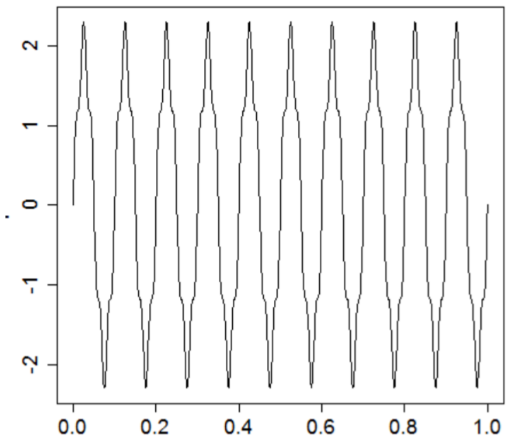
The Quantum Fourier Transform and Its Applications

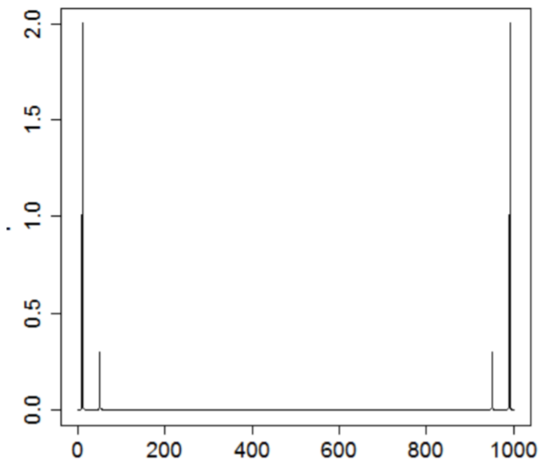
Claire Yang

2019-03-19

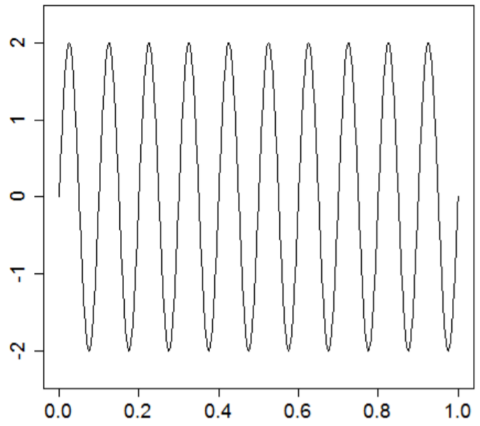
Discrete Fourier Transform



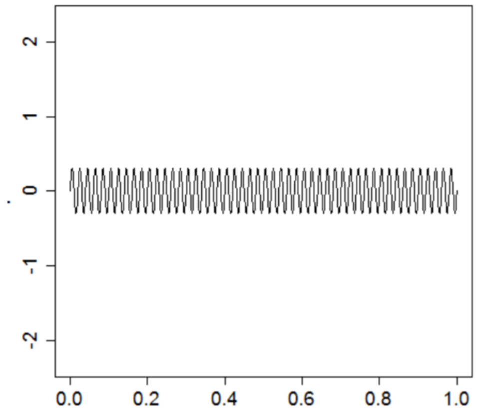
Input signal



Input signal in frequency domain



True signal: f=10, A=2



Noise signal: f=50, A=0.3

QFT is the quantum analogue of DFT:

— Map signal from **time** domain to **frequency** domain on quantum computer with a faster speed

— Basis for **quantum phase estimation**

Pictures: https://arxiv.org/pdf/1804.10068.pdf

Quantum Fourier Transform

Input

Output

$$\underbrace{x_1, x_2, \dots, x_N}_{\text{Length: N}} \Rightarrow \text{DFT: } y_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi ijk/N} \Rightarrow \underbrace{y_1, y_2, \dots, y_N}_{\text{Length: N}}$$

$$\underbrace{|x_1, x_2, \dots, x_n\rangle}_{\text{Length: n}} \Rightarrow \text{QFT: } |j\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle \Rightarrow \underbrace{|y_1, y_2, \dots, y_n\rangle}_{\text{Length: n}}$$

$$N = 2^n$$

Quantum Fourier Transform

QFT: $|j\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} |k\rangle$

$$\underbrace{|x_1, x_2, \dots, x_n\rangle}_{\text{Length: n}} \Rightarrow \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \omega_n^3 & \cdots & \omega_n^{N-1} \\ 1 & \omega_n^2 & \omega_n^4 & \omega_n^6 & \cdots & \omega_n^{2(N-1)} \\ 1 & \omega_n^3 & \omega_n^6 & \omega_n^9 & \cdots & \omega_n^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_n^{N-1} & \omega_n^{2(N-1)} & \omega_n^{3(N-1)} & \cdots & \omega_n^{(N-1)(N-1)} \end{bmatrix} \Rightarrow \underbrace{|y_1, y_2, \dots, y_n\rangle}_{\text{Length: n}}$$

N×N matrix

where $\omega_n := e^{\frac{2\pi i}{2^n}}$

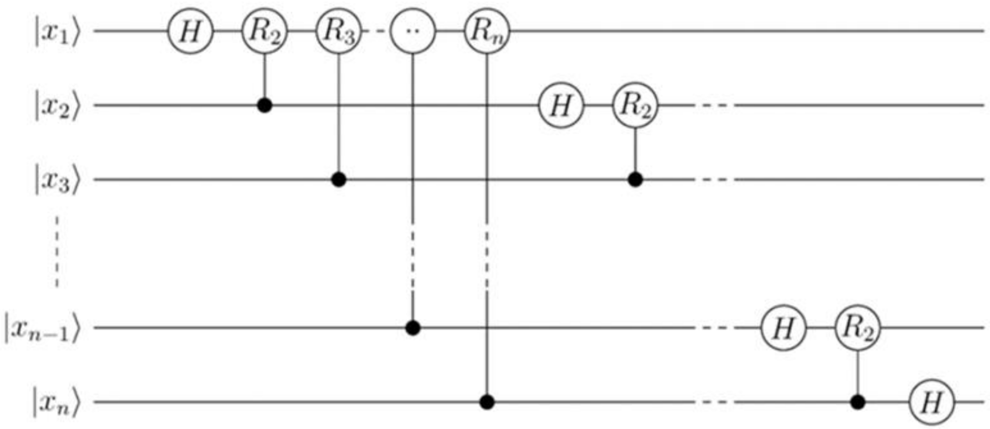
Example (2 qubits):

$$F_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}$$

Quantum Fourier Transform

QFT: $|j\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle$

$$\underbrace{|x_1, x_2, \dots, x_n\rangle}_{\text{Length: n}} \Rightarrow$$



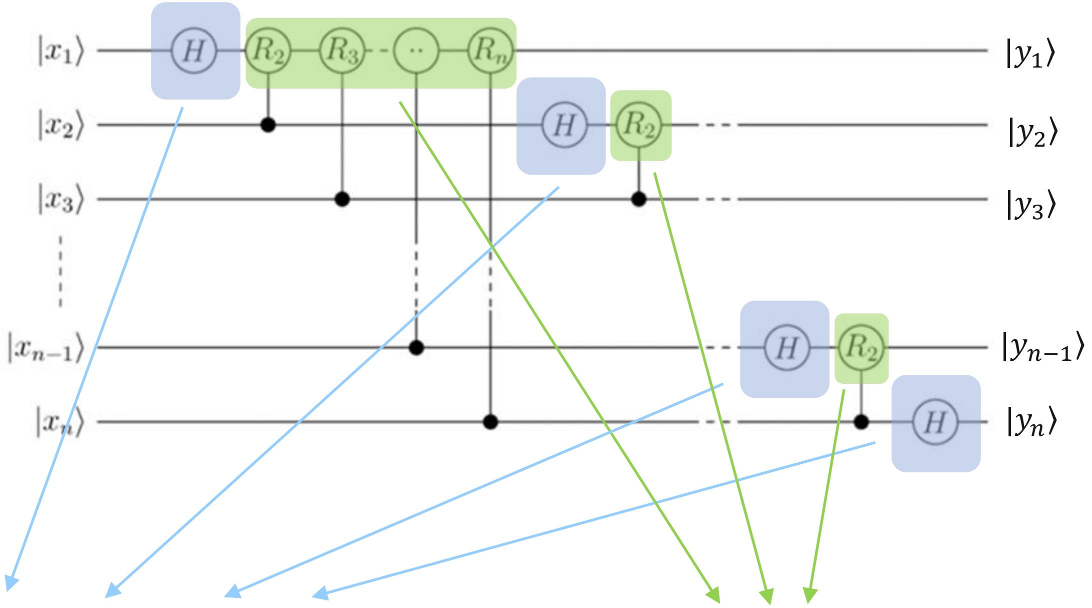
$$\Rightarrow \underbrace{|y_1, y_2, \dots, y_n\rangle}_{\text{Length: n}}$$

Quantum circuit for QFT

Picture: https://www.wikiwand.com/en/Quantum_Fourier_transform

Quantum Fourier Transform

Quantum circuit for QFT



Hadamard gate: create superposition

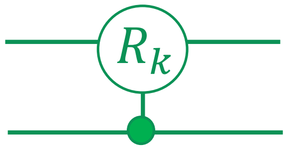
$$|0\rangle \rightarrow \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$|1\rangle \rightarrow \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Controlled phase gate

Target qubit

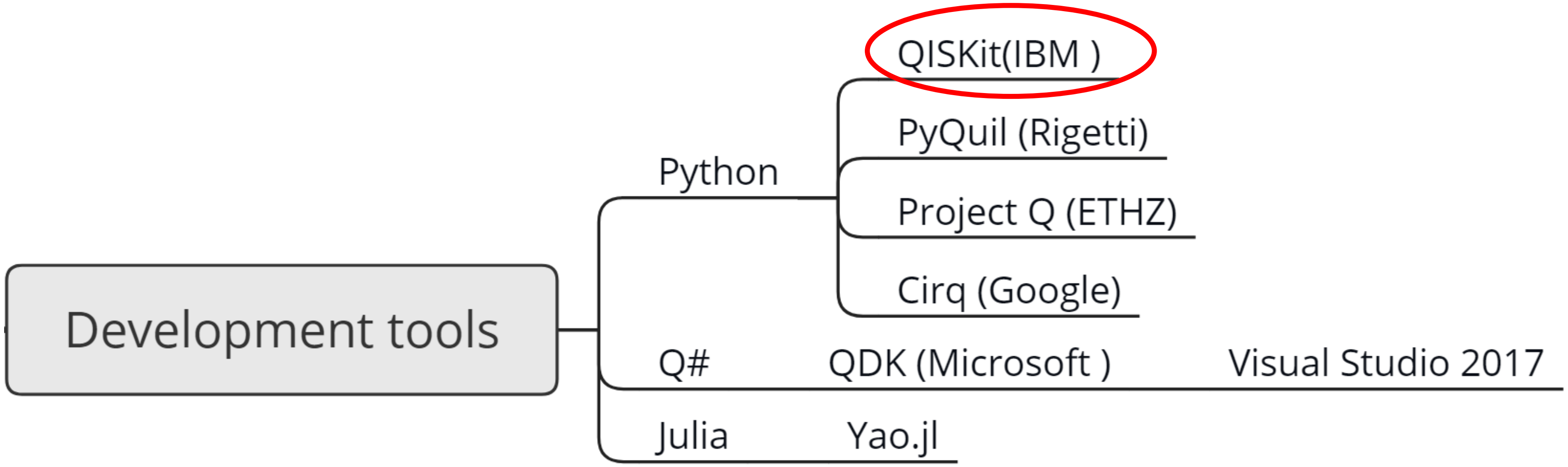
Control qubit



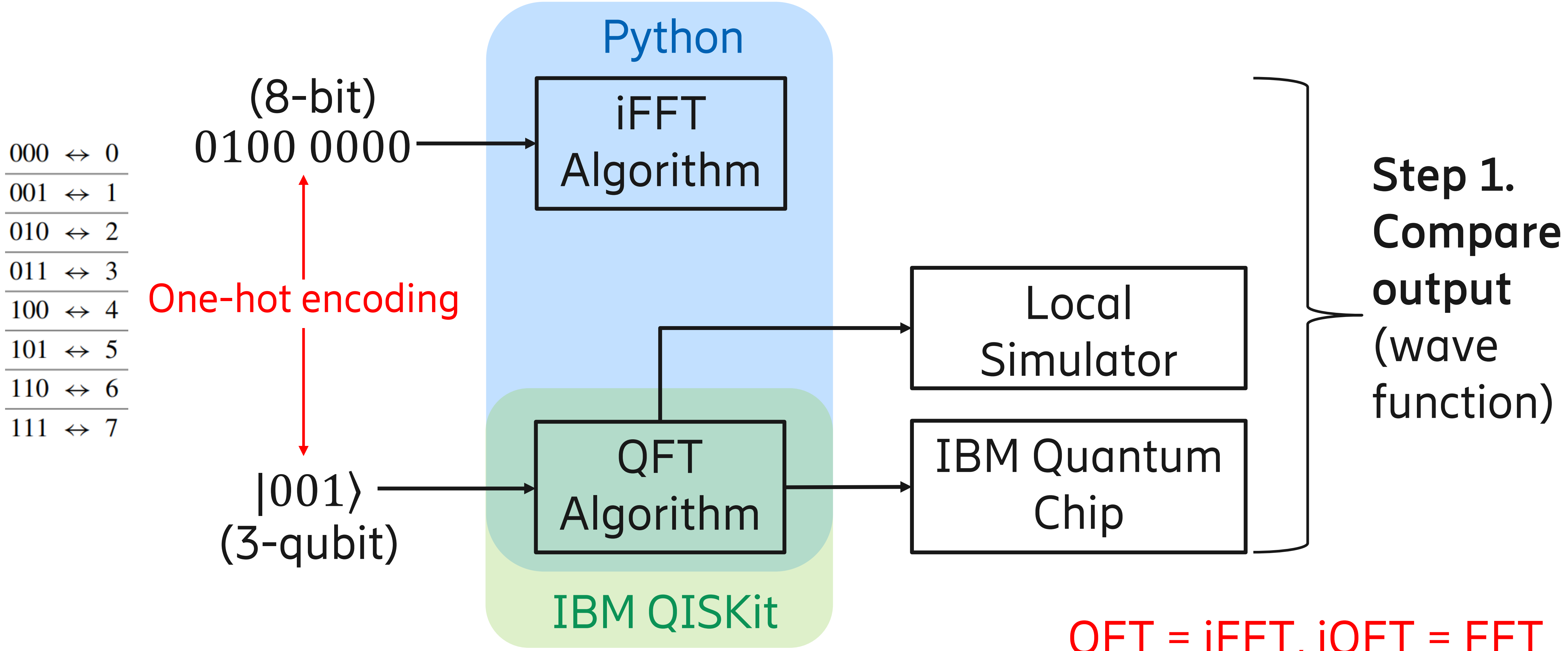
$$R_k \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{k}} \end{bmatrix}$$

Picture: https://www.wikiwand.com/en/Quantum_Fourier_transform

Development tools

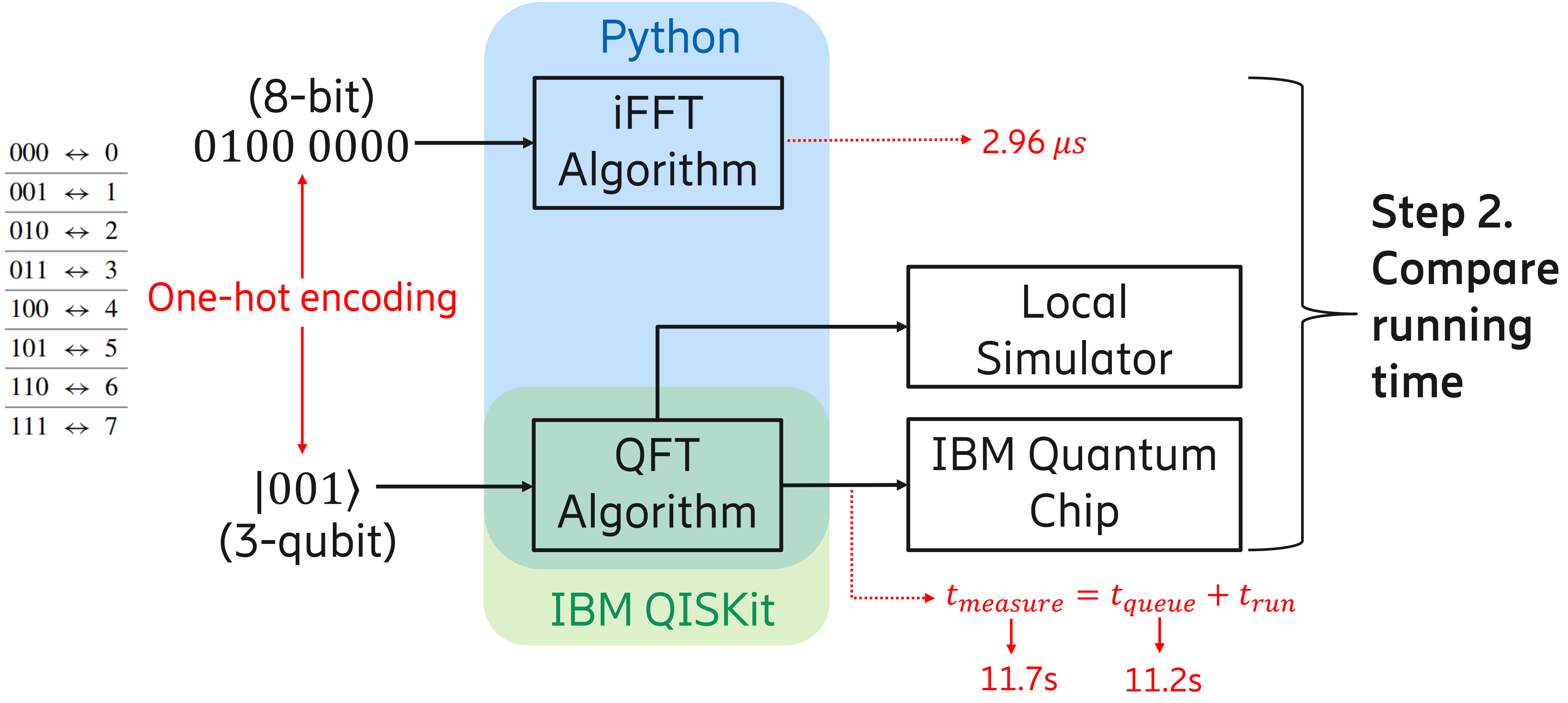


Implementation and comparison with iFFT



QFT = iFFT, iQFT = FFT

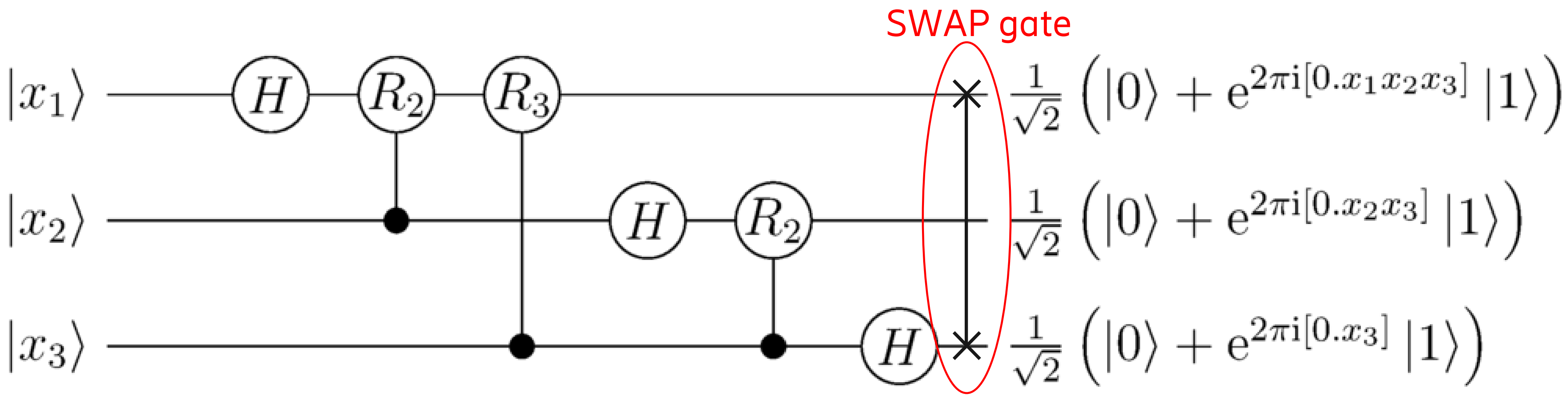
Implementation and comparison with iFFT



Estimation of the QFT running time

— Add up the delays on the longest path of the compiled circuit

— Single qubit gate: 80 ns, two qubits gate: 170<t<348 ns



— e.g. For 3-qubit circuit, the estimated running time is among: (920, 1632)ns

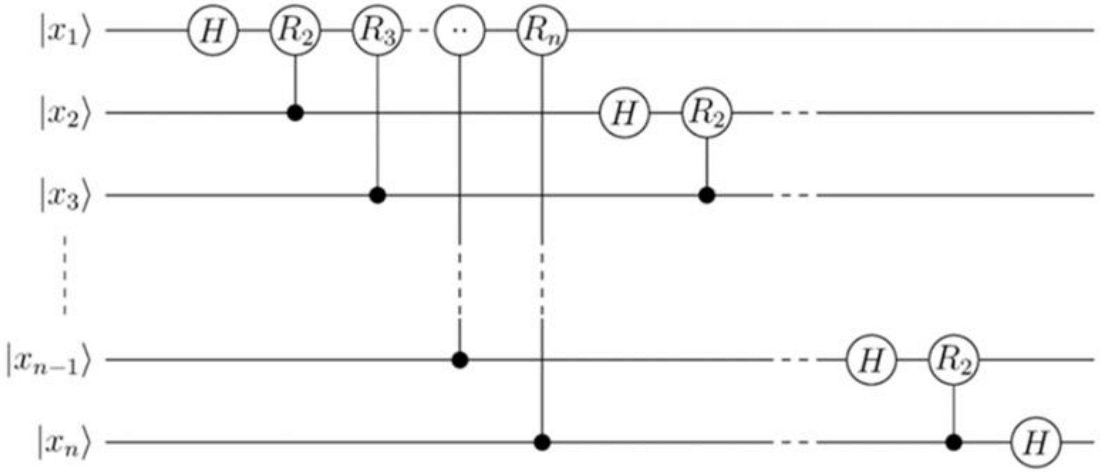
3 single qubit gates + 4 two qubits gate (SWAP gate)

— Computational cost for n qubits (or for 2^n bits): QFT $\Theta(n^2)$, FFT $\Theta(n \cdot 2^n)$

Picture: https://www.wikiwand.com/en/Quantum_Fourier_transform

Real cases usage

— The algorithm itself is easy to be generalized from 3-qubit to n-qubit.



— But it is hard to create a complex enough input qubit string, e.g. Sine wave.

Example: $01000000 \rightarrow |001\rangle$

$$12345678 \rightarrow \frac{1}{\sqrt{M}}(|000\rangle + 2|001\rangle + 3|010\rangle + 4|011\rangle + 5|100\rangle + 6|101\rangle + 7|110\rangle + 8|111\rangle)$$

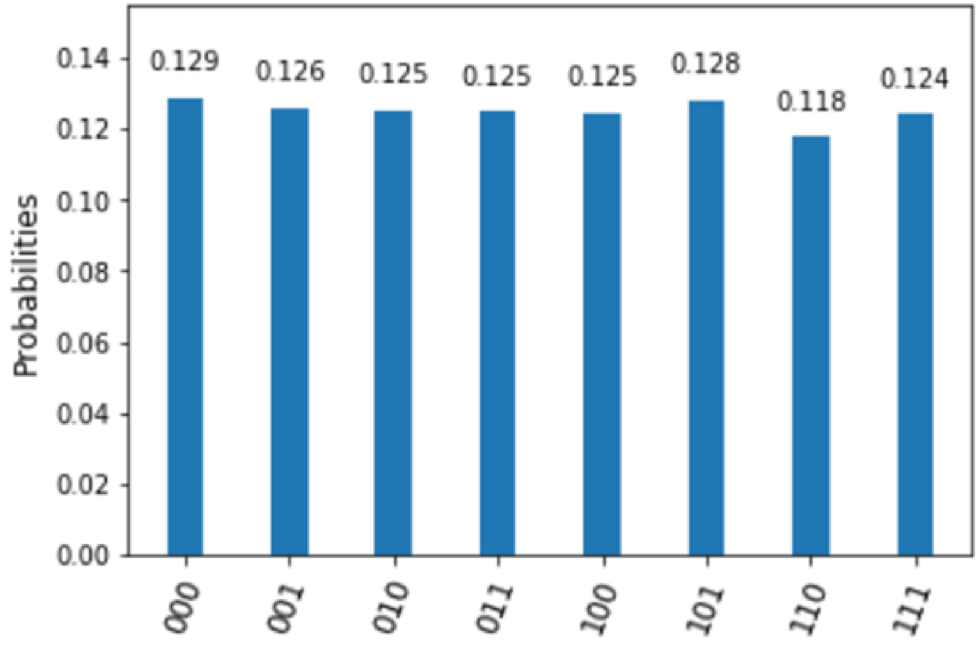
Where $M = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2$ is the normalization factor.

Picture: https://www.wikiwand.com/en/Quantum_Fourier_transform

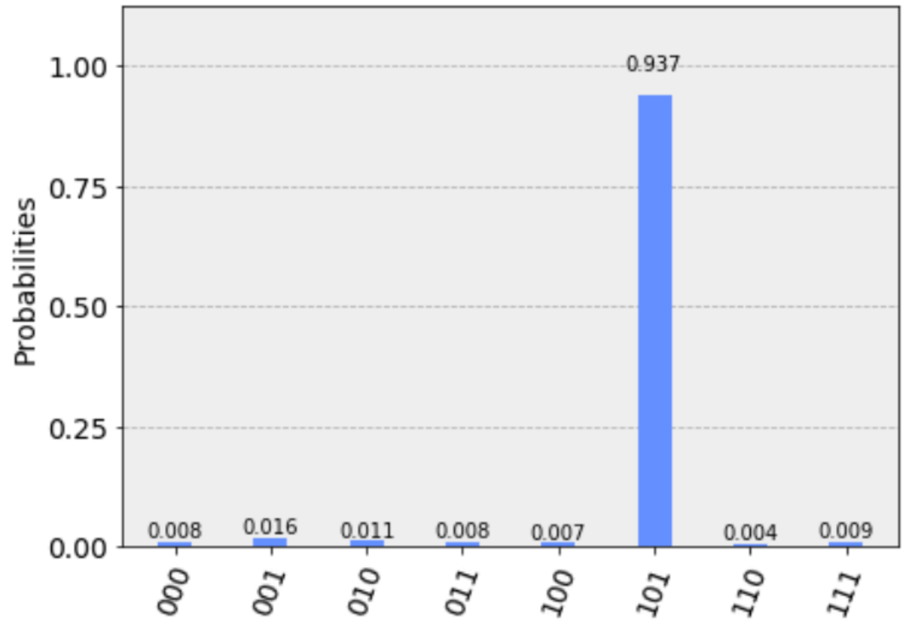
Real cases usage

$$12345678 \rightarrow \frac{1}{\sqrt{M}}(|000\rangle + 2|001\rangle + 3|010\rangle + 4|011\rangle + 5|100\rangle + 6|101\rangle + 7|110\rangle + 8|111\rangle)$$

Where $M = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2$ is the normalization factor.



Use Hadamard gates

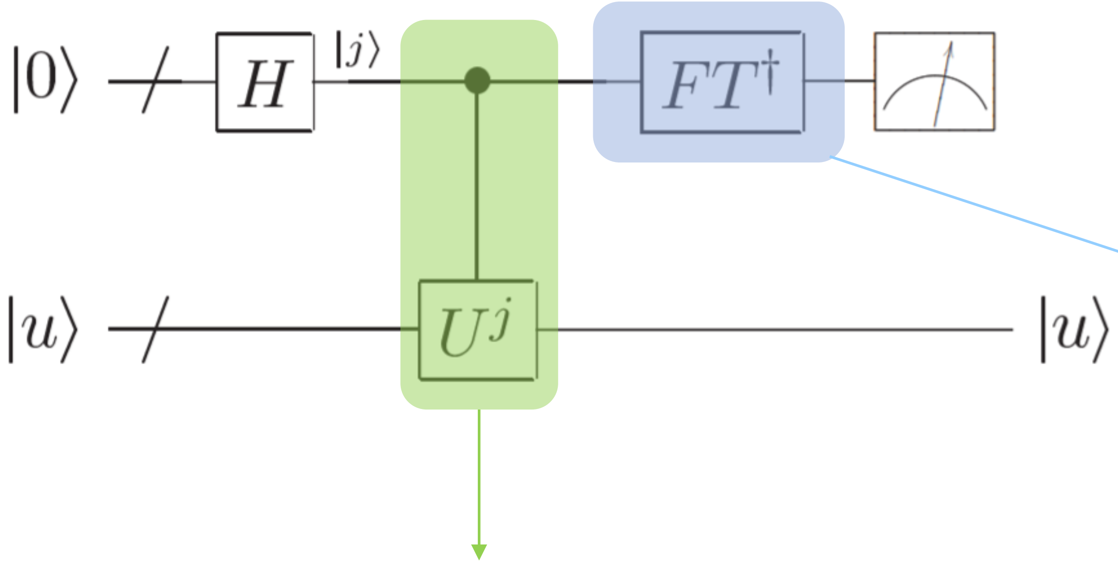


Grover's search

Picture: https://github.com/Qiskit/qiskit-tutorials/blob/master/community/algorithms/grover_algorithm.ipynb

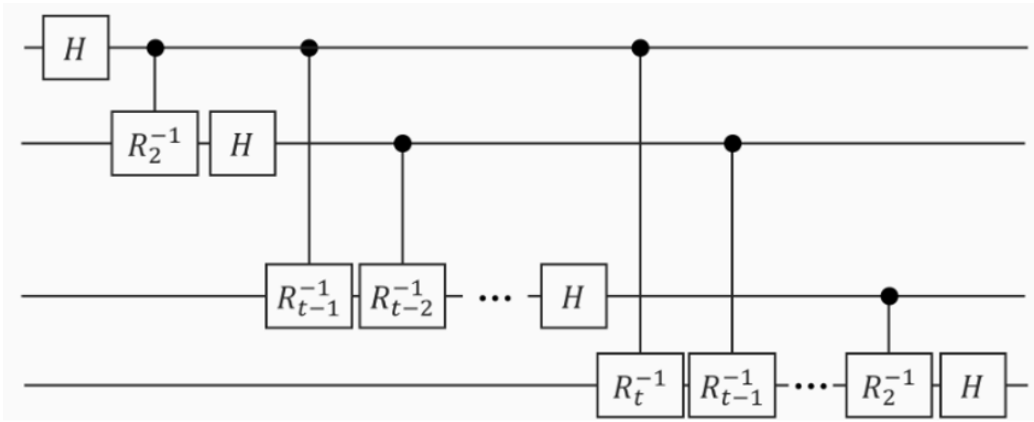
Application: Quantum Phase Estimation

Quantum phase estimation: to calculate the eigenvalue of U



Controlled-U gate

Inverse quantum Fourier transform:



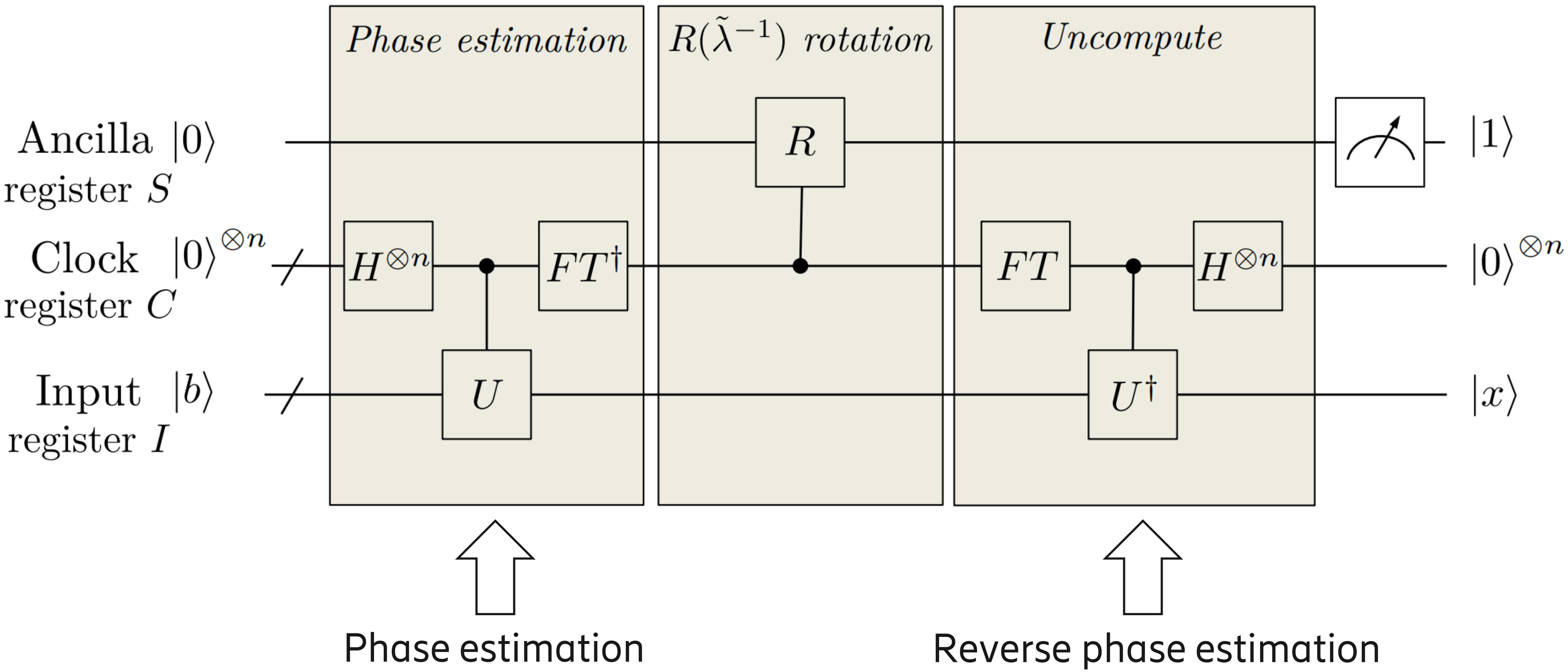
$$|j\rangle \longrightarrow \frac{1}{2^{\frac{n}{2}}} \sum_{k=0}^{2^n-1} e^{-2\pi i jk/2^n} |k\rangle$$

$$R_k{}^{-1} \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{-\frac{2\pi i}{k}} \end{bmatrix}$$

Picture: https://www.qtumist.com/info/PE-H5/index.html

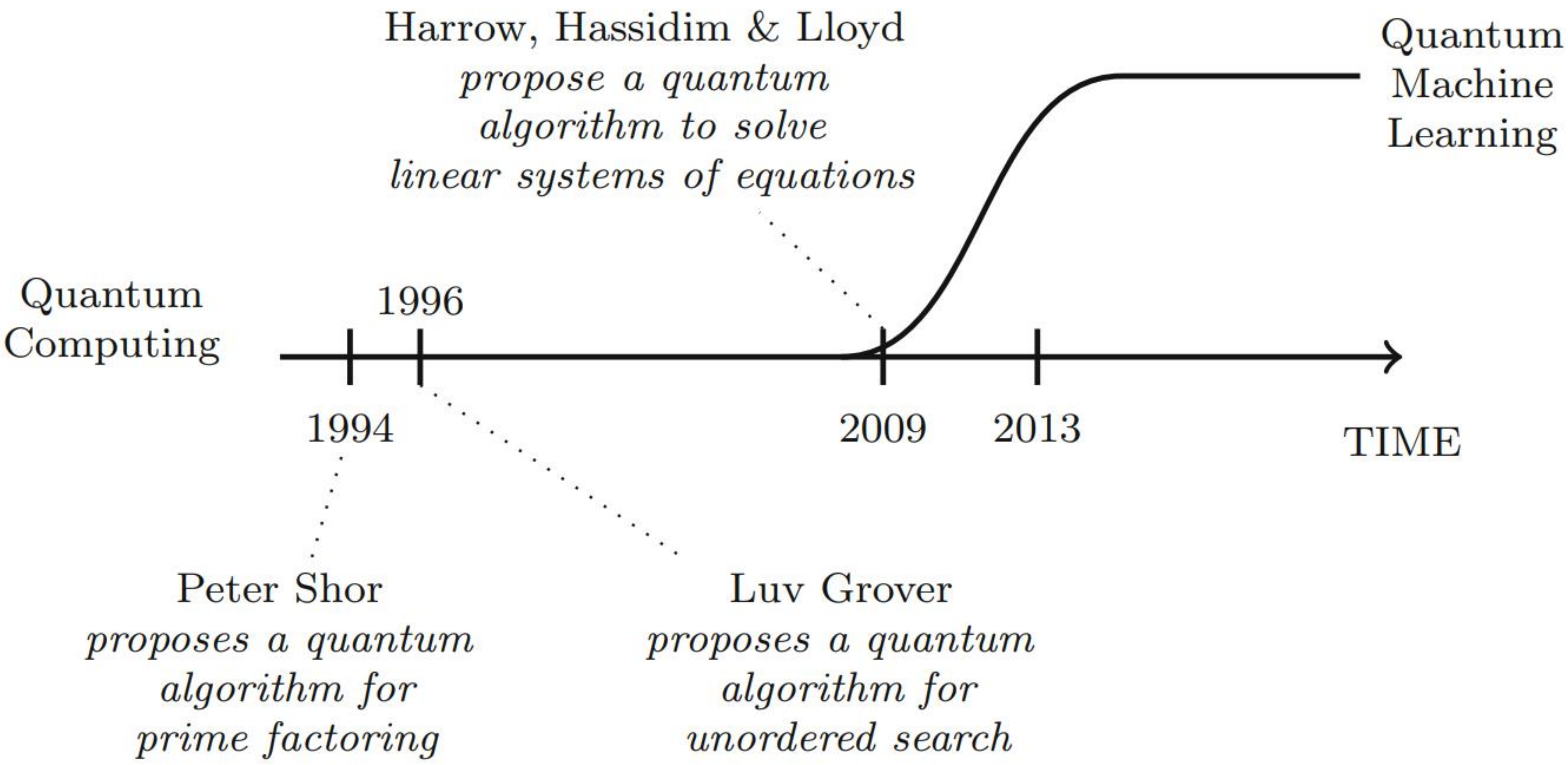
Application: HHL Algorithm - Quantum algorithm for linear systems of equations

Solve $\vec{A}x = \vec{b}$ using a quantum computer



Picture: https://arxiv.org/pdf/1802.08227.pdf

Application: HHL Algorithm- Milestone for QML



Usage: quantum machine learning algorithms including bayesian inference, qSVM, qPCA…

Picture: M. Schuld and F. Petruccione. Supervised learning with quantum computers. Springer, 2018.

Materials

- Books:
  - Quantum Computation and Quantum Information, 2002
  - Quantum Machine Learning: What Quantum Computing Means to Data Mining, 2014
  - Supervised Learning with Quantum Computers, 2018
- Online courses:
  - edX & University of Toronto: Quantum Machine Learning
  - edX & TU Delft: Quantum Cryptography
  - edX TUDelft: The building blocks of a quantum computer I
  - edX TUDelft: The building blocks of a quantum computer II