Foundational Quantum Algorithms

Part 1

Deutsch's and Grover's algorithms

John Watrous
IBM

A standard picture of computation

A standard abstraction of computation looks like this:

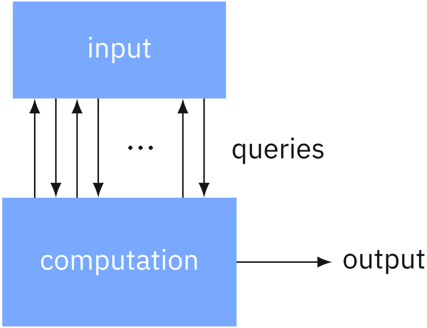
Different specific models of computation are studied, including *Turing machines* and *Boolean circuits.*

Key point

The *entire input* is provided to the computation (most typically as a string of bits); nothing is hidden from the computation.

The query model of computation

In the query model of computation, the input is made available in the form of a *function,* which the computation accesses by making *queries.*



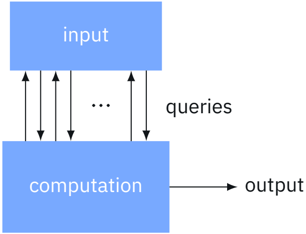
We often refer to the input as being provided by an *oracle* or *black box.*

The query model of computation

Throughout this lecture, the input to query problems is represented by a function

$$f : \Sigma^n \rightarrow \Sigma^m$$

where n and m are positive integers and $\Sigma = \{0, 1\}$.



Queries

To say that a computation *makes a query* means that it evaluates the function f once: $x \in \Sigma^n$ is selected, and the string $f(x) \in \Sigma^m$ is made available to the computation.

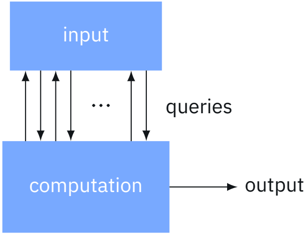
We typically measure the efficiency of query algorithms by counting the *number of queries* to the input they require.

The query model of computation

Throughout this lecture, the input to query problems is represented by a function

$$f : \Sigma^n \rightarrow \Sigma^m$$

where n and m are positive integers and $\Sigma = \{0, 1\}$.



Important points:

1. When we measure the efficiency of query algorithms by counting queries, we're *completely ignoring* the cost of implementing the oracle.

2. Query algorithms can nevertheless still be used in settings where it is necessary to *implement the oracle with a circuit* — and to consider the cost of doing this.

Examples of query problems

Or

Input: $f : \Sigma^n \rightarrow \Sigma$

Output: 1 if there exists a string $x \in \Sigma^n$ for which $f(x) = 1$

0 if there is no such string

Parity

Input: $f : \Sigma^n \rightarrow \Sigma$

Output: 0 if $f(x) = 1$ for an even number of strings $x \in \Sigma^n$

1 if $f(x) = 1$ for an odd number of strings $x \in \Sigma^n$

Minimum

Input: $f : \Sigma^n \rightarrow \Sigma^m$

Output: The string $y \in \{f(x) : x \in \Sigma^n\}$ that comes first in the natural (alphabetical) ordering of Σ^m

Examples of query problems

Sometimes we also consider query problems where we have a *promise* on the input. Inputs that don't satisfy the promise are considered as "don't care" inputs.

Unique search

Input:	$f : \Sigma^n \rightarrow \Sigma$
Promise:	There is a unique $z \in \Sigma^n$ for which $f(z) = 1$, with $f(x) = 0$ for all $x \neq z$
Output:	The string z

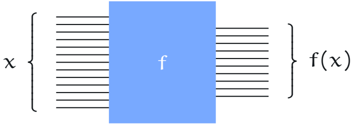
Or, Parity, Minimum, and Unique search are all very "natural" examples of query problems — but some query problems of interest aren't like this.

We sometimes consider very complicated and highly contrived problems, to look for extremes that reveal potential advantages of quantum computing.

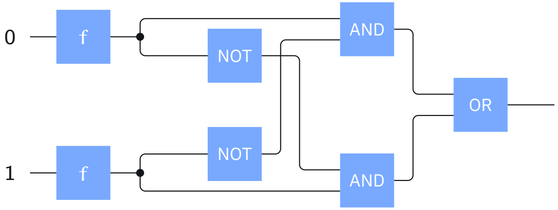
Query gates

For circuit models of computation, queries are made by *query gates.*

For Boolean circuits, query gates generally compute the input function f directly.



For example, the following circuit computes Parity for every $f : \Sigma \rightarrow \Sigma$.



Query gates

For the quantum circuit model, we choose a different definition for query gates that makes them *unitary* — allowing them to be applied to quantum states.

Definition

The *query gate* U_f for any function $f : \Sigma^n \rightarrow \Sigma^m$ is defined as

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

for all $x \in \Sigma^n$ and $y \in \Sigma^m$. (This gate is always unitary, for any choice of the function f.)

Notation

The string $y \oplus f(x)$ is the *bitwise XOR* of y and $f(x)$. For example:

$$001 \oplus 101 = 100$$

Query gates

For the quantum circuit model, we choose a different definition for query gates that makes them *unitary* — allowing them to be applied to quantum states.

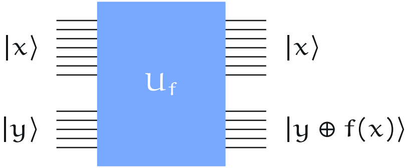
Definition

The *query gate* U_f for any function $f : \Sigma^n \to \Sigma^m$ is defined as

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

for all $x \in \Sigma^n$ and $y \in \Sigma^m$. (This gate is always unitary, for any choice of the function f.)

In circuit diagrammatic form U_f operates like this:



Deutsch's problem

Deutsch's problem is very simple — it's the Parity problem for functions of the form $f : \Sigma \to \Sigma$.

There are four functions of the form $f : \Sigma \to \Sigma$:

a	$f_1(a)$
0	0
1	0

a	$f_2(a)$
0	0
1	1

a	$f_3(a)$
0	1
1	0

a	$f_4(a)$
0	1
1	1

The functions f_1 and f_4 are *constant* while f_2 and f_3 are *balanced.*

Deutsch's problem

Input: $f : \Sigma \to \Sigma$

Output: 0 if f is constant, 1 if f is balanced

Deutsch's problem

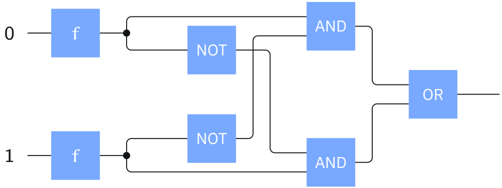
Deutsch's problem

Input: $f : \Sigma \to \Sigma$

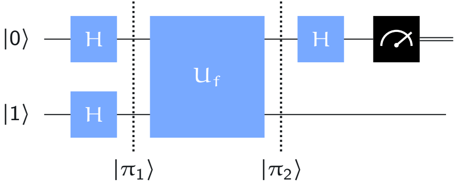
Output: 0 if f is constant, 1 if f is balanced

Every *classical* query algorithm must make 2 queries to f to solve this problem — learning just one of two bits provides no information about their parity.

Our query algorithm from earlier is therefore optimal among classical query algorithms for this problem.



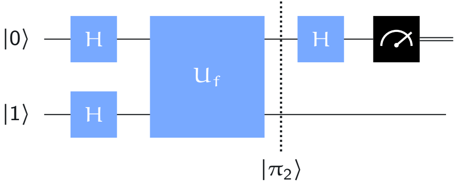
Deutsch's algorithm



$$|\pi_1\rangle = |-\rangle|+\rangle = \frac{1}{2}(|0\rangle - |1\rangle)|0\rangle + \frac{1}{2}(|0\rangle - |1\rangle)|1\rangle$$

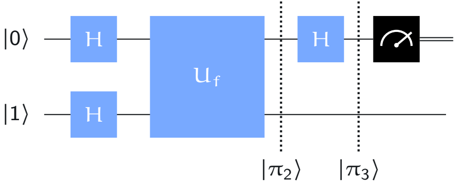
$$\begin{aligned}
|\pi_2\rangle &= \frac{1}{2}(|0 \oplus f(0)\rangle - |1 \oplus f(0)\rangle)|0\rangle + \frac{1}{2}(|0 \oplus f(1)\rangle - |1 \oplus f(1)\rangle)|1\rangle \\
&= \frac{1}{2}(-1)^{f(0)}(|0\rangle - |1\rangle)|0\rangle + \frac{1}{2}(-1)^{f(1)}(|0\rangle - |1\rangle)|1\rangle \\
&= |-\rangle\left(\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right)
\end{aligned}$$

Deutsch's algorithm



$$
\begin{aligned}
|\pi_2\rangle &= |-\rangle\left(\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right) \\
&= (-1)^{f(0)}|-\rangle\left(\frac{|0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle}{\sqrt{2}}\right) \\
&= \begin{cases} (-1)^{f(0)}|-\rangle|+\rangle & f(0)\oplus f(1) = 0 \\ (-1)^{f(0)}|-\rangle|-\rangle & f(0)\oplus f(1) = 1 \end{cases}
\end{aligned}
$$

Deutsch's algorithm

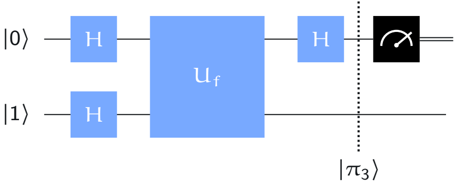


$$|\pi_2\rangle = \begin{cases} (-1)^{f(0)}|-\rangle|+\rangle & f(0) \oplus f(1) = 0 \\ (-1)^{f(0)}|-\rangle|-\rangle & f(0) \oplus f(1) = 1 \end{cases}$$

$$|\pi_3\rangle = \begin{cases} (-1)^{f(0)}|-\rangle|0\rangle & f(0) \oplus f(1) = 0 \\ (-1)^{f(0)}|-\rangle|1\rangle & f(0) \oplus f(1) = 1 \end{cases}$$

$$= (-1)^{f(0)}|-\rangle|f(0) \oplus f(1)\rangle$$

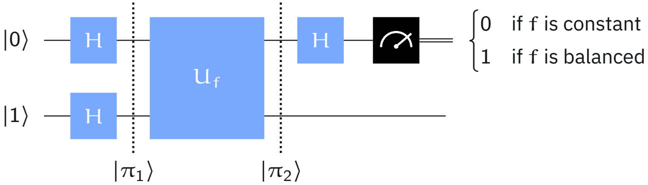
Deutsch's algorithm



$$\begin{cases} 0 & \text{if } f \text{ is constant} \\ 1 & \text{if } f \text{ is balanced} \end{cases}$$

$$|\pi_3\rangle = (-1)^{f(0)}|-\rangle|f(0) \oplus f(1)\rangle$$

Phase kickback



$$U_f(|-\rangle|a\rangle) = (-1)^{f(a)}|-\rangle|a\rangle \quad \longleftarrow$$ phase kickback

$$|\pi_1\rangle = |-\rangle|+\rangle$$

$$|\pi_2\rangle = U_f(|-\rangle|+\rangle) = \frac{1}{\sqrt{2}}U_f(|-\rangle|0\rangle) + \frac{1}{\sqrt{2}}U_f(|-\rangle|1\rangle)$$

$$= |-\rangle\left(\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right)$$

Unstructured search

Suppose we're given a function

$$f : \Sigma^n \rightarrow \Sigma$$

as an oracle (or that we can compute efficiently).

Our goal is to find a *solution,* which is a binary string $x \in \Sigma^n$ for which $f(x) = 1$.

Search

Input: $f : \Sigma^n \rightarrow \Sigma$

Output: a string $x \in \Sigma^n$ satisfying $f(x) = 1$, or "no solution" if no such strings exist

This is *unstructured* search because f is arbitrary — there's *no promise* and we can't rely on it having a structure that makes finding solutions easy.

Algorithms for search

Search

Input: $f : \Sigma^n \to \Sigma$

Output: a string $x \in \Sigma^n$ satisfying $f(x) = 1$, or "no solution" if no such strings exist

Hereafter let us write $N = 2^n$. By iterating through all $x \in \Sigma^n$ and evaluating f on each one, we can solve Search with N queries.

This is the best we can do with a *deterministic* algorithm.

Probabilistic algorithms offer minor improvements, but still require a number of queries linear in N.

Grover's algorithm is a *quantum algorithm* for Search requiring $O(\sqrt{N})$ queries.

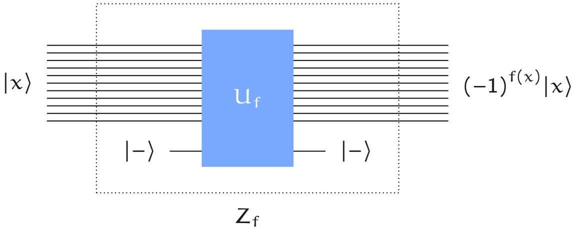
Phase query gates

We assume that we have access to the function $f : \Sigma^n \rightarrow \Sigma$ through a query gate:

$$U_f : |a\rangle|x\rangle \mapsto |a \oplus f(x)\rangle|x\rangle \qquad \text{(for all } a \in \Sigma \text{ and } x \in \Sigma^n\text{)}$$

A *phase query gate* for f operates like this:

$$Z_f : |x\rangle \mapsto (-1)^{f(x)}|x\rangle \qquad \big(\text{for all } x \in \Sigma^n\big)$$



Phase query gates

A *phase query gate* for f operates like this:

$$Z_f : |x\rangle \mapsto (-1)^{f(x)}|x\rangle \qquad \big(\text{for all } x \in \Sigma^n\big)$$

We're also going to need a phase query gate for the n-bit OR function:

$$\mathrm{OR}(x) = \begin{cases} 0 & x = 0^n \\ 1 & x \neq 0^n \end{cases} \qquad (\text{for all } x \in \Sigma^n)$$

$$Z_{\mathrm{OR}}|x\rangle = \begin{cases} |x\rangle & x = 0^n \\ -|x\rangle & x \neq 0^n \end{cases} \qquad (\text{for all } x \in \Sigma^n)$$

Grover's algorithm (description)

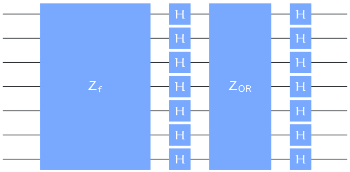
Grover's algorithm

1. *Initialize:* set n qubits to the state $H^{\otimes n}|0^n\rangle$.
2. *Iterate:* apply the *Grover operation* t times (for t to be specified later).
3. *Measure:* a standard basis measurement yields a candidate solution.

The Grover operation is defined like this:

$$G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$$

Z_f is the phase query gate for f and Z_{OR} is the phase query gate for the n-bit OR function.



Grover's algorithm (description)

Grover's algorithm

1. *Initialize:* set n qubits to the state $H^{\otimes n}|0^n\rangle$.
2. *Iterate:* apply the *Grover operation* t times (for t to be specified later).
3. *Measure:* a standard basis measurement yields a candidate solution.

The Grover operation is defined like this:

$$G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$$

Z_f is the phase query gate for f and Z_{OR} is the phase query gate for the n-bit OR function.

A typical way that Grover's algorithm can be applied:

1. Choose the number of iterations t (next section).
2. Run Grover's algorithm with t iterations to get a candidate solution x.
3. Check the solution. If $f(x) = 1$ then output x, otherwise either run Grover's algorithm again (possibly with a different t) or report "no solutions."

Solutions and non-solutions

We'll refer to the n qubits being used for Grover's algorithm as a register Q.

We're interested in what happens when Q is initialized to the state $H^{\otimes n}|0^n\rangle$ and the Grover operation G is performed iteratively.

$$G = H^{\otimes n} Z_{OR} H^{\otimes n} Z_f$$

These are the sets of non-solutions and solutions:

$$A_0 = \{x \in \Sigma^n : f(x) = 0\}$$
$$A_1 = \{x \in \Sigma^n : f(x) = 1\}$$

We will be interested in *uniform superpositions* over these sets:

$$|A_0\rangle = \frac{1}{\sqrt{|A_0|}} \sum_{x \in A_0} |x\rangle \qquad |A_1\rangle = \frac{1}{\sqrt{|A_1|}} \sum_{x \in A_1} |x\rangle$$

Analysis: basic idea

$$A_0 = \{x \in \Sigma^n : f(x) = 0\} \qquad A_1 = \{x \in \Sigma^n : f(x) = 1\}$$

$$|A_0\rangle = \frac{1}{\sqrt{|A_0|}} \sum_{x \in A_0} |x\rangle \qquad |A_1\rangle = \frac{1}{\sqrt{|A_1|}} \sum_{x \in A_1} |x\rangle$$

The register Q is first initialized to this state:

$$|u\rangle = H^{\otimes n}|0^n\rangle = \frac{1}{\sqrt{N}} \sum_{x \in \Sigma^n} |x\rangle$$

This state is contained in the subspace spanned by $|A_0\rangle$ and $|A_1\rangle$:

$$|u\rangle = \sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle$$

The state of Q *remains in this subspace* after every application of the Grover operation G.

Action of the Grover operation

We can better understand the Grover operation by splitting it into two parts:

$$G = \left(H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n}\right)\left(Z_f\right)$$

1. Recall that Z_f is defined like this:

   $$Z_f|x\rangle = (-1)^{f(x)}|x\rangle \qquad (\text{for all } x \in \Sigma^n)$$

   Its action on $|A_0\rangle$ and $|A_1\rangle$ is simple:

   $$Z_f|A_0\rangle = |A_0\rangle$$
   $$Z_f|A_1\rangle = -|A_1\rangle$$

Action of the Grover operation

We can better understand the Grover operation by splitting it into two parts:

$$G = (H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n})(Z_f)$$

2. The operation Z_{OR} is defined like this:

$$Z_{\mathrm{OR}}|x\rangle = \begin{cases} |x\rangle & x = 0^n \\ -|x\rangle & x \neq 0^n \end{cases} \qquad \text{(for all } x \in \Sigma^n\text{)}$$

Here's an alternative way to express Z_{OR}:

$$Z_{\mathrm{OR}} = 2|0^n\rangle\langle 0^n| - \mathbb{1}$$

Using this expression, we can write $H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n}$ like this:

$$H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} = H^{\otimes n}(2|0^n\rangle\langle 0^n| - \mathbb{1})H^{\otimes n} = 2|u\rangle\langle u| - \mathbb{1}$$

Action of the Grover operation

$$Z_f|A_0\rangle = |A_0\rangle$$
$$Z_f|A_1\rangle = -|A_1\rangle$$

$$|u\rangle = \sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle$$

$$\begin{aligned}
G|A_0\rangle &= \bigl(2|u\rangle\langle u| - \mathbb{1}\bigr)Z_f|A_0\rangle \\
&= \bigl(2|u\rangle\langle u| - \mathbb{1}\bigr)|A_0\rangle \\
&= 2\sqrt{\frac{|A_0|}{N}}|u\rangle - |A_0\rangle \\
&= 2\sqrt{\frac{|A_0|}{N}}\left(\sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle\right) - |A_0\rangle \\
&= \frac{|A_0| - |A_1|}{N}|A_0\rangle + \frac{2\sqrt{|A_0|\cdot|A_1|}}{N}|A_1\rangle
\end{aligned}$$

Action of the Grover operation

$$Z_f|A_0\rangle = |A_0\rangle$$
$$Z_f|A_1\rangle = -|A_1\rangle$$

$$|u\rangle = \sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle$$

$$G|A_0\rangle = \frac{|A_0| - |A_1|}{N}|A_0\rangle + \frac{2\sqrt{|A_0| \cdot |A_1|}}{N}|A_1\rangle$$

$$\begin{aligned}
G|A_1\rangle &= \left(2|u\rangle\langle u| - \mathbb{1}\right)Z_f|A_1\rangle \\
&= \left(\mathbb{1} - 2|u\rangle\langle u|\right)|A_1\rangle \\
&= |A_1\rangle - 2\sqrt{\frac{|A_1|}{N}}|u\rangle \\
&= |A_1\rangle - 2\sqrt{\frac{|A_0|}{N}}\left(\sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle\right) \\
&= -\frac{2\sqrt{|A_0| \cdot |A_1|}}{N}|A_0\rangle + \frac{|A_0| - |A_1|}{N}|A_1\rangle
\end{aligned}$$

Action of the Grover operation

$$Z_f|A_0\rangle = |A_0\rangle \qquad |u\rangle = \sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle$$
$$Z_f|A_1\rangle = -|A_1\rangle$$

$$G|A_0\rangle = \frac{|A_0| - |A_1|}{N}|A_0\rangle + \frac{2\sqrt{|A_0| \cdot |A_1|}}{N}|A_1\rangle$$

$$G|A_1\rangle = -\frac{2\sqrt{|A_0| \cdot |A_1|}}{N}|A_0\rangle + \frac{|A_0| - |A_1|}{N}|A_1\rangle$$

The action of G on $\text{span}\{|A_0\rangle, |A_1\rangle\}$ can be described by a 2×2 matrix:

$$M = \begin{array}{cc} \begin{pmatrix} \frac{|A_0|-|A_1|}{N} & -\frac{2\sqrt{|A_0|\cdot|A_1|}}{N} \\ \frac{2\sqrt{|A_0|\cdot|A_1|}}{N} & \frac{|A_0|-|A_1|}{N} \end{pmatrix} & \begin{array}{c} |A_0\rangle \\ |A_1\rangle \end{array} \\ \begin{array}{cc} \quad |A_0\rangle & \qquad\qquad |A_1\rangle \end{array} & \end{array}$$

Rotation by an angle

The action of G on $\mathrm{span}\{|A_0\rangle, |A_1\rangle\}$ can be described by a 2×2 matrix:

$$M = \begin{pmatrix} \frac{|A_0|-|A_1|}{N} & -\frac{2\sqrt{|A_0|\cdot|A_1|}}{N} \\ \frac{2\sqrt{|A_0|\cdot|A_1|}}{N} & \frac{|A_0|-|A_1|}{N} \end{pmatrix} = \begin{pmatrix} \sqrt{\frac{|A_0|}{N}} & -\sqrt{\frac{|A_1|}{N}} \\ \sqrt{\frac{|A_1|}{N}} & \sqrt{\frac{|A_0|}{N}} \end{pmatrix}^2$$

This is a *rotation* matrix.

$$\begin{pmatrix} \sqrt{\frac{|A_0|}{N}} & -\sqrt{\frac{|A_1|}{N}} \\ \sqrt{\frac{|A_1|}{N}} & \sqrt{\frac{|A_0|}{N}} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \qquad \theta = \sin^{-1}\left(\sqrt{\frac{|A_1|}{N}}\right)$$

$$M = \begin{pmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{pmatrix}$$

Rotation by an angle

$$M = \begin{pmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{pmatrix} \qquad \theta = \sin^{-1}\left(\sqrt{\frac{|A_1|}{N}}\right)$$

After the initialization step, this is the state of the register Q:

$$|u\rangle = \sqrt{\frac{|A_0|}{N}}|A_0\rangle + \sqrt{\frac{|A_1|}{N}}|A_1\rangle = \cos(\theta)|A_0\rangle + \sin(\theta)|A_1\rangle$$

Each time the Grover operation G is performed, the state of Q is rotated by an angle 2θ:

$$\begin{aligned} |u\rangle &= \cos(\theta)|A_0\rangle + \sin(\theta)|A_1\rangle \\ G|u\rangle &= \cos(3\theta)|A_0\rangle + \sin(3\theta)|A_1\rangle \\ G^2|u\rangle &= \cos(5\theta)|A_0\rangle + \sin(5\theta)|A_1\rangle \\ &\vdots \\ G^t|u\rangle &= \cos((2t+1)\theta)|A_0\rangle + \sin((2t+1)\theta)|A_1\rangle \end{aligned}$$

Geometric picture

Main idea

The operation $G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$ is a composition of *two reflections:*

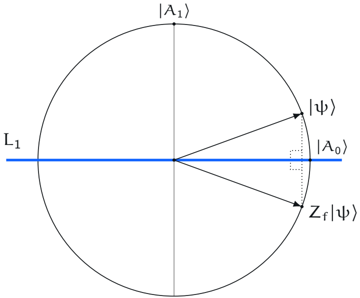
$$Z_f \quad \text{and} \quad H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n}$$

Composing two reflections yields a *rotation.*

1. Recall that Z_f has this action on $|A_0\rangle$ and $|A_1\rangle$:

$$Z_f|A_0\rangle = |A_0\rangle$$
$$Z_f|A_1\rangle = -|A_1\rangle$$

This is a *reflection* about the line L_1 parallel to $|A_0\rangle$.



Geometric picture

Main idea

The operation $G = H^{\otimes n} Z_{OR} H^{\otimes n} Z_f$ is a composition of *two reflections:*

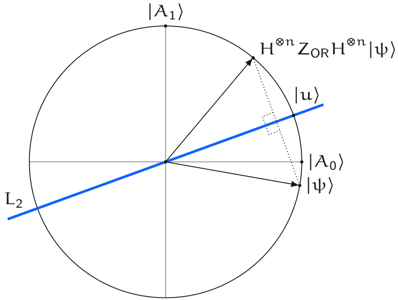
$$Z_f \quad \text{and} \quad H^{\otimes n} Z_{OR} H^{\otimes n}$$

Composing two reflections yields a *rotation.*

2. The operation $H^{\otimes n} Z_{OR} H^{\otimes n}$ can be expressed like this:

$$H^{\otimes n} Z_{OR} H^{\otimes n} = 2|u\rangle\langle u| - \mathbb{1}$$

Again this is a *reflection,* this time about the line L_2 parallel to $|u\rangle$.



Geometric picture

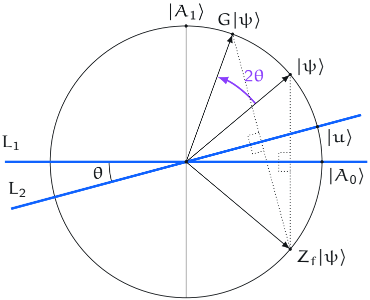
Main idea

The operation $G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$ is a composition of *two reflections:*

$$Z_f \quad \text{and} \quad H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n}$$

Composing two reflections yields a *rotation.*

When we compose two reflections, we obtain a *rotation* by twice the angle between the lines of reflection.



Choosing the number of iterations

Consider any quantum state of this form:

$$\alpha|A_0\rangle + \beta|A_1\rangle$$

Measuring yields a solution $x \in A_1$ with probability $|\beta|^2$.

The state of Q after t iterations in Grover's algorithm:

$$\cos\big((2t+1)\theta\big)|A_0\rangle + \sin\big((2t+1)\theta\big)|A_1\rangle \qquad \theta = \sin^{-1}\left(\sqrt{\frac{|A_1|}{N}}\right)$$

Measuring after t iterations gives an outcome $x \in A_1$ with probability

$$\sin^2\big((2t+1)\theta\big)$$

We wish to maximize this probability — so we may view that $|A_1\rangle$ is our *target state.*

Choosing the number of iterations

The state of Q after t iterations in Grover's algorithm:

$$\cos\big((2t+1)\theta\big)|\mathcal{A}_0\rangle + \sin\big((2t+1)\theta\big)|\mathcal{A}_1\rangle \qquad \theta = \sin^{-1}\left(\sqrt{\frac{|\mathcal{A}_1|}{N}}\right)$$

Measuring after t iterations gives an outcome $x \in \mathcal{A}_1$ with probability

$$\sin^2\big((2t+1)\theta\big)$$

To make this probability close to 1 and minimize t, we will aim for

$$(2t+1)\theta \approx \frac{\pi}{2} \qquad \Longleftrightarrow \qquad t \approx \frac{\pi}{4\theta} - \frac{1}{2} \quad \xrightarrow{\text{closest integer}} \quad t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor$$

Important considerations:

- t must be an integer
- θ depends on the number of solutions $s = |\mathcal{A}_1|$

Unique search

$$(2t+1)\theta \approx \frac{\pi}{2} \qquad \Leftarrow \qquad t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor$$

Unique search

Input: $f : \Sigma^n \rightarrow \Sigma$

Promise: There is a unique $z \in \Sigma^n$ for which $f(z) = 1$, with $f(x) = 0$ for all $x \neq z$

Output: The string z

For Unique search we have $s = |A_1| = 1$ and therefore

$$\theta = \sin^{-1}\left(\sqrt{\frac{1}{N}}\right) \approx \sqrt{\frac{1}{N}}$$

Substituting $\theta \approx 1/\sqrt{N}$ into our expression for t gives

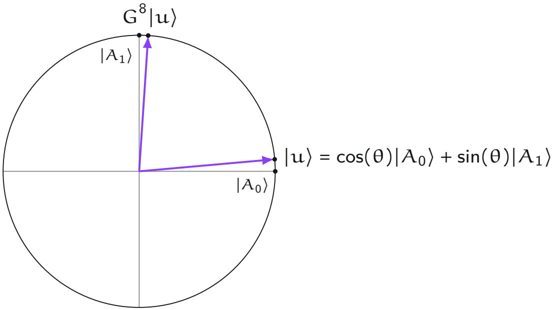
$$t \approx \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor \qquad \leftarrow O(\sqrt{N}) \text{ queries}$$

Unique search

Example: $N = 128$

$$\theta = \sin^{-1}\left(\frac{1}{\sqrt{N}}\right) = 0.0885\cdots$$

$$t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor = 8$$



Unique search

$$\theta = \sin^{-1}\left(\sqrt{\frac{1}{N}}\right) \qquad t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor$$

Measuring after t iterations gives the (unique) outcome $x \in A_1$ with probability

$$p(N,1) = \sin^2\big((2t+1)\theta\big)$$

Success probabilities for Unique search

N	$p(N,1)$
2	.5
4	1.0
8	.9453125
16	.9613190
32	.9991823
64	.9965857

N	$p(N,1)$
128	.9956199
256	.9999470
512	.9994480
1024	.9994612
2048	.9999968
4096	.9999453

Unique search

Measuring after t iterations gives the (unique) outcome $x \in A_1$ with probability

$$p(N, 1) = \sin^2\big((2t + 1)\theta\big)$$

Success probabilities for Unique search

N	$p(N,1)$
2	.5
4	1.0
8	.9453125
16	.9613190
32	.9991823
64	.9965857

N	$p(N,1)$
128	.9956199
256	.9999470
512	.9994480
1024	.9994612
2048	.9999968
4096	.9999453

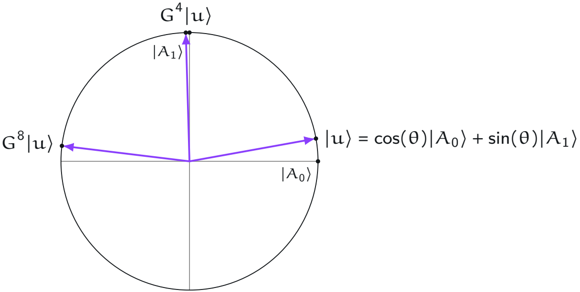
It can be proved analytically that $p(N, 1) \geq 1 - \frac{1}{N}$.

Multiple solutions

Example: $N = 128$, $s = 4$

$$\theta = \sin^{-1}\left(\sqrt{\frac{s}{N}}\right) = 0.1777\cdots$$

$$t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor = 4$$



Number of queries

$$\theta = \sin^{-1}\left(\sqrt{\frac{s}{N}}\right) \qquad t = \left\lfloor \frac{\pi}{4\theta} \right\rfloor$$

Each iteration of Grover's algorithm requires 1 query (or evaluations of f). How does the number of queries t depend on N and s?

$$\sin^{-1}(x) \geq x \qquad \big(\text{for every } x \in [0,1]\big)$$

$$\theta = \sin^{-1}\left(\sqrt{\frac{s}{N}}\right) \geq \sqrt{\frac{s}{N}}$$

$$t \leq \frac{\pi}{4\theta} \leq \frac{\pi}{4}\sqrt{\frac{N}{s}}$$

$$t = O\left(\sqrt{\frac{N}{s}}\right)$$

Unknown number of solutions

What do we do if we don't know the number of solutions in advance?

A simple approach

Choose the number of iterations $t \in \{1, \ldots, \lfloor \pi\sqrt{N}/4 \rfloor\}$ *uniformly at random.*

- The probability to find a solution (if one exists) will be at least 40%. (Repeat to boost probability.)
- The number of queries (or evaluations of f) is $O(\sqrt{N})$.

Unknown number of solutions

What do we do if we don't know the number of solutions in advance?

A simple approach

Choose the number of iterations $t \in \{1, \ldots, \lfloor \pi\sqrt{N}/4 \rfloor\}$ *uniformly at random.*

A more sophisticated approach

1. Set $T = 1$.
2. Run Grover's algorithm with $t \in \{1, \ldots, T\}$ chosen uniformly at random.
3. If a solution is found, output it and stop.
   Otherwise, increase T and return to step 2 (or report "no solution").

- The rate of increase of T must be carefully balanced: slower rates require more queries, higher rates decrease success probability. $T \leftarrow \lceil \frac{5}{4} T \rceil$ works.
- If the number of solutions is $s \geq 1$, then the number of queries (or evaluations of f) required is $O(\sqrt{N/s})$. If there are no solutions, $O(\sqrt{N})$ queries are required.

Concluding remarks

- Grover's algorithm is *asymptotically optimal.*
- Grover's algorithm is *broadly applicable.*
- The technique used in Grover's algorithm can be *generalized.*