SITIENTIBUS SÉRIE CIÊNCIAS FÍSICAS [ARTICLE]

Implementation of a quantum walk in a cycle of four nodes using an alternative method based on the Swap gate

Implementação de uma caminhada quântica em um ciclo de quatro nós utilizando um método alternativo com base na porta Swap

Aisis R. Barbosa and Dagoberto S. Freitas *

Departamento de Física – UEFS
Av. Transnordestina, s/n, Campus Universitário, Feira de Santana – BA – 44036-900

(SUBMITTED:[30/10/2022] - ACCEPTED:[03/08/2023] - PUBLISHED ONLINE:[05/10/2023])

This work has the objective of implementing quantum random walks in a graph with 4 nodes represented by a circle. It was observed that during the implementation of a discrete-time quantum walk, based on Hadamard's coin, the simulated results diverge from the theoretically expected results. In order to correct the discrepancies between the processed and theoretically expected results we use an alternative method based on the Swap port. To perform the simulation of the algorithms, the Qiskit framework and a real processor provided by IBM through remote access was used. The simulations performed in the real processor showed small fluctuations, however the quantum states were obtained with satisfactory probabilities.

Keywords: Quantum walk; IBM-Q; Qiskit.

Este trabalho tem como objetivo a implementação de passeios aleatórios quânticos em um grafo com 4 nós representado por um círculo. Observou-se que durante a implementação de uma caminhada quântica de tempo discreto, baseada na moeda de Hadamard, os resultados simulados divergem dos resultados esperados teoricamente. Para corrigir as divergências entre os resultados processados e esperados teoricamente utilizamos um método alternativo com base na porta Swap. Para a realização da simulação dos algoritmos serão utilizados a framework Qiskit e de um processador real disponibilizado pela IBM através de acesso remoto. As simulações realizadas no processador real apresentaram pequenas flutuações, entretanto os estados quânticos foram obtidos com probabilidades satisfatórias.

Palavras-chaves: Caminhada quântica; IBM-Q; Qiskit.

I. INTRODUCTION

In recent years, the possibility of generating quantum devices, such as a quantum computing, has stimulated the interest in the study of quantum information theory [1; 2]. In 1994, Peter Shor developed a quantum algorithm to efficiently factor numbers [3] interest in quantum information theory in several areas of scientific knowledge, particularly in physics and computer science. In the last decades, quantum computing has become a major field of study in basic science and frontier technology. In recent years, the interest in quantum computing has grown due to the possibility of quantum algorithms provide considerable computational gains in relation to classical analogues [1; 3]. As a result, we have experienced technological advances that allow us to begin to circumvent some practical problems for the scalable fabrication of quantum computers [1; 4; 5]. An important result of this technological advance was the emergence of quantum simulators [6], that is, quantum circuits simulated in a classical architecture. The main objective of the simulators is to provide a test tool for quantum circuits and algorithms, whether they are aimed at research and/or pedagogical practices. Regardless of the purpose of the simulator, they help to understand important aspects of quantum computing. Among some examples we can mention: LIQUI [7], focused at research and available by Microsoft; the Quantum Computing Playground [8], aimed at research and pedagogical practices, and the Quirk [9], a simulator with a wide variety of available quantum gates and easy implementation of circuits intended for research or pedagogical projects.

In this article we will follow the trajectory of one of the many surprising aspects of quantum information. We will focus our attention on quantum random walks. Quantum walks are considered the quantum analogue of classical walks, which are

* E-mail: dfreitas@uefs.br

useful for the development of classical random algorithms [10]. Quantum walks have already proved useful for designing quantum algorithms [11]. The most general definition of a quantum walk in a graph requires that its evolution in time obey the laws of quantum mechanics and be constrained by the graph locality [12]. The time evolution of quantum walks on graphs can be continuous or discrete [13; 14]. In the discrete-time case, the very first model [13], originally called "random quantum walks", nowadays known as coined quantum walks, has an internal coin space, which was considered mandatory for many years until two alternative restricted coinless models were proposed: Szegedy's [15] and Patel *et. al* [16]. Szegedy's model is defined on bipartite graphs, and Patel *et. al* model on hypercubic lattices. These two models are particular cases of the staggered quantum walk model [17], which is defined on arbitrary graphs. To define discrete-time quantum walks on arbitrary graphs without resorting to internal spaces, the number of local unitary operators, the product of which is the evolution operator, must be larger than two depending on the graph type because locality demands that the vertex set must be partitioned into cliques, inevitably leading to the notion of graph tessellation cover, and to the results proved in [18]. For instance, the evolution operator of discrete-time quantum walk on two-dimensional lattices must be the product of at least four local operators [19].

Various works have discussed the quantum random walks [20–24]. In the reference [20] an introductory survey on quantum random walks is presented. From a physical effect, the main aspects are illustrated, quantum random walks are discussed, reviewing some of their main properties and emphasizing the differences in relation to classical walks. Physical effects and their applications in computer science are discussed, presenting some of the main concepts and language of current quantum information science in this context. In the reference [21] discrete-time quantum walks with one and two walkers interacting in cycles are implemented in IBM quantum computers. In the reference [22] an implementation of quantum random walk using the Qiskit library is presented. The reference [23] discusses the possibility of a quantum walk algorithm yielding such an exponential speedup over possible classical algorithms, without the use of an oracle. Examples of highly symmetric graphs on which efficient quantum circuits implementing quantum walks can be built are provided. And in reference [24] an analysis is made in a real quantum processor, called IBM-Q experience (IBM-Q), as a pedagogical resource for the application of quantum computing concepts. Practical and methodological aspects of IBM-Q are characterized. The main conditions of access to this quantum processor are evaluated and the quality of the interface in terms of simplicity and execution.

II. QUANTUM RANDOM WALKS

Quantum walks can be thought of as the quantum analog of simple classical random walks [20]. The Coined Quantum Walk (CQW) is defined by three main componentes [22]:

(i) The position of the walker that is defined as a vector in na infinite but countable Hilbert space ($|position\rangle \in H_p$). If we consider as basis state $|n\rangle$, then $\{H_P = |n\rangle : n \in \mathbb{Z}\}$.

(ii) A coin operator C that is defined in a two-dimension Hilbert space ($|coin\rangle \in H_c$). If we consider as basis state $|0\rangle$ and $|1\rangle$, then the coin space becomes $H_c = \{|0\rangle, |1\rangle\}$.

(iii) An evolution/shift operator S that performs a step starting from an initial position (i.e., any given node of the graph). Applying this operator is equal to tossing a coin and moving the walker to the left or to the right depending on the outcome of the coin. Left-/Right movements are mirrored by incrementing or decrementing the actual position by 1.

The complete physical system of qubits in which we operate will be in a Hilbert space composed by the product of the two spaces defined above ($H = H_p \otimes H_c$). A state of CQW can be defined by the vector:

$$|\phi\rangle = |position\rangle_{initial} \otimes |coin\rangle_{initial} \ . \qquad (1)$$

The evolution of a discrete-time quantum walk is specified by repeatedly applying the product of the following unit operators [16]:

- Coin operator (C):

$$\begin{aligned} C|n,0> &= a|n,0> + b|n,1>, \\ C|n,1> &= c|n,0> + d|n,1>, \end{aligned} \qquad (2)$$

- Displacement Operator (S):

$$\begin{aligned} S|n,0> &= |n+1,0>, \\ S|n,1> &= |n-1,1>. \end{aligned} \qquad (3)$$

 doi:10.13102/sscf.v19i0.8872

As we will use the Hadamard operator in the proposed circuit, the operator responsible for the evolution of the quantum walk is described as follows:

$$U = S(C \otimes I). \tag{4}$$

To implement operations on the circuit, it is first necessary to observe that for a cycle of order 2^n, n qubits are needed to encode the nodes and 1 additional qubit is needed to encode the subnodes of the graph. The coin operator can be implemented by a single Hadamard gate acting on the qubit responsible for encoding the subnodes and the shift operator can be implemented with the aid of a cyclic permutation of states, in which each state must be mapped to an adjacent state, be it to the right or to the left. This permutation is achieved through the increment $(+)$ and decrement $(-)$ gate. The gates are composed of generalized CNOT or CCNOT gates [17].

The gates produce cyclic permutations, in both directions, of each of the states present in the nodes of the graph. The resulting displacement operator for application to a circuit would be:

$$S = Incr \otimes |1\rangle + Decr \otimes |0\rangle. \tag{5}$$

A. CQW IMPLEMENTATION

In reference [23] shows some examples of graphs for which relatively simple quantum circuits can be designed to efficiently implement quantum walks along them. To implement a quantum walk along a simple cycle it is necessary to note that each node has two adjacent edges and therefore two subnodes. Throughout the cycle, each node is assigned a bit string value so that adjacent nodes receive adjacent bit strings. For a cycle of order 2^n, n qubits are required to encode the nodes, and an additional qubit to encode the subnodes. The coin operation can be implemented by a single Hadamard gate acting on the subnode qubit, and the shifting operation by a cyclic permutation of the node states, in which each state (or bit-string) is mapped to an adjacent state (either higher or lower depending on the value of the subnode qubit). This permutation can be achieved via "increment" and "decrement" gates made up of generalized CNOT (CCNOT) gates, see reference [23]. These gates produce cyclic permutations (in either direction) of the node states. By the definition and properties of quantum gates, when applying the CNOT gate between the 1st and 2nd qubit we observe that if 2nd is $|1\rangle$ the 1st will be inverted. If the controlling qubit is $|1\rangle$ the other qubit will be inverted from $|1\rangle$ to $|0\rangle$ or from $|0\rangle$ to $|1\rangle$. For the case where the controlling quibit is $|0\rangle$ the controlled quibit will keep its state and the controlling qubit will remain unchanged [25]. This behavior can be seen in Table 1. In general, the algorithm that implements a quantum walk can be described by a small set of instructions:

- Initialize the system (position) represented as $|position\rangle\_initial$
- For each iteration of the walk do
- Flip the coin (coin operator)
- Change the position (displacement operator)
- End
- Measurement

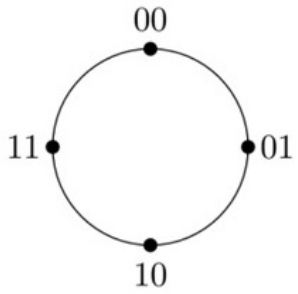

FIGURE 1 – Schematic of a binary graph with 4 nodes.

A simple structure for executing a quantum walk is composed of a space for the coin operator and a register for the displacement. Here we approach the implementation of CQW in a graph with $N = 4$ nodes (C4) shown in Figure 1. The circuit we relied on is shown in Figure 2. In Figure 2, the Coin operator which is a Hadamard gate, represented by H. This part of the circuit decides randomly whether to activate the second or the third block, which means to move right or left in the graph. Now we will explain how to construct this circuit in Qiskit and show the results of executing a one step walk on the IBM-Q. To build the circuit, first we need to encode the graph nodes in a binary notation to fit with qubits. A graph with 2^N nodes needs N qubits for encoding. Here we have $4 = 2^2$ nodes which means we need $N = 2$ encoding qubits. Figure 1 shows the position state for each node in qubit. Moreover, we need 1 qubit for the coin. To implement these components, we divide the circuit shown in Figure 2 into two blocks:

a) Right Shift: In Figure 3 is implements a right shift by increasing the current position by one. For example, applying this circuit to a system $|00\rangle$ produces $|01\rangle$. Note that after performing the walk steps, we will measure only the qubits that refer to the position (i.e. q_1 and q_0), described by the state $|q1q0\rangle$, discarding the qubit $q2$ which refers to the coin described by the state $|q2\rangle$. The full state is given by $|q2q1q0\rangle$ and the order used here ($q2, q1$ and $q0$) is due to the reading order on the IBM-Q processor. The right shift circuit is in Figure 2.

b) Left Shift: Its structure, showed in Figure 4, is very similar to block two except for having negative-controlled not gates that behave the opposite of controlled not gates.

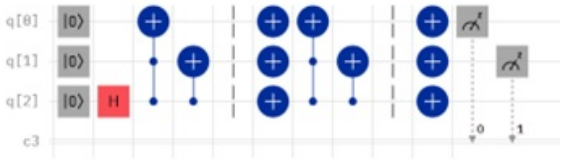

FIGURE 2 – Complete quantum circuit for the quantum walk in a cycle of 4 nodes.

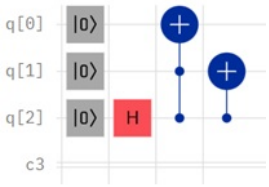

FIGURE 3 – Quantum circuit for the right shift operator.

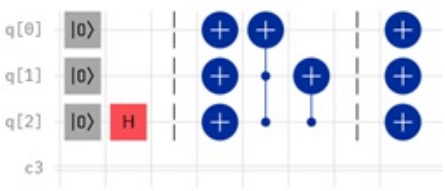

FIGURE 4 – Quantum circuit of the left shift operator.

FIGURE 5 – Results obtained from the IBM-Q platform, where we have the graph of the average results after measurements on qubits q_1 and q_2, with $N = 1024$.

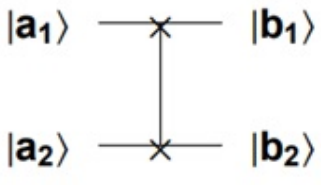

FIGURE 6 – Representation of the Swap gate.

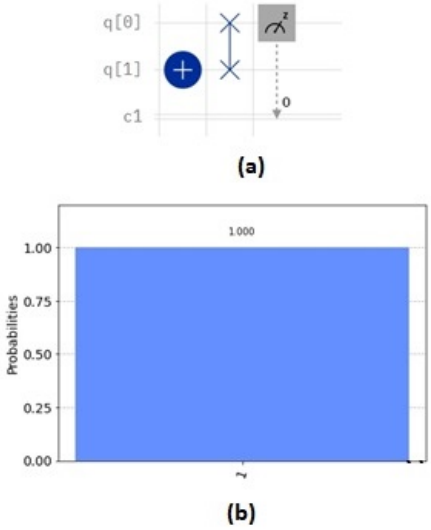

FIGURE 7 – Results obtained from the IBM-Q platform: (a) quantum circuit to be executed; (b) graph of relative frequency after measurements on qubit q_1 and q_2, with $N = 1024$ in a simulator.

To implement the quantum walk we will perform a circuit step using the IBM-Q processor and IBM-

 doi:10.13102/sscf.v19i0.8872

Oslo as the backend. We performed a total of 10 experiments with this provider and this backend.

When measuring the qubits q_1 and q_2 responsible for the displacement of the walker in the graph, Figure 4, we obtained the states $|10\rangle$ and $|11\rangle$, however, this result was not expected. The walker should be in states $|01\rangle$ and $|11\rangle$ with 50% probability each. This result was expected since the initial node $|00\rangle$ on the graph is connected only to states $|11\rangle$ and $|01\rangle$.

Note that the result shown by the platform for position should be read as $|q1q0\rangle$. One way to solve this problem would be to change the position of the gates responsible for measuring the circuit. However, as previously mentioned, the purpose of this circuit is to measure the qubits responsible for the displacement of the walker. Another alternative would be to use the Swap port, Figure 6, before carrying out the measurement and after the entire process of applying the coin operator and the displacement operator.

Previously		After	
Control	Target	Control	Target
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

TABLE I – CNOT gate application on the computational base.

The Swap gate [27] is a 2-qubit operation, which changes the state of the qubits involved in the operation. The Swap gate can be represented by the matrix:

$$Swap = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

The application of the Swap gate is showed in Table II.

The behavior of the Swap gate can be observed in the circuit proposed in Figure 7. Thus, the result demonstrates the exchange of qubit q_1 with qubit q_0. Thus, applying the Swap gate to the circuit that we implement the quantum walk, we obtain the exchange of qubits, returning the expected result.

So, with this change, the set of instructions to be followed would be changed as follows:

- Initialize the system (position) represented as $|position\rangle\_initial$
- For each iteration of the walk do
- Flip the coin (coin operator)
- Change the position (displacement operator)
- End
- Swap qubits (Swap port application)
- Measurement

Previously		After	
Control	Target	Control	Target
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$

TABLE II – Swap gate application on the computational base.

Let us evolve the walk for some steps starting in the initial state $|position\rangle\_initial = |\Psi_0\rangle = |00\rangle$ applying the Swap gate before the measurement,

$$\begin{aligned} |00\rangle \quad &= \quad |\Psi_0\rangle, \\ |\Psi_0\rangle \quad &H \rightarrow \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = |\Psi_1\rangle, \\ |\Psi_1\rangle \quad &S \rightarrow \frac{1}{\sqrt{2}}(|10\rangle + |11\rangle) = |\Psi_2\rangle, \\ |\Psi_2\rangle \quad &Swap \rightarrow \frac{1}{\sqrt{2}}(|01\rangle + |11\rangle) = |\Psi_3\rangle. \end{aligned} \tag{7}$$

The altered circuit is represented in figures 8 and 10.

FIGURE 8 – Example of a quantum circuit to be executed.

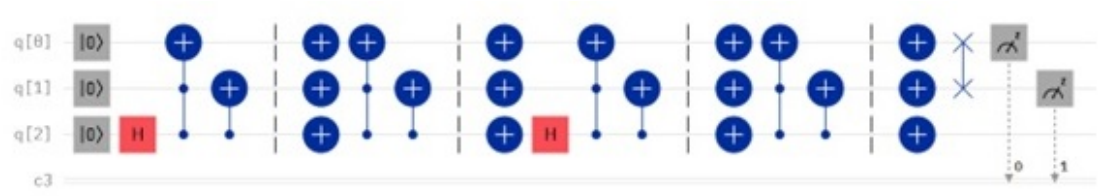

FIGURE 9 – Representation of 2 steps of a quantum walk in a cycle of 4 nodes.

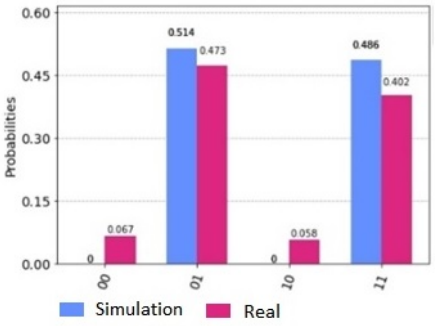

FIGURE 10 – Results obtained from the IBM-Q platform: graph of the relative frequency after measurements on qubits q_1 and q_2, with $N = 1024$.

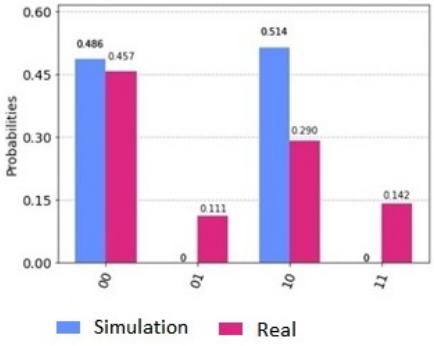

FIGURE 11 – Results obtained from the IBM-Q platform: 2nd step of a walk. All graphs show the relative frequencies after measurements on qubits q_1 and q_2, with $N = 1024$.

Performing the measurement now, the walker will be at $|01\rangle$ and $|11\rangle$, with a probability of approximately 50% each. This result is as expected since the initial node $|00\rangle$ is connected only to states $|01\rangle$ and $|11\rangle$. The results obtained with a real computer presented errors, however the results with the highest probabilistic predominance are the states $|01\rangle$ and $|11\rangle$. Therefore, despite the errors, the results obtained in a real computer are satisfactory.

To implement more steps to the walk, it is only necessary to continue applying the clockwise and counterclockwise displacements together with the Hadamard ports as many times as necessary before applying the port responsible for the measurement and the Swap gate as shown in Figure 9.

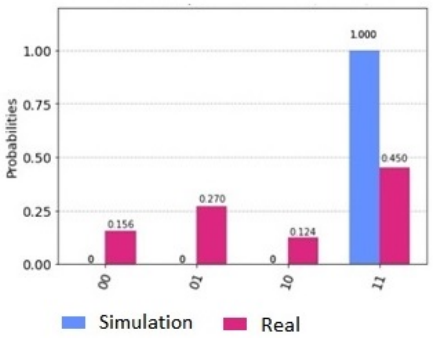

FIGURE 12 – Results obtained from the IBM-Q platform: 3nd step of a walk. All graphs show the relative frequencies after measurements on qubits q_1 and q_2, with $N = 1024$.

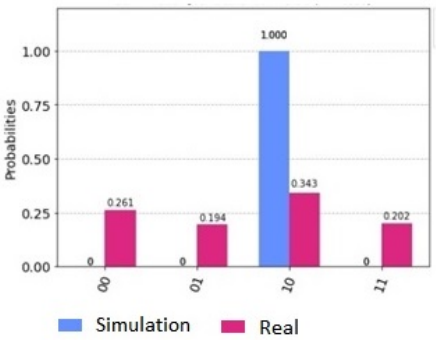

FIGURE 13 – Results obtained from the IBM-Q platform: 4nd step of a walk. All graphs show the relative frequencies after measurements on qubits q_1 and q_2, with $N = 1024$.

 doi:10.13102/sscf.v19i0.8872

In Figures 11–14 are presented the results of a quantum walk during five steps. The results presented show the effects of destructive interference in the 3rd and 4th step. In general, the results in a real quantum computer presented a non-zero probability to the other states of motion of the walker due to the presence of noise in the execution of the circuit.

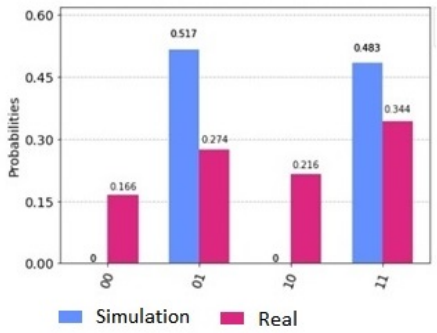

FIGURE 14 – Results obtained from the IBM-Q platform: 5nd step of a walk. All graphs show the relative frequencies after measurements on qubits q_1 and q_2, with $N = 1024$.

III. CONCLUSION

In this work an alternative circuit is presented, using the Swap gate for the implementation of a quantum walk in a graph with 4 nodes. The method used in the implementation must be applied to well-defined graphs because we are dealing with the implementation of a discrete-time walk. Such an implementation is due to the fact that it is necessary to apply values to randomize the line of displacement. This walk has a walker that is interacting with a set of vertices of a cycle, in which case if the walker goes through n steps both clockwise and counterclockwise it will return to its starting point. During this simulation, the errors due to the real computer were noticeable, but they did not influence the effects of the quantum walk. These errors can be associated with the noise present in the computers and can be reduced by performing the simulations in the most updated versions of providers. Quantum walks can work as a more efficient search algorithm than its classical analogue. If both walks are tested for a given search, the quantum walk will return the result faster.

REFERENCES

[1] M.A. Nielsen, I.L. Chuang, *Quantum Computation and Quantum Information*. Cambridge: Cambridge University Press (2000).

[2] R.J. Hughes, D.M. Alde, P. Dyer, G.G. Luther, G.L. Morgan, M. Schauer, *Quantum cryptography*. Contemp. Phys. **36**, (3) 149 (1995).

[3] P.W. Shor, *Algorithms for quantum computation: Discrete log and factoring*. In: S. Goldwasser (Editor). Proceedings of the 35th Annual Symposium on the Foundations of Computer Science, 124 (1994).

[4] P. Kaye, R. Laflamme, M. Mosca, *An Introduction to Quantum Computing*. New York: Oxford University Press (2007).

[5] J.M. Gambetta, M.J. Chow, M. Steffen. NPJ Quantum Information **3**, 2 (2017).

[6] The Quantiki, List of QC simulators, Quantiki. Accessed at: february, 07 (2018).

[7] The Liqui, The Quantum Architectures and Computation Group (QuArC) at Microsoft Research. Microsoft Liqui, Site accessed at february, 07 (2018).

[8] Quantum Playground. Accessed at: february, 07 (2018).

[9] AlgAssert. Accessed at: february, 07 (2018).

[10] R. Motwani, P. Raghavan, *Randomized algorithms*. ACM Comput. Surv. **28**, (1) 33 (1996).

[11] A. Ambainis, *Quantum walk algorithm for element distinctness*. SIAM J. Comput. **37**, (1) 210 (2007).

[12] R. Portugal, *Quantum Walks and Search Algorithms*. 2nd Edition. Cham: Springer Nature (2018).

[13] Y. Aharonov, L. Davidovich, N. Zagury, *Quantum random walks*. Phys. Rev. A **48**, (2) 1687 (1993).

[14] E. Farhi, S. Gutmann, *Quantum computation and decision trees*. Phys. Rev. A **58**, 915 (1998).

[15] M. Szegedy, *Quantum speed-up of Markov chain based algorithms*. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science. Washington: FOCS 04 (2004).

[16] A. Patel, K.S. Raghunathan, P. Rungta, *Quantum random walks do not need a coin toss*. Phys. Rev. A **71**, 032347 (2005).

[17] R. Portugal, R.A.M. Santos, T.D. Fernandes, D.N. Gonçalves, *The staggered quantum walk model*. Quantum Inf. Process. **15**, (1) 85 (2016).

[18] A. Abreu, L. Cunha, C. de Figueiredo, L. Kowada, F. Marquezino, D. Posner, R. Portugal, *The graph tessellation cover number: Chromatic bounds, e-cient algorithms and hardness*. Theoretical Computer Science **801**, 175 (2020).

[19] R. Portugal, T.D. Fernandes, *Quantum search on the two-dimensional lattice using the staggered model with Hamiltonians*. Phys. Rev. A **95**, 042341 (2017).

[20] J. Kempe. Contemporary Physics **44**, (4) 307 (2003).

[21] F. Acasiete, F.P. Agostini, J.K. Moqadam, *et al.*, *Implementation of quantum walks on IBM quantum computers*. Quantum Inf. Process. **19**, 426 (2020). doi:10.1007/s11128-020-02938-5.

[22] P. Olivieri, M. Askarpour, E. di Nitto, *Experimental Implementation of Discrete Time Quantum Walk with the IBM Qiskit Library*. IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE), 33 (2021). doi:10.1109/Q-SE52541.2021.00014.

[23] B.L. Douglas, J.B. Wang. Physical Review A **79**, 052335 (2009).

[24] W.R.M. Rabelo, M.L.M. Costa. Rev. Bras. Ens. Fis. **40**, (4) e4306 (2018).

[25] M.A. Nielsen, I.L. Chuang, *Computação quântica e informação quântica*. Porto Alegre: Bookman (2005).

[26] V.P. Gerdt, R. Kragler, A.N. Prokopenya, *On Simulation of Quantum Circuits with Mathematica*. Computer Algebra Systems. In: Teaching and Research/CASCR, 135 (2007).

[27] J.C. Garcia-Escartin, P. Chamorro-Posada, *A SWAP gate for qudits*. Quantum Inf. Process. **12**, (12) 3625 (2013).