INTRODUCTION TO PRACITICAL QUANTUM COMPUTING

QUANTUM COMPUTING INITIATIVE

Abstract. Many introductions to quantum computing get drowned in mathematical fornmalisms or discussions of theorhetical phenomena, that is why we aim to share this as a reference document explaining the foundations of quantum computing from a practical standpoint, focusing on abstracting the excess theory to focus on the necessary details for its applications. We hope to demystify quantum computing to make it more intuitive and therefore more accessible. While the underlying phenomena and formalisms are necessary for success in a more serious study of the subject, we restrict the content of this document to an intuitive overview sufficient for a layman's success.

The content covered in this document assumes the **qiskit** library is used for all quantum circuits. Its documentation can be found here: `https://quantum.cloud.ibm.com/docs/en/guides`

1. Qubits

1.1. What Are Qubits?.

Everyday computers represent information on a hardware level using bits with values of either 0s or 1s, however **qubits** are quantum mechanical systems used to represent information, and introduce special properties and phenomena due to the physics governing their systems. Fortunately, the technology is mature and abstracted enough to where performing computations on qubits does not require a thorough understanding of the physics behind it anymore.

1.2. Superposition.

One of the main reasons qubits are unique is due to their property to be in a **superposition** of the 2 states. This means the qubit's state can be represented as a probabilistic combination of being either 0 or 1.

For example a coin can be heads or tails, but while it's flipping in the air, we can describe its state as being a 50% chance of being a 0, and a 50% of it being a 1. Similarly, a qubit in superposition can have its state as being a 50% chance of being a 0, and a 50% of it being a 1.

Later we will introduce other transformations on the state of the qubit that allow it to be in a superposition with different probabilities, for example 25% it's a 0, and 75% it's a 1.

1.3. Entanglement.

Another property of qubits is **entanglement**. This is simply when the state of a qubit depends on the state of the other.

2. Gates

2.1. What Are Quantum Gates.

For those familiar with digital logic, **quantum gates** are analogous to logical gates that act on bits. They are devices that change the state of a qubit in a known way. Many quantum gates exist today but the nuance in quantum gates comes from combining them to create more profound logic.

2.2. Common Quantum Gates.

One of the most important gates is the **Hadamard** gate, which applies superposition to a qubit, and is implemented as `qc.H(<qubit_index>)` in Qiskit.

Some other common gates include the **X, Y, Z rotation** gates, respectively acccessed as `qc.X(<qubit_index>)`, `qc.Y(<qubit_index>)`, `qc.Z(<qubit_index>)`. There also exists a parameterized version of these gates which rotates the state of the qubit based on a numerical parameter. This is useful for encoding numerical data into a quantum circuit which will be mentioned later. This form of encoding data through a parameterized rotation gate is called rotation encoding.

2.3. Multi-Qubit Gates.

Another important concept is **multi-qubit gates**, which as the name implies, features gates that operate on multiple qubits. A common example is the **CNOT** gate, which flips one qubit based on the state another qubit. Similar versions exist for the rotation gates and other one qubit gates, where the state of one qubit controls whether or not another qubit is transformed.

These are important for introducing relationships between qubits to potentially mimic relations in the original data. This is beneficial for use cases like quantum machin learning, where the quantum circuit wants to learn the relations in the data.

3. Quantum Circuits

After developing an intuition for qubits and gates, quantum circuits become an obvious next step. The structure of the typical quantum circuit involves preparing the qubits, whether it's a specific quantum state that is required, or encoding classical data into the quantum circuit.

3.1. Encoding Classical Data.

For most applications, the data is classical, so encoding techniques will be necessary, notably rotation encoding or amplitude encoding. Rotation encoding is the common go-to technique due to its simplicity, scaling, and general utility. Rotation encoding involves rotating a qubit based on the numerical value in the data. Since the rotation angles are between 0 and 2π, it is useful to scale the data to fit this range before encoding it.

4. Hybird Algorithms

5. Introduction to Quantum Machine Learning