The Scientific Temper (2023) Vol. 14 (1): 93-99

E-ISSN: 2231-6396, ISSN: 0976-8653

Doi: 10.58414/SCIENTIFICTEMPER.2023.14.1.11

https://scientifictemper.com/

RESEARCH ARTICLE

Quantum programming: Working with IBM'S qiskit tool

Kumari Neha[1], Amrita[2]

Abstract

One of the greatest technological advancements in the last century lies in digital computer science. The idea of storing information and performing complicated calculations with the help of bits, i.e., 0 and 1. But due to a sudden surge in data, the classical computer system has been becoming weak in data processing. Quantum computers offer promising substantial speedup over classical computers for many applications. Quantum chip fabrication has made remarkable gains in recent years, with the number of qubits and fidelity growing. In general computing, a binary digit is the smallest unit of information or a bit. "In quantum computing, the term Qubit (Quantum Bit) serves the exact function of the term bit." IBM Research released the IBMQ Experience in 2018, the first quantum computer that anyone can use and make accessible to a huge audience of different countries through cloud access. IBM also introduced the tool QISKIT (Quantum information software kit), which enables teachers, researchers, and developers to write coding and run their coding on quantum machines. It also includes different packages of quantum computing. In this paper, the author has discussed different steps to install qiskit. Mainly this paper focused on the "programming and application side of quantum computing." Qiskit tools used in the python programming language. The "quantum circuits" are fabricated with the use of quantum gates and favorable algorithms with less execution time.

Keywords: Quantum computers, Qubit, Qiskit, Algorithm, Python programming language, Quantum circuit.

Introduction

Quantum computers offer significant speedups over conventional computers in a wide range of practical applications, including "quantum chemistry, optimization, machine learning, cryptography, quantum simulation, systems of linear equations, and many more." While formerly regarded as "future fantasies" that only energized the educational world, current achievements prominent to the first practical "quantum computers" that everyone can use make this issue increasingly important for the interested mainstream. "IBM Research," which debuted the IBMQ Understanding in 2017, is a driving factor in this evolution (Quantum Computing | IBM Research, 2001).

[1]Department of Physics, Patna University, Munger, Bihar, India

[2]Department of Physics, Patna Women's College, Patna University, Patna, Bihar, India

***Corresponding Author:** Kumari Neha, Department of Physics, Patna University, Munger, Bihar, India, E-Mail: singh.neha624@gmail.com

How to cite this article: Neha K, Amrita. (2023). Quantum programming: Working with IBM'S qiskit tool. The Scientific Temper, **14**(1):93-99

Doi: 10.58414/SCIENTIFICTEMPER.2023.14.1.11

Source of support: Nil

Conflict of interest: None.

This is the opening manufacturing effort to building comprehensive quantum processors and making them achievable to large researchers and learners via cloud access. While the initiative began in March 2017 with the use of the 5-qubit quantum processor IBMQ X2, it now has four machines ready, with more under development (Qiskit Tutorials, IBM QX backend information). As of now, more than 100,000 people have utilized IBMQ machines, resulting in more than 6 million experiments and hundreds of scholarly articles. Furthermore, a global network of Wealth 500 firms, research organizations, and more than 100,000 users have used IBM Q machines as of now, resulting in over 6.5 million experiments and over 100 scientific papers (Quantum Computing | IBM Research, 2001). Furthermore, a global network of Fortune 550 companies, research centers, and entrepreneurs collaborate to advance quantum computing. IBM provides cutting-edge simulators for performing quantum algorithms on conventional processors in addition to real-hardware devices. To write matching programs and do tests on such quantum computers, IBM built an open-source platform termed Qiskit aimed at investigators, researchers, students, developers, and generally enthusiastic. This is an excellent play area for the design automation community (Nielsen and Chuang, 2002). Automated techniques and experiences may provide ideal solutions to many challenges in the realm of quantum computing. Too little communication is happening

Received: 10/01/2023 **Accepted:** 03/02/2023 **Published** : 25/03/2023

between the quantum sectors and design automation at the moment. As a result, many preexisting automated methods focused on false issues or were unsuccessful in reaching the intended audience. Qiskit is an idealistic hub for connecting various groups so that they may leverage one another. This summary is meant to inspire confidence in that possibility by introducing Qiskit and showcasing a few representative case studies of effective collaboration and development with and around it. The information provided is meant to serve as a jumping-off point for the curious but uninitiated reader (Quantum Computing | IBM Research, 2001). This, together with the included links to publicly available implementations of Qiskit extensions and supplementary readings in the form of tutorials and scientific articles, should provide the reader with all the tools necessary to write and execute their programs on a genuine quantum computer.

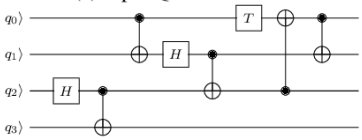
Quantum Computation

When compared to classical models of computation, quantum computing is radically different. In conventional computations and circuits, bits are the basic data storage and transfer unit. Instead, qubits are used to do computations in quantum circuits. All n qubits' 2n-basis-state configurations may be represented simultaneously, as these qubits can be in either the |0> or |1> basis state or in a superposition of the two (Zulehner, 2017). Algorithms built on quantum computers can execute orders of magnitude faster than those built for classical computers due to quantum correlations, including entanglement and quantum interference effects (Haner, et al. 2016). To that purpose, "quantum operations represented by quantum gates" are used to modify the qubits of the quantum circuit. These operations can be performed on a single or several qubits. We discriminate between target and control qubits in multi-qubit gates. When the qubits under control are reset to their initial state, the value of the qubits under attack changes as a result. As a universal set of "quantum operations," the Clifford+T library consists of the solitary gates (Nielsen, 2000, Niemann, et al. 2018): Hadamard gate "(H gate) and T (Phase shift by1/4)" and the two-qubit controlled-NOT gate (CNOT gate), i.e., all quantum computations can be done by a circuit composed of different gates from this library. To define quantum circuits, one can use either a high-level quantum language (like Scaffold or Quippier) or a quantum assembly language (like IBM's Open QASM 2.0), or a circuit diagram Quantum bits (qubits) are depicted as horizontal lines in a circuit diagram, and they go through quantum gates. This determines the sequence wherein quantum gateways are delivered to qubits and the nature of the connections between the gates, in contrast to traditional circuits (from left to right).

Example 1: Figure 1 shows an as well as schematic representation; the quantum circuit described in Open QASM (a) may be represented as (b).

```
OPENQASM 2.0;
include "qelib1.inc";
qreg q[4];
h q[2];
cx q[2],q[3];
cx q[0],q[1];
h q[1];
cx q[1],q[2];
t q[0];
cx q[2],q[0];
cx q[0],q[1];
```

(a) OpenQASM code



(b) Circuit diagram

Figure 1: Representation of a quantum circuit

IBMQ X and Corresponding Q X Architectures

IBM's Quantum Experience (IBMQ X) is a web-based service that enables the creation of quantum programs in Open QASM or using a graphical interface, followed by testing on IBMQ hardware or traditional quantum hardware emulations (IBM QX backend information). IBMQ is made up of superconducting transman qubits on silicon devices. Microwave pulses are used for control and measurement

transferred into and out of concentration refrigerators, where the substantial pieces are kept at a high temperature of roughly 15 m K. On-chip resonators are used for communication into, out of, and among the qubits (Steiger, 2018, Saeedi, et al. 2011).

The Qiskit Toolset

Qiskit is a free and free and open-source quantum computing software library providing support for everything from hardware emulation and simulation to full-fledged applications. Each of the four libraries in this building is named after one of the classical elements, earth, water, air, and fire, which reflect this organization. Every collection is briefly explored in the sections that follow (Q. 2023, Forest SDK Rigetti. [Online], Yamashita, and Markov, 2010).

Terra: All of Qiskit's low-level codes are located in the Terra library. Open QASM language-based tools for modeling and manipulating quantum circuits and tools for manipulating pulses using Open Pulse are among them. It provides trans pilers to improve the performance of quantum circuits on actual hardware, for example, by decreasing the number of CNOT gates. By doing so, the appropriator can able to construct a circuit that captures the required functionality without making significant optimizations for the hardware compatibility and then allow the transpolar to identify a more streamlined circuit while maintaining the precise functionality specified by the user (Abhari, et al. 2012, Zulehner, et al. 2018). Terra also includes the framework necessary to describe and model physical

noise processes. These are very useful for investigating the behavior of quantum algorithms in the presence of noise, as is the situation with current technology. Finally, Terra supplies the interfaces and data structures necessary to define and pass various quantum computing software constructs between both the different hardware and qiskit libraries.

Aqua: On the opposite end of the spectrum, the Aqua library supports advanced quantum algorithms for a huge range of utilities. To use quantum computing tools and simulators, users need not have to go into the nitty-gritty of building quantum circuits thanks to the availability of high-level user interfaces (Haner et al. 2016). Aqua creates the real quantum circuits utilizing Terra constructions once the user supplies the application structure and specifications. As a result of using classical processes for both the design and implementation of using the quantum circuit, each program becomes a mix of classical and quantum logic. Contrarily, the Aqua framework implements advanced quantum algorithms with a wide range of applications. Comprehensive, high-quality, user-focused APIs to utilize for the purpose of utilizing quantum equipment and simulators without learning the finer points of designing quantum circuits(Singh and Aarthi, 2021). Aqua creates the real quantum circuits utilizing Terra constructions once the user supplies the application structure and specifications. In addition, conventional flows are used in the design and execution of the quantum circuit, making each application a hybrid algorithm that combines classical and quantum computing features. Many of Aqua's uses depend on the Variational Quantum Eigen solver (VQE) technique. Either the user (by, say, identifying the optimization technique to be used by the algorithm) or Aqua itself can adjust the parameters of this algorithm. This means that Aqua can accommodate any need, from basic one-button apps to highly individualized solutions (Wille, et al. 2019).

Aer: Aer is anticipated to consist of a suite of emulators and imitators for deploying quantum circuits and functions on traditional hardware. There are many possible applications for this. For the learning of quantum circuits and algorithms, without ha to without any need for rarer quantum hardware, the Aer simulator will be a valuable educational tool that will also let scientists experiment with quantum equipment in safe environments by doing things like injecting different sound processes into networks and observing the effects. Last but not least, it will facilitate the rapid construction of quantum computation by providing an extremely practical means of testing preliminary versions of such programs on a normal computer. These techniques can be run on "cleaner" (noiseless) simulators to check their predictions and probe the design space (Zhang, et al. 2019). After that, the algorithms might be put through their paces on noisy simulators to examine the impact of actual noise on their performance.

Figure 2: Hadamard Gate

Ignis: Ignis will be a library that contains all the concepts and algorithms of methods used to characterize, verify, mitigate, and rectify quantum hardware. Methods for carefully classifying and characterizing noise processes in hardware include randomized benchmarking and multi-modality tomography simulated evaluations. It will also incorporate pulse approaches to alleviate recurring entrance difficulties, also a repertoire of error-correcting algorithms and codes (Aleksandrowicz, et al. 2019). As an added complication, the Ignis library has not been liberated to the public so far.

Because of their shared data structures and tight integration, the "four Qiskit libraries" form the most extensive back-to-back technology solutions for quantum computing (qiskit.org, n.d.). The "Qiskit software library" is accompanied by a comprehensive instructional collection covering topics as diverse as quantum theory and the creation of low-level notebook-assisted specialized quantum circuits (Grover, 1996).

Basic Quantum Gates

According to the "quantum circuit model of computation," a quantum gate (or a quantum logic gate) is the basis of a quantum circuit that acts on a finite number of qubits. It serves as the foundation for quantum circuits in the same way that classical logic gates serve as the foundation for ordinary digital circuits (Senekane, et al. 2021).

In contrast to their conventional counterparts, quantum logic gates can operate in both directions. For true classical computing, only reversible gates should be employed. Using auxiliary bits is sometimes necessary, but the reversible Toffoli gate can realize any Boolean function. In this way, the Toffoli gate demonstrates that any operations that conventional circuits can perform may also be performed by their quantum counterparts (Zulehner, et al. 2018).

A single unified matrix concerning some basis is used to represent nuclear gates, which are unitary operators. D-level quantum systems (such as a qubit, a quantum register, or qutrits and qudits) have orthogonal basis vectors that are often named using the computational basis (or written in binary notation) |0>, |1>,.....,|d-1>002E (Cheung, et al. 2007). Some basic quantum gates are explained below

Hadamard Gate

"Figure 2 depicts the Hadamard Gate. Hadamard gate, or H gate, is one of the most commonly used quantum gates. This gate can be used to convert the qubit from a clustering state to a uniform superposed state (Smith, and Thornton, 2019)."

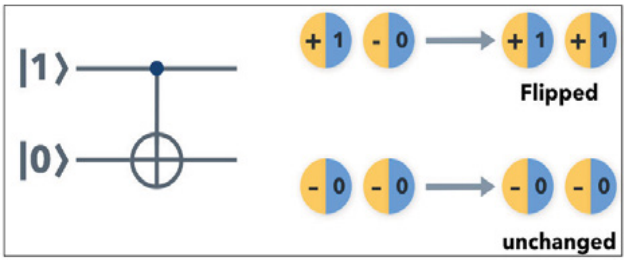


Figure 3: CNOT gate

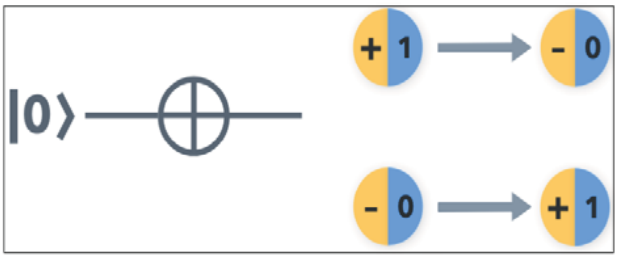


Figure 4: Pauli-X Gate

CNOT Gate

"The CNOT (Controlled-NOT) Gate is shown in Figure 3. Controlled Not Gate (denoted by cx in qiskit) is a two-qubit operation gate. In this gate, the first qubit is usually referred to as the control qubit and the second qubit as the target qubit (Linke, et al. 2017)."

Pauli – X Gate

"The Pauli-X Gate, as shown in Figure 4, accepts a single qubit. It operates a single bit. The Pauli-X Gate flips a tiny amount, negating the Qubits. Both a Ket |0> to Ket |1> and a Ket |1> to Ket |0> are negated. The function of the Pauli-X gate is to rotate single-qubit through π radians across the x-axis (Cross, et al. 2017)."

Programming language and basic qiskit programming

Though they can be created on conventional computers and teleported to quantum ones, quantum programs run on quantum computers. The Quantum Turing Machine may be used to identify quantum logic gates like Hadamard, CNOT, and Pauli-X through the computers where these Gates are employed and sent (Shor, 1999, Simon, 1997). Because it must live with a classical computer situated far from the internet, a quantum computer can run any software created for a desktop or laptop connected to the internet, but it cannot guarantee faster performance. Due to its memory constraints, a conventional computer might not be able to finish big computations like finding the reverse of huge matrixes or figuring out the prime factors of a number with hundreds of digits (Niemann, et al. 2015, Zulehner, et al. 2019).

The paradigm of classical computing is fundamentally different from the quantum computing paradigm. Bits are the standard data unit in modern computers and circuits. Quantum circuits, on the other hand, use qubits to carry out computations (Niemann, et al. 2014). These qubits can simultaneously represent all conceivable 2n basis states of n qubits since they can be in a superposition of both the |0> and |1> basis states. Quantum connections like quantum interference and entanglement are the foundation for algorithms that run much more rapidly on quantum computers compared to traditional machines. This is called "quantum parallelism (Highlights of the IBM Quantum Summit 2022, 2022)."

Qiskit Programming

IBM's QISKit is a quantum information science kit that Open-quantum QASM programs can feed. The input programmers are formatted and optimized; either an internal QASM simulator or remote quantum processing resources are used to run the programs. Open-QASM is a QASM variation developed for the precise control of a physical system through the use of a parametrized "Gate" set (Black, et al. 2002).

Quantum teleportation of qubits: Before we go to the algorithm of teleportation of qubits, let us first discuss quantum teleportation.

"Quantum teleportation" is a way to send quantum information from a sender in one place to a receiver in another place far away. In science fiction, teleportation is often used to move physical objects (Burgholzer and Wille, 2020). However, quantum teleportation only moves quantum information. The sender doesn't have to know the exact quantum state being sent. Quantum teleportation may also not reveal where the recipient is, but classical information must still be sent from the sender to the receiver for it to work. Cloning or copying an unknowable quantum state is not feasible in quantum mechanics.

Nevertheless, the quantum state may be transferred through conventional transmission (Simon, 1997). The original state is destroyed, but the receiver reconstructs the original quantum state perfectly. Because there is only one state left at the end, the No-Cloning theorem is valid. It is possible to think of quantum teleportation as the opposite of dense coding: whereas dense coding converts a quantum state into two classical bits, teleportation converts two classical bits into a quantum state. Without entanglement, neither process would be feasible (Q# And the Quantum Development Kit | Microsoft Azure, n.d., Forest SDK, Rigetti. [Online])

Working with Qiskit

Anyone may quickly define and carry out required quantum computations using Qiskit, from the cloud to the quantum computer itself. For the achievement of this, appropriate selection tools and techniques are offered, such as mapping relevant architectures or simulations. In the paragraphs that

follow, we provide a brief overview of the tool's installation and initial use to demonstrate our point (Saeedi, et al. 2011). By doing so, we offer a quick insight into the viewpoint of Qiskit's users. Qiskit could be downloaded using the links to the respective supplied at https://qiskit.org.

First of all, download Anaconda3 (Python 3.9.12 64-bits) and install it, after installation one should open the anaconda prompt when the terminal window opens; the following steps are to follow:

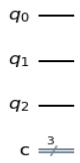
>pip install qiskit

It will take some time for the installation. After the installation of Qiskit, we are ready to use qiskit. Now, in the terminal window type,

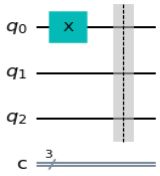
>jupyter notebook

It opens a new browser window, then goes to the new tab and clicks on the Python 3 tabs; it will open a new worksheet to write Qiskit programming in python-based language. Now we have to do confirmation of qiskit; for this, we should write some commands on the worksheet, which are – Import qiskit and press shift + enter; after this command, we are able to see the installed qiskit version which we are going to use for our programming. Now we are ready to start writing the code or algorithm; we have done the quantum teleportation algorithm for the three-qubit system in qiskit (Viamontes, et al. 2007). The algorithm is written as follows:

```
from qiskit import*
circuit=QuantumCircuit(3,3)
%matplotlib inlinecircuit.
draw(output='mpl')
```

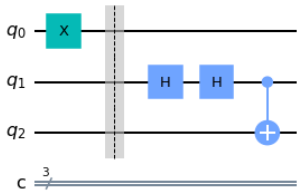



```
circuit.x(0)circuit.barrier()
circuit.draw(output='mpl')
```



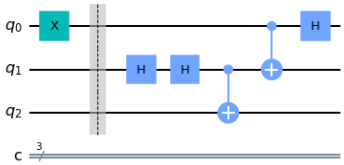
Circuit diagram

```
circuit.h(1)
circuit.cx(1,2)
circuit.draw(output='mpl')
```



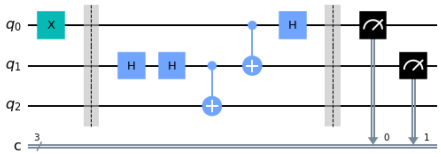
Circuit diagram

```
circuit.cx(0,1)
circuit.h(0)
circuit.draw(output='mpl')
```



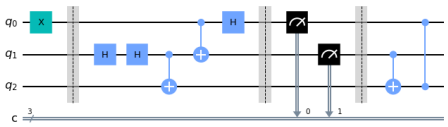
Circuit diagram

```
circuit.barrier()
circuit.measure([0,1], [0,1])
circuit.draw(output='mpl')
```



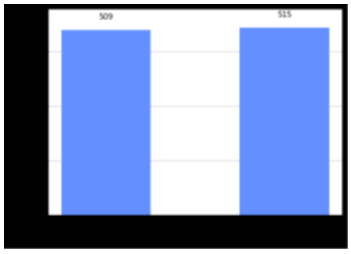
Circuit diagram

```
circuit.barrier()
circuit.cx(1,2)
circuit.cz(0,2)
circuit.draw(output='mpl')
```



Circuit diagram

```
circuit.measure(2,2)
simulator= Aer.get_backend('qasm_simulator')
result=execute(circuit,backend=simulator,shots=1024).
result()
counts=result.get_counts()
from qiskit.tools.visualization import plot_histogram
plot_histogram(counts)
```



Histogram of the Probability

Observation

In the past few years, interest in quantum computers has grown a lot. This is because real quantum computers are getting closer to becoming possible, and more and more algorithms for possible applications are being found. Quantum programming is the process of set-up together different instructions that can be run on a quantum computer. These sets of instructions are called quantum circuits. Quantum programming languages help quantum algorithms be written in a high-level way. IBMQ introduced an open-source programming software called QISKIT (Quantum information software kit), It is a software development kit (SDK) for working with circuits, pulses, and algorithms on quantum computers. It provides researchers, intellectuals, and beginners with a useful tool for making and changing quantum programmes and running them on ideal quantum devices on IBM Quantum Experience or on simulators on a classical computer. In the present scenario, quantum computers may be used to translate immeasurable industrial data. Also, with the help of quantum computing, more accurate weather forecasts would be possible and potentially save thousands of life from any natural disaster.

Discussion

Implementing quantum algorithms and quantum programming was a little challenging since the layouts of quantum processors were not completely disclosed. Consequently, it is safer to automate the process of customizing and improving quantum programmes from the quantum machine compiler in accordance with any given configuration. The IBM quantum computer has been accessed, and a programme created on our machine successfully ran for the IBM quantum experience. Quantum Circuits are developed successfully quicker in Python in several test situations. Nobody wants to miss out on the future of quantum computing, which has already ignited a sort of global arms competition. Countries all across the globe are making significant investments in subjects like decryption, genetics, astronomy, artificial intelligence, materials science, chemistry, quantum physics, and much more because these fields have the potential to benefit humanity as a whole. Even though having a quantum computer has numerous constraints and drawbacks, it still has advantages over traditional computers. Some of our most advanced cryptography may soon be rendered worthless by quantum machines, allowing for secure data transit without the need for public key encryption.

Conclusion

In this paper, the author discusses the installation method of IBM's qiskit tool with the open source and different quantum gates and their functions. We proposed a basic quantum teleportation algorithm of qubits in the python programming language. Because of the emerging trend of quantum computing through the platforms available in the cloud, we have discussed here, particularly the packages offered by IBM. It is having two different ways; one is the IBM site directly. Another one is the Qiskit tool which is an open-source and very versatile tool for a proper understanding of different circuits and the execution of the different aspects of different theories.

Future Scopes

Everyone is racing to get into the quantum computing future because no one wants to be left behind. As a result of the hope that new inventions would improve people's lives, governments all over the globe are pouring resources into areas including pharmaceuticals, cryptography, genetics, astronomy, AI, materials science, chemistry, quantum physics, and many more. Even though there are currently significant barriers to entry when it comes to building a quantum computer, the benefits of quantum supremacy may be realized without having to completely abandon conventional computers. Some of our most advanced cryptography may soon be rendered worthless by quantum computers, rendering ordinary public key encryption obsolete for secure data transmission.

Acknowledgment

I would like to express my sincere thanks and gratitude to my Guide for letting me work on this paper. I am very grateful to her for their support and guidance in completing this paper. I am thankful to my parents as well. I was able to successfully complete this paper with the help of their guidance and support. Finally, I want to thank all my dear friends as well.

Conflict of interest

The authors declare no potential conflicts of interest with respect to research, authorship and/or publication of this article.

References

Abhari, A. J., Faruque, A., Dousti, M. J., Svec, L., Catu, O., Chakrabati, A., ... & Chong, F. (2012). Scaffold: Quantum programming language. Princeton Univ NJ Dept of Computer Science.A. Zulehner, P. Niemann, R. Drechsler and R. Wille, "Accuracy and compactness in decision diagrams for quantum

computation", Design Automation and Test in Europe, 2019.
Aleksandrowicz, G., Alexander, T., Barkoutsos, P., Bello, L., Ben-Haim, Y., Bucher, D., ... & Marques, M. (2019). Qiskit: An open-source framework for quantum computing. Accessed on: Mar, 16.
Black, P. E., Kuhn, D. R., & Williams, C. J. (2002). Quantum computing and communication. In Advances in Computers (Vol. 56, pp. 189-244). Elsevier.
Burgholzer, L., & Wille, R. (2020). Advanced equivalence checking for quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 40(9), 1810-1824.
Cheung, D., Maslov, D., & Severini, S. (2007, December). Translation techniques between quantum circuit architectures. In Workshop on quantum information processing.
Cross, A. W., Bishop, L. S., Smolin, J. A., & Gambetta, J. M. (2017). Open quantum assembly language. arXiv preprint arXiv:1707.03429.
Forest SDK, Rigetti. [Online]. Available: https://rigetti.com/ forest (visited on 04/10/2020).
Grover, L. K. (1996, July). A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (pp. 212-219).
Haner, T., Steiger, D. S., Smelyanskiy, M., & Troyer, M. (2016, November). High performance emulation of quantum circuits. In SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 866-874). IEEE.
Highlights of the IBM Quantum Summit 2022. (2022). IBM Quantum Computing. Retrieved March 24, 2022, from https://www.ibm.com/quantum?utm_content=SRCWW&p1=Search&p4=43700071568483278&p5=p&gclid=Cj0KCQiAn4SeBhCwARIsANeF9DJ-eW4QN7vQYGrBF0NWGbTDlZpDyApw_kFTVBt9_Y9cpZgaO1RFIw4aAiPyEALw_wcB&gclsrc=aw.ds
IBM QX backend information. https://github.com/QISKit/ ibmqx-backend-information.
Linke, N. M., Maslov, D., Roetteler, M., Debnath, S., Figgatt, C., Landsman, K. A., ... & Monroe, C. (2017). Experimental comparison of two quantum computing architectures. Proceedings of the National Academy of Sciences, 114(13), 3305-3310.
Nielsen, M. (2000). A & Chuang, IL Quantum Computation and Information.
Nielsen, M. A., & Chuang, I. (2002). Quantum computation and quantum information.
Niemann, P., Wille, R., & Drechsler, R. (2014, July). Equivalence checking in multi-level quantum systems. In International Conference on Reversible computation (pp. 201-215). Springer, Cham.
Niemann, P., Wille, R., & Drechsler, R. (2018, March). Improved synthesis of Clifford+ T quantum functionality. In 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE) (pp. 597-600). IEEE.
Niemann, P., Wille, R., Miller, D. M., Thornton, M. A., & Drechsler, R. (2015). QMDDs: Efficient quantum function representation and manipulation. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 35(1), 86-99.
Q# and the Quantum Development Kit | Microsoft Azure. (n.d.). Q# And the Quantum Development Kit | Microsoft Azure. Retrieved June 14, 2022, from https://azure.microsoft.com/en-us/resources/development-kit/quantum-computing/
Q. (2023, January 8). GitHub - quantumlib/Cirq: A python framework for creating, editing, and invoking Noisy Intermediate Scale Quantum (NISQ) circuits. GitHub. Retrieved June 10, 2022, from https://github.com/quantumlib/Cirq
Qiskit Tutorials. https://nbviewer.jupyter.org/github/Qiskit/ qiskit tutorial/blob/master/index.ipynb.
Qiskit.org. (n.d.). qiskit.org. Retrieved April 23, 2022. from https://qiskit.org/
Quantum Computing | IBM Research. (2001, August 21). IBM Research. Retrieved March 14, 2022, from https://research.ibm.com/quantum-computing
Saeedi, M., Wille, R., & Drechsler, R. (2011). Synthesis of quantum circuits for linear nearest neighbor architectures. Quantum Information Processing, 10(3), 355-377.
Senekane, M., Mafu, M., Maseli, M., & Taele, B. M. (2021, September). A quantum algorithm for single parity check code. In 2021 IEEE AFRICON (pp. 1-6). IEEE.
Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review, 41(2), 303-332.
Simon, D. R. (1997). On the power of quantum computation. SIAM journal on computing, 26(5), 1474-1483.
Singh, P. N., & Aarthi, S. (2021, February). Quantum circuits–an application in qiskit-python. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 661-667). IEEE.
Smith, K. N., & Thornton, M. A. (2019, June). A quantum computational compiler and design tool for technology-specific targets. In Proceedings of the 46th International Symposium on Computer Architecture (pp. 579-588).
Steiger, D. S., Haner, T., & Troyer, M. (2018). ProjectQ: an open source software framework for quantum computing. Quantum, 2, 49.
Viamontes, G. F., Markov, I. L., & Hayes, J. P. (2007, November). Checking equivalence of quantum circuits and states. In 2007 IEEE/ACM International Conference on Computer-Aided Design (pp. 69-74). IEEE.
Wille, R., Van Meter, R., & Naveh, Y. (2019, March). IBM's Qiskit tool chain: Working with and developing for real quantum computers. In 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE) (pp. 1234-1240). IEEE.
Yamashita, S., & Markov, I. L. (2010, June). Fast equivalence-checking for quantum circuits. In 2010 IEEE/ACM International Symposium on Nanoscale Architectures (pp. 23-28). IEEE.
Zhang, X., Xiang, H., & Xiang, T. (2019). An efficient quantum circuits optimizing scheme compared with qiskit (short paper). In International Conference on Collaborative Computing: Networking, Applications and Worksharing (pp. 467-476). Springer, Cham.
Zulehner, A., & Wille, R. (2017). One-pass design of reversible circuits: Combining embedding and synthesis for reversible logic. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(5), 996-1008.
Zulehner, A., Hillmich, S., & Wille, R. (2019, November). How to efficiently handle complex values? Implementing decision diagrams for quantum computing. In 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (pp. 1-7). IEEE.M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, 2010.
Zulehner, A., Paler, A., & Wille, R. (2018). An efficient methodology for mapping quantum circuits to the IBM QX architectures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(7), 1226-1236.