst höhere Zustandsmenge $Q_{i+1}^{P_z}$ wechselt. Die Berechnung in Zustandsmenge $Q_z^{P_z}$ verläuft größtenteils analog zu der Berechnung in allen anderen Zustandsmengen. Anstatt die Menge zu verlassen, wird die Berechnung in Runde z abgebrochen, sobald ein rechtsseitiges 4-faches Turnier erfolgreich beendet wird. Der folgende Pseudocode verdeutlicht die Arbeitsweise von P_z. Durch die Zählvariable i wird die Zustandsmenge $Q_i^{P_z}$, in der sich der Automat gegenwärtig befindet, beschrieben.

$i := 1$;

WHILE $(i \leq z)$
    Schreibe einmal $\lhd w \rhd$;
    führe ein 4-faches Turnier aus;

    IF (Turnier erfolgreich)
        $i++$;
    ENDIF

ENDWHILE

Lemma 4.3-9
Sei $w \in \Sigma^n$ eine Eingabe für den 2-PFA P_z und z konstant. Die Ausgabe des Automaten ist dann $y = \lhd w \rhd \ldots \lhd w \rhd$. Die erwartete Anzahl an Vorkommen von $\lhd w \rhd$ in y ist $z(n+1)^4$. Die erwartete Laufzeit von P_z ist $O(n^6)$.

Beweis. Vor Beginn jedes Turniers wird das Zeichen $\lhd$ geschrieben. Danach liest der Automat die Eingabe w einmal von links nach rechts und schreibt dabei jedes Zeichen unverändert in die Ausgabe. Abschließend wird das Zeichen $\rhd$ geschrieben. Es findet in einem Turnier keine weitere Ausgabe statt. Vor jedem Turnier wird also genau einmal die Sequenz $\lhd w \rhd$ geschrieben. Es werden z Runden durchlaufen. In jeder Runde werden so lange Turniere durchgeführt, bis in einem Turnier viermal die linke Endmarkierung erreicht wird, ohne vorher die rechte Endmarkierung zu erreichen.

Wir werden nun die Wahrscheinlichkeit für dieses Ereignis untersuchen. Dabei helfen uns die Ergebnisse über das *Gamblers Ruin Problem* aus Kapitel 1.1.1. Der *2-PFA* simuliert in einem Turnier ein Spiel zwischen den Spielern S_1 und S_2. Die Kopfposition auf der Eingabe spiegelt die Vermögenssituation der Spieler wider. Wenn der Kopf über dem letzten Zeichen w_n der Eingabe steht, besitzt Spieler S_1 eine Geldeinheit, wenn er über dem vorletzten Zeichen steht, entsprechend zwei Geldeinheiten und so fort. Die maximal erreichbare Geldsumme ist somit $n+1$. In jedem Schritt führt der Automat ein Spiel zwischen S_1 und S_2 durch. Der Kopf bewegt sich mit Wahrscheinlichkeit $\frac{1}{2}$ entweder nach links oder nach rechts. Übertragen auf das *Gamblers Ruin Problem* bedeutet dies, dass jeder Spieler das Spiel mit Wahrscheinlichkeit $\frac{1}{2}$ gewinnt ($\mathcal{P}_1 = \mathcal{P}_2 = \frac{1}{2}$). Nach den Erkenntnissen aus Kapitel 1.1.1 treibt Spieler S_1 seinen Kontrahenten mit Wahrscheinlichkeit $\frac{1}{n+1}$ in den

Ruin (vgl. Gleichung 1.1 S. 9). Eine Runde endet, wenn dies dem Spieler S_1 vier mal in direkter Folge gelungen ist. Die Wahrscheinlichkeit hierfür ist $\frac{1}{(n+1)^4}$. Der Erwartungswert für die Anzahl der Turniere in einer Runde ist also $(n+1)^4$. Weil der Automat z Runden durchläuft, ist der Erwartungswert für die Anzahl der Vorkommen der Sequenz $\lhd w \rhd$ in der Ausgabe $z(n+1)^4$. Um die Eingabe $(n+1)^4$-mal zu schreiben, sind $O(n^4|w|) = O(n^5)$ Rechenschritte notwendig. Zusätzlich wird jedesmal, nachdem die Sequenz $\lhd w \rhd$ geschrieben wurde, mehrfach das *Gamblers-Ruin Problem* simuliert. Wir wissen schon, dass bei einer maximal erreichbaren Geldsumme von $n+1$ der Erwartungswert für die Dauer eines Spieles, in dem der Spieler S_1 ein Startkapital von a Geldeinheiten hat, $D_a = a(n+1-a)$ ist (vgl. Gleichung 1.2 S. 9). In unserem Fall erhalten wir also $D_1 = (n+1-1) = n$. Die zusätzlichen Kosten wie das einmalige Überlaufen der Eingabe, um den Kopf an die richtige Position zu bringen, schlagen maximal mit $O(n)$ zu Buche. Die erwartete Dauer, bis ein Spiel erfolgreich beendet wird, ist jedoch $2n^2$ (vgl. Satz 1.1-12 S. 11). Deswegen erhalten wir insgesamt als erwartete Rechenzeit $O(cn2n^2n^4) = O(n^6)$ für eine Konstante c. □

Wir haben schon erwähnt, dass wir den *1-QXFA* $A_X = (Q^{A_X}, \Sigma^{A_X}, q_0^{A_X}, \delta^{A_X}, Q_{rej}^{A_X})$ für L_{eq} aus Satz 4.2-4 als Vorlage für den *1-QFA* $A = (Q^A, \Sigma^A, q_0^A, \delta^A, Q_{acc}^A, Q_{rej}^A)$ nutzen werden. Der Startzustand $q_0^{A_X}$ ist auch Startzustand von A. Seien $\lhd, \rhd \notin \Sigma^A$. Abgesehen von den folgenden Änderungen werden wir die Zustandsmenge Q^{A_X}, das Eingabealphabet Σ^{A_X} und die Zustandsüberführungsfunktion δ^{A_X} unverändert in den *1-QFA* A übernehmen.

1. $\Sigma^A = \Sigma^{A_X} \cup \{\lhd, \rhd\}$,
2. $Q^A = Q^{A_X} \cup \{q_{acc}^2, q_{acc}^4, q_{acc}^5, q_{acc}^6\} \cup \{q_{rej}^1, q_{rej}^3\}$

   mit $q_{acc}^i, q_{rej}^j \notin Q^{A_X}, i \in \{2,4,5,6\}, j \in \{1,3\}$,
3. $Q_{acc}^A = \{q_{acc}^2, q_{acc}^4, q_{acc}^5, q_{acc}^6\}$,

   $Q_{rej}^A = Q_{rej}^{A_X} \cup \{q_{rej}^1, q_{rej}^3\}$,
4. Seien $q^1, \ldots, q^6 \in Q^{A_X}$ die sechs Zustände, in denen sich A_X beim Lesen der rechten Endmarkierung $\vdash$ befinden kann (vgl. Bemerkung 4.2-10, S. 61). Nach der Definition des Automaten aus Satz 4.2-4 führen davon zwei Zustände zum Verwerfen des Automaten. Aus den restlichen Zuständen wird die Berechnung des *1-QXFA* in einem weiteren Lesezyklus fortgesetzt. Seien q^1 und q^3 die beiden Zustände, die zum Verwerfen des Automaten führen und q^2, q^4, q^5, q^6 die restlichen Zustände. Für alle $q, q' \in Q^A$ und $i \in \{2,4,5,6\}, j \in \{1,3\}$, gilt

$$\delta^A(q,\sigma,q') = \begin{cases} \delta^{A_X}(q, \dashv, q'), & \text{falls } \sigma = \lhd, \\ \delta^{A_X}(q, \vdash, q'), & \text{falls } \sigma = \rhd, \\ 1, & \text{falls } q = q' = q_0^A \text{ und } \sigma = \dashv, \\ 1, & \text{falls } q = q^i, q' = q_{acc}^i \text{ und } \sigma = \vdash, \\ 1, & \text{falls } q = q^j, q' = q_{rej}^j \text{ und } \sigma = \vdash, \\ \delta^{A_X}(q,\sigma,q'), & \text{sonst.} \end{cases}$$

Lemma 4.3-10
Sei $\varepsilon = \sin(\frac{\pi}{10^{\lceil 3+2\log_{10} m\rceil+1}})$, $w \in \Sigma^m$ *und* $y = \lhd w \rhd \ldots \lhd w \rhd$ *die Eingabe für den 1-QFA A. Der Automat A akzeptiert jede Eingabe* $y = \lhd w \rhd \ldots \lhd w \rhd$ *mit* $w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ *mit Wahrscheinlichkeit Eins. Für* $w \notin L_{eq}$ *und* $|y| \geq (m+2)\frac{3}{\varepsilon^2} = O(m^5)$ *akzeptiert A mit Wahrscheinlichkeit* $1-(1-\varepsilon^2)^{\varepsilon^{-2}} = 1-\left(1-\frac{1}{O(m^4)}\right)^{O(m^4)}$

Beweis. Der Wert von ε ist mit $\varepsilon = \sin(\frac{\pi}{10^{\lceil 3+2\log_{10} m\rceil+1}})$ wie für den *1-QXFA* aus Satz 4.2-4 gewählt (vgl. auch Lemma 4.2-7). Die Berechnung wird wie folgt durchgeführt. Der Startzustand des Automaten ist q_0. Nachdem Lesen der linken Endmarkierung $\dashv$ bleibt der Automat in dem Zustand q_0. Danach verhält sich der *1-QFA A* auf den Teilen $\lhd w \rhd$ der Eingabe genauso wie der *1-QXFA* A_X für L_{eq} auf der Buchstabenfolge $\dashv\ w\ \vdash$, weil bei der Konstruktion der Überführungsfunktion δ^A aus δ^{A_X} die Zeichen $\dashv$ und $\vdash$ durch $\lhd$ bzw. $\rhd$ ersetzt werden. In welchem Zustand sich der *1-QFA* beim Lesen von $\vdash$ befindet, können wir deswegen der Überführungsfunktion δ^{A_X} entnehmen. A_X testet alternierend, ob $w \in L_{ab} = \{a^*b^*\}$ bzw. ob $w \in L_{a=b} = \{w | w \in \{a,b\}^*$ mit $|w|_a = |w|_b\}$ gilt. Im ersten Fall benötigt der Test zwei Lesezyklen, im zweiten Fall nur einen Lesezyklus. In jedem Fall befindet sich der *1-QXFA* nach jedem Lesen von $\dashv w \vdash$ in einer Überlagerung $\alpha|q\rangle + \beta|q'\rangle$. Dies geht aus der Arbeitsweise des *1-QXFA* für L_{eq} hervor. Analog zu Bemerkung 4.2-10 seien diese Überlagerungen mit $\alpha^i, \beta^i \in \mathbb{C}, i \in \{1,2,3\}$,

$$\begin{array}{ll} \alpha^1|q^1\rangle + \beta^1|q^2\rangle, & \text{falls } w \in L_{a=b} \text{ getestet wird,} \\ \alpha^2|q^3\rangle + \beta^2|q^4\rangle, & \text{falls der erste Teil von } w \in L_{ab} \text{ getestet wird,} \\ \alpha^3|q^5\rangle + \beta^3|q^6\rangle, & \text{falls der zweite Teil von } w \in L_{ab} \text{ getestet wird.} \end{array}$$

Genau für $w \in L_{eq}$ ist $\beta^1 = \beta^3 = 0$ (vgl. 4.2 auf Seite 61). Für $i \in \{2,4,5,6\}$ und $j \in \{1,3\}$ ist $\delta(q^i, \vdash, q^i_{acc}) = 1$ bzw. $\delta(q^j, \vdash, q^j_{rej}) = 1$. Dann wird eine Eingabe $w \in L_{eq}$ in jedem Fall akzeptiert, weil es keine Möglichkeit zu verwerfen gibt. Wir haben dies bereits in Lemma 4.1-11 und Lemma 4.2-5 diskutiert. Für $w \notin L_{eq}$ betrachten wir wie wahrscheinlich vor dem Erreichen der rechten Endmarkierung $\vdash$ verworfen wird.

Der Erwartungswert für die Anzahl an Lesezyklen, bis der *1-QXFA* eine Eingabe $\dashv w \vdash$ mit $w \notin L_{eq}$ verwirft, ist höchstens $\frac{3}{\varepsilon^2}$, weil der Automat in direkter Folge die Zugehörigkeit der Eingabe zu L_{ab} und $L_{a=b}$ testet. Dafür benötigt der Automat drei Lesezyklen, weil für den ersten Test zwei Lesezyklen notwendig sind. In jedem Lesezyklus verwirft der *1-QXFA* mindestens mit Wahrscheinlichkeit ε^2, wenn die zu testende Bedingung nicht erfüllt wird. Weil für $w \notin L_{eq}$ mindestens einer der Tests nicht erfüllt ist, verwirft der *1-QXFA* in je drei Lesezyklen mindestens mit Wahrscheinlichkeit ε^2. Mit Wahrscheinlichkeit $(1-\varepsilon^2)$ verwirft der Automat nach drei Lesezyklen nicht und entsprechend verwirft der Automat mit Wahrscheinlichkeit $(1-\varepsilon^2)^{\varepsilon^{-2}}$ nach dem einmaligen Lesen einer Eingabe der Länge $(m+2)\frac{3}{\varepsilon^2}$ nicht. Deswegen verwirft der Automat mit Wahrscheinlichkeit

$$1-\left(1-\varepsilon^2\right)^{\varepsilon^{-2}} = 1-\left(1-\frac{1}{O(m^4)}\right)^{O(m^4)}.$$

□

Mit Lemma 4.3-9 und Lemma 4.3-10 über A und P_z können wir die erwartete Laufzeit des *1-QFA*$^{(2\text{-}PFA)}$ K angeben, der aus A und P_z zusammengesetzt ist.

Satz 4.3-11
Sei K der 1-QFA$^{(2\text{-}PFA)}$, der aus dem 1-QFA A_X und dem 2-PFA P_z mit $z = 6 \cdot 10^{10}$ zusammengesetzt ist sowie $w \in \Sigma^m$. K hat für die Eingabe w die erwartete Laufzeit $O(m^6)$. Es gilt

1. *$\forall\ w \in L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ akzeptiert K mit Wahrscheinlichkeit Eins,*
2. *$\forall\ w \notin L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ akzeptiert K mit Wahrscheinlichkeit $2^{-O(m^4)}$.*

Beweis. Die erwartete Rechenzeit des *2-PFA* P_z bei einer Eingabe w der Länge m ist $O(m^6)$. Der Erwartungswert für die Anzahl der Vorkommen von $\lhd w \rhd$ in der Ausgabe des *2-PFA* ist $z(m+1)^4 = 6 \cdot 10^{10} \cdot (m+1)^4$ (vgl. Lemma 4.3-9). Die erwartete Länge von $y = \lhd w \rhd \ldots \lhd w \rhd$ ist also $6 \cdot 10^{10} \cdot (m+1)^4(m+2) = O(m^5)$. Es sind dann $O(m^5)$ Schritte des *1-QFA* A notwendig, um eine solche Eingabe einmal komplett zu lesen. Wir bekommen insgesamt eine erwartete Rechenzeit von $O(m^5) + O(m^6) = O(m^6)$ für den *1-QFA*$^{(2\text{-}PFA)}$ K.

Eine Eingabe $w \in L_{eq}$ wird von dem Automaten mit Wahrscheinlichkeit Eins akzeptiert, weil der nachgeschaltete *1-QFA* eine Eingabe $y = \lhd w \rhd \ldots \lhd w \rhd$ mit $w \in L_{eq}$ nicht verwerfen kann und beim Erreichen der rechten Endmarkierung $\vdash$ akzeptiert (vgl. Lemma 4.3-10) Sei nun $w \notin L_{eq}$ mit $|w| = m$. Wir wissen aus Lemma 4.3-10, dass für $\varepsilon = \sin(\frac{\pi}{10^{\lceil 3+2\log_{10} m\rceil+1}})$ der Erwartungswert für die Häufigkeit des Lesens der Sequenz $\lhd w \rhd$ um zu verwerfen $\frac{3}{\varepsilon}$ ist. Mit Satz 1.1-2 (siehe Seite 8) sehen wir, dass der Automat nur mit Wahrscheinlichkeit kleiner als

$$\left(\frac{e}{4}\right)^{\frac{3}{\varepsilon^2}} = \left(\frac{e}{4}\right)^{O(m^4)} \tag{4.3}$$

auch nach dem $\frac{6}{\varepsilon^2}$-sten Lesen von $\lhd w \rhd$ noch nicht verworfen hat. Aus Lemma 4.3-9 wissen wir, dass der Erwartungswert für die Anzahl der Vorkommen von $\lhd w \rhd$ in der Ausgabe $6 \cdot 10^{10} \cdot (m+1)^4$ ist. Nach Lemma 1.1-3 wird $\lhd w \rhd$ mit Wahrscheinlichkeit kleiner als

$$e^{-\frac{6 \cdot 10^{10} \cdot (m+1)^4}{8}} = e^{-O(m^4)} \tag{4.4}$$

weniger als $3 \cdot 10^{10} \cdot (m+1)^4$-mal geschrieben. Es ist

$$\frac{6}{\varepsilon^2} = \frac{6 \cdot 4 \cdot 10^{(\lceil 3+2\log_{10} m\rceil+1)^2}}{\pi^2} \leq \frac{8}{\pi^2} \cdot 3 \cdot 10^{10} m^4 < 3 \cdot 10^{10} m^4 < 3 \cdot 10^{10}(m+1)^4.$$

Mit 4.3 und 4.4 zusammen verwirft der *1-QFA*$^{(2\text{-}PFA)}$ K die Eingabe $w \notin L_{eq}$ mit Wahrscheinlichkeit

$$1 - \left(\left(\frac{e}{4}\right)^{O(m^4)} + e^{-O(m^4)} \right) = 2^{-O(m^4)}.$$

□

Bemerkung 4.3-12
Die Automaten P_z und A können dahingehend erweitert werden, dass offensichtlich falsche Eingaben schneller verworfen werden. Zum Beispiel kann der 2-PFA zunächst testen, ob w aus der Sprache L_{ab} ist, im negativen Fall eine entsprechende Ausgabe y_{rej} schreiben und dieBerechnung beenden. Der 1-QFA A verwirft dann beim Lesen von y_{rej} mit Wahrscheinlichkeit Eins. Der Test, ob w aus L_{ab} ist, braucht danach nicht mehr von A durchgeführt werden. Der Erwartungswert für die Anzahl an Lesezyklen, bis der 1-QFA eine Eingabe $w \notin L_{a=b}$ verwirft, ist kleiner als $\frac{1}{\varepsilon^2}$. Das entspricht einer Verbesserung um den Faktor $\frac{1}{3}$ im Vergleich zu der Analyse in Satz 4.3-11.

Satz 4.3-13
Es existiert ein 1-QFA$^{(2\text{-}PFA)}$*, der die Sprache $L_{pal} = \{w | w \in \{a,b\}^* \text{ und } w = w^{-1}\}$ erkennt. Für eine Eingabe $w \in \Sigma^n$ ist die erwartete Rechenzeit des Automaten $O(25^n)$.*

Beweis. Aus Satz 4.2-3 erhalten wir einen *1-QXFA* für L_{pal}. Für eine Eingabe $w \notin L_{pal}$ ist die erwartete Anzahl an Lesezyklen bis der Automat verwirft 25^n. Für zwei Zeichen $\lhd, \rhd$, die nicht in dem Bandalphabet des *1-QXFA* vorkommen, können wir ähnlich zu dem Vorgehen für den *2-PFA* P_z aus Lemma 4.3-9 einen *2-PFA* $P = (Q, \Sigma_{ein}, q_0, \delta, \Sigma_{aus}, Q_{stop})$ mit Ausgabe entwerfen, der eine Eingabe w exponentiell häufig in der Form $\lhd w \rhd$ in die Ausgabe y schreibt. Die erwartete Anzahl an Vorkommen der Sequenz $\lhd w \rhd$ in der Ausgabe ist 25^n. Aus diesen beiden Automaten kann analog zu dem Vorgehen bei der Konstruktion des *1-QFA*$^{(2\text{-}PFA)}$ für L_{eq} ein *1-QFA*$^{(2\text{-}PFA)}$ für die Sprache L_{pal} entworfen werden, dessen erwartete Rechenzeit $O(25^n)$ ist. Dazu müssen die Überführungsfunktion und das Eingabealphabet des *1-QXFA* an die Symbole $\lhd, \rhd$ angepasst werden. Wir wählen diese Zeichen wieder als Platzhalter für die Endmarkierungen $\dashv$ und $\vdash$. Die Zustandsmenge des Automaten ist $Q = \{q_0, q_1, q_2, q_{stop}\}$. Der Startzustand ist q_0 und $q_{stop} \in Q_{stop}$ ist der einzige haltende Zustand. Das Ausgabealphabet von P ist $\Sigma_{aus} = \Sigma_{ein} \cup \{\lhd, \rhd\}$. Die Überführungsfunktion δ des *2-PFA* mit Ausgabe ist für alle $\sigma \in \Gamma$ wie folgt definiert:

$$\delta(q_0, \dashv, q_0, \lhd, 1) = 1, \qquad \delta(q_0, \sigma, q_0, \sigma, 1) = \tfrac{1}{25},$$

$$\delta(q_0, \sigma, q_1, \sigma, 1) = 1 - \tfrac{1}{25}, \quad \delta(q_1, \sigma, q_1, \sigma, 1) = 1,$$

$$\delta(q_0, \vdash, q_{stop}, \rhd, 0) = 1, \qquad \delta(q_1, \vdash, q_2, \rhd, -1) = 1,$$

$$\delta(q_2, \sigma, q_2, \varepsilon, -1) = 1, \qquad \delta(q_2, \dashv, q_0, \varepsilon, 0) = 1.$$

Für alle anderen Belegungen ist $\delta(q, \sigma, q', \sigma', d) = 0$. Der Automat P liest die Eingabe wiederholt von links nach rechts und schreibt dabei jeweils die Ausgabe $\lhd w \rhd$. Solange er sich noch in dem Zustand q_0 befindet, wechselt der Automat unabhängig von dem gelesenen Eingabebuchstaben mit Wahrscheinlichkeit $1 - \frac{1}{25}$ in den Zustand q_1. Der Zustand q_1 kann erst beim Lesen der rechten Endmarkierung verlassen werden. Dann wird der Zustand q_2 erreicht, in dem sich der Lesekopf des Automaten wieder zurück auf das Zeichen $\dashv$ bewegt, ohne eine Ausgabe zu schreiben. Sollte das Zeichen $\vdash$ im Zustand q_0 gelesen werden, hält der Automat.

In jedem Rechenschritt, in dem sich der Automat zu Beginn im Zustand q_0 befindet, ist die Wahrscheinlichkeit, den Zustand q_0 nicht zu verlassen, $\frac{1}{25}$. Mit Wahrscheinlichkeit $\frac{1}{25^n}$ wird q_0 demnach in einem Lesezyklus in jedem der n Rechenschritte nicht verlassen. Der Erwartungswert für die Anzahl an Lesezyklen bis dieser Fall eintritt ist 25^n. □

Wir wissen aus Bemerkung 4.1-8 (siehe Seite 44), dass die Sprache L_{pal} von keinem *2-DFA* erkannt werden kann. Damit kann keiner der in Kapitel 2 vorgestellten Automaten diese Sprache erkennen.

Bemerkung 4.3-14
Es bleibt zu untersuchen, welche Sprachen von einem 1-QFA$^{vm^{(2\text{-}PFA)}}$ erkannt werden können.

Zusammenfassung und Ausblick

To read our E-mail, how mean
of the spies and their quantum machine;
be comforted though,
they do not yet know
how to factorize twelve or fifteen.
- Volker Strassen

Spannen wir noch einmal kurz den Bogen zum Anfang. Wir haben uns dort die Frage gestellt, ob es ein Automatenmodell gibt, das nur auf eine konstante Anzahl an Qubits zurückgreift und dennoch mehr als nur die Menge der regulären Sprachen erkennen kann. Wir haben in Kapitel 2 zunächst einige klassische Automatentypen vorgestellt. Diese Automaten haben keine Qubits verwendet. Bei polynomieller Rechenzeit erkennen alle klassischen Automaten genau die Menge der regulären Sprachen. Wenn exponentielle Rechenzeit zugelassen wird, kann ein *2-PFA* die nicht-reguläre Sprache $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ erkennen (siehe Satz 2.2-3 und Satz 2.2-4, S. 29). Von bekannten Varianten von QFAs bekommen wir zunächst ein zweigeteiltes Bild. Ein *2-QFA* kann alle regulären Sprachen erkennen und auch die nicht-reguläre Sprache $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ (Lemma 3.1-6, S. 37). Das Modell hat jedoch den Nachteil, dass der Automat für verschiedene Eingabelängen unterschiedlich viele Qubits verwendet. *1-QFA*s auf der anderen Seite erkennen nur eine echte Teilmenge der regulären Sprachen (Lemma 3.2-3 und Lemma 3.2-4, S. 39).

Wir haben in Kapitel 4 verschiedene Modifikationen von *1-QFA*s vorgestellt. *1-QFA*^{vm}s (Kapitel 4.1) können wie die klassischen Automaten aus Kapitel 2 in polynomieller Rechenzeit die Menge der regulären Sprachen erkennen (Satz 4.1-3, S. 42). Wir wissen nicht, ob es möglich ist, nicht-reguläre Sprachen mit diesem Modell zu erkennen. Die Größe der Zustandsmenge eines minimalen *1-QFA*^{vm} für eine Sprache L ist nicht größer als die Zustandsmenge eines minimalen *1-DFA* bzw. *1-QFA* für dieselbe Sprache (siehe Bemerkung 4.1-4, S. 43). Es ist jedoch nicht klar, ob es auch eine Sprache gibt, für die die Größe eines minimalen *1-QFA*^{vm} kleiner als die Größe jedes *1-DFA* bzw. *1-QFA* für diese Sprache ist.

Automaten mit zyklischem Band werfen verschiedene Fragen auf. Zwar können wir zeigen, dass es einen *1-QXFA*^{vm} und einen *1-QXFA* gibt, die die Sprache $L_{eq} = \{a^n b^n | n \in \mathbb{N}\}$ in erwarteter polynomieller Zeit erkennen (Satz 4.1-16, S. 52 und Satz 4.2-9, S. 60), dafür ist es jedoch notwendig, dass ein Erkennen durch Nicht-Halten des Automaten zugelas-

sen wird. Mit demselben Akzeptanzmodus können wir auch zeigen, dass die nicht-reguläre Sprache $L_{pal} = \{w | w \in \{a,b\}^* \text{ und } w = w^{-1}\}$ von einem *1-QXFA*^{vm} und einem *1-QXFA* erkannt werden kann. Für eine Eingabe $w \in \{a,b\}^*$ bekommen wir jedoch eine exponentielle erwartete Laufzeit von $O(25^{|w|})$ (Satz 4.1-9, S. 44 und Satz 4.2-3, S. 55). Wenn wir verlangen, dass der *1-QXFA* bzw. der *1-QXFA*^{vm} akzeptierend halten müssen um zu akzeptieren, ist nicht bekannt, ob mehr Sprachen als ohne ein zyklisches Band erkannt werden können.

Wir können Automaten mit einem zyklischen Band jedoch als Vorstufe zu einer Kombination verschiedener Automatentypen sehen. Wir haben einen *1-QFA*$^{(2\text{-}DFA)}$ sowie einen *1-QFA*$^{(2\text{-}PFA)}$ betrachtet (Kapitel 4.3). Beide Automaten können auf einfache Weise die Menge der regulären Sprachen erkennen (siehe Bemerkung 4.3-1, S. 62). Ein *1-QFA*$^{(2\text{-}DFA)}$ erkennt genau die Menge der regulären Sprachen (Satz 4.3-3, S. 63), ein *1-QFA*$^{(2\text{-}PFA)}$ kann sogar in polynomieller Zeit die Sprache L_{eq} entscheiden (Satz 4.3-11, S. 68). Außerdem kann die Sprache $L_{pal} = \{w | w \in \{a,b\}^* \text{ und } w = w^{-1}\}$ von einem *1-QFA*$^{(2\text{-}PFA)}$ in einer erwarteten Rechenzeit von $O(25^n)$ erkannt werden (Satz 4.3-13, S. 69). Der *1-QFA*$^{(2\text{-}PFA)}$ ist damit das mächtigste von uns betrachtete Automatenmodell, das auf ein Register aus konstant vielen Qubits zurückgreift. Die Beziehungen zwischen den einzelnen vorgestellten Automatentypen sind in dem folgenden Schaubild dargestellt. Für zwei Automatentypen besagt die Beziehung $A_1 < A_2$, dass der Automat A_1 weniger Sprachen erkennen kann, als der Automat A_2. Entsprechend bedeutet die Beziehung $A_1 \leq A_2$, dass der Automat A_2 mindestens ebensoviele Sprachen wie der Automat A_1 erkennen kann. Wir wissen dann nicht, ob die Menge der erkannten Sprachen für beide Automaten gleich ist. Wenn Automatentypen bei exponentieller Laufzeit mehr Sprachen erkennen können, sind die Modelle mit Indizes „[exp]“ und „[pol]“ gekennzeichnet. Das Automatenmodell mit dem Index „[pol]“ hat eine polynomielle Laufzeit, das Modell mit dem Index „[exp]“ dagegen eine exponentielle Laufzeit. Automaten, die zum Akzeptieren nicht halten, sind durch den Index „[nh]“ gekennzeichnet. Automatenmodelle, die dieselbe Menge an Sprachen erkennen können, sind zu einer Gruppe zusammengefasst. Das sind in diesem Schaubild die Automatentypen *1-DFA*,*2-DFA*,*2-RFA*,*2-PFA*$_{[pol]}$,*1-PFA*$^{(2\text{-}PFA)}$ und *1-QFA*$^{(2\text{-}DFA)}$.

$$
\textit{1-QFA}\left\{\begin{array}{l}
\leq \quad \textit{1-QXFA} \\
< \quad \left\{\begin{array}{l} \textit{1-DFA} \\ \textit{2-DFA} \\ \textit{2-RFA} \\ \textit{2-PFA}_{[\mathrm{pol}]} \\ \textit{1-PFA}^{(2\text{-}PFA)} \\ \textit{1-QFA}^{(2\text{-}DFA)} \end{array}\right\} \quad < \textit{1-QXFA}_{[\mathrm{nh}]} \leq \textit{1-QXFA}^{vm}_{[\mathrm{nh}]} \\
\qquad\qquad\qquad\qquad \leq \textit{1-QFA}^{vm} \leq \textit{1-QXFA}^{vm}
\end{array}\right\}
\begin{array}{lll}
< & \textit{1-QFA}^{(2\text{-}PFA)}{}_{[\mathrm{pol}]} & \leq \\
< & \textit{2-PFA}_{[\mathrm{exp}]} & < \\
< & \textit{2-QFA} &
\end{array}
\left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \textit{1-QFA}^{(2\text{-}PFA)}{}_{[\mathrm{exp}]}
$$

Wir können den *2-QFA* nicht zu den Modellen des *1-QFA*$^{(2\text{-}PFA)}{}_{[pol]}$, des *2-PFA*$_{[exp]}$ und des *1-QFA*$^{(2\text{-}PFA)}{}_{[exp]}$ in Beziehung setzen. Es ist nicht bekannt, welcher dieser Automatentypen mehr Sprachen erkennen kann und ob die Modelle vergleichbar sind. Eine vertiefende Untersuchung dieser Modelle ist notwendig, um unser Verständnis für ihre Mächtigkeit zu vervollständigen. Wegen der polynomiellen Rechenzeit scheint die Betrachtung des

1-QFA$^{(2\text{-}PFA)}_{[pol]}$ in diesem Kontext besonders interessant. Außerdem ist nicht klar, ob ein *1-QFA*$^{vm^{(2\text{-}PFA)}}$ mehr Sprachen erkennen kann, als ein *1-QFA*$^{(2\text{-}PFA)}$. Um eine komplette Gegenüberstellung mit klassischen Automaten zu ermöglichen, sollte auch das Modell des *2-PFA*$^{(2-PFA)}$ (vgl. Definition 2.2-7, S. 30) untersucht werden, über das wir gegenwärtig kaum Aussagen machen können.

Stichwortverzeichnis

Literaturverzeichnis

[AAN99] Ambainis, A., Ta-Shma, A., Nayak, A. und U. Vazirani: *Dense Quantum Coding and a Lower Bound for 1-way Quantum Automata.* Proceedings of the 31-st Annual ACM Symposium on the Theory of Computation (STOC'99), Seiten 376–383, 1999.

[ABFG99] Ambainis, A., Bonner, R., Freivalds, R., Golovkins, M. und M. Karpinski: *Quantum Finite Multitape Automata.* 20-th Conference on Current Trends in Theory and Practice of Informatics (SOFSEM'99), Lecture Notes in Computer Science, 1725:340–348, 1999.

[ABFK99] Ambainis,A., Bonner, R., Freivalds, R. und A. Kikusts: *Probabilities to Accept Languages by Quantum Finite Automata.* 2nd Annual International Computing and Combinatorics Conference (COCOON'99), Lecture Notes of Computer Science, 1627:174–183, 1999.

[AF98] Ambainis, A. und R. Freivalds: *1-Way Quantum Finite Automata: Strength, Weaknesses and Generalizations.* Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS), Seiten 332–341, 1998.

[AI99] Amano, M. und K. Iwama: *Undecidability on Quantum Finite Automata.* Proceedings of the 31st Annual ACM Symposium on the Theory of Computation (STOC'99), Seiten 368–375, 1999.

[Amb96] Ambainis, A.: *The Complexity of Probabilistic Versus Deterministic Finite Automata.* Lecture Notes of Computer Science, 1178:233–238, 1996.

[AW02] Ambainis, A. und J. Watrous: *Two-Way Finite Automata With Quantum and Classical States.* Theoretical Computer Science, 287:299–311, 2002.

[BFK01] Bonner, R., Freivalds, R. und M. Kravtsev: *Quantum Versus Probabilistic One-Way Finite Automata with Counter.* 22-th Conference on Current Trends in Theory and Practice of Informatics (SOFSEM'01), Lecture Notes in Computer Science, 2234:181–190, 2001.

[BP02] Brodsky, A. und N. Pippenger: *Characterizations of 1-Way Quantum Finite Automata.* SIAM Journal on Computing, 31(5):1456–1478, 2002.

[BV93] Bernstein, E. und U. Vazirani: *Quantum Complexity Theory.* Proceedings of the 25-th Annual ACM Syposium on the Theory of Computing (STOC'93), 20:11–20, 1993.

[Deu85] Deutsch, D.: *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer.* Proceedings of the Royal Society of London, A400:97–117, 1985.

[DS90] Dwork, C. und L. Stockmeyer: *A Time Complexity Gap for Two-Way Probabilistic Finite-State Automata.* SIAM Journal on Computing, 19(2):1011–1023, 1990.

[DS92] Dwork, C. und L. Stockmeyer: *Finite State Verifiers I: The Power of Interaction.* Journal of the Association for Computing Machinery, 39:800–828, 1992.

[Fel68] Feller, W.: *An Introduction to Probability Theroy and its Applications.*, John Wiley and Sons, Band 1, 1968.

[Fre81] Freivalds, R.: *Probabilistic Two-Way Machines.* Lecture Notes of Computer Science, 118:33–45, 1981.

[Gro96] Grover, L.K.: *A Fast Quantum Mechanical Algorithm for Database Search.* Proceedings of the 28-th Annual ACM Symposium on the Theory of Computation (STOC'96), Seiten 212–219, 1996.

[Hir01] Hirvensalo, M.: *Some Open Problems Related to Quantum Computing.* Bulletin of the European Association for Computer Science (EATCS), 74:154–170, 2001.

[Kra99] Kravtsev, M.: *Quantum Finite One-Counter Automata.* Proceedings of the 26-th Conference on Current Trends in Theory and Practice of Informatics (SOFSEM'99), Lecture Notes in Computer Science, 1725:431–440, 1999.

[KW97] Kondacs, A. und J. Watrous: *On the Power of Quantum Finite State Automata.* Proceedings of the 38th IEEE Conference on Foundations of Computer Science (FOCS), Seiten 66–75, 1997.

[MC00] Moore, C. und J.P. Crutchfield: *Quantum Automata and Quantum Grammars.* Theoretical Computer Science, 237:275–306, 2000.

[MR00] Motwani, R. und P. Raghavan: *Randomized Algorithms.* Cambridge University Press, 2000.

[NC02] Nielsen, M.A. und I.L. Chuang: *Quantum Computation and Quantum Information.* Cambridge University Press, 2002.

[Paz71] Paz, A.: *Introduction to Probabilistic Automata*, Academic Press, 1971.

[Pin97] Pin, J.E.: *On Reversible Automata.* Proceedings of Latin American Symposium on Theoretical Informatics (LATIN'92), Lecture Notes in Computer Science, 583:401–415, 1992.

[Sho97] Shor, P.W.: *Polynomial-Time Algorithms for Prime Factorizations and Discrete Logarithms on a Quantum Computer.* SIAM Journal on Computing, 26(5):1474–1483, 1997.

[Weg99] Wegener, I.: *Theoretische Informatik – eine algorithmische Einführung*, Teubner, 1999.