ECE 382 V - Introduction to Quantum Computing Systems
HW#1
Due: 15th September 2023 5:00 PM

1. [**Error model**] The impact of errors can be analytically modeled by using various parameters accounting for the failure probabilities corresponding to different types of errors. Let us estimate the probability of successfully executing a 3-qubit bell-pair circuit.
    a. Estimate the error-rate assuming only gate and measurement errors occurs
    b. Now append the error-model to include decoherence. How does the probability of successful circuit execution change?
    c. How can you further improve the accuracy of this error model? Estimate the probability of success by including at least one additional parameter.
    d. What happens to the complexity of the error-model when you increase the circuit size beyond three qubits?

    To analyze the impact of each source of error and better understand their impact on application fidelity, you will plug in realistic values in your model for the error-rates by assuming the `FakeWashingtonV2` *Architecture available on Qiskit. Get the error rates using Qiskit. Refer to the documentation if you need to.*

2. [**System calibrations**] Quantum systems are frequently calibrated to enable high fidelity quantum gate and measurement operations.
    a. Why is system calibration non-trivial?
    b. What are the trade-offs involved in too frequent system calibrations and infrequent system calibrations?
    c. Most device providers opt for localized recalibrations as opposed to full-system calibrations. Why? How does this impact the "performance" of the system?
    d. In the class, we discussed the snake optimizer routine for performing large-scale system calibrations. What are the potential drawbacks of this technique? How do you think this can be improved?

3. [**Compilation: Qubit Mapping And Routing**] Compilers translate a sequence of high-level instructions (program) into a functionally equivalent sequence of low-level native gates (assembly). Qubit mapping and routing are two fundamental steps involved in this process.
    a. How does the quality of initial qubit mapping or allocation impact the performance of the quantum computer for a given application?
    b. How does the routing policy impact the quality of the solutions for a given application?
    c. There exists numerous routing policies in the compilation space. Why? Which would you like to opt for in order to run any given program of your choice?
    d. What are the trade-offs involved in routing overheads and device topology?

4. [**Pulse compilation**] Most quantum algorithms can be described with circuit operations alone. When we need more control over the low-level implementation of our program, we can use *pulse gates*. Pulse gates remove the constraint of executing circuits with basis gates only, and also allow you to override the default implementation of any basis gate. Programmers can leverage this property to directly compile their programs into a series of pulses. Often these pulses are more compact compared to the pulses generated using only the basis gate set. *For example, a SWAP pulse is typically shorter than the pulse generated by concatenating the CNOT pulse thrice.*
    a. What are the potential advantages of this approach?
    b. What are the key limitations in this approach?
    c. How do you think a programmer can best exploit this trade-off to their advantage?
    d. Can you transfer the pulse schedule from one machine to another?