On generalization of reversible second-order cellular automata

Alexander Yu. Vlasov

March 25, 2014

Abstract

A cellular automaton with \( n \) states may be used for construction of reversible second-order cellular automaton with \( n^2 \) states. Reversible cellular automata with hidden parameters discussed in this paper are generalization of such construction and may have number of states \( N = nm \) with arbitrary \( m \). Further modification produces reversible cellular automata with reduced number of states \( N' < N = nm \).

1 Introduction

Second-order cellular automaton (CA) may be used for construction of reversible CA [1, 2]. The approach let us construct reversible CA with set of states \( D = S \times S \simeq \mathbb{Z}_{n^2} \) from arbitrary CA with states from some \( S \simeq \mathbb{Z}_n \). In the paper is suggested extension of such idea with \( D = S \times H \simeq \mathbb{Z}_{nm} \) with further reduction to some \( R \subset D \). Here instead of second term \( S \) corresponding to previous state is used arbitrary set \( H \) considered as hidden parameter.

Some basic structures are introduced in Sec. 2, yet a preliminary acquaintance with properties of CA is supposed. The construction of reversible second-order CA is revisited in Sec. 3. Generalizations \( D = S \times H \) and \( R \subset D \) are introduced in Sec. 4 and Sec. 5 respectively. Some examples are presented for clarification in Sec. 6.

2 Preliminaries

A space of cells \( X \) is defined here in a general way as a finite or infinite discrete set. Simple examples of such spaces are \( \mathbb{Z}^d \) and some subset of such lattices. Element \( x \in X \) is called here location (of cell).
Set of states of each cell is a finite discrete set $S$ with $n$ elements. A map $I_S : S \to \mathbb{Z}_n$ enumerates the states by indexes $0 \ldots n - 1$.

Complete description of each cell is a pair $(s, x)$ also denoted here as $(s)_x$ with $x \in X$, $s \in S$. Configuration of whole CA is a map $S^X : X \to S$.

For a space of cells such as $X = \mathbb{Z}^d$ the neighborhood of given cell $\vec{x} \in \mathbb{Z}^d$ with $j$ adjacent cells is often described via shifts $(\vec{x} + \vec{v}_1, \ldots, \vec{x} + \vec{v}_j)$ [2, 3], but for more general $X$ it should be defined as some function $\nu : X \to X^j$.

Local update rule is a function $f : S^{j+1} \to S$, where $(s, s_1, \ldots, s_j) \in S^{j+1}$ is the vector of states for a neighborhood together with the cell itself. For a cell $(s)_x$ the notation $f[s]_x$ or $f[s]$ may be used for $f(s, s_1, \ldots, s_j)$.

Consideration of cellular automata on Penrose’s tilings [4] demonstrates that such a definition of neighborhood and local update rules may be still not enough to describe some models. The problem is not only varying number $j_x$ of adjacent cells, but also a few possible geometries and orientations of the neighborhood.

To include such models, let us use formal notation $[S]$ for a domain of definition of local update rule. Element of such set is denoted here as $[s] \in [S]$. In simplest case discussed above $[S] = S^{j+1}$, $[s] = (s, s_1, \ldots, s_j)$. For variable size of neighborhood it may be defined $[S] = \bigcup_k S^{j_k+1}$, $[s] = (k, (s, s_1, \ldots, s_{j_k}))$.

For any location $x$ and each configuration $S^X$ localization is a map

$$\Xi : (S^X, x) \to [S].$$

The value of such a map for given $x$ is designated further as $[s]_x$ and for local update function may be used notation $f[s]$ or $f[s]_x$ already introduced earlier.

In simpler and more regular cases localization $\Xi$ of configuration $S^X$ may be defined as composition of two maps. First map $\nu$ gives neighboring cells for given $x$. Localization $S^{j+1} = [S]$ is a restriction of configuration $S^X$ to given neighborhood.

3 Second-order cellular automata

Let us consider arbitrary CA with set of states $S = \mathbb{Z}_n$ and local update rule $f$. Second-order CA [1] produced from such a rule has the same set of locations $X$, but the state is a pair from set $D = S \times S$ with $n^2$ elements and update rule is

$$f_R : (s, s') \mapsto (f[s] \ominus s', s),$$

2
where ‘⊖’ is subtraction modulo $n$.

The update rule Eq. (2) may be decomposed into two reversible steps, an update:

$$f_u : (s, s') \mapsto (s, f[s] \ominus s')$$

(3)

and the swap:

$$w : (s, s'') \mapsto (s'', s).$$

(4)

The rule $f_R = w f_u$ is reversible $f_R^{-1} = f_u^{-1} w^{-1}$, where $w^{-1} = w$, $f_u^{-1} = f_u$.

The global reversibility of local operation $w$ Eq. (4) is rather obvious, because it acts on each cell separately. Operation $f_u$ Eq. (3) uses values of neighboring cells, but it acts only on second parameter of the state. Due to construction of the $f_u$ Eq. (3) neighboring cells are not affected by the second parameter (before application of $w$) and such a property produces global reversibility of $f_u$ in spite of the fact that definition Eq. (3) is local.

Generalization for less regular spaces $X$ with different configurations and number of neighboring cells is rather straightforward with localization $\Xi$ of configuration $D^X$ Eq. (1). In fact, it is enough to consider localization $\Xi$ of $S^X$ to construct $f_u$ Eq. (3) and swap $w$ only acts on cell itself.

4 Generalization with hidden parameters

The term second-order describes possibility to use information about previous state [1]. In local update rule such as Eq. (2) the information is accessible only to cell itself. It is possible to consider yet another interpretation of such update rules. The set of states is direct product of two components, but only one is accessible for neighboring cells.

Such alternative approach may be used for generalizations, if to consider the second component not as previous state, but as an arbitrary hidden parameter of a cell inaccessible for neighbors. In such a case size of both components may be different and instead of update Eq. (3) and exchange Eq. (4) could be used more general functions.

Let us consider CA with states from set of pairs $D = S \times H$, $S = \mathbb{Z}_n$, $H = \mathbb{Z}_m$ and update rule represented as composition of two steps

$$f_D = \varpi f_u.$$  

(5)

Here $\varpi$ is a fixed reversible transformation of $D$ and

$$f_u : (s, h) \mapsto (s, F[s](h)),$$

(6)
where \( F[s](h) = F(s, s_1, \ldots, s_j)(h) \) must be reversible function on set \( H \) for any vector of states \( (s, s_1, \ldots, s_j) \).

In fact, not only second-order CA is particular case of such extension for \( H = S \), but also arbitrary \( n \)-order CA may be considered in such a way with \( H = S^{n-1} \) storing \( n - 1 \) previous states and \( D = S \times H = S^n \).

In most general case, reversible functions on \( H \) may be represented via transpositions \( \pi_m \) of \( m \) elements of \( H \) and so instead of \( f : S^{j+1} \to S \) (cf usual CA recollected in Sec. 2) should be used map \( F \)

\[
F : S^{j+1} \to \pi_m.
\]

Only for set with two states both \( S = \mathbb{Z}_2 \) and \( \pi_2 \) have two elements and such distinction is not essential.

Unlike the usual second-order CA discussed in Sec. 3 such an extended version is not necessary directly derived from some CA. On the other hand, it may be useful sometimes to start with a second-order CA and to consider local update rule with bigger number of output states Eq. (7). Simple example is discussed further in Sec. 6.2 for CA with three states.

Construction for spaces \( X \) with variable number of neighboring cells is also straightforward due to localization \( \Xi \) of configuration \( D^X \) Eq. (1). It is again enough to consider localization of \( S^X \) to construct \( F \) Eq. (7) and \( \varpi \) affects only state of cell itself.

## 5 Reduced number of states

Set with \( nm \) states used above may be redundant. Reduction of previous construction on some subset \( R \subset S \times H \) is discussed below.

Let us consider some discrete set \( R \) with two projections \( p_1 : R \to S \) and \( p_2 : R \to H \). So, each state \( r \in R \) corresponds to pair \( r \simeq (p_1(r), p_2(r)) \).

As an example, Fig. 1 with 12 states may be represented as set of pairs: \{\((0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (0, 1), (1, 1), (1, 3), (0, 2), (1, 2), (0, 3), (0, 4)\}\.

Let us denote \( m(s) \) number of states \( r \) with \( p_1(r) = s \), e.g., \( m(0) = 5, m(1) = 3, m(2) = m(4) = 1, m(3) = 2 \) for Fig. 1.

Due to representation of states from \( R \) via set of pairs it is possible again to use two-steps rule such as Eq. (5) with \( \varpi \) is transposition on \( R \) and \( f_\nu \) is an analogue of Eq. (6)

\[
f_\nu : (p_1(r), p_2(r)) \mapsto (p_1(r), F[p_1(r)](p_2(r))),
\]

where \( F[p_1(r)] = F(p_1(r), p_1(r_1), \ldots, p_1(r_j)) \) is reversible function (transposition) on set with \( m(p_1(r)) \) states. For example on Fig. 1 function \( F[0] \)
Figure 1: Scheme with reduced set of states

may exchange only five states \( \{0, 5, 8, 10, 11\} \), \( F[1] = \{1, 6, 9\} \), etc. The \( F \)
depends only on \( p_1 \) projections of states in the neighborhood and rearranges
states of the cell without change of \( p_1 \).

There is a formal difficulty in description of \( F \) in Eq. (8). A map such
as Eq. (7) may not be used directly, because value \( m \) in \( \pi_m \) for a state \( r \in R \)
depends on \( p_1(r) = s \in S \).

A simple way to resolve such a problem — is to consider bigger formal
set of states \( S \times H \supseteq R \), Fig. 2, but to use functions \( F \) and \( \varpi \) changing only
states from \( R \). In such a case it is possible to use \( F : S^j \rightarrow \pi_{m_{\text{max}}} \), where
\( m_{\text{max}} \geq m(s) \) for any \( s \).

Another method is to consider a family of functions

\[
F_s : S^j \rightarrow \pi_{m(s)}.
\]  

(9)

Here \( S^j \) is used instead of \( S^{j+1} \), because state of cell itself formally is not
used as an argument of a function \( F_s \) and is treated in a special way as an
index inside of the family.

It is not necessary to introduce formal set \( S \times H \) with auxiliary states
in such a case. Such approach may also require an insignificant change of
notation such as \( F[s] = F_s(s_1, \ldots, s_j) \) instead of \( F[s] = F(s, s_1, \ldots, s_j) \).
The construction of CA with variable number of neighboring cell may be again carried out with localization $\Xi$ of configuration $R^X$ or $S^X$ Eq. (1). It works both with formal consideration of $R$ as a reduction of auxiliary space $S \times H$ and with definition of $R$ via family of functions $F_s$.

6 Examples

6.1 CA with $2 \times 2 = 4$ states

Let us consider two-states cellular automaton with transition function $f$. For two states there is no difference between subtraction and addition modulo two, the same binary operation is also known as XOR (eXclusive OR) and Eq. (2) may be written in more familiar way:

$$f_R : (s, s') \mapsto (f[s] \text{XOR} s', s).$$  \hspace{1cm} (10)

The Eq. (10) is transition function for reversible cellular automaton with four states Fig. 3.

The local transition function of initial CA may have only two values and they realize two possible reversible function on set with two elements.
Figure 3: Second order CA from CA with two states

Eq. (10)

\[
\begin{array}{c|c|c}
 f & s' & f \text{XOR} s' \\
\hline
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
\end{array}
\]  

(11)

With lack of other generalizations of function \( f_\nu \) Eq. (6) for two states only \( \varpi \) may be changed.

6.2 CA with \( 3 \times 3 = 9 \) states

Let us consider for comparison initial CA with three states.

\[
f_R : (s, s') \mapsto (f[s] + 3 - s' \mod 3, s).
\]  

(12)

The Eq. (12) is transition function for reversible cellular automaton with nine states Fig. 4.

Only three between six reversible transformations (\( 3! = 6 \)) are represented in Eq. (12) and so it is possible to consider extensions with six different \( f_\nu \) Eq. (6). For three states all six possible functions may be written quite naturally, e.g.

\[
\begin{align*}
f_0(h) &= h \\
f_1(h) &= h + 1 \mod 3 \\
f_2(h) &= h + 2 \mod 3 \\
f_3(h) &= 3 - h \mod 3 \\
f_4(h) &= 4 - h \mod 3 \\
f_5(h) &= 2 - h.
\end{align*}
\]  

(13a)-(13f)
It is clear, that only $f_3$, $f_4$ and $f_5$ are used in Eq. (12). Even without change of 'swap' function $\varpi$ there are possible extensions for such CA, due to possibility to associate six different actions Eq. (13) with local update rule instead of three. The inverse for $f_3$, $f_4$, $f_5$ and $f_0$ — is the function itself and $f_{1}^{-1} = f_2$.

6.3 Example of $\varpi$ for CA with $3 \times 2 = 6$ states

In examples above the operation $\varpi$ in Eq. (5) may simply exchange two components of the state because $H = S$. Such a swap for CA with $2 \times 2$ states is shown on Fig. 5.

Let us consider $3 \times 2$ CA with six states, where $S = (0, 1, 2)$, $H = (0, 1)$. An example of operation $\varpi$ for such CA is shown on Fig. 6.
An example of application of constructions discussed in Sec. 5 is considered below. Let us start with some standard or extended second-order CA based on two-states CA such as Sec. 6.1.

It is useful sometimes to add special “blank” cells. A cellular space of CA may be restricted to some domain of arbitrary shape if to set all complementary cells to such “blank” value. Such a state may not be changed, but other states must be updated by the update function of initial CA without “blank” state.

On the Fig. 7 is shown extension of $2 \times 2$ CA with “blank” state (4). The function $\varpi$ is denoted by arrows and exchanges only two states (1 ↔ 2),
the function $f_\nu$ should be derived from initial CA and may change only first four states.

It should be mentioned, that term “reduced number of states” here should be treated with some care — given example could be formally described as reduction of CA with $6 = 2 \times 3$ states, but it would not reflect properly idea of “blank” cells used for construction of the CA.

References

[1] T. Toffoli and N. Margolus, “Invertible cellular automata: a review,” *Physica D* **45**, 229–253 (1990).

[2] S. Wolfram, *Cellular automata and complexity: Collected papers*, (Addison-Wesley, Reading MA 1994).

[3] J. Kari and S. Taati, “Statistical mechanics of surjective cellular automata,” arXiv:1311.2319 [math] (2013).

[4] N. Owens and S. Stepney, “Investigations of Game of Life cellular automata rules on Penrose tilings: lifetime and ash statistics,” *Automata 2008, Bristol, UK* pp. 1-34 (Luniver Press, 2008).