Chapter 7

Grover's Search Algorithm

Searching an item in an unsorted table or array of size N costs a classical computer $O(N)$ running time. If N is large, this is like searching for a needle in a haystack. Can a quantum computer search for a needle in a haystack more efficiently than its classical counterpart? Grover, in 1995, affirmatively answered this question by proposing a search algorithm that consults the table only $O(\sqrt{N})$ times. In contrast to algorithms like quantum factoring which provide exponential speedups, the search algorithm only provides a quadratic improvement. However, the algorithm is quite important because it has broad applications, and because the same technique can be used to speedup algorithms for NP-complete problems.

One might wonder whether there are even faster quantum algorithms for search. However, it turns out that the quadratic speedup is optimal. This was proved in 1994, even before Grover's algorithm. Any quantum algorithm for search must consult the table at least some constant times $\sqrt{N}$ times. There are two ways to think about Grover's algorithm, and we will describe both ways below.

7.1 Quantum Search

Idea of the Algorithm

The Grover search algorithm strives to solve this exact problem: We are given a boolean function $f : \{1, \ldots, N\} \to \{0, 1\}$, and are promised that for exactly one $a \in \{1, \ldots, N\}$, $f(a) = 1$. Of course, a is the item we are searching for.

The basic idea of the Grover search algorithm is best described geometrically. Because our black box function has only two outcomes, we can identify two important states: $|a\rangle$, the state we are looking for; and everything else, call it $|e\rangle = \sum_{x \neq a} \frac{1}{\sqrt{N-1}} |x\rangle$. These two vectors span a two dimensional subspace, which contains the uniform superposition $|\psi_0\rangle = \sum_x \frac{1}{\sqrt{N}} |x\rangle$. Furthermore, $|a\rangle$ and $|e\rangle$ are orthogonal. We can represent this two dimensional subspace geometrically.

Because $|\psi_0\rangle$ is $N - 1$ parts $|e\rangle$ and only one part $|a\rangle$, it lies very close to $|e\rangle$. Grover's algorithm works by starting with the state $|\psi_0\rangle$ and successively increasing the angle between it and $|e\rangle$, to eventually get closer and closer to $|a\rangle$. It does this by a sequence of reflections: first by reflecting about $|e\rangle$, and then by reflecting about $|\psi_0\rangle$. The net effect of these two reflections, as we will

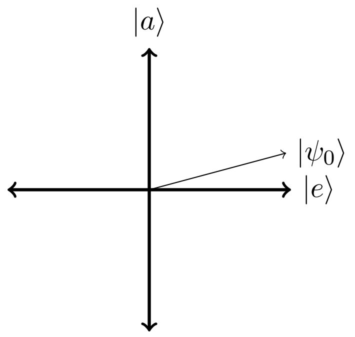

Figure 7.1: Two dimensional space spanned by $|a\rangle$ and $|e\rangle$, and the uniform superposition $|\psi_0\rangle$.

see, is to increase the angle between the state and $|e\rangle$. Repeating this pair of reflections moves the state farther and farther from $|e\rangle$, and therefore closer and closer to $|a\rangle$. Once it is close enough, measuring the state results in outcome a with good probability.

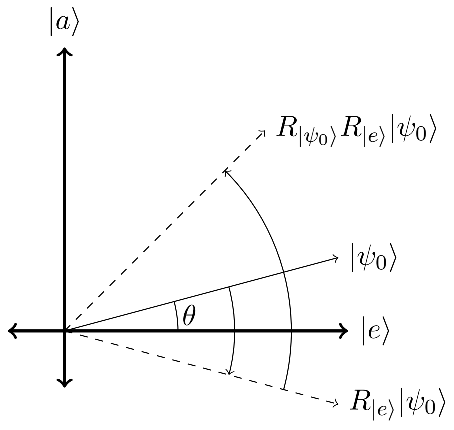
$|a\rangle$
$R_{|\psi_0\rangle}R_{|e\rangle}|\psi_0\rangle$
$|\psi_0\rangle$
θ
$|e\rangle$
$R_{|e\rangle}|\psi_0\rangle$

Figure 7.2: First reflect about $|e\rangle$, then reflect about $|\psi_0\rangle$.

Now the question is just how exactly to carry out these two reflections.

The quantum oracle

From our boolean function $f : \{1, \ldots, N\} \to \{0, 1\}$, we can construct a quantum circuit U_f to carry out this computation. Since we know f can be computed classically in polynomial time, we can also compute it in superposition:

$$\sum_x \alpha_x |x\rangle|0\rangle \to \sum_x \alpha_x |x\rangle|f(x)\rangle$$

There is also a tricky way to put our result into a form that equally contains all of the information relevant to our problem. We can put the answer register in the superposition $|-\rangle$, so that when we implement f the information is stored in the phase or sign of the result:

$$\sum_x \alpha_x |x\rangle|-\rangle \to \sum_x (-1)^{f(x)}\alpha_x|x\rangle|-\rangle$$

In more detail:

$$\begin{aligned} U_f :\ \sum_x \alpha_x|x\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) &\mapsto \sum_x \alpha_x\left(\frac{|x\rangle|f(x)\rangle-|x\rangle|\overline{f(x)}\rangle}{\sqrt{2}}\right)\\ &= \sum_x \alpha_x|x\rangle\left(\frac{|f(x)\rangle-|\overline{f(x)}\rangle}{\sqrt{2}}\right)\\ &= \sum_x \alpha_x|x\rangle(-1)^{f(x)}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \end{aligned}$$

U_f has the property that when $x = a$, the phase of the state will be multiplied -1. We will see that this implementation of the circuit is equivalent to a reflection over the vector $|e\rangle$.

Grover's algorithm

Grover's algorithm finds a in $O(\sqrt{N})$ steps. As before, consider the two dimensional subspace spanned by the two states: $|a\rangle$ and $|e\rangle$, where $|e\rangle$ is as above. Let θ be the angle between $|e\rangle$ and $|\psi_0\rangle = \sum_x \frac{1}{\sqrt{N}}|x\rangle$. See Figure 7.1 for an illustration of these vectors.

Since $|a\rangle$ is the target, we want to increase θ: that is, rotate our input. One way to rotate a vector is to make two reflections. In particular, we can rotate a vector $|v\rangle$ by 2θ by reflecting about $|e\rangle$ and then reflecting about $|\psi_0\rangle$. This transformation is also illustrated in Figure 7.2.

You can see that each time we implement these two reflections, we rotate by 2θ. This means that we need $\frac{\pi}{2}/2\theta = \pi/\theta$ iterations for the algorithm to complete. But what is θ?

$$\langle\psi_0|a\rangle = \cos(\pi/2-\theta) = \sin(\theta), \qquad \langle\psi_0|a\rangle = \sum_x \frac{1}{\sqrt{N}}\langle x|a\rangle = \frac{1}{\sqrt{N}}$$

so that

$$\sin(\theta) = \frac{1}{\sqrt{N}}$$

Since $1/\sqrt{N}$ is very small, $\sin\theta \approx \theta$, and $\theta \approx \frac{1}{\sqrt{N}}$. Thus, we need $O(\sqrt{N})$ iterations for the algorithm to complete. In the end, we get very close to $|a\rangle$, and then with high probability, a measurement of the state yields a.

This gets us everything except for the exact mechanism for each reflection. Because the reflection over $|\psi_0\rangle$ is not dependent on knowledge of $|a\rangle$, we should be able to construct it purely from our regular unitary gates. The reflection over $|e\rangle$, however, requires knowledge of $|a\rangle$, so we will need to use our oracle U_f to construct this reflection.

1. As it turns out, reflection about $|e\rangle$ is easy. All we need to do is flip the phase of the component in the direction of $|a\rangle$. We already saw how to achieve this using the second implementation of f that we showed earlier.

2. For the reflection about $|\psi_0\rangle$, we use the Diffusion operator D (assume $N = 2^n$), which works as follows. First, apply H_{2^n}, which maps $|\psi_0\rangle \mapsto |00\ldots0\rangle$. Then reflect around $|00\ldots0\rangle$ (this is accomplished by the circuit U_g, where g is a function such that $g(00\ldots0) = 0$ and $g(x) = 1$ for $x \neq 00\ldots0$. Finally, apply H_{2^n} to return to the original basis. (Note that this is simply a reflection around the zero vector in the Hadamard basis. You might understand this operation better after the second description of the algorithm below.)

Another approach

Let's look at the search algorithm differently, with all superpositions.

Claim .1 *The Diffusion operator D has two properties:*

1. *It is unitary and can be efficiently realized.*

2. *It can be seen as an "inversion about the mean."*

Proof:

1. For $N = 2^n$, D can be decomposed and rewritten as:

$$\begin{aligned}
D &= H_N \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \end{pmatrix} H_N \\
&= H_N \left(\begin{pmatrix} 2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} - I \right) H_N \\
&= H_N \begin{pmatrix} 2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} H_N - I \\
&= \begin{pmatrix} 2/N & 2/N & \cdots & 2/N \\ 2/N & 2/N & \cdots & 2/N \\ \vdots & \vdots & \ddots & \vdots \\ 2/N & 2/N & \cdots & 2/N \end{pmatrix} 1I \\
&= \begin{pmatrix} 2/N-1 & 2/N & \cdots & 2/N \\ 2/N & 2/N-1 & \cdots & 2/N \\ \vdots & \vdots & \ddots & \vdots \\ 2/N & 2/N & \cdots & 2/N-1 \end{pmatrix}
\end{aligned}$$

Observe that D is expressed as the product of three unitary matrices (two Hadamard matrices separated by a conditional phase shift matrix). Therefore, D is also unitary. Regarding the implementation, both the Hadamard and the conditional phase shift transforms can be efficiently realized within $O(n)$ gates.

2. Consider D operating on a vector $|\alpha\rangle$ to generate another vector $|\beta\rangle$:

$$D \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_i \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_i \\ \vdots \\ \beta_N \end{pmatrix}$$

If we let $\mu = 1/N \sum_j \alpha_j$ be the mean amplitude, then the expression $2\mu - \alpha_i$ describes a reflection of α_i about the mean. This might be easier to see by writing it as $\mu + (\mu - \alpha_i)$. Thus, the amplitude of $\beta_i = -\frac{2}{N} \sum_j \alpha_j + \alpha_i = -2\mu + \alpha_i$ can be considered an "inversion about the mean" with respect to α_i.

■

The quantum search algorithm iteratively improves the probability of measuring a solution. Here's how:

1. Start state is $|\psi_0\rangle = \sum_x \frac{1}{\sqrt{N}}|x\rangle$
2. Invert the phase of $|a\rangle$ using f
3. Then invert about the mean using D
4. Repeat steps 2 and 3 $O(\sqrt{N})$ times, so in each iteration α_a increases by $\frac{2}{\sqrt{N}}$

This process is illustrated in Figure 7.3.

Notice that at any point in the algorithm, the state can be described by two numbers, the amplitude α_a of a, and the amplitude α' of any $x \neq a$. Initially $\alpha' = \alpha_x = 1/\sqrt{N}$. As we run the algorithm α_a increases and α' decreases. Suppose we just want to find a with probability $\frac{1}{2}$. Then we only need to run the algorithm until $\alpha_a \approx 1/\sqrt{2}$. At this point, $\alpha' \approx \frac{1}{\sqrt{2N}}$. This means that $\alpha' \geq \frac{1}{\sqrt{2N}}$ during the entire execution of the algorithm. Since in each iteration of the algorithm, α_a increases by at least $2\alpha'$ it follows that the increase is at least $\frac{2}{\sqrt{2N}} = \sqrt{\frac{2}{N}}$. Since our target is $\alpha_a = \frac{1}{\sqrt{2}}$, the number of iterations $\leq \frac{1}{\sqrt{2}}/\sqrt{\frac{2}{N}} = \sqrt{N}$.

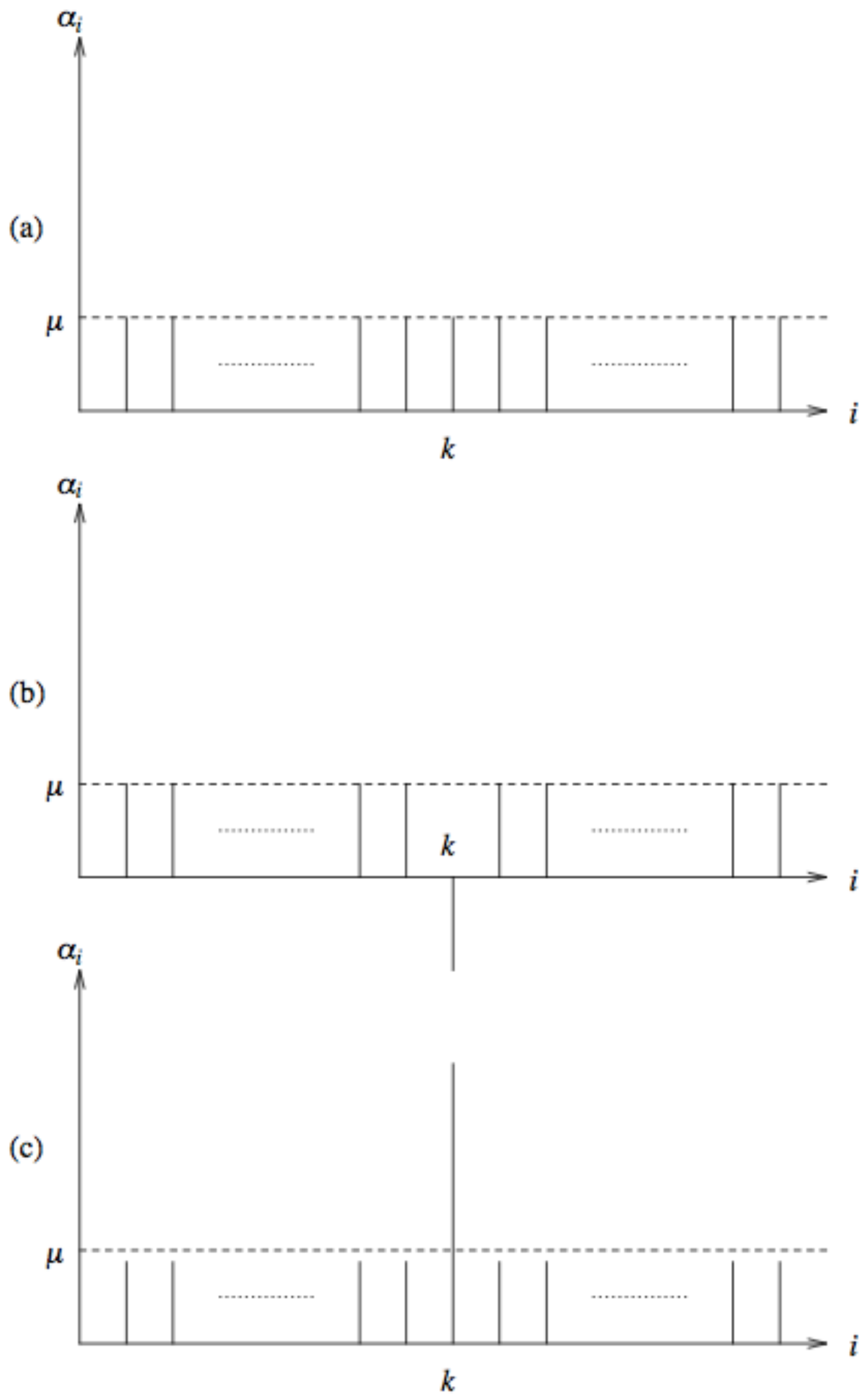

Figure 7.3: The first three steps of Grover's algorithm. We start with a uniform superposition of all basis vectors in (a). In (b), we have used the function f to invert the phase of α_k. After running the diffusion operator D, we amplify α_k while decreasing all other amplitudes.