Heterogeneous Computing

Grover's Algorithmus

Abschlussprojekt

Universität Trier
FB IV - Informatikwissenschaften
Professur für Systemsoftware und Verteilte Systeme

Semester: SoSe 2024

Name und Matrikelnummer:

Simon Szulik, 1474315

Inhaltsverzeichnis

1. Suchprobleme

Suchprobleme sind grundlegende Herausforderungen in der Informatik, bei denen es darum geht, ein bestimmtes Element innerhalb einer Menge von Daten zu finden. Formal lässt sich ein Suchproblem als eine Funktion $f : \Sigma^n \to \{0, 1\}$ definieren, wobei Σ ein Alphabet und n die Länge der Eingabe bezeichnet. Die Funktion $f(x)$ bewertet, ob eine gegebene Eingabe $x \in \Sigma^n$ eine bestimmte Bedingung erfüllt, wobei $f(x) = 1$ bedeutet, dass x die Bedingung erfüllt (also eine Lösung des Problems darstellt), und $f(x) = 0$, dass dies nicht der Fall ist. Das Ziel eines Suchproblems besteht somit darin, ein $x \in \Sigma^n$ zu finden, für das $f(x) = 1$ gilt.

1.1 Unstrukturierte Suche

Suchprobleme treten in vielen verschiedenen Kontexten auf. Ein einfaches Beispiel ist die Suche nach einem bestimmten Element in einer Liste, bei der die Funktion $f(x)$ überprüft, ob x das gesuchte Element ist. Ein weiteres Beispiel ist das Lösen eines logischen Rätsels, bei dem $f(x) = 1$ bedeutet, dass die Lösung x alle logischen Bedingungen des Rätsels erfüllt.
Suchprobleme lassen sich in zwei Hauptkategorien unterteilen: strukturierte und unstrukturierte. Bei strukturierten Suchproblemen ist die Datenmenge so organisiert, dass die Suche effizienter abläuft, zum Beispiel durch Sortierung oder Indexierung. Bei unstrukturierten Suchproblemen hingegen gibt es keine zusätzliche Information über die Ordnung oder Struktur der Datenmenge, was die Suche erheblich erschwert, sodass im schlimmsten Fall jedes Element der Datenmenge überprüft werden, um das gesuchte Element zu finden. Dies führt bei klassischen Algorithmen zu einer linearen Zeitkomplexität von $O(N)$, wobei N die Anzahl der Elemente in der Datenmenge ist. Im Durchschnitt sind $N/2$ Vergleiche nötig, um das gesuchte Element zu finden.

2. Darstellung von Informationen

In der Quanteninformatik, wie auch in der klassischen Informatik, ist die Darstellung von Informationen von zentraler Bedeutung. Während klassische Computer Informationen in Form von Bits speichern und verarbeiten, die entweder den Zustand 0 oder 1 annehmen können, verwendet das Quantencomputing sogenannte Qubits. Diese Qubits basieren auf den Prinzipien der Quantenmechanik und können sich in einer Überlagerung von Zuständen befinden. Dadurch können sie gleichzeitig die Zustände 0 und 1 einnehmen, was eine wesentliche Grundlage für die potenziell überlegene Rechenleistung von Quantencomputern darstellt.

2.1 Qubits

Ein Qubit kann als ein zweidimensionaler Vektor im sogenannten Hilbertraum dargestellt werden, einem abstrakten mathematischen Raum, der in der Quantenmechanik zur Beschreibung von Zuständen verwendet wird. Der Zustand eines Qubits wird in der Quantenmechanik üblicherweise durch die sogenannte Bracket-Schreibweise ausgedrückt. Ein allgemeiner Zustand eines Qubits $|\psi\rangle$ wird als Linearkombination der Basiszustände $|0\rangle$ und $|1\rangle$ beschrieben:

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

Hierbei sind α und β komplexe Zahlen, die die Wahrscheinlichkeit für die Messung des Qubits in den Zuständen $|0\rangle$ bzw. $|1\rangle$ repräsentieren. Diese Wahrscheinlichkeiten sind gegeben durch $|\alpha|^2$ und $|\beta|^2$, wobei die Normierungsvorschrift $|\alpha|^2 + |\beta|^2 = 1$ sicherstellt, dass die Gesamtwahrscheinlichkeit 1 beträgt.

2.2 Bracket-Schreibweise (Dirac-Notation)

Die Bracket-Schreibweise ist eine kompakte und leistungsfähige Methode, um Zustände in der Quantenmechanik darzustellen, die insbesondere die Berechnung von inneren und äußeren Produkten vereinfacht [KaLM07]. Es gibt zwei grundlegende Elemente dieser Notation:

- **Ket-Vektoren (Ket-Schreibweise)**: Ein Zustand wird als „Ket“ dargestellt, z.B. $|0\rangle$, $|1\rangle$ oder $|\psi\rangle$. Die Kets repräsentieren Vektoren im Hilbertraum, in dem die Quanteninformation codiert ist. Für ein Qubit entspricht $|0\rangle$ dem Zustand $\begin{pmatrix}1\\0\end{pmatrix}$ und $|1\rangle$ dem Zustand $\begin{pmatrix}0\\1\end{pmatrix}$.

- **Bra-Vektoren (Bra-Schreibweise)**: Zu jedem Ket-Vektor $|\psi\rangle$ gibt es einen zugehörigen Bra-Vektor $\langle\psi|$. Der Bra-Vektor ist das adjungierte (komplex konjugierte und transponierte) Pendant zum Ket-Vektor. Zum Beispiel ist zu $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ der Bra-Vektor $\langle\psi| = \alpha^*\langle 0| + \beta^*\langle 1|$.

Diese Notation erlaubt es, innere Produkte und äußere Produkte auf einfache Weise darzustellen:

- **Inneres Produkt**: Das innere Produkt von zwei Zuständen $|\psi\rangle$ und $|\phi\rangle$ wird als $\langle\psi|\phi\rangle$ geschrieben. Es handelt sich hierbei um einen Skalar, der die Ähnlichkeit der beiden Zustände ausdrückt.
- **Äußeres Produkt**: Das äußere Produkt von $|\psi\rangle$ und $\langle\phi|$ ergibt einen Operator, der auf Zustände im Hilbertraum wirkt und wird als $|\psi\rangle\langle\phi|$ dargestellt. Dieser Operator projiziert auf den Zustand $|\psi\rangle$ oder transformiert Zustände entsprechend der Eigenschaften von $|\phi\rangle$.

In Systemen mit mehr als einem Qubit wird der Gesamtzustand des Systems durch das Tensorprodukt der Zustände der einzelnen Qubits beschrieben. Beispielsweise ist der Zustand eines Zwei-Qubit-Systems, bei dem das erste Qubit im Zustand $|0\rangle$ und das zweite im Zustand $|1\rangle$ ist, durch das Tensorprodukt $|0\rangle \otimes |1\rangle$ beschrieben, oft auch kurz als $|01\rangle$ notiert. Dieses Tensorprodukt wird auch als Kronecker-Produkt bezeichnet und ist entscheidend für die Darstellung komplexer Quantenstates.

3. Grover's Algorithmus

Der Grover-Algorithmus bietet eine Methode zur Lösung unstrukturierter Suchprobleme, indem er die quantenmechanische Gatter nutzt, um die Suche signifikant zu beschleunigen [Nann20]. Durch die Verwendung von Qubits, die sich in einer Superposition von Zuständen befinden können, ist der Grover-Algorithmus in der Lage, gleichzeitig eine Vielzahl von möglichen Lösungen zu untersuchen. Der Algorithmus basiert auf einer iterativen Verstärkung der Amplitude des gesuchten Elements, sodass dieses mit hoher Wahrscheinlichkeit nach einer quadratisch reduzierten Anzahl von Schritten gefunden wird. Insgesamt baut er sich aus den folgenden drei zentralen Schritten auf: der Vorbereitung des Zustands, dem Orakel und dem Diffusionsoperator.
In der Zustandsvorbereitung wird der Suchraum erstellt, der alle möglichen Fälle umfasst, die als Antwort in Frage kommen könnten. In unserem oben erwähnten Beispiel mit der Liste würde der Suchraum alle Elemente dieser Liste abdecken. Das Orakel ist der Teil des Algorithmus, der die richtige Antwort oder die gesuchten Antworten durch Phaseninversion markiert. Es führt eine gezielte Inversion der Phase des gesuchten Elements durch, wodurch es sich von allen anderen unterscheidet. Der Diffusionsoperator, verstärkt diese markierten Antworten durch eine Inversion um den Mittelwert, sodass sie am Ende des Algorithmus deutlich hervortreten und messbar sind. Nach etwa $\sqrt{N}$ Iterationen, wobei N die Anzahl der möglichen Lösungen ist, erreicht der Grover-Algorithmus die höchste Wahrscheinlichkeit, das gesuchte Element beim Messen der Zustände zu identifizieren.

3.1 Vorgehensweise

Sei im folgenden L eine unsortierte Liste von N Elementen und ω unser gesuchtes Element. Bevor wir die Liste der Elemente betrachten, wissen wir nicht, wo sich das gesuchte Element befindet. Daher ist jede Vermutung über seinen Position so gut wie jede andere, was durch eine gleichmäßige Superposition ausgedrückt werden kann:

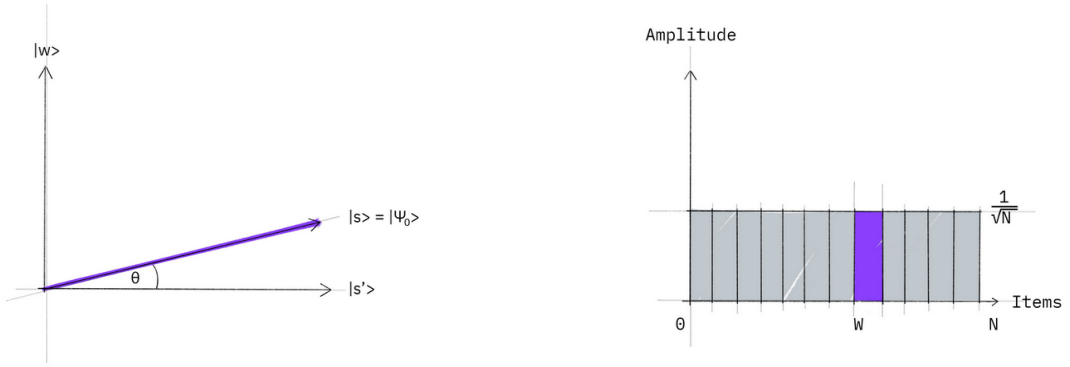
$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle.$$

Wenn wir zu diesem Zeitpunkt in der Standardbasis $\{|x\rangle\}$ messen würden, würde die Superposition mit gleicher Wahrscheinlichkeit auf einen der Basiszustände zurückfallen, was bedeutet, dass die Chance, den richtigen Wert ω zu finden $\frac{1}{N}$ beträgt. Da N in der Regel groß ist, entspricht dies einer Wahrscheinlichkeit von $\frac{1}{2^n}$, wobei n die Anzahl der Qubits ist. Im Durchschnitt müssen wir also etwa $N/2 = 2^{n-1}$ Versuche

unternehmen, um das gesuchte Element zu finden. Hier kommt die sogenannte Amplitudenverstärkung ins Spiel, eine Technik, die einem Quantencomputer hilft, die Wahrscheinlichkeit, das richtige Element zu finden, erheblich zu steigern. Dabei wird die Amplitude des gesuchten Elements verstärkt, während die der anderen Elemente reduziert wird. Dadurch ist die Wahrscheinlichkeit, beim Messen des Endzustands das richtige Element zu erhalten, nahezu sicher.
Dieser Algorithmus lässt sich geometrisch interpretieren indem man ihn als eine Rotation in der durch den Suchzustand und den Zielzustand aufgespannten zweidimensionalen Ebene betrachtet, wobei zwei Reflexionen nacheinander angewendet werden, um die Amplitude des Zielzustands zu verstärken. Die beiden speziellen Zustände, die dabei eine Rolle spielen, sind der Gewinnerzustand $|w\rangle$ und die gleichmäßige Superposition $|s\rangle$. Diese beiden Vektoren spannen eine zweidimensionale Ebene im Vektorraum $\mathbb{C}^N$ auf. Da $|w\rangle$ in der Superposition mit der Amplitude $N^{-1/2}$ vorkommt, sind die beiden Vektoren nicht exakt orthogonal. Dennoch kann ein weiterer Zustand $|s'\rangle$ definiert werden, der in der durch diese beiden Vektoren aufgespannten Ebene liegt und senkrecht zu $|w\rangle$ steht. Dieser Zustand entsteht aus $|s\rangle$, indem $|w\rangle$ entfernt und der verbleibende Vektor neu skaliert wird.

3.1.1 Schritt 1: Superposition

Das Verfahren der Amplitudenverstärkung beginnt mit der gleichmäßigen Superposition $|s\rangle$, die einfach durch $|s\rangle = H^{\otimes n}|0\rangle^n$ oder durch die Verwendung anderer symmetrischer, verschränkter Zustände erzeugt werden kann.



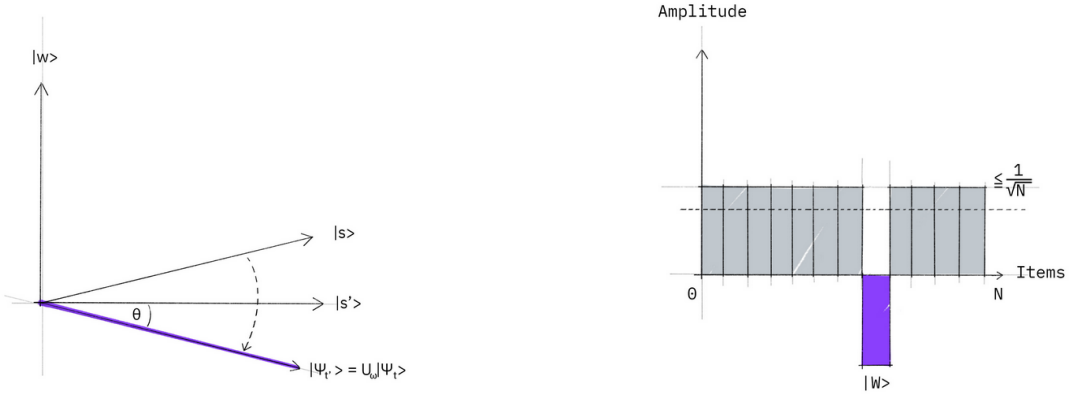
(a) Zustände in geografischer Lage (b) Amplitude und Wahrscheinlichkeiten

Abbildung 3.1: Startzustand des Algorithmus [Casa23]

Die linke Grafik in 3.1 entspricht der zweidimensionalen Ebene, die von den senkrechten Vektoren $|w\rangle$ und $|s'\rangle$ aufgespannt wird. Dadurch lässt sich der Anfangszustand als $|s\rangle = \sin(\theta)|w\rangle + \cos(\theta)|s'\rangle$ ausdrücken, wobei $\theta = \arcsin(\langle s|w\rangle) = \arcsin\left(\frac{1}{\sqrt{N}}\right)$ ist. Die rechte Grafik zeigt ein Balkendiagramm der Amplituden des Zustands $|s\rangle$.

3.1.2 Schritt 2: Reflexion ins Negative

Im nächsten Schritt wenden wir die Reflexion U_f auf den Zustand $|s\rangle$ an, was geometrisch einer Spiegelung des Zustands $|s\rangle$ um $|s'\rangle$ entspricht. Diese Transformation bewirkt, dass die Amplitude vor dem Zustand $|w\rangle$ negativ wird. Ein Nebeneffekt der Verringerung von $|w\rangle$ ist, dass die durchschnittliche Amplitude sinkt. Dies ist zu erkennen an der gestrichelten Linie in Plot 3.2b

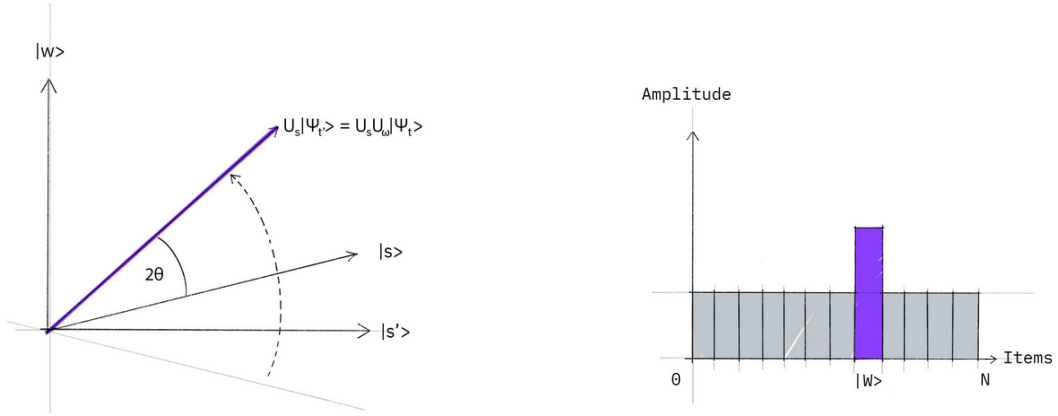


(a) Zustände in geografischer Lage (b) Amplitude und Wahrscheinlichkeiten

Abbildung 3.2: Zustand nach der Reflexion [Casa23]

3.1.3 Schritt 3: Rück-Reflexion

Mit Hilfe einer weiteren Spiegelung (U_s) um den Zustand $|s\rangle$, wie folgt notiert: $U_s = 2|s\rangle\langle s| - 1$, erhalten wir den Zustand $U_s U_f |s\rangle$ und beenden somit die Transformation und eine Grover-Iteration.



(a) Zustände in geografischer Lage (b) Amplitude und Wahrscheinlichkeiten

Abbildung 3.3: Zustand nach der Rück-Reflexion [Casa23]

Die beiden Reflexionen entsprechen dabei einer Rotation. Die Transformation $U_s U_f$ rotiert den Anfangszustand $|s\rangle$ näher an den Gewinnerzustand $|w\rangle$. Im Amplitudenbardiagramm kann die Wirkung der Reflexion U_s als Spiegelung um die durchschnittliche Amplitude verstanden werden. Da die durchschnittliche Amplitude durch die

erste Reflexion verringert wurde, verstärkt diese Transformation die negative Amplitude von $|w\rangle$ auf etwa das Dreifache ihres ursprünglichen Wertes, während sie die anderen Amplituden verringert. Die beiden Reflexionen ab dem zweiten Schritt lassen sich mehrfach wiederholen, bis man möglichst den Gewinnerzustand erreicht.

3.2 Orakel Beispiel

Der erste Schritt im Grover-Algorithmus ist die Vorbereitung des Startzustands. Wie bereits erwähnt, umfasst der Suchraum alle möglichen Werte, die wir durchsuchen müssen, um die gewünschte Antwort zu finden. Wie im obigen Kapitel besteht unsere 'Datenbank' aus allen möglichen rechnerischen Basiszuständen, in denen sich unsere Qubits befinden können. Wenn wir beispielsweise 3 Qubits haben, besteht unsere Liste aus den Zuständen $|000\rangle, |001\rangle, \ldots |111\rangle$ (d.h. die Zustände $|0\rangle \rightarrow |7\rangle$). In diesem Fall beträgt die Größe unseres Suchraums $N = 2^3 = 8$.
Der zweite und wichtigste Schritt im Grover-Algorithmus ist das Orakel. Orakel fügen den Lösungszuständen eine negative Phase hinzu, damit sie sich von den anderen abheben und gemessen werden können. Das bedeutet, dass für jeden Zustand $|x\rangle$ in der rechnerischen Basis gilt:

$$U_\omega|x\rangle = \begin{cases} |x\rangle & \text{wenn } x \neq \omega \\ -|x\rangle & \text{wenn } x = \omega \end{cases}$$

Dieses Orakel wird eine diagonale Matrix sein, bei der der Eintrag, der dem markierten Element entspricht, eine negative Phase aufweist. Wenn wir zum Beispiel drei Qubits haben und $\omega = 101$, sieht unsere Orakel-Matrix folgendermaßen aus:

$$U_\omega = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Literaturverzeichnis

[Casa23] P. A. M. Casares. Fault-tolerant quantum algorithms. 2023.

[KaLM07] P. Kaye, R. Laflamme und M. Mosca. *An Introduction to Quantum Computing*. Oxford University Press, Inc., USA. 2007.

[Nann20] G. Nannicini. An Introduction to Quantum Computing, Without the Physics, 2020.