4 Rooks Problem

Wittmann Goh

QCHack 2021

1 Introduction

We build a simple quantum program using Grover's Algorithm to solve a chessboard problem.

We have a 4×4 chessboard with 3 rooks on it, and we want to find the square that is not attacked by any rook. For the simplest example, we can assume that there is only one unique answer.

2 Algorithm

2.1 Encoding of information

To convert this into something that a quantum algorithm can tackle, we need to decide how to encode the information into qubits. For our purposes, it is simple: Label each position on the board with its coordinates (x, y). We can use the formula $p = x + 4y$ to encode each coordinate as an integer. Then the position can be encoded into 4 bits, which can be converted into a 4-qubit state:

$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$
$(1,0)$	$(1,1)$	$(1,2)$	$(1,3)$
$(2,0)$	$(2,1)$	$(2,2)$	$(2,3)$
$(3,0)$	$(3,1)$	$(3,2)$	$(3,3)$

Figure 1: Coordinates of cells

$\rightarrow$

$\|0000\rangle$	$\|1000\rangle$	$\|0100\rangle$	$\|1100\rangle$
$\|0010\rangle$	$\|1010\rangle$	$\|0110\rangle$	$\|1110\rangle$
$\|0001\rangle$	$\|1001\rangle$	$\|0101\rangle$	$\|1101\rangle$
$\|0011\rangle$	$\|1011\rangle$	$\|0111\rangle$	$\|1111\rangle$

Figure 2: Qubit representation

2.2 Oracle

Now suppose we have a rook on $(0, 0)$. How do we check which cells it attacks? Notice that this is very simple - any cell in the same column as the rook will share the same first two qubits, and any cell in the same row as the rook will share the same last two qubits.

If we keep an ancilla qubit $|\text{anc}\rangle$, then we can do a `Controlled X` gate on the first two qubits to see if it matches the column, and another `Controlled X` gate on the last two qubits to check the same for the row. Finally, note that if the qubits are in the same state as the rook qubit, then this will flip once under each `Controlled X` and so will become unmarked. We can solve this by doing one last `Controlled X` based on all 4 qubits. The end result of this is that $|\text{anc}\rangle = 1$ only if the cell represented by the state is not under attack by the rook (and is also not the rook itself).

$\|0000\rangle$	$\|1000\rangle$	$\|0100\rangle$	$\|1100\rangle$
$\|0010\rangle$	$\|1010\rangle$	$\|0110\rangle$	$\|1110\rangle$
$\|0001\rangle$	$\|1001\rangle$	$\|0101\rangle$	$\|1101\rangle$
$\|0011\rangle$	$\|1011\rangle$	$\|0111\rangle$	$\|1111\rangle$

Table 1: The rook is in the top left cell. The row shares the same last two qubits (red) and the column shares the same first two qubits (blue)

Finally, since we have 3 rooks, we need to keep an array of 3 ancilla qubits, one for each rook. Recall that a position on the board is safe if it is not attacked by any rook. Therefore we keep a final oracle qubit $|\text{or}\rangle$, and we do a `Controlled X` with all the ancilla qubits as control and $|\text{or}\rangle$ as output. Then $|\text{or}\rangle = 1$ only if it is a safe square, which is what we want.

2.3 Grover's Algorithm

The only thing left to do is to use the oracle in Grover's algorithm. We start by making a 4 qubit state $|\text{register}\rangle = H^{\otimes 4}|0000\rangle$, a uniform superposition. Next, we define the 3 ancilla qubits $|\text{anc}\rangle = |000\rangle$ and the oracle qubit $|\text{or}\rangle = H|0\rangle$. In $n = \sqrt{16} = 4$ iterations, we can amplify the solution state to its maximum amplitude, and then measure the register to recover the solution to the problem.

3 Extensions

Right now the code only works with powers of two since we make use of the observation that the digits match up along rows and columns. A more generic algorithm would be able to handle arbitrary board sizes. We also assumed that there is a unique solution, which may not be the case (for example, if two rooks are on the same rank or file).