Lecture 8: Phase estimation

February 14, 2006

Last time we discussed reversible computation. We established that any classical Boolean circuit can be converted to a reversible (and therefore unitary) circuit that efficiently implements the function computed by the original circuit. Specifically, if the original Boolean circuit computes the function

$$f : \{0,1\}^n \rightarrow \{0,1\}^m,$$

then the reversible circuit efficiently implements the transformation

$$B_f \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|y \oplus f(x)\right\rangle$$

for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, possibly using some ancilla qubits that we do not mention explicitly. Moreover, if

$$f : \{0,1\}^n \rightarrow \{0,1\}^n$$

is invertible and we have Boolean circuits for efficiently computing both f and f^{-1}, then we can construct an efficient reversible circuit that performs the transformation

$$P_f \left|x\right\rangle = \left|f(x)\right\rangle$$

for all $x \in \{0,1\}^n$.

The next step toward Shor's Algorithm for factoring integers efficiently on a quantum computer will be to subject a particular reversible circuit based on one of the arithmetic problems we saw previously to a process known as *phase estimation*. This was not historically the way that Shor's algorithm was first described, but it is a nice way to understand how it works.

The process of phase estimation can be described in a more general context than what we need. So, although we will eventually apply it to a particular reversible transformation based on an arithmetic function, we will speak in greater generality when discussing phase estimation.

The phase estimation problem

Suppose that we have a description of some quantum circuit Q acting on n qubits. Associated with Q is some $2^n \times 2^n$ unitary matrix U. Obviously, if n is reasonably large such as $n = 1,000$, it would be impossible to write down an explicit description of U because it is too large—all of the computers in the world could only store a tiny fraction of its entries. Even if you just wanted to compute a single entry of U from the description of Q, you might be faced with a computationally difficult task.

Because U is unitary, we know from linear algebra that it has a *complete, orthonormal collection of eigenvectors*

$$\left|\psi_1\right\rangle, \ldots, \left|\psi_N\right\rangle$$

(where $N = 2^n$), and associated eigenvalues having the form

$$e^{2\pi i\theta_1}, \ldots, e^{2\pi i\theta_N}$$

where $\theta_1, \ldots, \theta_N \in [0, 1)$. This means that

$$U |\psi_j\rangle = e^{2\pi i\theta_j} |\psi_j\rangle$$

for each $j \in \{1, \ldots, N\}$, and furthermore that

$$\langle\psi_j|\psi_k\rangle = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases}$$

for all $j, k \in \{1, \ldots, N\}$. The reason why each of the eigenvalues has the form $e^{2\pi i\theta}$, which is equivalent to saying that these eigenvalues are on the complex unit circle, is that U is unitary and therefore preserves Euclidean length.

The problem that we will focus on may be stated as follows:

Phase Estimation Problem

Input: A quantum circuit Q that performs a unitary operation U, along with a quantum state $|\psi\rangle$ that is promised to be an eigenvector of U:

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

Output: An approximation to $\theta \in [0, 1)$.

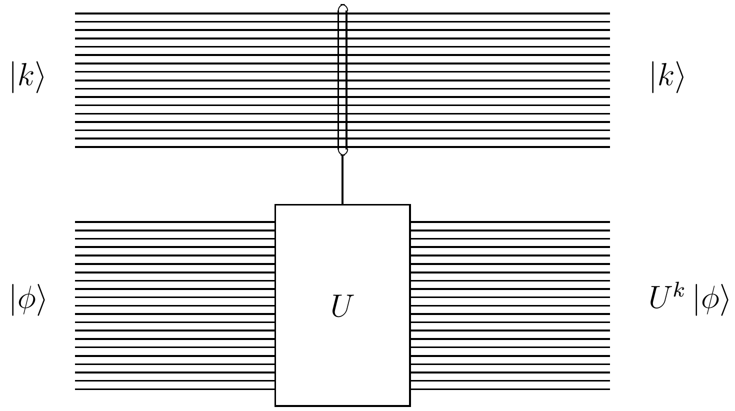
This problem is somewhat informally stated, because no specific requirements have been placed on the precision to which θ must be approximated. It will turn out that for an arbitrary circuit Q and eigenvector $|\psi\rangle$, the number θ can efficiently be approximated by the procedure that we will describe, but only to low precision (to a logarithmic number of bits in the circuit size). However, for certain choices of U it will be possible to achieve much higher precision, and when we apply our method to factoring this is the case in which we will be interested.

The phase estimation procedure

In order to describe the quantum procedure for phase estimation, let us introduce some notation. Suppose that U is a unitary transformation acting on n qubits, and suppose m is any positive integer. Then we let $\Lambda_m(U)$ denote the unique unitary transformation on $m+n$ qubits that satisfies

$$\Lambda_m(U) |k\rangle |\phi\rangle = |k\rangle \left(U^k |\phi\rangle\right)$$

for all choices of $k \in \{0, \ldots, 2^m - 1\}$ and an arbitrary n-qubit vector $|\phi\rangle$. In words, the first m qubits specify the number of times that U is to be applied to the remaining n qubits. We sometimes denote this transformation as suggested in the following quantum circuit diagram:

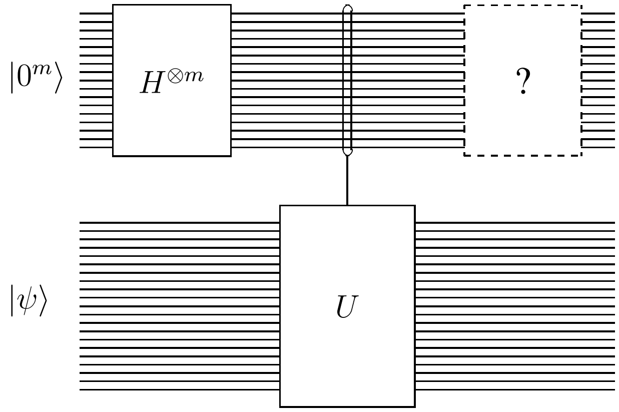


Note that it is possible that n and m differ significantly.

Now, if someone gives you a quantum circuit Q that performs the unitary operation U, there is no guarantee that you will be able to build a quantum circuit that efficiently implements $\Lambda_m(U)$ for your choice of m. For example, if $m \approx n$, you would probably require that U has some special properties to allow an efficient implementation. This is because the number of times k that U may effectively need to be performed can be exponential in m. If $m = O(\log n)$, however, you can always construct an efficient implementation of $\Lambda_m(U)$ (assuming you have a circuit Q that efficiently implements U).

We will not go into the details of how one would construct $\Lambda_m(U)$ for small values of m given a circuit Q implementing U, because our interest will be focused on a particular choice of U where it is easy to implement $\Lambda_m(U)$, even for $m = \Theta(n)$. Specifically, the transformation U will correspond to modular multiplication by a fixed number a, and so $\Lambda_m(U)$ will correspond to modular exponentiation, which we have already shown is efficiently implementable. However, for now let us just forget about these specifics and suppose that we have a quantum circuit implementing $\Lambda_m(U)$ for some particular choice of m (which may be large or small).

Now, consider the following quantum circuit diagram:



Note that the input to the second collection of qubits is the state $|\psi\rangle$, which is promised to be an eigenvector of U. We will try to fill in the missing part momentarily, but for now let us just consider the state after the $\Lambda_m(U)$ operation is performed. The initial state is $|0^m\rangle|\psi\rangle$, and after the Hadamard transforms are performed the state becomes

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}|k\rangle|\psi\rangle .$$

Then the $\Lambda_m(U)$ transformation is performed, which transforms the state to

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}|k\rangle\left(U^k|\psi\rangle\right).$$

Now, let us use the fact that $|\psi\rangle$ is an eigenvector of U to simplify this expression. Specifically we assume that

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

for some real number $\theta \in [0,1)$, which is the value we are trying to approximate. Applying U^k to $|\psi\rangle$ is equivalent to applying U to $|\psi\rangle$ a total of k times, so

$$U^k|\psi\rangle = \left(e^{2\pi i\theta}\right)^k|\psi\rangle = e^{2\pi ik\theta}|\psi\rangle .$$

Thus, we can rewrite the state of the circuit after the $\Lambda_m(U)$ gate has been performed as

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}|k\rangle\left(e^{2\pi ik\theta}|\psi\rangle\right) = \frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}e^{2\pi ik\theta}|k\rangle|\psi\rangle .$$

Notice that the same "phase kickback" effect has happened as for some of the algorithms we saw previously. The first m qubits and the last n qubits are uncorrelated at this point, given that they are in a tensor product state:

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}e^{2\pi ik\theta}|k\rangle|\psi\rangle = \left(\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}e^{2\pi ik\theta}|k\rangle\right)|\psi\rangle .$$

So, if we discard the last n qubits, we are left with the state

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1}e^{2\pi ik\theta}|k\rangle .$$

A simple case: $\theta = j/2^m$

Now, recall that our goal is to approximate θ. Suppose for the moment that θ happens to have a special form:

$$\theta = \frac{j}{2^m}$$

for some integer $j \in \{0, \ldots, 2^m - 1\}$. In general we cannot assume that θ has this form, but it is helpful to consider this case first. Then the above state can be written

$$\frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} e^{2\pi i \frac{jk}{2^m}} |k\rangle = \frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} \omega^{jk} |k\rangle$$

for $\omega = e^{2\pi i/2^m}$.

Several lectures ago we discussed the problem where a set of states is known, an unknown state from that collection is given to you, and your goal is to determine which of the possible states it is. It is possible to solve the problem perfectly if the set of states is orthonormal. This is the situation we have here. Let us define

$$|\phi_j\rangle = \frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} \omega^{jk} |k\rangle$$

for each $j \in \{0, \ldots, 2^m - 1\}$. We know that the state of the first m qubits of our circuit is one of the states $\{|\phi_j\rangle : j = 0, \ldots, 2^m - 1\}$ and the goal is to determine which one. Once we know j, we know θ as well (because we are still considering the special case $\theta = j/2^m$).

We have

$$\langle \phi_j | \phi_{j'} \rangle = \frac{1}{2^m} \sum_{k=0}^{2^m-1} \omega^{k(j'-j)} = \frac{1}{2^m} \sum_{k=0}^{2^m-1} \left(\omega^{j'-j}\right)^k .$$

Using the formula

$$\sum_{k=0}^{n-1} x^k = \frac{x^n - 1}{x - 1}$$

for $x \neq 1$ (and $\sum_{k=0}^{n-1} 1^k = n$ of course), along with the observation that $\omega^{2^m l} = 1$ for any integer l, we obtain

$$\langle \phi_j | \phi_{j'} \rangle = \begin{cases} 1 & \text{if } j = j' \\ 0 & \text{if } j \neq j'. \end{cases}$$

Thus, the set $\{|\phi_0\rangle, \ldots, |\phi_{2^m-1}\rangle\}$ is indeed orthonormal.

There is therefore a unitary transformation F that satisfies

$$F|j\rangle = |\phi_j\rangle$$

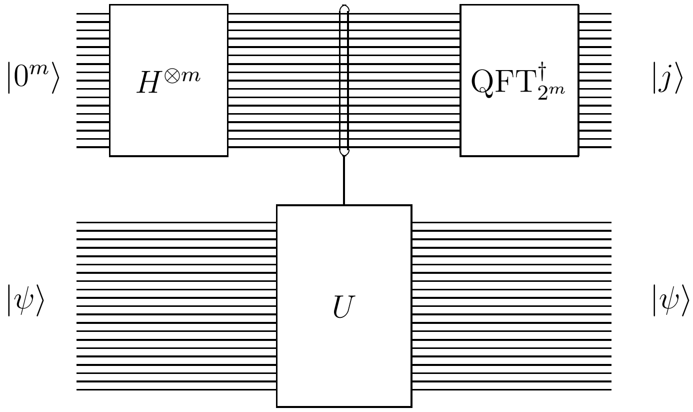
for $j = 0, \ldots, 2^m - 1$. We can describe this matrix explicitly by allowing the vectors $|\phi_j\rangle$ to determine the columns of F:

$$F = \frac{1}{\sqrt{2^m}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{2^m-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(2^m-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{2^m-1} & \omega^{2(2^m-1)} & \cdots & \omega^{(2^m-1)^2} \end{pmatrix}$$

This matrix defines a linear transformation that is of tremendous importance in many areas of science: the *discrete Fourier transform*. When we refer to this transformation in the context of quantum computing we call it the *quantum Fourier transform*, and for that reason it is also sometimes denoted QFT_{2^m}. For convenience, let us write it explicitly:

$$\mathrm{QFT}_{2^m} |j\rangle = \frac{1}{\sqrt{2^m}} \sum_{k=0}^{2^m-1} e^{2\pi i jk/2^m} |k\rangle .$$

Plugging the inverse of this transformation into our circuit from before, we obtain:



Thus, measuring the first m qubits and dividing by 2^m tells us precisely the value θ.

Keep in mind, however, that this picture assumes that

$$U |\psi\rangle = e^{2\pi i j/2^m} |\psi\rangle .$$

We still have to worry about the more general case that θ does not have the form $j/2^m$ for some integer j. We will use exactly the same circuit for the general case, but the analysis will become more complicated (and the answer will only be an approximation to θ). In fact we have two major issues to address at this point:

1. What can be said about the measurement outcome in the case that θ does not have the form $j/2^m$ for some integer j, and
2. Can the quantum Fourier transform be implemented efficiently?

We will need the remainder of this lecture and much of the next to address these issues.

General values of θ

The analysis from above showed that, for an arbitrary value of θ, the state of the above circuit immediately before the inverse of the QFT is applied is

$$\frac{1}{2^{m/2}} \sum_{k=0}^{2^m-1} e^{2\pi i k\theta} |k\rangle |\psi\rangle .$$

We know that the second collection of qubits is uncorrelated with the first m qubits, so we are free to disregard these qubits and consider just the first m qubits. Applying the transformation $\mathrm{QFT}_{2^m}^{\dagger}$ to these qubits results in the state

$$\frac{1}{2^m}\sum_{k=0}^{2^m-1}\sum_{j=0}^{2^m-1} e^{2\pi i(k\theta - kj/2^m)}\,|j\rangle = \sum_{j=0}^{2^m-1}\left(\frac{1}{2^m}\sum_{k=0}^{2^m-1} e^{2\pi i k(\theta - j/2^m)}\right)|j\rangle .$$

The probability that the measurement results in outcome j is therefore

$$p_j = \left|\frac{1}{2^m}\sum_{k=0}^{2^m-1} e^{2\pi i k(\theta - j/2^m)}\right|^2$$

for each $j \in \{0,\ldots,2^m-1\}$.

We have already dealt with the case that $\theta = j/2^m$ for some choice of $j \in \{0,\ldots,2^m-1\}$, so let us assume that this is not the case: assume

$$e^{2\pi i(\theta - j/2^m)} \neq 1$$

for every integer j. Using the same formula for the sum of a geometric series from last time,

$$\sum_{k=0}^{n-1} x^k = \frac{x^n-1}{x-1}$$

for $x \neq 1$, we may then simplify:

$$p_j = \frac{1}{2^{2m}}\left|\frac{e^{2\pi i(2^m\theta - j)}-1}{e^{2\pi i(\theta - j/2^m)}-1}\right|^2$$

Our goal will be to show that the probability p_j is large for values of j that satisfy $j/2^m \approx \theta$ and small otherwise. We will do this in the next lecture.