CPEN 400Q Companion report

Policy Gradient Approach to Compilation of Variational Quantum Circuits

By: Ross Mojgani, Owen Guo, Kobe Ng, Akhil Veeraghanta

1. Introduction

Classical compilers are capable of taking high level code, and turning it into executable instructions that a machine is able to understand. For quantum computers, high level compilers are not nearly as mature as classical ones, but will need to be developed to allow programmers and scientists to be productive without the need to deal with the complexities of quantum computing hardware.

Quantum compilation involves the process of taking a general unitary matrix, and representing it in terms of a base universal gate set [1]. This problem is often approached by variational quantum algorithms which classically train a parametrized quantum circuit. A common issue with variational quantum algorithms are barren plateau regions where the cost gradients vanish exponentially in the number of qubits [2]. The paper, Policy Gradient Approach to Compilation of Variational Quantum Circuits, uses policy gradient methods to get around the issue of barren plateaus in the parameter space.

2. Programming framework

A core component of the project was to reproduce the results of the paper. The reproduction of results requires the use of a framework capable of simulating quantum circuits. This leads to an important decision to be made on which programming framework to perform the simulation in. The two major simulators that were considered were Qiskit, and Pennylane. Other simulators such as pyQuil and Q# were briefly considered, but the advantages from Qiskit and Pennylane made it clear that the final decision would be one of the two.

The major advantage from Qiskit stems from it having a large well supported community. Documentation on Qiskit is generally more complete, and there appeared to me more examples online of how to achieve things on Qiskit than PennyLane. For example, on the

quantum computing stack exchange there are 1353 questions tagged qiskit compared to 25 tagged pennylane at the time of writing [3]. Choosing Qiskit as the framework would most likely provide access to a larger community to ask questions and get help from. As an additional benefit, the paper deals with reinforcement learning which requires time to train models. Qiskit is generally a faster framework which would reduce computation time.

On the other hand, PennyLane is a less widely used framework. However, it provides the massive benefit that the group is more familiar with using PennyLane from assignments in class. In theory, this familiarity mitigates the advantage of a larger community and more extensive documentation. We also believed that the familiarity would help with debugging and understanding the software that we would write. As none of the group members have experience with reinforcement learning, minimising the overhead required to understand the software and debug any issues was a key consideration in the programming framework.

Due to the reasoning provided above, the group decided to use PennyLane as the programming framework.

3. Theory

3.1 Compilation with variational quantum circuits

The high level goal of the paper is to provide a method to compile unknown quantum unitiaries using a variational quantum circuit. The structure proposed for the circuit can be

seen in the lower image of Figure 1.

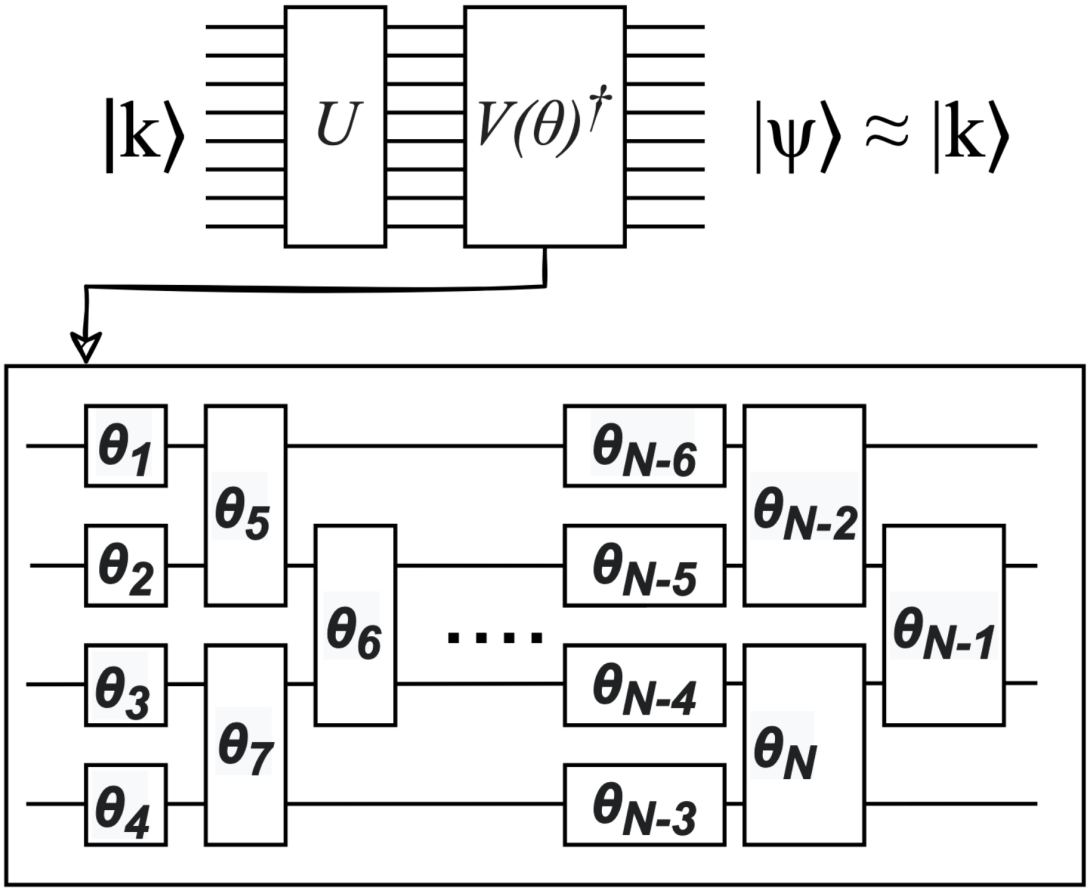

Figure 1. Single qubit gates are RY rot gates, and double qubit gates are RZZ

In order to evaluate the performance of compilation, the paper proposes a metric which corresponds to the fidelity between the initial and the final state as seen in the upper image of Figure 1.

$$\hat{F}(\theta) = \frac{1}{m}\sum_{k}^{m} |\langle k|V(\theta)^{\dagger}U|k\rangle|^2 \qquad (1)$$

If the compilation is effective, then the output state of the circuit as shown in the upper image of Figure 1 for any given input state will be very close to the original state. An effective compilation will therefore lead to the projection term being close to 1 while an ineffective compilation will lead to a lower value for F(θ).

The problem now lies in how for a given unitary U, the values of theta for the ansatz circuit should be chosen to best maximise this fidelity metric. The paper proposes the use of a “Policy Gradient” approach as both an efficient and noise resistant choice compared to other possible solutions.

3.2 Policy gradient for compilation

In the context of this work, Policy Gradient (PG) Reinforcement Learning (RL) seeks to optimise our choice of thetas in the ansatz through the optimization of parameters of a “policy” instead. The paper mentions that several policy optimization techniques exist, but suggests the use of an algorithm called “REINFORCE”, in which policies themselves are parameterized. The chosen policy of the paper follows of the form of a multivariate Gaussian distribution, parameterized by a mean vector μ and covariance matrix Σ.

$$\theta \sim \pi(x;\mu,\Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} e^{-(x-\mu)\Sigma^{-1}(x-\mu)^T} \qquad (2)$$

The goal is to find μ,Σ such that sampling from this distribution is likely to produce thetas that optimise our fidelity metric.

3.2.1 The PGRL “REINFORCE” Algorithm

The first step in PG REINFORCE is to define an objective function for the chosen policy that we wish to optimise.

$$J = \mathbb{E}_{\pi_\mu}[F] = \sum_k^m p_k \sum_\theta \pi(\theta|\mu,\Sigma)|\langle k|V(\theta)^\dagger U|k\rangle|^2 \qquad (3)$$

As we can see here the objective function J is based on the fidelity metric seen in Equation 1. Using the methods of “policy gradient theorem” the authors of the paper derive the gradient of this objective function.

$$\nabla_\mu J = \sum_k^m p_k \sum_\theta \pi(\theta|\mu,\Sigma)\nabla_\mu \log \pi(\theta|\mu,\Sigma)|\langle k|V(\theta)^\dagger U|k\rangle|^2 \qquad (4)$$

Firstly we note that the gradient was only considered for the mean vector parameter μ. This is intentional as the paper states that while learning in the covariance parameter is possible, they instead choose to follow a fixed iteration that over time scales the covariances to 0.

$$\Sigma(t) = (1 - t/T)\Sigma_i + t/T\Sigma_f$$

Secondly we note that the paper derived the gradient using the “Log-likelihood trick” where the log of the policy gradient of μ is defined as follows.

$$\nabla_\mu \log \pi(x; \mu, \Sigma) = \Sigma^{-1}(x - \mu)$$

Once the gradient has been calculated, we can then update our policy parameters using a RMSprop gradient update rule as seen in Equations 12 and 13 below.

$$\sigma_g^{(t)} = \gamma\sigma_g^{(t-1)} + (1 - \gamma)(\nabla_\xi J|_g)^2 \tag{12}$$

$$\xi \leftarrow \xi + \eta \frac{\nabla_\xi J|_g}{\sqrt{\sigma_g^{(t)} + \varepsilon}} \tag{13}$$

Note that these equations are written for the general case where $\xi \in \{\mu, \Sigma\}$, but we are only interested in updating the mean vector μ. After this step we have completed a single iteration of the Policy Gradient REINFORCE algorithm as should have a slightly better choice for μ,Σ. We can then continue to repeat this step to further optimise our policy parameters. The paper suggests a maximum number of iterations around 2000-3000 is necessary to saturate the objective function.

4. Results

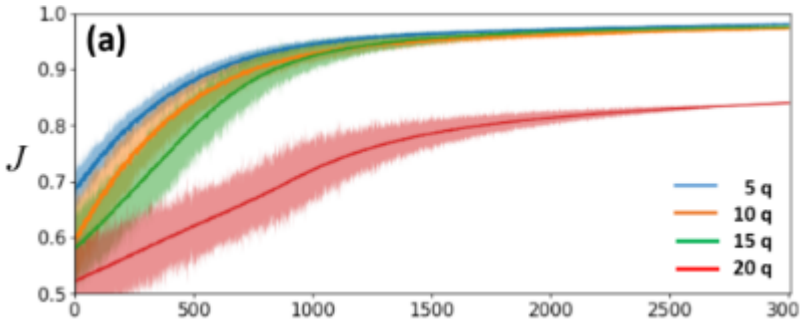

Figure 2. The core result of the paper in the form of a reward vs iteration graph for a various number of qubits. The last value of the x-axis should likely be 3000, but it is cut off in the original paper as well.

The core result of the original paper is shown in the graph above. The x-axis of the plot is iterations.

In the 5, 10, and 15 qubit case, the reward function reaches around 0.95 and levels off after about 1200 iterations. In the 20 qubit case, the reward function levels off at around 0.8 In all cases, the reward function significantly increases in iterations.

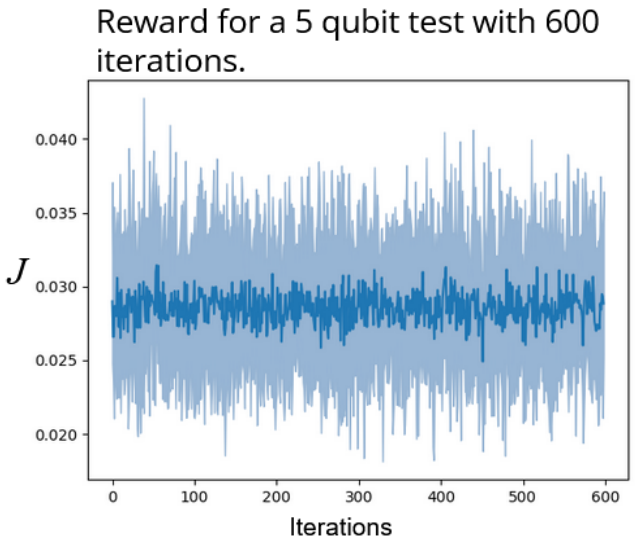

Figure 3. Reward function vs iterations graph produced by our software when attempting to replicate the results

Unfortunately, we did not notice any improvement in the reward function even after 600 iterations. We turned hyperparameters such as the learning rate, number of iterations, and how many averages we take, but we did not see any learning.

The paper creates a number of graphs with the key result of the paper being in figure 2. We wanted to fully reproduce that figure before moving on to other results. This could not

happen due to a number of reproducibility challenges which will be outlined in the following section.

5. Challenges reproducing the paper

While reproducing this paper, we ran into significant challenges which made it difficult to reproduce the results of the paper. Many of these issues relate to ambiguities or lack of detail on key equations/parameters in the paper.

The first and most relevant issue revolved around ambiguities in equations (2) and (3) which describe the reward function as well as the gradient of the reward function. The ambiguity of these equations stem from the function denoted by $\Sigma\pi(\theta|\mu,\Sigma)$. The most obvious interpretation of this term in the equations would refer to the probability density function described in equation 2. However, when we calculated the values of the probability density function, we found that the values were large leading to reward functions on the order of 10^12 which seemed incorrect. Another reasonable interpretation would be that the term simply signifies that the thetas used in $V(\theta)$ would be sampled from the gaussian. This is not standard notation, and the usage of the summation term is unusual. Unfortunately, this interpretation did not lead to any improvement in regards to reproducibility. The last possibility that we considered was that the term signified the averaging of various samples of theta. In this case, the notation is unclear as the amount of samples to average over is not described. We were unable to see any amount of improvement after trying various amounts of samples to average over.

The other large area we found ambiguous was how the author sampled their unitary matrices and states to run on the circuit. The paper mentions that matrices in SU(2^n) can be arbitrarily long, leading them to generate circuits with low depth to use as the unitary matrix. However, the actual unitary matrices or the method for generating the circuits was left unstated. In a similar manner, the states |k> are mentioned to be a “massively downsampled subensemble of orthogonal states" in the appendix, but the exact states or any way to generate the subensemble are not provided. There is a section of the paper which mentions that a training set for a later graph was created by applying various rotations, hadamard gates, and pauli operators, but it is unclear if this is the same set used in the figure above. Additionally, there is no mention of the number of gates used to create the states making it hard to replicate the training set.

The number of possibilities on how to approach the various ambiguities lead to around 30 different combinations of reasonable interpretations to test. The large amount of time required to train a model to test even one set of interpretations made it impossible to fully test every combination of them. Even with more time available, we are unsure if any of the combinations would lead to the same steps that the author took. The number of ambiguities lead to difficulties in reproducing the results.

Conclusion

The paper, Policy Gradient Approach to Compilation of Variational Quantum Circuits, used reinforcement learning methods to perform quantum compilation stating that the methods would mitigate the issue of barren plateaus in the parameter space. The paper was able to successfully perform quantum compilation as assessed by a reward function significantly increasing. However, we were unable to reproduce the results due to perceived ambiguity in core equations as well as ambiguities in methods to sample the set of states and unitary matrices tested.

References

[1] Herrera-Martí, D. A. (χ.χ.). *Policy Gradient Approach to Compilation of Variational Quantum Circuits*. doi:10.48550/ARXIV.2111.10227
[2] Arrasmith, A., Cerezo, M., Czarnik, P., Cincio, L., & Coles, P. J. (2021). Effect of barren plateaus on gradient-free optimization. *Quantum*, *5*, 558. https://doi.org/10.22331/q-2021-10-05-558
[3] Quantum Computing Stack Exchange. (n.d.). Retrieved April 8, 2022, from https://quantumcomputing.stackexchange.com/