Die Überlegenheit von Quantenalgorithmen
Lektion 4 – Der Grover-Algorithmus

Stell dir vor, du möchtest einen Safe mit einem Zahlenschloss öffnen, das aus vier Ziffern von 0 bis 9 besteht. Du hast keine Ahnung, welche Zahlen richtig sind. Wenn du sehr viel Glück hast, triffst du beim ersten Versuch die richtige Kombination. Wie viele Versuche würdest du im schlimmsten Fall benötigen? Wie viele im Durchschnitt? Erinnere dich an die Suche in unstrukturierten Daten aus Lektion 1.

Das Suchproblem

In Lektion 1 suchten wir nach einer bestimmten Zahl. Diese kann als Nummer betrachtet werden, die mit 1 markiert ist, während alle anderen mit 0 markiert sind. Es gibt also eine Liste von N Zahlen, in der genau eine Zahl den Wert 1 (die richtige Zahl) hat, während alle anderen 0 sind. Die Aufgabe besteht darin, die Position der Zahl zu finden, die 1 ist – und zwar mit möglichst wenigen Abfragen.

Das Problem kann mathematisch als eine Funktion beschrieben werden, bei der genau ein r aus {0, …, N-1} existiert mit:

$$f(x) = \begin{cases} 1 & falls\ x = r \\ 0 & für\ alle\ anderen\ x \end{cases}$$

Klassische Lösung

Aufgabe 1: Klassische Lösung

Angenommen, du weißt nicht, was r ist. Bestimme die durchschnittliche Anzahl an Versuchen von x, die du in die Funktion f eingeben musst, um r zu finden. Welche Strategie würdest du verwenden?

Aufgabe 2: Binärdarstellung

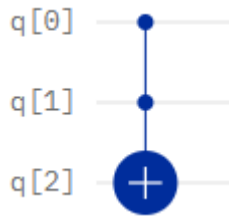
Wir verwenden die binäre Darstellung der Zahlen von 0 bis N-1, also $N = 2^n$ und r wird als Zeichenkette der Länge n aus 0en und 1en dargestellt.

Gib die Binärdarstellung von r = 5 als 4Bit-Binärzahl an. Wie groß ist N bei 4 Bit?

ENTNOMMEN AUS QUANTEN-COMPUTING IM MINT-UNTERRICHT
Ein Projekt von Science on Stage Deutschland e.V.

Implementierung der Suchfunktion mit Quantengattern

Um die Suchfunktion zu implementieren, müssen wir ein neues Gatter einführen, das sogenannte **Toffoli-Gatter**. Dieses ähnelt dem **CNOT-Gatter**, wirkt aber auf mehrere Eingangs-Qubits, zum Beispiel:



Das Toffoli-Gatter führt eine NOT-Operation auf dem Output-Qubit q[2] nur dann aus, wenn beide Input-Qubits (q[0] und q[1]) den Zustand $|1\rangle$ haben. Wenn das Output-Qubit anfangs im Zustand $|0\rangle$ ist, entspricht dies einer UND-Operation in einem klassischen Computer.

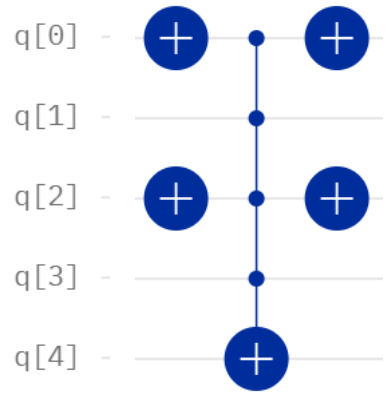
Die folgende Tabelle zeigt den finalen Zustand der Output-Qubits q[2] für alle Kombinationen der beiden Input-Qubits:

q[0]	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
q[1]	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
q[2] (final)	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$

Um die versteckte Funktion für die Grover-Suche zu konstruieren, verwenden wir eine Kombination aus einem n-Input-Toffoli-Gatter und NOT-Gattern.

Für die Implementation des Grover Algorithmus verwenden wir ein Beispiel mit 4 Qubits, wobei wir den versteckten String 0101 suchen. Um die Schaltung zu bauen, benötigen wir ein 4-Qubit-Toffoli-Gatter und NOT-Gatter für die Qubits, die der 0 in der versteckten Zeichenkette entsprechen. Diese NOT-Gatter kehren den Zustand dieser beiden Eingänge von $|0\rangle$ zu $|1\rangle$ um, so dass das Toffoli-Gatter den Zustand des Output-Qubits nur dann zu $|1\rangle$ ändert, wenn alle Eingänge der versteckten Zeichenkette entsprechen. Danach werden die Zustände wieder mit NOT-Gattern zurückgesetzt.

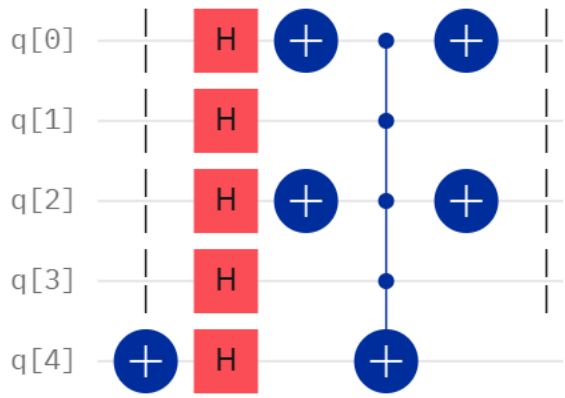
Gatter-Schaltung für die versteckte Zeichenkette 0101



Der Unterschied zwischen Grover-Algorithmus und Bernstein-Vazirani

In der vorherigen Lektion haben wir gesehen, dass die versteckte Funktion im Bernstein-Vazirani-Algorithmus in einem einzigen Schritt gefunden werden kann. Das lag daran, dass sie sich vollständig mit CNOT-Gattern aufbauen ließ, weshalb die Anwendung von Hadamard-Gattern sofort zur Lösung führte.

Der Grover-Algorithmus hingegen soll das Problem lösen, ob ein bestimmtes Datenelement in einer unstrukturierten Datenmenge existiert. Hier sind Toffoli-Gatter erforderlich, weshalb der Bernstein-Vazirani-Ansatz nicht funktioniert. Jedoch können wir wie beim Deutsch-Algorithmus und Bernstein-Vazirani zunächst alle Input-Qubits auf $|0\rangle$ setzen und das Output-Qubit auf $|1\rangle$. Anschließend werden auf alle Qubits Hadamard-Gatter angewendet:



Die folgende Tabelle zeigt die Auswirkungen auf die Input-Qubits durch diese Schaltung:

	States															
	0 |0000>	1 |0001>	2 |0010>	3 |0011>	4 |0100>	5 |0101>	6 |0110>	7 |0111>	8 |1000>	9 |1001>	10 |1010>	11 |1011>	12 |1100>	13 |1101>	14 |1110>	15 |1111>
	Amplitudes															
Hadamard	0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25	
Function	0.25		0.25		0.25		0.25		0.25		-0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25		0.25	

Die Anwendung der Hadamard-Gatter erzeugt eine Superposition aller 16 unterschiedlichen Zustände (alle Binärdarstellungen von 0 bis 15). Die Wahrscheinlichkeit für jeden Basiszustand ist 1/16 und die Ausgangsamplitude ist ¼. Dies ist derselbe Ausgangspunkt wie bei den vorherigen Algorithmen.

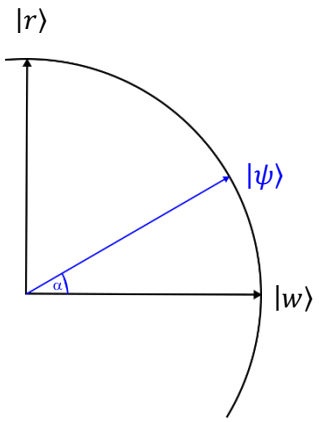
Die Anwendung der versteckten Funktion hat folgenden Effekt: Alle Koeffizienten der falschen Antworten bleiben gleich, während der Koeffizient der richtigen Antwort nun negativ ist. Dies allein wird uns nicht helfen, die versteckte Zeichenkette zu finden, da die Wahrscheinlichkeit (das Quadrat des Koeffizienten) bei einer Messung zu diesem Zeitpunkt noch gleich wäre.

Durch weitere Operationen auf diesen Superpositionszustand wird der Koeffizient (und damit die Wahrscheinlichkeit) der versteckten Zeichenkette verstärkt, während alle anderen abgeschwächt werden. Im Folgenden ist in einer geometrischen Darstellung gezeigt, wie dies umgesetzt wird.

Grover-Algorithmus: Die Quantencomputer-Lösung

Grovers Algorithmus löst unser Problem mit nur $\sqrt{N}$ Anfragen an die Funktion. Der Algorithmus ist in den folgenden Schritten implementiert:

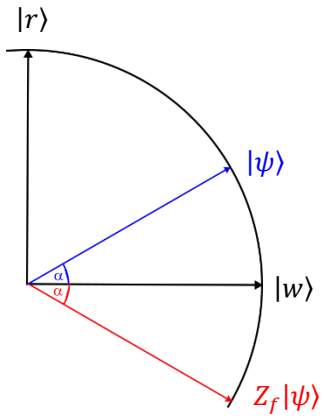
Schritt I: Erzeuge n Qubits im Zustand $|0\rangle$ und wende ein Hadamard-Gatter auf jedes Qubit an. Dies erzeugt eine Superposition aller möglichen Basiszustände von n Qubits.
Dieser Zustand $|\psi\rangle$ kann als Überlagerung des gesuchten (richtigen) Zustands $|r\rangle$ und der (normalisierten) Summe aller anderen (falschen) Zustände $|w\rangle$ geschrieben werden, wobei $|w\rangle$ orthogonal zu $|r\rangle$ ist.



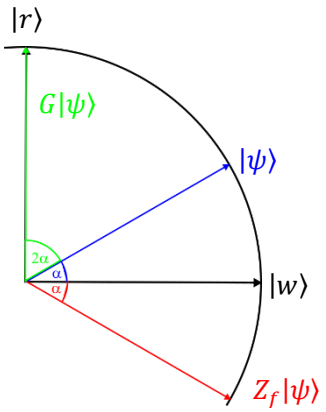
Beispiel:

Bei N = 4 erhalten wir $|\psi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$.
Nehmen wir an, dass die gesuchte Zahl r = 2 ≙ 10 ist, wird dies
als $|\psi\rangle = \frac{1}{2}|r\rangle + \frac{\sqrt{3}}{2}|w\rangle$ geschrieben
mit $|w\rangle = \frac{1}{\sqrt{3}}(|00\rangle + |01\rangle + |11\rangle)$.
Der Winkel α wird als Arkussinus des Koeffizienten des gesuchten Zustands berechnet: $\alpha = \sin^{-1}\left(\frac{1}{2}\right) = 30°$

Schritt II: Wende das Orakel-Gatter Z_f auf $|\psi\rangle$ an. Dieses Gatter führt genau eine Abfrage an die Funktion f aus.
Wie wir im vorherigen Abschnitt gesehen haben, wird der Koeffizient von $|r\rangle$ negativ, während der Koeffizient von $|w\rangle$ unverändert bleibt. Geometrisch bedeutet dies, dass der Zustandsvektor von $|\psi\rangle$ an dem Vektor $|w\rangle$ gespiegelt wird.



Schritt III: Spiegele den Zustandsvektor $Z_f|\psi\rangle$ am Vektor $|\psi\rangle$ (wie man in Aufgabe 5 sieht, wird dies durch eine Kombination mehrerer Gatter erzeugt). Die Kombination von Schritt II und Schritt III heißt Grover-Operation G. Insgesamt wird der resultierende Zustandsvektor um 2α, verglichen zum Ausgangszustand $|\psi\rangle$, in die Richtung des gesuchten Zustands rotiert.



Im Beispiel N = 4 haben wir den gesuchten Zustand mit genau einer Grover-Operation erreicht. Im Allgemeinen muss die Grover-Operation wiederholt werden wie im nächsten Schritt.

Schritt IV: Wiederhole die Grover-Operation (Schritt II und III) t mal (d. h. die Anzahl der Anfragen an f ist t), bis der Zustandsvektor fast parallel zu $|r\rangle$ ist und führe eine Messung auf allen n Qubits durch.

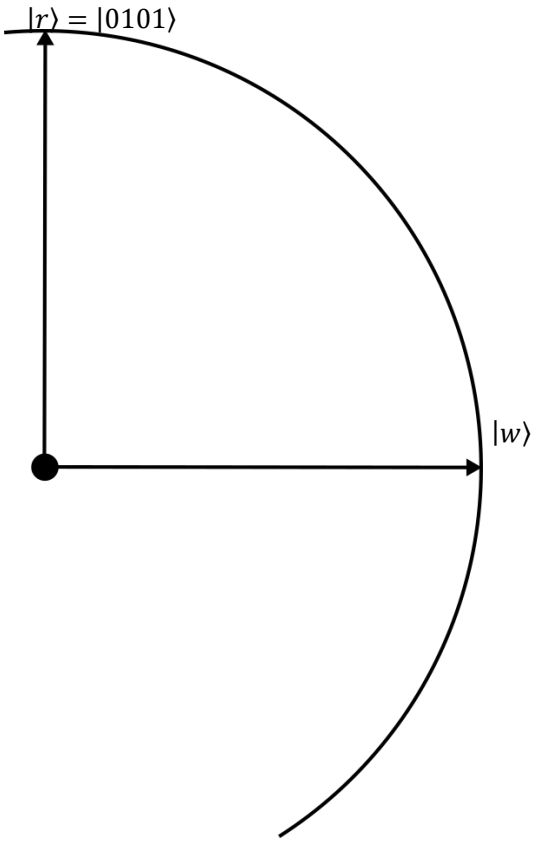
Aufgabe 3: Grovers Algorithmus Schritt für Schritt verstehen

Du wirst nun den Grover-Algorithmus schrittweise nachvollziehen und verstehen, wie oft die Grover-Operation wiederholt werden muss.

Wir verwenden $n = 4$, also $N = 2^4 = 16$. Die gesuchte Zahl ist $r = 5 \triangleq 0101$.

Schritt I: Anwenden der Hadamard-Gatter auf alle vier Qubits im Zustand $|0000\rangle$ ergibt $|\psi\rangle = \frac{1}{\sqrt{2^4}}(|0000\rangle + |0001\rangle + \ldots + |1111\rangle)$, bzw. $|\psi\rangle = \frac{1}{\sqrt{2^4}}|0101\rangle + \sqrt{\frac{15}{2^4}}|w\rangle$.

Berechne den Winkel α aus dem Koeffizienten $\frac{1}{\sqrt{2^4}} = \frac{1}{4}$ und zeichne den entsprechenden Vektor $|\psi\rangle$ in der Abbildung ein:



Schritt II: Zeichne den Zustandsvektor $Z_f|\psi\rangle$ durch Spiegelung von $|\psi\rangle$ an $|w\rangle$.

Schritt III: Zeichne den Zustandsvektor $G|\psi\rangle$ durch Spiegelung von $Z_f|\psi\rangle$ an $|\psi\rangle$.

Schritt IV: Wiederhole Schritt II und Schritt III (also $t = 2$)**.**

Aufgabe 4: Reflektiere deine Lösung

- Denkst du, dass zwei Wiederholungen der Grover-Operation ausreichend sind? Was würde passieren, wenn man dies häufiger wiederholen würde?

- Der Grover-Algorithmus findet die richtige Lösung nur mit einer hohen Wahrscheinlichkeit, aber nicht zwingend exakt. Erkläre dies anhand deiner Lösung.

- Wie ändert sich der Winkel α, wenn n erhöht wird? Was bedeutet dies für die notwendige Anzahl an Wiederholungen?

Schlussfolgerungen

Warum benötigt der Grover-Algorithmus nur $O(\sqrt{N})$ Schritte?

Allgemein ist der Winkel α gleich $\sin^{-1}\left(\frac{1}{\sqrt{N}}\right)$, was bei hohem N (und kleinem α) zu $\frac{1}{\sqrt{N}}$ angenähert werden kann.

Nach einer Grover-Operation ist der aktuelle Winkel 3α, nach der zweiten Grover-Operation 5α usw. Für t Grover-Operationen ergibt sich ein Ergebniswinkel von $(2t+1)\cdot\alpha$.

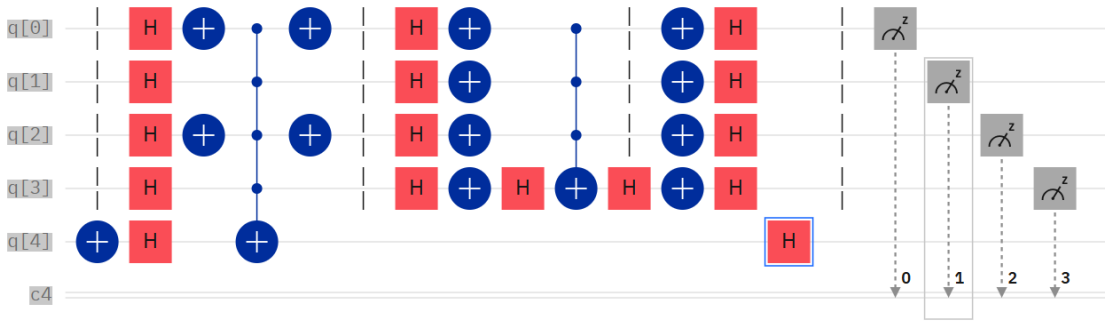
Um den richtigen Zustand mit einer hohen Wahrscheinlichkeit zu finden, sollte dieser Winkel nahe $\frac{\pi}{2}$ sein.
Mit $(2t+1)\cdot\frac{1}{\sqrt{N}} \approx \frac{\pi}{2}$ erhält man $t \approx \frac{1}{2}\cdot\left(\sqrt{N}\cdot\frac{\pi}{2}-1\right)$, welches die optimale Anzahl an Grover-Operationen ist.

Wie kann ich sichergehen, die richtige Antwort gefunden zu haben?

Der Grover-Algorithmus liefert das richtige Ergebnis mit hoher Wahrscheinlichkeit. Bei den meisten Zahlen besteht aber auch eine kleine Wahrscheinlichkeit für ein falsches Ergebnis. Man kann das Resultat leicht prüfen, indem man es ins Orakel einsetzt. Falls es nicht stimmen sollte, wiederholt man den Algorithmus – und überprüft erneut.

Aufgabe 5: Überprüfe den Grover-Algorithmus im Quantum Composer

Die Schaltung zeigt die 4-Bit-Gover-Operation mit dem gesuchten Zustand $|r\rangle = |0101\rangle$.



Präparation | Orakel | Spiegelung an $|\psi\rangle$

- Implementiere die Schaltung im Quantum Composer.
- Die Schaltung implementiert nur eine Anwendung der Grover-Operation. Wiederhole die Grover-Operation und vergleiche das Ergebnis des Composers mit deinem Ergebnis aus Aufgabe 3.