2022 IonQ & Qcenter Quantum Challenge

Dohun Kim
EE Department, POSTECH, South Korea
dohunkim@postech.ac.kr

October 4, 2022

1. Variational Quantum Eigensolver
2. Generative Adversarial Network Problem

2.1 For small size images

2.1.1 Describe how GAN would be trained and its loss functions. Consider both the Generator and Discriminator.

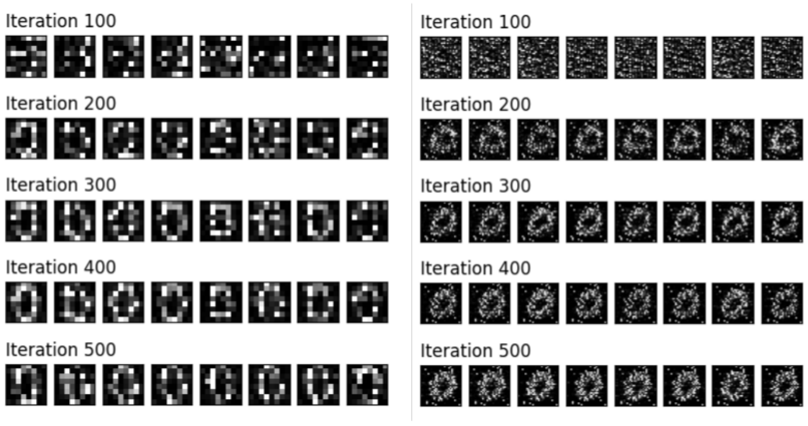
GAN comprises two neural networks, a generator and a discriminator, which are trained simultaneously. The generator G generate the fake images from the noise (latent vector), and the discriminator D classifies whether the incoming images is fake or real. These two network improves each other, so that G can generate a realistic images. G and D are trained by loss functions L_G and L_D, respectively.

$$\begin{aligned} L_D &= -\left(y\log(D(x)) + (1-y)\log(D(G(z))\right) \\ L_G &= ((1-y)\log(1-D(G(z)))) \end{aligned} \tag{1}$$

From (1), x, y, and z means real image, label (0: fake, 1: real), and latent vector. L_D is designed to maximize the probability of discriminating real and fake images ($D(x)$ and ($D(G(z))$). On the other hand, L_G minimized the probability of the generated image being discriminated as fake. At the early stage of training, the discriminator is will learn to recognize the difference between the fake images and real images. After few steps, the discriminator starts to mark the generated images as fake, then the generator starts to modify itself so that can deceive the discriminator, making more realistic images.
For the *Quantum patch GAN* specifically, the generator is patch-based quantum neural network, and the discriminator is a classical neural network. In this paper, the experimental results was 8x8 gray-scale image of digit 0, which is generated by quantum patch GAN. They used 5 qubit latent space and taking 1 qubit as ancilla, the output of single subgenerator was $2^{5-1} = 16$. Therefore, the number of subgenerator was set to 4, which makes $4 \times 2^{5-1} = 64 = 8$.

2.1.2 Design a GAN described in the paper and train it with a single digit 0. The result images should be recognizable as 0. Submit the python code, Dockerfile and a brief document describing the results of the solution.

We generated both 8x8 and 28x28 image with quantum patch GAN, One with the 4 subgenerator, and the other one with 49 subgenerator ($49 \times 2^{5-1} = 28$). The shape of the discriminator was set to [784, 64, 16, 1] with sigmoid activation at last layer. The layer depth of every subgenerator was set to 6 and optimized by SGD.



(a) 8x8 (b) 28x28

Figure 1: Images generated with fixed noise at every 100 iterations.

2.2 For large size images

2.2.1 Describe how you would approach this problem for large images with the limited number of qubits.

Since this task is an image generation, adopting the structure of convolutional neural networks is natural. The representative CNN-based GAN is Deep Convolutional GAN (DCGAN). The discriminator of DCGAN comprises 2D convolution layers and leaky-ReLU activation. The generator of DCGAN is the reversed form of discriminator, replacing the 2D Convolution with Transposed Convolution. The transposed convolution is the convolution but with sparsified input feature, making the output feature is larger than input.

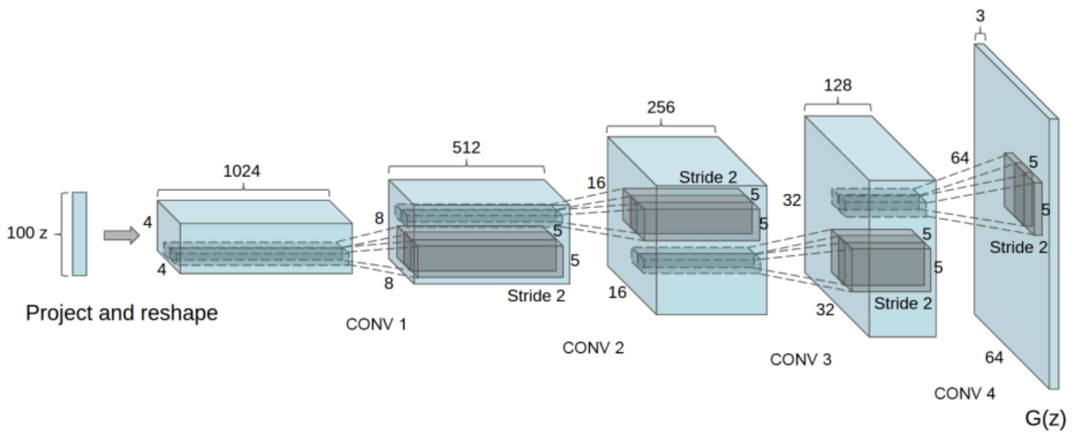


Figure 2: The structure of DCGAN generator

Furthermore, the convolution operation itself has the inherent parameter reduction compared to fully-connected network because the same weight of the filter is applied to the every patch of input data. Also, quantum circuit has ability to exponential scaling up, it suits the objective of limited hardware. Therefore, we suggest the quantum transposed convolution (Fig. 3), the multi-step generator with subgenerator using. In this design, every quantum circuits require only 4 qubits, and in-layer circuit reusing.

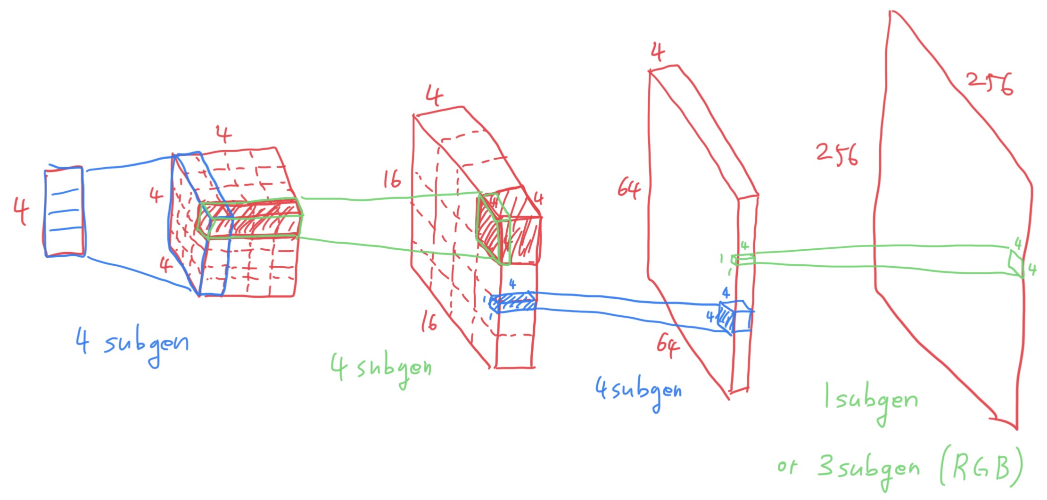


Figure 3: Quantum generator structure with proposed quantum transposed convolution

---

2.2.2 Explain the function of the GAN you are designing. Design and train it with datasets that have an image size of 256x256. Evaluate the performance of the results. Submit the python code, Dockerfile and a brief document describing the evaluation and results of the solution.

---

3. Quantum Phase Estimation Challenge

3.1 Task 1: Show that $|\Psi_1\rangle$ and $|\Psi_2\rangle$ are Eigenstates of the Hadamard gate and find their respective eigenvalues

$$\begin{aligned} |\Psi_1\rangle &= N_1 \begin{pmatrix} 1+\sqrt{2} \\ 1 \end{pmatrix} \\ |\Psi_2\rangle &= N_2 \begin{pmatrix} 1-\sqrt{2} \\ 1 \end{pmatrix} \end{aligned} \tag{2}$$

where N_1 and N_2 are normalization factors.

$$\begin{aligned} \forall x \in \mathbb{R}^2 \quad Hx = \lambda x \quad &\Rightarrow \quad (H - \lambda I)\,x = 0 \\ &\Rightarrow \quad \det(H - \lambda I) = 0 \\ &\Rightarrow \quad \begin{vmatrix} 1/\sqrt{2} - \lambda & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} - \lambda \end{vmatrix} = 0 \\ &\Rightarrow \quad \lambda^2 - 1 = 0 \\ &\Rightarrow \quad \lambda_1 = 1,\ \lambda_2 = -1 \end{aligned} \tag{3}$$

$$\begin{aligned} \begin{pmatrix} 1/\sqrt{2} - 1 & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} - 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0 \quad &\Rightarrow \quad \frac{1}{\sqrt{2}} \begin{pmatrix} \left(1-\sqrt{2}\right)x_1 + x_2 \\ x_1 - \left(1+\sqrt{2}\right)x_2 \end{pmatrix} = 0 \\ &\Rightarrow \quad x_1 = \left(1+\sqrt{2}\right)x_2 \\ &\Rightarrow \quad v_1 = \begin{pmatrix} 1+\sqrt{2} \\ 1 \end{pmatrix} \end{aligned} \tag{4}$$

$$\begin{aligned}
\begin{pmatrix} 1/\sqrt{2}+1 & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2}+1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0 \quad &\Rightarrow \quad \frac{1}{\sqrt{2}}\begin{pmatrix} \left(1+\sqrt{2}\right)x_1 + x_2 \\ x_1 - \left(1-\sqrt{2}\right)x_2 \end{pmatrix} = 0 \\
&\Rightarrow \quad x_1 = \left(1-\sqrt{2}\right)x_2 \\
&\Rightarrow \quad v_2 = \begin{pmatrix} 1-\sqrt{2} \\ 1 \end{pmatrix}
\end{aligned} \tag{5}$$

$$\therefore \quad |\Psi_1\rangle = \frac{v_1}{\|v_1\|} = N_1\begin{pmatrix} 1+\sqrt{2} \\ 1 \end{pmatrix}, \quad |\Psi_2\rangle = \frac{v_2}{\|v_2\|} = N_2\begin{pmatrix} 1-\sqrt{2} \\ 1 \end{pmatrix} \tag{6}$$

where corresponding eigenvalues are $\lambda_1 = 1, \lambda_2 = -1$, respectively

---

3.2 Task 2: Quite frequently, in quantum computation, estimating the eigenvalue of a unitary require estimating the phase. Because all unitary matrices can have their eigenvalues in the form of $e^{i\theta}$ where theta is real. Such that, $U|\Psi\rangle = e^{i\theta}|\Psi\rangle$. Please explain why this is the case.

$$\begin{aligned}
U|\Psi\rangle = \lambda|\Psi\rangle \;&\Rightarrow\; \langle\Psi| U^\dagger = \langle\Psi|\lambda^* \\
&\Rightarrow\; \langle\Psi|\Psi\rangle = \langle\Psi|I|\Psi\rangle = \langle\Psi|U^\dagger U|\Psi\rangle = \langle\Psi|\lambda^*\lambda|\Psi\rangle = |\lambda|^2\langle\Psi|\Psi\rangle \quad (\because U : \text{unitary}) \\
&\Rightarrow\; |\lambda| = 1 \\
&\Rightarrow\; \lambda = e^{i\theta} \text{ for some } \theta \in \mathbb{R}
\end{aligned} \tag{7}$$

3.3 Task 3: In estimating the eigenvalues of the unitary gate, it is necessary to apply multiple controlled unitary operations. Show how to construct a controlled version of an arbitrary unitary and provide an example quantum circuit (with executable code). Show that the controlled version is indeed constructed correctly and include a justification for your validation.

For arbitrary unitary gate, the controlled version of it can be described as below.

$$U = \begin{pmatrix} \cos\theta & -e^{i\lambda}\sin\theta \\ e^{i\phi}\sin\theta & e^{i(\phi+\lambda)}\cos\theta \end{pmatrix} \quad \Rightarrow \quad c(U) = \begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta & -e^{i\lambda}\sin\theta \\ 0 & 0 & e^{i\phi}\sin\theta & e^{i(\phi+\lambda)}\cos\theta \end{pmatrix} \tag{8}$$

By ZYZ decomposition, any unitary U can be decomposed to

$$U = e^{i\frac{\phi+\lambda}{2}} R_z(\phi) R_y(2\theta) R_z(\lambda) \,, \tag{9}$$

where R_y and R_z are the rotation gates:

$$R_y(\theta) = \begin{pmatrix} \cos\theta/2 & -\sin\theta/2 \\ \sin\theta/2 & \cos\theta/2 \end{pmatrix}, \quad R_z(\theta) = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix} . \tag{10}$$

Then we can modify these gates to include two X gates, which is known as ABC decomposition:

$$U = e^{i\alpha} AXBXC \,, \quad \text{such that } ABC = I \,, \tag{11}$$

we obtain

$$\alpha = \frac{\phi+\lambda}{2} \,, \quad A = R_z(\phi)R_y(\theta) \,, \quad B = R_y(-\theta)R_z\left(-\frac{\phi+\lambda}{2}\right) \,, \quad C = R_z\left(\frac{\lambda-\phi}{2}\right) . \tag{12}$$

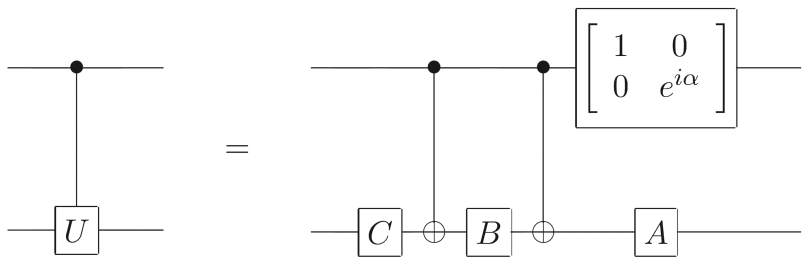


Figure 4: Circuit implementation of controlled unitary gate.

Fig. 4 shows the circuit implementation of controlled unitary gate based on the ABC decomposition. If the state of control (upper) qubit is $|0\rangle$, then nothing happens to the target (lower) qubit ($\because ABC = I$), while if the control qubit is on $|1\rangle$, then U is applied to the target qubit ($\because U = e^{i\alpha}AXBXC$).
The validation of the equivalence can be done by simply multiplying the matrices.

$$\begin{aligned}(\text{circuit in Fig. 4}) &= (P(\alpha)\otimes I)(I\otimes A)c(X)(I\otimes B)c(X)(I\otimes C)\\ &= \begin{pmatrix} I & \\ & e^{i\alpha}I\end{pmatrix}\begin{pmatrix} A & \\ & A\end{pmatrix}\begin{pmatrix} I & \\ & X\end{pmatrix}\begin{pmatrix} B & \\ & B\end{pmatrix}\begin{pmatrix} I & \\ & X\end{pmatrix}\begin{pmatrix} C & \\ & C\end{pmatrix}\\ &= \begin{pmatrix} ABC & \\ & e^{i\alpha}AXBXC\end{pmatrix} = \begin{pmatrix} I & \\ & U\end{pmatrix} = c(U)\end{aligned} \tag{13}$$

3.4 Task 4:The Discrete Fourier Transform is an incredibly useful tool in mathematics and is used to transform a signal into the frequency domain representation. That is, the DFT of a wave will reveal the relative strengths of the periodic components. The general form of the DFT is:

$$\phi_k = \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} a_j\omega^{jk}\ , \quad \omega = e^{2\pi i/N}\ . \tag{14}$$

Show that the DFT can be written in Matrix / Vector notation.

$$\begin{aligned}\phi_k &= \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} a_j\omega^{jk}\\ &= \frac{1}{\sqrt{N}}\left(a_0 + a_1\omega^k + a_2\omega^{2k} + \cdots + a_{N-1}\omega^{(N-1)k}\right)\end{aligned} \tag{15}$$

$$\begin{aligned}\phi = \begin{pmatrix}\phi_0\\ \phi_1\\ \phi_2\\ \vdots\\ \phi_{N-1}\end{pmatrix} &= \frac{1}{\sqrt{N}}\begin{pmatrix} a_0 & + & a_1 & + & a_2 & + & \cdots & + & a_{N-1}\\ a_0 & + & a_1\omega & + & a_2\omega^2 & + & \cdots & + & a_{N-1}\omega^{N-1}\\ a_0 & + & a_1\omega^2 & + & a_2\omega^4 & + & \cdots & + & a_{N-1}\omega^{2(N-1)}\\ \vdots & & \vdots & & \vdots & & & & \vdots\\ a_0 & + & a_1\omega^{N-1} & + & a_2\omega^{2(N-1)} & + & \cdots & + & a_{N-1}\omega^{(N-1)^2}\end{pmatrix}\\ &= \underbrace{\frac{1}{\sqrt{N}}\begin{pmatrix} 1 & 1 & 1 & \cdots & 1\\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1}\\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)^2}\end{pmatrix}}_{:= F}\underbrace{\begin{pmatrix} a_0\\ a_1\\ a_2\\ \vdots\\ a_{N-1}\end{pmatrix}}_{:= a} = Fa\end{aligned} \tag{16}$$

3.5 Task 5: Prove the matrix computing the DFT is unitary and therefore computable via a quantum gate by showing that for a given ith and jth column of a QFT:

$$\begin{aligned} &\text{if } i = j;\ \left\langle F_i \middle| F_j \right\rangle = 1 \\ &\text{if } i \neq j;\ \left\langle F_i \middle| F_j \right\rangle = 0 \end{aligned} \tag{17}$$

$$|F_k\rangle = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 \\ \omega^k \\ \omega^{2k} \\ \vdots \\ \omega^{(N-1)k} \end{pmatrix}, \quad \langle F_k| = |F_k\rangle^\dagger = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & \omega^{-k} & \omega^{-2k} & \cdots & \omega^{-(N-1)k} \end{pmatrix} \tag{18}$$

$$\left\langle F_i \middle| F_j \right\rangle = \frac{1}{N} \begin{pmatrix} 1 & \omega^{-i} & \omega^{-2i} & \cdots & \omega^{-(N-1)i} \end{pmatrix} \begin{pmatrix} 1 \\ \omega^j \\ \omega^{2j} \\ \vdots \\ \omega^{(N-1)j} \end{pmatrix} = \frac{1}{N} \sum_{n=0}^{N-1} \omega^{n(j-i)} \tag{19}$$

$$\begin{aligned} i = j \ &\Rightarrow\ \left\langle F_i \middle| F_j \right\rangle = \frac{1}{N} \sum_{n=0}^{N-1} \omega^0 = 1 \\ i \neq j \ &\Rightarrow\ \left\langle F_i \middle| F_j \right\rangle = \frac{1}{N} \sum_{n=0}^{N-1} e^{i\theta_n} = 0 \quad \left(\theta_n = 2\pi(j-i)\frac{n}{N}\right) \end{aligned} \tag{20}$$

$\therefore$ (17) is true.

$$F = \begin{pmatrix} |F_0\rangle & |F_1\rangle & \cdots & |F_{N-1}\rangle \end{pmatrix}, \quad F^\dagger = \begin{pmatrix} \langle F_0| \\ \langle F_1| \\ \vdots \\ \langle F_{N-1}| \end{pmatrix} \tag{21}$$

$$\begin{aligned} F^\dagger F &= \begin{pmatrix} \langle F_0| \\ \langle F_1| \\ \vdots \\ \langle F_{N-1}| \end{pmatrix} \begin{pmatrix} |F_0\rangle & |F_1\rangle & \cdots & |F_{N-1}\rangle \end{pmatrix} \\ &= \begin{pmatrix} \langle F_0|F_0\rangle & \langle F_0|F_1\rangle & \cdots & \langle F_0|F_{N-1}\rangle \\ \langle F_1|F_0\rangle & \langle F_1|F_1\rangle & \cdots & \langle F_1|F_{N-1}\rangle \\ \vdots & \vdots & & \vdots \\ \langle F_{N-1}|F_0\rangle & \langle F_{N-1}|F_1\rangle & \cdots & \langle F_{N-1}|F_{N-1}\rangle \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} = I \end{aligned} \tag{22}$$

$\therefore$ F is unitary.

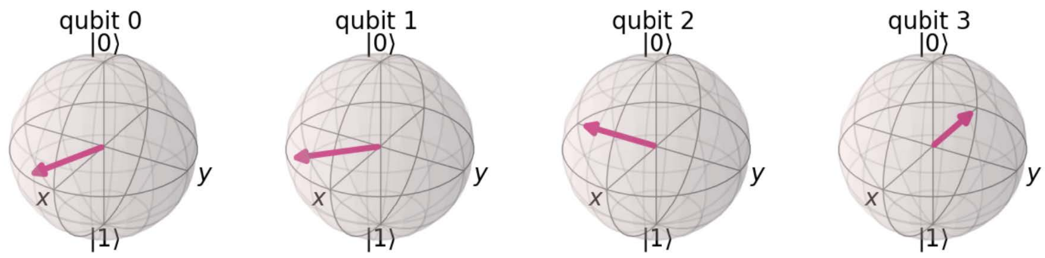
3.6 Task 6: Having shown that the Fourier transform is unitary, we can efficiently construct it using quantum gates. Taking the example where $N = 2$, $\omega = e^{\pi i} = -1$, show that:

$$QFT = H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{23}$$

$$QFT = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)^2} \end{pmatrix} \tag{24}$$

$$QFT_{N=2} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & \omega \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & e^{\pi i} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = H \tag{25}$$

3.7 Task 7: Create a Quantum Fourier Transform circuit and apply it to the state: $|1111\rangle$. Run your circuit on both a QPU and a simulator, then discuss and compare your results. The resulting qubits states should be in the following states:



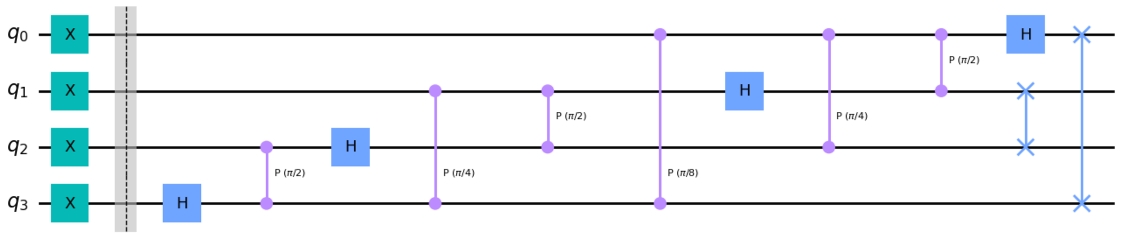
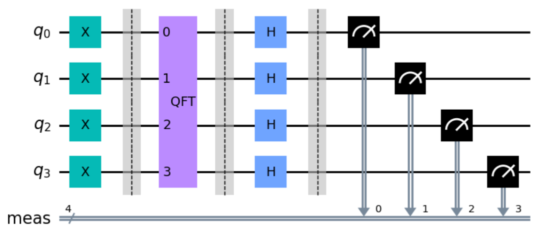
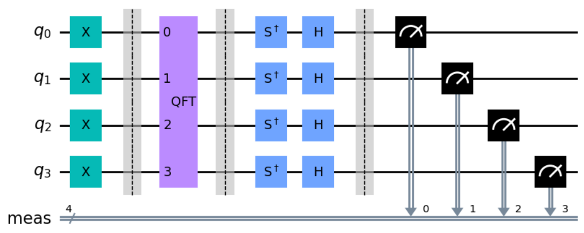


Figure 5: Basic QFT circuit with state initialization

QFT circuit can be created by using the Qiskit circuit library. Then we set the initial state to $|1111\rangle$ by applying X gate to every qubits. Fig. 5 shows the constructed quantum circuit. To run this circuit on a QPU or a simulator, the measurement for all qubits are required. However, since the final state of the QFT is on Fourier basis, which means bloch vector lies on x-y plane of bloch sphere, the measurement output in a computational basis (z basis) always have probability {0: 50%, 1: 50%}, dropping the phase information. Therefore, we need a method to obtain the phase of the final state.



(a) Circuit with the measurement in x basis



(b) Circuit with the measurement in y basis

Figure 6: Implementation of X and Y measurement by rotating the basis.

This phase calculation can be done with measurement in x and y basis. The X measurement can be implemented with Hadamard gate followed by the Z measurement. Similarly, the Y measurement is equivalent to $S^{\dagger}H$ followed by the Z measurement. With this basis-rotating method, the outcomes of Z measurement becomes the probabilities in x and y basis. Then, we can extract the phase from these probabilities. According to the Born rule, the probability that the given state $|\Psi\rangle$ is measured to $|1\rangle$ is:

$$P(1|\Psi) = |\langle 1|\Psi\rangle|^2 \tag{26}$$

and the arbitrary state of single qubit is described by:

$$|\Psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle . \tag{27}$$

By plugging (27) into (26), we obtain:

$$P(1|\Psi) = \left|\cos\frac{\theta}{2}\langle 0|1\rangle + e^{i\phi}\sin\frac{\theta}{2}\langle 1|1\rangle\right|^2 = \left|e^{i\phi}\sin\frac{\theta}{2}\right|^2 = \sin^2\frac{\theta}{2} = \frac{1}{2}(1-\cos\theta) . \tag{28}$$

Let θ_x and θ_y denote the angle with respect to the x and y basis, respectively. Then we get:

$$\phi = \theta_x , \quad \phi = \theta_y + \frac{\pi}{2}, \tag{29}$$

where ϕ is the phase of final state. For the probabilities of $|1\rangle$ in rotated x basis and y basis (p_x and p_y), the relationship with ϕ can be described as:

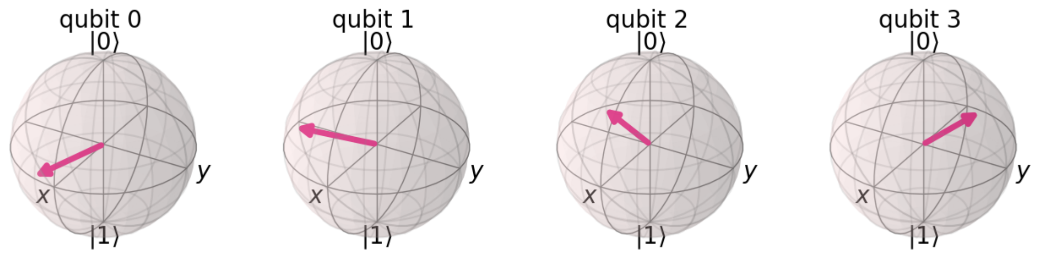
$$\begin{gathered} p_x = P_x(1|\Psi) = \frac{1}{2}(1-\cos\theta_x) = \frac{1}{2}(1-\cos\phi) \\ p_y = P_y(1|\Psi) = \frac{1}{2}\left(1-\cos\theta_y\right) = \frac{1}{2}(1-\sin\phi) \\ \cos\phi = 1-2p_x , \quad \sin\phi = 1-2p_y \\ \therefore \phi = \operatorname{atan2}\left(1-2p_y,\ 1-2p_x\right) \end{gathered} \tag{30}$$

The Table 1 shows the result of the execution on a QPU and a simulator.

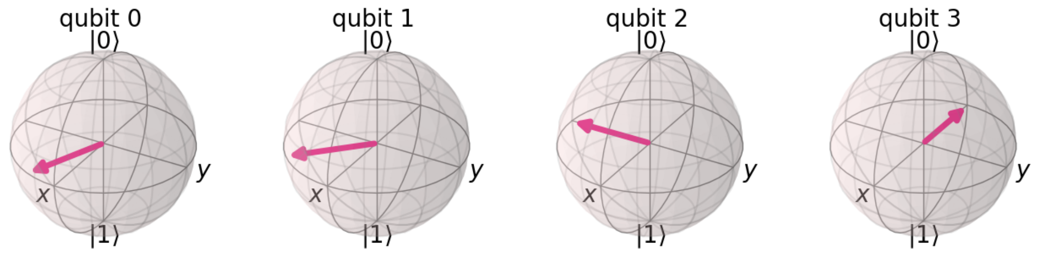
Table 1: Results of experiments on a QPU and a simulator

Device	QPU (shot=10,000)				Simulator (shot=10,000)			
Qubit	q_0	q_1	q_2	q_3	q_0	q_1	q_2	q_3
p_x	0.02213	0.43640	0.73119	0.99030	0.03501	0.14071	0.49490	0.99999
p_y	0.64712	0.99594	0.94334	0.40200	0.68381	0.84773	0.99997	0.49620
ϕ	-0.29865	-1.44324	-2.05148	2.94432	-0.37646	-0.76905	-1.56060	3.13399
$\Delta\phi = \phi^* - \phi$	-0.09405	0.65784	0.48069	0.19728	-0.01624	-0.01635	-0.01020	0.00760

We know that the exact value of ϕ is $\phi^* = [-\pi/8,\ -\pi/4,\ -\pi/2,\ \pi]$, thus we can get the error $\Delta\phi$.



(a) Bloch sphere plot of the output statevector on QPU



(b) Bloch sphere plot of the output statevector on Simulator

Figure 7: Visualization of the statevector reconstructed by X-Y measurement-based phase calculation

The magnitude of the error $\Delta\phi$ is larger on QPU compared to simulator. Also, the results on QPU tend to be over-rotated (clockwise). This is expected to be related to the physical implementation of quantum gates and its fidelity.

3.8 Task 8: Construct a circuit that estimates the phase of a given unitary U to 10-bit precision, given an eigenvector $|v\rangle$ using the inverse Quantum Fourier Transform. Check your answer and justify your methodology.

To validate the 10-bit precision QPE circuit in Fig. 8, we first generate random unitary matrices, then show how the estimations are accurate statistically. The random unitary is generated by random sampling of the θ, ϕ, λ from (8) with a uniform distribution in $[0, 2\pi]$. Note that we can derive the exact eigenstate and its corresponding eigenvalue by eigenvalue decomposition. Fig. 9 shows the examples of success and failure case. As a result, we got exact success rate of 67.3%, which means the $2^n\theta$ exactly matches with ground truth values. Also, Fig. 10 shows the distribution of error of estimated phase.

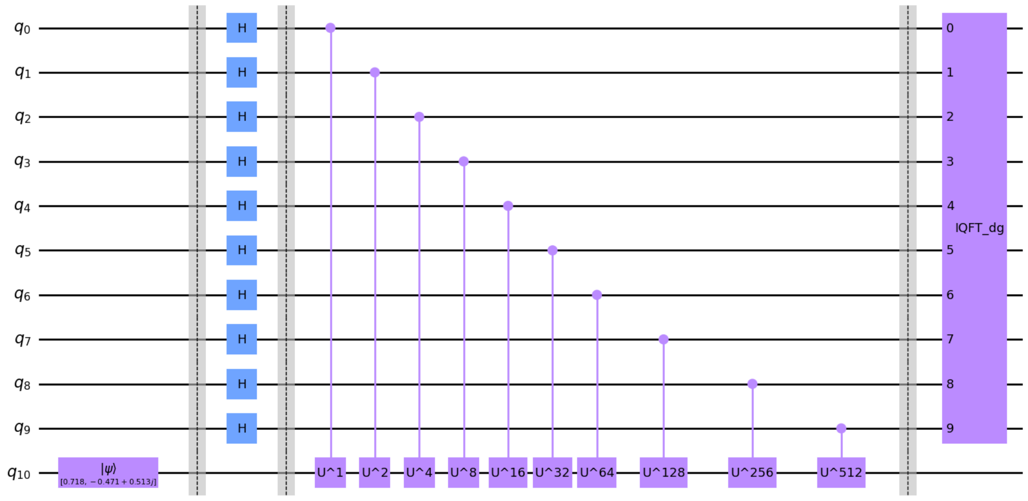
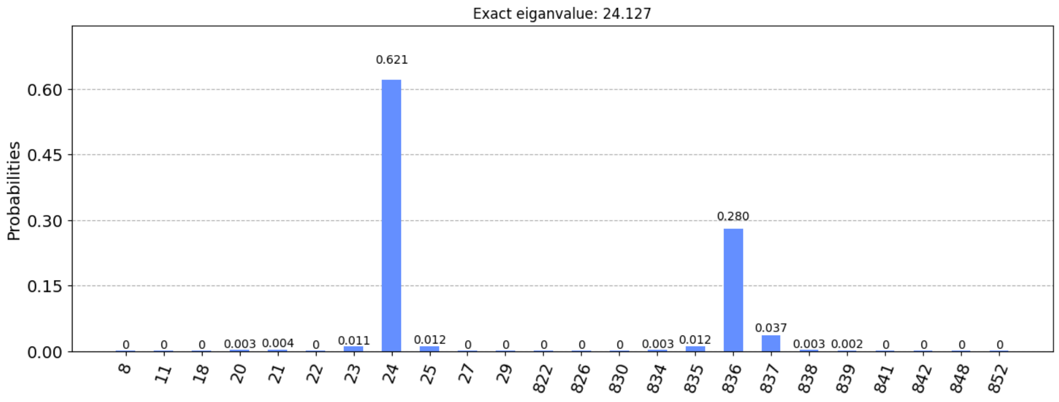
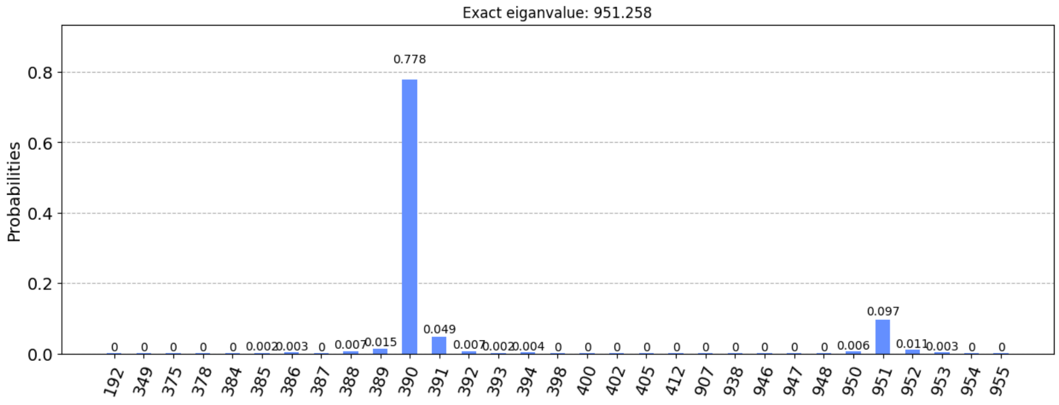


Figure 8: QPE circuit with 10-bit precision. (Classical bits and measurement blocks are omitted.)

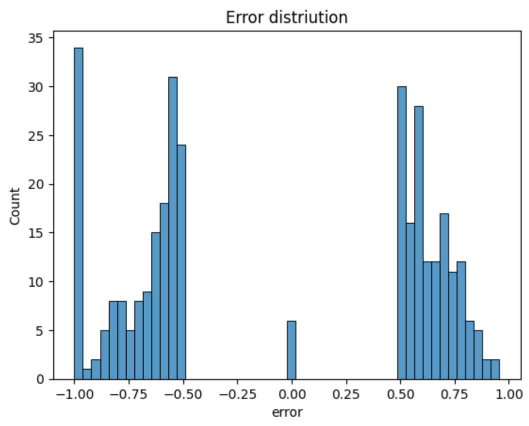


(a) Success

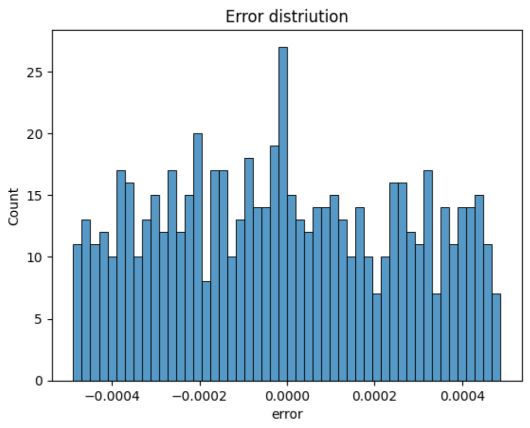


(b) Failure

Figure 9: Simulation results on random unitary gates with 1024 shots.



(a) Success



(b) Failure

Figure 10: Phase error distributions of 1000 random unitary gates.

3.9 Task 9: Implement a 3-bit precision version of your circuit on a quantum processor with the following values:

$$U = \begin{pmatrix} 1 & 0 \\ 0 & e^{2i\pi/4} \end{pmatrix}, \ |v\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{31}$$

Interpret the result by comparing it with a noise-free simulator case, and discuss the error of your result.

$$U = \begin{pmatrix} 1 & 0 \\ 0 & e^{2i\pi/4} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = S \ , \quad |v\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{32}$$

As shown in (32), the given unitary U is S gate, and the eigenvector $|v\rangle$ is $|1\rangle$. Therefore, the powers of U, which is utilized in QPE circuit, becomes S, Z, and I gate.

$$U^{2^0} = S = P(\pi/2) \ , \quad U^{2^1} = S^2 = Z \ , \quad U^{2^2} = S^4 = Z^2 = I \tag{33}$$

Accordingly, the controlled version of these are $CP(\pi/2), CZ$ and I gate, which are directly executable on both QPU and simulator. Fig. 11 shows the constructed 3-bit QPE circuit.

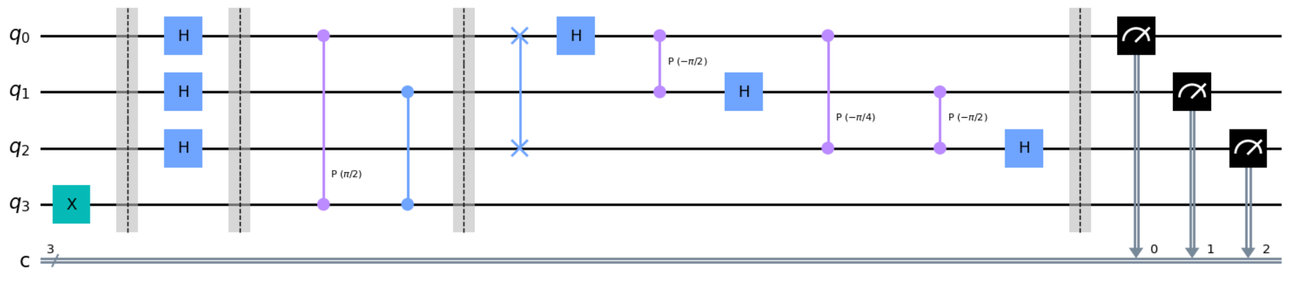
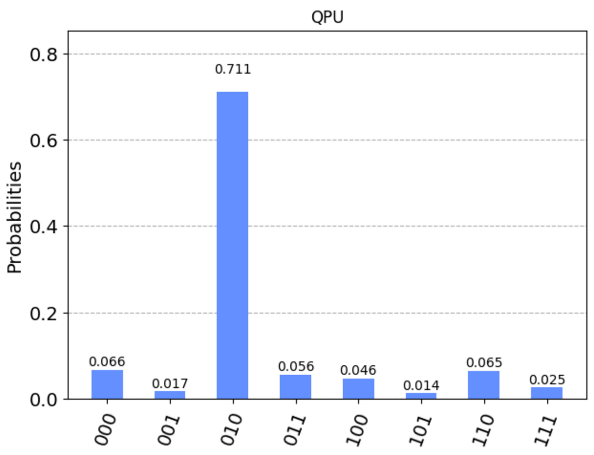
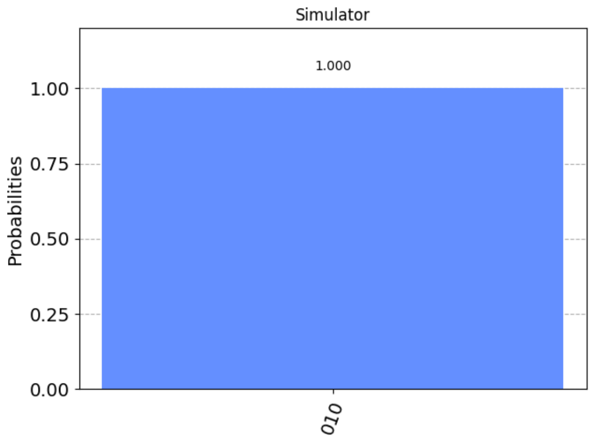


Figure 11: QPE circuit with 3-bit precision

For the measurement outcome $|q_0q_1q_2\rangle$, we make a phase estimation with $\theta = \dfrac{(q_0q_1q_2)_2}{2^3}$ corresponding to the eigenstate $|v\rangle$, such that $U|v\rangle = e^{2i\pi\theta}|v\rangle$.



(a) Probability distribution of output on QPU



(b) Probability distribution of output on Simulator

Figure 12: Visualization of the statevector reconstructed by X-Y measurement-based phase calculation

Fig. 12a shows that the value with highest probability on QPU result is $010(binary) = 2(decimal)$, and Fig. 12b also shows the output of $010(binary)$ with 100% probability. It means that the phase estimation is $\theta = 2/2^3 = 0.25$, which is a exact value since $e^{2i\pi/\theta} = e^{i\pi/2} = i$ is the corresponding eigenvalue of $|1\rangle$. Thus, 3-bit QPE has enough precision for the given unitary. However, the noise of physical quantum device (QPU) can generate the unexpected measurement outcomes. For example, the second most likely output is $000(binary) = 0(decimal)$, resulting to make estimation $\theta = 0$. This estimation is totally wrong since the eigenvalue $e^{2i\pi\theta} = e^0 = 1$ corresponds to the eigenstate $|0\rangle$. Therefore, it is better to make sufficient amount of shots and choose the most likely output for the estimation when using QPU.