Quantum Information Processing

Lecture 11

Dr. Ahmad Khonsari

Uiversity of Tehran

Fall 2022

December 16, 2022

Slides Prepared by Mahdi Dolati

Quantum search

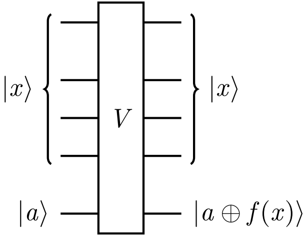
- In 1996 Indian-American computer scientist Lov Grover published a quantum search algorithm, which to this day remains one of the most important quantum algorithms.
- The algorithm concerns searching an unstructured database with N entries.
- If we are searching for a unique marked entry, then classically this would take a maximum of N queries, and $N/2$ queries on average.
- Grover's algorithm enables the task to be completed with only $O(\sqrt{N})$ queries.

Lov Grover

Image Source: dotquantum.io

Oracles revisited

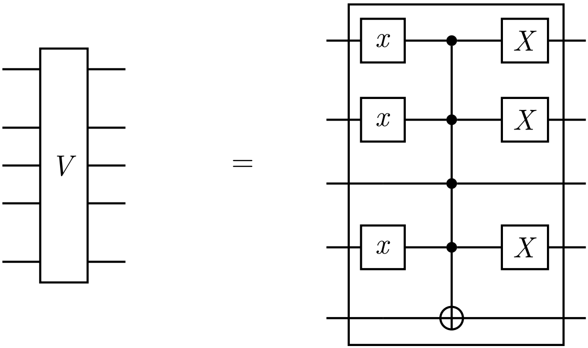
Recall that a core component of the Deutsch-Jozsa algorithm is the “oracle”, which can be thought of as something that can recognise a correct answer when it sees one. Specifically, we think of an oracle as a black box “marking” some input binary strings as 0 (i.e., $f(x) = 0$) and some as 1 (i.e., $f(x) = 1$), of the form:



- In Grovers algorithm, we consider a search oracle, V, which marks a single element as 1, and all the others as 0.
- The goal is to find the element marked 1, and we assume that there is no algorithmic short-cut to find it, so classically we would have to perform a brute-force search.

An example of an oracle

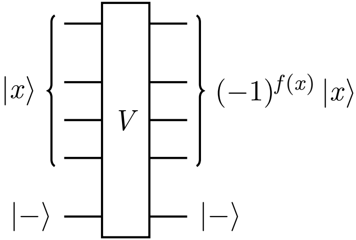
Say that 0010 is the unique "marked" binary string, then an appropriate oracle would be:



Which is in the form of a general control gate.

The action of a search oracle on $|-\rangle$

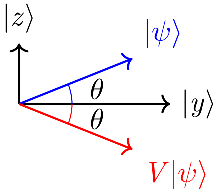
- Consider an input binary string x and let the final qubit input be set as $|-\rangle$, (i.e., the input is $|x\rangle|-\rangle$)
- The oracle transforms this to $(-1)^{f(x)}|x\rangle|-\rangle$, That is, if $f(x) = 0$ then nothing happens, whereas if $f(x) = 1$ then the last qubit is changed from $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ to $\frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) = -\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ because of the modulo-2 addition.



Visualising the action of a search oracle on $|-\rangle$

Initially we apply $H^{\otimes n}$ to the search register, that is we put the input in the uniform superposition over all bitstrings of length $n = log_2 N$ (for convenience we let the number of elements in the search space, N, be a power of 2). Now we consider the effect of the search oracle on this input, as can be visualised in a two-dimensional space where:

- $|y\rangle = \frac{1}{\sqrt{N-1}} \sum_{x \text{ s.t. } f(x)=0} |x\rangle$, i.e., a unit vector in the direction of the uniform superposition of all unmarked strings.
- $|z\rangle = |x\rangle$ where $f(x) = 1$, i.e., the marked element.

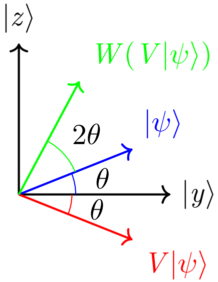


Where $|\psi\rangle = \frac{1}{\sqrt{N}} \sum_x |x\rangle$ is the uniform superposition. The action of the search oracle can be seen as a reflection in the $|y\rangle$ axis.

The second step

Reflecting about the original superposition

- As we have seen, the search oracle leads to a reflection in the $|y\rangle$ axis.
- But we need to rotate the superposition towards the $|z\rangle$ axis to increase the probability of measuring the marked state (i.e., x s.t. $f(x) = 1$) as is required.
- So we follow up the oracle step with another reflection, about the line of the original uniform superposition, which is given by $W = (2|\psi\rangle\langle\psi| - I)$ where $|\psi\rangle = |+\rangle^{\otimes n}$.



The operation of W

To see that the unitary W does indeed perform the reflection as claimed, consider an arbitrary (real) vector $|\phi\rangle$, decomposed into a component in the direction of the uniform superposition (denoted $|\psi\rangle$ as previously defined), and a component perpendicular to the uniform superposition, which we denote $|\psi^{\llcorner}\rangle$:

$$|\phi\rangle = a|\psi\rangle + b|\psi^{\llcorner}\rangle$$

So it follows:

$$\begin{aligned} W|\phi\rangle &= (2|\psi\rangle\langle\psi| - I)|\phi\rangle \\ &= (2|\psi\rangle\langle\psi| - I)(a|\psi\rangle + b|\psi^{\llcorner}\rangle) \\ &= 2a|\psi\rangle \underbrace{\langle\psi|\psi\rangle}_{=1} - 2|\psi\rangle \underbrace{\langle\psi|\psi^{\llcorner}\rangle}_{=0} - a|\psi\rangle - b|\psi^{\llcorner}\rangle \\ &= 2a|\psi\rangle - a|\psi\rangle - b|\psi^{\llcorner}\rangle \\ &= a|\psi\rangle - b|\psi^{\llcorner}\rangle \end{aligned}$$

That is, the desired reflection about the uniform superposition.

Implementing W with gates from a universal set

We still have to show that W can be implemented with gates from our universal set. We start by decomposing $W = 2|\psi\rangle\langle\psi| - I$ (where $|\psi\rangle = |+\rangle^{\otimes n}$):

$$\begin{aligned} H^{\otimes n} W H^{\otimes n} &= 2|0^n\rangle\langle 0^n| - H^{\otimes n} I H^{\otimes n} \\ &= 2|0^n\rangle\langle 0^n| - I \end{aligned}$$

because H is self-inverse, and $H|+\rangle = |0\rangle$ (note $|0^n\rangle = |0\rangle^{\otimes n}$). We can also use the fact that $X|0\rangle = |1\rangle$:

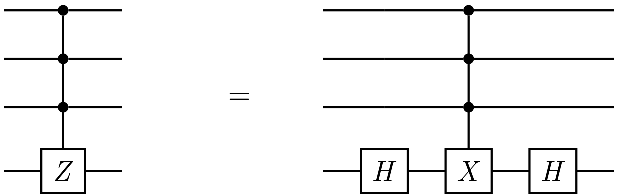
$$\begin{aligned} X^{\otimes n}(H^{\otimes n} W H^{\otimes n}) X^{\otimes n} &= X^{\otimes n}(2|0^n\rangle\langle 0^n| - I) X^{\otimes n} \\ &= 2|1^n\rangle\langle 1^n| - I \end{aligned}$$

which is the matrix:

$$2\begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix} = -\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & -1 \end{bmatrix}$$

Implementing W with gates from a universal set (cont.)

We have that $(-2|1^n\rangle\langle 1^n| + I)$ is a n-qubit generalisation of the CZ gate, which we denote $C_{n-1}Z$:

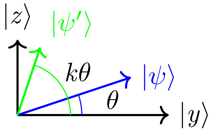


As $Z = HXH$, we have that $C_{n-1}Z = (I_{n-1} \otimes H)C_{n-1}X(I_{n-1} \otimes H)$, which can be efficiently implemented using Toffoli gates with some extra workspace qubits.
Putting this altogether we have:

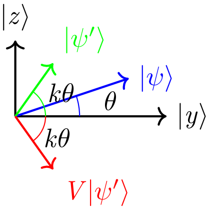
$$W = -H^{\otimes n}X^{\otimes n}(I_{n-1} \otimes H)C_{n-1}X(I_{n-1} \otimes H)X^{\otimes n}H^{\otimes n}$$

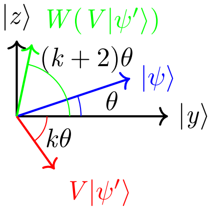
Iterating these two reflections

Say we have rotated to an angle $k\theta$:



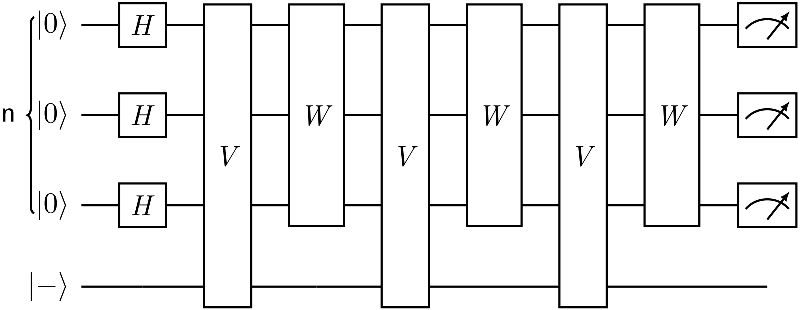
Then the action of the matrices V and then W will be:





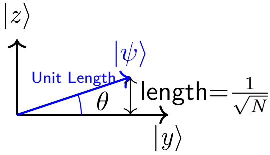
Thus, each occurrence of the Grover iterate, WV, leads to a 2θ rotation.

It follows that Grover's algorithm consists of an iteration over V and W, which rotates the uniform superposition $|\psi\rangle$ towards a superposition consisting of only (or dominated by) the marked element, $|z\rangle$:



How many iterations?

By simple trigonometry we can express θ:

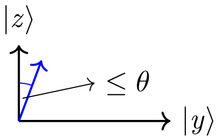


So $\sin\theta = (\frac{1}{\sqrt{N}})/1$. If we are searching a large database then $\theta \approx \sin\theta$. Each iteration rotates the superposition 2θ, and we need to rotate $(\frac{\pi}{2} - \theta)$ in total, thus we require n_{it} iterations, where:

$$\frac{\pi}{2} - \theta = n_{it} \times 2\theta$$
$$\Rightarrow n_{it} = \frac{\pi}{4\theta} - \frac{1}{2}$$
$$\approx \frac{\pi\sqrt{N}}{4}$$

so we require only $O(\sqrt{N})$ iterations of WV in Grovers algorithm, as opposed to checking all N elements in a classical brute-force search.

Other things to note



- In general N may be such that the superposition never quite aligns with $|z\rangle$, however we can always rotate to within θ of $|z\rangle$. Therefore we will measure the marked answer with probability at least:

$$(\cos\theta) = 1 - (\sin\theta)^2 = 1 - (\frac{1}{\sqrt{N}})^2 = \frac{N-1}{N}$$

- Rotating too far reduces the probability of getting the right answer.
- Grovers algorithm also works when there is more than one marked answer (where the aim is to find any marked answer).
- It has been shown that $\Theta\sqrt{N}$ is a lower bound for search of an unstructured database.

Oracles and complexity

Recall that an oracle can be thought of as something that can recognise an answer.

- In general an oracle can be any function that outputs a 0 or 1.
- If we now think about NP-complete problems, by definition the solution can be verified by a function which can be computed in polynomial time... thus we can encode such a function as an oracle.

So it follows that, if we want to solve a NP-complete problem by a brute force search through all possible solutions, then we can use the polynomial verifying function as an oracle, and thus Grovers algorithm can yield a quadratic speed-up (note there is a subtlety here regarding the fact that in general a NP-complete problem may have an unknown number of solutions, so we may not know how many iterations of Grovers algorithm to perform to get the correct rotation, but this can be taken care of using other techniques).