Chapter 5

Quantum Fourier Transform

Many problems in physics and mathematics are solved by transforming a problem into some other problem with a known solution. Some notable examples are Laplace transform, Legendre transform, etc., but by far the most commonly used transformations is the Fourier transformation. A discrete version Fourier transformation can be defined as

$$y_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i\, j\, k/N} \tag{5.1}$$

where $x = (x_{0,} x_1, ..., x_N)$ is the vector that is being transformed.

The quantum Fourier transform in based on essentially the same idea with the only difference that the vectors x and y are state vectors, i.e.

$$\begin{aligned} x &= \sum_{j=0}^{N-1} x_j |j\rangle & (5.2) \\ y &= \sum_{j=0}^{N-1} y_j |j\rangle. & (5.3) \end{aligned}$$

Then the action on the components of state vector x is described by 5.1 so that

$$\sum_{j=0}^{N-1} x_j |j\rangle \to \sum_{k=0}^{N-1} y_k |k\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \sum_{j=0}^{N-1} x_j e^{2\pi i\, j\, k/N} |k\rangle \tag{5.4}$$

or

$$\sum_{j=0}^{N-1} x_j |j\rangle \to \sum_{j=0}^{N-1} x_j \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i\, j\, k/N} |k\rangle \right). \tag{5.5}$$

In other words the incoming amplitude x_j of a given basis vector $|j\rangle$ (in original or "position" space) is distributed among all basis vector (in Fourier or "momentum" space)

$$x_j|j\rangle \to x_j \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i\, j\, k/N} |k\rangle \right). \tag{5.6}$$

What is, however, different is that both the original vector x and the transformed vector y is recorded using the very same Hilbert space.

5.1 Quantum Circuit

In the case of quantum computation the basis vectors $|j\rangle$ are the computational basis vectors (i.e. $j \in \mathbb{N}$) for let say n q-bits. Then it will be useful to adopt the binary representation

$$j\ : \mathbb{N} \to \{0,1\}^n \tag{5.7}$$

such that

$$j \equiv [j_1 j_2 ... j_n] \equiv \sum_{i=1}^{n} j_i 2^{n-i} \tag{5.8}$$

and binary function

$$0.\ : \{0,1\}^m \to (0,1) \tag{5.9}$$

such that

$$0.(j_1, j_2, ..., j_m) \equiv \sum_{i=1}^{m} j_i 2^{-m}. \tag{5.10}$$

Then (5.6) implies

$$\begin{aligned}
|j\rangle = |[j_1 j_2 ... j_n]\rangle \;\to\; & 2^{-\frac{n}{2}} \sum_{k=0}^{2^n-1} \exp\left(2\pi i\, j\, k\, 2^{-n}\right) |[k_1 ... k_n]\rangle = \\
= \; & 2^{-\frac{n}{2}} \sum_{k_1=0}^{1} ... \sum_{k_n=0}^{1} \exp\left(2\pi i\, j \left(\sum_{l=1}^{n} k_l 2^{n-l}\right) 2^{-n}\right) |[k_1 ... k_n]\rangle \\
= \; & 2^{-\frac{n}{2}} \sum_{k_1=0}^{1} ... \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} \left(\exp\left(2\pi i\, j\, k_l 2^{-l}\right) |k_l\rangle\right) \\
= \; & 2^{-\frac{n}{2}} \bigotimes_{l=1}^{n} \left(\sum_{k_l=0}^{1} \exp\left(2\pi i\, j\, k_l 2^{-l}\right) |k_l\rangle\right) \\
= \; & 2^{-\frac{n}{2}} \bigotimes_{l=1}^{n} \left(|0\rangle + \exp\left(2\pi i\, j\, 2^{-l}\right) |1\rangle\right) \\
= \; & 2^{-\frac{n}{2}} \bigotimes_{l=1}^{n} \left(|0\rangle + \exp\left(2\pi i \left(\sum_{k=1}^{n} j_k 2^{n-k}\right) 2^{-l}\right) |1\rangle\right) \\
= \; & 2^{-\frac{n}{2}} \left(|0\rangle + \exp\left(2\pi i\, 0.(j_n)\right) |1\rangle\right) ... \left(|0\rangle + \exp\left(2\pi i\, 0.(j_1 ... j_n)\right) |1\rangle\right)
\end{aligned} \tag{5.11}$$

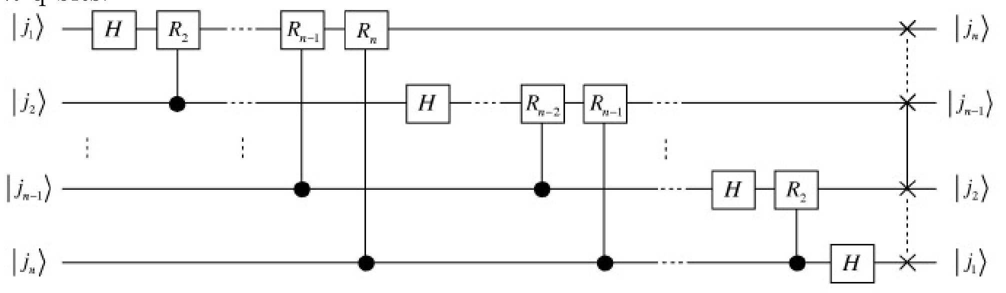
where in the last step we used

$$\begin{aligned}
\exp\left(2\pi i \left(\sum_{k=1}^{n} j_k 2^{(n-l)-k}\right)\right) &= \exp\left(2\pi i \left(\sum_{k=1}^{n-l} j_k 2^{(n-l)-k}\right)\right) \times \exp\left(2\pi i \left(\sum_{k=n-l+1}^{n} j_k 2^{(n-l)-k}\right)\right) \\
&= \exp\left(2\pi i \left(\sum_{k=n-l+1}^{n} j_k 2^{(n-l)-k}\right)\right).
\end{aligned} \tag{5.12}$$

Then we can construct a collection of two q-bit gates which describe rotations of relative phase

$$R_k \equiv \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i\, 2^{-k}} \end{pmatrix}. \tag{5.13}$$

These rotations can be used to construct the following circuit on a system of n q-bits:



It is easy to see that the above circuit would perform a quantum Fourier transform on the input gates. After the first Hadamard gate we get

$$2^{-\frac{1}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_1)}|1\rangle\right)|j_2...j_n\rangle \tag{5.14}$$

since

$$e^{2\pi i\, 0.(j_1)} = \begin{cases} +1 & \text{if } j_1 = 0 \\ -1 & \text{if } j_1 = 1, \end{cases} \tag{5.15}$$

after the controlled-R_2 gate

$$2^{-\frac{1}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_1 j_2)}|1\rangle\right)|j_2...j_n\rangle \tag{5.16}$$

and so on until after the controlled-R_n gate we get

$$2^{-\frac{1}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_1 j_2 .. j_n)}|1\rangle\right)|j_2...j_n\rangle. \tag{5.17}$$

Then we do a similar operation on the second bit to get

$$2^{-\frac{2}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_1 j_2 .. j_n)}|1\rangle\right)\left(|0\rangle + e^{2\pi i\, 0.(j_2 .. j_n)}|1\rangle\right)|j_3...j_n\rangle \tag{5.18}$$

and so on until we get

$$2^{-\frac{n}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_1 j_2 ... j_n)}|1\rangle\right)\left(|0\rangle + e^{2\pi i\, 0.(j_2 ... j_n)}|1\rangle\right)...\left(|0\rangle + e^{2\pi i\, 0.(j_n)}|1\rangle\right) \tag{5.19}$$

and finally we can perform a swap operation to reverse the order of q-bits

$$2^{-\frac{n}{2}}\left(|0\rangle + e^{2\pi i\, 0.(j_n)}|1\rangle\right)...\left(|0\rangle + e^{2\pi i\, 0.(j_1 j_2 .. j_n)}|1\rangle\right) \tag{5.20}$$

which is the desired result according to the product expansion of (5.11).

How many gates have we used? It is

$$n + (n-1) + ... + 1 \tag{5.21}$$

plus $n/2$ swaps each using three C-NOT gates

$$n + (n-1) + ... + 1 + 3 \times \frac{n}{2} = O(n^2) \tag{5.22}$$

swaps. The best classical algorithm of preforming Fourier transform (namely Fast Fourier Transform or FFT) on a vector in 2^n dimensional space uses exponentially many gates

$$\Theta\left(n2^n\right). \tag{5.23}$$

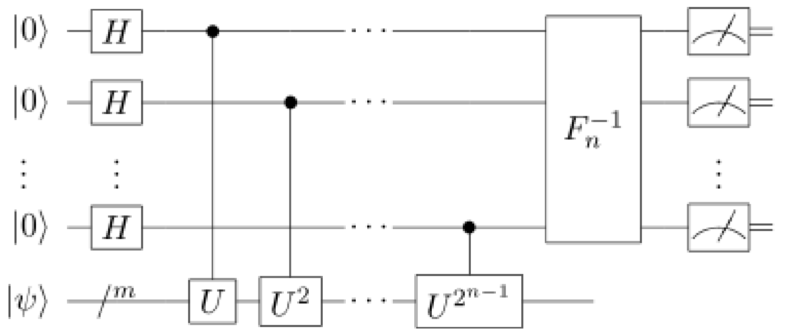
Of course, keep in mind that not all of the information about the Fourier transformed state vector can be retrieved, but one can still use it to design efficient quantum algorithms using the so-called phase estimation procedure.

5.2 Phase Estimation

Consider a unitary operator U whose eigenvector $|\psi\rangle$ has eigenvalue $e^{2\pi i\varphi}$, i.e.

$$U|\psi\rangle = e^{2\pi i\varphi}|\psi\rangle. \tag{5.24}$$

Let us also suppose that we are able to preform controlled-U^{2^k} (for any k) operations on state vector $|\psi\rangle$ and our task is to estimate φ. Then we can construct the following quantum circuit:



where F_n^{-1} represents an inverse Quantum Fourier Transform on n q-bits. The initial state of the system is

$$|0\rangle^{\otimes n}|\psi\rangle \tag{5.25}$$

after applying the Hadamard gates it is

$$2^{-\frac{n}{2}} \sum_{j=0}^{2^n-1} |j\rangle|\psi\rangle \tag{5.26}$$

all of the controlled operations (but before F_n^{-1}) the state of the systems is

$$2^{-\frac{n}{2}} \left(|0\rangle + e^{2\pi i\,2^0\varphi}|1\rangle\right)\left(|0\rangle + e^{2\pi i\,2^1\varphi}|1\rangle\right)...\left(|0\rangle + e^{2\pi i\,2^{n-1}\varphi}|1\rangle\right)|\psi\rangle = 2^{-\frac{n}{2}}\sum_{j=0}^{2^n-1} e^{2\pi i\,\varphi j}|j\rangle|\psi\rangle. \tag{5.27}$$

For example if

$$\varphi = [0.\varphi_1\varphi_2...\varphi_n] \tag{5.28}$$

then Eq. (5.27) can be written as

$$2^{-\frac{n}{2}} \left(|0\rangle + e^{2\pi i\,[0.\varphi_1\varphi_2...\varphi_n]}|1\rangle\right)\left(|0\rangle + e^{2\pi i\,[0.\varphi_2...\varphi_n]}|1\rangle\right)...\left(|0\rangle + e^{2\pi i\,[0.\varphi_n]}|1\rangle\right)|\psi\rangle. \tag{5.29}$$

This is exactly what we we had in (5.19), and thus by reversing the Quantum Fourier Transformation circuit (but without swapping bits) we get

$$|\tilde{\varphi}\rangle|\psi\rangle \tag{5.30}$$

where $\tilde{\varphi}$ is an estimated value of φ (to first n bits in binary expansion).

5.3 Order-finding

For positive integers x and N such that $x < N$ the order of x modulo N is defined to be the smallest integer r such that

$$x^r = 1(\mod N). \tag{5.31}$$

or in other words there exist $R \in \mathbb{N}$ such that

$$x^r = RN + 1. \tag{5.32}$$

Note that

$$x^0 \,(\mod N) = x^r \,(\mod N) = x^{2r} \,(\mod N) \ldots \tag{5.33}$$

The problem of finding r when x and N have no common factors is believed to be a hard problem on a classical computer in a sense that no algorithm is known to solve the problem using polynomial resources in the number of bits, i.e.

$$O\left((\log N)^k\right) \tag{5.34}$$

for any k. There is however an efficient quantum algorithm based on phase estimation.

Consider a collection of unitary operators U_x which act on state vectors as

$$U_x|y\rangle \equiv |xy \,(\mod N)\rangle \tag{5.35}$$

where $y \in \{0,1\}^L$ and L is the number of q-bits. It is also assumed that U acts non-trivially only if $y < N$, i.e.

$$xy\,(\mod N) = \begin{cases} xy\,(\mod N) & \text{if } 0 \le y < N \\ y & \text{if } N \le y \le 2^L - 1. \end{cases} \tag{5.36}$$

Then the eigenstates of U are

$$|u_s\rangle \equiv r^{-1/2} \sum_{k=0}^{r-1} \exp\left(\frac{-2\pi i\, sk}{r}\right) \, |x^k \,(\mod N)\rangle \tag{5.37}$$

with eigenvalues

$$\exp\left(\frac{2\pi i\, s}{r}\right) \tag{5.38}$$

for $0 \leq s \leq r-1$. This can be verified by direct computation

$$\begin{aligned}
U|u_s\rangle &= r^{-1/2}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^{k+1}\,(\mod N)\rangle \\
&= \exp\left(\frac{2\pi i\, s}{r}\right) r^{-1/2}\sum_{k=1}^{r}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^{k}\,(\mod N)\rangle \\
&= \exp\left(\frac{2\pi i\, s}{r}\right) r^{-1/2}\left(|x^{r}\,(\mod N)\rangle + \sum_{k=1}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^{k}\,(\mod N)\rangle\right) \\
&= \exp\left(\frac{2\pi i\, s}{r}\right) r^{-1/2}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^{k}\,(\mod N)\rangle \\
&= \exp\left(\frac{2\pi i\, s}{r}\right)|u_s\rangle. \qquad (5.39)
\end{aligned}$$

So, if we are able to create such a state then we can use the phase estimation procedure to determine s/r form which the order r can be determined. But for that we should first show how to prepare any of the state vectors $|u_s\rangle$ (with non-trivial eigenvalues) and also to implement controlled-U^{2^k} (for any integer k) used in the phase estimation procedure.

In fact the application of controlled-U^{2^k} gates phase estimation procedure is equivalent to

$$\begin{aligned}
|z\rangle|y\rangle &\rightarrow |z\rangle U^{z_t 2^{k-1}}\ldots U^{z_1 2^0}|y\rangle \\
&= |z\rangle|x^{z_t 2^{k-1}}\ldots x^{z_1 2^0}y(\mod N)\rangle \\
&= |z\rangle|x^{z}y(\mod N)\rangle. \qquad (5.40)
\end{aligned}$$

which can be accomplished using the third register,

$$\begin{aligned}
|z\rangle|y\rangle|0\rangle &\rightarrow |z\rangle|y\rangle|x^{z}(\mod N)\rangle \\
&\rightarrow |z\rangle|yx^{z}(\mod N)\rangle|z\rangle|yx^{z}(\mod N)\rangle \\
&\rightarrow |z\rangle|yx^{z}(\mod N)\rangle|0\rangle. \qquad (5.41)
\end{aligned}$$

What is less trivial is how to prepare a vector $|u_s\rangle$ (without a prior knowledge of r) if not in an eigenstate, at least in a useful superposition. Note that we can express $|1\rangle$ in eigenbasis $|u_s\rangle$ as

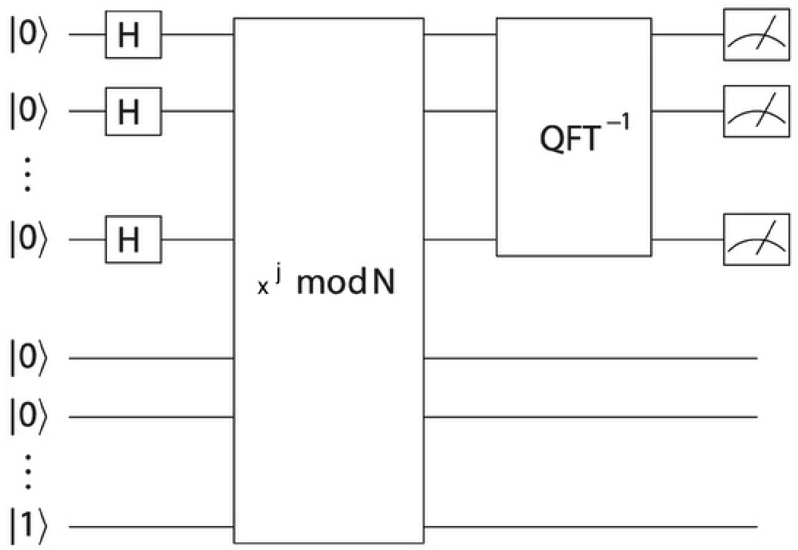
$$\begin{aligned} r^{-1/2}\sum_{s=0}^{r-1}|u_s\rangle &= r^{-1/2}\sum_{s=0}^{r-1} r^{-1/2}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^k\,(\mod N)\rangle \\ &= r^{-1}\sum_{s=0}^{r-1}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^k\,(\mod N)\rangle \\ &= \sum_{k=0}^{r-1}\delta_{0k}\,|x^k\,(\mod N)\rangle \\ &= |1\rangle. \end{aligned} \tag{5.42}$$

Therefore

$$\begin{aligned} U^j|1\rangle &= U\,r^{-1/2}\sum_{s=0}^{r-1}|u_s\rangle \\ |x^j(\mod N)\rangle &= r^{-1/2}\sum_{s=0}^{r-1}e^{2\pi isj/r}|u_s\rangle \end{aligned} \tag{5.43}$$

5.4 Shore algorithm

Now we can describe the Shor's algorithm to find r.



Start with a state (according to 5.42)

$$|0...0\rangle|0...01\rangle = |0\rangle^{\otimes t}r^{-1}\sum_{s=0}^{r-1}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\,|x^k\,(\mod N)\rangle \tag{5.44}$$

after Hadamard gates (according to 1.65) the state is

$$2^{-t/2}\sum_j |j\rangle r^{-1}\sum_{s=0}^{r-1}\sum_{k=0}^{r-1}\exp\left(\frac{-2\pi i\, sk}{r}\right)\ |x^k\ (\mod N)\rangle \tag{5.45}$$

after modular exponentiation is applied (according to 5.44) the state is

$$2^{-t/2}\sum_j |j\rangle r^{-1}\sum_{s=0}^{r-1}\sum_{k=0}^{r-1}\exp\left(-\frac{2\pi i\, sj}{r}-\frac{2\pi i\, sk}{r}\right)\ |x^k\ (\mod N)\rangle \tag{5.46}$$

after measuring second register the state is

$$r^{-1}\sum_{s=0}^{r-1}\left[2^{-t/2}\sum_j \exp\left(-\frac{2\pi i\, sj}{r}\right)|j\rangle\right]\exp\left(-\frac{2\pi i\, sk}{r}\right)|x^k\ (\mod N)\rangle \tag{5.47}$$

and after the inverse Quantum Fourier Transformation

$$r^{-1}\sum_{s=0}^{r-1}|\widetilde{s/r}\rangle\exp\left(-\frac{2\pi i\, sk}{r}\right)|x^k\ (\mod N)\rangle. \tag{5.48}$$

Then we use the so-called continued fraction expansion algorithm to find r, i.e.

$$a_0+\frac{1}{a_1+\frac{1}{a_2+\frac{1}{a_{3\ldots}}}} \tag{5.49}$$

For example if

$$\widetilde{s/r}=0.010110=\frac{32\times 0+16\times 1+8\times 0+4\times 1+2\times 1+1\times 0}{64}=\frac{22}{64} \tag{5.50}$$

then

$$\frac{22}{64}=\frac{1}{2+\frac{20}{22}}=\frac{1}{2+\frac{1}{1+\frac{1}{10}}}\approx\frac{1}{3} \tag{5.51}$$

and thus

$$r=3. \tag{5.52}$$

Another example

$$\widetilde{s/r}=0.010011=\frac{32\times 0+16\times 1+8\times 0+4\times 0+2\times 1+1\times 1}{64}=\frac{19}{64} \tag{5.53}$$

then

$$\frac{19}{64}=\frac{1}{3+\frac{7}{19}}=\frac{1}{3+\frac{1}{2+\frac{5}{7}}}=\frac{1}{3+\frac{1}{2+\frac{1}{1+\frac{2}{5}}}}=\frac{1}{3+\frac{1}{2+\frac{1}{1+\frac{1}{2+\frac{1}{2}}}}}\approx\frac{8}{27} \tag{5.54}$$

and

$$r = 27. \tag{5.55}$$

Since the value of s/r is only approximate, the algorithm might fail but it is easy to verify for a given r if

$$x^r = 1(\mod N). \tag{5.56}$$

The most important application of Shor's algorithm is for factoring of large numbers N (into prime factors). Here is how it works:

1. If N is even output the factor 2. Uses $O(1)$ steps.

2. Search though all possible $a \geq 1$ and $b \geq 2$ to determine if $N = a^b$. If so, output a (perhaps b times). Uses $O(n^3)$ steps.

3. Choose a random x in the range $1 < x < N$ and (using Euclid's algorithm) find the greatest common divisor of x and N (or $\gcd(x, N)$). If $\gcd(x, N) > 1$ then output it. Uses $O(n)$ steps.

4. Use Shor's algorithm to find the order r of x modulo N, i.e. smallest r such that
$$x^r = 1(\mod N).$$

5. If r is even and $x^{r/2} \neq N - 1(\mod N)$ (both of which happen with probability $O(1)$) then compute $\gcd(x^{r/2} - 1, N)$ and $\gcd(x^{r/2} + 1, N)$ to see if any of these give us a non-trivial factor of N,
$$x^r - 1 = \left(x^{r/2} - 1\right)\left(x^{r/2} + 1\right) = N = 0(\mod N)$$
Uses $O(n^3)$ steps.