Overview and Comparison of Gate Level Quantum Software

Ryan LaRose
Quantum Journal
3/25/2019
Dylan Miracle
ICS 698-02
Spring 2021
Mar 2, 2021
Dr. Jigang Liu

Overview and Comparison of Gate Level Quantum Software

Dylan Miracle
Department of Computer Science
Metropolitan State University
St. Paul, Minnesota, USA
dylan.miracle@my.metrostate.edu

March 2, 2021

1 Background

One way to define a quantum computing program is as a series of unitary transforms on a qubit. Unitary transforms on qubits are often referred to as gates. A series of gates operating on qubits is called a quantum circuit. Most quantum algorithms currently developed are developed using the gate model and quantum circuits.

2 Main idea/conclusion

Several options are available to programming quantum circuits. This paper offers and overview of four popular quantum computing frameworks: Forest, Qiskit, ProjectQ, and Quantum Development Kit (QDK). These represent a cross section of the industry as each is primarily developed by a different company including Rigetti, IBM and Microsoft. This paper compares frameworks based on many features including ease of use, access to simulators and availability of hardware. It is established that the python based languages are typically the easiest to get started with. However if a developer already has experience with C# the Quantum Development Kit may be an easier place to start.

3 Support facts/algorithms/methods

We are shown how to create a random coin-flip algorithm in the various programming languages so we have a comparison of the syntax of each. The projects that use an intermediate representation show those as well. This demonstrates how a high level program is compiled into gate level code.

Each framework is broken down by what methods and gates are available out of the box (built in methods), as supported libraries, as examples in documentation, or as a tutorial. From this we can see that Qiskit and QDK are the most mature projects.

4 Arguments/disagreements/concerns

This is a quickly growing field and many of the findings of this paper, especially in regard to what algorithms are available on different platforms are likely to change. Since the writing of this paper it is likely that many new samples have been created within the frameworks as well as in third party libraries.

One of the most useful features of any quantum frameworks for those learning quantum computing is visual representation of data. This paper does not cover the visual representation of data from any of the frameworks. It would be interesting to compare the tools provided by each framework for drawing circuits and plotting result data.

5 Interesting findings

It is interesting to note that the python based languages Forest and Qiskit both break the python code down into a simple flat code block of a simpler gate code. The code for Forest is represented in a code called Quil, and the Qiskit code is shown in QASM.

All the projects reviewed have simulators associated however only some have simulators available to users on their local machines. The Forest simulator is only accessible through cloud resources, Qiskit comes with several simulators, ProjectQ has the fastest local simulator, and QDK simulators were not tested.

6 Quotations

A critical mass of effort from researchers in industry and academia alike has produced small quantum devices that operate on the circuit model of quantum computing.

For those who may be still undecided, we offer the following subjective suggestions. As Python is generally an easier language to pick up than C-style languages, either Forest, Qiskit, or ProjectQ may be more appropriate for beginners. For those with experience in C#, the Quantum Development Kit may be easier to pick up. Forest, Qiskit, and the QDK all contain good resources for learning about quantum computing. To test algorithms on real quantum computers, Forest and Qiskit are the obvious choices. ProjectQ is great for simulating algorithms on a large number of qubits.