Levon Keshishoghlyan

Final Project

CS 339 Introduction to Quantum Computing

2024-05-20

Description of the Problem

Consider the famous puzzle game Master Mind played by two players. Given n different colors, the first player - the keeper, secretly forms a sequence of n colored pins, where several pins may share the same color, or all of them may be of different colors. The task of the second player – the guesser, to disclose the hidden sequence with minimal guesses. Work on any of the following two versions:

- **Version 1:** each guess is graded by the keeper with two digits – the first being the number of correct pins in their correct positions and the second being the number of pins with correct colors but in wrong places. The game stops when the grade of the most recent guess is n_0.
- **Version 2:** each guess is graded by the keeper with a single digit – the number of correct pins in their correct positions. The game stops when the grade of the most recent guess is n.

Solution after the meeting

After the last discussion, I've made several changes to the algorithm. First of all, as discussed during the class I won't use Grover's algorithm to immediately get the result, I will also not use Bernstein-Vazirani algorithm. Instead of it I give several guesses to the keeper and then use them to get the correct result. The algorithm will work on the binary 2^n case when there are two colors and n qubits, but this can be easily parallelized by performing my algorithm n times in parallel for K colors. In the second case, you will simply run it for $k = \log K$ qubits and get the correct color for each pin.
But for simplicity, I will show the case with 2 colors and n qubits. I have also excluded discussions about the trivial cases when the guesser makes an instant guess by giving an exact number or its inverse as for the first option keeper will give n and for the second case 0 thus instantly allowing the guesser to get the correct answer

Task 1

The link of the repo was shared by the preliminary date

Task 2

I've performed several trials and different tests and came to conclusion that even with quantum acceleration we will need at least n or n-1 (depending on the case if n is odd or even) guesses made. Each guess will act as a similarity test as discussed during the class. The Hamming distance will be returned for each guess and based on the number probabilities of some states will be increased or set to zero. Each guess will be as a multiple of 2, e.g. $2^0, 2^1, \ldots, 2^{n-1}$ as this decreases all the unnecessary checks. The complexity of the classical algorithm will be $O(Nn)$ as it will need to do n separate guesses and for each of them $\frac{N}{2}$ Hamming distance calculations. The number is $\frac{N}{2}$ as after only the first bit differs and we can derive distance by simply taking the previous distance mod n. The Quantum algorithm will have a complexity of $O(\sqrt{N/k}n)$. It will still need to have n guesses, but for each of these guesses finding optimal states will be much more effective by the use of Grover's algorithm. $\sqrt{N/k}$ is Grover's algorithm when there are k searchable cases (taken from Wikipedia's article on Grover's algorithm).

Task 3

To describe the algorithm we will need one register (x) with n qubits that will be responsible for the searched state, second similarly sized register for the guessed sequence that will be used for the XOR operation and one register(g) with $\lceil \log n \rceil$ qubits to keep track of the matching colors.. The first register will be initialized to all zeros at first and then Hadamard will be applied to it to get it into the superposition. Afterward, x and g will be passed into the oracle that will set probabilities of states with matching Hamming distance of g to the $\frac{1}{k}$ where k is the number of such states and all of the other states will have the probability of zero. As described above n iterations of algorithm will be performed and at the end first register will be hadamared and its result will be the desired state.

Task 4

To implement the classical algorithm we will need to do n guesses. For each guess we will get keepers response r that shows the number of differing positions. Then we will need to check every possible state to the guessed sequence and mark the ones that have the distance values equal to r. Those states will pass to the next iteration of the algorithm and all of the other states will be dropped. We can get the Hamming distance by applying XOR and then summing up all 1s. A very simple implementation of this process is in the jupyter notebook.

Task 5

To implement the quantum algorithm we will use Grover's search algorithm instead of the one-by-one looking of the possible states. In this case, for each guess, we will pass all registers to the oracle. To be honest, I'm not really sure if the next part is correct, but I think that the correct oracle in this case will be if $f(x) = 1$, SUM(x XOR g) = r and 0 otherwise.