Math and proofs for the quantum algorithms implemented in the code

Joseph Iosue – January 28, 2018

Contents

Note, if you get probabilities that sum to more than one, you probably applied a controlled gate with repeated qubits; i.e. apply a controlled-not gate with control and target both being the same qubit.

1 Grover search algorithm

Let $\{|x\rangle \mid 0 \leq x < N\}$ be a basis in order (for the code, $|0\rangle$ corresponds to $|00..00\rangle$, $|2\rangle$ corresponds to $|00..10\rangle$, $|N-1\rangle$ corresponds to $|11..11\rangle$, ...)

Consider that we are given some unitary operator U_ω such that

$$U_\omega |x\rangle = \begin{cases} -|x\rangle & \text{if } x = \omega \\ |x\rangle & \text{if } x \neq \omega \end{cases} = I - 2|\omega\rangle\langle\omega|$$

Our goal is to find ω (U_ω is a 'black box', we cannot look inside). Let $|s\rangle$ be the equal superposition of all basis states $|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$, and the Grover diffusion operator be defined as $D = 2|s\rangle\langle s| - I$.

1.1 Steps

1. Start with the register in state $|\psi\rangle = |0\rangle = |00...00\rangle$; there must be at least $\log_2(N)$ number of qubits in order to have the necessary N basis states
2. Apply the Hadamard gate to each qubit so that $|\psi\rangle = |s\rangle$
3. Apply U_ω to the system $|\psi\rangle$, then apply D to the system $|\psi\rangle$
4. Perform step 3 a total of r times, where r is the closest integer to $\frac{\pi}{4\arcsin\left(1/\sqrt{N}\right)} - 1/2$
5. Perform a measurement to collapse the system into a basis state; the resulting basis state will, with high probability, be $|\psi\rangle = |\omega\rangle$

1.2 Why it works

Let $|s'\rangle = \frac{1}{\sqrt{N-1}}\sum_{x=0,\neq\omega}^{N-1}|x\rangle$ so that $|s\rangle = \sqrt{\frac{N-1}{N}}|s'\rangle + \frac{1}{\sqrt{N}}|\omega\rangle$. $|s'\rangle$ and $|\omega\rangle$ are orthonormal; use them to define an orthonormal basis $\{|s'\rangle, |\omega\rangle\}$. In this basis,

$$\langle s| = \begin{pmatrix} \sqrt{\frac{N-1}{N}} & \frac{1}{\sqrt{N}} \end{pmatrix} \qquad \Rightarrow D = \frac{1}{N}\begin{pmatrix} N-1 & 2\sqrt{N-1} \\ 2\sqrt{N-1} & -(N-2) \end{pmatrix} \qquad U_\omega = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Thus, the total operator that we define in step 3 above is

$$DU_\omega = \frac{1}{N}\begin{pmatrix} N-2 & -2\sqrt{N-1} \\ 2\sqrt{N-1} & N-2 \end{pmatrix}$$

Notice that this can be written as the standard rotation operator

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

with

$$\begin{aligned} \frac{N-2}{N} &= \cos\theta \\ &= \cos^2(\theta/2) - \sin^2(\theta/2) \\ &= 2\cos^2(\theta/2) - 1 \quad \longrightarrow \end{aligned} \qquad \begin{aligned} \frac{2\sqrt{N-1}}{N} &= \sin\theta \\ &= 2\sin(\theta/2)\cos(\theta/2) \\ &= 2\sin(\theta/2)\sqrt{\frac{N-1}{N}} \end{aligned}$$

where we plug in the result of $\cos(\theta/2)$ on the third line on the left into the third line on the right. Thus, at every iteration, we rotate our state $|\psi\rangle$ and angle $\theta = 2\arcsin\left(1/\sqrt{N}\right)$ in the $(|s'\rangle, |\omega\rangle)$ plane.

Note that we start off rotated counterclockwise at an angle ϕ in the $(|s'\rangle, |\omega\rangle)$ plane, where $\cos\phi = \langle s|s'\rangle = \sqrt{\frac{N-1}{N}}$. This is equivalent to $\sin\phi = 1/\sqrt{N}$, or $\phi = \theta/2$. We apply the operation DU_ω r times to our state $|\psi\rangle$ that is initially at $|s\rangle$. The goal is to make it that $\langle\omega|\psi\rangle = \langle\omega| (DU_\omega)^r |s\rangle$ is maximum. Thus, we need to rotate an angle of $\pi/2 - \phi$, because we start off offset from $|s'\rangle$ by ϕ and we want to get to $|\omega\rangle$ which is offset from $|s'\rangle$ by $\pi/2$.

Therefore, we want that $r\theta = \pi/2 - \phi$. Plugging in for $\phi = \theta/2$ and $\sin(\theta/2) = 1/\sqrt{N}$, we find that

$$r = \frac{\pi}{4\arcsin\left(1/\sqrt{N}\right)} - \frac{1}{2}$$

Of course, we need to apply it an integer number of times, so we pick the closest integer to r. Note that for large N, $r \approx \pi\sqrt{N}/4$.

Applying the Hadamard gate to each qubit at the beginning is how we initialize the initial $|\psi\rangle = |s\rangle$ state.

2 Quantum Fourier transform (QFT)

The quantum Fourier transform is very similar to the classical DFT. It is defined to be the unitary operator F that operates on the basis states as follows;

$$F|j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \exp(2\pi ijk/N)\,|k\rangle$$

With n qubits, there are $N = 2^n$ basis states. We can see then that in this basis, the matrix of F is defined component wise by

$$F_{jk} = \frac{\omega^{jk}}{\sqrt{N}} \qquad \text{where } \omega = \exp(2\pi i/N)$$

I will not go into detail on how this transformation can be mapped to a series of one and two qubit logic gates; [1] pages 217-220 has a very good explanation, and I fear I would just plagiarize if I tried to explain it here.

In the end, we apply the Hadamard gate $\mathcal{O}(n)$ times and the Controlled-Phase gate $\mathcal{O}(n^2)$ times, then the swap gate $\mathcal{O}(n/2)$ times (see below). The algorithm can be seen in the code.

2.1 Swap gate

When swapping two qubits, we have the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, so the operations corresponding to the first and last basis element should be the identity, and the operations corresponding to the middle two should be the NOT gate. Thus,

$$U_{\text{swap}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Consider the Controlled-Not gate in this basis. If we want to control with the first qubit and NOT the second, then the operations corresponding to the first two basis elements should be the identity, because the control is zero, and the operations corresponding to the second two should be the NOT gate. Call this U^{12}_{CNot}. If we want to control with the second qubit and NOT the first, then the operations corresponding to the first and third basis elements should be the identity because the control is zero, and the operations corresponding to the second and fourth basis elements should be the NOT gate. Call this U^{21}_{CNot}. Thus,

$$U^{12}_{\text{CNot}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad U^{21}_{\text{CNot}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

By matrix multiplication, we see that

$$U_{\text{swap}} = U^{12}_{\text{CNot}} U^{21}_{\text{CNot}} U^{12}_{\text{CNot}}$$

In other words, applying the swap gate to qubits i and j is equivalent to applying the Controlled-Not gate three times; the first and third time with the control being qubit i and target qubit j, and the second time with the control being qubit j and the target qubit i.

3 Shor factorization algorithm

Using quantum computers to compute the factors of a number.

3.1 Necessary lemmas and theorems

Lemma 1. $\gcd(a, b) = \gcd(a, b + a)$

Proof. Let $g_1 = \gcd(a, b)$ and $g_2 = \gcd(a, b + a)$. Since $g_1|a$ and $g_1|b$, it must be that $g_1|a + b$. Similarly, since $g_2|a$ and $g_2|a + b$, it must be that $g_2|b$. Thus, $g_1 = g_2$. □

Theorem 1. *In order to compute* $\gcd(a, b)$*, we execute the following three assignments in order until* $b = 0$*:*

$$t = b;\ b = \mathit{rem}(a/b);\ a = t;$$

When $b = 0$*,* a *has become the gcd.*

Proof. Show that $g = \gcd(a, b) = \gcd(a, \text{rem}(b/a))$. From this, the algorithm is clear, as is it just a recursive occurrence.

We have that $a \equiv r \mod b$ where r is $\text{rem}(b/a)$. Thus, $a - r = zb$ for some integer z. By Lemma 1,

$$\gcd(b, r) = \gcd(b, r + b) = \gcd(b, r + 2b) = ... = \gcd(b, r + zb) = \gcd(b, a)$$

□

Lemma 2. *Bezout's identity states that for any nonzero integers* a *and* b*, there exists integers* x *and* y *such that* $ax + by = \gcd(a, b)$*.*

Proof. This is easy to see; since $a/\gcd(a, b)$ and $b/\gcd(a, b)$ are both integers, we have that $xz_1 + yz_2 = 1$ where z_1, z_2 are integers. Thus, pick y to be an integer with $yz_2 \equiv 1 \mod z_1$, then $x = (1 - yz_2)/z_1$ is an integer. Note that we can pick such a y so long as $z_1 \neq z_2$. If $z_1 = z_2$, then $a = b$, and the identity is trivial. □

Lemma 3. *If* $N|b^2 - 1$*, then*

1. $\gcd(b - 1, N) = 1 \longrightarrow b \equiv -1 \mod N$

2. $\gcd(b + 1, N) = 1 \longrightarrow b \equiv 1 \mod N$

Proof.

1. By Lemma 2, there exists x and y such that $(b - 1)x + Ny = 1$. Thus, $(b^2 - 1)x + N(b + 1)y = b + 1$. Since $N|b^2 - 1$, it must be that $N|b + 1$ in order for that relation to hold. Thus, $b \equiv -1 \mod N$.

2. By Lemma 2, there exists x and y such that $(b+1)x + Ny = 1$. Thus, $(b^2-1)x + N(b-1)y = b-1$. Since $N|b^2-1$, it must be that $N|b-1$ in order for that relation to hold. Thus, $b \equiv 1 \mod N$.

$\square$

Theorem 2. *Let N not be even or an integer power of a prime (modular arithmetic is often different when dealing with primes). If a is coprime with N (ie $\gcd(a,N) = 1$) and the period of $f(x) = a^x \mod N$ is $r \in$ evens with $a^{r/2} \not\equiv -1 \mod N$, then $\gcd(a^{r/2}+1, N)$ and $\gcd(a^{r/2}-1, N)$ are both nontrivial factors of N.*

Proof. Let r be the smallest integer such that $a^r \equiv 1 \mod N \quad \Rightarrow N|(a^r - 1)$.

Assume that we have found r and it is even. Define $b \equiv a^{r/2} \mod N$. We know that $a^{r/2} \not\equiv 1 \mod N$ because we found that r was the smallest integer such that $a^r \equiv 1 \mod N$. Thus, $b \not\equiv 1 \mod N$, and by assumption $b \not\equiv -1 \mod N$. (Note that there exists an a such that $a^{r/2} \not\equiv 1, -1 \mod N$ via the Chinese remainder theorem since N is not a prime power).

Thus, $d = \gcd(b-1, N)$ is a proper factor of N, because $1 < d < N$;

- if $d = 1$, then by Lemma 3, since $N|(a^r - 1 = b^2 - 1)$, $b \equiv -1 \mod N$, which contradicts our assumption.
- if $d = N$, then $N|b-1$ meaning that $b \equiv 1 \mod N$ which contradicts what we found earlier.

Similarly, $f = \gcd(b+1, N)$ is a proper factor of N, because $1 < f < N$;

- if $f = 1$, then by Lemma 3, since $N|(a^r - 1 = b^2 - 1)$, $b \equiv 1 \mod N$, which contradicts what we found earlier.
- if $f = N$, then $N|b+1$ meaning that $b \equiv -1 \mod N$ which contradicts our assumption.

$\square$

3.2 Classical period-finding method

Method to find the period of the function $f(x)$. ie the smallest r such that $f(x) = f(x+r)$ (Brent's algorithm).

1. Initialize the step limit ($p = 1$), the hare's position ($h = 0$), the turtle's position ($t = 0$), and the counter ($r = 0$)
2. The hare takes a step ($h = f(h)$), and the counter is incremented ($r = r + 1$)
3. If the hare has come back around and reached the turtle ($t = h$), then the period is $r \rightarrow$ *done*
4. If we've reached the step limit ($r = p$), then we reset ($t = h$, $r = 0$) and increase the step limit ($p = 2p$)
5. Continue at step 2

3.3 Quantum period-finding method

TODO

3.4 Finding factors with Shor's algorithm

Method to find factors of a number N. (Note that N must not be even or an integer power of a prime number.

1. Randomly pick a positive integer $a < N$. If $\gcd(a, N) \neq 1$, then we have found a nontrivial factor. *done*
2. Otherwise, a is coprime with N. Use either the classical or quantum period-finding method discussed above to compute the period r of $f(x) = a^x \mod N$.
3. If r is odd or $a^{r/2} \equiv -1 \mod N$, restart at step 1. Otherwise, we have sufficient conditions to use Theorem 2. Thus, both $\gcd(a^{r/2}+1, N)$ and $\gcd(a^{r/2}-1, N)$ are nontrivial factors of N. *done*

4 Quantum addition

If we initialize our register to be a particular state with probability one, than we can add deterministically by simply applying particular gates. If our register is in a superposition, then the each state will add in a particular way, but the result is random from the measurement.

4.1 Ripple carry adder

This is not exactly a quantum algorithm; I will not explain in detail, as this algorithm is essentially the same as the addition of binary numbers via the classical ripple carry algorithm, the only difference being that AND and XOR gates are renamed to Tofolli and Controlled-Not gates (unitary operators).

Anyone interested should simply add two binary numbers by hand and see how it works and how to carry digits. For the addition of two n bit numbers, the algorithm I implement uses $3n$ qubits, with qubits 0 through $n-1$ storing the first number, n through $2n-1$ storing the second number, $2n$ through $3n-2$ being intermediate bits that store information about carrying over, and the $3n-1$ bit is an overflow (since adding two n bit numbers can result in a $n+1$ bit number).

To see the full algorithm, see the code.

4.2 Adder using the QFT

Maybe implement in the future

`http://web.mit.edu/2.111/www/2010/ps5_2010Sol.pdf`

Acknowledgments

I love Wikipedia. Quantumplayground.net was also used a few times. [1] is a great resource that I used as well.

References

[1] Isaac Chuang and Michael Nielsen, *Quantum Computation and Quantum Information*, 2000. ISBN: 978-1-107-00217-3.