CPEN 400Q Lecture 12
The quantum Fourier transform (QFT)

Monday 24 February 2025

Announcements

fourier

- Quiz 5 today
- Literacy Assignment 2 due tomorrow at 23:59
- Assignment 2 due Thursday at 23:59
- Tutorial tomorrow: intro to variational algorithms
  - helpful for many project groups
- First project peer assessment survey this week
  - Qualtrics link will be posted in Piazza

☆ email me if you optimized your circuit more

Where have we been?

Basics

Gates Meas. States
Entanglement

Superdense coding
Teleportation

Deutsch's algorithm

Grover's algorithm

oracle-based

square root speedup

$O(2^n) \to O(\sqrt{2^n})$

Where are we going?

Shor's algorithm (ca. 1996-1997)

```
def shors_algorithm(N):
    p, q = 0, 0

    while p * q != N:
        a = np.random.choice(list(range(2, N - 1)))

        if np.gcd(a, N) != 1:
            p = np.gcd(a, N)
            q = N // p
            return p, q

        sample = get_sample(a, N)
        phase = fractional_binary_to_float(sample)
        candidate_order = phase_to_order(phase, N)

        if candidate_order % 2 == 0:
            square_root = (a ** (candidate_order // 2)) % N

            if square_root not in [1, N - 1]:
                p = np.gcd(square_root - 1, N)
                q = np.gcd(square_root + 1, N)

    return p, q
```

next wed.

order finding (hard)

next Monday

quantum phase estimation

Wednesday

quantum Fourier transform

Today

Module 3 learning outcomes

Learning outcomes:

- define, and state the scaling of, the quantum Fourier transform
- use quantum phase estimation to determine the eigenvalues of a unitary matrix
- use the QFT and QPE as subroutines to implement order finding, and simulate Shor's factoring algorithm
- identify cryptographic schemes susceptible to quantum attack
- describe the societal and ethical implications of quantum technology

Today

Learning outcomes:

- Express floating-point values in fractional binary representation
- Describe the behaviour of the quantum Fourier transform
- Construct a circuit for the quantum Fourier transform and analyze its resource usage

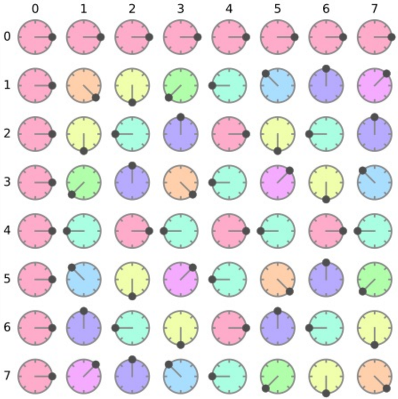
The DFT and FFT (which implements it efficiently) convert between time and frequency domains in digital signal processing.

$$DFT = \underbrace{\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \bar{\omega} & \bar{\omega}^2 & \cdots & \bar{\omega}^{N-1} \\ 1 & \bar{\omega}^2 & \bar{\omega}^4 & \cdots & \bar{\omega}^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \bar{\omega}^{N-1} & \bar{\omega}^{2(N-1)} & \cdots & \bar{\omega}^{(N-1)(N-1)} \end{pmatrix}}_{N} \quad N$$

where $\bar{\omega} = e^{-2\pi i/N}$.

The discrete Fourier transform

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \bar{\omega} & \bar{\omega}^2 & \cdots & \bar{\omega}^{N-1} \\ 1 & \bar{\omega}^2 & \bar{\omega}^4 & \cdots & \bar{\omega}^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \bar{\omega}^{N-1} & \bar{\omega}^{2(N-1)} & \cdots & \bar{\omega}^{(N-1)(N-1)} \end{pmatrix}$$



Given a signal $x[n]$, the DFT computes

$$X[k] = \sum_{n=0}^{N-1} e^{-\frac{2\pi i k n}{N}} x[n] = \sum_{n=0}^{N-1} \bar{\omega}^{nk} x[n]$$

The inverse DFT computes

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} e^{\frac{2\pi i k n}{N}} X[k] = \frac{1}{N} \sum_{k=0}^{N-1} \omega^{nk} X[k]$$

where $\omega = e^{2\pi i/N} = \bar{\omega}^{-1}$

The quantum Fourier transform (QFT) is the quantum analog of the **inverse DFT**.

$$QFT = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega^{jk} |k\rangle\langle j| \quad , N = 2^n$$

integers

Exercise: Apply the QFT to an n-qubit basis state $|x\rangle$

$$QFT|x\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega^{jk} |k\rangle\langle j|x\rangle$$

$$= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{xk} |k\rangle$$

$\langle j|x\rangle$

$j = x$

As a matrix, it looks a lot like the DFT:

$$QFT = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{pmatrix}$$

$$\omega = e^{\frac{2\pi i}{N}}$$

How do we *synthesize* a circuit for it?

Exercise: Start with special case $n = 1$ ($N = 2$).

$$QFT = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad \omega = e^{\frac{2\pi i}{2}} = e^{\pi i} = -1$$

$\Rightarrow$

Next case: $n = 2$ ($N = 4$)

$$QFT = \frac{1}{\sqrt{4}}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{pmatrix}$$

$$= \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}$$

each H gives $\frac{1}{\sqrt{2}}$

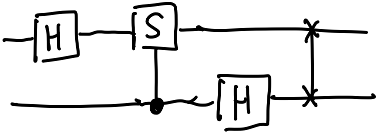
$$\omega = e^{\frac{2\pi i}{4}} = e^{\frac{\pi i}{2}} = i$$

$$\omega^2 = -1 \quad \omega^3 = -i$$

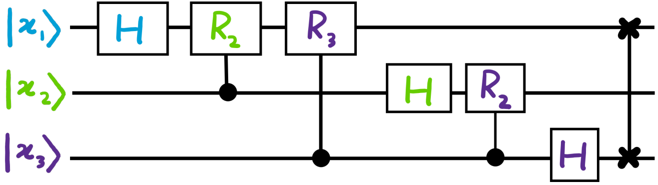
$\Rightarrow$ # Hadamards: 2

$i/-i$ = S gates

+ entangling gate

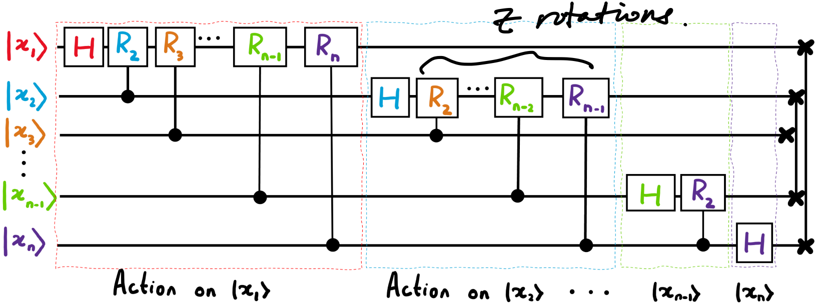


Circuit for $n = 3$ ($N = 8$):



Here, $R_2 = S$ and $R_3 = T$.

Image credit: Xanadu Quantum Codebook node F.3



We will derive this by reverse-engineering the analytical definition,

$$|x\rangle \longmapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{xk} |k\rangle$$

Image credit: Xanadu Quantum Codebook node F.3

$$|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{xk} |k\rangle \qquad \omega = e^{\frac{2\pi i}{N}}$$

Here x and k are integers, which have binary equivalents $|x\rangle = |x_1 \cdots x_n\rangle$, $|k\rangle = |k_1 \cdots k_n\rangle$:

$$x = 2^{n-1} x_1 + 2^{n-2} x_2 + \cdots + 2x_{n-1} + x_n$$

and similarly for k.

We are working with

$$\omega^{xk} = e^{2\pi i x (k/N)}$$

with $N = 2^n$.

We can write a fraction $k/2^n$ in a 'decimal version' of binary:

$$\frac{k}{2^n} = 0.k_1k_2k_3\cdots k_n$$
$$= \frac{k_1}{2} + \frac{k_2}{2^2} + \frac{k_3}{2^3} + \cdots + \frac{k_n}{2^n}$$
$$= 2^{-1}k_1 + 2^{-2}k_2 + \cdots + 2^{-n}k_n$$

Exercise: let $k = 0.11010$. What is the numerical value of k?

$$k = 0.8125$$

$$0.11010 = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 1 + \frac{1}{8} \cdot 0 + \frac{1}{16} \cdot 1 + \frac{1}{32} \cdot 0$$

$$= 0.5 + 0.25 + 0.0625$$

$$= 0.8125$$

We will reexpress k/N in fractional binary notation, then reshuffle and *factor* the output state to uncover the circuit structure.

$$|x\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{xk} |k\rangle$$

k is int: $k_1 k_2 \cdots k_n$

$$= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{xk} |k_1 k_2 \cdots k_n\rangle \qquad \frac{k}{N}$$

$$= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x \cdot \frac{k}{N}} |k_1 k_2 \cdots k_n\rangle$$

$$k = 2^{n-1} \cdot k_1 + 2^{n-2} k_2 + \cdots + 2 \cdot k_{n-1} + k_n$$

$$\frac{k}{N} = \frac{2^{n-1} \cdot k_1 + 2^{n-2} k_2 + \cdots + 2 \cdot k_{n-1} + k_n}{2^n}$$

$$= \frac{2^{n-1}}{2^n} \cdot k_1 + \frac{2^{n-2}}{2^n} k_2 + \cdots + \frac{k_n}{2^n}$$

$$= 2^{-1} \cdot k_1 + 2^{-2} k_2 + \cdots + 2^{-n} \cdot k_n$$

$$= 0.k_1 k_2 \cdots k_n$$

A circuit for the QFT

We will reexpress k/N in fractional binary notation, then reshuffle and *factor* the output state to uncover the circuit structure.

$$\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x \cdot \frac{k}{N}} |k_1 k_2 \cdots k_n\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x \cdot \left(\frac{k_1}{2} + \frac{k_2}{2^2} + \cdots + \frac{k_n}{2^n}\right)} |k_1 \cdots k_n\rangle$$

☆ We will restart this on Wednesday.
Sorry for the mess.

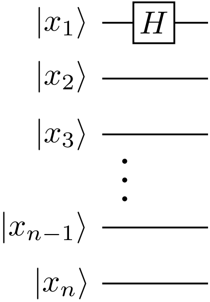
A circuit for the QFT

A circuit for the QFT

Exercise: Starting from

$$|x\rangle = |x_1 \cdots x_n\rangle,$$

apply Hadamard to qubit 1, then express the phase in terms of x_1 using fractional binary notation.



A circuit for the QFT

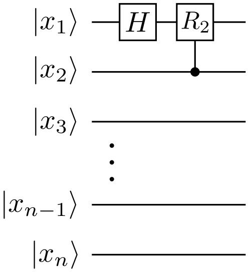
Recall: trying to make the state

$$|x\rangle \to \frac{\left(|0\rangle + e^{2\pi i 0.x_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.x_{n-1}x_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.x_1\cdots x_n}|1\rangle\right)}{\sqrt{N}}$$

Every qubit has a different *relative phase*. Define

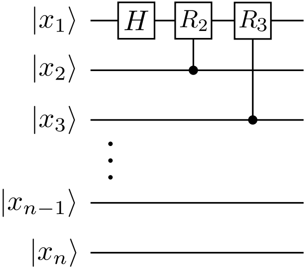
A circuit for the QFT

Apply controlled R_2 from $2 \to 1$



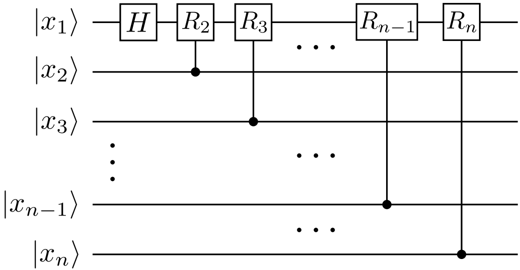
First qubit picks up a phase:

Apply controlled R_3 from $3 \to 1$

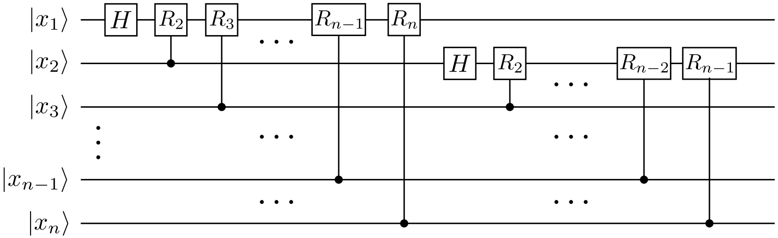


First qubit picks up another phase:

Apply a controlled R_4 from $4 \to 1$, etc. to get

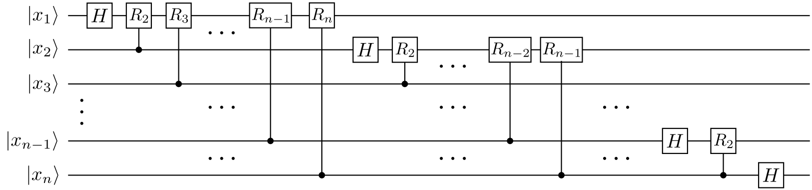


Repeat with the second qubit: apply *H* then controlled rotations from qubits 3 to *n* to get



Repeat for remaining qubits to obtain the big state from earlier:

$$|x\rangle \to \frac{\left(|0\rangle + e^{2\pi i 0.x_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.x_{n-1}x_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.x_1\cdots x_n}|1\rangle\right)}{\sqrt{N}}$$



The qubits are “backwards” - easily fixed with SWAP gates.

Exercise: What are the gate counts and depth of this circuit?

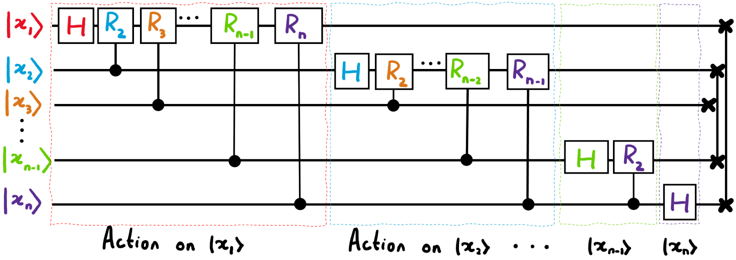


Image credit: Xanadu Quantum Codebook node F.3

Next time

Content:

- Variational algorithms (tutorial)
- Quantum phase estimation

Action items:

1. LA2 and A2
2. Work on project

Recommended reading:

- For this class: Codebook module QFT, Nielsen & Chuang 5.1
- For next class: Codebook module QPE, Nielsen & Chuang 5.2