Einführung in den Shor-Algorithmus

Ausarbeitung des Seminarvortrags

Daniel Truhn

1 Faktorisieren einer Zahl

Um das Problem zu lösen, eine Zahl in Primfaktoren zu zerlegen ist es offensichtlich, dass es ausreicht einen effizienten Algorithmus zu finden, der eine Zahl in zwei beliebige Faktoren zerlegt. Dieser Algorithmus kann dann iterativ angewendet werden, bis man bei der Primfaktorzerlegung angekommen ist.

Gesucht sei im Folgenden also die Zerlegung einer Zahl N in 2 Faktoren. Es wird sich zeigen, dass die Rechnungen Modulo N diese Aufgabe vereinfachen können: Angenommen, wir hätten eine Zahl r^2 gefunden, die (in den Modulo N Rechnungen) eine nichttriviale Darstellung der 1 ist:

$$r^2 = 1 \text{ mod N } \Leftrightarrow r^2 - 1 = nN \Leftrightarrow (r+1)(r-1) = nN$$

In Worten besagt diese Gleichung unter anderem als dass alle Primfaktoren, die auf der rechten Seite stehen auch auf der linken Seite stehen müssen. Wenn nun weder $(r+1)$ noch $(r-1)$ ein Vielfaches von N ist, so stecken notwendigerweise einige (genauer gesagt zumindest einer) der Faktoren in $(r+1)$ und auch einige in $(r-1)$. Dann können wir uns darauf beschränken, den größten gemeinsamen Teiler von $(r+1)$ und N zu finden. Glücklicherweise gibt es für diese Aufgabe bereits einen effizienten Algorithmus (den sogenannten euklidischen Algorithmus). Hätten wir also eine Zahl r mit obigen Eigenschaften gefunden, so wäre das Problem gelöst. Der Grund, warum die Faktorisierung einer Zahl so schwierig ist liegt aber gerade in dieser Aufgabe. Gemeinhin versucht man sie zu bewältigen, indem man die Ordnung einer Zahl Modulo N findet. Was heißt das?

Als Ordnung einer Zahl q Modulo N bezeichnet man die kleinste ganze Zahl k mit der Eigenschaft, dass

$$q^k \text{ mod N } = 1$$

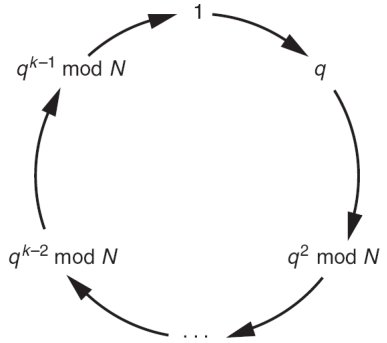
Hier ist ein Beispiel sicherlich ganz nützlich: Wir suchen die Ordnung von 2 Modulo 15.

$$2^1 = 2 \bmod 15$$
$$2^2 = 4 \bmod 15$$
$$2^3 = 8 \bmod 15$$
$$2^4 = 16 \bmod 15 = 1 \bmod 15$$

Die gesuchte Ordnung ist in diesem Fall also 4

An diesem Beispiel sieht man auch bereits, inwieweit das Ordnungsfinden weiterhelfen kann: ist die Ordnung zufällig eine gerade Zahl 2l(was mit genügend hoher Wahrscheinlichkeit der Fall ist), so wählt man einfach $r = q^l$. In unserem Beispiel wäre also $l = \frac{4}{2} = 2$ und somit $r = 2^2 = 4$. Nach obiger Argumentation hat $(r-1) = (4-1) = 3$ einen gemeinsamen Teiler mit $N = 15$. (Trivialerweise 3) auch $(r+1) = (4+1) = 5$ hat den gemeinsamen Teiler 5 mit 15. Man geht so vor, dass man sich eine zufällige Zahl $a < N$ vorgibt, deren Ordnung berechnet, und dann (falls die Ordnung gerade ist) leicht nichttriviale Faktoren von N ausrechnen kann.

Die Brute Force Methode die Ordnung einer Zahl q mod N zu bestimmen, wäre nacheinander die Potenzen $q^1, q^2, q^3, ...$ auszurechnen, bis man bei 1 angekommen ist. Offensichtlich ist dieses Vorgehen bei sehr großen Zahlen N nicht mehr sinnvoll. Eine andere Beobachtung kann helfen, die Ordnung effizienter auszurechnen: geht man obige Folge immer weiter durch, so sieht man leicht, dass es eine periodische Folge mit der Ordnung als Periode ist:



Wenden wir uns nun mehr der quantenmechanischen Betrachtungsweise zu: Definieren wir den unitären Operator $\hat{U}$, der auf den Zustand $|x\rangle$ folgendermaßen wirken soll (wobei der größte gemeinsame Teiler von N und q eins ist, dies garantiert die Existenz der Einheit in den mod N Rechnungen):

$$\hat{U}\,|x\rangle = |xq \bmod \mathrm{N}\,\rangle$$

Der Operator dreht den Zyklus sozusagen eine Stufe weiter. Wenn der Operator k mal (k sei die Ordnung von a mod N) angewendet wird, so muss

der Zustand der reingesteckt wird wieder exakt der gleiche sein, der nach der Anwendung des Operators rauskommt, denn $\hat{U}^k = I$. Daher lässt sich folgende wichtige Aussage über die Eigenwerte des Operators $\hat{U}$ treffen:

$$\hat{U}\,|u_s\rangle = \lambda_s\,|u_s\rangle \Rightarrow \lambda_s^k = 1 \Leftrightarrow \lambda_s = e^{i2\pi s/k}$$

Die Eigenvektoren zu diesem Operator lassen sich sogar explizit konstruieren. Intuitiv scheint es sinnvoll, dass sich die Eigenzustände aus den Fourierzuständen rekrutieren. Die Situation ist analog zu einem eindimensionalen Gitter mit periodischen Randbedingungen. Nach ein wenig Nachdenken lassen sich die Eigenzustände hinschreiben:

$$|u_s\rangle := \frac{1}{\sqrt{k}} \sum_{l=0}^{k-1} e^{-i\frac{2\pi}{k}sl}\,|x^l \bmod \mathrm{N}\,\rangle$$

Wie erwartet sind die Eigenwerte $\lambda_s = e^{\frac{2\pi}{k}s}$.

Ein zentraler Schritt des Shor-Algorithmus besteht darin, diese Eigenwerte zu bestimmen. Hat man die Eigenwerte nämlich rausgefunden, so lässt sich damit die Ordnung k mit etwas weiterem Aufwand bestimmen (Continued Fraction Methode).

2 Die diskrete Fouriertransformation

Gegeben seien n Werte $(x_0, x_2, ..., x_{n-1})$. Die Fouriertransformation erzeugt aus diesen Werten umkehrbar eindeutig einen neuen Vektor $(y_0, y_2, ..., y_{n-1})$ nach der Formel:

$$y_k = \frac{1}{\sqrt{n}} \sum_{l=0}^{n-1} e^{i\frac{2\pi}{n}lk} x_l$$

Dies lässt sich auch einfacher in Matrixschreibweise darstellen, wenn man die Matrix F definiert als

$$(F)_{kl} := \frac{1}{\sqrt{n}} e^{i\frac{2\pi}{n}(l-1)(k-1)}$$

Dann lautet die diskrete Fouriertransformation (in Matrixschreibweise, x und y als Vektor gelesen):

$$y = Fx$$

Die Matrix F ist unitär:

$$\sum_{l=1}^{n}(F)_{kl}(F^{\dagger})_{lm} = \frac{1}{n}\sum_{l=1}^{n} e^{i\frac{2\pi}{n}(k-1)(l-1)}e^{-i\frac{2\pi}{n}(l-1)(m-1)} =$$

$$\frac{1}{n}\sum_{l=1}^{n}(e^{i\frac{2\pi}{n}(k-m)})^{(l-1)} = \begin{cases} 1 & k=m; \\ \frac{1}{n}\frac{1-(e^{i\frac{2\pi}{n}(k-m)})^{n}}{1-(e^{i\frac{2\pi}{n}(k-m)})} = 0 & k \neq m. \end{cases}$$

3 Diskrete Fouriertransformation für Qubits

Angenommen, wir hätten ein System von n Qubits gegeben. Um eine unitäre Transformation eindeutig festzulegen, genügt es, die Wirkung auf die reinen Produktzustände anzugeben (ein solcher Zustand für n=4 Qubits wäre z.B.: $|0\rangle \otimes |1\rangle \otimes |0\rangle \otimes |1\rangle := |5\rangle$, oder auch $|0\rangle \otimes |0\rangle \otimes |1\rangle \otimes |0\rangle := |2\rangle$): Man definiert folgende unitäre Operation (Wirkung auf einen Basiszustand $|j\rangle$):

$$|j\rangle \mapsto \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1} e^{i\frac{2\pi}{n}jk}\,|k\rangle$$

Nach einer kleinen Rechnung sieht man, welche Wirkung dieser unitäre Operator auf einen beliebigen Zustand $|\Psi\rangle$ hat:

$$\sum_{j=0}^{N-1} x_j\,|j\rangle \mapsto \sum_{j=0}^{N-1} x_j \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^{i\frac{2\pi}{N}jk}\,|k\rangle =$$

$$\sum_{k=0}^{N-1}\frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} x_j e^{i\frac{2\pi}{N}jk}\,|k\rangle = \sum_{k=0}^{N-1} y_k\,|k\rangle$$

Die oben definierte unitäre Transformation wirkt also einfach als diskrete Fouriertransformation auf die Vorfaktoren der Basiszustände $|0\rangle, |1\rangle, ..., |N-1\rangle$.

Wenden wir uns nun dem Problem zu, die Fouriertransformation als eine Schaltung von quantenmechanischen Gattern durchzuführen, wobei die Gatter jeweils nur auf 2 Teilchen wirken. Hierfür ist es sinnvoll, in der Dualdarstellung zu arbeiten:

$$j = \sum_{l=1}^{n} j_l \cdot 2^{n-l}$$

Für n Qubits gibt es $N = 2^n$ verschiedene Produktzustände. Damit gilt:

$$|j\rangle = |j_1\rangle \otimes |j_2\rangle \otimes \cdots \otimes |j_n\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{i\frac{2\pi}{2^n}jk} |k\rangle =$$

$$\frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} \cdots \sum_{k_n=0}^{1} e^{i2\pi j(\sum_{l=1}^{n} k_l 2^{-l})} |k_1\rangle \otimes |k_2\rangle \otimes \cdots \otimes |k_n\rangle =$$

$$\frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} \cdots \sum_{k_n=0}^{1} \prod_{l=1}^{n} e^{i2\pi j(k_l 2^{-l})} |k_1\rangle \otimes |k_2\rangle \otimes \cdots \otimes |k_n\rangle =$$

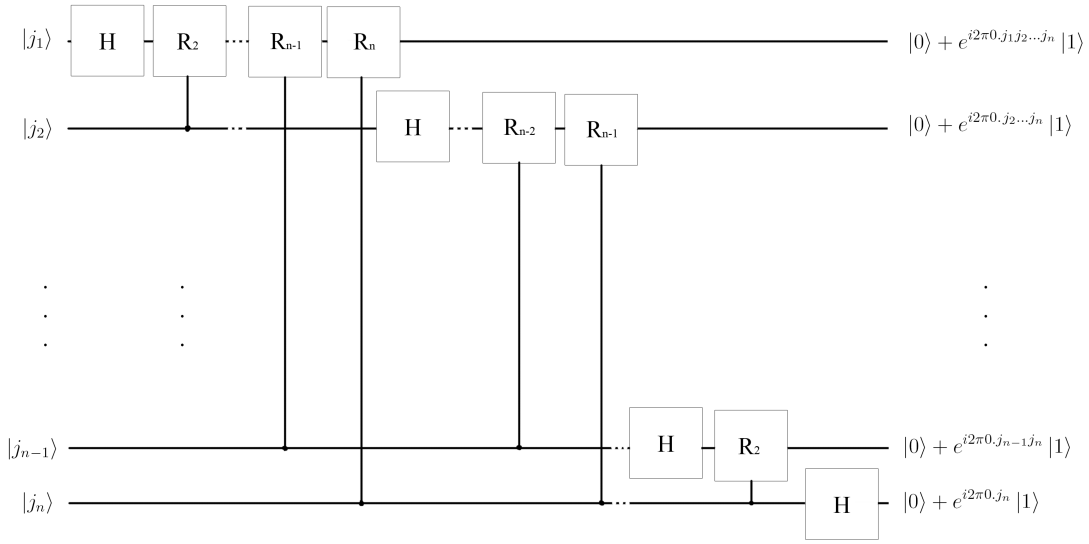
$$\frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} [\sum_{k_l=0}^{1} e^{i2\pi j(k_l 2^{-l})} |k_l\rangle] =$$

$$\frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} [|0\rangle + e^{i2\pi j 2^{-l}} |1\rangle] =$$

$$\frac{1}{\sqrt{2^n}} [|0\rangle + e^{i2\pi 0.j_n} |1\rangle] \otimes [|0\rangle + e^{i2\pi 0.j_{n-1}j_n} |1\rangle] \otimes \cdots \otimes [|0\rangle + e^{i2\pi 0.j_1 j_2 \ldots j_n} |1\rangle]$$

Zur Erläuterung, wie $0.j_1j_2...j_n$ zu verstehen ist: gemeint ist hier ebenfalls die Dualdarstellung, die die Notation etwas vereinfacht. Ein kurzes Beispiel sagt mehr als lange Erklärungen: $0.101 = 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}$

Das Gatter, das die Fouriertransformation vornimmt sieht dann folgendermaßen aus:



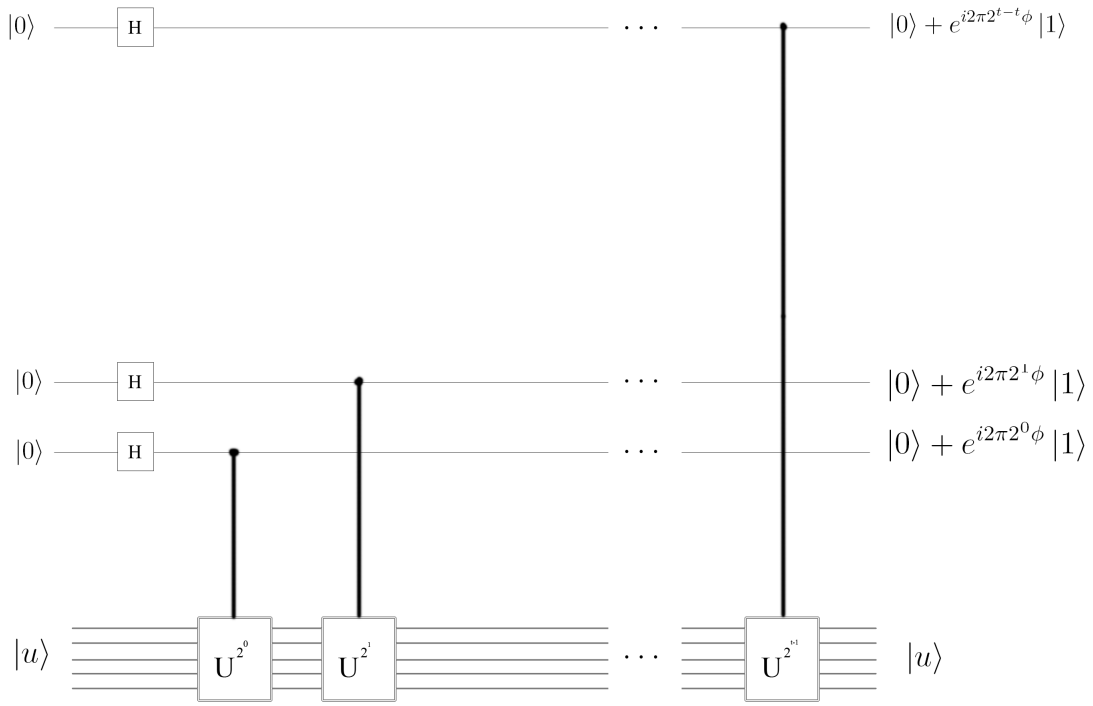
Die unitäre Transformation R_k ist dabei folgendermaßen definiert:

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$$

Das Gatter versucht man am besten komponentenweise zu verstehen: Das Hadamard-Gate, das auf das unterste Gate wirkt, erzeugt genau das 1. Qubit der Fouriertransformierten. Wenn nämlich $|j_n\rangle$ auf 0 gesetzt ist, erzeugt es den Zustand $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, im anderen Fall den Zustand $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Man überlegt sich leicht, dass die Exponentialfunktion im 1. Qubit der Fouriertransformierten 1 ist für den Fall, dass $j_n = 0$ und -1 für den Fall dass $j_n = 1$. Damit haben wir also den gewünschten Zustand des 1. Qubits. (In obiger Skizze sind der Übersichtlichkeit halber die Vorfaktoren $\frac{1}{\sqrt{2}}$ weggelassen worden. Wichtig ist, dass man das Hadamard Gate als letztes anwendet, um die Information in j_n vorher noch für die anderen Qubits zu nutzen. (Die volle Information steckt natürlich auch noch nach Anwendung des Hadamard Gates in dem Zustand, nur ist sie nicht mehr so einfah zu nutzen wie bei obigem Gate) Die weiteren Operationen um die anderen Qubits zu erzeugen erklären sich beinahe von selbst. Die unitäre Operation R_k setzt immer noch einen weiteren Phasenfaktor vor den Zustand $|1\rangle$. Die kontrollierte Operation sorgt dafür, dass dann mit keinem Phasenfaktor multipliziert wird, wenn das betreffende Kontrollbit 0 ist. Um tatsächlich den gewünschten Zustand zu erzeugen fehlt allerdings noch ein weiterer Schritt. Bedingt durchh obiges Problem, erst das letzte Qubit zu erzeugen, liegen die Qubits am Ausgang in umgekehrter Reihenfolge vor. Es ist also noch eine Vertauschung der Reihenfolge der Qubits nötig, ein sogenannter Bit-swap.

4 Phase Estimation

Der Algorithmus der im Folgenden vorgestellt wird, dient dazu, die Eigenwerte eines unitäten Operators $\hat{U}$ zu bestimmen. Zunächst setzen wir voraus, dass wir einen Eigenzustand $|u\rangle$ des Operators gegeben haben, den wir in den Algorithmus einsetzen können. Der unbekannte Eigenwert von $|u\rangle$ sei $e^{2\pi i\phi}$. $|u\rangle$ selber wird in dem Algorithmus unverändert bleiben, der "'Speicher"', auf den wir schreiben besteht aus den t qubits die anfangs im Zustand $|0\rangle$ sind.



Dieses Gatter führt auf den neuen Zustand des "'Speichers"':

$$\frac{1}{\sqrt{2^t}}[|0\rangle + e^{i2\pi 2^{t-1}\phi}|1\rangle] \otimes [|0\rangle + e^{i2\pi 2^{t-2}\phi}|1\rangle] \otimes \cdots \otimes [|0\rangle + e^{i2\pi 2^{0}\phi}|1\rangle]$$

Dieser Zustand hat große Ähnlichkeit mit jenem, den wir aus der Fouriertransformation erhalten hatten. Lässt sich nämlich ϕ in der Dualdarstellung mit bis zu t Nachkommastellen exakt ausdrücken ($\phi = 0.\phi_1\phi_2...\phi_t$), so haben wir:

$$\frac{1}{\sqrt{2^t}}[|0\rangle + e^{i2\pi 0.\phi_t}|1\rangle] \otimes [|0\rangle + e^{i2\pi 0.\phi_{t-1}\phi_t}|1\rangle] \otimes \cdots \otimes [|0\rangle + e^{i2\pi 0.\phi_1\phi_2...\phi_t}|1\rangle]$$

Das ist in diesem Fall genau der Zustand aus der Fouriertransformation. D.h. wenn wir die inverse Transformation anwenden erhalten wir exakt den Zustand (da das Gatter reversibel ist, braucht man es gedanklich lediglich von rechts nach links zu durchlaufen und erhält dann ein Gatter, das die inverse Fouriertransformation vornimmt)

$$|\phi\rangle \otimes |u\rangle$$

Und die Messung der ersten t qubits ergibt den gesuchten Wert ϕ. Nun ist in der Eigenwert in den allermeisten Fällen nicht mit t qubits genau

darstellbar. Man kann allerdings zeigen, dass man dann dennoch einen guten Schätzwert erhält, der sich dem wahren Wert beliebig genau annähert, wenn man die Anzahl der qubits t erhöht.

Ein Problem beim Shor Algorithmus ist, dass man den Eigenzustand $|u\rangle$, nicht zur Verfügung hat, denn dieser würde bereits die Kenntnis der Ordnung voraussetzen. Daher setzt man anstelle von $|u\rangle$ den Zustand $|1\rangle$ ein. Hierfür gilt:

$$|1\rangle = \frac{1}{\sqrt{k}} \sum_{s=0}^{k-1} |u_s\rangle$$

Man erhält daher als Messergebnis einen Zufälligen der k verschiedenen Eigenwerte. Dies ist aber immer noch ausreichend, um die Ordnung k effizient bestimmen zu können.

5 Zusammenfassung

Nachdem wir nun die einzelnen Schritte des Shor-Algorithmus in groben Zügen besprochen haben, nun nocheinmal der gesamte Algorithmus in der Zusammenfassung:
Gegeben sei eine Zahl N, gesucht ist ein nichttrivialer Faktor.

- 1. Schritt: Prüfe ob N gerade ist, falls ja → Faktor ist 2.
- 2. Schritt: Prüfe ob N die Potenz einer Primzahl ist. Wenn ja bestimme diese mit einem effizienten klassischen Algorithmus und liefere die Primzahl als Faktor.
- 3. Schritt: wähle eine zufällige Zahl $1 < q < N$ und prüfe ob der größte gemeinsame Teiler 1 ist. Falls der ggt größer als 1 ist liefere diese Zahl als Faktor zurück
- 4. Schritt: Benutze den quantenmechanischen Algorithmus, um die Ordnung k von q mod N zu bestimmen
- 5. Schritt: Falls k gerade ist, dann prüfe, ob der größte gemeinsame Teiler von $q^{\frac{k}{2}} - 1$ und N einen nichttrivialen Faktor liefert. Falls ja, so liefere ihn zurück. Falls nein, so beginne wieder bei Schritt 3

6 Literatur

Michael A. Nielsen and Isaac L. Chuang: Quantum Computation and Information

Meyberg, Vachenauer: Höhere Mathematik I

Artur Ekert and Richard Jozsa: Quantum Computation and Shor's factoring algorithm

A. Galindo and M.A. Martin-Delgado: Classical and quantum information

From Factoring to Phase Estimation, Los Alamos Science Number 27, 2002