Quantum Computing and Quantum Information

QI 31415 Final Project

Quantum Phase Estimation

Abdulah Amer

May 22nd 2020

1 Introduction

Quantum phase estimation (QPE) is the basis of many interesting algorithms in the realm of quantum information and quantum computing. Unitary operators (U) are special in quantum computing because they make up all the operations that act on qubits and evolve them in time. This allows Quantum Circuits to run forwards and backward, permitting any computation to be undone by it's conjugate transpose $U^\dagger$. This follows from the fact that for a unitary operator the following is true: $UU^\dagger = I$. Therefore, one could always get a state back to where it was if desired as seen below.

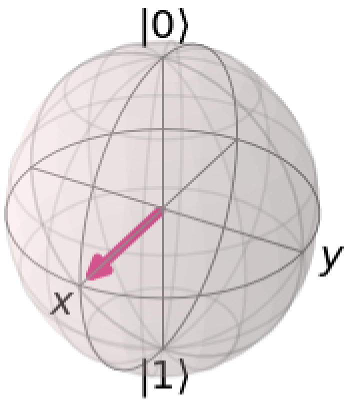
$$|\psi\rangle \xrightarrow{U} |\psi'\rangle \xrightarrow{U^\dagger} |\psi\rangle$$

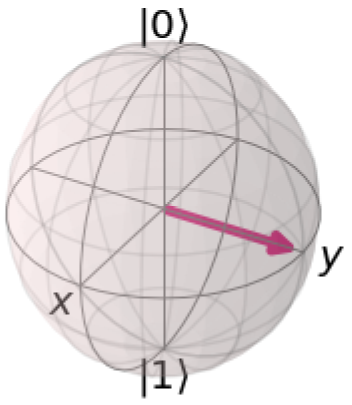
A Unitary operator is normal and has eigenvalues on the unit circle. Therefore its eigenvalues can be represented in the form of $e^{i\phi}$. The action of some unitary operator U on an eigenstate $|\psi\rangle$ can be seen here.

$$U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle$$

Begin with an eigenstate $|\psi\rangle$ of U and corresponding eigenvalue $e^{2\pi i\theta}$. With these in hand, one is able to search for values of θ that represent the angle which a qubit's Bloch vector has precessed on the Bloch sphere.

All operations on a qubit can be decomposed into rotations on the Bloch sphere. Rotations around the Z-axis impart a phase factor on the state, but do not change the probabilities of the state. An example can be seen below where a qubit is in a superposition between the $|0\rangle$ state and the $|1\rangle$ state and then rotated around the Z-axis. After the rotation, the Bloch vector is still in between the two computational states. Through this rotation, the qubit has gained an additional $\frac{\pi}{2}$ phase. QPE uses this relationship to guess θ and that is the goal of the measurement.





2 The QPE Algorithim

The global phase of a quantum state can not be measured. However, relative phases can be seen when using an ancilla qubit, a controlled unitary operation, and a Hadamard gate (H) for each qubit in a quantum register. It is important to note that there are two types of quantum registers in this algorithm. There is the counting register and the ancilla register. The counting register holds multiple qubits that will be used to count a binary string. The more counting qubits one uses, the better the phase estimation will be. The ancilla register, holds the ancilla qubit which will be prepared in an eigenstate of U. The ancilla qubit will be acted on freely with control unitary operators. The counting register qubits are used as control qubits and, the ancilla qubit is used as a target qubit for every controlled unitary operation. When the U operator is applied, with each application the qubit rotates and adds up multiples of $e^{2\pi i\theta}$ phase factors. The multiples take the form of $\{e^{2\pi i\theta(2^n-1)}...e^{2\pi i\theta(2^n-2)}......e^{2\pi i\theta(2^0)}\}$.

Repetition like this encourages a physicist to use a Fourier transfrom, in this case the Quantum Fourier transform (QFT). The pattern of phase factors brings about a situation where the inverse QFT ($QFT^\dagger$) can be used to learn more about it. Applying $QFT^\dagger$ takes one from a space in the counting register to a space that peaks at a state $|2^n\theta\rangle$. This will be explained in the following section concerning the mathematical treatment of the algorithm.

Finally, one makes multiple measurements of the counting qubits. A total of 4096 measurements are made and each singular measurement is called a shot. There is no particular reason to make 4096 measurements, an arbitrary large number shots are needed due to the randomness of measuring the correct answer. Just because the correct answer is more liekly than others it does not mean one will get it the first time. However, over multiple shots one can see the correct answer as the one that comes up most frequently. One measures in the computational basis $\{|0\rangle, |1\rangle\}$. This measurement will be represented by a binary string. The state with the most shots is turned from a binary string into a decimal number and returns the value of $2^n\theta$. To find θ one simply divides by 2^n, noting that n is the number of counting qubits in the counting register being used.

Finding this phase is important to help quantum computers solve an array of difficult problems in the class of hidden subgroup problems. For example, this class of problems includes factoring large numbers and is what makes Shor's algorithm so powerful. Prime factorization is akin to the hidden subgroup problem for finite Abelian groups. Group theory is out of the scope of this paper but is integral in understanding the QPE algorithm and its applications at the highest level possible.

3 Mathematics of QPE

The QPE algorithm begins by setting up a circuit that has an initial state $|\phi_0\rangle$, with n qubits in the counting register starting in the $|0\rangle$ state and 1 qubit in the ancilla register. The ancilla qubit is prepared in an eigenstate $|\psi\rangle$ of an operator U, whose phase is being estimated. Then the Hadamard gate is used on all of the counting qubits to create a superposition between the $|0\rangle$ and $|1\rangle$ states. The initial state is,

$$|\phi_0\rangle = |0\rangle^{\otimes n}|\psi\rangle$$

Now the Hadamard gate is applied on $|\phi_0\rangle$ as shown by

$$H^{\otimes n}|\phi_0\rangle = H^{\otimes n}|0\rangle^{\otimes n}|\psi\rangle$$

To be thorough, note that the use of the Hadamard gate on the $|0\rangle$ state does the following

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

Counting qubits simply do this n times, each qubit in the register gets an H acted on it. This yields the next state in the QPE algorithm $|\phi_1\rangle$

$$|\phi_1\rangle = \frac{1}{2^{\frac{n}{2}}}(|0\rangle + |1\rangle)^{\otimes n}|\psi\rangle$$

Next are the controlled unitary operations or CU operations. They operate on multiple qubits by employing a control qubit and a target qubit. If the control qubit is in the $|1\rangle$ state then the controlled operation is performed on the target qubit, otherwise they do nothing. An example with the first qubit being the control qubit, and the second qubit being the target qubit is shown below.

$$CX(|0\rangle \otimes |0\rangle) = |0\rangle \otimes |0\rangle$$

$$and$$

$$CX(|1\rangle \otimes |0\rangle) = |1\rangle \otimes |1\rangle$$

The action of some CU operation on a single counting qubit and an ancillary qubit, after applying H to the counting qubit is seen below

$$CU\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\theta}|1\rangle) \otimes |\psi\rangle$$

Next is generating the necessary pattern that the $QFT^\dagger$ can be used on. This is achieved by using several repetitive actions of U depending on which counting qubit is being used for control. The nth qubit has U acting on it 2^{n-1} times. The first counting qubit gets hit 2^{n-1} times. The last counting qubit gets hit 2^0 times. Everytime a phase of $e^{2\pi i\theta}$ is picked up. Phase factors on the $|1\rangle$ states will be $\{e^{2\pi i\theta 2^n-1}...e^{2\pi i\theta 2^{n-2}}......e^{2\pi i\theta 2^0}\}$ for the $\{n-1, n-2, ...0\}$ qubits respectively. The QFT on a single state looks like the output seen below. Now the $QFT^\dagger$ can be used to get desired results.

$$|\phi\rangle_2 = \frac{1}{2^{\frac{n}{2}}}((|0\rangle + e^{2\pi i\theta 2^n-1}|1\rangle) \otimes (|0\rangle + e^{2\pi i\theta 2^n-2}|1\rangle) \otimes ... \otimes (|0\rangle + e^{2\pi i\theta 2^0}|1\rangle)) \otimes |\psi\rangle$$

This result can be written more compactly as a sum from 0 to 2^{n-1}.This is very similar to the action of the QFT on some state. For example, say $|x\rangle$ shown here,

$$QFT|x\rangle = \frac{1}{2^{\frac{n}{2}}}((|0\rangle + e^{\frac{2\pi i\theta}{2}x}|1\rangle) \otimes (|0\rangle + e^{\frac{2\pi i\theta}{2^2}x}|1\rangle) \otimes ... \otimes (|0\rangle + e^{\frac{2\pi i\theta}{2^n}x}|1\rangle)$$

Which can be written as a sum,

$$\frac{1}{2^{\frac{n}{2}}}\sum_{k=0}^{n} e^{\frac{2\pi i x}{2^k}}|x\rangle$$

x could be any state, $|2^n\theta\rangle$ is substituted for x and the result for $|\phi\rangle_2$ is derived. Now the $QFT^\dagger$ is applied on $|\phi\rangle_2$ and the result can be written in the form of two sums here.

$$QFT^\dagger|\phi\rangle_2 = \frac{1}{2^n}\sum_{x=0}^{2^{n-1}}\sum_{k=0}^{2^{n-1}} e^{\frac{-2\pi i k}{2^n}(x-2^n\theta)}|x\rangle \otimes |\psi\rangle$$

This sum peaks at $x = 2^n\theta$. One makes a measurement of $|2^n\theta\rangle$ and the result is a number represented as a binary string. That is the value of $2^n\theta$, as said previously, divide by 2^n and θ is found.

An interesting remark of this project concerns probability of measurement. There was a lower probability of getting the correct θ when $2^n\theta$ was not an integer. This is explored later in the analysis section.

4 The QPE Algorithm Circuit

Shown below is the QPE algorithm using three total qubits, two of which are counting qubits and one which is an ancilla qubit. This is a small circuit, and the code was written to allow any circuit of arbitrary size with large counting registers to be made. This allows us to find θ to a desired level of precision. The C denotes a classical register where measurements are stored in classical bits. Only the counting qubits are measured, requiring two classical qubits in this case. The circuit can be shown in the figure below.

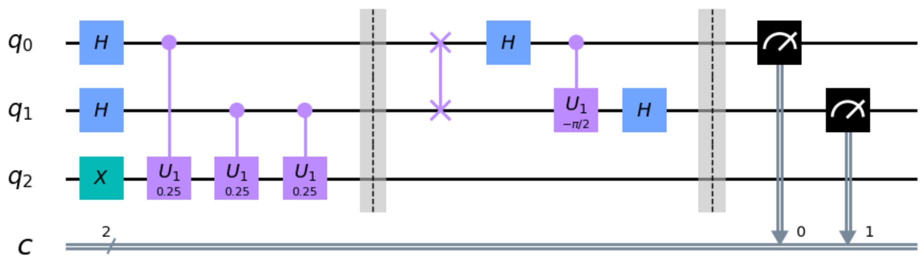


Figure 1: The QPE

5 Analysis of Output

An interesting component of the QPE algorithm is the high probability of finding the $|2^n\theta\rangle$ state when $2^n\theta$ is an integer value. In contrast, when it is a non-integer it has evidently non-zero error. This can be shown by the table below representing non-zero errors for measurements of non-integer values of $2^n\theta$.

For 4 counting qubits; therefore $2^n = 16$

θ	$2^n\theta$	Error
1	16.0	0.0
1/2	8.0	0.0
1/3	5.33333	0.00021
1/4	4.0	0.0
1/5	3.2	0.00025
1/6	2.66667	0.00021
1/7	2.28571	9e-05
1/8	2.0	0.0
1/9	1.77778	7e-05
1/10	1.6	0.00012
1/11	1.45455	0.00017
1/12	1.33333	0.00021
1/13	1.23077	0.00024
1/14	1.14286	0.00027
1/15	1.06667	0.00029
1/16	1.0	0.00031

This is because of the inner workings of the QPE algorithm that require reaching beyond the scope of this project at this time. These inner workings however, have been a driving force of interest for this project. The treatment I am following comes from the work done by Nielsen and Chuang[1].

Please note my treatment will be as thorough and correct as it can be while writing this to keep me honest and to push my boundaries as a physicist.

Refer back to what was discussed previously when introducing the QPE algorithim. The QPE aims to solve the hidden subgroup problem. The QPE is used to find the periodicity of periodic functions, which in this case would be the value of θ. Consider a group that contains elements that are all the numbers that can be made by the counting register. In the counting register are n qubits that can represent a binary string from 0 to 2^n. It is reasonable to assume that $2^n\theta$ is part of that group, given that θ is limited to fractional parts of 2π, only if $2^n\theta$ is an integer. However, if $2^n\theta$ is not an integer it could not possibly be in the group of elements that can be represented by the binary string result. Intuition follows that the larger the group, the easier it is to find some element in the group ϕ such that $\phi\theta \approx 2^n\theta$. When the $QFT^\dagger$ is applied and measurements, are taken a result can be found which is measured most frequently. This section is the most rewarding part of the project and is the big idea of QPE's relationship with errors, whether due to a non-integer value of $2^n\theta$ or to the number of counting qubits used. Hopefully these insights lead to more projects and maybe even an original algorithm of my own one day.

6 The Code

Programming the circuit necessary to carry out the QPE algorithm was not entirely difficult given the beautiful simplicity of the algorithm. Functions were created to take parameters such as desired values of θ and numbers of counting qubits to create arbitrary circuits that allow any intended level of precision. This can be adjusted to take a closer look at what happens to the circuit and its results, and helps build intuition quickly.

The experiment was set up with a tunable CU gate, which could be programmed to be any rotation that is desired for a particular experiment. Creating modular source code was a primary goal to have a basis for more experiments and functionality in the future. One θ is used as an intial parameter to run the algorithim, and then a measurement of θ is made at the end. These values are checked for any discrepancies, this check generates errors which correspond to changes in our initial parameters.

A link to raw and unfiltered source code will be attached here(note this not the final version of the source code and it will be improved in the future). Things that I would improve on include creating smaller subroutines to clean up function calls and hide more functionality from someone else who could be working with the source code. A web application with a GUI could be a future project for educational purposes so that others can learn and understand the

QPE. All of this work features IBM's Qiskit[2] package.

References

[1] Michael A. Nielsen and Isaac L. Chuang. 2011. Quantum Computation and Quantum Information: 10th Anniversary Edition (10th ed.). Cambridge University Press, New York, NY, USA.

[2] https://qiskit.org/