The Quantum Fourier Transform (QFT) implements the analog of the classical Fourier Transform. It transforms a state space of size 2^n from the amplitude to the frequency domain (just as the Fourier transform can be viewed as a transform from 2^n numbers into a range of size 2^n containing the frequency components from the amplitude domain). This quantum algorithm is all about manipulating the phase of the qubits.

The classical Fourier Transform is defined as:

$$y_k \equiv \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} x_j e^{2\pi i jk/2^n}$$

The QFT is similarly defined:

$$\big|j\big\rangle \longrightarrow \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i jk/2^n} \big|k\big\rangle$$

Thus an arbitrary quantum state is transformed:

$$\sum_{j=0}^{2^n-1} x_j \big|j\big\rangle \longrightarrow \sum_{k=0}^{2^n-1} y_k \big|k\big\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \sum_{j=0}^{2^n-1} x_j e^{2\pi i jk/2^n} \big|k\big\rangle$$

So how do we implement the QFT? This derivation is in Nielsen and Chuang at pages 216-219, but is expanded in parts here for clarity.

We are going to work with the transform of a single quantum state, defined as:

$$\big|j\big\rangle \longrightarrow \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i jk/2^n} \big|k\big\rangle \qquad (1)$$

Note that j is a binary number ($j \leq 2^n$) and can be decomposed into the form:

$$j = j_1 2^{n-1} + j_2 2^{n-2} + \ldots + j_n 2^0 = \sum_{i=1}^{n} j_i 2^{n-i}$$

Similarly for k

$$k = \sum_{i=1}^{n} k_i 2^{n-i}$$

Use the k decomposition and leave j alone for now, to re-express the transform as

$$\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i j \sum_{l=1}^{n} k_l 2^{n-l}/2^n} \big|k\big\rangle$$

Canceling the 2^n terms we have:

$$\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i j \sum_{l=1}^{n} k_l 2^{-l}} \big|k\big\rangle$$

Now write the exponent out explicitly:

$$\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i jk_1 2^{-1}} \times e^{2\pi i jk_2 2^{-2}} \times \ldots \times e^{2\pi i jk_n 2^{-n}} \big|k\big\rangle$$

Now, decompose the summation over k as a sum over the two allowed binary values 0 and 1 of each bit k_i:

$$\frac{1}{\sqrt{2^n}}\sum_{k_1=0}^{1}\sum_{k_2=0}^{1}\dots\sum_{k_n=0}^{1} e^{2\pi i j k_1 2^{-1}} \times e^{2\pi i j k_2 2^{-2}} \times \dots \times e^{2\pi i j k_n 2^{-n}} \big|k_1 k_2 \dots k_n\big\rangle$$

Now, pull out the n'th component:

$$\frac{1}{\sqrt{2^n}}\sum_{k_1=0}^{1}\sum_{k_2=0}^{1}\dots\sum_{k_{n-1}=0}^{1} e^{2\pi i j k_1 2^{-1}} \times e^{2\pi i j k_2 2^{-2}} \times \dots \times e^{2\pi i j k_n 2^{-n}} \big|k_1 k_2 \dots k_{n-1}\big\rangle \sum_{k_n=0}^{1} e^{2\pi i j k_n 2^{-n}} \big|k\big\rangle$$

This last factor for the n'th component is equal to:

$$\Big(\big|0\big\rangle + e^{2\pi i j 2^{-n}}\big|1\big\rangle\Big)$$

where the first component comes from the $k_n = 0$ term and the second component from the $k_n = 1$ term. Repeating this for all k_i components leads to:

$$\frac{1}{\sqrt{2^n}}\Big(\big|0\big\rangle + e^{2\pi i j 2^{-1}}\big|1\big\rangle\Big)\Big(\big|0\big\rangle + e^{2\pi i j 2^{-2}}\big|1\big\rangle\Big)\dots\Big(\big|0\big\rangle + e^{2\pi i j 2^{-n}}\big|1\big\rangle\Big)$$

So now we have a tensor product of qubit states, each of which contains a different phase factor, $e^{2\pi i\left(\frac{j}{2^k}\right)}$, where $1 \le k \le n$. So if we can systematically generate these phase factors with quantum gates, we have a means of implementing the QFT.

We now put the phase factors in a form in which it is easy to see how to generate them.

First we define a new binary notation for a fraction - this is the analog of a decimal in base 10. For a number lying between 0 and 1, the binary fraction is simply the expansion in powers of $1/2$, which is written in the 'decimal' form as:

$$0.j_l j_{l+1} \dots j_m = \frac{j_l}{2} + \frac{j_{l+1}}{2^2} + \frac{j_m}{2^{m-l+1}}$$

where each $j_i = 0$ or 1.

Now since $k \le n$ and $0 \le j \le 2^n$, the quantity $\frac{j}{2^k}$ is not necessarily an integer. We can use our binary fraction notation to write it as a 'rational binary' number:

$$\begin{aligned}\frac{j}{2^k} &= \sum_{\nu}^{n} j_\nu 2^{n-\nu-k} \\ &= j_1 j_2 \dots j_{n-k} . j_{n-k+1} \dots j_n\end{aligned}$$

For example, if $n = 8$ and $k = 3$, we have

$$\begin{aligned}j &= j_1 2^7 + j_2 2^6 + j_3 2^5 + j_4 2^4 + j_5 2^3 + j_6 2^2 + j_7 2^1 + j_8 2^0 \\ \text{and } \frac{j}{2^3} &= j_1 2^4 + j_2 2^3 + j_3 2^2 + j_4 2^1 + j_5 2^0 + j_6 2^{-1} + j_7 2^{-2} + j_8 2^{-3}.\end{aligned}$$

From this it is clear that the last three terms are the binary fraction $0.j_6 j_7 j_8$, while the first five terms constitute an integer.

Now coming back to the phase factor $e^{2\pi i\left(\frac{j}{2^k}\right)}$, we now see that the integer part of $\frac{j}{2^k}$ will merely contribute a factor of 1 and that the phase is therefore entirely determined by the binary fraction:

$$e^{2\pi i\left(\frac{j}{2^k}\right)} = 1 \times e^{2\pi i 0.j_{n-k+1}\dots j_n}$$

We can now apply this to every term in the transform, to rewrite it as

$$\frac{1}{\sqrt{2^n}}\left(\big|0\big\rangle + e^{2\pi i 0.j_n}\big|1\big\rangle\right)\left(\big|0\big\rangle + e^{2\pi i 0.j_{n-1}j_n}\big|1\big\rangle\right)\dots\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2 \dots j_n}\big|1\big\rangle\right) \qquad (2)$$

To see how to actually implement this with quantum gates, lets look at any one of the qubits and break down the transformation to see how we can get to it from one of the starting qubits $\big|j_1\big\rangle\big|j_2\big\rangle\ldots\big|j_n\big\rangle$. Lets take the lth qubit, $\big|j_l\big\rangle$. According to eq. (2) above, this needs to be transformed from $\big|j_l\big\rangle$ into the state $\frac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi 0.j_l\ldots j_n}\big|1\big\rangle\right)$

(where we have relabelled the fractional indices for convenience).

1. We start by pulling off the first component of the phase on the second term:
$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_l}\times e^{2\pi 0.0 j_{l+1}\ldots j_n/2}\big|1\big\rangle\right)$$
2. This first component can be rewritten as:
$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_l}\big|1\big\rangle\right) = \tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i j_l/2}\big|1\big\rangle\right) = \tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + (-1)^{j_l}\big|1\big\rangle\right)$$
since $0.j_l = j_l/2$ and using $e^{i\pi j_l} = (-1)^{j_l}$ where $j_l = 0,1$. So the first term in the phase (most significant digit) can be implemented by operating with an H gate on $\big|j_l\big\rangle$.
3. What about the remaining components of the phase, $e^{2\pi 0.0 j_{l+1}\ldots j_n/2}$? For this we can use a sequence of rotations of the form
$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix},$$
each of which is controlled by the value of the j_k'th qubit, i.e., if j_k is equal to 1, we apply R_k, while if $j_k = 0$, we do nothing. Each such controlled rotation will multiply the phase of the second component $\big|1\big\rangle$ by $e^{2\pi i 0.0\ldots j_k}$. We implement this sequence of controlled rotations starting with the next most significant digit first, i.e., j_{l+1} in the above example. Note that the phases become very small as the qubit index increases.

Lets go through the entire procedure now. We will start with getting the transformed state of the last qubit in Eq. (2), i.e., we want to achieve

$$\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2\ldots j_n}\big|1\big\rangle\right).$$

Start with $\big|j_1\big\rangle\big|j_2\ldots j_n\big\rangle$.

Apply H on qubit 1 to obtain

$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_1}\big|1\big\rangle\right)\big|j_2\ldots j_n\big\rangle$$

Apply a controlled R_2 rotation on qubit 1, with qubit 2 the control, to obtain

$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2}\big|1\big\rangle\right)\big|j_2\ldots j_n\big\rangle$$

Apply controlled R_3 on qubit 1, with qubit 3 the control, to obtain

$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2 j_3}\big|1\big\rangle\right)\big|j_2\ldots j_n\big\rangle$$

Continue down to qubit n, to obtain

$$\tfrac{1}{\sqrt{2}}\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2\ldots j_n}\big|1\big\rangle\right)\big|j_2\ldots j_n\big\rangle$$

Notice that this procedure has put the desired transformed state into the first qubit instead of the last qubit as we originally intended (compare with Eq. (2)). No problem - we will swap the states of the transformed qubits at the end.

This entire procedure is now repeated for the other qubits, i.e., j_2, then j_3, etc. etc., resulting in the final transformed state:

$$\tfrac{1}{\sqrt{2^n}}\left(\big|0\big\rangle + e^{2\pi i 0.j_1 j_2\ldots j_n}\big|1\big\rangle\right)\left(\big|0\big\rangle + e^{2\pi i 0.j_2\ldots j_n}\big|1\big\rangle\right)\ldots\left(\big|0\big\rangle + e^{2\pi i 0.j_n}\big|1\big\rangle\right)$$

That's *almost* it: comparing this with Eq. (2), we see that the procedure has ended up with the states of the qubits in reverse order from that in the desired Fourier transform state. So we still have to swap the states of the qubits pairwise, starting from the ends and moving to the middle. This can be done using a SWAP circuit using 3 CNOT gates in alternating orientation (prove this!). Alternatively, we could just relabel the qubits. After this we finally have the QFT state of Eqs. (2) and (1):

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

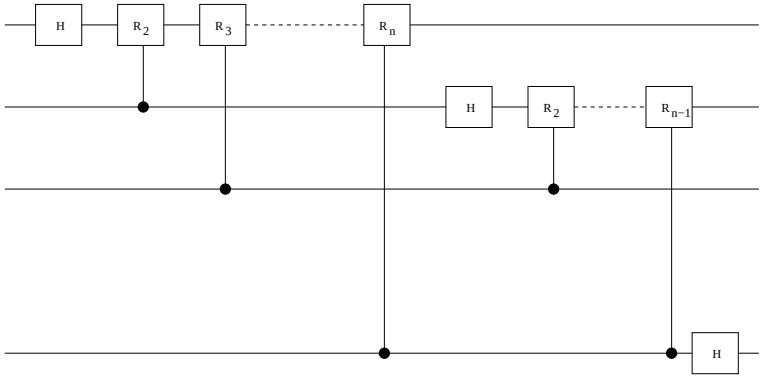

Figure 1: Quantum circuit implementing the Quantum Fourier Transform (QFT) on a quantum state input at the left. The first qubit is at the top, as usual. Note that the outputs are reversed in their bit-significance, i.e., qubit 1 contains the state of qubit n, etc. Following this circuit by a series of SWAP gates will then produce the final QFT state.

How many gates are required? Qubit 1 required H and $n-1$ controlled R gates, so a total of n gates. Qubit 2 required H and $n-2$ controlled R gates, so a total of $n-1$ gates. Continuing, we see that altogether $n+(n-1)+(n-2)\ldots+1 = n(n+1)/2$ gates are required, plus the final series of SWAP gates. These are of order $n/2$ (depending whether n is even or odd), so that the overall scaling of the QFT is $O(n^2)$. So we have polynomial scaling of the number of gates with the number of input qubits - an efficient quantum algorithm!

How does this compare with classical Fourier Transforms? Well, the simple Fourier transform shown at the very beginning of the lecture can be written as a matrix times a vector, where the matrix is of size $N = 2^n$. Thus the direct classical Fourier Transform scales as $O((2^n)^2)$, which is clearly exponential. In physics the scaling is often written as N^2 but don't let that fool you - remember to ask how many bits n there are! There exists a more efficient classical algorithm, the fast fourier transform or FFT, which improves on this to give a scaling $O(NlnN)$. Clearly this is still exponential in n. So the QFT provides a truly significant quantum speedup.

Note that the QFT is unitary (since we could construct a unitary circuit for it). The classical transform is also unitary (check this!).

Reading for QFT

Benenti et al., Ch. 3.11

Kaye et al., Ch. 7.1

Nielsen and Chuang, Quantum Computation and Quantum Information, Ch. 5.1