QCI Day 8

Surpassing classical computers [1]: Grover's Algorithm

Big O Notation

```
# linear search: O(n)
[31, 41, 59, 26, 53, 58, 97, 93, 23, 84, 62, 64]
# binary search: O(log n)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

P vs NP

Speaker notes

P-class problems: searching a list, sorting a list

NP-complete problems: sudoku, travelling salesman, etc.

Whether or not these two classes of problems are the same (P = NP) is a major unsolved problem in computer science and is awaiting proof (Milennium Prize Problem)

Searching a list

Speaker notes

It is very easy to verify if we found the solution, but it is harder to actually find the solution

Unstructured database search can be solved in polynomial time, but it is a good analogy to some further applications of Grover's Algorithm

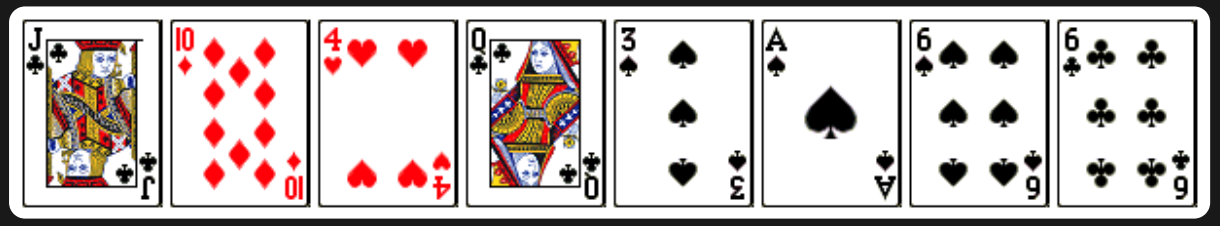
Grover's Algorithm

Speaker notes

Often presented as an unstructured database search algorithm in $O(\sqrt{n})$, and this is how we're going to learn the algorithm, but it's really more of a tool for simplifying both P and NP problems.

This does NOT mean P = NP (we only get a quadratic speedup)

The Oracle



Did we find the winner?

Speaker notes

Let's play a game! Guess what card I'm thinking of.

The idea of Grover's Algorithm: This oracle is a black box that tells us whether or not our input is the correct answer. Our goal is to find the correct answer (winner).

I was the oracle in our card game

Uniform Superposition

$$\left[\, |00\rangle, \quad |01\rangle, \quad |10\rangle, \quad |11\rangle \,\right]^T$$

$$\left[\, \frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}} \,\right]^T$$

Speaker notes

When we start, each position is equally likely. A random guess is just as good as another. We can thus put our qubits into uniform superposition.

We only need 2 qubits to represent 4 states.

This just comes from putting Hadamards on all the qubits. This one represents a 2-qubit system.

The row on the bottom represents the probability for each value

Phase Inversion

$$U_\omega |x\rangle = \begin{cases} |x\rangle & \text{if } x \neq \omega \\ -|x\rangle & \text{if } x = \omega \end{cases}$$

$$\left[\frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{-1}{\sqrt{4}} \right]^T$$

Speaker notes

The first trick we use is phase inversion. The oracle, represented by U_ω, inverts the phase of the winner state.

Let's say the winner is the $|11\rangle$ state

Oracle Matrix

$$U_\omega = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} = \mathrm{CZ}$$

$$\left[\frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{1}{\sqrt{4}}, \quad \frac{-1}{\sqrt{4}} \right]^T$$

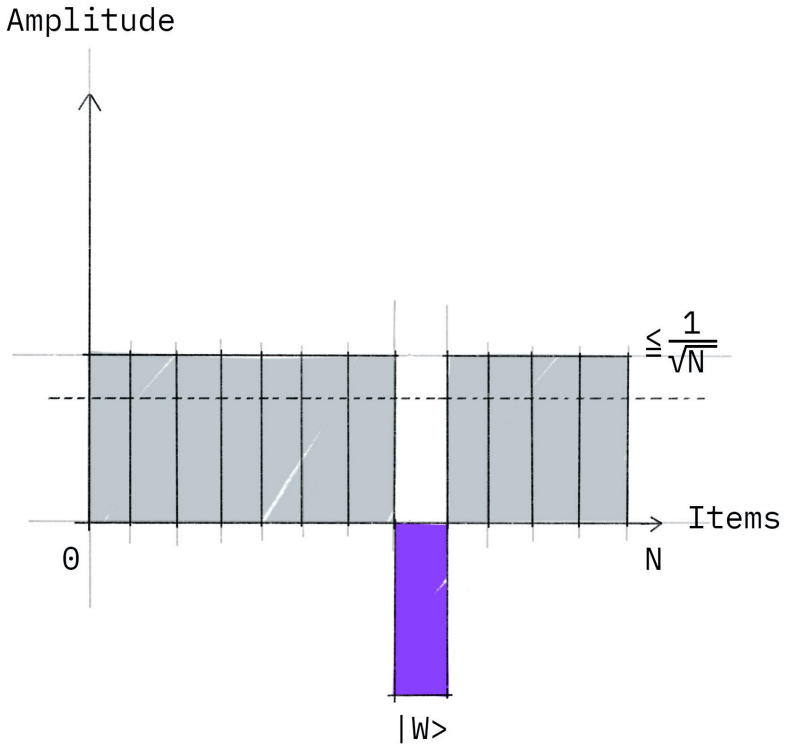
Speaker notes

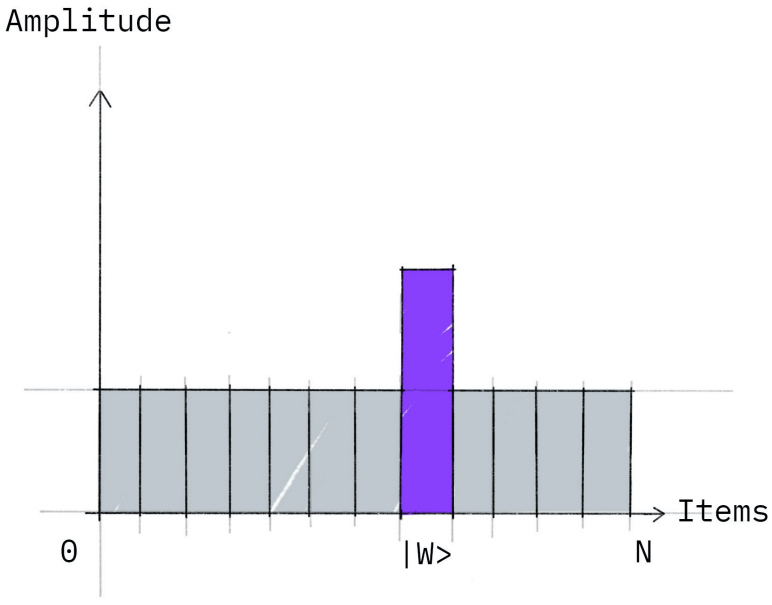
This matrix flips the sign of the $|11\rangle$ state (on iPad show correspondence between the rows and columns in matrices)

The $|11\rangle$ state maps to the $|11\rangle$ state but with a negative sign (phase flip). That's how we can tell it's controlled-Z

Show matrix multiplication on iPad

Inversion About the Mean





Speaker notes

We repeat this step up to $\sqrt{2^n}$ times, where n is the number of items

This will amplify our winning state and ultimately reduce the probabilities of the other states

Show on iPad if necessary

Coding in Qiskit

Grover's Dinner Party

References

- Yanofsky, Mannucci. Quantum Computing for Computer Scientists
- What's the point of Grover's algorithm if we have to search the list of elements to build the oracle?
- Qiskit Textbook. Grover's Algorithm
- Grovers Algorithm — Programming on Quantum Computers — Coding with Qiskit S2E3
- Dinner Party using Grover's Algorithm — Programming on Quantum Computers — Coding with Qiskit S2E5