Quantum algorithms for Hidden Subgroup problem

Mentor: Jaideep Mulherkar

Group number: 4

Group members

Vivek davara - 202003009

Binal Patel - 202003035

Hardik Jani - 202003041

Introduction

The hidden subgroup problem(HSP) involves finding a hidden structure within a larger group by querying the group with different elements and analyzing the resulting information.

- Finding structure or patterns within a given set of elements that are not directly observable.
- Given a set of elements and a group operation, the goal is to identify a subgroup that is hidden in some way.
- Applications in many areas, such as factoring large numbers, designing secure cryptographic protocols, and solving graph isomorphism problems.

Problem Statement

The HSP can be defined as follows: Given a function f : G -> X, where G is a finite group and X is a finite set, the goal is to identify a hidden subgroup H < G such that f(g) = f(g') if and only if gg'^{-1} is an element of H for all g, g' in G.

Classical approaches for HSP

1)Exhaustive Search:

- Checking each possible subgroup and comparing the values of the function f for each element pair in the group.
- For large groups number of possible subgroups grows exponentially.

2) Random Sampling:

- Randomly sample elements from the group
- Not guaranteed to find the hidden subgroup efficiently, especially for large or complex groups.

3) Approximation Algorithms.

Hadamard gates

- In the algorithm we use hadamard gates to gate all the possible states of the given length binary string. This is because we have to superposition all the possible states to measure the result.
- The H-gate (Hadamard gate) for |0> is H|0> = (|0〉 - |1〉) / √2 and for |1> is

  H|1> = (|0〉 - |1〉) / √2

- Like for |101> binary string input ,

  H|101〉 = (|0〉 - |1〉) / √2 ⊗ (|0〉 + |1〉) / √2 ⊗ (|0〉 + |1〉) / √2

- So the state after applying the Hadamard gate to |101〉 is a superposition of all possible binary strings of length 3.

Oracle function

- The oracle function gives us that given binary string is hidden subgroup or not. function return 0 and 1 , 0 then hidden subgroup and 1 then not a hidden subgroup.
- The function applies the oracle function to the circuit using controlled gates. It iterates over each character of the input string and checks if it's equal to '1'. If it is, it applies an X gate (bit-flip gate) to the corresponding qubit. This step effectively flips the qubits based on the input string.
- The function applies a controlled-X gate between the first qubit (index 0) and the target qubit (index n) and if first qubit |1> then the function apply C-not gate on it.

Oracle function

- It iterates over each character of the input string again and applies an X gate to the corresponding qubit if the character is '1'. This step ensures that the qubits are restored to their original states.

- Finally, the function converts the quantum circuit into a gate by calling circuit.to_gate(). This gate represents the oracle function and can be used in other quantum circuits. The gate object is then returned by the function.

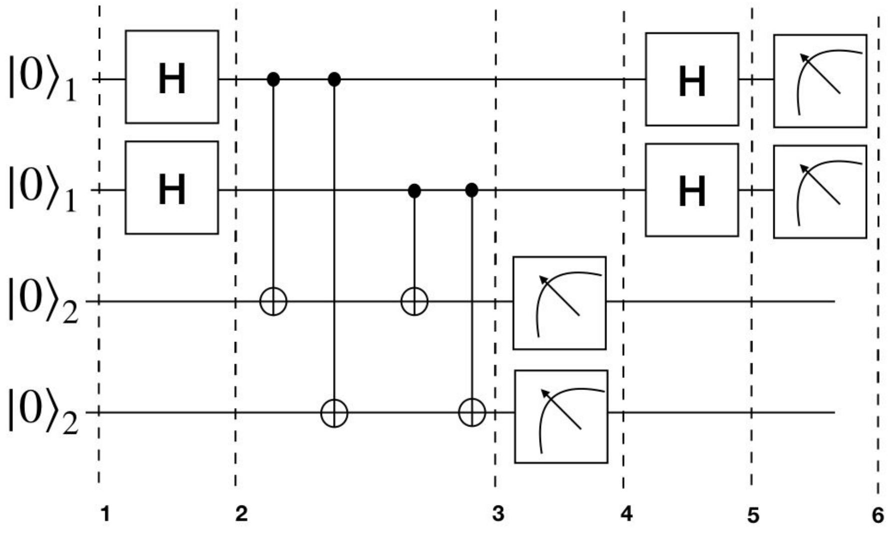
Simon´s Algorithm

1. Prepare a quantum circuit with 2n qubits, where n is the size of the input string.
2. Apply Hadamard gates to the first n qubits.
3. Apply the oracle function as a black box. The oracle takes n input qubits and produces n output qubits. The oracle is designed such that it maps each input state to a unique output state, where the output states for different input states belong to the same hidden subgroup.
4. Apply Hadamard gates to the first n qubits again.
5. Measure the first n qubits.

Simon's Algorithm

- ➔ Repeat the steps above multiple times to obtain a set of measurement results.
- ➔ Analyze the measurement results to find a set of linearly independent equations that the hidden subgroup satisfies. This can be done by examining the measurement results and looking for patterns.
- ➔ Solve the set of linear equations to find the hidden subgroup. This step requires classical post-processing.
- ➔ Once the hidden subgroup is found, it can be used to solve problems that are related to the subgroup, such as factoring large numbers or solving certain types of computational problems.

Circuit for Simon's Algorithm



2 Qubit example for Simon's Algorithm

Ref: Implementing Simon's Algorithm in Qiskit | by MR.Asif | CodeX | Medium

Example

Here's an example how the gates are applied in Simon's algorithm.

1. Prepare a quantum circuit with 4 qubits: two input qubits (q[0] and q[1]) and two output qubits (q[2] and q[3]).
2. Apply Hadamard gates (H-gates) to the input qubits: H(q[0]) and H(q[1]).
3. Apply the oracle function as a black box using Controlled-Oracle gates (C-oracle gates): For each possible input x, apply a C-oracle gate controlled by q[0] and q[1] and target on q[2] and q[3]. The gate applies the function f to the input x, resulting in the output f(x). Additionally, if the equation $f(x) = f(x \oplus s)$ holds, we apply an XOR gate between q[2] and q[3] to implement the correlation $q[2] \oplus q[3] = s$, where s is the hidden subgroup string.
4. Apply Hadamard gates (H-gates) to the input qubits again: H(q[0]) and H(q[1]). Measure the input qubits: measure(q[0]) and measure(q[1]).

By analyzing the measurement outcomes, we can obtain information about the hidden subgroup s. The correlation between the output qubits q[2] and q[3] will reveal the hidden subgroup string s, which satisfies the equation $q[2] \oplus q[3] = s$.

Python code

Oracle function takes an input string and creates a quantum circuit representing the oracle function for the Hidden Subgroup Problem. It marks a specific subgroup by applying a controlled-X gate on the target qubit (n) based on the values of the input qubits. The input string is used to determine which input qubits are flipped using the X gate. Finally, the circuit is converted to a gate and returned.

This HSP function takes the oracle gate as input and creates a quantum circuit for the HSP algorithm. The circuit consists of n+1 qubits (n input qubits and 1 target qubit). It applies Hadamard gates to the input qubits, applies the oracle gate, applies Hadamard gates again, and then measures the input qubits.

Python Code

An Input string '101' is provided. The oracle gate is created by calling the oracle_function with the input string. Then, the HSP circuit is created by calling hsp_algorithm with the oracle gate. The circuit is executed on a quantum simulator backend ('qasm_simulator') using the execute function from Qiskit. The shots parameter specifies the number of times the circuit is run. The measurement results are obtained by calling result.get_counts(), which returns a dictionary containing the counts of different measurement outcomes. The measurement outcomes are represented as binary strings, where the length of the string corresponds to the number of input qubits. Finally, the measurement results are printed, and a histogram of the results is plotted using plot_histogram.

Code file:
https://drive.google.com/file/d/1A4qOk9I2wrm8LaPx-aSo6Maq-wzAJZF8/view?usp=share_link

Thank You