IBIMA Publishing
Communications of International Proceedings
https://ibimapublishing.com/p-articles/44AI/2024/4425824/
Vol. 2024 (22), Article ID 4425824

Circuit Compatibility for Retired Quantum Systems*

Marek WRÓBLEWSKI and Marek SAWERWAIN

Institute of Control & Computation Engineering
University of Zielona Góra,
Zielona Góra, Poland

Correspondence should be addressed to: Marek WRÓBLEWSKI, m.wroblewski@issi.uz.zgora.pl

* Presented at the 44th IBIMA International Conference, 27-28 November 2024 Granada, Spain

Abstract

The article presents the consequences and implications of the development of quantum computing technologies in the context of designing circuits based on quantum gates, which are mechanisms for artificial intelligence algorithms. The main motive behind this article is to indicate the possibility of adapting currently existing quantum applications and circuits created on their basis using the Qiskit framework to currently available IBM Q quantum platforms. The differences in code implementations are so significant that they should be explicitly indicated in the chapters below, so that it would be possible to adapt and develop existing algorithmic solutions in modern quantum platforms. When conducting the latest literature review, it is noted that there is no explicit indication of how to proceed in the case of performing circuit adaptation in order to further develop current quantum solutions and later run them in the IBM Q quantum environment. For this purpose, methodologies from the field of software engineering, including quantum engineering, and characteristics of the area of revers engineering were used, with the aim of developing a method enabling circuit adaptation for retired hardware quantum architectures. The dynamism of the development of quantum computer platforms necessitates the adaptation of originally programmed quantum artificial intelligence methods using quantum circuits to the currently available quantum hardware environment.

Keywords: recommendation mechanism, quantum computing, quantum circuit, implementation algorithm, machine learning, transpiling, circuit adaptation, backward compatibility

Introduction

The constant scientific development of the medical and industrial branches means that the tasks set for modern artificial intelligence algorithms require increasingly large hardware resources for data processing. This forces the continuous and unceasing development of classical and also quantum technologies, especially in the context of hybrid solutions, which means that modern quantum computing systems and platforms constantly increase the hardware resources of available qubits within separate architectures. Key changes and achieving milestones in the development of artificial intelligence technology based on quantum solutions, which does not only consist in increasing the number of available qubits but also the efficiency of quantum coprocessors themselves and optimizing built-in algorithms, e.g. transpiling, forces the introduction of large, diametrical changes in the form of new architectures, which means that the current ones are completely replaced by new ones. From a scientific and programming point of view, it is important to be able to run already developed solutions on new introduced

Cite this Article as: Marek WRÓBLEWSKI and Marek SAWERWAIN, Vol. 2024 (22) "Circuit Compatibility for Retired Quantum Systems " Communications of International Proceedings, Vol. 2024 (22), Article ID 4425824, https://doi.org/10.5171/2024.4425824

architectures. This is not always possible directly. In certain cases, it is necessary to adapt current codes and circuits in order to run on modern quantum platforms introduced as a replacement, and it is in this article that the author discusses and presents this issue of code and circuit adaptation. This is very important from the point of view of further development of currently developed computational solutions, including hybrid artificial intelligence solutions, both in the field of recommendation systems and profiling mechanisms, which is presented in this article.

The method of optimizing the Oracle part in Grover's algorithm, which is comparable in performance to the Unitary Gate method, was discussed by Sanchez-Rivero et all (2023), while the optimization using Karnaugh maps for the process of building a quantum circuit and the number of quantum gates used was described in detail and analyzed by Bae et all (2020). The result of such an application is reduced circuit complexity, which in certain cases increases the efficiency and throughput for the hardware quantum architecture implementing such a circuit. An important element of the algorithm implementation that speaks about its effectiveness is the final result. Schutski et all (2020) mentioned and presents a different method of evaluating the result from the simulation of quantum computations, which consists in using an algorithm based on interpolation between the evaluation of the full state and single probability amplitudes.

An important application of modern quantum computers is the example of cooperation with Daimler AG Mercedes-Benz presented by IBM concerning work on the development of modern batteries for electric vehicles based on lithium-sulfur technology, the use of which in electric vehicles will eliminate the current disadvantages such as low capacity and long charging time of currently used batteries. In order to model the dipole moment of three molecules containing lithium, a quantum computer available in the IBM Q platform was used by Garcia et all (2023). This shows the practical application of available quantum technology in industry 4.0 branches.

The organization of the article is as follows: in the part of 1.1 the basis of a synthetic introduction to quantum computing calculations is presented. Section 2 presents the genesis and effects and consequently the development of modern quantum systems caused by the withdrawal of outdated from the development perspective current quantum architectures. Section 3 discusses the method of code adaptation of quantum circuits in the context of enabling their circuit implementation in new available quantum architectures for the IBM Q platform. Section 4 presents an analysis of the obtained results of the performance of the final measurement operation of newly implemented circuits for currently available quantum platforms. The summary of the article is given in the section 5, where the nal conclusions and some comments about possible further work on the method described in this article were given. The article ends with a list of the cited bibliography.

Introduction to Quantum Preliminaries

In order to better understand the content presented in the article and the notation used, as well as the research carried out in the form of implementation and adaptation of quantum circuits for the IBM Q computing platform, it is necessary to present the basis for a synthetic introduction to quantum computing calculations.

When introducing the basic unit of quantum information, which is a qubit, it is necessary to present what the classical bit model is. In contrast to the qubit, the smallest unit of information used in classical computer science is the classical bit, which takes on two values 0 or 1, by means of which it describes the state of the system at a given time, as indicated by the equation (1).

$$b = 0 \quad or \quad b = 1. \tag{1}$$

Combining classical bits creates a classical register. However, it should be remembered that at one given moment of time the register can assume only one state of the indicated system, which is indicated in equation (2).

$$r = b_0 b_1 b_2 b_3 b_4 ... b_{15}, \tag{2}$$

Moving on to the quantum representation of information, it is necessary to present the concept and structure of a two-level quantum system constituting the basic unit of information in quantum computing, called a qubit. In reference to the classical representation of information, a qubit has two states $|0\rangle$ and $|1\rangle$, which can occur

simultaneously at the same time, which is called a superposition of two states presented in the books Nielsen et all (2000) and Bengston et all (2017). Using the Dirac notation (3), it is possible to define two basic base states, which correspond to the states for a classical bit representing information. The indicated base states constitute a standard computational basis.

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ . \tag{3}$$

The presented states are the basis states corresponding to the classical states of the information bit. The presented basis states are referred to as the standard computational basis. In turn, the linear superposition of two basis states for any given qubit state $|\psi\rangle$ is defined by:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \ , \tag{4}$$

where after transformation we get:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \quad \text{where} \ \ \alpha, \beta \in \mathbb{C} \ . \tag{5}$$

The values of α and β satisfy the necessary relations due to the requirement of normalization of probability amplitudes:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{6}$$

A system built of qubits that can be considered as a complex system is called a quantum register. The structure of such a system is described by the state of photons and atoms. To build a quantum register, the tensor product operation is performed, which for a register composed of n qubits looks as follows:

$$|\psi\rangle = |\psi_0\rangle \otimes |\psi_1\rangle \otimes \ldots \otimes |\psi_{n-1}\rangle \overset{\sqrt{X}}{\cdot} \qquad \mathbb{H} \tag{7}$$

Superposition of states in a quantum register enables the storage of an exponential number of combinations of bit systems, which makes it possible to store an exponential number of states of a classical register mentioned by Brylinski et all (2002). For a complex quantum circuit model, single- and multi-qubit quantum gates are the basic element and building block of such a circuit. The main task of such quantum gates is to transform a previously indicated quantum state into another state. This allows for performing elementary transformations and calculations for quantum algorithms, which also consist, among others, in implementing the mechanisms of action based on the matrix representation showed by Steeb et all (2018) and Venturelli et all (2018). It is not possible to use a direct analogy for quantum gates directly from all classical gates. This is due to the fundamental properties resulting from the linearity of quantum systems. Elementary single-qubit gates are the Pauli gates X, Y and Z, the Hadamard gate H and two phase change gates S and T, as well as the square root gate of NOT, i.e. , X and the gate P. Using the unitary operator U, one-qubit quantum gates implement any operation in the two-dimensional Hilbert space :

$$U : \mathbb{H}_2 \to \mathbb{H}_2. \tag{8}$$

The general form of the unitary matrix for a qubit M_{Qubit} is given by:

$$M_{\text{Qubit}} = e^{it} \begin{pmatrix} \alpha & \beta^* \\ \beta & -\alpha^* \end{pmatrix} \tag{9}$$

assuming that α, β ∈ C , including |α|2 + |β|2 = 1, and t ∈ R.

In the measured system, the execution of a measurement operation for a quantum register causes state disturbances, which may result in additional decoherence of states. The measurement of a quantum register is

performed as a result of the interaction of the external environment with the quantum system. The final result of the measurement operation is obtaining the measurement result along with the transformation of the measured system. The generalized measurement operation is described using sets of measurement operators $\{P_n\}$ is prezented by Steeb et all (2018), Venturelli et all (2018) and Brukner et all (2017). It should be remembered that the operators P_n operate in the state space of the measured system, and the state after the projector is applied directly refers to the result of the measurement operation, in which case for the system $|\psi\rangle$ the probability of obtaining the result for index n is expressed by:

Retired systems

$$P(n) = \langle \psi | P_n^\dagger P_n | \psi \rangle. \tag{10}$$

The large development of technology required fundamental architectural changes that were impossible to implement in previous systems. This resulted in the introduction and replacement by new, more efficient quantum systems. Focusing on the analysis of the development of the IBM Q quantum computing cloud, it is noted that previously used quantum systems were withdrawn and replaced by new ones, which is justified. Older quantum architectures were withdrawn instead of updated. This is determined not only for technological reasons, but also for developmentally strategic reasons in the context of the development of technologies and components that are part of the quantum computing environment. This is influenced by the speed of technology development and huge leaps in performance and capabilities. The withdrawn architectures had a limited number of qubits in the range from 5 to 127 in different connection configurations in the quantum coprocessor, as well as lower quality and layout of connections between them, which resulted in limited application. The assumed modernization would not bring the expected benefits compared to the construction of new systems. The introduction of new architectures ensured the use of more advanced cooling systems based on cryogenics. This is an important move, if only from the point of view of implementing recommendation algorithms operating on big-data is whowed by Weinberg et all (2017) and Vasiliu et all(2017).

The implementation of new quantum architectures and the withdrawal of current ones considered obsolete are closely related to the life cycle of quantum computers, in which guidelines are indicated related to the usability of the equipment in terms of current research on the development of theoretical foundations of quantum computing algorithms. At this stage of the cycle, the theoretical foundations of quantum computers are developed, based on quantum mechanics, in particular focused on the basic units of quantum information, i.e. qubits and on the method of interqubit connection or gathering into groups of subsystems. An important issue also concerns work on adaptations and modern implementations of quantum algorithms.

The cycle responsible for prototyping allows the creation of the first physical models of quantum computers, which is associated with the use of the phenomenon of superconductivity to obtain a quantum state, as well as the use of ion traps and quantum photonics. The key is testing and validation of the created quantum prototypes in terms of quantum errors, decoherence and technical problems that appear only at the testing stage. The last stage in the indicated cycle is the construction of final versions subjected to tests. In the optimization cycle, mechanisms and methods of error management for correcting quantum errors and methods of scaling the system are taken into account, where increasing the number of qubits and their connections is crucial. The implementation cycle includes the introduction of quantum computers to the industry, and in particular to research for applications in cryptography, optimization methods, and the areas of physical, medical and chemical simulations. Maintaining the performance of quantum computers, and in particular constant monitoring of systems, updating hardware and software and conducting research on material technologies for qubits, cooling systems or components occurring in the architecture, and the development of tools and platforms for programming in quantum environments such as Qiskit are tasks carried out in the development cycle. The development of courses and enabling access to quantum technologies and the possibility of creating circuits in the conditions of emerging academic laboratories is a task represented by the educational cycle, which also aims to support and disseminate knowledge about the availability and possibilities of quantum technologies.

The adaptation of current solutions and the desire to increase the speed of current algorithms and calculations by delegating the computational part to a quantum coprocessor is associated with the development of hybrid technologies combining classical and quantum computing. This is part of the cycle responsible for the integration of technologies. A perfect example is the area of research on hybrid classical-quantum recommendation systems, in which the computational part of the algorithm is delegated to a quantum computer, while the classical side is responsible for the method of storing and organizing data sets. The aforementioned

cycles significantly affect decisions made about the modernization or complete replacement of current quantum architectures.

The computational quality of qubits and quantum gates compared to systems present in retired systems was significantly affected by the low qubit coherence coefficient of several dozen to several hundred microseconds (the time for which qubits maintain their quantum state) and relatively high error rates for the performed implementations of quantum circuits. In the Falcon architecture, coherence was limited by the design of superconducting qubits.

Paying attention to the qubit coherence times T1 (relaxation time) and T2 (decoherence time), it is noted that they were about 50 to 100ms for superconducting qubits. In a popular 5-qubit processor, the times were T1=90ms and T2=80ms, respectively. Currently, the 127-qubit Eagle architecture provides times in the range of 200ms. This determines increased stability of qubits and a longer time to perform quantum operations. The times achieved are the result of material optimization of superconductors, the use of new generation resonators, modern cryogenic cooling and the use of modernized electromagnetic screens protecting qubits from external fluctuations. This reduces the impact of the decohort phenomenon on the currently measured states and reduces the error rate showed by IBM Q platform (2024) and IBM Q Development & Innovation Roadmap (2024).

In the withdrawn systems, due to the insufficient number of qubits and hardware limitations, there were limited possibilities of implementing error correction algorithms that allowed for their effective compensation. In the current versions of the systems, the existing method of communication between available qubits has been improved, so that practically each can directly interact with each other in a direct connection. This improves the efficiency of calculations, exchange of quantum information and the possibility of using advanced error correction algorithms for complex circuits and single gates. In the withdrawn systems, the topology of inter-qubit connections was very limited, e.g. in the case of IBM Q 53, each qubit could be connected directly to 2 or 3 other qubits. Improving the inter-qubit topology allows for increasing the efficiency of performing quantum operations of algorithms requiring large interaction between qubits, such as Grover's and Shor's algorithms.

The efficiency of using two-qubit and multi-qubit gates has improved, which allow for more precise performance of operations, which can be seen in the obtained final probability amplitude. In 5-qubit architectures, single-qubit gates had an execution time of around 50ns. Two-qubit gates achieved times in the range of 200 to 400ns. Higher gate operation times mean a higher risk of errors, because operations must be performed before qubit decoherence. The introduction of a precise electromagnetic control mechanism and improved resistance to external noise in the Eagle and Osprey systems allowed achieving times for one-qubit gates below 40ns, and for two-qubit gates values of the order of 100 to 150ns. The accuracy of quantum gate operations is determined by the Fidelity factor, which in both 5 and 16 qubit architectures for one-qubit gates was about 99.5\%, and for two-qubit gates from 96 to 98\%. In the case of currently available Eagle systems, these values are 99.9\% for one-qubit gates, 99.5\% for two-qubit gates. Fidelity value has been improved by introducing new error correction technologies, higher qubit stability and more precise gate control, which reduced the number of errors resulting from logical operations described by IBM Q Research (2024) and Linke et all (2017).

Currently, in order to improve the final result of the circuit implementation, the error correction technique of surface coding is used. In this technique, to protect one qubit from errors, it is necessary to use 100 other physical qubits to implement the surface coding algorithm.

The development of cryogenic technology enables better control over temperature, which has a direct impact on reducing qubit decoherence and increasing the precision of performed quantum operations. Insufficient efficiency of the cooling system causes greater temperature fluctuations, which negatively affects the stability of qubits. The development of cooling technology enables the use of cryogenic systems with constant stability. In decommissioned systems, cryogenic systems cooled superconducting qubits to a temperature of 15 to 20 mK (millikelvins) above absolute zero. These systems had limited energy efficiency and temperature stability, which affected the accuracy of quantum operations. Currently used cryostats maintain the temperature in the range of 10 to 15 mK with greater stability, which minimizes thermal fluctuations and the impact on qubits.

An important element is the possibility of scaling for a large number of qubits. Scaling the system significantly affects the development of existing quantum or hybrid classical-quantum solutions, also looking at the application in terms of big-data solutions prezented by Weinberg et all (2017) and Vasiliu et all (2017). In the case of retired systems, adding new qubits to the existing ones was difficult due to the physical design constraints of the architecture. Qubits must be arranged in such a way as to minimize interference and error

problems. The currently available quantum platform assumes a flexible increase in the number of qubits without an adequately proportionately large increase in decoherence problems. This is possible thanks to the modernization of the way of connections between qubits and increased physical stability of the entire system, which allows the construction of complex systems that can operate on big-data mentioned by Chen et all (2024) and Cross et all (2018).

Circuit implementation in the new quantum architecture

The process of designing quantum circuits that perform specific tasks can be carried out using direct and indirect implementation. Direct implementation will require access to the IBM Q quantum computing cloud, where the circuit can be implemented using the built-in Composer tool used to design the circuit based on available control lines representing qubits and quantum gates, which are predefined. This is a graphical design mode environment. The second way is to implement the circuit using QASM, an editor whose implementation is identified with Composer. Indirect implementation is carried out using the Qiskit package, which allows programming a circuit that performs a specific task using the Python language. It is possible to indicate by referring to specific methods whether quantum calculations in this case, the implementation of the circuit, should be performed on a real quantum computer available within the IBM Q resources assigned to the account, or using a quantum computation simulator also provided by IBM.

In order to implement the circuit on a real machine, it is necessary to have an active account in IBM Q, and to communicate the application based on the ID token assigned to the account. It can be clearly indicated that the above methods of implementing circuits and programming tasks are compatible with each other. This chapter discusses a method aimed at adapting older versions of circuits or entire applications to the requirements of currently available quantum systems.

The approach of implementing a complete application will be considered, together with the process of implementing the circuit for the Qiskit computing package environment, which allows the circuit to be executed on the indicated quantum machines. The adaptation of the application code can be performed manually or in an automated manner. In the case of manual adaptation, it is a long and unrepeatable process, therefore, in order to facilitate the adaptation method, it is recommended to create your own method enabling the modification of the application code in its indicated parts, changing individual values in the code. This is to enable the launch of old circuits on new, currently available quantum machines. The article presents the functionality of the method, indicating the elements necessary for its correct operation. In the example of the automatic adaptation method presented below, the differences between the versions of the implementation code of the identical recommendation task will be discussed by Wróblewski et all (2023) and Sawerwain et all (2019). The method can be divided into four parts. For each of them, what is subject to exchange is indicated. In the case of the adaptive method, each of the indicated methods should be treated as a separate element for conversion, which will allow for the unmistakable distinction of the elements subject to exchange in the source code. The main argument in the invocation of the adaptive method is the indication of the file with the application code, which is subject to conversion and adaptation in order to be compatible with the new IBM Q quantum devices.

The first part is responsible for modifying headers and libraries imported directly into the application. The new version uses different libraries, so it is necessary to indicate the correct ones in order to correctly locate the implemented methods and gateways. Below in Fig. 1 a comparison of both versions of the application part is presented. However, it should be remembered that the changes also result from the fact that qiskit-ibmq-provider has been deprecated. It should also be remembered that QuasmSimulator from qiskit.providers.aer has been replaced in favor of IBMProvider from qiskit ibm provider and Aer-Simulator from qiskit aer. This is important in the context of the adaptation method declaration modifying the part of the code responsible for indicating the libraries used by the application.

```
import numpy as np
from qiskit import QuantumCircuit, transpile, Aer, IBMQ
from qiskit.tools.jupyter import *
from qiskit.visualization import *
from ibm_quantum_widgets import *
from qiskit.providers.aer import QasmSimulator
```

```
import qiskit
from qiskit_aer import AerSimulator
from qiskit_ibm_runtime import QiskitRuntimeService
from qiskit.visualization import plot_histogram
from qiskit import QuantumCircuit, transpile
from qiskit.visualization import *
from ibm_quantum_widgets import *
from qiskit_ibm_runtime import QiskitRuntimeService, Sampler, Estimator, Session, Options
from qiskit_ibm_provider import IBMProvider
```

Fig. 1. Differences in library and header declarations. Old code call on the left, new code call on the right

The second part of the code is to establish communication using the specified IBM Q account ID. Below in Fig. 2 a comparison of both versions of the application part is presented. Note that IBMQ.load account() has been replaced with IBMProvider.save account() where the token ID to the IBM Q account must be provided.

```
provider = IBMQ.load_account()
```

```
IBMProvider.save_account('', overwrite=True)
```

Fig. 2. Differences in communication with the IBM environment. Old code call on the left, new code call on the right

The third part is the so-called implementation part of the quantum circuit, in which the explicit implementation of the quantum circuit takes place. Here, the method of implementation and reference to individual gates has changed. The change concerns the format of arguments. Below in Fig. 3 a comparison of both versions of the application part is presented. Here, the changes concern the method of declaring the quantum circuit and implementing the circuit using single- and multi-qubit quantum gates. Calling the QuantumCircuit(q,c) method, where for q the number of quantum registers was assigned using theQuantumRegister() method, and for c the number corresponding to classical registers was assigned using the ClassicalRegister() method call. This was replaced by the new QuantumCircuit() method from qiskit, which simplifies the method of declaration and assumes that the number of classical and quantum registers for the constructed circuit is the same. Further changes concern the method of indicating and referring to method arguments for marking and implementing quantum gates in the circuit model. The implementation has been changed from qc.h(q[0]) to qc.h(0) for a single-qubit gate, while for multi-qubit gates the implementation for the CX gate example qc.cx(q[1], q[2]) has been changed to qc.cx(1, 2).

```
q = QuantumRegister(3)
c = ClassicalRegister(3)
qc = QuantumCircuit(q, c)

qc.x(q[0])
qc.x(q[1])
qc.h(q[2])
qc.ccx(q[0], q[1], q[2])
qc.h(q[2])
qc.x(q[0])
qc.x(q[1])
```

```
qc = qiskit.QuantumCircuit(3)

qc.x(0)
qc.x(1)
qc.h(2)
qc.ccx(0, 1, 2)
qc.h(2)
qc.x(0)
qc.x(1)
```

Fig. 3. Differences in circuit implementation. Old code call on the left, new code call on the right

The fourth is the way of calling methods for executing the circuit on a simulator or a real quantum computer. Below in Fig. 4 a comparison of both versions of the application part is presented. The change in the way of calling the method from qc.measure(q, c) to qc.measure all() responsible for performing the final measurement of the circuit is important. This also results from the unification of the way of declaring the size of the circuit and indicating the number of registers. The changes also concern methods responsible for starting, e.g. the simulator and indicating the backend of the circuit execution, which is presented in Fig. 4.

```
qc.measure(q, c)
backend = BasicAer.get_backend('qasm_simulator')
job = execute(qc, backend=backend, shots=10000)
result = job.result()
circuit_drawer(qc)
counts = result.get_counts(qc)
```

```
aersim = AerSimulator()
result_ideal = aersim.run(qc).result()
counts_ideal = result_ideal.get_counts(0)
print('Counts(ideal simulator):', counts_ideal)
```

Fig. 4. Differences in the way the circuit is called. On the left is the old code call, and on the right is the new call

The indicated changes are necessary for the adaptive method to be fully effective and its invocation on the code to be effective and enable error-free execution of the old code on the new quantum computer. The presented code example comes from a mechanism that is an element of a hybrid classical-quantum recommendation system.

Analysis of measurement operation results

For the proposed solution, the circuit was executed by performing the final measurement operation resulting in obtaining the distribution of probability amplitudes for the indicated circuit. As mentioned earlier in the article, the circuit indicated in Fig.6 and Fig.5 concerned the mechanism of marking and amplifying the feature in the hybrid classical-quantum recommendation system. It is worth noting that the presented fragment of the circuit is only a part of the complex hybrid classical-quantum recommendation system, on which the measurement was performed on a delegated, no longer current device, as well as on a currently available one to check the operation and effectiveness of the proposed solution in the form of an adaptive method.

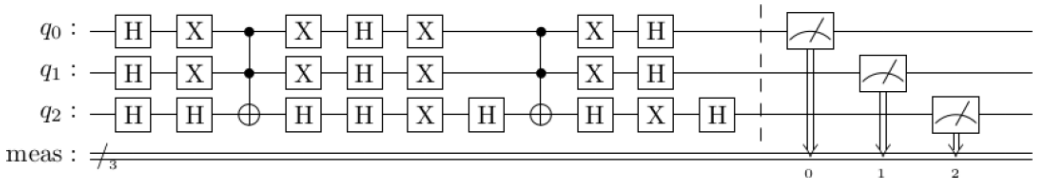

Fig. 5. Graphic design of a quantum circuit designed for the purpose of testing the correctness of the adaptive method

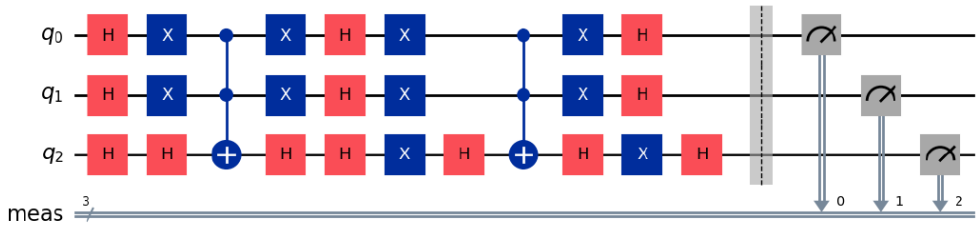

Fig. 6. Graphical implementation of the designed quantum circuit in the IBM Q system

The obtained results are presented below in the form of histograms in Fig. 7 and Fig. 8 containing the distribution of probability amplitudes. Comparing the obtained results for the application before and after the application of the adaptive method, it is stated that the proposed solution realizes the assumed functionalities and the obtained final result does not contain interferences related to the proposed method. The measurement operations were carried out in different time intervals and devices, hence the different distributions of the obtained values. However, analyzing the presented results, it is observed that the assumed operation of the indication and amplification functions of the selected feature in the case of applying the proposed method is correct. The highest indication and thus the amplitude value is still obtained by the index marked in the process. The distribution of probability amplitude values is different due to different environments, however, the maximum value of the probability amplitude always falls on the indicated index 100, which confirms the effectiveness and correctness of the method [19], [20].

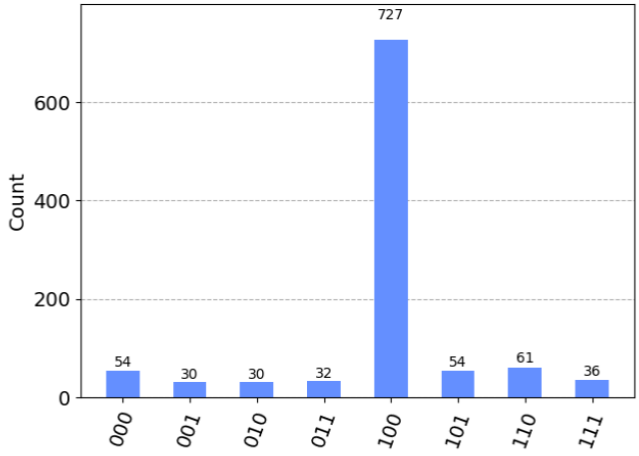

Fig. 7. Distribution of probability amplitudes for feature indication with index 100 derived from execution on the now inactive - retired IBM Q quantum architecture

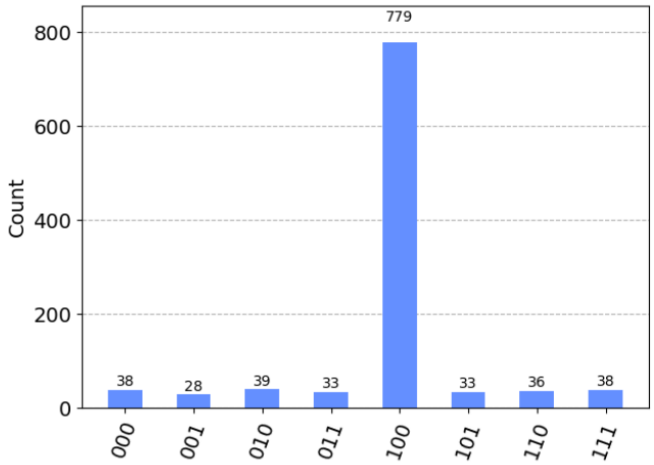

Fig. 8. Distribution of probability amplitudes for feature indication with index 100 derived from the execution on the current available IBM Q quantum architecture

Summary

The article discusses the origins and conditions that prompted the search for ways and methods of adapting designed circuits for previous-generation quantum computers in the IBM Q platform, which offers access to real quantum computers via the cloud.

The key issue considered in the text is the factors resulting from the impact of the need to introduce modern quantum architectures and the conditions they bring. This influenced the decision to retire previously available architectures. First of all, how to run the currently developed purely quantum or hybrid classical-quantum solutions in the face of change and the introduction of new quantum systems.

It should be emphasized that the Authors involvement in the analysis of the quantum life cycle focused on the stages mentioned in the article and the life cycles indicated for quantum computers. Particularly noteworthy are the works conducted in the area of research and development of the theoretical foundations of quantum artificial intelligence algorithms. The influence of noise occurring in the quantum system on the results of the implementation of algorithms generating the indicated quantum circuits. The kNN and Grover algorithms deserve special attention. Optimization of the circuit in terms of the existing method of interqubit communication in the circuit.

A significant contribution is also related to the construction of prototypes and implementation of implementations of quantum computational methods in hybrid classical-quantum recommendation systems. In this context, one of the accompanying studies is the study of the influence of the physical properties of the runtime environment and post-calibration times on the accuracy and quality of the result of the probability amplitude distribution for prototyped quantum artificial intelligence circuits. Closely related to this are maintenance, circuit adaptation and continuous development, as well as adaptation of the circuit to new quantum platforms and integration with classical systems of already developed quantum artificial intelligence algorithms. Modern quantum circuits must be adapted to hardware limitations, such as the maximum number of available qubits, the characteristics of the connections between them (topology), as well as the level of noise and quantum errors. The proposed solution in the form of an adaptive method for applications implementing functions in the form of quantum circuits works well in the case of adapting existing code to a new form, which was discussed and presented in the above chapters of this article.

Looking into the future, it is certain that the dynamism of progress and rapid development of quantum technology and its related accompanying technologies will make the development of current architectures economically and technically unjustified not only from an economic point of view but also from a technical point of view, creating a huge cost and thus another introduction of quantum computers of the next, even newer generation may occur, completely replacing the currently available ones. This will encourage the search for further methods and ways of adapting the code and circuits in order to maintain the currently developed purely quantum and hybrid classical-quantum applications in usability in the future.

References

- Sanchez-Rivero, J., Talav´an, D., Garcia-Alonso, J., Ruiz-Cort´es, A., Murillo, J.M., Automatic generation of an efficient less-than oracle for quantum amplitude amplification, 2023 IEEE/ACM 4th International Workshop on Quantum Software Engineering (Q-SE), pp. 26–33, IEEE, (2023).
- Bae, J-H., Alsing, P.M., Ahn, D., Miller, W.A., Quantum circuit optimization using quantum Karnaugh map, Scientific reports, Vol. 10, 1, pp.15651, Nature Publishing Group UK London, (2020).
- Schutski, R., Lykov, D., Oseledets, I., Adaptive algorithm for quantum circuit simulation, Physical Review A, Vol. 101, 4, pp.042335, APS, (2020).
- Garcia, J., Mandelbaum, R., Davis, R., Janechek, J., Letzter, R., Harishankar, R., Thorpe, L., Murphy, M., Cases Quantum Use., IBM and Daimler use quantum computer to develop next-gen batteries, Quantum, (2023).
- Nielsen, M., Chuang, I., Quantum computation and quantum information, Cambridge University Press, New York (2000).
- Bengtsson, I., Z˙ yczkowski, K., Geometry of Quantum States: An Introduction to Quantum Entanglement, Cambridge University Press, Cambridge (2017).
- Brylinski, K., Chen, G., Mathematics of Quantum Computation, Chapman and Hall/CRC Press, (2002).

- Steeb, W-H., Hardy, Y., Problems and Solutions in Quantum Computing and Quantum Information 4th Edition, World Scientific Publishing Co Pte Ltd, (2018).
- Venturelli, D., Do, M., Rieffel, E. and Frank, J., Compiling quantum circuits to realistic hardware architectures using temporal planners, Quantum Science and Technology, Vol. 3, No. 2, pp. 025004, (2018).
- Brukner, C., On the quantum measurement problem, Quantum [Un] Speakables II: Half a Century of Bell's Theorem, pp. 95-117, (2017).
- Weinberg, A., Last, M., Interpretable decision-tree induction in a big data parallel framework, International Journal of Applied Mathematics and Computer Science, Vol. 27, 4, 737-748 (2017).
- Vasiliu, L., Pop, F., Negru, C., Mocanu, M., Cristea, V., Kolodziej, J., A Hybrid Scheduler for Many Task Computing in Big Data Systems. International Journal of Applied Mathematics and Computer Science, Vol. 27, 2, 385- 99 (2017).
- IBM Q platform, https://quantum.ibm.com/, last accessed – september (2024).
- IBM Q Development & Innovation Roadmap, IBM, (2024).
- IBM Q Research, https://research.ibm.com/quantum-computing, last accessed - september (2024).
- Linke, N.M., Maslov, D., Roetteler, M., et. all., Experimental comparison of two quantum computing architectures, National Acad Sciences, vol. 114, 13, pp. 3305–3310, (2017).
- Chen, T., Ding, H., et. all., Direct Probe of Topology and Geometry of Quantum States on IBM Q, arXiv preprint arXiv:2403.14249, (2024).
- Cross, A., The IBM Q experience and QISKit open-source quantum computing software, APS March meeting abstracts, vol. 2018, pp. L58–003, (2018).
- Wróblewski, M., Quantum Computational Methods in Hybrid Classical –Quantum Recommendation Systems, University of Zielona G´ora Press, (2023).
- Sawerwain, M., Wróblewski, M., Recommendation systems with the quantum k-NN and Grover algorithms for data processing, International Journal of Applied Mathematics and Computer Science, Vol. 29, No. 1, 139–150, DOI: 10.2478/amcs-2019-0011, (2019).