Grover's Algorithm (Draft notes)

1 Basic Grover algorithm

Grover's algorithm solves a general search problem. Given an unstructured database of N items, Grover's algorithm can find a desired item using $O(\sqrt{N})$ quantum queries to the database. Grover's algorithm has been enormously influential in the development of quantum algorithms, and as search is such a useful primitive, it often appears as a subroutine in other algorithms.

We motivate the workings of Grover's algorithm by first looking at a simple example. Suppose we have a function $f : \{0,1\}^2 \to \{0,1\}$ with the property that $f(x) = 1$ for exactly one $x \in \{0,1\}^2$. We want to find this x. We will see that we can do this by making only one query to the function f. A classical deterministic algorithm would need 4 queries to do this, in the worst case.

For Grover's algorithm, no workspace will be needed. We will only use the query register. We begin in the all-zero state $|00\rangle$. By the symmetry of the situation, a natural thing to do for the first query is to prepare the uniform superposition over all query indices. We can do this by applying the Hadamard transformation to the starting state:

$$\frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{bmatrix}.$$

Now we make a query (using the simplified phase oracle). We consider the four possible outcomes of the state after a query, depending on which of the four inputs to f takes on value 1 (indicated in the subscript of Ψ).

$$|\Psi^1_{00}\rangle = \begin{bmatrix} -1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{bmatrix}, |\Psi^1_{01}\rangle = \begin{bmatrix} 1/2 \\ -1/2 \\ 1/2 \\ 1/2 \end{bmatrix}, |\Psi^1_{10}\rangle = \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}, |\Psi^1_{11}\rangle = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ -1/2 \end{bmatrix}.$$

A remarkable thing has happened. These four states are already orthogonal! This means that we can perfectly distinguish them. If we next apply the unitary operation

$$\begin{bmatrix} -1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & -1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 & -1/2 \end{bmatrix},$$

this will leave us in the state which has value 1 in the coordinate corresponding to the marked input and zeros everywhere else. Measuring in the computational basis will then reveal the x such that $f(x) = 1$ with certainty.

Now let's move on to the case where $f : \{0,1\}^n \to \{0,1\}$ is a function with the property that there is *unique* x such that $f(x) = 1$. Our goal is to find this special *marked element* x. Apart from the promise of uniqueness, which we will discuss later, this is a generic search problem over a search space of size $N = 2^n$. We will now see how Grover's algorithm can solve this task with $O(\sqrt{N})$ many quantum queries.

Again we will only work with the query register. The algorithm begins in the state $|0^n\rangle$. Our goal is to end up in a state with large overlap with $|x\rangle$ for the unique x such that $f(x) = 1$. To indicate this special element we will use O_x for the query operator when the input is the function f where x is the unique string with $f(x) = 1$. That is,

$$\mathsf{O}_x|y\rangle = (-1)^{f(y)}|y\rangle = \begin{cases} -|y\rangle & \text{if } y = x \\ |y\rangle & \text{otherwise} \end{cases}$$

There are three important vectors in Grover's algorithm. The first is $|x\rangle$. Our goal is to end up in a state with large overlap with $|x\rangle$. Also the query operator $\mathsf{O}_x = I - 2|x\rangle\langle x|$ is exactly the reflection about the hyperplane defined by $|x\rangle$.

The second important vector for the algorithm is the uniform superposition $|u\rangle = \frac{1}{\sqrt{N}} \sum_{y\in\{0,1\}^n} |y\rangle$. As in the two-bit example above, the first operation of the algorithm will be to apply the Hadamard transformation $H^{\otimes n}$ to the starting state $|0^n\rangle$ to arrive at $|u\rangle$, as

$$H^{\otimes n}|0^n\rangle = \frac{1}{\sqrt{N}} \sum_{y\in\{0,1\}^n} |y\rangle \quad .$$

Think of the two-dimensional plane spanned by $|x\rangle$ and $|u\rangle$. Our algorithm begins (after performing the Hadamard transformation) in the state $|u\rangle$ and we want to end up close to $|x\rangle$. Grover's algorithm can be completely described in this two dimensional plane. It is based on the *Grover iterate* G, a unitary operation which can be performed with one application of the oracle O_x and which is a rotation in this plane by an angle of approximately $1/\sqrt{N}$. Applying this iterate $O(\sqrt{N})$ times rotates $|u\rangle$ to a vector near $|x\rangle$.

The Grover iterate is defined as $G = (2|u\rangle\langle u| - I)(I - 2|x\rangle\langle x|) = (2|u\rangle\langle u| - I)\mathsf{O}_x$. To analyze the action of the Grover iterate, we introduce the third important vector $|u'\rangle$, which is the uniform superposition over all strings that are not x

$$|u'\rangle = \frac{1}{\sqrt{N-1}} \sum_{y\neq x} |y\rangle \quad .$$

We can think of $|u'\rangle$ as the uniform superposition over all the *bad* strings, that is strings that are not the solution to our problem. Note that the starting state $|u\rangle = \frac{1}{\sqrt{N}}(\sqrt{N-1}|u'\rangle + |x\rangle)$ lies in the span of $|u'\rangle, |x\rangle$, and moreover $|u'\rangle$ is orthogonal to $|x\rangle$. As the subspace spanned by $|u'\rangle, |x\rangle$ is invariant under the action of G, we can describe the entire algorithm by examining the action of G in this subspace.

With respect to the basis $|u'\rangle, |x\rangle$ the action of O_x in the two-dimensional subspace is given by the matrix

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} .$$

Say that the angle between $|u\rangle$ and $|u'\rangle$ is θ (we will shortly calculate this angle). We can find the matrix representing reflection about $|u\rangle$ as follows. First we rotate $|u\rangle$ clockwise to $|u'\rangle$, then we reflect about $|u'\rangle$, then we rotate by θ counterclockwise. This shows that reflection about $|u\rangle$ is given by the matrix

$$\begin{bmatrix} \cos(2\theta) & \sin(2\theta) \\ \sin(2\theta) & -\cos(2\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$$

Thus overall the action of G in the two-dimensional subspace spanned by $|u'\rangle$ and $|x\rangle$ is

$$\begin{bmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{bmatrix} .$$

In other words it is a rotation counterclockwise by the angle 2θ. Recall that θ is the angle between $|u\rangle$ and $|u'\rangle$. As $\langle u|u'\rangle = \sqrt{1-1/N}$ we have $\theta = \arccos(\sqrt{1-1/N}) = \arcsin(\sqrt{1/N})$.

Grover's algorithm proceeds as follows. We first apply the Hadamard transformation on $|0^n\rangle$ to reach the state

$$|u\rangle = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} .$$

Now we apply the Grover iterate k times for a k to be chosen shortly. Note that as G is a rotation by 2θ we have

$$G^k = \begin{bmatrix} \cos(2k\theta) & -\sin(2k\theta) \\ \sin(2k\theta) & \cos(2k\theta) \end{bmatrix} .$$

Each application of the Grover iterate can be performed with one query. Thus after k queries we are in the state

$$|\Psi_x^k\rangle = G^k|u\rangle = \begin{bmatrix} \cos((2k+1)\theta) \\ \sin((2k+1)\theta) \end{bmatrix} .$$

To end the algorithm we are going to measure in the computational basis, i.e. apply the measurement $\{|y\rangle\langle y|\}_{y\in\{0,1\}^n}$. If we apply this measurement to $|\Psi_x^k\rangle$, the probability that we get x is $(\sin((2k+1)\theta))^2$. To find x with certainty, we would like to take the number of iterations to be $\bar{m} = \frac{\pi}{4\theta} - 1/2$, but the trouble is that this may not be an integer. By taking the number of iterations to be $m = \lfloor \frac{\pi}{4\theta} \rfloor$ we have that $|m - \bar{m}| \leq 1/2$, and so

$$\sin((2m+1)\theta) \geq \sin(\pi/2 - \theta) = \cos(\theta) = \sqrt{1-1/N} .$$

Therefore $(\sin((2m+1)\theta))^2 \geq 1 - 1/N$. Also, using $\sin(\theta) \leq \theta$ we have that $\theta \geq 1/\sqrt{N}$ and so $m \leq \frac{\pi}{4}\sqrt{N}$. Thus by making at most $\frac{\pi}{4}\sqrt{N}$ queries we can find the marked element x with probability at least $1 - 1/N$.

1.1 Amplitude amplification

Again consider the case where the input is a function $f : \{0,1\}^n \to \{0,1\}$ and our goal is to find an x such that $f(x) = 1$. In the previous discussion, we assumed that there was a *unique* x with $f(x) = 1$. In this section we loosen this restriction to let $T = |f^{-1}(1)|$ be arbitrary but *known* in advance. In the next section, we will see how to deal with the case where $|f^{-1}(1)|$ is unknown. We again use $N = 2^n$ for the size of the domain of f.

Let

$$|\Psi_{good}\rangle = \frac{1}{\sqrt{T}} \sum_{y \in f^{-1}(1)} |y\rangle, \qquad |\Psi_{bad}\rangle = \frac{1}{\sqrt{N-T}} \sum_{y \in f^{-1}(0)} |y\rangle \ .$$

Suppose that we can create the state $\sqrt{p}|\Psi_{good}\rangle + \sqrt{1-p}|\Psi_{bad}\rangle$. Then by measuring in the computational basis we find an x with $f(x) = 1$ with probability p. Thus our goal is to create a state with large overlap with $|\Psi_{good}\rangle$ in a query-efficient manner.

The query oracle for input f behaves in the following way:

$$\mathsf{O}_f|y\rangle = \begin{cases} -|y\rangle & \text{if } f(y) = 1 \\ |y\rangle & \text{if } f(y) = 0 \end{cases} .$$

In other words, $\mathsf{O}_f = I - 2|\Psi_{good}\rangle\langle\Psi_{good}|$ is a reflection about the hyperplane defined by Ψ_{good}.

A natural generalization of the algorithm from the last section is able to find an x with $f(x) = 1$ with high probability after making $O(\sqrt{N/T})$ many calls to O_f.

We will explain this algorithm here in a more general setting known as *amplitude amplification*. In this general setting, we have a partition of $\{0,1\}^n$ into two sets G and B, for "good" and "bad". Our goal is to find an $x \in G$. In the setup for amplitude amplification we assume that we are given two things: one is a unitary F able to "recognize" a good element, specifically with the following behavior:

$$F|y\rangle = \begin{cases} -|y\rangle & \text{if } y \in G \\ |y\rangle & \text{if } y \in B \end{cases} . \tag{1}$$

We also suppose we are given a unitary $\mathcal{A}$ which is cheap in terms of queries, or simple to obtain, and which produces a state that has some overlap with the good elements. Specifically, let

$$\mathcal{A}|0^n\rangle = \sum_{y \in \{0,1\}^n} \alpha_y |y\rangle \ .$$

We let

$$|\Psi_{good}\rangle = \sum_{y \in G} \alpha_y |y\rangle, \qquad |\Psi_{bad}\rangle = \sum_{y \in B} \alpha_y |y\rangle \ ,$$

and say that $p = \||\Psi_{good}\rangle\|^2$. Thus if we were to apply $\mathcal{A}$ to $|0^n\rangle$ and measure in the computational basis, our success probability would be p. The goal is to efficiently use $\mathcal{A}$ as a subroutine to obtain an algorithm with large success probability. If we were simply to repeatedly apply $\mathcal{A}$ and measure, we would expect to have to do this $1/p$ times until we found a good element. Amplitude amplification is a method that finds a good element with high probability after $O(1/\sqrt{p})$ applications of $\mathcal{A}, \mathcal{A}^*$ and F.

As an example of how one might choose such an $\mathcal{A}$, let's return to our original problem of finding an x such that $f(x) = 1$ when f has T many ones. Let $\mathcal{A} = H^{\otimes n}$. Then

$$H^{\otimes n}|0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} |y\rangle = \sqrt{\frac{T}{N}}|\Psi_{good}\rangle + \sqrt{\frac{N-T}{N}}|\Psi_{bad}\rangle \ .$$

By applying amplitude amplification we can therefore find a good element with $O(\sqrt{N/T})$ many applications of $H^{\otimes n}$ and O_f.

The setup in amplitude amplification is very similar to the basic version of Grover's algorithm we have already covered. The entire algorithm can be described in the two-dimensional subspace with orthogonal basis $|\Psi_{bad}\rangle, |\Psi_{good}\rangle$. The Grover iterate G is the composition of a reflection in the hyperplane defined by $|\Psi_{good}\rangle$ and a reflection about the "starting state" $|u\rangle = \mathcal{A}|0\rangle$

$$G = (2|u\rangle\langle u| - I)(I - 2|\Psi_{good}\rangle\langle\Psi_{good}|) \ .$$

Note that F from Equation 1 is exactly the operation $(I - 2|\Psi_{good}\rangle\langle\Psi_{good}|)$. To see how to perform a reflection about $|u\rangle$, note

$$2|u\rangle\langle u| - I = 2\mathcal{A}|0\rangle\langle 0|\mathcal{A}^* - I = \mathcal{A}(2|0\rangle\langle 0| - I)\mathcal{A}^* \ .$$

Thus we can perform this operation using $\mathcal{A}$ and $\mathcal{A}^* = \mathcal{A}^{-1}$. Thus overall G can be performed with one application of F, an application of $\mathcal{A}$ and an application of $\mathcal{A}^{-1}$.

The algorithm is now $G^{\otimes k}\mathcal{A}|0^n\rangle$ for a value of k to be specified shortly. We can analyze this very similarly to last time. Let θ be the angle between $|u\rangle$ and Ψ_{bad}. Now $\langle u|\Psi_{bad}\rangle = \sqrt{1-p}$ thus $\theta = \arccos(\sqrt{1-p}) = \arcsin(\sqrt{1/p})$. In the orthogonal basis $|\Psi_{bad}\rangle, |\Psi_{good}\rangle$ we can represent the action of G on the two-dimensional subspace spanned by $|\Psi_{bad}\rangle, |\Psi_{good}\rangle$ as

$$\begin{bmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{bmatrix} \ .$$

Thus after k applications of G to $\mathcal{A}|0^n\rangle$ we are in the state

$$\begin{bmatrix} \cos((2k+1)\theta) \\ \sin((2k+1)\theta) \end{bmatrix} \ ,$$

and our success probability measuring in the computational basis is $(\sin((2k+1)\theta))^2$. By the same computation as last time, choosing $k = \lfloor \frac{\pi}{4\theta} \rfloor \leq \frac{\pi}{4}\sqrt{p}$ gives an error probability of at most $1/p$.

Now we finally tackle the case where the number of solutions is unknown. The following result is from [BBHT96].

Theorem 1. *Let $x \in \{0,1\}^N$ with $|x| \geq k_0$, where k_0 is known. There is a quantum algorithm that can find a j with $x_j = 1$ with probability at least $3/4$ after $O\left(\sqrt{\frac{N}{k_0}}\right)$ queries to x.*

For the proof we will need the following lemma.

Lemma 2. *For any real number α and any positive integer m*

$$\sum_{j=0}^{m-1} \cos((2j+1)\alpha) = \frac{\sin(2m\alpha)}{2\sin(\alpha)} \ .$$

Proof of Theorem 1. Say that $T = |x|$, and let $\theta = \arcsin(\sqrt{T/N})$. We first choose a random set $S \subseteq [N]$ with $|S| = 100$ and check if $x_j = 1$ for any $j \in S$. If $T \geq N/2$ then with high probability this initial procedure will find a j with $x_j = 1$. If it does not, we continue to the second phase of the algorithm where the analysis will assume that $T < N/2$.

If we apply Grover's iterate ℓ times and measure in the computational basis, our success probability is $\sin^2((2\ell+1)\theta)$. If we choose ℓ uniformly at random in $[m]$ (to be chosen later) then using the cosine double angle formula $\sin^2(\alpha) = (1-\cos(2\alpha))/2$ our success probability p is

$$\begin{aligned} p = \frac{1}{m}\sum_{\ell=0}^{m-1}(\sin((2\ell+1)\theta))^2 &= \frac{1}{2m}\sum_{\ell=0}^{m-1} 1 - \cos((2\ell+1)2\theta) \\ &= \frac{1}{2} - \sum_{\ell=0}^{m-1}\cos((2\ell+1)2\theta) \end{aligned}$$

Now applying Lemma 2 gives

$$p = \frac{1}{2} - \frac{\sin(4m\theta)}{4m\sin(2\theta)} \ .$$

This is at least $1/4$ as long as $m \geq 1/\sin(2\theta)$. In particular, taking $m = \lceil\sqrt{N/k_0}\rceil$ suffices to achieve success probability $1/4$ (here we use that $T \leq N/2$).

Repeating this procedure a constant number of times gives the theorem. □

Theorem 3. *Given oracle access to $x \in \{0,1\}^N$ with $T = |x| > 0$, there is a quantum algorithm that finds a j with $x_j = 1$*

- *with an expected number of queries that is $O(\sqrt{N/T}\log(N))$.*
- *with probability at least $3/4$ using $O(\sqrt{N})$ queries in the worst case.*

Proof. We run the following algorithm:

Algorithm 1 Quantum search with unknown number of solutions

```
n ← ⌈log n⌉
k ← 2^{n-2}
while k ≥ 1 do
    Run the algorithm from Theorem 1 assuming |x| ≥ k
    if a marked item is found then
        break
    else
        k ← k/2
    end if
end while
```

Once $k \leq |x|$ then by Theorem 1 the probability a marked item is found is at least $3/4$. Let $\ell = \lfloor \log |x| \rfloor$. The expected number of queries can be upper bounded by

$$\sum_{j=\ell}^{n} \sqrt{\frac{N}{2^j}} + \frac{1}{4^{\ell-j}} \sum_{j=0}^{\ell} \sqrt{\frac{N}{2^j}} = O\left(\sqrt{\frac{N}{2^\ell}} \log n\right) \quad .$$

In the worst case the algorithm makes $\sum j = 0^n \sqrt{\frac{N}{2^j}} = O(\sqrt{N})$ many queries. The algorithm will succeed with probability at least $3/4$ by Theorem 1 as at some point in the loop $k \leq |x|$. $\square$

We note that the first item can be improved to $O(\sqrt{N/T})$, see [BBHT96].

1.2 Application

Let's look at an application of these ideas to the *collision problem*. In the collision problem the input is a string $x \in [m]^n$, with $|m| \geq |n|$, and we are promised that either all the entries of x are distinct (x defines a 1-to-1 function), or that each entry of x appears exactly twice (x is 2-to-1).

The following algorithm for the collision problem is due to Brassard, Hoyer, and Tapp [BHT16]. First we query k entries of x, for a positive integer k to be chosen later. We then check if two of these entries are the same. If so, then we output that x is 2-to-1 and stop. Otherwise, the $H \subseteq [m]$ be the set of entries seen, which must all be distinct and so $|H| = k$. we now define a function $f : \{k+1, \ldots, n\} \to \{0, 1\}$ such that $f(j) = 1$ iff $x_j \in H$. If x is 2-to-1 then $|f^{-1}(1)| = k$, which is known to the algorithm, and therefore Grover's algorithm can find a j such that $f(j) = 1$ with probability at least $3/4$ after $\sqrt{n/k}$ many queries. If x is 1-to-1 then f is the constant 0 function and no such index exists. We thus can run Grover's algorithm on f assuming the number of marked elements is k and report that x is 2-to-1 if we find a marked element and that f is 1-to-1 if we do not. This algorithm makes $O(\sqrt{n/k})$ many queries and succeeds with probability at least $3/4$.

The total number of queries made is of the order $k + \sqrt{n/k}$. This is optimized by taking $k = n^{1/3}$, which gives $O(n^{1/3})$ queries overall.

References

[BBHT96] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. Technical Report 9605034, arXiv, 1996.

[BHT16] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum algorithm for the collision problem. In *Encyclopedia of Algorithms*, pages 1662–1664. 2016.