Sabil, Novak, & Gould

Professor Dutt

Foundations of Quantum Computing

16 April 2024

Grover's Search Algorithm

Abstract

Grover's search algorithm is a quantum algorithm that uses amplitude amplification to increase the probability of finding the queried item. It is one of the best quantum algorithms to date providing a quadratic speed up to the best classical search algorithm. This is achieved by running the circuit through a series of gates until the correct item has reached its maximum probability. This algorithm works due to the Grover operator. The Grover operator consists of two parts, the oracle and the diffusion operator. These operators are made specific to the item you are searching for. The Grover operator is repeated an optimal number of times until the queried item reaches its maximum probability. This algorithm is extremely efficient, as it allows the runtime of a search to go from O(N) to O(√N) [1]. We were able to successfully code this algorithm to efficiently find solutions to boolean expressions.

Objectives

Our project aims to successfully implement Grover's algorithm into a quantum circuit to solve a three-variable Boolean satisfiability problem, more commonly known as 3-SAT. This application uses Grover's algorithm to identify solutions that satisfy any given Boolean statement. Although this program can be used for any number of variables, we will be focusing on three variable satisfiability problems for this project. Our ultimate goal is to demonstrate that this type of problem can be easily solved using Grover's algorithm and to discuss the limitations and advantages of the algorithm in other applications.

Background

Boolean satisfiability problems were proven to be NP-complete in 1997 by computer scientist Stephen Arthur Cook. This means that the time needed to find the solution to a boolean expression scales nonpolynomial to the number of possible inputs. [3] When you have a simple satisfiability problem with 3 boolean inputs, you only have a total of 8 incoming states that you must check to obtain a certain answer. However, as the amount of boolean input grows, the amount of total input states grows exponentially. This makes evaluating complex boolean expressions with many variables extremely taxing and practically impossible through classical

means. With Grover's Algorithm, iteration through each element individually is not necessary. Instead, the superior time complexity of $O(\sqrt{N})$ arises from the repeated step of amplitude amplification, where correct inputs are separated from incorrect inputs.

N-SAT problems generally require an exhaustive search in an unstructured database to find certain solutions. The term 'unstructured' tells us that the order of our bits does not inherently give us any clue about which input will yield a desired result. While they are 'unstructured', however, one can easily check if a given state is in our database. The full set of possible states will simply be 0 to 2^N-1 in binary. This range of possibilities means that our Grover's Search Algorithm will be conducted over an implicit database, which is far easier to understand and implement than an explicit database. According to Hayes, explicit databases generally refer to physical things and do not usually grow exponentially. The advantage of Grover's Algorithm will not be significant for such use cases because N is too low. [4]

Grover's Algorithm can be easily carried out today using publicly available Quantum Computers from companies like IBM. While Grover's algorithm can be used to speed up any algorithm that uses an exhaustive search, its real-world implementation for practical purposes has yet to be realized. Scalability remains a large problem when attempting to implement this algorithm. For many problems, the implementation and integration of Quantum technology is too burdensome and not fast enough to warrant its use above classical methods. However, If capacity increased and costs decreased, Grover's Algorithm could be used to scan over vast implicit sets of boolean input states and find useful solutions to practical complex problems. [4]

Nevertheless, Lov Grover's contributions to the fields of Quantum Computing and the importance of his algorithm cannot be overstated. The discovery and development of this algorithm effectively displayed to academia and the public that Quantum Computers can theoretically outperform the best supercomputers at specific tasks. Furthermore, Grover's method of Amplitude Amplification still finds its place as a subroutine in various other Quantum Algorithms which may find practical uses much sooner than Grover's.

Method

To demonstrate how Grover's Algorithm can be used to solve 3-SAT problems, we will be using various Python libraries to create all of the necessary components and solve a set of 3-SAT problems. We will then exhaustively search through all possible input states via classical means and ensure that Grover's Algorithm provides correct answers.

Firstly, we created our desired 3-SAT boolean expression. We then used the PhaseOracle function from Qiskit to create the quantum oracle needed for the algorithm. Following these steps, we moved on to complete the Grover Oracle. The Grover oracle consists of the aforementioned quantum oracle and the diffusion operator. The diffusion operator allows us to reverse the correct outputs from the oracle and find their initial input state. This is known as ' Amplitude Amplification'. It increases the probability of the key bit string while lessening the probabilities of the non-key bit strings. From there we created the circuit and ran the algorithm on it using the IBM Quantum Computing simulators.

We expected it to work because of how we created the quantum gates. It is useful to think of the process in terms of a quantum circuit diagram. Firstly, all of the N input bits are passed through Hadamard gates. This now gives each input bit a 50% chance of being measured as 0 or 1. If we measure all of the bits at this point, each of the 2^N possible states of the system has an equal probability of appearing. These states are then passed through the oracle, which serves as a quantum equivalent to the boolean expression previously declared. The circuit then flips the sign of the amplitude of the states that passed the boolean expression test. We need more than this to solve the problem because states with negative amplitude have the same probability of collapsing upon measurement.

Following this, the bits are moved into the diffusion operator. The diffusion operator rotates each state by the average amplitude. Because the desired state(s) are now negative, they will be very far from the mean value. After the diffusion operator is applied, their amplitude will be far greater than the mean, and the other states will be decreased slightly. By repeating the diffusion operator many times, this difference becomes more pronounced, and the magnitude of the desired state(s) will dwarf the undesired states. A singular measurement of the entire system after Grover's Algorithm has been applied will then have a high probability of collapsing into a state that has a high probability of solving the boolean problem.

Results

Our implementation proved successful in solving a set of boolean satisfiability problems. We successfully ran the code on the IBM quantum simulator 'Simulator_Statevector'. Our code returned the correct answer to the 3-Sat boolean expression and showed that we had successfully

used amplitude amplification to make the correct string the highest probability. We exhaustively searched with a classical algorithm to verify our answers, and everything matched.

We also proved that this implementation works when there is more than one correct answer. When this is the case all correct answers get their probability increased. As a final test to show our code was correct, we demonstrated that when there is no correct answer to the boolean expression all probabilities are the same. Unfortunately, the IBM quantum computer we had access to, ibmq_lima, was offline so we were unable to run our code on it.

Discussion

This research sparked a deep fascination and ample understanding of Grover's Algorithm for all parties involved. When we first began writing our Quantum Code, however, there were some hiccups. Certain libraries were difficult to install and did not behave properly until we restarted the Kernel. Furthermore, the Lima quantum computer was offline for the complete duration of our research project. This greatly disappointed our group, as running our code on that machine would have marked our first legitimate interaction with a Quantum Computer.

These problems were very miniscule and if it were not for this deadline we would hopefully be able to run our code on the actual quantum computer. In future work, we can make sure our libraries are correctly installed and have set up our kernel correctly. As stated before we were not able to run our code on the quantum computer so we did not see any differences. If we did run the code, we predict that the wrong answers will have different amplitudes because of noise and interference. In general, we believe that a simulator would be able to closely match actual quantum computation for such a small N value as seen in our examples. It would be interesting to see differences between simulated and actual results with a large N value and more complicated boolean expressions.

Conclusion

Grover's Algorithm allows one to parse through the Quantum domain and find useful information. While its implementation has yet to come to fruition in practical situations, the core concepts at its foundation are essential to anyone looking to become well-versed in Quantum Computing. We believe that while completing this project, we have broadened our horizons and strengthened our footing in the field of Quantum Computing. We began by introducing the topic and its potential benefits. With this introductory understanding, we went on to select and argue for our specific use case of Grover's Algorithm: boolean satisfiability problems. To implement

our version of Grover's Algorithm we took advantage of the tweedledum and qiskit python libraries. Using these we were able to easily create our custom oracle. We then used the oracle to create the Grover operator for each of our boolean expressions. From there we calculated the optimal number of times to repeat the Grover operator and made a quantum circuit as such. After creating and testing our Quantum circuit on the IBM simulator, we cross-checked with classical exhaustive search methods to ensure our Algorithm produced the right outputs. Our testing data included boolean expressions of various complexity and number of solutions, demonstrating the versatility of Grover's Algorithm. Finally, we explain how this method works by translating sections of the Python code into unitary gates and using intuitive concepts to understand their behavior.

References

[1] "Grovers Algorithm" IBM Quantum Learning

[2] "A Generic Variable Inputs Quantum Algorithm for 3-SAT Problem" Wang et al

[3] "The P Versus NP Problem" Stephen Cook

[4] "Is Quantum Search Practical" John P Hayes