Quantum walks via quantum cellular automata

Pedro C. S. Costa · Renato Portugal · Fernando de Melo

Received: 12 March 2018 / Accepted: 10 July 2018 / Published online: 26 July 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract

Very much as its classical counterpart, quantum cellular automata are expected to be a great tool for simulating complex quantum systems. Here we introduce a partitioned model of quantum cellular automata and show how it can simulate, with the same amount of resources (in terms of effective Hilbert space dimension), various models of quantum walks. All the algorithms developed within quantum walk models are thus directly inherited by the quantum cellular automata. The latter, however, has its structure based on local interactions between qubits, and as such it can be more suitable for present (and future) experimental implementations.

Keywords Quantum cellular automata · Quantum walks · Models of quantum computation · Quantum simulations

1 Introduction

Random walks, a certain type of stochastic processes, were introduced at the beginning of the twentieth century [1]. Since then, random walks have turned into a powerful tool and found applications in areas as diverse as economics [2], ecology [3], computer science [4], and physics [5,6]. Quantum mechanics was developed roughly at the same time. However, it took almost 90 years for the quantum version of random walks, nowadays known as quantum walks (QWs), to be proposed [7]. Despite of its relative young
age, the quantum counterpart of random walks is also being employed in various applications, most notably in quantum simulations [8,9] and in quantum algorithms [10]. Quantum walks, like its predecessor, comes in various flavors—coined [7], Szegedy [11], staggered [12], and staggered with Hamiltonians [13], to cite a few—each with its own specificity and tuned to better deal with a given problem at hand. Foremost, some models of quantum walks were shown to form a universal platform for quantum computation [14,15]. All this versatility and possible applications prompted an intense experimental activity with various realizations of quantum walks, for instance trapped ions [16], optical lattices [17] and more [18].

Most of quantum walk models, however, suffer from two major implementation drawbacks: i) the space associated to the walkers position is usually large, and as such it requires a high-dimensional quantum system to encode it. ii) The unitary evolving the walker is a global one, being applied in all the space in every step. These two points together tremendously limit the size of experimental implementations, and fundamentally put the promising advantages of quantum walks into the far future.

On the classical world, the computational model of cellular automata (CA) is also largely employed [19]. Cellular automata are abstract dynamical/computational systems with discrete space and time. The space consists of a \( d \)-dimensional network of cells. At each time step, individual cells assume one state from a finite number of possible states. The evolution of the network is given in parallel at discrete time steps: at each time \( t \), the state of each cell depends only on the state of its neighboring cells at time \( t - 1 \) through a local update rule [20]. Classical cellular automata, first proposed by von Neumann and Ulam [21], are used in applications ranging from cryptography [22] up to the simulation of fluid dynamics [23], and the description of biological systems [24].

Despite of all the results for classical cellular automata, its quantum counterpart, the quantum cellular automata (QCA), does not share the same amount of research activity. The phenomenology and applications of quantum cellular automata are very scarce. One possible reason for the QCA to not be yet widely employed is that a clear definition, that inherits the main qualities of classical CAs while allowing for a more powerful quantum processing, was not available until recently. The idea of quantum cellular automata was first pursued by Gröss and Zeilinger [25], with a model that later proved to not abide by the locality constraint expected of cellular automata [26]. After this first tentative various models of quantum cellular automata were put forward [27]. After a period of discussion about the proper quantization of cellular automata, in recent years an agreement was obtained and many of the definitions were proved equivalent [28,29].

The main goal of this contribution is to start exploring the model of quantum cellular automata, as a suitable platform for quantum algorithms. We do so by showing that various models of regular quantum walks can be translated to a quantum cellular automata dynamics using the same amount of resources (effective Hilbert space dimension). More concretely, we show how to implement two models of QWs—namely, the coined and the staggered QW with Hamiltonians, which are known to encompass a large class of QW models [13]—within the partitioned QCA model (defined below). Quantum cellular automata, besides employing qubits—which are the base of most quantum computers architectures [30–32]—they do not suffer from
the above-mentioned drawbacks present in QWs: interactions in QCA are local and translation invariant (both in space and time). These QCA advantages, when combined with the various algorithms developed for QWs, might render the implementation of large quantum algorithms viable to current technology.

The article is organized as follows: in Sect. 2 we present the QCA formal definition and work with some examples. Moving forward, in Sect. 3 we show how to translate some quantum walk models into our quantum cellular automata model. We begin with the coined model, starting with a concrete example in the one-dimensional lattice for which we establish the a time-by-time translation of its equation of motion into QCA language. After that, we present a general recipe to translate coined quantum walks into quantum cellular automata. Following the same structure, we then show how to simulate the staggered quantum walk with Hamiltonians with a quantum cellular automata. Finally in Sect. 4, we state some final considerations with possible advantages and applications of these translations.

2 Quantum cellular automata: definition and examples

In classical cellular automata (think for instance on the Wolfram’s elementary CA [20]), the evolution of the cell at position \( x \) is given in two steps: (i) First the state of the neighboring cells of \( x \) is read. (ii) Second, conditioned on this first read measurement, the state of the cell \( x \) is updated according to a fixed local rule. This is repeated for all the cells in the automata in a translation-invariant way, i.e., using the same updating rule for all cells and without a specific updating ordering. The locality of the automata is guaranteed by the neighboring scheme \( \mathcal{N} \) which states that only the cells in the set \( \mathcal{N}_x = \mathcal{N} + x \) will influence the update of cell at position \( x \).

This idea of cellular automata clearly cannot be directly translated to the quantum realm: the state of quantum systems cannot be read without disturbing the system’s state. Moreover, the possible non-commutativity of quantum operations imposes some constraints on the translation-invariant update. Ultimately, the challenges to properly define a quantum cellular automata are rooted at the impossibility of cloning quantum states [33]. Besides that, the classical automata dynamics may be irreversible, which is in contrast with a unitary quantum evolution. One way to define reversible automata is by employing the Margolus block scheme [34]. Here we define a partitioned unitary quantum cellular automata (PUQCA) combining the Margolus block scheme (also employed in [28]), the unitary aspects of the QCA introduced by Pérez-Delgado and Cheung [35], and some dynamical attributes present in the QCA initiated by Watrous [36]. This combination makes clear the locality and translation invariance properties, besides giving a direct quantum generalization of classical partitioned cellular automata.

Two features will become central to our definition of partitioned quantum cellular automata: first each cell is to be divided into \( n \) subcells. This subdivision of each cell generates a finer description for the space where the automaton is defined—be it a lattice or the more general case of a graph. Second, we will employ different tilings over this finer description of the space. Each tiling covers the full space, with
an individual tile covering a finite subset of the subcells. The tiling is then a uniform partition of the set of subcells, with each tile an element of this partition.

With these elements, we are in position to define our partitioned unitary quantum cellular automata.

**Definition 1 (PUQCA)** A partitioned unitary quantum cellular automata is a 5-tuple \((L, \mathcal{N}, \Sigma, \{\mathcal{T}_i\}, \{W_i\})\) consisting of:

1. a \(d\)-dimensional lattice of cells indexed by integers \(L \subseteq \mathbb{Z}^d\);
2. a finite neighborhood scheme \(\mathcal{N} \subseteq L\);
3. a finite set \(\Sigma\) of orthogonal basis states with \(\mathcal{H}_\Sigma = \text{span}\{\sigma | \sigma \in \Sigma\}\). Each cell is divided in \(n\) subcells, and to the \(i\)-th subcell we assign a copy \(\mathcal{H}_\Sigma_i\) of \(\mathcal{H}_\Sigma\). The total space associated to each cell is then \(\mathcal{H}_\Xi = \bigotimes_{i \in \{0, \ldots, n-1\}} \mathcal{H}_\Sigma_i\);
4. a finite set of tilings \(\{\mathcal{T}_i\}_{i=0}^{N-1}\). Each tiling is the union of identical non-overlapping tiles, \(\mathcal{T}_i = \bigcup_j T_j^{(i)}\), with each tile \(T_j^{(i)}\) containing only subcells of neighboring cells.
5. a set of local unitary functions \(\{W_i\}_{i=0}^{N-1}\). The same unitary \(W_i\) is applied to each tile \(T_j^{(i)}\) of the tiling \(\mathcal{T}_i\);

With this definition, the transition function \(\mathcal{E} : (\mathcal{H}_\Xi)^{\otimes L} \mapsto (\mathcal{H}_\Xi)^{\otimes L}\), which updates the automaton state from the time \(t\) to \(t+1\), is given by

\[
\mathcal{E} = \prod_{i=0}^{N-1} \left( \bigotimes_{T_j^{(i)} \in \mathcal{T}_i} W_i \right). \tag{1}
\]

To work with these tilings more precisely, it is convenient to put labels in each subcell. Given the cell at position \(i \in L\), its subcells are denoted by \(i_j\), with \(j \in \{0, \ldots, n-1\}\). For instance, suppose we have an one-dimensional lattice where each cell has two subcells, and the neighboring scheme is \(\mathcal{N}_i = \{i-1, i, i+1\}\). In this case, two tilings are sufficient to evolve the automaton: The first tiling is given by \(\mathcal{T}_0 = \bigcup_{i \in \mathbb{Z}} T_i^{(0)}\) with each tile defined as \(T_i^{(0)} = \{i_0, i_1\}\). For the second tiling, \(\mathcal{T}_1 = \bigcup_{i \in \mathbb{Z}} T_i^{(1)}\), each tile is given by \(T_i^{(1)} = \{i_1, (i+1)_0\}\). It is then clear that the first tiling is responsible for “reading” the state of each cell, while the second one is responsible for the interaction between the neighboring cells—resembling the unitary quantum cellular automata defined in [35]. Now that the tilings’ structure is established, the action of the unitary functions is apparent:

\[
W_0 : (\mathcal{H}_\Sigma_0)_i \otimes (\mathcal{H}_\Sigma)_i \to (\mathcal{H}_\Sigma_0)_i \otimes (\mathcal{H}_\Sigma)_i,
\]

\[
W_1 : (\mathcal{H}_\Sigma_1)_i \otimes (\mathcal{H}_\Sigma)_i \to (\mathcal{H}_\Sigma_1)_i \otimes (\mathcal{H}_\Sigma)_i,
\]

for all \(i \in \mathbb{Z}\). Therefore, in this example we can explicit our transition function as

\[
\mathcal{E} = \left( \bigotimes_{T_i^{(1)} \in \mathcal{T}_i} W_1 \right) \left( \bigotimes_{T_i^{(0)} \in \mathcal{T}_0} W_0 \right). \tag{2}
\]
Fig. 1 1-dimensional automaton. In top-panel (a) it is shown how each cell is split in two subcells, and how the unitary operators $W_i$ are applied in accordance with the two tilings. In the bottom-panel (b) it is shown the same 1-d automaton, but now in the graph perspective. The tiling $T_0$ is represented by the red ellipses, while the tiling $T_1$ is shown in blue (Color figure online).

Fig. 2 Graph perspective: 2-dimensional automaton. For the automaton defined over the 2-dimensional lattice, with four neighbors, each cell is transformed into a complete graph with 4 vertices (K4). The first tiling is shown in red, with corresponding unitary operation $W_0$ acting only on the subcells of each cell. The second tiling is depicted in blue, with the unitary operation $W_1$ being responsible for the interaction between neighboring cells (Color figure online).

All this construction can be apprehended from Fig. 1a.

The above definition of the PUQCA immediately generalizes to QCA over a regular graph $G = (V, E)$. In this situation, the neighborhood scheme is represented by the edge set $E$, and the tilings are defined over partitions of the graph $G$ into complete subgraphs—in every tiling all the vertices are included, and the union of the tilings must contain all the edges. Within graph theory, tilings are usually called tessellations (see the definitions used in Sec. 3.2). We also can see how the one-dimensional example is represented in the “graph perspective” in Fig. 1b. Another example of a QCA in the graph picture is shown in Fig. 2.

In what follows, we will mostly employ the graph perspective, as it makes clear the translation procedures from quantum walks to quantum cellular automata.

3 Translating quantum walks into quantum cellular automata

This section contains our main result: how to translate a large class of regular quantum walk models into quantum cellular automata models. We focus on the Coined Quantum Walk over a $d$-regular graph (CQW$_d$) and on the regular Staggered Quantum Walk...
with Hamiltonians (SQWH), as these two models were already shown to encompass various models of quantum walks [13]. The requirement of regularity for these models (homogeneous coin operator for CQWs and polygons of equal size for each tessellation for the SQWH—see below) is necessary as we are employing homogeneous QCA, i.e., the operators \( W_i \) do no change with the cell position. At the end of each translation, we compare the amount of resources used by the quantum walk model with the resources consumed by the corresponding quantum cellular automaton.

3.1 CQW_d \( \subseteq \) QCA

3.1.1 Coined quantum walk over a \( d \)-regular graph

Coined quantum walks were the first to be devised [7], likely due to their similarity to the classical case. The main difference from the classical to the quantum walk lies in the possibility, in the latter, of superpositions of the coin values and walker positions. Since these early times, the definition of quantum walks evolved, specially to cope with general dynamics in arbitrary finite graphs [37]. Given a graph \( G = (V, E) \), its vertices are associated to the walker’s classical positions, while its edges define the possible directions the walker can take in a single step. To the position \( i \in V \) and direction \((i, j) \in E\), we associate the state \(|(i, j), i \rangle\).

### Definition

As each edge is connected to exactly two vertices, the dimension of \( \mathcal{H}_G \) is \( 2|E| \).

The dynamics of a quantum walker is given by first the “flip” of a quantum coin, followed by a coin-dependent coherent displacement. Given a walker in the state \(|(i, j), i \rangle\), the coin operator must create a superposition of all allowed directions, i.e., a superposition of the states \(|(i, k), i \rangle\) for all \(k \in \mathcal{N}_i^G\). The action of the coin operator is more easily understood if we decompose \( \mathcal{H}_G \) in blocks related to each vertex:

\[
\mathcal{H}_G = \bigoplus_{i \in V} \mathcal{H}_G^{(i)}, \quad \text{with} \quad \mathcal{H}_G^{(i)} = \text{span}(|(i, j), i \rangle|(i, j) \in E, \ j \in \mathcal{N}_i^G).
\]

For a \( d \)-regular graph, each such a block is a \( d \)-dimensional Hilbert space. The action of the coin operator \( C : \mathcal{H}_G \rightarrow \mathcal{H}_G \) can be written as:

\[
C = \bigoplus_{i \in V} C_i, \quad (3)
\]

where each \( C_i \) is a unitary acting on \( \mathcal{H}_G^{(i)} \). If the coin flip is independent of the vertex, which is usually the case, we take the action of the coin operator to be operationally identical in each vertex-subspace. This homogeneity condition induces a labeling of the edges connected to a given vertex—to each edge \((i, j)\) connected to a vertex \( j \) we can give a label from \( \{0, 1, \ldots, d - 1\} \) which is defined by the ordering of the basis

---

1 Note that the space \( \mathcal{H}_G \) is not given by the tensor product of a space associated to the edges and another space associated to the vertices.
elements \(|(i, j), i)| \in \mathcal{H}_{G}^{(i)}\) when writing each \(C_i\). To finish the walker step, we must define the shift operator \(S_{\pi} : \mathcal{H}_{G} \to \mathcal{H}_{G}\). As the walker can only walk “over” the edges, the action of the most general shift operator is given by:

\[
S_{\pi} |(i, j), i) = |(\pi(i, j), j), j), \forall i \in V \text{ and } \forall(i, j) \in E.
\]

Here \(\pi\) is a function that fixes the walker’s direction after the shift operation, i.e., \(\pi\) takes the outgoing vertex \(i\) and the receiving vertex \(j\) and returns one of the \(d\) neighbors of the receiving vertex, i.e., \(\pi(i, j) \in \mathcal{N}_{G}^{j}\). As we are dealing with regular CQWs, here we focus on shift operators for which the action of \(\pi\) depends only on the induced labels by the coin operator (see above). In such a case, there are possibly only \(d!\) shift operators. Moreover, since the function \(\pi\) does not change the vertex value, we can choose it as \(\pi(i, j) = i\), and absorb some possible relabeling in the coin operator. The shift operator for this choice of function will be denoted by \(S_I\) and its action is given by \(S_I |(i, j), i) = |(i, j), j)\). This shift operator is known as the flip-flop shift as \(S_I^2 = 1\). Given all that, the one time-step evolution of the quantum walker is given by the unitary \(U_G : \mathcal{H}_{G} \to \mathcal{H}_{G}\), whose action can be written as:

\[
U_G = S_I \cdot C.
\]

In what follows we show how to translate each element of the coined quantum walk into a quantum cellular automata description. We start with a simple example, followed by a general translation procedure.

### 3.1.2 One-dimensional example

The most paradigmatic example of coined quantum walk is the one-dimensional case, with a two-state coin. This construction closely resembles the classical random walk, and it was the main inspiration for the generalization to the quantum domain [7]. In this instance, the walk happens in a one-dimensional lattice, which can be seen as an infinite 2-regular graph. In such a graph perspective, the set of vertices is \(V = \mathbb{Z}\), and the set of edges is \(E = \{(i, i + 1)|i \in V\}\), leading to a vertex-neighborhood \(\mathcal{N}_{G}^{i} = \{i - 1, i + 1\}\). Moreover, if the walker is on the vertex \(i\) moving to the right we associate to it the state \(|(i, i + 1), i)|\), while if it is moving to the left we describe it by the state \(|(i, i - 1), i)|\). For the walker in such a position, the coin operator must superpose the two directions of movement, \(|(i, i + 1), i)|\) and \(|(i, i - 1), i)|\). This can be done by choosing \(C_i : \mathcal{H}_{G}^{(i)} \to \mathcal{H}_{G}^{(i)}\) as a SU(2) operator

\[
C_i = \begin{pmatrix} q & p \\ -p^* & q^* \end{pmatrix} \forall i \in V,
\]

with \(p, q \in \mathbb{C}\) respecting the unitarity constraints \(|p|^2 + |q|^2 = 1\), and the matrix is written in the basis \(|(i, i - 1), i)|, \(|(i, i + 1), i)|\}. The total coin operator is then \(C = \bigoplus_{i \in V} C_i\). Traditionally, the shift operator does not change the coin value, i.e.,
it keeps the direction of the movement. This shift is known as the “moving” operator, and it acts on $\mathcal{H}_G$ as follows:

$$S_X | (i, i - 1), i \rangle = | (i - 2, i - 1), i - 1 \rangle,$$

$$S_X | (i, i + 1), i \rangle = | (i + 2, i + 1), i + 1 \rangle.$$ (6)

The relation of between the moving shift $S_X$ to the flip-flop $S_I$ one is simple: $S_X = X \cdot S_I$, where $X = \bigoplus_{i \in V} X_i$ with

$$X_i = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \forall i \in V.$$ A single-step evolution is then given by $U = X \cdot S_I \cdot C$.\(^2\) If the state of the system at time $t$ is

$$| \psi(t) \rangle = \sum_{i \in V} \left( \psi_{(i, i - 1)} (i, t) | (i, i - 1), i \rangle + \psi_{(i, i + 1)} (i, t) | (i, i + 1), i \rangle \right),$$ (7)

where $\psi_{(i, j)} (i, t) := (i, j) | \psi(t) \rangle$ is the amplitude of the walker being located at the vertex $i$ pointing to the vertex $j$ at time $t$. Then the state at $t + 1$ is obtained by $U | \psi(t) \rangle$, which gives

$$\sum_{i \in V} \left[ (q \psi_{(i, i - 1)} (i, t) + p \psi_{(i, i + 1)} (i, t)) | (i - 2, i - 1), i - 1 \rangle + (q^* \psi_{(i, i + 1)} (i, t) - p^* \psi_{(i, i - 1)} (i, t)) | (i + 2, i + 1), i + 1 \rangle \right].$$

It is now immediate to obtain the recurrence relations that govern the quantum walk dynamics:

$$\psi_{(i - 2, i - 1)} (i - 1, t + 1) = q \psi_{(i, i - 1)} (i, t) + p \psi_{(i, i + 1)} (i, t),$$

$$\psi_{(i + 2, i + 1)} (i + 1, t + 1) = q^* \psi_{(i, i + 1)} (i, t) - p^* \psi_{(i, i - 1)} (i, t).$$

These are the dynamical equations that fully describe the walker dynamics in one-dimensional lattice with a moving shift.

Now we want to reproduce the dynamics achieved above within the cellular automata formalism. In order to do that, each vertex $i \in V$ we change to a cellular automaton cell. As each vertex has two neighbors (leading to a two-dimensional coin), each automaton cell is divided into two subcells. A cell $i$ is then composed by two subcells labeled by $(i)_{(i - 1)}$ and $(i)_{(i + 1)}$. In each subcell, we place a two-dimensional quantum system (a qubit), i.e., $\mathcal{H}_{(i)_{(i - 1)}} \simeq \mathcal{H}_{(i)_{(i + 1)}} \simeq \mathbb{C}^2$, for all $i \in V$. We then encode the quantum walk state $| (i, i - 1), i \rangle$ with the automaton state $| \ldots, (0, 0)_{i - 1}, (1_{i - 1}, 0)_{i + 1}, (0, 0)_{i + 1}, \ldots \rangle$, while the state $| (i, i + 1), i \rangle$ is written as $| \ldots, (0, 0)_{i - 1}, (0_{i - 1}, 1_{i + 1})_{i}, (0, 0)_{i + 1}, \ldots \rangle$. In this way, an excited subcell (a

\(^2\) Note that since the evolution for $t$ time steps is given by $U^t$, then the operator $X$ can be absorbed in the coin operator by suitably applying corrections on the initial and final state: $U^t = X \cdot (S_I \cdot C X) X^{-1}$.
qubit in the 1 state) indicates the walker position and its movement direction. All the encoding is done within the single excitation subspace.

For the dynamics translation, for clarity, we will employ one tiling for each quantum walk unitary. As such, we require three tilings. The first one, \( T_0 \), is related to the coin operator. As this only changes the edge state, without moving the walker, the tiles are the very subcells of each cell: \( T_i^{(0)} = \{(i)_{i-1}, (i)_{i+1}\} \). The unitary \( W_0 \) associated to this tiling is of the same form as the unitary \( C_i \) employed in the coin \( C \):

\[
W_0 = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & q & p & 0 \\
0 & -p^* & q^* & 0 \\
0 & 0 & 0 & 1
\end{pmatrix},
\]

also fulfilling the requirements that \(|p|^2 + |q|^2 = 1\). Note that now such operator acts on a two-qubit system, \( W_0 : \mathcal{H}_{\Sigma_{i(i-1)}} \otimes \mathcal{H}_{\Sigma_{i(i+1)}} \rightarrow \mathcal{H}_{\Sigma_{i(i-1)}} \otimes \mathcal{H}_{\Sigma_{i(i+1)}} \), and it is written in the basis \( \{|0, 0\rangle_i, |1, 0\rangle_i, |0, 1\rangle_i, |1, 1\rangle_i\} \). Moreover, notice that \( W_0 \) only acts non-trivially on the single excitation subspace. Writing a general state \(|\Psi(t)\rangle\) for the automaton at time \( t \) as:

\[
\sum_{i \in \mathbb{Z}} \left[ |\Psi_{i-1}(i,t)\rangle \cdots |(0,0)_{i-1} (1)_{i-1} (0)_{i+1} \rangle + |\Psi_{i+1}(i,t)\rangle \cdots |(0,0)_{i-1} (0)_{i-1} (1)_{i+1} \rangle \right],
\]

where \( |\Psi_{i\pm 1}(i,t)\rangle \) is the probability amplitude of finding an “excitation” at the subcell \( i \pm 1 \) of the \( i \)-th cell at time \( t \), after the evolution corresponding to the first tiling,

\[
\left( \bigotimes_{T_0} W_0 \right) |\Psi(t)\rangle,
\]

we get:

\[
\sum_{i \in \mathbb{Z}} \left[ \left( q|\Psi_{i-1}(i,t)\rangle + p|\Psi_{i+1}(i,t)\rangle \right) \cdots |(0,0)_{i-1} (1)_{i-1} (0)_{i+1} \rangle + \left( q^*|\Psi_{i+1}(i,t)\rangle - p^*|\Psi_{i-1}(i,t)\rangle \right) \cdots |(0,0)_{i-1} (0)_{i-1} (1)_{i+1} \rangle \right].
\]

Now we use a second tiling \( T_1 \), to simulate the flip-flop shift operator \( S_I \). The second tiling is thus responsible for the interaction between the cells. As such, the associated unitary operation \( W_1 \) must act on tiles that contain subcells from neighboring cells. As the walker wave-function can only spread by one vertex in a single step, to the left and to the right, the automaton neighborhood is given by \( N_j = \{j - 1, j, j + 1\} \). From the definition of the flip-flop shift operator \( S_I \) and the encoding of the coin-walker state in the automaton language, we note that the probability amplitude in the subcell \( i + 1 \) of the \( i \)-th cell \( (i+1)_i \) is to be transfered to the subcell \( i \) of the \( (i+1) \)-th cell \( ((i+1))_i \), and vice versa—in the graph perspective, we create an edge \((i, i + 1)\) for all \( V \). Given that, for the tiling \( T_1 \) we set the tiles \( T_i^{(1)} = \{(i)_{i+1}, (i+1)_{i}\} \), and the unitary function \( W_1 \) is simply the swap operator. After the evolution due to these two tilings,

\[
\left( \bigotimes_{T_1} W_1 \right) \left( \bigotimes_{T_0} W_0 \right) |\Psi(t)\rangle,
\]

a general state \(|\Psi(t)\rangle\) the state of the automaton reads:
Lastly, we need to simulate the action of the $X$ operator with a third tiling, $\mathcal{T}_2$. Like for the coin operator, the $X$ operator does not change the walker’s position. As such, its corresponding operator acts only on individual cells: $T_i^{(2)} = \{(i)_{i-1}, (i)_{i+1}\}$. Moreover, as the action of each $X_i$ is simply to change the movement direction, here we only need to take $W_2$ as a swap operator acting in each cell to get:

$$
\sum_{i \in \mathbb{Z}} \left[ (q \Psi_{(i-1,i)} (i, t) + p \Psi_{(i+1,i)} (i, t)) \mid \ldots, (0, 0), (0, 0), (0, 0) \ldots \right] +
$$

$$
+ \left[ (q^* \Psi_{(i,i+1)} (i, t) - p^* \Psi_{(i,i-1)} (i, t)) \mid \ldots, (0, 0), (0, 0), (0, 0) \ldots \right].
$$

That is the automaton state after on time step, i.e., after the action of $\mathcal{E} = \left( \bigotimes_{T_j^{(2)} \in \mathcal{T}_2} W_2 \right) \left( \bigotimes_{T_j^{(1)} \in \mathcal{T}_1} W_1 \right) \left( \bigotimes_{T_j^{(0)} \in \mathcal{T}_0} W_0 \right)$. It is now immediate to obtain the recurrence relations that describe the automaton dynamics:

$$
\Psi_{(i-2,i-1)} (i-1, t+1) = q \Psi_{(i-1,i)} (i, t) + p \Psi_{(i+1,i)} (i, t),
$$

$$
\Psi_{(i+2,i+1)} (i+1, t+1) = q^* \Psi_{(i,i+1)} (i, t) - p^* \Psi_{(i,i-1)} (i, t) (i, t).
$$

These are exactly the same recurrence relations for the 1-d coined quantum walk. Both dynamics are thus identical at every time step.

### 3.1.3 General recipe

Now we give a prescription to find the QCA correspondent to a given coined QW on a $d$-regular graph. As input we take a $d$-regular graph $G = G(V, E)$ (or lattice $L$) where the quantum walk is defined; the coin operator $C$; and the shift operator $S_\pi$. As output we must return a complete PUQCA whose evolution is the same as the CQW. The steps to find this translation are enumerated below.

1. The number of cells in the automaton is given the number of vertices, $|V|$, of the graph $G$. As the graph is $d$-regular, each cell is split in $d$ subcells. We place one qubit in each subcell, and then a total of $\lfloor V, d \rceil$ qubits are employed (see the resources discussion below).

2. The automaton neighborhood scheme $\mathcal{N}$ is determined by the graph-neighborhood $\mathcal{N}^G$, by the simple inclusion of the “central” cell: $\mathcal{N}_i = \mathcal{N}_i^G \cup \{i\}$.

3. To each CQW$_d$ state $| (i, j), i \rangle$, we associate the automaton (single-excitation subspace) state $| \ldots (0_h, \ldots, 1_j, \ldots, 0_m), i \rangle \ldots \rangle$, where $h, m$ and all other subindex labeling the subcells of cell $i$ belong to $\mathcal{N}_i^G$. While the subindex $i$ gives us in which cell the excitation is located, the subindex $j$ tells us its subcell location (corresponding to the movement direction). In this way, the space associated to each cell is $\mathcal{N}_\Sigma = (\mathbb{C}^2)^d$. 
4. To simulate the CQW\(_d\) dynamics within the automata language three tilings are sufficient.\(^3\) The first tiling is related to the action of the coin, with each tile given by all subcells that belong to the same cell: \(T_i^{(0)} = \{(i) j | j \in N_i^G\}\). The second tiling is devoted to the simulation of the flip-flop shift acting on neighboring cells. From the definition of \(S_f\), we can see that in terms of QCA we have the subcell \(j\) in the cell \(i\) interacting with the subcell \(i\) of the \(j\)-th cell, and vice versa. Therefore \(T_i^{(1)} = \{(i) j, (j)i\}\) where \((i, j) \in E\). The third tiling is responsible for simulating the operator that connects \(S_f\) to \(S_\pi\). As this operation is “local” in each vertex, then \(T_i^{(2)} = \{(i) j | j \in N_i^G\}\).

5. To each tiling, we associate one unitary operator. To the first tiling, the unitary operator \(W_0: H_\Xi \rightarrow H_\Xi\) is directly related to the unitary operator \(C_i\) by employing the unary representation for the cell states (see two steps above) within the single-excitation subspace. Out of this subspace, the action of \(W_0\) is trivial, being completed by ones in the diagonal entries. Since the flip-flop operator is translated as a swap between the two subcells of neighboring cells, the unitary \(W_1: H_{(i)j} \otimes H_{(j)i} \rightarrow H_{(i)j} \otimes H_{(j)i}\), for \((i, j) \in E\), is the swap gate between them. Lastly, \(W_2: H_\Xi \rightarrow H_\Xi\) implements the function \(\pi\) on the cell \(i\), by encoding the operator associated to \(\pi\) in the same way as for the coin operator.

These steps give the full translation between a CQW\(_d\) and a QCA.

This recipe can be easily generalized to translate a CQW whose dynamics is given by a succession of different coin and shift operators, i.e., \(U_G = (S_\pi^{(n)} \cdot C^{(n)}) \cdot (S_\pi^{(n-1)} \cdot C^{(n-1)}) \cdot \ldots \cdot (S_\pi^{(1)} \cdot C^{(1)})\), as discussed for instance in [38]. In this case, steps 4 and 5 of the above recipe have to be iterated \(n\) times, and in each iteration a different set of \(W_0\) and \(W_2\) might be necessary. Note that the number of qubits necessary does not change.

Before we move on, we compare the resources required in each model. The dimension of the Hilbert space in the quantum walk model is \(2|E|\), which is equal \(|V|d\) due to the assumed \(d\)-regularity of the graph. For the corresponding QCA at first sight, we would need a Hilbert space of dimension \(|V|2^d\) (as we have \(|V|\) cells with \(d\) qubits each). Nevertheless, as we only need the single-excitation subspace for our construction, whose dimension is \(d\), we in fact only use an effective Hilbert space with dimension \(|V|d\). Therefore, both models require the same amount of resources.

### 3.2 SQWH \(\subseteq\) QCA

#### 3.2.1 Staggered quantum walk with hamiltonians

Sometime after the development of the coined quantum walk model, it was realized that similar coherent dynamics could be obtained even without a coin space. In such models, the coin is removed and the walker’s dispersion over a graph is obtained via the alternation of operators directly acting on the walker’s position space. The first “coinless” quantum walk model was proposed by Szegedy [11], followed by the

\(^3\) Here again the action of the third tiling can be absorbed in the action of the first one, plus modifications in the initial and final state. We, however, present the translation with three tilings for clarity reasons.
staggered quantum walk [12,13,39,40], and, more recently, by the staggered quantum walks with Hamiltonians (SQWH) [13]. The latter was shown to encompass the previous two models of coinless quantum walks, and even some coined models [13]. Furthermore, the SQWH has been shown to be useful for quantum search [41], and an experimental proposal is already in place [42]. Our aim here is thus to show that the PUQCA can simulate the SQWH, and with that inherit the simulation of various coinless, and coined, models.

We start by describing the SQWH over a graph \( G = (V,E) \) [13]. Since we aim at translating the SQWH dynamics into a homogeneous QCA, we focus on \( d \)-regular graphs. As before, to each vertex \( i \in V \) we associate a unit vector \( |i\rangle \), with \( \langle i|j\rangle = \delta_{ij} \) for all \( i, j \in V \). As such, to the vertices of \( G \) we associate the Hilbert space \( \mathcal{H}_V = \text{span} \{ |i\rangle | i \in V \} \). Crucial to the SQWH is the concept of a graph tessellation: A graph tessellation \( \mathcal{T} \) is a partition of the graph into complete subgraphs, i.e., into cliques. Such a partition directly induces a partition of \( \mathcal{H}_V \): let \( \alpha \) be an element of \( \mathcal{T} \), then \( \mathcal{H}_V = \bigoplus_{\alpha \in \mathcal{T}} \mathcal{H}_\alpha \), where \( \mathcal{H}_\alpha = \text{span} \{ |i\rangle | i \in \alpha \} \). Each element \( \alpha \) is called a polygon, as it is related to a clique. Due to the regularity of the graph, we will take tesselations in which all the polygons have the same size. It is now simple to define a rank-one projector \( |\alpha\rangle\langle\alpha| \) into \( \mathcal{H}_\alpha \), by defining the vector \( |\alpha\rangle = \sum_{i \in \alpha} a(i) |i\rangle \), where \( a(i) \in \mathbb{C} \) and \( \sum_{i \in \alpha} |a(i)|^2 = 1 \). As the polygons are of equal size, the coefficients \( a \) do not depend on the polygon in a given tessellation, but they do depend on the label given to each vertex within polygon [13]. A dynamics that does not connect different subspaces \( \mathcal{H}_\alpha \) can be given by the Hamiltonian operator associated with this tessellation as

\[
H_{\mathcal{T}} = 2 \sum_{\alpha \in \mathcal{T}} |\alpha\rangle\langle\alpha| - 1. \tag{8}
\]

Such operator is known as the orthogonal reflection of the graph [40], and it is Hermitean and unitary, implying that \( H_{\mathcal{T}}^2 = 1 \). The dynamics generated by this Hamiltonian is then \( U_{\mathcal{T}} = \exp(i\theta H_{\mathcal{T}}) \), with \( \theta \in [0, 2\pi] \). This propagator respects the partition of \( \mathcal{H}_V \) into subspaces related to the tessellation polygons, as \( U_{\mathcal{T}} = \bigoplus_{\alpha \in \mathcal{T}} U_\alpha \) with

\[
U_\alpha = e^{-i\theta} \mathbb{1}_\alpha + 2i \sin(\theta) \sum_{i,j \in \alpha} a^*(i)a(j) |i\rangle\langle j|; \tag{9}
\]

where \( \mathbb{1}_\alpha := \sum_{i \in \alpha} |i\rangle\langle i| \) is the identity operator in the \( \alpha \) subspace.

If the dynamics of the walker was to be given simply by the propagator \( U_{\mathcal{T}} \), then a walker starting in a vertex \( i \in V \) would remain stuck in the polygon that contains such a vertex. As Szegedy noticed [11], a walker dispersion over a graph can be obtained without a coin if we alternate propagation operators, with each of them acting within a different subspace partition of \( \mathcal{H}_V \). At this point we observe that a given tessellation contains all the vertices of a graph, but not necessarily all its edges. In [13] it was defined a set of tessellations, a tessellation cover \( \{\mathcal{T}_0, \ldots, \mathcal{T}_{N-1}\} \), whose union also

\( \odot \) Springer
This picture shows the two tessellations used in the 1−d example of translation between the SQWH and the PUQCA. Within the SQWH the operator \( U_0 (U_1) \) acts on the red (blue) polygons. In the PUQCA picture, this is translated to the action of \( W_0 (W_1) \) on the qubits in the red (blue) tiles (Color figure online).

covers the edge set. Each tessellation \( \mathcal{T}_k \) induces a different subspace partition of \( \mathcal{H}_V \), with associated Hamiltonian \( H_{\mathcal{T}_k} \) constructed in the same way as in Eq.(8). One time-step evolution of the SQWH is then generated by the operator

\[
U = \prod_{k=0}^{N-1} e^{i \theta_k H_{\mathcal{T}_k}}
\]

with \( \theta_k \in [0, 2\pi] \) for all \( k \in \{0, \ldots, N-1\} \).

We are now ready to show how to translate the SQWH model into a QCA one. As previously, before giving a general recipe we first show a simple example of such translation.

### 3.2.2 One-dimensional example

In this example, we consider a SQWH over a 1-d lattice. The vertex set is \( V = \mathbb{Z} \), and thus \( \mathcal{H}_V = \text{span}(\{|i\rangle | i \in \mathbb{Z}\}) \). The smallest tessellation cover for such 2-regular (infinite) graph is composed of two tessellations \( \{\mathcal{T}_0, \mathcal{T}_1\} \), with \( \mathcal{T}_0 = \{\{\mathbb{Z}, i \mid i \in \mathbb{Z}\}\} \) and \( \mathcal{T}_1 = \{\{\mathbb{Z}+1 \mid i \in \mathbb{Z}\}\} \). These tessellations are shown in Fig. 3. Each tessellation induces a different partition of \( \mathcal{H}_V \) as \( \bigoplus_{\alpha_k \in \mathcal{T}_k} \mathcal{H}_{\alpha_k} \), with \( k \in \{0, 1\} \). For this example, we take general projectors into each polygon-subspace via the vectors

\[
|\{\text{odd} \}, i \rangle = a_0 |2i\rangle + \tilde{a}_0 |2i+1\rangle,
\]

\[
|\{\text{even} + 1 \}, i \rangle = a_i |2i\rangle + \tilde{a}_i |2i+2\rangle,
\]

for all \( i \in \mathbb{Z} \), and where the coefficients are constrained to \( |a_k|^2 + |\tilde{a}_k|^2 = 1 \) with \( k \in \{0, 1\} \). With these projectors we follow Eq.(8) to construct the evolution operator as

\[
U = e^{i \theta_1 H_{\mathcal{T}_1}} e^{i \theta_0 H_{\mathcal{T}_0}}.
\]

As the evolution propagator is composed by the product of similar operators, \( e^{i \theta_k H_{\mathcal{T}_k}} \), acting in similar ways in different partitions of \( \mathcal{H}_V \), below we only show how to translate a single of these operators, say \( e^{i \theta_0 H_{\mathcal{T}_0}} \), into automata language. Still within the SQWH language, with the aid of Eq. (9), the propagator in each subspace...
Let a general walker state at time $t$ be expressed as $|\psi(t)\rangle = \sum_{i \in \mathbb{Z}} \psi(i, t) |i\rangle$, where $\psi(i, t)$ is the probability amplitude for vertex $i$ at time $t$. The state after the action of $U_{\Sigma_0}$ is then given by

$$U_{\Sigma_0} |\psi(t)\rangle = \sum_{i \in \mathbb{Z}} \left\{ \left[ (e^{-i\theta_0} + 2i \sin(\theta_0)|a_0|^2)\psi(2i, t) + 2i \sin(\theta_0)a_0\tilde{a}_0^* \psi(2i + 1, t) \right] | 2i \rangle + \\
\left[ (e^{-i\theta_0} + 2i \sin(\theta_0)|\tilde{a}_0|^2)\psi(2i + 1, t) + 2i \sin(\theta_0)a_0\tilde{a}_0^* \psi(2i, t) \right] | 2i + 1 \rangle \right\}.$$  \hfill (11)

This is the evolution that we want to simulate within the quantum cellular automata model.

For the QCA simulation, in each lattice vertex we place one qubit. The two tessellations needed for the walker dynamics will now give us two tilings. The polygons of the first tessellation determine now the tiles of the first tiling: $T_i^{(0)} =\{ 2i, 2i + 1 \}$. Similarly, for the second tiling the correspondence implies the tiles $T_i^{(1)} =\{ 2i + 1, 2i + 2 \}$. There are two possibilities to define the cellular automaton cell: first is to take the tiles of the first tiling as forming a single cell with two subcells; second is to consider each vertex as a single cell with no subcell division. Both cases are equivalent, and we take the second choice as it simplifies the simulation description. The encoding of the walker state into the QCA framework is then simply given by $|i\rangle \rightarrow |0, -1, 1, 0, 1, 0, \ldots\rangle$, for all $i \in \mathbb{Z}$. A general state for the automaton at time $t$ is then written as $|\Psi(t)\rangle = \sum_{i \in \mathbb{Z}} \psi(i, t) |\ldots, 0, -1, 1, 0, 1, \ldots\rangle$, with $\psi(i, t)$ the amplitude of finding on “excitation” at the $i$-th qubit at time $t$. Now we need to simulate the evolution operator $U_{\Sigma_0}$ with the action of the first tiling. To that we note that each polygon in $\Sigma_0$ corresponds exactly to a tile in $\mathcal{T}_0$. Therefore, the propagator $U_{\Sigma_0} = \bigoplus_{a_0 \in \Sigma_0} U_{a_0}$ is then translated into $\bigotimes_{T_i^{(0)} \in \mathcal{T}_0} W_0$, where

$$W_0 = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & e^{-i\theta_0} + 2i \sin(\theta_0)|a_0|^2 & 2i \sin(\theta_0)a_0\tilde{a}_0^* & 0 \\
0 & 2i \sin(\theta_0)a_0^*\tilde{a}_0 & e^{-i\theta_0} + 2i \sin(\theta_0)|\tilde{a}_0|^2 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix},$$

when written in the computational basis $\{|0, 2i, 0, 2i + 1\}, |1, 2i, 0, 2i + 1\}, |0, 2i, 1, 2i + 1\}, |1, 2i, 1, 2i + 1\rangle$. With such a encoding, the evolution given by $\bigotimes_{T_i^{(0)} \in \mathcal{T}_0} W_0$ leads a general state of the automaton at time $t$ to the state:

\(\varepsilon\) Springer
\[
\sum_{i \in \mathbb{Z}} \left[ \left( (e^{-i\theta_0} + 2i \sin(\theta_0) |a_0|^2) \Psi(2i, t) + 2i \sin(\theta_0) a_0 a_0^* \Psi(2i, t) \right) \right. \\
\left. | \ldots , 0_{2i-1}, 1_{2i}, 0_{2i+1}, 0_{2i+2}, \ldots \rangle \\
\left. + \left( (e^{-i\theta_0} + 2i \sin(\theta_0) |a_0|^2) \Psi(2i+1, t) + 2i \sin(\theta_0) \tilde{a}_0 \tilde{a}_0^* \Psi(2i, t) \right) \right. \\
\left. | \ldots , 0_{2i-1}, 1_{2i}, 1_{2i+1}, 0_{2i+2} \ldots \rangle \right].
\] (12)

After decoding, this state is exactly equivalent to the SQWH state shown in Eq.(11).

This example shows that the tessellations in the SQWH play the role of the tilings in the QCA, and the set of operators \( \{ W_i \} \) in the QCA are obtained from the polygon-subspace operators associated with the tessellations of the SQWH.

### 3.2.3 General recipe

The general procedure to translate a SQWH into a PUQCA takes as input a \( d \)-regular graph \( G = G(V, E) \) (or lattice \( L \)), a tessellation cover \( \{ \mathcal{T}_k \}_{k=0}^{N-1} \) with polygons \( \alpha_k \) for each tessellation, a set of coefficients \( \{ a_k(i) \}_{i=0}^{|\alpha_k|} \), and a set of angles \( \{ \theta_k \}_{k=0}^{N-1} \). As result, the procedure outputs a well-formed PUQCA by following the subsequent recipe.

1. The PUQCA is established over the same graph \( G \) employed by the SQWH.
2. Since the graph is the same, the neighborhood scheme for the corresponding automaton is simply \( N_i = \{ j | (i, j) \in E \} \cup \{ i \} \).
3. No subcell structure is required for this translation (\( n = 0 \)). In each vertex we place a qubit, and thus \( \mathcal{H}_Z \cong \mathbb{C}^2 \). To each vertex state \( | i \rangle \) on the SQWH side, we associate the automaton state \( | \ldots, 0_{i-1}, 1_i, 0_{i+1}, \ldots \rangle \in \mathcal{H}_Z ^{\otimes |V|} \).
4. For each tessellation \( \mathcal{T}_k \) for the SQWH, we have an equivalent tiling \( \mathcal{R}_k \) in the PUQCA. Moreover, the vertices belonging to the polygon \( \alpha_k \) of the \( k \)-th tessellation yield the elements of the tile \( \mathcal{T}_k^{(l)} \).
5. With the input data, following the prescription of the SQWH, we construct the evolution operator \( U_{\mathcal{T}_k} = \bigoplus_{\alpha_k \in \mathcal{T}_k} U_{\alpha_k} \) to each tessellation. As the polygons \( \alpha_k \) are of equal size for a \( d \)-regular graph, the operators \( U_{\alpha_k} \) acting in each polygon-subspace are also identical. Such operator is then directly translated to the PUQCA model as the unitary function \( W_k \), via the unary encoding described in the item 3 above.

This completes the translation from the SQWH to the PUQCA model.

As we have done before, let us analyze the resources required by the PUQCA formalism for a given a SQWH dynamics. For a graph \( G = G(V, E) \), the Hilbert space associated with the SQWH is a \( |V| \)-dimensional one. For the PUQCA, we employed \( |V| \) qubits, yielding a \( 2^{|V|} \)-dimensional total space. However, here again, we only use the single-excitation subspace, which is \( |V| \)-dimensional. Once more, the effective amount of resources required by the PUQCA are the same as the original model.
4 Discussion and conclusions

In this work, we introduced a partitioned quantum cellular automata, the PUQCA, which is at the same time a well-formed quantum cellular automata, and also a conceptually simple and versatile construction. Such characteristics allowed us to show that various models of quantum walks can be readily translated into a PUQCA dynamics. This has the immediate advantage to employ qubits, the most common basic architecture of quantum computers. As such it is more convenient for near term experimental implementations of, for instance, quantum search algorithms. It is important to notice that the translation from quantum walk models to quantum cellular automata proposed here requires the same amount of effective dimensions.

One question that immediately raises from our translation results (QW→QCA) is whether quantum walks and quantum cellular automata are equivalent, i.e., if there is for every QCA an correspondent QW (QW←QCA). Of course this depends on how strict the definitions of the models are. For instance, it is usually accepted that in a coined quantum walk, the shift operator does not create superposition of the walker’s position states. When translating CQW into PUQCA, this implied that all the interactions between cells were simply swap gates. Therefore, if a given PUQCA has an interaction between cells other than a swap gate, our results suggest that there is no equivalent coined quantum walk to such a PUQCA. Another interesting perspective of our work is to allow for comparing different types of resources across different models. We can compare, for example, how the coherence created within the staggered quantum walk models is translated into entanglement between qubits in the quantum cellular automata model. Finally, it will be highly interesting to generalize our translation procedure in a way that it allows us to simulate many walkers that can interact [43–45]. Such a simulation, including particle statistics [46], will open the possibility of simulating many-body systems and quantum field theories [44,46] within the partitioned quantum cellular automata paradigm.

To conclude, we recall that the idea of quantum computers was arguably created by Feynman as a way (possibly the only one) to simulate complex quantum systems. In his seminal article entitled “Simulating Physics with Computers” [47], he constructs the idea of a simulator of physical systems from the idea of a cellular automata. At that time, quantum cellular automata were not yet invented. From there on, various definitions of QCA were developed, but the potential of quantum cellular automata models as simulators of complex quantum systems remains largely unexplored. We hope that our first results in this direction, together with a simple formulation of a QCA model and its possibility of implementation with current technology, will serve as the catalyst for the development of a whole new phenomenology of simulation of quantum systems.

Acknowledgements We would like to thank Osvaldo J. Farias for various discussions on the topic of quantum cellular automata. We acknowledge financial support from the National Institute for Science and Technology of Quantum Information (INCT-IQ/CNPq, Brazil).
References

1. Spitzer, F.: Principles of Random Walk, 2nd edn. Springer Science+Business Media, LLC, Berlin (1976)
2. Singal, V.: Beyond the Random Walk. Oxford University Press, Oxford (2004)
3. Shiller, R.J., Perron, P.: Testing the random walk hypothesis: power versus frequency of observation. Econ. Lett. 18(4), 381 (1985). https://doi.org/10.1016/0165-1765(85)90058-8
4. Fouss, F., Pirotte, A., Renders, J.M., Saerens, M.: Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Trans. Knowl. Data Eng. 19(3), 355 (2007). https://doi.org/10.1109/TKDE.2007.46
5. Feg, E., Landau, D.P.: Efficient, multiple-range random walk algorithm to calculate the density of states. Phys. Rev. Lett. 86, 2050 (2001). https://doi.org/10.1103/PhysRevLett.86.2050
6. Jones, R.A.: Soft Condensed Matter, vol. 6. Oxford University Press, Oxford (2002)
7. Aharonov, Y., Davidovich, L., Zagury, N.: Quantum random walks. Phys. Rev. A 48, 1687 (1993). https://doi.org/10.1103/PhysRevA.48.1687
8. Molfetta, G.D., Prez, A.: Quantum walks as simulators of neutrino oscillations in a vacuum and matter. New J. Phys. 18(10), 103038 (2016). https://doi.org/10.1088/1367-2630/18/10/103038
9. Arrighi, P., Facchini, S., Forets, M.: Quantum walking in curved spacetime. Quantum Inf. Process. 15(8), 3467 (2016). https://doi.org/10.1007/s11128-016-1335-7
10. Portugal, R.: Quantum Walks and Search Algorithm. Springer, Berlin (2013)
11. Szegedy, M.: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science. IEEE Computer Society, Washington, DC, USA, FOCS’04, pp. 32–41 (2004). https://doi.org/10.1109/FOCS.2004.53
12. Portugal, R., Santos, R.A.M., Fernandes, T.D., Gonçalves, D.N.: The staggered quantum walk model. Quantum Inf. Process. 15(1), 85 (2016). https://doi.org/10.1007/s11128-015-1149-z
13. Portugal, R., de Oliveira, M.C., Moqadam, J.K.: Staggered quantum walks with Hamiltonians. Phys. Rev. A 95, 012328 (2017). https://doi.org/10.1103/PhysRevA.95.012328
14. Childs, A.M.: Universal computation by quantum walk. Phys. Rev. Lett. 102, 180501 (2009). https://doi.org/10.1103/PhysRevLett.102.180501
15. Childs, A.M., Gosset, D., Webb, Z.: Universal computation by multi-particle quantum walk. Science 339, 791 (2013). https://doi.org/10.1126/science.1229957
16. Zähringer, F., Kirchmair, G., Gerritsma, R., Solano, E., Blatt, R., Roos, C.F.: Realization of a quantum walk with one and two trapped ions. Phys. Rev. Lett. 104, 100503 (2010). https://doi.org/10.1103/PhysRevLett.104.100503
17. Dür, W., Raußendorf, R., Kendon, V.M., Briegel, H.J.: Quantum walks in optical lattices. Phys. Rev. A 66, 052319 (2002). https://doi.org/10.1103/PhysRevA.66.052319
18. Wang, K.M.: Physical Implementation of Quantum Walks. Springer, Berlin (2013)
19. Chopard, B., Droz, M.: Cellular Automata Modeling of Physical Systems. Cambridge University Press, Cambridge (2005)
20. Wolfram, S.: A New Kind of Science. Wolfram Media, Champaign (2002)
21. von Neumann, J.: Theory of Self-Reproducing Automata. University of Illinois Press, Urbana (1996)
22. Nandi, S., Kar, B.K., Chaudhuri, P.P.: Theory and applications of cellular automata in cryptography. IEEE Trans. Comput. 43(12), 1346 (1994). https://doi.org/10.1109/12.338094
23. Frisch, U., Hasslacher, B., Pomeau, Y.: Lattice-gas automata for the Navier–Stokes equation. Phys. Rev. Lett. 56, 1505 (1986). https://doi.org/10.1103/PhysRevLett.56.1505
24. Green, D.G.: Cellular automata models in biology. Math. Comput. Model. 13(6), 69 (1990). https://doi.org/10.1016/0895-7177(90)90010-K
25. Grössing, G., Zeilinger, A.: Quantum cellular automata. Complex Syst. 2(2), 197 (1988). http://www.complex-systems.com/abstracts/v02_i02_a04/
26. Meyer, D.A.: From quantum cellular automata to quantum lattice gases. J. Stat. Phys. 85(5), 551 (1996). https://doi.org/10.1007/BF02199356
27. Wiesner, K.: Quantum Cellular Automata, pp. 2351–2360. Springer, New York (2012). https://doi.org/10.1007/978-1-4614-1800-9_146
28. Schumacher, B., Werner, R.F.: Reversible quantum cellular automata, arXiv preprint arXiv:quant-ph/0405174 (2004)
29. Arrighi, P., Grattage, J.: Partitioned quantum cellular automata are intrinsically universal. Nat. Comput. 11(1), 13 (2012). https://doi.org/10.1007/s11047-011-9277-6
30. Boixo, S., Rønnow, T.F., Isakov, S.V., Wang, Z., Wecker, D., Lidar, D.A., Martinis, J.M., Troyer, M.: Evidence for quantum annealing with more than one hundred qubits. Nat. Phys. 10, 218 (2014). https://doi.org/10.1038/nphys2900

31. Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J.M., Gambetta, J.M.: Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature 549, 242 (2017). https://doi.org/10.1038/nature23879

32. Foxen, B., Mutus, J.Y., Lucero, E., Graff, R., Megrant, A., Chen, Y., Quintana, C., Burkett, B., Kelly, J., Jeffrey, E., Yang, Y., Yu, A., Arya, K., Barends, R., Chen, Z., Chiaro, B., Dunsworth, A., Fowler, A., Girvin, C., Giustina, M., Huang, T., Klimov, P., Neely, C., Roushan, P., Sank, D., Vainsencher, A., Wenner, J., White, T.C., Martinis, J.M.: Qubit compatible superconducting interconnects. Quantum Sci. Technol. 3(1), 014005 (2018). https://doi.org/10.1088/2058-9565/aa94fc

33. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. Nature 299, 802 (1982). https://doi.org/10.1038/299802a0

34. Toffoli, T., Margolus, N.: Cellular Automata Machines. Series in Scientific Computation. MIT Press, Cambridge (1987)

35. Pérez-Delgado, C.A., Cheung, D.: Local unitary quantum cellular automata. Phys. Rev. A 76, 032320 (2007). https://doi.org/10.1103/PhysRevA.76.032320

36. Watrous, J.: Proceedings of IEEE 36th Annual Foundations of Computer Science, pp. 528–537 (1995). https://doi.org/10.1109/SFCS.1995.492583

37. Philipp, P., Portugal, R.: Exact simulation of coined quantum walks with the continuous-time model. Quantum Inf. Process. 16(1), 14 (2016). https://doi.org/10.1007/s11128-016-1475-9

38. Gross, D., Nesme, V., Vogts, H., Werner, R.F.: Index theory of one dimensional quantum walks and cellular automata. Commun. Math. Phys. 310(2), 419 (2012). https://doi.org/10.1007/s00220-012-1423-1

39. Portugal, R., Boettcher, S., Falkner, S.: One-dimensional coinless quantum walks. Phys. Rev. A 91, 052319 (2015). https://doi.org/10.1103/PhysRevA.91.052319

40. Portugal, R.: Staggered quantum walks on graphs. Phys. Rev. A 93, 062335 (2016). https://doi.org/10.1103/PhysRevA.93.062335

41. Portugal, R., Fernandes, T.D.: Quantum search on the two-dimensional lattice using the staggered model with Hamiltonians. Phys. Rev. A 95, 042341 (2017). https://doi.org/10.1103/PhysRevA.95.042341

42. Khatibi Moqadam, J., de Oliveira, M.C., Portugal, R.: Staggered quantum walks with superconducting microwave resonators. Phys. Rev. B 95, 144506 (2017). https://doi.org/10.1103/PhysRevB.95.144506

43. Ahlbrecht, A., Alberti, A., Meschede, D., Scholz, V.B., Werner, A.H., Werner, R.F.: Molecular binding in interacting quantum walks. New J. Phys. 14(7), 073050 (2012). https://doi.org/10.1088/1367-2630/14/7/073050

44. Bisio, A., D’Ariano, G.M., Perinotti, P.: Quantum cellular automaton theory of light. Ann. Phys. 368, 177 (2016). https://doi.org/10.1016/j.aop.2016.02.009

45. Bisio, A., D’Ariano, G.M., Perinotti, P., Tosini, A.: Thirring quantum cellular automaton. Phys. Rev. A 97, 032132 (2018). https://doi.org/10.1103/PhysRevA.97.032132

46. Bisio, A., D’Ariano, G.M., Tosini, A.: Quantum field as a quantum cellular automaton: the Dirac free evolution in one dimension. Ann. Phys. 354, 244 (2015). https://doi.org/10.1016/j.aop.2014.12.016

47. Feynman, R.P.: Simulating physics with computers. Int. J. Theor. Phys. 21(6), 467 (1982). https://doi.org/10.1007/BF02650179