# Lecture 6

*In which we describe the quantum Fourier transform.*

## 1 The Discrete Fourier Transform

The discrete Fourier transform is a linear operator that happens to be unitary and, very fortunately, to be efficiently realizable as a quantum circuit. It is the main step in an efficient quantum algorithm that finds the period of a periodic function which, in turns, leads to efficient quantum algorithms for factoring and discrete log. A quantum Fourier transform (of a different type from the one described today) also plays a role in Grover's search algorithm. One could say that the Fourier transform is the only known unitary operator that is efficiently computable by a quantum circuit and that is useful to speed up algorithms for useful problems. It remains an open problem to find more such unitary operators.

We begin by defining the discrete Fourier transform. The Fourier transform of a function is just the collection of coefficients that are used to write the function as a linear combination of the elements of a certain nicely chosen basis.

First recall the following fact: if $V$ is an $n$-dimensional vector space with an inner product $\langle \cdot, \cdot \rangle$, and $v_1, \ldots, v_n$ are vectors such that $\langle v_i, v_j \rangle = 0$ for all $i \neq j$ and $\langle v_i, v_i \rangle = 1$ for all $i$, then $v_1, \ldots, v_n$ is called an *orthonormal basis* for $V$, and every vector $x \in V$ can be written as a linear combination

$$x = a_1 v_1 + \cdots a_n v_n$$

where the coefficient $a_i$ satisfy $a_i = \langle x, v_i \rangle$. The coefficients also satisfy *Parseval's identity*, which is essentially a version of the Pythagorean theorem):

$$||x||^2 = \sum_i |a_i|^2 \tag{1}$$

Let us now fix an integer $N \geq 1$, and consider the $N$-dimensional vector space of functions

$$f : \{0, \ldots, N-1\} \to \mathbb{C}$$

with the inner product

$$\langle f, g \rangle := \sum_x f(x)\overline{g(x)}$$

For every integer $s \in \{0, \ldots, N-1\}$, define the function

$$\chi_s(x) := \frac{1}{\sqrt{N}} e^{-2\pi i \cdot sx/N}$$

To simplify notation, in the following, we call $\omega := e^{2\pi i/N}$. Note that $\omega$ is a *primitive $N$-th root of* 1, that is, $\omega^N = 1$ and $\omega^k \neq 1$ for all $1 \leq k \leq N-1$.

It is easy to see that the functions $\chi_s$ form an orthonormal basis for the set of functions $f : \{0, \ldots, N-1\} \to \mathbb{C}$. Indeed, we have

$$\langle \chi_s, \chi_t \rangle = \sum_x \frac{1}{\sqrt{N}} \omega^{-sx} \cdot \frac{1}{\sqrt{N}} \omega^{tx} = \frac{1}{N} \sum_x \omega^{(t-s)\cdot x}$$

So when $s = t$ we have $\langle \chi_s, \chi_s \rangle = 1$. When $s \neq t$, let us call $d := t - s$, then

$$\cdot \langle \chi_s, \chi_t \rangle = \frac{1}{N} \sum_x \omega^{dx}$$

Let us consider the sum $S := \sum_x \omega^{dx}$. We have

$$S = 1 + \omega^d + \omega^{2d} + \cdots + \omega^{(N-1)d}$$

and if we multiply the sum by $\omega^d$ we have

$$\omega^d \cdot S = \omega + \omega^{2d} + \cdots + \omega^{(N-1)d} + 1$$

where we use $\omega^N = 1$. So the two sums are clearly the same and so we have

$$(1 - \omega^d) \cdot S = 0$$

which implies $S = 0$ because $d$ is not a multiple of $N$ and so $\omega^d$ cannot equal 1. In conclusion, we proved that if $s \neq t$ then

$$\langle \chi_s, \chi_t \rangle = 0$$

and so the functions $\chi_s$ form an orthonormal basis.

This means that every function $f : \{0, \ldots, N-1\} \to \mathbb{C}$ can be written as a linear combination

$$f(x) = \sum_s \hat{f}(s)\chi_s(x)$$

where the coefficients $\hat{f}(s)$ can be computed as

$$\hat{f}(s) = \langle f, \chi_s \rangle = \frac{1}{\sqrt{N}} \sum_x f(x)\omega^{sx}$$

The collection of coefficients $\hat{f}(s)$ can itself be thought of as a function

$$\hat{f} : \{0, \ldots, N-1\} \to \mathbb{C}$$

such a function $\hat{f}$ is called the *discrete Fourier transform* of $f$, and the values $\hat{f}(s)$ are called the *Fourier coefficients* of $f$.

In the classical setting, there is an algorithm that, given the values of $f$, computes the values of $\hat{f}$ in time $O(N \log N)$. This algorithm, called the *Fast Fourier Transform* algorithm,[1] is one of the most frequently executed non-trivial algorithm. Every device capable of handling digital audio, such as the one that is probably in your pocket right now, contains an implementation of the FFT algorithm, and it executes it repeatedly when it plays audio or during a phone call.

In the quantum setting, we will show that, in a certain sense, the discrete Fourier transform can be computed in time $O((\log N)^2)$, a fact that will lead to exponential speed-ups for certain algorithmic applications.

## 2 The Quantum Fourier Transform Algorithm

Note, that because of Parseval's identity (1), for every function $f : \{0, \ldots, N-1\} \to \mathbb{C}$ we have

$$\sum_x |f(x)|^2 = \sum_s |\hat{f}(s)|^2$$

this means that if $q = \sum_x f(x)|x\rangle$ is a quantum state over the state space $\Omega := \{0, \ldots, N-1\}$, then $q' := \sum_s \hat{f}(s)|s\rangle$ is also a quantum state, and the transformation

[1] It is one of the two famous algorithms that boast about their efficiency in their name.

that maps $q$ to $q'$ is a unitary operator, which we call $U_{DFT}$. As we will see, this unitary operator can be computed by a quantum circuit of size $O((\log N)^2)$ when $N = 2^n$ is a power of two.

Let us fix $N = 2^n$, and call $\omega = e^{2\pi i/N}$ a primitive $N$-th root of unity. If we can construct a quantum circuit that, on input a classical state $|z\rangle$ outputs $U_{DFT}|z\rangle$, then, by linearity, the circuit will compute $U_{DFT}$ given in input any quantum state.

For $z \in \{0, \ldots, N\}$, the state $|z\rangle$ corresponds to $\sum_x f(x)|x\rangle$ for the function $f()$ that is zero everywhere except for $f(z) = 1$. Computing the Fourier coefficients of such a function gives us

$$U_{DFT}|z\rangle = \frac{1}{\sqrt{N}} \sum_s \omega^{sz}|s\rangle \tag{2}$$

Since we want to design a qubit-based quantum circuit to compute $U_{DFT}$, we represent an integer $z$ as the bit-string $(z_1, \ldots, z_n)$ that is the binary representation of $z$, meaning that $z = 2^{n-1}z_1 + \cdots + 2z_2 + z_1$. We can rewrite $\omega^{sz}$ as

$$\begin{aligned}
\omega^{sz} &= \omega^{(\sum_{i=1}^n 2^{n-i} s_i)\cdot(\sum_{j=1}^n 2^{n-j} z_j)} \\
&= \prod_{i=1}^n \omega^{s_i \cdot \sum_{j=1}^n 2^{2n-i-j} z_j} \\
&= \prod_{i=1}^n \omega^{s_i \cdot \sum_{j=n-i+1}^n 2^{2n-i-j} z_j}
\end{aligned}$$

where the last identity follows from the fact that $\omega^{2^n} = 1$. The last expression show that the amplitude of $|s\rangle$ in the right-hand side of (2) can be written as a product of $n$ terms, each depending on only one of the bits of $s$. This means that the state $U_{DFT}|x\rangle$ can be written as a tensor product

$$U_{DFT} = \bigotimes_{i=1}^n \frac{1}{\sqrt{2}} \left( |0\rangle + \omega^{\sum_{j=n-i+1}^n 2^{2n-i-j} z_j} |1\rangle \right)$$

and the $i$-th bit of the output can be computed using one Hadamard gate and $i$ phase-modifying gates, each operating on 2 qubits and hence implementable with a constant number of CNOT and 1-qubit gates. The circuit, of size $O(n^2)$, is shown in the textbook at page 219.