Runtime estimation of VQA

Author: Hyejun Im (ohyeah92929@gmail.com)

$$T_{sim} = t_{Compiler} + \left(T_{VQA\_quantum} + t_{VQA\_classical}\right) \times n_{samples\_per\_iteration} \times n_{iteration} + t_{VQA\_final}$$

$$T_{VQA\_quantum} = \sum_{s=0}^{S} \left(t_{s\_qubit\_swap} + \max_{g} t_{sg} + t_{measurement} + t_{QEC\_detection} + t_{QEC\_repair} \right)$$

$t_{sg} = execution\ time\ of\ a\ gate\ g\ in\ stage\ s$

S = total number of quantum circuit stages

0. Before the simulation

Ideal cost function ($C(\theta)$), ansatz code, and initial parameters (θ etc) should be prepared prior to a VQA simulation. The minimum value of the cost function is the solution to VQA. Ansatz is a quantum operation that uses a minimalized parameter to represent states. Ansatz is usually consisted of rotation gates that can take parameters.

Coming up with proper initial parameters and ansatz can highly enhance the trainability of the simulation and ease some challenges regarding barren plateau.

This takes zero runtime, but it is too important not to mention.

1. $t_{Compiler}$

VQA will take a predefined ansatz and execute it in a quantum circuit. However, before quantum code gets executed, an environment that runs the code needs to be ready, and the code should be converted into a quantum readable pulse and initialize a quantum state. Quantum compiler does this work, trying to do the optimal conversion.

Compilers will decompose bigger gates into universal gates, optimize the given circuit considering hardware limitations and add quantum error correction. In the process of decomposition, the compiler will remove or cancel out all redundant gates. Also, the compiler will take the actual hardware into account to optimize. Quantum code can use an infinite number of qubits and order executions freely, but the actual hardware is limited. The compiler will map quantum code qubits into hardware qubits considering hardware limitations such as topology and maximum qubits. Topology is an important factor to consider because qubits need to be next to each other to execute gates that are bigger than 2-qubit gates. If two qubits are not next to each other but need to perform a 2-qubit gate, the circuit needs to swap qubit positions and this process takes time. QEC is an important part of quantum computing to increase fidelity. The more extra qubits for QEC, the higher fidelity. QEC is done at every stage of the circuit to prevent errors propagating to the result, and the compiler adds QEC to the quantum code provided by an engineer.

$t_{Compiler}$ takes polynomial time, so it might not be a big overhead in an overall quantum simulation time. However, I personally found quantum compiler interesting and worth mentioning as a fundamental part that not only VQA but all quantum simulations need to go through.

An interesting quantum compiler article I referred to: https://arxiv.org/pdf/2108.02099.pdf

2. $(T_{VQA\ quantum} + t_{VQA\ classical}) \times n_{Samples\ per\ iteration} \times n_{Iteration}$

This is the hybrid loop where the quantum part and classical part collaborate to solve the minimum value of a parameter θ.

2.1. $T_{VQA\ circuit}$

$T_{VQA\ circuit}$ is an execution time of the quantum part. I came up with the following equation to represent the quantum runtime. I focused on what should be done in each stage of the parameterized quantum circuit. Here, I assume 1) gate executions in the same stage, 2) qubit measurements in the same stage, 3) QEC repair in the same stage are executed in parallel.

$$T_{VQA\_quantum} = \sum_{s=0}^{S} \left(t_{s\_qubit\_swap} + \max_{g} t_{sg} + t_{measurement} + t_{QEC\_detection} + t_{QEC\_repair} \right)$$

$$t_{sg} = execution\ time\ of\ a\ gate\ g\ in\ stage\ s$$

S = total number of quantum circuit stages

2.1.1. $t_{s\_qubit\_swap}$

This is the total amount of time it takes to swap qubits in stage s. Two qubits need to be next to each other to perform two-qubit gate operations. If two target qubits are not neighbors but need to do two-qubit gate operations, changing qubits' positions become necessary.

2.1.2. $\max_{g} t_{sg}$

g is a gate in stage s and is an execution time of a gate g. I only consider the maximum gate execution time in stage s because all gates in stage s will be executed in parallel.

2.1.3. $t_{measurement}$

This is a qubit measurement execution time. For QEC purposes, this measurement is done to all qubits at every stage. This can be done in parallel, so only one $t_{measurement}$ is considered.

2.1.4. $t_{QEC\_detection}$

After qubits are measured, qubits need to be observed to detect any errors. This will detect any bit flip or phase flip errors.

2.1.5. $t_{QEC_{repair}}$

This is a time for QEC repair. If any error is detected from the previous stage, a qubit needs to be repaired and maintain the correct value throughout the process.

An interesting QEC article I referred to: https://www.nature.com/articles/ncomms7983.pdf

2.2. $t_{VQA\ classical}$

This is the classical part that optimizes parameters. There are many ways to optimize parameters and the most popular one is to use a gradient.

One thing to note is that the quantum part can also play a role to decide the parameters.

2.3. Iterations

The quantum part and classical part (a hybrid loop) are iterated $n_{Samples\ per\ iteration} \times n_{Iteration}$ times. This is related to iteration concepts in VQA.

3. t_{VQA_final}

This is the final stage of VQA that sends the result and analyzing it. This runtime will depend on what type of mathematical analysis an engineer is doing.

4. Postmortem

I chose the runtime estimation task because this will allow me to learn more about quantum hardware which I was always curious about. I didn't know anything about quantum hardware and VQA when I started this task, but now I know a little bit better about those topics. I think I gained good broad and shallow knowledge. Now I can go into more depth about specific topics that I'm interested in.

Thanks a lot for the interesting task – it was tough but this really helped me learn a lot. I strongly hope to be part of the mentorship program so that I can accelerate my learning in-depth.

5. Future to-do

- Learn more about quantum hardware and the technology.