Problem Set 2

1. Generalized Bernstein-Vazirani Let M, n be a positive integers. Let $s \in \mathbb{Z}_M^n$ and define the function $f_s : \mathbb{Z}_M^n \to \mathbb{Z}_M$ by $f_s(x) = \langle s, x \rangle \mod M$. Given access to an oracle O_{f_s} which for $x \in \mathbb{Z}_M^n, b \in \mathbb{Z}_M$ acts as $O_{f_s}|x\rangle|b\rangle = |x\rangle|b + f_s(x) \bmod M\rangle$, design a quantum algorithm that computes s with one application of O_{f_s}.

Hint: You may want to generalize the "phase-kickback trick" to show with the oracle O_{f_s} you can also implement an oracle O'_{f_s} with the behavior

$$O'_{f_s}|x\rangle|b\rangle = \omega^{-f_s(x)\cdot b}|x\rangle|b\rangle$$

where $\omega = e^{2\pi \mathrm{i}/M}$.

2. Continued fractions In the classical post-processing of Shor's period finding algorithm we have a fraction b/N and want to find the best rational approximation to this number whose denominator is at most M. In lecture we said this can be done in polynomial time as the task can be written as a two-variable integer linear program. Now we see a direct way to do this via continued fraction expansion. A nice discussion of continued fractions, including all the material below, can be found in Chapter 10 of Hardy and Wright's An introduction to the theory of numbers.

A finite continued fraction is an expression of the form

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots + \cfrac{1}{a_t}}}}} \quad .$$

We will denote this number by $[a_0, \ldots, a_t]$. For $0 \le j \le t$ we call $[a_0, \ldots, a_j]$ the j^{th} convergent to $[a_0, \ldots, a_p]$. A continued fraction $[a_0, \ldots, a_t]$ is called *simple* if $a_1, \ldots, a_p$ are all positive integers (a_0 can be non-positive). Every rational number can be represented by a finite simple continued fraction.

Here is an algorithm to find such a representation. Let x be a positive rational number. Then set

$$\begin{aligned} a_0 &= \lfloor x \rfloor, \quad x_1 = \frac{1}{x - a_0} \\ a_1 &= \lfloor x_1 \rfloor, \quad x_2 = \frac{1}{x_1 - a_1} \\ a_2 &= \lfloor x_2 \rfloor, \quad x_3 = \frac{1}{x_2 - a_2} \\ &\cdots \end{aligned}$$

The essential principle at work here is that $x = a_0 + \frac{1}{a_1'}$ where $a_1' = \frac{1}{x - a_0}$. Then since $[a_0, [a_1, \ldots, a_t]] = [a_0, a_1, \ldots, a_t]$ our task becomes to find a continued fraction expansion of a_1' which we do by the same procedure.

One can also find an inductive expression for $[a_0, \ldots, a_j]$. If

$$\begin{aligned} p_0 &= a_0, & p_1 &= a_1 a_0 + 1, & p_j &= a_j p_{j-1} + p_{j-2} \\ q_0 &= 1, & q_1 &= a_1, & q_j &= a_j q_{j-1} + q_{j-2} \end{aligned}$$

then $[a_0, \ldots, a_j] = \frac{p_j}{q_j}$ and this is in lowest terms. Note that $q_j \geq 2q_{j-2}$ thus q_j increases at least exponentially. An important property of the continued fraction expansion for the application in Shor's algorithm is that if

$$|x - \frac{c}{d}| \leq |x - \frac{p_j}{q_j}|$$

then $d \geq q_j$.

Now the questions:

1. Find the continued fraction expansion of $\frac{527}{1024}$.
2. Look at the j^{th} convergents of your expression and make a conjecture about the even and odd numbered convergents (you do not need to prove it).
3. (Optional but could be helpful for Problem 3) Write a program in any language to compute a continued fraction of an input number up to a given accuracy.

3. Factoring 21 Let's factor the number $M = 21$ using Shor's algorithm.

1. List all numbers in $\mathbb{Z}_{21}$ that are relatively prime to 21. These are the elements of the multiplicative group $\mathbb{Z}_{21}^\times$. Compute the order $\text{ord}_{21}(x)$ of all elements in $\mathbb{Z}_{21}^\times$.
2. Recall that in Shor's algorithm we want to find an x of even order d such that $x^{d/2} \neq -1 \mod M$. Call such an x *good*. Identify all the good $x \in \mathbb{Z}_{21}^\times$ with $\text{ord}_{21}(x) = 6$ and for these verify that $\gcd(x^3 \pm 1, 21)$ gives a nontrivial factor of 21.
3. Choose a good x of order 6 from the previous step. Now let's simulate finding the period of $f(j) = x^j \mod 21$. Using the Octave FTperiod program `https://github.com/troyjlee/qalgo/tree/main/CODE` with $N = 21^2, s = 6$. (You can run Octave programs online at `https://octave-online.net/`.) This simulates randomly sampling a state $|g_t\rangle$ and measuring $F_N|g_t\rangle$ to see an index b. Use continued fraction expansion on b/N and see if you can recover $\text{ord}_{21}(x)$. It may take several attempts. Record the values you see and how many attempts it takes.

4. Assumptions Where in the proof of correctness of Shor's algorithm for the general period finding problem with a function $f : \mathbb{Z}_N \to [M]$ do we use the assumption that $N > M^2$? What can go wrong without this assumption?

5. Nielsen and Chuang At the end of this problem set is attached the section from Nielsen and Chuang on period finding (section 5.4.1). Compared to our description in lecture, the Nielsen and Chuang presentation removes a key assumption about the function f. Does the algorithm still work without this assumption? If not, what can go wrong? You can describe it in words or show a numerical example in your favourite language.

6. Finding all ones Let $N = 2^n$ and $x \in \{0,1\}^N$ and *assume you know* that x has k many ones.

1. In lecture we showed how to find an $i \in N$ such that $x_i = 1$ with constant probability by a quantum algorithm after $O(\sqrt{N/k})$ many queries to x. Show how to boost this success probability to $1 - 1/N^2$ using $O(\sqrt{N/k}\log(N))$ many queries to x.

2. Give a quantum algorithm to find *all* the ones in x with constant probability after $O(\sqrt{kN}\log(N))$ many queries to x.

of the function may be very intricate. In order to present this problem in the most approachable manner, we begin with two more specific applications: period-finding (of a one-dimensional function), and discrete logarithms. We then return to the general hidden subgroup problem. Note that the presentation in this section is rather more schematic and conceptual than earlier sections in this chapter; of necessity, this means that the reader interested in understanding all the details will have to work much harder!

5.4.1 Period-finding

Consider the following problem. Suppose f is a periodic function producing a single bit as output and such that $f(x + r) = f(x)$, for some unknown $0 < r < 2^L$, where $x, r \in \{0, 1, 2, \ldots\}$. Given a quantum black box U which performs the unitary transform $U|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$ (where $\oplus$ denotes addition modulo 2) how many black box queries and other operations are required to determine r? Note that in practice U operates on a finite domain, whose size is determined by the desired accuracy for r. Here is a quantum algorithm which solves this problem using *one* query, and $O(L^2)$ other operations:

Algorithm: Period-finding

Inputs: (1) A black box which performs the operation $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, (2) a state to store the function evaluation, initialized to $|0\rangle$, and (3) $t = O(L + \log(1/\epsilon))$ qubits initialized to $|0\rangle$.

Outputs: The least integer $r > 0$ such that $f(x + r) = f(x)$.

Runtime: One use of U, and $O(L^2)$ operations. Succeeds with probability $O(1)$.

Procedure:

1.	$\|0\rangle\|0\rangle$	initial state
2.	$\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} \|x\rangle\|0\rangle$	create superposition
3.	$\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} \|x\rangle\|f(x)\rangle$	apply U
	$\approx \frac{1}{\sqrt{r2^t}} \sum_{\ell=0}^{r-1} \sum_{x=0}^{2^t-1} e^{2\pi i \ell x/r} \|x\rangle\|\hat{f}(\ell)\rangle$	
4.	$\rightarrow \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} \widetilde{\|\ell/r\rangle}\|\hat{f}(\ell)\rangle$	apply inverse Fourier transform to first register
5.	$\rightarrow \widetilde{\ell/r}$	measure first register
6.	$\rightarrow r$	apply continued fractions algorithm

The key to understanding this algorithm, which is based on phase estimation, and is nearly identical to the algorithm for quantum order-finding, is step **3**, in which we introduce the state

$$|\hat{f}(\ell)\rangle \equiv \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} e^{-2\pi i \ell x/r} |f(x)\rangle \,, \tag{5.63}$$

the Fourier transform of $|f(x)\rangle$. The identity used in step **3** is based on

$$|f(x)\rangle = \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} e^{2\pi i \ell x/r} |\hat{f}(\ell)\rangle \,, \tag{5.64}$$

which is easy to verify by noting that $\sum_{\ell=0}^{r-1} e^{2\pi i \ell x/r} = r$ for x an integer multiple of r, and zero otherwise. The approximate equality in step **3** is required because 2^t may not be an integer multiple of r in general (it need not be: this is taken account of by the phase estimation bounds). By Equation (5.22), applying the inverse Fourier transform to the first register, in step **4**, gives an estimate of the phase ℓ/r, where ℓ is chosen randomly. r can be efficiently obtained in the final step using a continued fraction expansion.

Box 5.5: The shift-invariance property of the Fourier transform

The Fourier transform, Equation (5.1), has an interesting and very useful property, known as *shift invariance*. Using notation which is useful in describing the general application of this property, let us describe the quantum Fourier transform as

$$\sum_{h \in H} \alpha_h |h\rangle \rightarrow \sum_{g \in G} \tilde{\alpha}_g |g\rangle \,, \tag{5.65}$$

where $\tilde{\alpha}_g = \sum_{h \in H} \alpha_h \exp(2\pi i g h/|G|)$, H is some subset of G, and G indexes the states in an orthonormal basis of the Hilbert space. For example, G may be the set of numbers from 0 to $2^n - 1$ for an n qubit system. $|G|$ denotes the number of elements in G. Suppose we apply to the initial state an operator U_k which performs the unitary transform

$$U_k |g\rangle = |g + k\rangle \,, \tag{5.66}$$

then apply the Fourier transform. The result,

$$U_k \sum_{h \in H} \alpha_h |h\rangle = \sum_{h \in H} \alpha_h |h + k\rangle \rightarrow \sum_{g \in G} e^{2\pi i g k/|G|} \tilde{\alpha}_g |g\rangle \tag{5.67}$$

has the property that the magnitude of the amplitude for $|g\rangle$ does not change, no matter what k is, that is: $|\exp(2\pi i g k/|G|)\tilde{\alpha}_g| = |\tilde{\alpha}_g|$.
In the language of group theory, G is a group, H a subgroup of G, and we say that if a function f on G is constant on cosets of H, then the Fourier transform of f is invariant over cosets of H.

Why does this work? One way to understand this is to realize that (5.63) is approximately the Fourier transform over $\{0, 1, \ldots, 2^L - 1\}$ of $|f(x)\rangle$ (see Exercise 5.20), and the Fourier transform has an interesting and very useful property, known as *shift invariance*, described in Box 5.5. Another is to realize that what the order-finding algorithm does is just to find the period of the function $f(k) = x^k \bmod N$, so the ability to find the period of a general periodic function is not unexpected. Yet another way is to realize that the implementation of the black box U is naturally done using a certain unitary operator whose eigenvectors are precisely $|\hat{f}(\ell)\rangle$, as described in Exercise 5.21 below, so that the phase estimation procedure of Section 5.2 can be applied.