Grover's Search Algorithm

Peter Young

(Dated: November 26, 2019)

I. INTRODUCTION

Grover's algorithm discussed in this handout is of a different type from Shor's algorithm. Whereas Shor (and related algorithms like Simon's) depend on a quantum Fourier transform (of some sort), Grover's algorithm involves *amplitude amplification*.

To motivate Grover's algorithm consider looking up someone in a phone directory. It is straightforward to lookup a person's phone number in a directory if one is given the name, because names are in alphabetic order. To locate the name systematically one would go to the midpoint of the list, see which half the name is in, divide that half in two, again see which half the number is in, and so on. One continues this procedure until the size of the region containing the desired entry is just one. For a directory with N entries, this *bisection* method takes $\log_2 N$ operations (rounded up to the nearest integer if N is not a power of 2) since one halves the range over which the special entry could be at each stage.

By contrast, suppose one is given the number and asked which person has that number. Since the numbers are not ordered, all one can do is go through the entries one at a time and see if each one has the desired name. On average this would take $N/2$ operations before success was achieved.

If N is large this is a huge difference. For example if $N = 10^6$ then $\log_2 N = 20$, to be compared with $N/2 = 5 \times 10^5$.

The quantum search algorithm algorithm discussed in this handout, due to Grover, is often presented as such a search of an unstructured database.[1] Grover's algorithm requires a quantum computer running a subroutine for which the input is a number corresponding to an entry in the database, and which performs a test to see if this is the special value being searched for. For large N it will determine the special value, with probability close to 1, by calling the subroutine only $(\pi/4)\sqrt{N}$ times. This is a *quadratic* speedup compared with a classical computer. While less spectacular than the exponential speedup of Shor's algorithm, it can potentially be applied to a wide variety of problems[2].

[1] Though it is doubtful it would ever be used in this way since it would be a very extravagant use of a precious resource to use qubits to store classical information.

[2] However, most applications of practical interest have some structure, whereas Grover is designed for problems

II. GROVER'S ALGORITHM

A. The Black Box (Oracle)

To formulate the problem we consider n-bit integers, one of which, a, is special. The goal is to find a. We need a subroutine which outputs 1 if the input value x is equal to a and outputs 0 otherwise, i.e.

$$f(x) = 0, \quad (x \neq a); \quad f(a) = 1\,. \tag{1}$$

As usual, the function will be determined from a unitary transformation acting on an n-qubit input register and a 1-qubit output register which is flipped or not flipped depending on whether x is the special number a or not:

$$U|x\rangle_n|y\rangle_1 = |x\rangle_n|y \oplus f(x)\rangle_1\,. \tag{2}$$

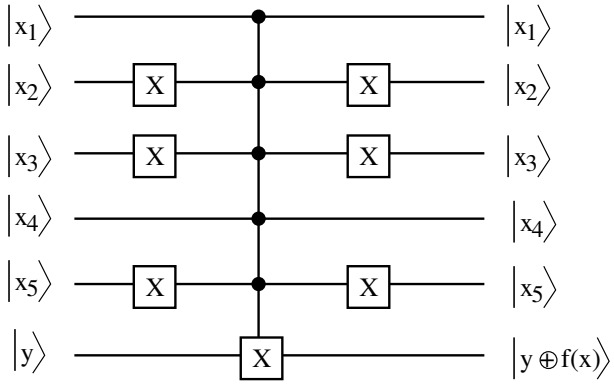

FIG. 1: A black box circuit that executes the first part of a Grover iteration, Eq. (2), in which $f(x) = 0$ if $x \neq a$ and $f(a) = 1$, for the case of $n = 5$ qubits and where the special number a is 01001. The 6-qubit gate in the center is a five-fold-controlled-NOT gate which acts to flip the target qubit y only if all the control qubits are 1. The X gates on the left flip qubits x_2, x_3 and x_5. Hence the target qubit is flipped if and only if $x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 1, x_5 = 0$, which are the bits of a. The X-gates on the right flip back those qubits which had previously been flipped, thus leaving the input register (the $|x_i\rangle$) unchanged.

A simple example of such a function for $n = 5$ and $a = 01001$ is shown in Fig. 1. (Recall that x_1 is the least significant bit.) The target bit is flipped only if all five of the control bits are one,

with no structure. In most cases that Grover could potentially be applied, the structure of the problem allows an efficient classical algorithm which outperforms Grover. Thus it is debated whether the Grover algorithm would be of practical utility, even if one could overcome the severe experimental difficulties of building a large quantum computer.

which requires $x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 1, x_5 = 0$ (the bits of a). How to construct such a five-fold-controlled-NOT gate out of 1-qubit and 2-qubit elementary gates is discussed in Mermin[1] §4.2.

Such a black box function is often called an oracle. The oracle can give a yes or no answer as to whether the input is the special number. For the implementation in Fig. 1 you might object and say that surely we *already* know the answer since it is built into the quantum device by placing the X-gates only on those qubits where the special number has a 0 bit. Can't we just open up the black box and look? In this case the answer is "yes". However, the implementation of the black box in Fig. 1 is just a simple example. The Grover algorithm can also be applied in more useful situations where the value of $f(x)$ is *not* built in explicitly but has to be calculated in a non-trivial way. Examples are discussed in Mermin[1] and Nielsen and Chuang[2].

It is useful to initially set the output bit y to be 1 and then apply a Hadamard gate before applying U. The output bit is then

$$H|1\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle - |1\rangle \right) . \tag{3}$$

If the result of U is $f(x) = 0$ than the output bit is unchanged. If the result is $f(x) = 1$ then $|0\rangle \to |1\rangle$ and vice-versa, so the output bit changes sign. Consequently

$$U \left(|x\rangle_n \otimes H|1\rangle_1 \right) = (-1)^{f(x)} |x\rangle_n \otimes H|1\rangle_1 . \tag{4}$$

We see that the output bit remains unchanged and so, for simplicity, can be ignored in what follows. Thus we consider the following unitary operator $\hat{O}$ acting only on the n-qubit input register:

$$\hat{O}|x\rangle = (-1)^{f(x)}|x\rangle = \begin{cases} |x\rangle, & x \neq a, \\ -|a\rangle, & x = a. \end{cases} \tag{5}$$

Since U, and hence $\hat{O}$, are linear acting with $\hat{O}$ on a superposition changes the sign of the component along $|a\rangle$ but leaves the component perpendicular to $|a\rangle$ unchanged. Hence if

$$|\psi\rangle = \sum_x c_x |x\rangle, \tag{6}$$

then

$$|\psi'\rangle \equiv \hat{O}|\psi\rangle = \sum_{x \neq a} c_x |x\rangle - c_a |a\rangle = \sum_x c_x |x\rangle - 2c_a |a\rangle = |\psi\rangle - 2|a\rangle\langle a|\psi\rangle \tag{7}$$

since $c_a = \langle a|\psi\rangle$. You should check that $\langle a|\psi'\rangle = -\langle a|\psi\rangle (= -c_a)$ and, for $x \neq a$, that $\langle x|\psi'\rangle = \langle x|\psi\rangle (= c_x)$, as required. You should also verify that $|\psi'\rangle$ is correctly normalized if $|\psi\rangle$ and $|a\rangle$ are.

We initialize the n-qubit input register into a uniform superposition of all basis states by acting with n Hadamards on $|0\rangle_n$:

$$|\psi_0\rangle = H^{\otimes n}|0\rangle_n = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n \,, \tag{8}$$

in which we recall that bits of the n-digit number x are the values of the qubits in the computational basis.

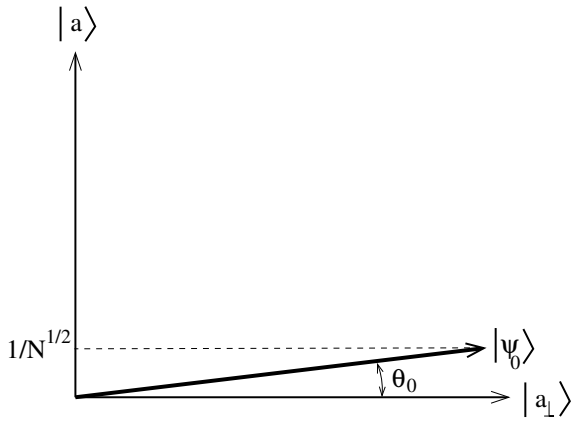

FIG. 2: Projection of the 2^N-dimensional space on to a 2-dimensional space spanned by $|a\rangle$ and $|a_\perp\rangle$, the latter being a (normalized) equal linear combination of all basis states except for $|a\rangle$ itself, see Eq. (9). The vector in bold is the equal linear combination of all basis states $|\psi_0\rangle$, see Eq. (8). The vector $|\psi_0\rangle$ has a projection $1/\sqrt{N}$ on to $|a\rangle$, so $\sin\theta_0 = 1/\sqrt{N}$, where θ_0 is the angle between $|\psi_0\rangle$ and $|a_\perp\rangle$.

It turns out that all the states generated during the Grover algorithm can be written as a linear combination of $|a\rangle$ and a *uniform* superposition of all basis states perpendicular to $|a\rangle$, i.e.

$$|a_\perp\rangle = \frac{1}{\sqrt{2^n-1}} \sum_{x\neq a} |x\rangle_n \,. \tag{9}$$

Hence the states generated by the algorithm can be conveniently drawn as vectors in the 2-dimensional space spanned by these two basis vectors, see Fig. 2. The component of the initial state $|\psi_0\rangle$ in the direction of $|a\rangle$ is $1/2^{n/2} = 1/\sqrt{N}$, so the vector for $|\psi_0\rangle$ makes an angle θ_0 with the $|a_\perp\rangle$ axis where

$$\sin\theta_0 = \frac{1}{\sqrt{N}} \,. \tag{10}$$

In practice we are interested in large N for which we can replace $\sin\theta_0$ by θ_0.

As shown in Eq. (7) the action of $\hat{O}$ is to invert the component along $|a\rangle$ of the vector it acts on, while keeping the component perpendicular to $|a\rangle$ unchanged. The net effect is to *reflect* about the the $|a_\perp\rangle$ axis. Figure 3 shows the effect of $\hat{O}$ on the initial state $|\psi_0\rangle$.

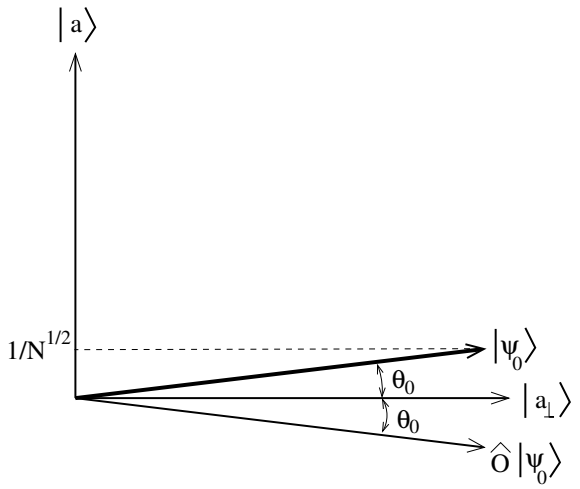

FIG. 3: Figure showing that the action of the operator $\hat{O}$ is to reflect the state it is acting on, in this case $|\psi_0\rangle$, about the $|a_\perp\rangle$ axis.

The goal of the Grover algorithm is to iteratively rotate the vector representing the state of the input register from its initial direction, that of $|\psi_0\rangle$ (which is close to the $|a_\perp\rangle$ axis), to a direction close to the $|a\rangle$ axis, because a measurement of it will then give a with a high probability. To perform this rotation we need a second unitary operation that we will discuss in the next subsection.

B. The second step of the Grover iteration

The second stage of a single Grover iteration is independent of the special number a. It changes the sign of the component perpendicular to the initial state $|\psi_0\rangle$ and keeps unchanged the component along $|\psi_0\rangle$. Denoting this operation by $\hat{S}$ we have

$$|\phi\rangle \to |\phi'\rangle = \hat{S}|\phi\rangle = 2|\psi_0\rangle\langle\psi_0|\phi\rangle - |\phi\rangle, \tag{11}$$

where $|\phi\rangle$ is an arbitrary state. You should check that $\langle\psi_0|\phi'\rangle = \langle\psi_0|\phi\rangle$, so the component along $|\psi_0\rangle$ is unchanged, and for a state $|\mu\rangle$ which is orthogonal to $|\psi_0\rangle$ (i.e. $\langle\mu|\psi_0\rangle = 0$), $\langle\mu|\phi'\rangle = -\langle\mu|\phi\rangle$, showing that the component perpendicular to $|\psi_0\rangle$ has the sign changed. The net result if to reflect $|\phi\rangle$ about the direction of $|\psi_0\rangle$. Figure 4 shows the effects of $\hat{S}$ acting on the state generated by $\hat{O}|\psi_0\rangle$. The combined effect of $\hat{O}$ followed by $\hat{S}$ is to rotate the initial state $|\psi_0\rangle$ by $2\theta_0$ in an anti-clockwise direction, i.e. $2\theta_0$ towards the desired direction of the $|a\rangle$ axis. The combination of these two operations is called a Grover iteration.

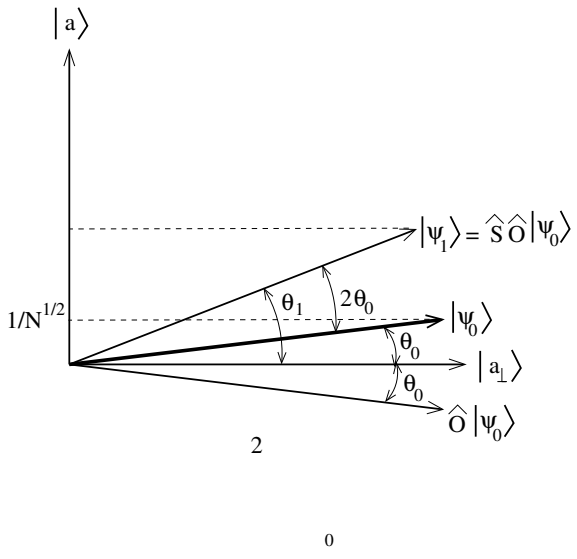

FIG. 4: Figure showing that the action of the operator $\hat{S}$ is to reflect the state it is acting on, in this case $\hat{O}|\psi_0\rangle$, about the direction of $|\psi_0\rangle$ which is defined in Eq. (8). The net result of the two operations, $\hat{O}$ followed by $\hat{S}$, is to rotate the direction of $|\psi_0\rangle$ by $2\theta_0$ in an anti-clockwise direction. We will call the new state $|\psi_1\rangle$. It is at an angle $\theta_1 = \theta_0 + 2\theta_0$ to the $|a_\perp\rangle$ axis.

The effect of the first Grover iteration, therefore, is to take the initial state $|\psi_0\rangle$ and rotate it anti-clockwise by $2\theta_0$. We will call the resulting state $|\psi_1\rangle$. It is at an angle θ_1 to the $|a_\perp\rangle$ axis, where

$$\theta_1 = \theta_0 + 2\theta_0, \tag{12}$$

see Fig. 4.

C. Subsequent iterations

Subsequent Grover iterations perform the same two steps: $\hat{O}$ which reflects about $|a_\perp\rangle$ followed by $\hat{S}$ which reflects about $|\psi_0\rangle$. The overall circuit implementing the Grover algorithm is shown in Fig. 5.

If m iterations have already been done, so the current state is $|\psi_m\rangle$, Fig. 6 shows the effect of doing an additional iteration. Now $|\psi_m\rangle$ makes an angle θ_m with the $|a_\perp\rangle$ axis, so $\hat{O}$ rotates the direction by $2\theta_m$ clockwise, while $\hat{S}$ rotates it by $2(\theta_m + \theta_0)$ anti-clockwise. The net result is a rotation by $2\theta_0$ (independent of θ_m) anti-clockwise, which is towards the desired direction, $|a\rangle$, i.e.

$$\theta_{m+1} = \theta_m + 2\theta_0, \tag{13}$$

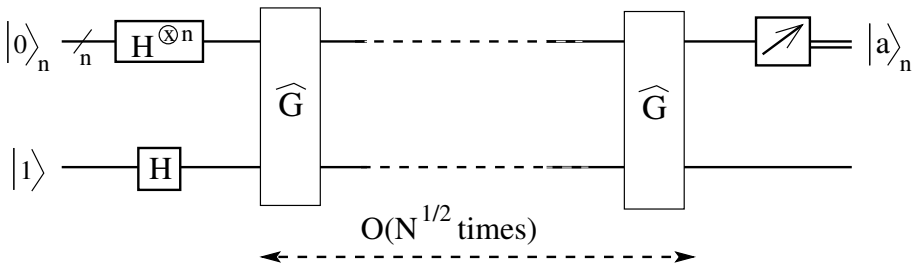

FIG. 5: Circuit implementing the Grover algorithm. $\hat{G}$ is the Grover operator and is given by $\hat{G} = \hat{S}\hat{O}$ where $\hat{O}$ and $\hat{S}$, given by Eqs. (7) and (11) respectively, act only on the n input qubits (the upper line). The output qubit (the lower line) remains unchanged by $\hat{G}$. After $O(\sqrt{N})$ iterations of the Grover operator, the result of a measurement on the input qubits is the special value a with high probability.

which gives

$$\theta_m = (2m+1)\theta_0 \tag{14}$$

The relationship between $|\psi_m\rangle, |a\rangle$ and $|a_\perp\rangle$ is

$$|\psi_m\rangle = \cos\theta_m |a_\perp\rangle + \sin\theta_m |a\rangle. \tag{15}$$

Note that $|\psi_m\rangle, |a_\perp\rangle$ and $|a\rangle$ are all normalized.

According to Eq. (15), the amplitude for $|\psi_m\rangle$ to be measured in state $|a\rangle$, i.e. $\langle a|\psi_m\rangle$, is $\sin\theta_m = \sin[(2m+1)\theta_0]$, the projection on to the vertical axis in Fig. 6. This increases as m increases up to the point where $\theta_m = \pi/2$ but then decreases. One therefore takes the number of Grover iterations, m, to be such that $\theta_m \simeq \pi/2$. From Eqs. (14) and (10) we see that we need

$$(2m+1)\sin^{-1}\frac{1}{\sqrt{N}} = \frac{\pi}{2}, \tag{16}$$

which, for large N, gives

$$m = \frac{\pi}{4}\sqrt{N}. \tag{17}$$

When $\theta_m \simeq \pi/2$ measuring the state gives a with high probability.

We do not have to get the number of iterations precisely right. After m iterations, the probability that a measurement gives a is $\sin^2\theta_m = \sin^2[(2m+1)\theta_0]$. Any value of θ_m in the range

$$\frac{\pi}{4} < \theta_m < \frac{3\pi}{4} \tag{18}$$

will get determine a correctly with a probability greater than $1/2$. For large N this corresponds to

$$\frac{\pi}{8}\sqrt{N} < m < \frac{3\pi}{8}\sqrt{N}. \tag{19}$$

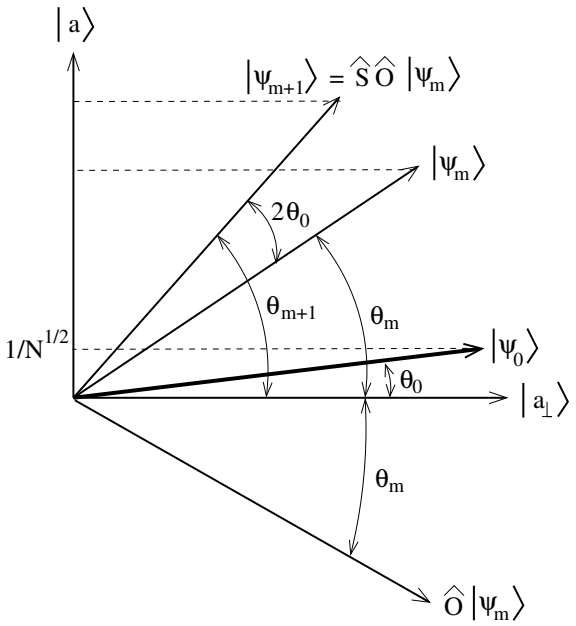

FIG. 6: After the m-th iteration of the Grover algorithm, the state $|\psi_0\rangle$ has been rotated to $|\psi_m\rangle$, which makes an angle θ_m with the $|a_\perp\rangle$ axis. At the next iteration of the Grover algorithm, firstly the action of $\hat{O}$ reflects $|\psi_m\rangle$ about the $|a_\perp\rangle$ axis as shown. This is equivalent to a clockwise rotation by $2\theta_m$ so $\hat{O}|\psi_m\rangle$ is at an angle θ_m below the $|a_\perp\rangle$ axis. Secondly, the state $\hat{O}|\psi_m\rangle$ is acted on by $\hat{S}$ which reflects about the direction of $|\psi_0\rangle$. This is equivalent to an anti-clockwise rotation by $2(\theta_m + \theta_0)$. The net effect of the two operations is to rotate $|\psi_m\rangle$ by an angle $2\theta_0$ in an anti-clockwise direction. Hence the new state $|\psi_{m+1}\rangle$ is at an angle $\theta_{m+1} = \theta_m + 2\theta_0$ to the $|a_\perp\rangle$ axis. The amplitude for the state $|\psi_m\rangle$ to be $|a\rangle$ is the projection on to the vertical axis, which increases with m, at least up to the point where $\theta_m = \pi/2$.

Note that the probability *decreases* for $m > (\pi/4)\sqrt{N}$, unlike many algorithms where increasing the number of iterations progressively improves the probability of success.

The operation count of the Grover algorithm is $O(\sqrt{N})$ which is a quadratic speedup compared with the $O(N)$ count on a classical computer. The quantum speedup comes, of course, from quantum parallelism; all $N = 2^n$ values of $f(x)$ are evaluated in parallel, so naively it looks as though we should have a speedup by a factor of N, i.e. an operation count of $O(1)$. However, if one measured directly after computing the function, one would just get one value of x and the corresponding $f(x)$, which is no better than on a classical computer. It requires additional operations, in the form of the Grover operator $\hat{G}$ applied iteratively, to extract a speedup, which in this case only reduces the operation count to $O(\sqrt{N})$ not $O(1)$. One can show that the $O(\sqrt{N})$ operation count of the Grover algorithm is optimal. An operation count of $O(1)$ is *proved* to be impossible.

III. EXTENSIONS

In the standard implementation of the Grover algorithm it is assumed that there is only one special value. If there are M solutions then one can show[1–3] that one finds one of the special values, selected at random, after $(\pi/4)\sqrt{N/M}$ iterations of the Grover operator.

This is only useful if one *knows* in advance how many special values, M, there are. It turns out that one can determine M, and also get one of the special values, by combining the Quantum Fourier Transform that we saw in Shor's algorithm with Grover's algorithm. In fact this "quantum counting" algorithm will even tell you whether or not a special value exists at all, i.e. whether or not $M = 0$. However, quantum counting is more advanced material and will not be covered in this course.

Nonetheless, in the last part of the course, in which we discuss a different type of quantum device called a *Quantum annealer*, we will describe an important class of problems called "*optimization*" problems. The main question we will consider is whether a quantum annealer can solve these problems faster than a classical computer. However, if time, we also will briefly discuss, *in general terms*, whether quantum counting might play a role in speeding up the solution to optimization problems.

---

[1] N. D. Mermin, *Quantum Computer Science* (Cambridge University Press, Cambridge, 2007).

[2] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).

[3] R. Vathsan, *Introduction to Quantum Physics and Information Processing* (CRC Press, Boca Raton, 2016).