# Quanten-Faktorisierung

Bernhard Ömer

15. Mai 1995

## 1 Motivation

Im Gegensatz zur Auffindung und Multiplikation großer Primzahlen ist zur Faktorisierung großer Zahlen bisher kein effizienter klassischer Algorithmus bekannt. Unter einem effizienten Algorithmus versteht man ein Verfahren, dessen Ausführungszeit polynomiell mit der Länge des Inputs (in bits) zunimmt. So benötigt der beste bisher bekannte klassische Algorithmus zu Faktorisierung eine $N$-stelligen Binärzahl $O(\exp[(64/9)^{1/3}(\ln N)^{1/3}(\ln\ln N)^{2/3}])$ elementare Rechenoperationen.

Dieser Umstand wird von vielen kryptographischen Algorithmen wie dem RSA-Verfahren verwendet, das sich dadurch auszeichnet, das der Code zum Entschlüsseln der Nachricht nicht aus dem Verschlüsselungscode hergeleitet werden kann, weshalb letzterer öffentlich zugänglich gemacht werden kann (public key cryptosystem).

1994 wurde von P.W. Shor ein Quanten-Algorithmus veröffentlicht, der (für den Fall, daß die dazu nötige Hardware je gebaut wird) eine Faktorisierung in polynomieller Zeit ermöglicht.

## 2 Klassiche vs. Quantencomputer

Ein Computer ist eine Maschine die einen Zustand $s$ aus einem wohl definierten Zustandsraum $S$ darstellen und bestimmte Operationen $O \subset \{o : S \to S\}$ auf diesen Zustäden ausführen kann. Eine Folge von Operationen $< o_1, o_2, ...o_n >$ mit $o_i \in O$ heißt ein Programm. (Die Festlegung des Zustandraums $S$ sowie die Unterscheidung zwischen Daten und Programm ist bei einem real existierenden Computer nicht eindeutig!)

Bei einem klassischen Computer mit $N$ bit Speicherkapazität ist $S = Z_2^N$ und $O$ eine Teilmenge der boolschen Funktionen über $S$. Bei einem Quantencomputer ist $S$ der $2^N$ dimensionale Hilbertraum $C^{2^N}$ der Zustände von $N$ qubits und $O$ eine Teilmenge der unitären Operatoren auf $S$.

## 2.1 Reversible Berechnung

Die Beschränkung auf unitäre Operatoren und die daraus folgende Erhaltung der Information folgt aus dem 2. Hauptsatz der Thermodynamik: Jede nicht-reversible Zustandsänderung d.h. jede Löschung von Information wäre notwendigerweise von einer Energiedissipation begleitet, die die Kohärenz des Zustandes zerstören würde.

Viele elementare logische Operationen (z.B. AND) sind nicht reversibel und können daher nicht direkt durch unitäre Operatoren dargestellt werden. Sie können jedoch durch erweiterte Operationen ersetzt werden, in denen das Funktionsargument erhalten bleibt und ein zusätzliches zuvor auf $|0\rangle$ gesetztes Quantenregister den Funktionswert aufnimmt.

$$F : |x\rangle \to |f(x)\rangle \text{ nichtunitär}, \; U : |x, 0\rangle \to |x, f(x)\rangle \text{ unitär}$$

## 2.2 Speichermanagement

Um ein unnötiges Anwachsen des Speicheraufwands durch die Mitführung der Funktionsargumente zu vermeiden, hat C.H. Bennet eine Methode vorgeschlagen um diese "junk"-qubits wiederzuverwerten: Nach erfolgter Berchnung des Funktionswertes $|f(x)\rangle$ unter Verwendung eines scratch-space von $K$ qubits $|j(x)\rangle$ wird $|f(x)\rangle$ in ein bisher leeres (d.h. $|0\rangle$) Register kopiert und dann die Unkehrfunktion (d.h. der inverse Operator) berechnet, wodurch der scratch-space wieder auf $|0\rangle$ gesetzt und erneut verwendet werden kann.

$$\begin{aligned}
U : |x, 0, 0, 0\rangle &\to |x, f(x), j(x), 0\rangle \\
FANOUT : |x, f(x), j(x), 0\rangle &\to |x, f(x), j(x), f(x)\rangle \\
U^+ : |x, f(x), j(x), f(x)\rangle &\to |x, 0, 0, f(x)\rangle
\end{aligned}$$

## 2.3 Vollständigkeit

Um alle möglichen boolschen Berechnungen (und damit auch alle arithmetischen Berechnungen) durchführen zu können, ist ein vollständiger Satz von Funktionen (z.B. {AND, NOT}, {NAND} oder {NOR}) notwendig.

Mit der linearen Ionenfalle (Cirac-Zoller) sind Ein-Bit-Rotationen ($U^{(j)}$) und sog. Controlled-NOT gates ($C_{[i_1,\ldots,i_k],j}$) relativ leicht zu implementieren.

$$U^{(j)} : \begin{pmatrix} |0\rangle_j \\ |1\rangle_j \end{pmatrix} \to \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} |0\rangle_j \\ |1\rangle_j \end{pmatrix}$$

$$C_{[i_1,\ldots,i_k],j} : |e_1\rangle_{i_1} \ldots |e_k\rangle_{i_k} |e\rangle_j \to |e_1\rangle_{i_1} \ldots |e_k\rangle_{i_k} |e \oplus (e_1 \wedge \ldots \wedge e_k)\rangle_j$$

Mit $U^2$ läßt sich (bis auf einen Phasenfaktor) ein NOT, und mit $C_{[k,l],j} : |e\rangle_j \to |e \oplus (e_k \wedge e_l)\rangle_j$ sowohl AND als auch XOR für beliebige qubits realisieren.

# 3 Shors Algorithmus

## 3.1 Zahlentheoretische Grundlage

Sei $N = n_1 n_2$ mit $\mathrm{ggT}(n_1, n_2) = 1$ die zu faktorisierende Zahl, $y$ eine zufällig gewählte zu $N$ teilerfremde Zahl und $F_N$ folgende Funktion mit der Periode $r$:

$$F_N(a) = y^a \bmod N,\ F_N(a+r) = F_N(a),\ y^r \equiv 1 \bmod N$$

Ist $r$ gerade, so könne wir ein $x = y^{r/2}$ definieren, daß der Gleichung $x^2 \equiv 1 \bmod N$ genügt und somit auch Lösung einer der folgenden vier Gleichungssyteme ist:

$$x_1 \equiv 1 \bmod n_1 \equiv 1 \bmod n_2$$

$$x_2 \equiv -1 \bmod n_1 \equiv -1 \bmod n_2$$

$$x_3 \equiv 1 \bmod n_1 \equiv -1 \bmod n_2$$

$$x_4 \equiv -1 \bmod n_1 \equiv 1 \bmod n_2$$

Die ersten beiden Gleichungen haben die trivialen Lösungen $x_1 = 1$, $x_2 = -1$, die letzten beiden haben die nichttrivialen Lösungen $x_3 = a$, $x_4 = -a$. Die Existenz der Lösungen folgt direkt aus dem Chinesischen Restsatz, der besagt, das jedes System von Kongruenzen in den teilerfremden Modulen $n_1, \ldots, n_k$ genau eine Lösung modulo $n_1 \cdot \ldots \cdot n_k$ hat.

Wenn also $r$ gerade und $x \neq \pm 1$, dann ist $(a+1)$ oder $(a-1)$ ein Teiler von $N$. Man kann zeigen, das diese Bedingungen mit einer Wahrscheinlichkeit $p > \frac{1}{2}$ erfüllt sind wenn $N$ nicht von der Form $N = p^\alpha$ oder $N = 2p^\alpha$ ist. Da reine Primpotenzen (und erst recht der Faktor 2) durch effiziente klassische Algorithmen erkannt werden können, stellt diese Einschränkung kein Problem dar.

## 3.2 Durchführung

Zur Faktorisierung einer Zahl $N$ werden 2 $L$ qubits Register (plus der zur Berechnung notwendigen scratch-Register) benötigt mit $N^2 < q = 2^L < 2N^2$. Alle Register seien mit $|0\rangle$ initialisiert.

Durch Ausführung einer diskreten Fourier Transformation ($DFT_q$) oder der bitweisen Anwendung von $U^{(j)}$ auf das erste Register wird ein aus allen Zahlen $0 \leq a < q$ zusammengesetzter Zustand erreicht:

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a, 0\rangle$$

Im zweiten Register wird für ein zufälliges $y$ $F_N(a)$ berechnet:

$$\frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}|a, y^a \bmod N\rangle$$

Am zweiten Register wird nun eine Messung vorgenommen, wodurch der Zustand des zweiten Registers auf $z$ und des ersten Registers auf alle Werte $a$ mit $y^a \bmod N \equiv z$ reduziert wird. Sei $z = y^l$ so gilt $a = l + rj$ wobei $r$ die gesucht Periode von $F_N(a)$ ist.

$$\frac{1}{\sqrt{A+1}}\sum_{j=0}^{A}|l + rj, z = y^l\rangle$$

Leider kann die Periode $r$ nicht durch einfache Messung des ersten Registers ermittelt werden, das der Offset $l$ unbekannt ist. Um den Offset zu eliminieren, wird am ersten Register eine diskrete Fouriertransformation durchgeführt. Die führt auf

$$\frac{\sqrt{r}}{q}\sum_{c=0}^{q-1}\sum_{j=0}^{q/r-1}\exp\left(2\pi i\frac{jrc}{q}\right)\exp\left(2\pi i\frac{lc}{q}\right)|c, z\rangle$$

Die zweite Summe ist nur dann ungleich Null, wenn $c = kq/r$. Eine Messung von $c$ führ auf $c/q = k/r$. Wenn $\mathrm{ggT}(r, k) = 1$ kann $r$ durch Kürzen von $c/q$ ermittelt werden. Dazu sind $O(\ln r) < O(\ln N)$ Versuche notwendig, was aber die Effizienz des Verfahres nicht beeinträchtigt.

# Literatur


[1] P.W. Shor. *1994 Algorithms for quantum computation: Discrete logarithms and factoring*

[2] Artur Ekert und Richard Jozsa. *Shor's Quantum Algorithms for Factoring Numbers*

[3] David Beckman et al. *Efficient networks for quantum factoring*