# A Gentle Introduction to Quantum Computing Algorithms

Elliot Catt[1] Marcus Hutter[1,2]

[1]Australian National University, [2]Deepmind

{elliot.carpentercatt,marcus.hutter}@anu.edu.au

A

RESTRICTED

Strong abuse of notation

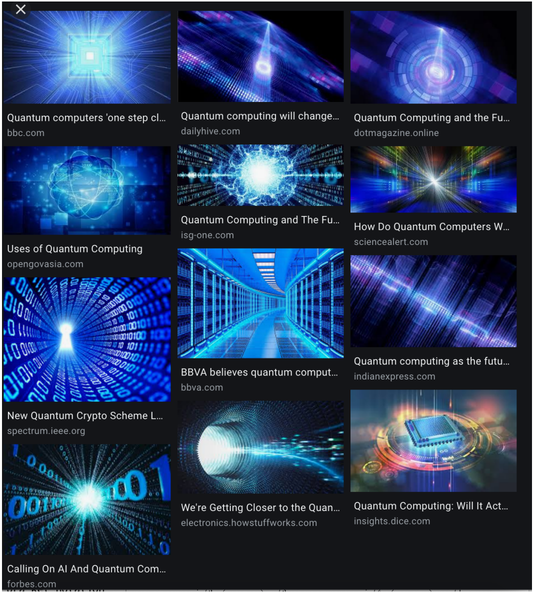
Mild use of Mathematics

Set scenes

Moderate explanations

# Table of Contents

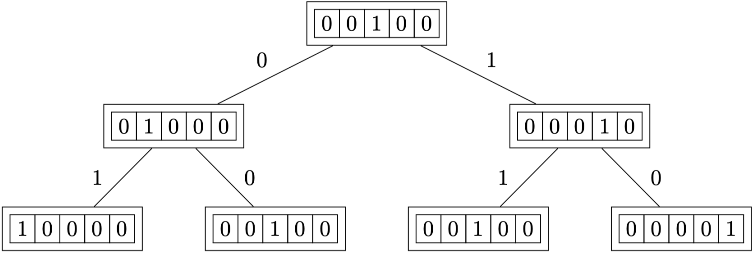
# What is Quantum Computing?



# Quantum Turing Machine

- In a certain sense, a quantum computing (/Turing machine) is essentially a probabilistic computing (/Turing machine) which uses the $L^2$ norm instead of the $L^1$ norm
- Has complex-valued amplitudes in the place of non-negative real probabilities
- A complex-valued unitary transition matrix instead of a stochastic one

# (Deterministic) Turing Machine



# (Deterministic) Turing Machine

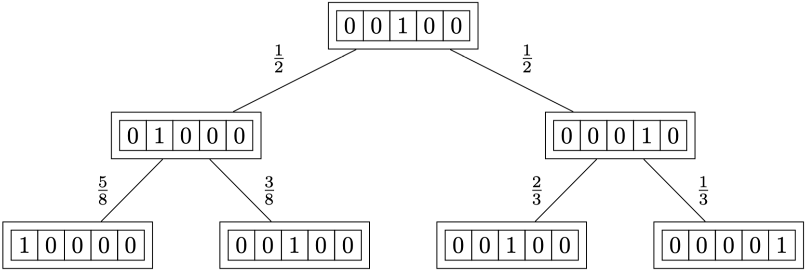
## Definition (Bernstein & Vazirani (1997))

A deterministic Turing machine is a triplet $(\Sigma, Q, \delta)$, where $\Sigma$ is a finite alphabet with an identified blank symbol $\#$, $Q$ is a finite set of states with identified initial state $q_0$ and finial state $q_f \neq q_0$, and $\delta$, a deterministic transition function, is a function

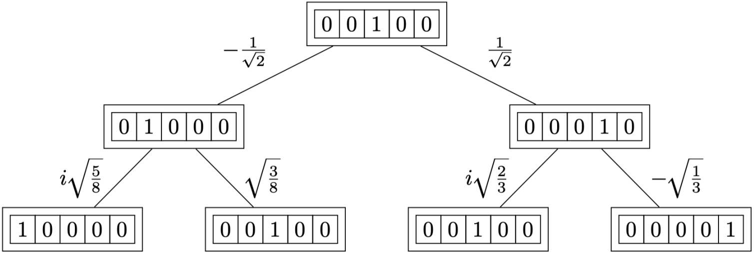
$$\delta \; : \; Q \times \Sigma \to \Sigma \times Q \times \{L, R\} \tag{1}$$

Here $\{L, R\}$ denote left and right, directions to move on the tape. The state $q_f$ is also called the Halting state.

# Probabilistic Turing Machine



# Quantum Turing Machine



# Quantum Turing Machine

### Definition (Bernstein & Vazirani (1997))

Call $\tilde{\mathbb{C}}$ the set consisting of $\alpha \in \mathbb{C}$ such that there is a deterministic algorithm that computes the real and imaginary parts of $\alpha$ to within $2^{-n}$ in time polynomial in $n$.

If we do not use this restriction "it is possible to smuggle hard-to-compute quantities into the transition amplitudes, for instance by letting the $i$th bit indicate whether the $i$th deterministic TM halts on a blank tape."

# Quantum Turing Machine

## Definition (Bernstein & Vazirani (1997))

A Quantum Turing Machine $M$ is defined, much like a classical Turing Machine (Definition 1), by a triplet $(\Sigma, Q, \delta)$ where $\Sigma$ is a finite alphabet with an identified blank symbol $(\#)$, $Q$ is a finite set of states with identified initial state $q_0$ and final state $q_f \neq q_0$, and $\delta$, the quantum transition function,

$$\delta \ : \ Q \ \times \Sigma \ \rightarrow \ \tilde{\mathbb{C}}^{\Sigma \ \times \ Q \ \times \ \{L,R\}}.$$

The QTM $M$ has a two-way infinite tape of cells indexed by $\mathbb{Z}$, each holding symbols from $\Sigma$, and a single read/write tape head that moves along the tape. A configuration or instantaneous description of the QTM is a complete description of the contents of the tape, the location of the tape head, and the state $q \in Q$ of the finite control.

# Bra and ket

The Dirac bra-ket (Dirac, 1939) notation is as follows: first we use it to represent the standard basis vectors of $\mathbb{C}^2$

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

with a single qubit being described as

$$|\phi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}.$$

This is called a ket. Where $\alpha, \beta \in \mathbb{C}$

# Bra and ket

For $|a\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}$ and $|b\rangle = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$, and $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \mathbb{C}$, we will also define the tensor product in bra-ket notation as follows:

$$|a\rangle \otimes |b\rangle = |a\rangle\,|b\rangle = |ab\rangle = \begin{pmatrix} \alpha_0\beta_0 \\ \alpha_0\beta_1 \\ \alpha_1\beta_0 \\ \alpha_1\beta_1 \end{pmatrix}$$

# Bra and ket

For example, instead of writing $|0\rangle \otimes |0\rangle \otimes |1\rangle$ we will write

$$|001\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \cdot 1 \cdot 0 \\ 1 \cdot 1 \cdot 1 \\ 1 \cdot 0 \cdot 0 \\ 1 \cdot 0 \cdot 1 \\ 0 \cdot 1 \cdot 0 \\ 0 \cdot 1 \cdot 1 \\ 0 \cdot 0 \cdot 0 \\ 0 \cdot 0 \cdot 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

.

# Bra and ket

We will also raise some qubits to the power of tensors, for example:

$$|a\rangle^{\otimes 4} = |a\rangle \otimes |a\rangle \otimes |a\rangle \otimes |a\rangle = \begin{pmatrix} \alpha_0 \cdot \alpha_0 \cdot \alpha_0 \cdot \alpha_0 \\ \alpha_0 \cdot \alpha_0 \cdot \alpha_0 \cdot \alpha_1 \\ \vdots \\ \alpha_1 \cdot \alpha_1 \cdot \alpha_1 \cdot \alpha_0 \\ \alpha_1 \cdot \alpha_1 \cdot \alpha_1 \cdot \alpha_1 \end{pmatrix}$$

# Bra and ket

Additionally we will define the conjugate transpose as

$$\langle a| := |a\rangle^\dagger = (\bar{\alpha_0}, \bar{\alpha_1})$$

where $\bar{\alpha}$ is the complex conjugate of $\alpha$. This notation is called a bra.

$$\langle a|\,|b\rangle = \langle a\,|b\rangle = \bar{\alpha_0}\beta_0 + \bar{\alpha_1}\beta_1$$

# Superposition

- A collection of qubits (vector) $v \in \mathbb{C}^{2^n}$ is said to be in a superposition if $\langle v| |v\rangle = 1$.
- We require that the operators (matrices) we apply to collections of qubits (vectors) preserve this superposition property.

# Unitary Matrices

- The property we are interested in is called unitary. An operator (matrix) $U$ is unitary if inverse of $U$ is also the conjugate transpose, i.e. $UU^{\dagger} = I$.
- If any linear operator was allowed, Quantum Computing would be *unreasonably powerful* (Aaronson, 2005).

# Quantum Gates

## Definition

The Hadamard gate $H$ acts on a single qubit and corresponds to the following unitary matrix

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

For instance $H|0\rangle = \left|\frac{1}{2}\right\rangle := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $HH|0\rangle = H\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = |0\rangle$.

It is important to note that the Hadamard gate is both self-adjoint and its own inverse. That is, $HH = HH^{\dagger} = I$.

# Quantum Gates

## Definition

The controlled-not gate, $CNOT$, acts on two qubits and performs the not (bit flip) operation on the second qubit if the first qubit is $|1\rangle$. This equates to the following unitary matrix

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

For instance $CNOT\,|0\rangle\,|a\rangle = |0\rangle\,|a\rangle$ and $CNOT\,|1\rangle\,|a\rangle = |1\rangle \otimes \begin{pmatrix} \alpha_1 \\ \alpha_0 \end{pmatrix}$

# Quantum Gates

## Definition

The $\pi/8$ gate, $R_{\pi/4}$, corresponds to a rotation of the $|1\rangle$ qubit by $\pi/4$. The matrix representing this rotation is

$$R_{\pi/4} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}, \; R_{\pi/4}\,|a\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 e^{i\frac{\pi}{4}} \end{pmatrix}.$$

The gate is called the $\pi/8$ gate for historical reasons, even though the gate is a rotation of $\pi/4$.

# Quantum Gates

These three gates are important as they form a universal set of gates for two qubits.
This means that any classical two bit circuit can be constructed using only these three gates (Nielsen & Chuang, 2002).

# Quantum Gates

The Solovay–Kitaev theorem states that there exists universal sets of gates such that any unitary matrix can be efficiently approximated by a finite sequence of gates from this set. Nielsen & Chuang (2002)

# Measurement

The final part of any quantum computing algorithm is measurement, when the superpositions collapse. With $\alpha_i \in \mathbb{C}$ for all $i \in \{0,1\}^n$, measurement outputs $i$ with probability $|\alpha_i|^2$. That is,

$$\left( \sum_{x \in \{0,1\}^n} \alpha_x \left| x \right\rangle \right) \to i \quad \text{with probability } |\alpha_i|^2$$

Where $i \in \{0,1\}^n$. Although we can only ensure an outcome with some probability, we can repeat the computation and reduce the probability of error.

# Submodules

## Quantum Oracle

- The quantum oracle is used when we want to apply a function $f : \{0,1\}^n \to \{0,1\}$ to a superposition of all elements of $\{0,1\}^n$.
- Since all transforms in quantum computing are reversible (and indeed unitary) there needs to be some way to keep the information so that the transform can be reversed.
- Classically for $x \in \{0,1\}^n$ we could take $x \to f(x)$, however when performing this transform in quantum computing we do the following

$$U_f |x\rangle |y\rangle = |x\rangle |y \oplus f(x)\rangle .$$

Where $y \in \{0,1\}$ is representing an extra qubit used for this reversibility.

The quantum Fourier transform ($\mathcal{QFT}$) is a linear operator which acts on a vector $|j\rangle$ of size $2^n$ as follows,

$$\mathcal{QFT}\,|j\rangle = \frac{1}{2^{n/2}} \sum_{k \in \{0,1\}^n} e^{2\pi i jk/2^n}\,|k\rangle\,. \tag{2}$$

# Submodules

## Inverse Quantum Fourier Transform

$$\mathcal{QFT}^{-1}\left(\frac{1}{2^{n/2}}\sum_{k\in\{0,1\}^n} e^{-2\pi ijk/2^n}\left|k\right\rangle\right) = \left|j\right\rangle \tag{3}$$

# Quantum Algorithms

## Overview

- Take some initial state such as $|0\rangle^{\otimes n}$
- Quantumize to create a uniform superposition over all possible qubits, often done with the Hadamard gate $H^{\otimes n}$
- Perform computation of some function in simultaneous states of this superposition
- Uncompute the superposition, often done with the Hadamard gate or the inverse Quantum Fourier transform
- Measurement of some or all of the circuit

# Deutsch-Jozsa Algorithm

- The Deutsch-Jozsa Algorithm is the first example of an exponential "quantum-speedup".
- Imagine we are given a function $f : \{0,1\}^n \to \{0,1\}$ that has the property that either all values map to 0, or half of them do.
- Our objective is to determine whether every value maps to 0, or half of them do.
- To check this classically, one must perform at most $2^{n-1}+1$ function evaluations.
- This is because the moment the function outputs a 1 we know that $f$ outputs 1 on half the inputs.
- The Deutsch-Jozsa Algorithm requires only 1 function evaluation.

# Deutsch-Jozsa Algorithm

$$|0\rangle^{\otimes n}|1\rangle \to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle(|0\rangle - |1\rangle)$$

Hadamard $H^{\otimes n} \otimes H$

# Deutsch-Jozsa Algorithm

$$|0\rangle^{\otimes n}|1\rangle \to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) \qquad \text{Hadamard } H^{\otimes n} \otimes H$$

$$\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) \qquad f \text{ oracle}$$

# Deutsch-Jozsa Algorithm

$$
\begin{aligned}
|0\rangle^{\otimes n}|1\rangle &\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) && \text{Hadamard } H^{\otimes n} \otimes H \\
&\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) && f \text{ oracle} \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle) && \text{since } f(x) = 0, 1
\end{aligned}
$$

# Deutsch-Jozsa Algorithm

$$
\begin{aligned}
|0\rangle^{\otimes n}|1\rangle &\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) && \text{Hadamard } H^{\otimes n} \otimes H \\
&\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) && f \text{ oracle} \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle) && \text{since } f(x) = 0, 1 \\
&\to \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left[ \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle \right] |1\rangle && \text{Hadamard } H^{\otimes n} \otimes H
\end{aligned}
$$

# Deutsch-Jozsa Algorithm

$$
\begin{aligned}
|0\rangle^{\otimes n}|1\rangle &\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) && \text{Hadamard } H^{\otimes n} \otimes H \\
&\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) && f \text{ oracle} \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle) && \text{since } f(x) = 0, 1 \\
&\to \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left[ \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle \right] |1\rangle && \text{Hadamard } H^{\otimes n} \otimes H \\
&= \frac{1}{2^n} \sum_{y\in\{0,1\}^n} \left[ \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y + f(x)} \right] |y\rangle\, |1\rangle && \text{Re-ordering}
\end{aligned}
$$

# Deutsch-Jozsa Algorithm

$$
\begin{aligned}
|0\rangle^{\otimes n}|1\rangle &\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) && \text{Hadamard } H^{\otimes n} \otimes H \\
&\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) && f \text{ oracle} \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle) && \text{since } f(x) = 0, 1 \\
&\to \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left[ \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle \right] |1\rangle && \text{Hadamard } H^{\otimes n} \otimes H \\
&= \frac{1}{2^n} \sum_{y\in\{0,1\}^n} \left[ \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y+f(x)} \right] |y\rangle\, |1\rangle && \text{Re-ordering} \\
&\to \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \right|^2 && \text{Measurement on first } n \text{ qubits}
\end{aligned}
$$

# Deutsch-Jozsa Algorithm

$$
\begin{aligned}
|0\rangle^{\otimes n}|1\rangle &\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|0\rangle - |1\rangle) && \text{Hadamard } H^{\otimes n} \otimes H \\
&\to \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} |x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) && f \text{ oracle} \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle(|0\rangle - |1\rangle) && \text{since } f(x) = 0, 1 \\
&\to \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left[ \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle \right] |1\rangle && \text{Hadamard } H^{\otimes n} \otimes H \\
&= \frac{1}{2^n} \sum_{y\in\{0,1\}^n} \left[ \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y + f(x)} \right] |y\rangle\, |1\rangle && \text{Re-ordering} \\
&\to \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \right|^2 && \text{Measurement on first } n \text{ qubits} \\
&= \begin{cases} 1 & \text{if } f(x) = 0\ \forall x \in \{0,1\}^n \\ 0 & \text{if } f(x) = 0 \text{ for half the } x \in \{0,1\}^n \end{cases}
\end{aligned}
$$

# Deutsch-Jozsa Algorithm

The quantum circuit below is exactly the transforms described above.

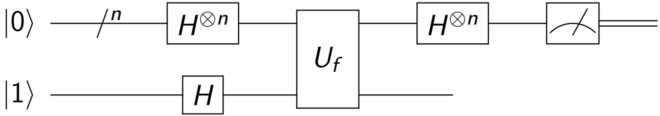



Figure: Quantum circuit for the Deutsch-Jozsa Algorithm (Nielsen & Chuang, 2002)

# Harrow-Lloyd Algorithm for Linear equations (Harrow et al., 2009)

- Given some $N \times N$ matrix $A$ and some vector $b$, finding the solution $x$ to the equation $Ax = b$ is known as the linear equation problem
- Classically this can be done in many ways, such as matrix inversion (finding $A^{-1}$ such that $x = A^{-1}b$)
- Classically the fastest algorithm takes $O(N\kappa)$ time, where $\kappa$ is the condition number of the matrix $A$
- The Harrow-Lloyd algorithm (Harrow et al., 2009) is able to achieve an exponential speedup in $N$ by taking $O(\log(N)\kappa^2)$ time, if $\kappa = O(1)$

Note that when $\kappa = O(N)$ this algorithm provides no speedup.

# Harrow-Lloyd Algorithm for Linear equations (Harrow et al., 2009)

- At this point the reader may question the existence of the algorithm since to output an $N$ long vector $x$, one must use at least $N$ steps
- This is correct, however, if one is interested in some property of $x$, such as $||Mx||_{tr}$ for some matrix $M$, it will provide an exponential speedup over classical methods
- The procedure relies on the quantum phase estimation and hamiltonian simulation, for both of which there are fast quantum algorithms

# Grover Search

Grover's search (Grover, 1996), formally described in Nielsen & Chuang (2002), takes a function $f$ such that there is at least one $s$ such that $f(s) = 1$, a set $S = \{0, 1\}^n$ of inputs of size $|S| = N = 2^n$, and is able to find an $s \in S$ which satisfies $f(s) = 1$ in $O(\sqrt{N})$ time.

# Grover Search

The Grover Search Algorithm is effectively $O(\sqrt{N})$ application of the grover iteration.
The Grover iteration looks like

$$G = (H^{\otimes n}(2\left|0\right\rangle^{\otimes n}\left\langle 0\right|^{\otimes n} - I_n)H^{\otimes n})U_\omega$$

Nielsen & Chuang (2002).

# Grover Search

- To demonstrate how the Grover operator is able to give the desired answer, a geometric analysis is quite useful.
- Let $M$ denote the number of solutions to $f(s) = 1$, that is $M = |\{s \in S : f(s) = 1\}|$.
- Let $|\beta\rangle := \frac{1}{\sqrt{M}} \sum_{x \in M} |x\rangle$ be the vector of all $M$ solutions
- And $|\eta\rangle := \frac{1}{\sqrt{N-M}} \sum_{x \in S \setminus M} |x\rangle$ be the vector of $N - M$ non solutions

We can write the uniform state as

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle = \sqrt{\frac{N-M}{N}} |\eta\rangle + \sqrt{\frac{M}{N}} |\beta\rangle .$$

# Grover Search

The oracle transform reflects $|\beta\rangle$ about $|\eta\rangle$; mathematically we can write this as

$$\begin{aligned} U_\omega(p\,|\eta\rangle + q\,|\beta\rangle)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) &= p\,|\eta\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) + q\,|\beta\rangle\left(\frac{|1\rangle - |0\rangle}{\sqrt{2}}\right) \\ &= (p\,|\eta\rangle - q\,|\beta\rangle)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \end{aligned}$$

# Grover Search

- The transform $(H^{\otimes n}(2\left|0\right\rangle^{\otimes n}\left\langle 0\right|^{\otimes n} - I_n)H^{\otimes n})$ is a reflection about $\frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}\left|x\right\rangle$.
- Performing these two reflections together gives a rotation.

# Grover Search

Let $\cos\frac{\theta}{2} = \sqrt{\frac{N-M}{N}}$, then we have that $\sin\frac{\theta}{2} = \sqrt{\frac{M}{N}}$ and we can re-write the uniform state as,

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle = \cos\frac{\theta}{2}|\eta\rangle + \sin\frac{\theta}{2}|\beta\rangle .$$

Then applying the Grover iteration to both sides we get

$$\begin{aligned} G\left(\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\right) &= (H^{\otimes n}(2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - I_n)H^{\otimes n})U_\omega\left(\cos\frac{\theta}{2}|\eta\rangle + \sin\frac{\theta}{2}|\beta\rangle\right) \\ &= (H^{\otimes n}(2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - I_n)H^{\otimes n})\left(\cos\frac{\theta}{2}|\eta\rangle - \sin\frac{\theta}{2}|\beta\rangle\right) \\ &= \cos\left(\frac{3\theta}{2}\right)|\eta\rangle + \sin\left(\frac{3\theta}{2}\right)|\beta\rangle \end{aligned}$$

# Grover Search

Applying the iteration $k$ times leads to

$$G^k \left( \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \right) = \cos\left(\frac{2k\theta + \theta}{2}\right) |\eta\rangle + \sin\left(\frac{2k\theta + \theta}{2}\right) |\beta\rangle \, .$$

# Grover Search

Thus we perform the iteration a number of times so that $\sin\left(\frac{2k\theta+\theta}{2}\right)$ is close to 1, which leads to

$$k = \left\lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rceil .$$

# Grover Search

This can be derived by

$$
\begin{aligned}
\sin\left(\frac{2k\theta + \theta}{2}\right) &\approx 1 \\
\frac{2k\theta + \theta}{2} &\approx \frac{\pi}{2} \\
\theta\frac{2k + 1}{2} &\approx \frac{\pi}{2} \\
2k + 1 &\approx \frac{\pi}{\theta} \\
k &\approx \frac{\pi}{2\theta} - \frac{1}{2} \\
k &\approx \frac{\pi}{4}\sqrt{\frac{N}{M}} - \frac{1}{2}
\end{aligned}
$$

# Grover Search

The algorithm for the case when $M = 1$ and $f(x') = 1$ can be defined as follows:

$$|0\rangle^{\otimes n}|0\rangle \to \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle (|0\rangle - |1\rangle) \quad \text{Hadamard}$$

then repeat the Grover iteration $\lceil (\pi\sqrt{N}/4) \rceil$ times

$$\to ((H^{\otimes n}(2|0\rangle^{\otimes n}\langle 0|^{\otimes n} - I_n)H^{\otimes n})U_\omega)^{\lceil (\pi\sqrt{N}/4)\rceil} \left( \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle (|0\rangle - |1\rangle) \right)$$

$$= G^{\lceil (\pi\sqrt{N}/4)\rceil} \left( \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \{0,1\}^n} |x\rangle (|0\rangle - |1\rangle) \right)$$

$$\approx |\beta\rangle \left( \frac{|0\rangle - |1\rangle}{2} \right)$$

$$= |x'\rangle \left( \frac{|0\rangle - |1\rangle}{2} \right)$$

$$\to x' \quad \text{Measurement on first } n \text{ qubits}$$

Figure: Quantum Search Algorithm (Nielsen & Chuang, 2002)

# Grover Search

To produce a quantum circuit, we can just write out each transform used in order.

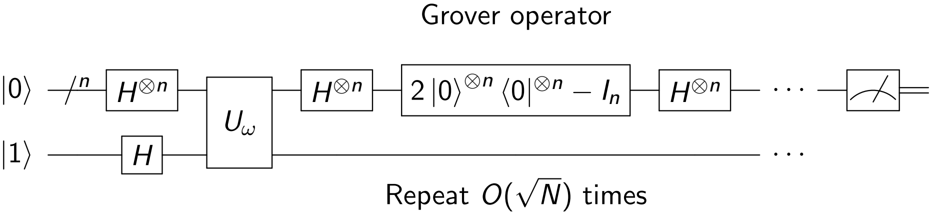



Figure: Quantum Circuit for Grover's algorithm (Nielsen & Chuang, 2002; Wikipedia, 2017a)

# Quantum Counting Algorithm

- The Quantum Counting Algorithm, proposed in Brassard et al. (1998) and described in Nielsen & Chuang (2002), is a combination of Grover search and phase estimation.
- Given an oracle indicator function $f_B : A \to \{0, 1\}$ of $B \subseteq A$, with $|A| = N = 2^n$, the Quantum Counting Algorithm finds $M = |B|$.

# Quantum Counting Algorithm

To find $M$ the Quantum Counting Algorithm finds a solution $\theta$ to the equation

$$\sin^2\left(\frac{\theta}{2}\right) = \frac{M}{2N} \tag{4}$$

then solves for $M$.

# Quantum Counting Algorithm

- Phase estimation, is a subroutine used in quantum algorithms to estimate the phase of the eigenvalue of some unitary operator (in this case $G$) to some precision.
- Phase estimation relies on the fact that when the eigenvalue is written in the form $e^{2\pi ij\phi}$ for phase $\phi$, the inverse Fourier transform will transform

$$\frac{1}{\sqrt{N}} \sum_{j \in \{0,1\}^{\lceil \log_2 N \rceil}} e^{2\pi ij\phi} \left| j \right\rangle$$

to an approximation of $\phi$ in the form $\left| \tilde{\phi} \right\rangle$, where $\tilde{\phi}$ is the binary approximation of $\phi$.

# Quantum Counting Algorithm

To achieve $m$ bits of accuracy of $\theta$ with probability $1 - \epsilon$, the algorithm works on two registers. The first register is of size $t = m + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$, and the second register of size $n + 1$.

# Quantum Counting Algorithm

The algorithm is much like the phase estimation.

$$
\begin{aligned}
|0\rangle^{\otimes t}|0\rangle^{n+1} &\rightarrow \frac{1}{2^{t/2}} \sum_{k\in\{0,1\}^t} |k\rangle \frac{1}{2^{(n+1)/2}} \sum_{s\in\{0,1\}^{n+1}} |s\rangle && \text{Hadamards} \\
&\rightarrow \frac{1}{2^{t/2}} \sum_{k\in\{0,1\}^t} e^{2\pi i\phi k} |k\rangle \frac{1}{2^{(n+1)/2}} \sum_{s\in\{0,1\}^{n+1}} |s\rangle && \text{Controlled-}G \\
&= \frac{1}{2^{t/2}} \left(|0\rangle + e^{2\pi i 2^{t-1}\phi}|1\rangle\right)\left(|0\rangle + e^{2\pi i 2^{t-2}\phi}|1\rangle\right) \\
&\quad \ldots \left(|0\rangle + e^{2\pi i 2^{0}\phi}|1\rangle\right) \frac{1}{2^{(n+1)/2}} \sum_{s\in\{0,1\}^{n+1}} |s\rangle \\
&= \frac{1}{2^{t/2}} \left(|0\rangle + e^{2\pi i 0.\phi_t}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.\phi_{t-1}\phi_t}|1\rangle\right) \\
&\quad \ldots \left(|0\rangle + e^{2\pi i 0.\phi_1\ldots\phi_t}|1\rangle\right) \frac{1}{2^{(n+1)/2}} \sum_{s\in\{0,1\}^{n+1}} |s\rangle \\
&\rightarrow \left|\tilde{\phi}\right\rangle \frac{1}{2^{(n+1)/2}} \sum_{s\in\{0,1\}^{n+1}} |s\rangle && \text{Inverse Fourier transform} \\
&\rightarrow \tilde{\phi} && \text{Measurement on first register}
\end{aligned}
$$

# Quantum Counting Algorithm

The circuit of the algorithm is as follows,

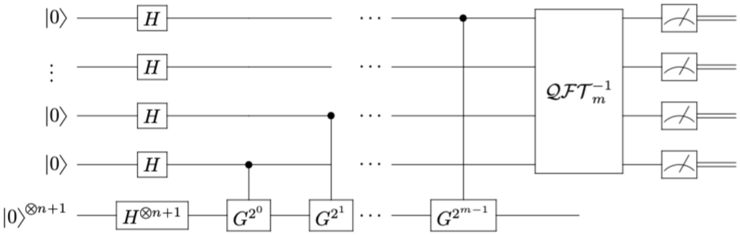



Figure: Quantum Circuit for the Quantum Counting algorithm (Nielsen & Chuang, 2002; Wikipedia, 2017b)

# Quantum Counting Algorithm

- If we choose $m = \lceil n/2 \rceil + 1$ and $\epsilon$ sufficiently small (such as $1/10$), then the algorithm will take $O(\sqrt{N})$ Grover iterations.
- This means that the function $f$ will only be called $O(\sqrt{N})$ times. Note that this is in contrast to a classical (deterministic or probabilistic) algorithm which will take $O(N)$ oracle calls to achieve the same accuracy.

# Quantum Counting Algorithm

## Theorem (Quantum Counting Correctness)

*Given a function* $f : \{0,1\}^n \to \{0,1\}$ *such that* $M = |\{x \in \{0,1\}^n : f(x) = 1\}|$ *and* $\sin^2\left(\frac{\theta}{2}\right) = \frac{M}{2N}$, *to find* $\theta$ *with* $m$ *bits of accuracy, with probability* $1 - \epsilon$ *the Quantum Counting Algorithm requires* $O(m + n + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil)$ *registers and* $O(\sqrt{N})$ *time.*

# Quantum Complexity Theory

Quantum complexity classes we are interested in are **BQP** (Bounded Error Quantum Polynomial time), an analogue of **BPP**, and **EQP** (Exact Quantum Polynomial Time).

### Definition

**BQP** is defined as the set of languages that are accepted with probability $\frac{2}{3}$ by some polynomial time Quantum Turing Machine.

### Definition

**EQP** is defined as the set of languages that are accepted with probability 1 by some polynomial time Quantum Turing Machine.

# Quantum Complexity Theory

To compare Quantum complexity classes to classical complexity classes, Bernstein & Vazirani (1997) proved

- **P** $\subseteq$ **EQP**
- **BPP** $\subset$ **BQP**
- **BQP** $\subseteq$ **PSPACE**

They additionally proved that there exist problems which are in **BQP** but are not in **BPP**, showing that Quantum Computing has strict advantages over classical deterministic (or probabilistic) computing.

To be continued...

# References I


Aaronson, S. (2005). Quantum computing, postselection, and probabilistic polynomial-time. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 461 (pp. 3473–3482).: The Royal Society.

Aaronson, S. (2013). *Quantum computing since Democritus*. Cambridge University Press.

Bernstein, E. & Vazirani, U. (1997). Quantum complexity theory. *SIAM Journal on Computing*, 26(5), 1411–1473.

Brassard, G., Høyer, P., & Tapp, A. (1998). Quantum counting. *Automata, languages and programming*, (pp. 820–831).

Dirac, P. A. M. (1939). A new notation for quantum mechanics. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 35 (pp. 416–418).: Cambridge University Press.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (pp. 212–219).: ACM.

# References II


Harrow, A. W., Hassidim, A., & Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15), 150502.

Nielsen, M. A. & Chuang, I. (2002). Quantum computation and quantum information.

Wikipedia (2017a). Grover's algorithm — Wikipedia, the free encyclopedia. [Online; accessed 17-November-2017].

Wikipedia (2017b). Quantum counting algorithm — Wikipedia, the free encyclopedia. [Online; accessed 17-November-2017].