QC Project: Boolean SAT using Grover’s Algorithm

My name is Aarav Ratra and I am currently a fresher at IIT Roorkee. I am pursuing my B.Tech in Engineering Physics. I am a newly recruited core member of the Quantum Computing Group in IIT Roorkee, and this is my first QC Project.

The aim of this project is to implement Grover's Algorithm to solve a Boolean SAT Problem. The problem is as follows:

*Frank wants to throw a dinner party to celebrate Alice and Bob’s engagement. He is also considering inviting their mutual friends Charles, Dave and Eve. However, he is aware that Charles will come to the party only if Dave comes without Eve. Frank wants to know what possible combinations of invitations he can write for his friends Alice, Bob, Charles, Dave and Eve. Help Frank calculate all the possible combinations using Grover’s algorithm.*

Grover’s Algorithm is a well-known unstructured quantum search algorithm. It provides a quadratic speedup over classical search techniques like linear and binary search, as it uses O(√N) evaluations instead of O(N). The basic idea behind Grover’s algorithm is to create a uniform superposition of all states and amplify the amplitudes of the winning states i.e. those states which we are searching for. The first step is to introduce a phase reversal to the winning states by designing an oracle which introduces a phase kickback. An ancilla qubit can be used for this. After that, we use a diffuser which causes an increase in amplitude of the states with a phase reversal. This is repeated π/4\*√N times. Hence, on repeated shots we can obtain the probabilities of all states and hence obtain information on the winning states.

Geometric representation of Grover’s Algorithm (courtesy: Wikipedia)

A screenshot of a computer

Description automatically generated with low confidence

A Boolean Satisfiability Problem aims to check if a given Boolean expression is satisfiable or not i.e. we check if there exist any set of values which when assigned to the variables yield a TRUE result for the Boolean expression. We can do this by checking each and every possible combination of variables and evaluating the Boolean. However, using Grover’s algorithm, we can design an oracle which would intake a uniform superposition and check for satisfiability, hence not only confirming satisfiability but also showing the combinations for which our Boolean expression evaluates to be TRUE. Hence, we can model our problem to be a Boolean expression and solve for all possibilities using Grover’s.

The problem can be expressed as a Boolean Expression:

A • B • (C • D • ~E + ~C) , which is equivalent to (A • B • C • D • ~E)+(A • B • ~C)

Hence, we design a Grover's Reflection Oracle which would intake a uniform superposition and induce a phase kickback for whichever input satisfies the given Boolean. While this can be done by formulating a Unitary Matrix using classical methods and implementing it as a gate, I have done it in a different manner which encodes the Boolean Expression using known quantum gates including the H gate and Generalised Toffoli gates.

As you have seen, I have expanded the Boolean Expression as ABCD~E + AB~C . It appears I have split it into two parts ABCD~E , AB~C , and I would use Generalised Toffoli Gate with X gates and an ancilla qubit in |-> state as the target qubit, so that a phase kickback is introduced whenever the Boolean is satisfied. The phase kickback corresponds to a reflection of the winning states.

Diagram

Description automatically generated with medium confidence

Do note that I used an additional ancilla qubit to design a C5X gate here since the C5X (and in general N-Control Toffoli Gates) were not available in Qiskit. Ideally I would use only 1 ancilla qubit. I used the C3X and C4X Gates available in Qiskit since designing them with only CX gates would use too many ancilla qubits.

Each Generalised Toffoli Gate (with bit flips wherever required) represented the elements of the Boolean Expression joined together with AND, and multiple such gates in conjunction summed them up (i.e. in OR). This is how I managed to encode the Boolean Expression in terms of these gates.

Now coming to the second part i.e. the Diffuser.

The diffuser matrix is of the form 2|D><D|-I. |D> represents the uniform superposition created by applying Hadamard gate on n qubits initialised to |0> state. This geometrically refers to a reflection about the original |D> state of the Qubits before entering the Oracle, hence leading to an increase in amplitude for the winning states. A general diffuser circuit for 3 qubits is shown below:

Diagram

Description automatically generated

For an n qubit system, we would need a multi-controlled Z gate, which was not available in Qiskit. However, using a C4X Toffoli, a single CZ gate and an Ancilla Qubit (which I recycled), I was able to design an effective 5-controlled Z gate:

Chart, box and whisker chart

Description automatically generated

Now with both the oracle and diffuser ready, we can configure the circuit. One call to the oracle and diffuser is sufficient to give us accurate answers. Chart, diagram

Description automatically generated

Before measurement, I used a state-vector simulator to show the probabilities prior to performing the measurements. As per that, the probability distribution obtained is as follows: A picture containing graphical user interface

Description automatically generated

Note that the Qubits are in order ABCDE. I have discarded the values for the Ancilla qubits since they were both set to |0> and hence their states were known.

P.F.A. some sample measurements performed on QASM simulator:

Graphical user interface

Description automatically generated with low confidence Graphical user interface

Description automatically generated with low confidence A picture containing graphical user interface

Description automatically generated A picture containing graphical user interface

Description automatically generated

As clearly visible, we are getting the desired output and hence the solution to our SAT.

Bibliography:

* Qiskit Youtube Channel
* Qiskit Textbook
* Qiskit Documentation
* QuantumComputingUK.org
* QuantumComputing StackExchange
* Quantum Walks and Search Algorithms (for Grover’s Algorithm)
* Quantum Computation and Quantum Information by Nielsen and Chaung