Lecture 10: Factorization on a quantum computer

Rajat Mittal

IIT Kanpur

We have already seen examples of HSP's and saw that they can be solved on a finite Abelian group. Shor's algorithm for integer factorization is one of the leading results in the field of quantum computing; it works by noticing that factorization can be reduced to order finding. The second observation is that order finding is an HSP over $\mathbb{Z}$ (Abelian but not a finite group), still the HSP algorithm can be modified to work for $\mathbb{Z}$. We will see both steps, reduction to order finding and then quantum algorithm for order finding.

1 Reduction of factorization to order finding

We will introduce the two problems first.

Factorization: As one would guess, the problem is to find a non-trivial factor of a composite number n. Notice that the input size is $\log n$ and hence we are looking for algorithms which run in polylog(n) time. Since the number of factors are at most $\log n$, we can find all factors of n by applying this algorithm at most $\log n$ times.

Order finding over a group G: Given an element g in a group G, find the order of g in G (smallest r, s.t., $g^r = 1$). It turns out that order finding is an HSP over $\mathbb{Z}$, you will prove this in the assignment.

Since group $\mathbb{Z}$ is not finite, the HSP algorithm needs to be modified slightly to solve order finding. In section 2, the quantum algorithm for order finding will be described.

In this section, we will reduce the factorization of n to order-finding in the group $\mathbb{Z}_n^*$. The group $\mathbb{Z}_n^*$ denotes the set of all elements of $\mathbb{Z}_n$ coprime to n with multiplication as operation. In other words, we are just asking, given n and a (coprime to n), what it the smallest r such that $a^r = 1 \mod n$.

Exercise 1. What is the order of 2 in $\mathbb{Z}_7^*$? What is its order in $\mathbb{Z}_{16}^*$?

Notice a subtlety here, factorization reduces to order finding over $\mathbb{Z}_n^*$ (a finite group). Though, order finding over a group (including $\mathbb{Z}_n^*$) is same as solving HSP over $\mathbb{Z}$ (an infinite group).

Coming back to factorization, We will first get rid of the trivial cases. It can be easily checked if the number is even or if $n = m^k$ (take the square root, cubic root etc. up to $\log n$).

Exercise 2. How can we find square root (and other roots) efficiently?

This allows us to assume that the input to the factorization problem, n, is a number of type kk', where k and k' are co-prime and odd. We are interested in finding a non-trivial factor of n (not 1 or n). As mentioned earlier, we can repeat the procedure to find the complete factorization. The reduction shows that if we can solve order finding on $\mathbb{Z}_n^*$, then we can find a non-trivial factor of n (where n is of above type).

The basic idea of the reduction is to find a non-trivial square root b of 1, i.e.,

$$b: \quad b^2 = 1 \mod n, \; b \neq \pm 1 \mod n$$

Look at the possible square roots of $1 \mod n$, i.e., b for which $b^2 = 1 \mod n$. Clearly there are two trivial solutions, $b = \pm 1 \mod n$. Though, if there exists a square root $b \neq \pm 1 \mod n$, then $b^2 - 1$ is divisible by n and $b \pm 1$ is not. In this case, $gcd(b \pm 1, n)$ will give non-trivial factors of n.

The reduction from factorization to order-finding basically searches for such a non-trivial square root b. The algorithm takes a random $a < n$ and finds its order r. If r is even and $a^{r/2} \neq \pm 1 \mod n$, we have found a non-trivial square root.

```
Check if $n$ is even or of the form $n = m^k$ ;
Pick an $a$, s.t., $gcd(a, n) = 1$ (else we have already found a non-trivial factor of $n$) ;
for $i = 1, \cdots$ do
    Find the order of $a$ and call it $r$ (use the quantum algorithm for order-finding) ;
    if $r$ is odd or $a^{r/2} = -1 \mod n$ then
        Pick another $a$ co-prime to $n$ ;
    else
        Found $b = a^{r/2} \neq \pm 1 \mod n$, square root of 1 ;
        Find the non-trivial factors from $gcd(b \pm 1, n)$ ;
        Break;
    end
end
```

Algorithm 1: Algorithm for factorization using order-finding

Exercise 3. What if a is not in $\mathbb{Z}_n^*$?

Exercise 4. Look at Algorithm 1, and convince yourself that it will output a non-trivial factor.

We are only left to prove that this algorithm works with high probability. That is equivalent to showing that there are enough a's such that,

- order r of a is even,
- and $b = a^{r/2} \neq \pm 1 \mod n$.

Exercise 5. Can it happen that $a^{r/2} = 1 \mod n$?

Note 1. The quantum algorithm is a probabilistic algorithm, hence showing that there are enough *good* a's works.

Following theorem proves that there are enough good a's. It follows from Chinese remaindering and standard number theory arguments. A casual reader can directly skip to the algorithm for order finding.

Theorem 1. *Suppose n is a product of two co-prime numbers $k, k' > 1$. For a randomly chosen a, the probability that a has an even order r and $a^{r/2} \neq -1 \mod n$ is at least $1/4$.*

Before we prove the theorem, we will prove a lemma about numbers of the form $q = p^k$. We introduce the notation, $pow2(z)$, the highest power of 2 that divides any number z. Let $m := \phi(q)$ denote the value of Euler's totient function on q, also the size of group $\mathbb{Z}_q^*$.

Lemma 1. *For a random element from $\mathbb{Z}_q^*$, its order r satisfies $pow2(r) = pow2(m)$ with probability exactly $1/2$.*

Proof. It is easy to calculate Euler function $\phi(q)$.

Exercise 6. Show, $m = p^{k-1}(p-1)$.

Note 2. Read about Euler's totient function ϕ, if you don't know it.

Suppose $l = pow2(m)$ (m is even and hence $l \geq 1$). We need to show, for a random co-prime element to q, exactly half of them have order r for which $pow2(r) = l$.

By theorem 2 in Sec. 1.1, $\mathbb{Z}_q^\times$ is cyclic. Let g be a generator of $\mathbb{Z}_q^*$ (smallest m, s.t., $g^m = 1 \mod q$).

We know that g^t has order,

$$\text{ord}(g^t) = \frac{m}{gcd(m,t)}.$$

If t is odd, then $\gcd(m, t)$ is odd and $pow2(r) = l$. Similarly, if t is even then $gcd(m, t)$ is even and $pow2(r) < l$. Since half the t's are odd and half even, lemma follows.

□

Proof of Thm. 1. Now consider the prime factorization $n = p_1^{i_1} \cdots p_s^{i_s}$. By Chinese remainder theorem,

$$\mathbb{Z}_n^* \cong \mathbb{Z}_{p_1^{i_1}}^* \times \cdots \times \mathbb{Z}_{p_s^{i_s}}^*.$$

So, to randomly chose a, it is equivalent to pick random $a_1, \cdots, a_s$ from the respective $\mathbb{Z}_{p^i}^*$'s. Say, r_j are the orders of a_j modulo $p_j^{i_j}$. Then, r is the LCM of r_j's by Chinese remainder theorem.

Claim. Suppose the order r of a is odd or $a^{r/2} = -1 \mod n$. Then, $pow2(r_j)$ is same for all j.

Proof. The order is odd iff all r_j's are odd. Otherwise, if $a^{r/2} = -1 \mod p_j^{i_j}$ then none of r_j divide $r/2$ (we use the fact that p_i's are not 2).

All r_j's divide r but not $r/2$, so $pow2(r_j)$ is the same (equal to $pow2(r)$). □

From lemma 1, with half the probability, The order r_j of a_j will be such that $pow2(r_j) = l_j$ (where $l_j = pow2(p_j^{i_j - 1}(p_j - 1))$). Call the case when $pow2(r_j) = l_j$ as the "first" case and other the "second" case. We know that both cases happen with probability 1/2.

Notice that l_j's only depend on n. If all l_j are equal, pick a_1's from first case and a_2's from the second case. If they are unequal, say $l_1 \neq l_2$, then pick a_1, a_2 from the first case.

In either scenario, r_j's can't be all equal. Which implies r is even and $a^{\frac{r}{2}} \neq 1 \mod n$ (by claim). Since we have only fixed at most 2 cases out of s, the probability is at least 1/4.

□

Hence the reduction from factorization to order finding is complete.

1.1 Extra reading: $\mathbb{Z}_{p^k}$ is cyclic

The proof of the following theorem is given for completeness. Interested readers can take a look.

Theorem 2. *If $n = p^k$ for some power k of an odd prime p then $G = \mathbb{Z}_n^\times$ is cyclic.*

Note 3. This is not true for even prime, $\mathbb{Z}_8^\times$ is not cyclic.

Exercise 7. Find out where did we use the fact that p is odd.

Proof. Assume that $t = p^{k-1}(p-1)$, the order of the group G.

We know that $\mathbb{Z}_p^\times$ is cyclic [2], and hence have a generator g. We will use g to come up with a generator of G. First notice that,

$$(g+p)^{p-1} = g^{p-1} + (p-1)g^{p-2}p \neq g^{p-1} \mod p^2.$$

So either $(g+p)^{p-1}$ or g^{p-1} is not $1 \mod p^2$. We can assume the latter case, otherwise replace g by $g+p$ in the argument below.

So $g^{p-1} = 1 + k_1 p$ where $p \nmid k_1$. So using binomial theorem,

$$g^{p(p-1)} = (1 + k_1 p)^p = 1 + k_2 p^2,$$

where $p \nmid k_2$.

Exercise 8. Continuing this process, show that,

$$g^{p^{e-1}(p-1)} = 1 + k_e p^e,$$

with $p \nmid k_e$.

From the previous exercise $g^t = 1 \mod p^k$ but $g^{t/p} \neq 1 \mod p^k$. The only possible order of g then is $p^{k-1}d$ where d is a divisor of $p-1$ (because the order has to divide t, Lagrange's theorem).

If the order is $p^{k-1}d$, then

$$g^{p^{k-1}d} = 1 \mod p^k = 1 \mod p.$$

But $g^p = g \mod p$ (why?). That implies $g^d = 1 \mod p$. Since $p-1$ is the order of g modulo p (g is the generator), implies $d = p-1$. Hence proved.

□

2 Order finding algorithm

You will prove in the assignment that order finding can be thought of as an HSP over $\mathbb{Z}$. Since $\mathbb{Z}$ is not finite (though abelian), we will solve it by a seemingly different technique (this perspective of factoring algorithm is due to Kitaev). The explanation here is inspired from John Watrous's course notes (https://cs.uwaterloo.ca/ watrous/QC-notes/QC-notes.10.pdf). In assignment, you will show that this technique is very similar to solving HSP over a finite abelian group (also look at the circuit for order finding given below). You can read about the usual approach (HSP based) from multiple sources, e.g., [3] and [1].

For factorization, we need to solve order finding over group $\mathbb{Z}_n^*$. That means, given an a co-prime to n, we need to find the smallest positive r for which $a^r = 1 \mod n$.

Note 4. Order finding is not an HSP over $\mathbb{Z}_n^*$ but on $\mathbb{Z}$.

Let us take a look at the algorithm for order finding using phase estimation. Suppose, we are interested in finding the order of a modulo n (a and n are given and they are coprime).

Exercise 9. How can we efficiently find if two numbers are coprime?

First approach:

Let k be the smallest number, such that, $2^k \geq n$. Consider the Hilbert space $\mathbb{C}^{2^k}$ spanned by $|b\rangle$, where b ranges from 0 to $2^k - 1$. Define the unitary operator,

$$U|b\rangle = |ab \mod n\rangle \textit{ for } b \in \mathbb{Z}_n^*.$$

The unitary is not completely specified, but we are only interested in these basis states. The action on the other basis states can be assumed to be identity.

Exercise 10. Show that it is a unitary operator because a is coprime to n. How can we implement this unitary?

Since a has order r, it can be observed that $U^r = I$. From spectral decomposition, the eigenvalues of U have to be rth roots of unity. In other words, possible eigenvalues of U are $e^{2\pi i \frac{s}{r}}$, where s is an integer between 0 and $r-1$. The following exercise explicitly finds the eigenvalues and eigenvectors of U.

Exercise 11. Show that $|u_s\rangle$ is an eigenvector of U with the eigenvalue $e^{2\pi i \frac{s}{r}}$. Where,

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{2\pi i j \frac{s}{r}} |a^j \mod n\rangle.$$

This analysis seems to suggest a very simple algorithm given s and the eigenvector $|u_s\rangle$. We can apply phase estimation on the unitary U with state $|u_s\rangle$. The output will be a good approximation of $\frac{s}{r}$, we can get r from this information. Notice that the powers U^{2^j} can be implemented using repeated squaring. Unfortunately, we don't have state $|u_s\rangle$ for any s.

Modified approach:

The idea would be to apply phase estimation on a superposition of eigenvectors. By linearity, we will get a particular eigenvalue with probability according to the amplitude of the corresponding eigenvector in superposition (we saw this in the phase estimation lecture). It turns out that the state $|1\rangle$ (which we can definitely prepare) can be written as a linear combination of these eigenvectors. In fact, it is an easy exercise to prove that,

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle.$$

Notice that even the amplitude on each eigenvector is same! We will apply phase estimation on $|1\rangle$ for unitary U; we will get a good approximations of $\frac{s}{r}$, with equal probability for all s. How can we obtain r from $\frac{s}{r}$?

We can repeat this phase estimation multiple times and obtain various approximations of s/r for different s (all s occur with equal probability). These actually correspond to different characters trivial on the subgroup $r\mathbb{Z}$. As in all hidden subgroup problems, we need to find our subgroup from these characters. The strategy for finite Abelian group will not work here.

Finishing the algorithm, classical part:

Our remaining task is to find r given multiple approximations to various $\frac{s}{r}$. The next insight is the following theorem about continued fractions (for reference, look at [3]). It shows that given a *good enough* approximation to an $\frac{s}{r}$, where s and r are coprime, there exist a continued fractions algorithm which can recover s and r from this approximation.

Theorem 3 ([3]). *Suppose $\frac{s}{r}$ is a rational number in the lowest form (no common factor between s and r), s.t.,*

$$\left|\frac{s}{r} - \phi\right| \leq \frac{1}{2r^2}.$$

If $r < n$, then s, r can be obtained in poly(L) time from ϕ, where $L = \lceil \log n \rceil$.

Note 5. The continued fractions algorithm is a very beautiful mathematical fact. Unfortunately, it will not be covered in this course due to lack of time. To read more about the continued fractions algorithm, please refer to [3] and references therein.

The consequence of continued fraction theorem is: if we get a *good enough* approximation of s/r (for an s coprime to r), we can find r. Let us take care of these two issues.

Exercise 12. It will be helpful if you remember the details of phase estimation. Please take a look at the notes for phase estimation.

- *Precision*: Since $r \leq n$, it is sufficient to get an approximation which is $\frac{1}{2n^2}$ close. That means, we need the first $2L+1$ bits of phase to be correct (remember $L = \lceil \log n \rceil$). So, we need to run phase estimation on $2L + 1 + f(\epsilon)$ qubits and it will give us phase with probability $1 - \epsilon$ (this follows from our discussion on phase estimation). We can extend U to $\mathbb{C}^{2^{2L+1}}$ by trivial action on other basis states.

  Exercise 13. Do you remember the definition of U? What was its action on basis states?

- *Coprime s*: Can you think of a way to find an s coprime to r? It will again be our usual trick, find lots of s, one of them will be coprime to r. Prime number theorem states that the number of primes less than n are around $O(\frac{n}{\log n})$.

  Exercise 14. Show that the number of s, co-prime to r, are at least $O(\frac{1}{L})$ fraction of total s using prime number theorem.

  If we have $O(L)$ different s (picked randomly), with high probability one of them will be coprime to r. Continued fraction algorithm on that s/r will give us the order r.

Exercise 15. Can you summarize the complete algorithm for factorization now? Try to draw a circuit for it.

Hence the order finding algorithm can be summed up as: repeat the phase estimation on U at least $O(L)$ times. The precision for phase estimation would be $2L + 1 + f(\epsilon)$. The circuit for order finding is given in Fig. 1.

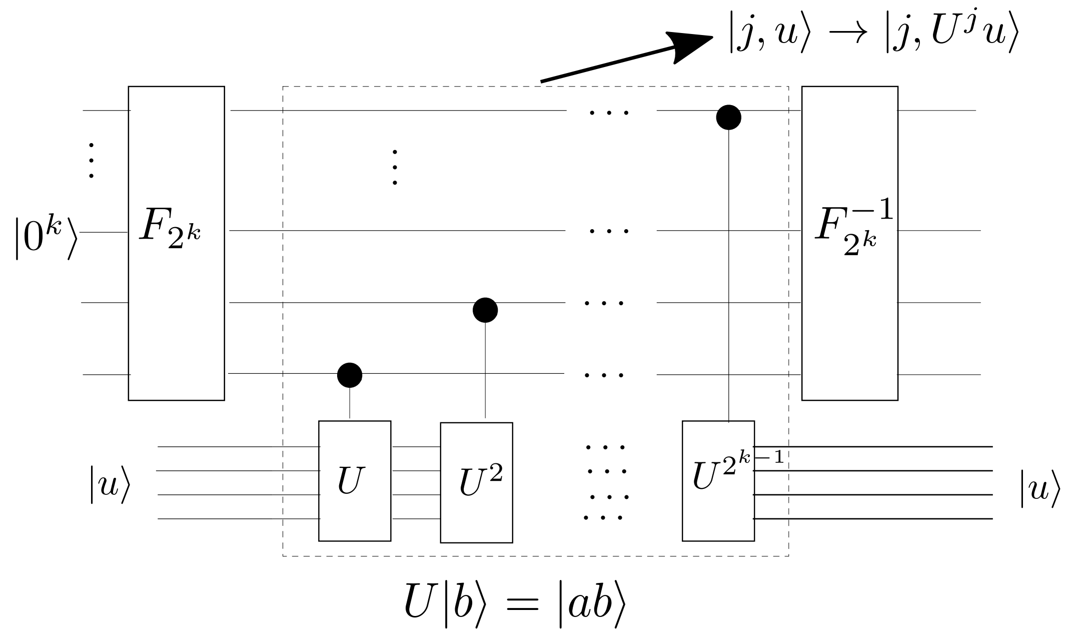

Fig. 1. Order finding algorithm

3 Assignment

Exercise 16. **Lemma 2.** *Given a positive number α, let there exists a sequence of rationals $\frac{p_n}{q_n}$, such that, $|q_n\alpha - p_n| \neq 0$ tends to zero. Then, α is irrational.*

Prove the lemma by showing that if $\alpha = a/b$ then $|q_n\alpha - p_n| \geq 1/b$. Using the lemma, show that e is irrational.

Exercise 17. We had claimed that order finding is an HSP.

- Show that $g^j \neq g^k$ if $0 \leq j < k < r$. Where r is the order of g.
- Show that order finding is an HSP in $\mathbb{Z}$.

Exercise 18. Using Chinese remainder theorem, show that there exists a b, such that, $b^2 = 1 \mod n$ and $b \neq \pm 1 \mod n$. Here n contains at least two distinct primes in its factorization.

Exercise 19. Why is the phase estimation algorithm for order finding, described above, is same as the HSP algorithm.

Exercise 20. This exercise is kind of a sanity check on continued fractions algorithm. Show that there is no other rational number $\frac{s'}{r'}$, where $r' < r$ and

$$\left|\frac{s'}{r'} - \phi\right| \leq \frac{1}{2r^2}.$$

References

1. A. Childs. Quantum algorithms, 2013.
2. R. Lidl and H. Niederreiter. *Finite Fields.* Cambridge University Press, 1997.
3. M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge, 2010.