L. Hales
Vazirani Class #
December 9, 1997

Kitaev's Factoring Algorithm

1 Introduction

Today we will present Kitaev's algorithm for factoring. This algorithm, somewhat surprisingly, turns out to be very similar to Shor's, but the approach and ideas are very different. We begin by discussing a basic building block of the algorithm.

2 Finding an Eigenvalue of a Unitary Matrix

Let $U : C^{2^n} \to C^{2^n}$ be some unitary transformation on n qubits with eigenvectors $|e_1\rangle$,...,$|e_n\rangle$.Let λ_j be the eigenvalue of $|e_j\rangle$. Note that $|\lambda_j| = 1$, i.e. $\lambda_j = e^{2\pi i \theta_j}$ for some θ_j such that $0 \leq \theta_j \leq 1$. Suppose we are given as input a circuit for computing U and some superposition of qubits which is an eigenvector $|e\rangle$ of U and we are asked to find the corresponding eigenvalue λ to some precision.

To this end we construct the following circuit - a "controlled U" circuit - where if $b = 0$ then $y = x$ and if $b = 1$ then $y = Ux$, as shown below:

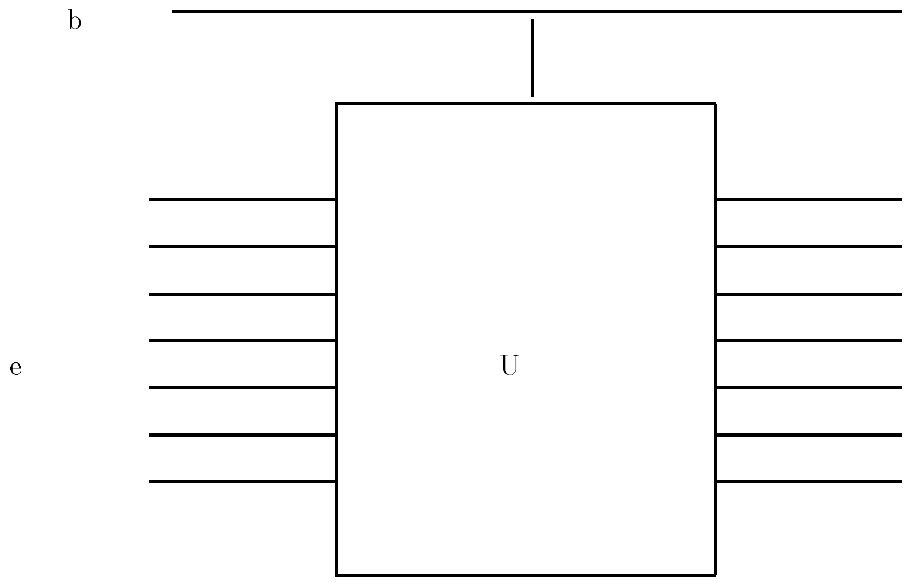

Figure 1: A Controlled U Circuit

We then sandwich the controlled U circuit between two Hadamard gates with b initially set equal to zero, as shown:

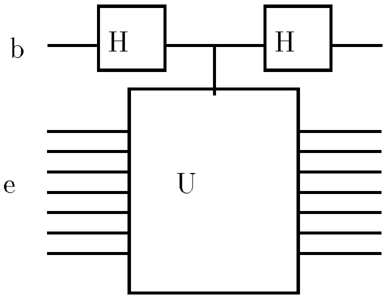

Figure 2: Controlled U with Hadamards

The circuit above performs the following sequence of transformations:

$$
\begin{aligned}
|0\rangle|e\rangle \quad & \rightarrow_H \quad \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|e\rangle & (1)\\
& \rightarrow_{controlU} \quad \frac{1}{\sqrt{2}}|0\rangle|e\rangle + \frac{\lambda}{\sqrt{2}}|1\rangle|e\rangle & (2)\\
& = \quad \frac{1}{\sqrt{2}}(|0\rangle + \lambda|1\rangle)|e\rangle & (3)
\end{aligned}
$$

(Notice that at this point we have put λ into the phase, thus applying another Hadamard and measuring the first bit on the tape will give information about λ.)

$$
\begin{aligned}
\rightarrow_H \quad & \frac{1}{\sqrt{2}}(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|e\rangle + \frac{1}{\sqrt{2}}\lambda(|0\rangle + |1\rangle)|e\rangle) & (4)\\
= \quad & \frac{1}{2}[(1+\lambda)|0> + (1-\lambda)|1>]|e\rangle & (5)
\end{aligned}
$$

Notice that if $\lambda = 1(-1)$ then if we measure the output of the circuit we will see $0(1)$ on the tape with probability 1, thus we can recover λ. In general, we will still get some information about λ. In particular, letting $\lambda = e^{2\pi i\theta} = cos\theta + isin\theta$ and $P(0)$ be the probability of seeing a zero, we see that

$$P(O) = |\frac{1}{2}[1 + cos2\pi\theta + isin2\pi\theta]|^2 = \frac{1 + cos2\pi\theta}{2}$$

and

$$P(1) = |\frac{1}{2}[1 - cos2\pi\theta - isin2\pi\theta]|^2 = \frac{1 - cos2\pi\theta}{2}.$$

If we compose the circuit with itself, as shown below, we can perform a number of independent trials thereby recovering more information about θ. How many such trials need to be performed to estimate θ to a specific precision?(Here we are blurring the distinction between estimating θ and estimating $cos(\theta)$.) Estimating the bias of a coin to within ϵ with probability at least $1-\delta$ requires $t=\Theta(\frac{log(1/\delta)}{\epsilon^2})$ samples. Unfortunately, then, high accuracy is not cheap: to achieve m bits of accuracy we would need approximately 4^m samples. Without some additional assumptions on U, which we will discuss in the next section, this is all we can achieve.

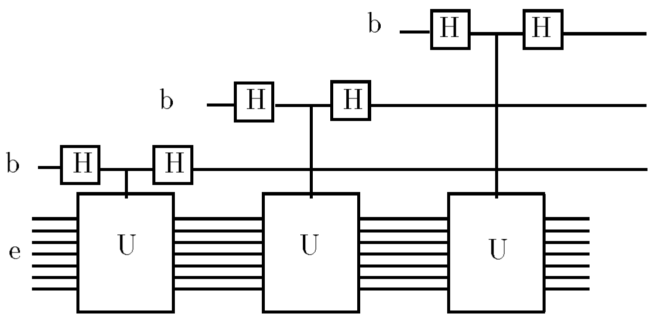

Figure 3: Independent Samples

3 An Additional Assumption

Our additional assumption on U will be that we can compute the transformation U^{2^j} in $poly(j,n)$ steps, in other words we have a circuit C such that $C(x,m)=U^m(x)$ and C is polynomial in the lengths of its inputs. This is a fairly strong assumption, but it happens to be satisfied in many interesting cases. Note that if $|e\rangle$ is an eigenvector of U with eigenvalue λ then $|e\rangle$ is an eigenvector of U^{2^j} with eigenvalue λ^{2^j}. We can use this assumption in the following way:

Let U and $|e\rangle$ be given. Suppose we have used our earlier method to determine which quadrant $2\pi\theta$ is in, i.e. we have determined the first two bits of θ with high confidence. (Since this only requires ϵ to be a constant this can be done efficiently.) Now we wish to determine the next two bits of θ or, equivalently, the first two bits of $4\theta(mod 1)$. But this is the eigenvalue of the matrix U^{2^2} corresponding to the eigenvector $|e\rangle$. Thus, since we can efficiently compute this transformation by assumption, we can apply our earlier method to recover the next two bits of θ with high confidence.

In general, then, if we sample $2^j\theta(mod 1)$ with accuracy $\epsilon=1/8$ and confidence $1-2\delta/m$ for $j=1$ to $m/2$, we will recover the first m bits of θ with confidence $1-\delta$. The number of samples required will be $O(mlog(\frac{1}{\delta}))$. This is efficient, then, as long as the required number of bits, m is not too large.

4 Kitaev's Factoring Algorithm

4.1 Two Preliminary Observations

First, suppose that U satisfies $U^s = I$. Then for each j, $\lambda_j = e^{2\pi i\theta}$ where $\theta = a/b$ for integers a, b satisfying $b \leq s$. Thus we can recover the precise value of λ by finding θ to within $1/s^2$ and then using continued fractions to find the closest such ratio. Hence we need only $log(s^2)$ bits of θ.

Second, in the previously described algorithm we were given as input a specific eigenvector, $|e\rangle$, and we output $|e\rangle|\lambda\rangle$ where λ was the eigenvalue corresponding to $|e\rangle$. If we are given instead an arbitrary superposition of eigenvectors as input, $\sum_j \alpha_j |e_j\rangle$, clearly our algorithm will terminate in the superposition $\sum_j \alpha_j |e_j\rangle|\lambda_j\rangle$ where λ_j is the eigenvalue corresponding to e_j. In particular if we begin with a uniform superposition of eigenvectors then when we measure the final superposition we will see an eigenvalue chosen uniformly at random.

4.2 The Final Algorithm

Like Shor's algorithm, we use the fact that to factor N it is sufficient to be able to compute $ord(a)$ in Z_{N^*} for a randomly chosen a.

Fix $a \in Z_{N^*}$. We wish to find r such that $a^r = 1(modN)$.

We can efficiently compute U, the transformation taking $|x\rangle$ to $|ax(modN)\rangle$, as well as all of its powers. Note also that $U^r = I$, thus we need only $O(log(N))$ bits of θ to find the exact eigenvalues of U. Since U is multiplication by a group element, the eigenvectors of U should be the elements of the fourier basis, i.e. they are of the form

$$|e_j\rangle = \frac{1}{\sqrt{r}} \sum_{l=0}^{r-1} \omega_r^{jl} |ba^l\rangle$$

where b ranges over cosets of the cyclic subgroup generated by a and ω_r is a primitive rth root of unity. It is easy to check that the eigenvalue corresponding to $|e_j\rangle$ is $e^{-\frac{2\pi ij}{r}} = \omega_r^{-j}$.

Since $|1\rangle = \sum_j |e_j\rangle$ is a uniform superposition of eigenvectors if we perform our algorithm using $|1\rangle$ as input, when we measure we will see a random eigenvalue, or equivalently j/r for a random j. Since with high probability j and r are relatively prime, we can recover r by writing the fraction in lowest terms.