AQAlg: Advanced Quantum Algorithms **2025 - 2026**

Lecture 1: QFT, phase estimation and Shor's algorithm

Lecturer: Simon Apers (apers@irif.fr)

1 Quantum Fourier transform

One of the key building blocks used in quantum algorithms is the quantum Fourier transform. First, we recall the classical (discrete) Fourier transform. For $N \in \mathbb{N}$, let $\omega_N = e^{2\pi i/N}$. The Fourier transform $F_N : \mathbb{C}^N \mapsto \mathbb{C}^N$ is defined by

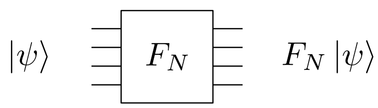
$$F_N = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & \omega_N & \dots & \omega_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{N-1} & \dots & \omega_N^{(N-1)(N-1)} \end{bmatrix}.$$

More concisely, $(F_N)_{j,k} = \omega_N^{jk}/\sqrt{N}$ for $j, k \in \{0, \dots, N-1\}$. The rows or columns of F_N are the Fourier modes

$$|\tilde{k}\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} |j\rangle, \quad k \in \{0, \dots, N-1\}. \tag{1}$$

Since these form an orthonormal basis, the Fourier transform F_N is a unitary operation.

It follows that we can think of the Fourier transform as a quantum operation. Assuming that $N = 2^n$, the operation F_N acts on an n qubit state:



If $|\psi\rangle = \sum_{k=0}^{N-1} \alpha_k |k\rangle$ then this returns the state

$$F_N |\psi\rangle = \sum_{j=0}^{N-1} \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{jk} \alpha_k \right) |j\rangle .$$

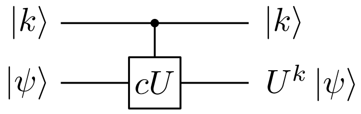
As we will see later, this is an incredibly useful quantum operation. Moreover, while the classical Fourier transform takes time poly(N), we can implement the quantum Fourier transform in time only poly(n)!

Lemma 1. *Let $N = 2^n$. We can implement the quantum Fourier transform F_N with $O(n^2)$ 2-qubit gates.*

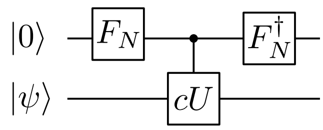
2 Quantum phase estimation

A first important application of the quantum Fourier transform is *quantum phase estimation.* Assume access to a unitary U and eigenvector $|\psi\rangle$ such that $U|\psi\rangle = e^{2\pi i\theta} |\psi\rangle$ for some $\theta \in [0, 1)$. We can use the QFT to estimate the phase θ. The intuition behind this is that repeatedly applying U to $|\psi\rangle$ yields a "signal" $e^{i\theta t} |\psi\rangle$ that rotates with angular velocity θ.

For some $N = 2^n$, we assume that θ is such that $N\theta$ is an integer. Consider the controlled version of U, represented by the following circuit:



where $k \in \{0, 1, \ldots, N-1\}$. The circuit for quantum phase estimation is the following:



We can track the evolution:

$$\begin{aligned} |0^n\rangle\,|\psi\rangle \quad &\overset{F_N}{\mapsto} \quad \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle\,|\psi\rangle \\ &\overset{cU}{\mapsto} \quad \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle\, U^j\,|\psi\rangle = \left(\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i\theta j}\,|j\rangle \right) |\psi\rangle\,. \end{aligned}$$

Rewriting $e^{2\pi i\theta j} = \omega_N^{(N\theta)j}$, we see that the first register now corresponds to a simple Fourier mode $|\tilde{k}\rangle$ with $k = N\theta$ (see Eq. (1)). Applying the inverse Fourier transform yields the final state

$$\left(\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{2\pi i(\theta N)j}\,|j\rangle \right) |\psi\rangle \overset{F_N^\dagger}{\mapsto} |N\theta\rangle\,|\psi\rangle\,,$$

from which we can read off the phase θ.

The complexity of phase estimation is typically dominated by the maximum number of times we have to implement the unitary U, which is $N-1$ times. If the phase $\theta \in [0, 1)$ does not have an exact n-bit expansion, then quantum phase estimation returns with high probability an n-bit approximation to θ. In particular, we have the following lemma.

Lemma 2. *Consider a unitary U and eigenvector $|\psi\rangle$ such that $U\,|\psi\rangle = e^{2\pi i\theta}\,|\psi\rangle$ with $\theta \in [0, 1)$. Using quantum phase estimation, it is possible to obtain an additive ϵ-approximation to θ by making $O(1/\epsilon)$ calls to U.*

3 Shor's algorithm

We now move on to one of the crown jewels of quantum computing, which is Shor's quantum algorithm for factoring integers. Consider an n-bit integer N such that $2^{n-1} \leq N < 2^n$. Classically it is possible to *check* whether N is prime in time poly(n). However, if we wish to actually find a nontrivial factor of N, then the best classical algorithm takes time exponential in some power of n. Shor's algorithm is a quantum algorithm that factorizes a composite number in time poly(n) on a quantum computer.

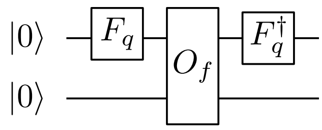
An important yet non-quantum component of Shor's algorithm is a reduction from factoring to the following problem:

> *Given access to a function $f : \mathbb{N} \to \{0, \ldots, N-1\}$ for which there exists $r \in \{0, \ldots, N-1\}$ such that $f(a) = f(b)$ iff $a = b \pmod{r}$, find r.*

In the following we describe a relatively simple quantum algorithm that solves this problem in time poly(n).

3.1 Quantum algorithm for period finding

Let $q = 2^\ell$ be such that $N^2 < q \leq 2N^2$, and define the oracle $O_f \left|a\right\rangle \left|0\right\rangle = \left|a\right\rangle \left|f(a)\right\rangle$ for $a \in \{0, 1, \ldots, q-1\}$ to access f. The gist of the algorithm is described by the following simple circuit:



We can again track the evolution:

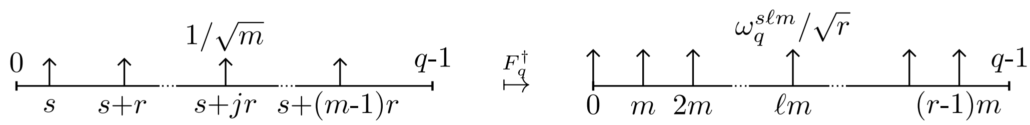
$$\begin{aligned} \left|0\right\rangle \left|0\right\rangle &\overset{F_q}{\mapsto} \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} \left|a\right\rangle \left|0\right\rangle \\ &\overset{O_f}{\mapsto} \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} \left|a\right\rangle \left|f(a)\right\rangle \end{aligned}$$

Now, for simplicity, assume that r divides q (i.e., $m = q/r$ is integer). Then, by the periodicity assumption on f, we can rewrite this as

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} \left|a\right\rangle \left|f(a)\right\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left(\frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} \left|s + jr\right\rangle \right) \left|f(s)\right\rangle .$$

Now notice that the first register contains a superposition of r-periodic "signals" of the form $\frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} \left|s + jr\right\rangle$. By standard Fourier analysis (see exercises), we see that

$$\frac{1}{\sqrt{m}} \sum_{j=0}^{m-1} \left|s + jr\right\rangle \quad \overset{F_q^\dagger}{\mapsto} \quad \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} \omega_q^{-s\ell m} \left|\ell m\right\rangle ,$$



We can hence summarize the full circuit by the mapping

$$\left|0\right\rangle \left|0\right\rangle \quad \overset{F_q^\dagger O_f F_q}{\mapsto} \quad \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left(\frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} \omega_q^{-s\ell m} \left|\ell m\right\rangle \right) \left|f(s)\right\rangle .$$

If we measure the first register of this state, we retrieve an integer $b = cm$ for uniformly random $c \in \{0, 1, \ldots, r-1\}$. Now recall that $m = q/r$ and so $b/q = c/r$, where we know both b and q. If c is coprime to r (which happens with good probability), then c and r are obtained by reducing b/q to lowest terms.

The overall complexity of the algorithm is poly(n). If we omit our simplifying assumption (r divides q) then the integer b will only be approximately equal to cm, yet we can still recover r from the so-called "continued-fraction expansion" of b.