Partitioning of Cellular Automata Rule Spaces

Rodrigo A. Obando
TSYS School of Computer Science
Columbus State University
4225 University Avenue
Columbus, GA 31907, USA
Obando_Rodrigo@ColumbusState.edu

Even though cellular automata rules are simple and deterministic, there has been no way to know the class of behavior from the rule itself until it is evolved. This is not a real problem in the elementary rule space, since there is a relatively small number of rules in it. An extensive survey of their behavior has been done already. The problem arises when the neighborhood is expanded, such as for $r = 2$, where the number of rules becomes 2^{32} and an exhausting analysis becomes extremely time consuming. This problem is aggravated even more for $k = 3$ and $r = 1$, where the number of rules is 3^{27}. A partition of the rule space is presented based on new parameters λ_0 and λ_1, along with the primitives p_0 and p_1 and the Gray code permutation. This partition helps identify regions of similar global behavior in the rule space. The approach is illustrated for $k = 2$ and integer r, but it should be able to be extended to other values.

1. Introduction

Cellular automata are simple systems that can produce complex behavior and are ideal for the study of a great variety of topics, such as thermodynamics [1], biological systems [2], landscape change [3], and others. These automata may be defined for one, two, or more dimensions, as well as for k cell values and for a neighborhood of size r. The elementary cellular automata (ECA) are the simplest of the spaces that produce interesting behavior. This rule space is one dimensional, with $k = 2$ and $r = 1$. Given these parameters, there are 256 elementary rules, although only 88 of them are nonequivalent classes. Each rule may be encoded in a byte where each individual bit indicates the action of the rule for a given combination of the input. In the case of elementary rules, the input is the cell to be updated (central cell) and the two neighboring cells, one to the left and one to the right of the central cell.

https://doi.org/10.25088/ComplexSystems.24.1.27

A one-dimensional cellular automaton is used to update a row of cells. Each cell is updated individually and at the same time. Each step in the evolution of the automaton updates all the cells in the row. The original row of cells is the input to the automaton. Each one of these rules produces a different behavior given either a simple input, such as a single black cell on a white background, or a disordered input, such as that arising from a random number generator. A classification of the various behaviors, given a disordered input, was given by Wolfram [4] as classes 1, 2, 3, and 4. Class 1 relates to an automaton that evolves to a homogeneous state. Class 2 evolves simple separated periodic structures. Class 3 evolves into chaotic aperiodic patterns. Finally, class 4 generates complex patterns of localized structures. An updated survey on classification of the cellular automata has been done by Martinez [5].

Even though these rules are simple and deterministic, there has been no way to know the class of behavior from the rule itself until it is evolved. This is not a real problem in the elementary rule space, since there is a relatively small number of rules in it. An extensive survey of their behavior has been done already. The problem arises when the neighborhood is expanded, such as for $r = 2$, where the total number of rules becomes 2^{32} and an exhausting analysis becomes extremely time consuming. This problem is aggravated even more for $k = 3$ and $r = 1$, where the number of rules is 3^{27}.

There have been different parametric approaches to try to understand how the rules are distributed within a rule space. One of the oldest approaches is by Langton [6], where he defines the parameter λ, which basically stands for the number of ones in the rule with respect to the total possible number of ones. This parameter varies from 0.0 to 1.0. Other classifications were also defined by Li and Packard in [7]. Another approach has been performed by de Oliveira et al. [8], where five different parameters are defined and used for finding rules with certain characteristics. There are some other approaches, such as the one proposed by Cattaneo et al. [9], which explores equivalent classes through double permutations. Yet another approach that also explores the dynamic characteristics of rules was done by Mizraji [10], where the functions that define the rules in the ECA are analyzed in terms of what he calls dyadic spectra. This approach is related to the approach in this paper, in the sense that it splits the rule function into two subfunctions but with respect to a moving cell. The approach in the present paper maintains the central cell as a constant reference, and the overall approach diverges from the one followed in [10].

A better understanding of the distribution of these behavior classes is required for the exploration of these big spaces. This was also set forth by Wolfram [4] in his open question, how is different behavior

https://doi.org/10.25088/ComplexSystems.24.1.27

distributed in the space of cellular automaton rules? This particular question has been addressed by Zenil in [11], using two complexity measures. The current paper takes a different approach in providing a framework for exploration. This requires the definition of rule properties that would allow a partition of the rule space in a way that highlights the possibility of finding regions of these different global behaviors without necessarily defining which classification to use. It is well known that we are dealing with undecidability and reachability issues when defining the behavior of the rules, and this paper only provides a framework to investigate how rules relate to each other in their global behavior.

2. Rule Equations

A one-dimensional cellular automaton is a set of cells with k possible values, where each cell is updated based on its value and the values of its neighboring cells. We normally designate such a neighborhood by its radius r. The following discussion is specifically for $k = 2$ and r being an integer.

Each rule in a cellular automata rule space can be expressed as a function of the central cell a_0 and the cells included to the left and to the right, up to the radius r number of cells. This equation is denoted as ϕ, and the 256 equations for the ECA are tabulated in [4, 12].

A given value of r defines the set of all Boolean functions of v variables $\phi : \{0, 1\}^v \to \{0, 1\}$ denoted by $B_{v,1}$ or simply B_v as defined in [13], with $v = 2\,r + 1$:

$$B_v = \left\{\phi_0, \phi_1, \ldots, \phi_{2^{2^v}-1}\right\}.$$

We index each function in the customary way, using the bits that define the mapping and expressing that as a decimal number.

The cardinality of this set is

$$\left|B_v\right| = 2^{2^v}.$$

We denote the variables as $a_{-r}, \ldots, a_0, \ldots, a_r$ for B_v and $a_{-r}, \ldots, a_r$ except a_0 for B_{v-1}. Given this, we are able to express $\phi_q \in B_v$ in terms of $\phi_i, \phi_j \in B_{v-1}$ as

$$\phi_q = \overline{a}_0\, \phi_i + a_0\, \phi_j. \tag{1}$$

We note that $q \in \left\{0, \ldots, 2^{2^v}\right\}$ and $i, j \in \left\{0, \ldots, 2^{2^{v-1}}\right\}$. We call ϕ_i and ϕ_j the primitives of ϕ_q because all functions in B_v can be generated by the functions in B_{v-1} using equation (1). ϕ_i corresponds to

https://doi.org/10.25088/ComplexSystems.24.1.27

the behavior of the rule when $a_0 = 0$ and will be labeled p_0. ϕ_j corresponds to the behavior of the rule when $a_0 = 1$ and will be labeled p_1.

Table 1 shows all the primitive functions for B_3. We can express the function of any rule in the space B_3 by using these primitive functions. For example, rule ϕ_{110} has primitives $p_0 = \phi_{10}$ and $p_1 = \phi_7$, and the function for the rule is

$$\phi_{110} = \bar{a}_0\, \phi_{10} + a_0\, \phi_7 = \bar{a}_0\, a_1 + a_0(\bar{a}_{-1} + \bar{a}_1) = \bar{a}_0\, a_1 + a_0\, \bar{a}_{-1} + a_0\, \bar{a}_1.$$

i	ϕ_i	i	ϕ_i
0	0	8	$a_{-1}\, a_1$
1	$\bar{a}_{-1}\, \bar{a}_1$	9	$\bar{a}_{-1}\, \bar{a}_1 + a_{-1}\, a_1$
2	$\bar{a}_{-1}\, a_1$	10	a_1
3	$\bar{a}_{-1}$	11	$\bar{a}_{-1} + a_1$
4	$a_{-1}\, \bar{a}_1$	12	a_{-1}
5	$\bar{a}_1$	13	$a_{-1} + \bar{a}_1$
6	$\bar{a}_{-1}\, a_1 + a_{-1}\, \bar{a}_1$	14	$a_{-1} + a_1$
7	$\bar{a}_{-1} + \bar{a}_1$	15	1

Table 1. Primitive equations for B_3.

A simpler example is the function for rule ϕ_{15}, which is the complement of a right-shift operation. The primitives are $p_0 = \phi_3$ and $p_1 = \phi_3$, and the function for the rule is

$$\phi_{15} = \bar{a}_0\, \phi_3 + a_0\, \phi_3 = \bar{a}_0\, \bar{a}_{-1} + a_0\, \bar{a}_{-1} = \bar{a}_{-1}(\bar{a}_0 + a_0) = \bar{a}_{-1}.$$

Rule ϕ_{105} has more active variables. It has primitives $p_0 = \phi_9$ and $p_1 = \phi_6$, and the function is

$$\begin{aligned}
&\phi_{105} = \bar{a}_0\, \phi_9 + a_0\, \phi_6 = \\
&\quad \bar{a}_0(\bar{a}_{-1}\, \bar{a}_1 + a_{-1}\, a_1) + a_0(\bar{a}_{-1}\, a_1 + a_{-1}\, \bar{a}_1) = \\
&\quad \bar{a}_0\, \bar{a}_{-1}\, \bar{a}_1 + \bar{a}_0\, a_{-1}\, a_1 + a_0\, \bar{a}_{-1}\, a_1 + a_0\, a_{-1}\, \bar{a}_1.
\end{aligned}$$

We can generate all 256 functions tabulated in [4, 12] by simply applying equation (1) to the Cartesian product of the primitive equations in Table 1. This can be generalized for any other $k = 2$ cellular automata rule space $B_{2\,r+1}$.

Let us define λ_0 and λ_1 similarly to Langton's λ [6]. The original definition of λ is the ratio of the number of ones in the rule definition to the total possible number of ones. Instead we define λ_0 and λ_1 as

https://doi.org/10.25088/ComplexSystems.24.1.27

integers as follows:

$$\lambda_0 = \text{number of ones in the binary representation of the primitive,}$$

$$\lambda_1 = \text{number of zeros in the binary representation of the primitive.}$$

Let us look at how the primitive equations get partitioned using λ_0 and λ_1. Table 2 shows this partition. We may observe that λ_0 and λ_1 partition the primitive equations in such a manner that all equations within a group have the same general form.

λ_0	λ_1	i	ϕ_i
0	4	0	0
1	3	1	$\overline{a}_{-1}\,\overline{a}_1$
		4	$a_{-1}\,\overline{a}_1$
		8	$a_{-1}\,a_1$
		2	$\overline{a}_{-1}\,a_1$
2	2	6	$\overline{a}_{-1}\,a_1 + a_{-1}\,\overline{a}_1$
		9	$\overline{a}_{-1}\,\overline{a}_1 + a_{-1}\,a_1$
2	2	3	$\overline{a}_{-1}$
		5	$\overline{a}_1$
		12	a_{-1}
		10	a_1
3	1	7	$\overline{a}_{-1} + \overline{a}_1$
		13	$a_{-1} + \overline{a}_1$
		14	$a_{-1} + a_1$
		11	$\overline{a}_{-1} + a_1$
4	0	15	1

Table 2. Partition of primitive equations in B_3.

An interpretation of λ_0 and λ_1 is as follows. If a primitive has a high λ_0 and we use it as p_0, then the rule will have higher dynamic behavior. If a primitive has a high λ_1 and we use it as p_1, then the rule will also have a higher dynamic behavior. Each primitive has both λ_0 and λ_1, but the effect will depend on whether we use the primitive as p_0 or p_1. Both λ_0 and λ_1 are related to each other. We may also normalize the values of λ_0 and λ_1 similarly to Langton's λ [6], and this is particularly useful when referring to rules in rule spaces with different radius r. For easier development of findings, we do not normalize these properties. These parameters are not based on Langton's lambda but only compared to it. Part of the reason that Langton's measure would not work is that ones in a rule are significant only in certain bit positions. The parameters presented here address this problem by dividing the rule into two parts and having the number of zeros be important for one half and the number of ones be important for the other half.

Notation. We will use a shorthand notation for equation (1) for the rest of the paper as follows:

$\phi_q = \overline{a}_0\, \phi_i + a_0\, \phi_j$ will be equivalent to

$$q = \varphi(i, j, r),$$

where r is the radius that defines the rule space.

The examples we used before are expressed in this form as follows:

$$110 = \varphi(10, 7, 1)$$
$$15 = \varphi(3, 3, 1)$$
$$105 = \varphi(9, 6, 1).$$

Let us examine some examples of creating rules. Let us use primitives that will provide the least amount of dynamic behavior and determine the resulting rule. Looking at Table 2, we find that the primitive with the smallest λ_0 is 0 and the primitive with the smallest λ_1 is 15. The expected rule is one that will not have dynamic behavior:

$$\varphi(0, 15, 1) = 204.$$

Rule 204 in the elementary rule space is the identity function, and the state of the cells will not change. This result makes sense given the selected primitives. Let us now select the primitives with the highest dynamic behavior:

$$\varphi\,(15, 0, 1) = 51.$$

Rule 51 in the elementary rule space is the complement function, and the state of the cells will change in every update. This result also follows from the selection of the primitives. Other examples are

$$\varphi(0, 0, 1) = 0,$$
$$\varphi(15, 15, 1) = 255.$$

The output for the rules 0 and 255 will reach 0 and 1, respectively, for every cell in the automaton. Primitive 0 has $\lambda_0 = 0$ and $\lambda_1 = 4$, which indicate that once the cell value is 0, there will be no change, and if the initial value of the cell is 1, then the next value is always 0. All the cell outputs will become 0. Likewise, for rule 255 all cells will become 1.

https://doi.org/10.25088/ComplexSystems.24.1.27

Table 3 shows all four of these rules as a Cartesian product of the primitives.

$p_1 \downarrow p_0 \rightarrow$	0	15
15	$\varphi(0, 15, 1) = 204$	$\varphi(15, 15, 1) = 255$
0	$\varphi(0, 0, 1) = 0$	$\varphi(15, 0, 1) = 51$

Table 3. The four extreme rules in the ECA for λ_0 and λ_1.

We may characterize each rule not only by its primitives p_0 and p_1, but also by the λ_0 of p_0 and the λ_1 of p_1. This characterization allows us to define the partition in Section 3.

3. Partition of the Rule Space and Primitive Space

The partition described in this section is for any rule space such that $k = 2$. We may partition the rule space by using the values of λ_0 and λ_1 for each rule. This partition is defined as

$$\Pi = \{P_{m,n}\} \text{ for all } m, n \in \left\{0, \ldots, \frac{b}{2}\right\}, \tag{2}$$

where b is the number of bits in the rule and $P_{m,n}$ is a set of rules with $\lambda_0 = m$ and $\lambda_1 = n$. There are a total of 25 sets in Π for the elementary rule space, and this may be calculated for rule spaces with b number of bits as

$$N(\Pi) = \left(\frac{b}{2} + 1\right)^2. \tag{3}$$

The number of rules in each set $P_{m,n}$ depends on m and n and is given by

$$N(P_{m,n}) = \binom{\frac{b}{2}}{m}\binom{\frac{b}{2}}{n}. \tag{4}$$

The results for $r = 1$ are shown in Table 4. The distribution of rules in the plane is symmetric with respect to the central set $P_{2,2}$ or $P_{b/4,b/4}$ in general.

https://doi.org/10.25088/ComplexSystems.24.1.27

$N(P_{m,n})$	0	1	2	3	4
0	1	4	6	4	1
1	4	16	24	16	4
2	6	24	36	24	6
3	4	16	24	16	4
4	1	4	6	4	1

Table 4. Number of rules in each $P_{m,n}$.

The following are the sets $P_{1,4}$ and $P_{2,0}$ in the ECA.

$$P_{1,4} = \{1, 2, 16, 32\},$$
$$P_{2,0} = \{207, 221, 222, 237, 238, 252\}.$$

Similarly, we may partition the primitives (p_0 or p_1) according to the values of λ_0 and λ_1, respectively:

$$\Pi^p = \left\{P_m^p\right\} \text{ for all } m \in \left\{0, \ldots, \frac{b}{2}\right\} \text{ and } p \in \{0, 1\}. \tag{5}$$

The number of sets in the partition Π^p is

$$N\left(\Pi^p\right) = \frac{b}{2} + 1. \tag{6}$$

The number of primitives in each set P_m^p is given by

$$N\left(P_m^p\right) = \binom{\frac{b}{2}}{m}. \tag{7}$$

The following are the sets P_2^0 and P_3^1:

$$P_2^0 = \{3, 5, 6, 9, 10, 12\},$$
$$P_3^1 = \{1, 2, 4, 8\}.$$

We may express $N(\Pi)$ in equation (3) by using equation (6), as in

$$N(\Pi) = N\left(\Pi^0\right) N\left(\Pi^1\right). \tag{8}$$

Likewise, we may express $N\left(P_{m,n}\right)$ in equation (4) by using equation (7), as in

$$N\left(P_{m,n}\right) = N\left(P_m^0\right) N\left(P_n^1\right). \tag{9}$$

https://doi.org/10.25088/ComplexSystems.24.1.27

Theorem 1. There is a given number of primitives for a rule space, and they are partitioned differently depending on whether we do it on p_0 or p_1. We have that

$$P_m^p = P_{(b/2)-p}^{1-p} \text{ for } p \in \{0, 1\}. \tag{10}$$

Proof. All primitives in P_m^p are such that the number of bits different from p is equal to m:

$$\text{If } a \in P_m^p \Rightarrow \lambda_p = m \Rightarrow \lambda_{1-p} = \frac{b}{2} - m \Rightarrow a \in P_{(b/2)-m}^{1-p}$$

and vice versa. □

Theorem 2.

$$ca = \varphi(p_0, p_1, r) \text{ for } p_0 \in P_m^0 \text{ and } p_1 \in P_n^1 \Rightarrow ca \in P_{m,n}. \tag{11}$$

Proof.

$$p_0 \in P_m^0 \Rightarrow \lambda_0 = m,\ p_1 \in P_n^1 \Rightarrow \lambda_1 = n,\ \Rightarrow ca \in P_{n,m}. \ \square$$

This result leads to the following corollary.

Corollary 1. All the elements of the set $P_{m,n}$ can be generated by using φ over the Cartesian product of all elements of P_m^0 and P_n^1:

$$\varphi(p_0, p_1, r) \in P_{m,n}\ \forall\ (p_0, p_1) \in P_m^0 \times P_n^1. \tag{12}$$

Proof. Using Theorem 2, we see that any rule ca with $p_0 \in P_m^0$ and $p_1 \in P_n^1$ is in $P_{m,n}$. We need to prove the converse, so

$$ca \in P_{m,n} \Rightarrow \lambda_0 = m,\ \lambda_1 = n,\ \Rightarrow p_0 \in P_m^0,\ p_1 \in P_n^1. \ \square$$

We can partition the rule space further, but we need to introduce a cyclic permutation that preserves λ_0 and λ_1.

4. Gray Code Permutation

A permutation of the bits in each of the primitives yields rules that preserve both λ_0 and λ_1. There are a total of $4! = 24$ permutations in 4-bit primitives, but we only consider the following permutation. For an arbitrary r, we have as many as $bp! = 2^{2\,r}!$ permutations of the primitives in a rule.

https://doi.org/10.25088/ComplexSystems.24.1.27

Let us define the Gray code permutation g_n as

$$b_i \to b_j \text{ where } j = \text{ nextGrayCode}(i, n) \tag{13}$$

b_i is the bit at position i, b_j is the bit at position j, and b_0 is the least significant bit. n is the number of bits in the primitive. For a 4-bit primitive, and assuming the elements in the permutation start at position 0, the Gray code permutation g_4 is

$$g_4 = (0132).$$

Table 5 shows the application of the Gray code permutation to all primitives in the ECA, and Table 6 shows examples of the application of the Gray code permutation to both primitives of some rules, as in

$$g(ca) = \varphi(g(p_0), g(p_1), r). \tag{14}$$

p	$g(p)$	p	$g(p)$
0	0	8	2
1	4	9	6
2	1	10	3
3	5	11	7
4	8	12	10
5	12	13	14
6	9	14	11
7	13	15	15

Table 5. Application of the Gray code permutation on all primitives.

ca	$g(ca)$
9	20
26	37
30	101
74	133
110	199
141	92
197	216
204	204
239	223

Table 6. Application of the Gray code permutation to sample rules.

An alternate way to show the effect of the permutation is by showing the cycles that are formed by the multiple use of the permutation.

https://doi.org/10.25088/ComplexSystems.24.1.27

This is shown in Table 7 for the ECA. There are three types of cycles that are generated by multiple applications of this permutation; they have degrees 1, 2, and 4. We designate these cycles by the permutation used and the $\log_2$ of the degree of the cycles C_{g0}, C_{g1}, and C_{g2}.

Cycles
{0}
{1, 4, 8, 2}
{6, 9}
{3, 5, 12, 10}
{7, 13, 14, 11}
{15}

Table 7. Cycles induced by the Gray code permutation on the primitives.

Theorem 3. Taking two cycles of primitives C_{gi} and C_{gj} generated by the Gray code permutation and using the φ function over the Cartesian product of all elements of these cycles generates l number of C_{gc} cycles of rules, where $l = \min\left\{2^i, 2^j\right\}$ and $c = \max\{i, j\}$.

Proof. There are a total of $2^i\, 2^j = 2^{i+j}$ rules generated by the composition of individual primitives from cycles C_{gi} and C_{gj}. Starting at an arbitrary primitive of the longest cycle, we combine it with a primitive from the other cycle and create one rule using φ. We continue with the next primitive until we exhaust all primitives from the shortest cycle. If there are more primitives left on the longest cycle, we again take the first primitive in the shortest cycle until we exhaust all primitives in the longest cycle. All cycles of rules will be of the same length as the longest of C_{gi} and C_{gj}, hence $C_{gc} = C_{g\max\{i,j\}}$. If there are a total of 2^{i+j} rules and each cycle is of length $\max\left\{2^i, 2^j\right\}$, then there are as many as $\min\left\{2^i, 2^j\right\}$ cycles. □

5. Partitioning of the Rule Space

The dimensions of this partitioning of the ECA are defined by the primitives, and they are ordered according to the values of λ_0 and λ_1. Let us use the definitions of the sets P_i^p and create Table 8 as an overview of the partition of the rule space.

https://doi.org/10.25088/ComplexSystems.24.1.27

We can also use the Gray code permutation to further partition each P_i^p. In the case of the ECA, we have that only P_2^0 and P_2^1 are partitioned further, as shown in Table 9.

$P_{i,j}$	P_0^0	P_1^0	P_2^0	P_3^0	P_4^0
P_0^1	$P_{0,0}$	$P_{1,0}$	$P_{2,0}$	$P_{3,0}$	$P_{4,0}$
P_1^1	$P_{0,1}$	$P_{1,1}$	$P_{2,1}$	$P_{3,1}$	$P_{4,1}$
P_2^1	$P_{0,2}$	$P_{1,2}$	$P_{2,2}$	$P_{3,2}$	$P_{4,2}$
P_3^1	$P_{0,3}$	$P_{1,3}$	$P_{2,3}$	$P_{3,3}$	$P_{4,3}$
P_4^1	$P_{0,4}$	$P_{1,4}$	$P_{2,4}$	$P_{3,4}$	$P_{4,4}$

Table 8. Sets of rules in the rule space.

$C_{g\,m}$	$C_{g\,0}$	$C_{g\,2}$	$C_{g\,1}$	$C_{g\,2}$	$C_{g\,2}$	$C_{g\,0}$
$C_{g\,0}$	$C_{g\,0}$	$C_{g\,2}$	$C_{g\,1}$	$C_{g\,2}$	$C_{g\,2}$	$C_{g\,0}$
$C_{g\,2}$	$C_{g\,2}$	$4\,C_{g\,2}$	$2\,C_{g\,2}$	$4\,C_{g\,2}$	$4\,C_{g\,2}$	$C_{g\,2}$
$C_{g\,1}$	$C_{g\,1}$	$2\,C_{g\,2}$	$2\,C_{g\,1}$	$2\,C_{g\,2}$	$2\,C_{g\,2}$	$C_{g\,1}$
$C_{g\,2}$	$C_{g\,2}$	$4\,C_{g\,2}$	$2\,C_{g\,2}$	$4\,C_{g\,2}$	$4\,C_{g\,2}$	$C_{g\,2}$
$C_{g\,2}$	$C_{g\,2}$	$4\,C_{g\,2}$	$2\,C_{g\,2}$	$4\,C_{g\,2}$	$4\,C_{g\,2}$	$C_{g\,2}$
$C_{g\,0}$	$C_{g\,0}$	$C_{g\,2}$	$C_{g\,1}$	$C_{g\,2}$	$C_{g\,2}$	$C_{g\,0}$

Table 9. Cycles induced by the Gray code permutation.

We may map each rule in this two-dimensional space and use color to classify each of the rules in ECA using Wolfram classes [4]. This mapping is shown in Figure 1. It should be noted that the values of λ_0 and λ_1 are the main parameters that partition the rule space in the table.

We can observe some patterns or clusters on the distribution of classes on the rule space. Class 1 exists mostly on two of the upper rows and two of the left-hand columns, where the values for $\lambda_0 = 0$ or $\lambda_1 = 1$ and some for $\lambda_0 = 1$ and $\lambda_1 = 1$:

$$\begin{aligned} &\forall\, ca = \varphi\,(p_0, p_1, 1), \\ &\qquad p_0 = 0 \text{ or } p_1 = 15 \Rightarrow ca \text{ is class } 1, \\ &\qquad p_0 = 8,\, p_1 \neq 7 \text{ or } p_1 = 14,\, p_0 \neq 1 \Rightarrow ca \text{ is class } 1. \end{aligned}$$

p_0 ($b_5 b_4 b_1 b_0$)

p_1 ($b_7 b_6 b_3 b_2$)

r=1	λ_0	0	1				2						3				4
λ_1	$p_1 \backslash p_0$	0	4	8	2	1	9	6	3	5	12	10	14	11	7	13	15
0	15	204	220	236	206	205	237	222	207	221	252	238	254	239	223	253	255
1	13	196	212	228	198	197	229	214	199	213	244	230	246	231	215	245	247
	14	200	216	232	202	201	233	218	203	217	248	234	250	235	219	249	251
	11	140	156	172	142	141	173	158	143	157	188	174	190	175	159	189	191
	7	76	92	108	78	77	109	94	79	93	124	110	126	111	95	125	127
2	6	72	88	104	74	73	105	90	75	89	120	106	122	107	91	121	123
	9	132	148	164	134	133	165	150	135	149	180	166	182	167	151	181	183
	3	12	28	44	14	13	45	30	15	29	60	46	62	47	31	61	63
	5	68	84	100	70	69	101	86	71	85	116	102	118	103	87	117	119
	12	192	208	224	194	193	225	210	195	209	240	226	242	227	211	241	243
	10	136	152	168	138	137	169	154	139	153	184	170	186	171	155	185	187
3	8	128	144	160	130	129	161	146	131	145	176	162	178	163	147	177	179
	2	8	24	40	10	9	41	26	11	25	56	42	58	43	27	57	59
	1	4	20	36	6	5	37	22	7	21	52	38	54	39	23	53	55
	4	64	80	96	66	65	97	82	67	81	112	98	114	99	83	113	115
4	0	0	16	32	2	1	33	18	3	17	48	34	50	35	19	49	51

Wolfram Classes
1
2
3
4

Figure 1. Partition of rule space by λ_0 and λ_1 with Wolfram classes identified.

In terms of class 2, we can make the following observations:

$$\forall\ ca = \varphi(p_0, p_1, 1),$$

$$p_0 \in P_3^0 \cup P_4^0 \text{ and } p_1 \in C_{g\,2} \in P_2^1 \Rightarrow ca \text{ is class } 2,$$

$$p_1 \in P_3^1 \cup P_4^1 \text{ and } p_0 \in C_{g\,2} \in P_2^0 \Rightarrow ca \text{ is class } 2.$$

Class 3 seems to exist for P_m^0 and P_m^1 for $m > 1$, with the exception of rules 126 and 129. In particular, primitives 6 and 9 in either p_0 or p_1 are part of a large number of class 3 rules, except for rules 60, 102, 195, and 153, which are one of the $C_{g\,2}$ cycles of rules in $P_{2,2}$. In other words, we may also say that if the rule has the behavior of class 3, then either p_0 or p_1 is either 6 or 9, except for rules 60, 102, 126, 129, 195, and 153. It can be observed that for class 3, either $\lambda_0 = 2$ with $p_0 \in \{6, 9\}$ or $\lambda_1 = 2$ with $p_1 \in \{6, 9\}$ for most of the cases.

Class 4 is more elusive in terms of finding a pattern. We need more information to highlight or pinpoint the rules that are class 4.

The observations made in this section are with respect to the ECA, which have been extensively studied. This should serve as a guide to study other rule spaces, for example, for $r > 1$.

https://doi.org/10.25088/ComplexSystems.24.1.27

6. Monotone Boolean Functions and Class 4 Behavior

Every set of Boolean functions B_v has a subset of functions M_v called monotone Boolean functions [13]. A Boolean function m belongs to this subset if and only if

$$a \leq b \Rightarrow m(a) \leq m(b). \tag{15}$$

We identify the primitive equations that belong to M_v for $CA^p_{2,1}$, as shown in Table 10.

Identifying class 4 behavior has not been possible without actually evolving the cellular automata rules. In the case of the ECA rule space, we have only six rules classified as class 4. Four of these rules belong to the same equivalent class {110, 124, 137, 193} and the other two belong to another equivalent class {54, 147}. The primitives that make up these rules happen to be particularly related to the monotone Boolean functions, and this is shown in Table 11.

m_i	p	ϕ
m_0	0	0
m_1	8	$a_{-1}\, a_1$
m_2	10	a_1
m_3	12	a_{-1}
m_4	14	$a_{-1} + a_1$
m_5	15	1

Table 10. Isotone monotone Boolean functions in B_2.

Rule	p_0	p_1
110	m_2	$\overline{m}_1$
124	m_3	$\overline{m}_1$
137	$\overline{m}_4$	m_2
193	$\overline{m}_4$	m_3
54	m_4	$\overline{m}_4$
147	$\overline{m}_1$	m_1

Table 11. Class 4 rules in B_3 and their primitive functions.

Although this is not conclusive, it seems that the rules with class 4 behavior might be made up of monotone Boolean functions and their complements, at least in the ECA. Whether the monotonicity of the functions is related to the rule's behavior is an open question that

https://doi.org/10.25088/ComplexSystems.24.1.27

may be explored in a future paper. Following this idea, we looked at the rule space for $k = 2$ and $r = 2$, found the primitive equations, and identified the monotone functions. We selected a monotone function for p_0 and the complement of a monotone function for p_1 following the pattern of rule 110, as shown in Table 12. We did not select the trivially mapped rule from $r = 1$ to $r = 2$. A sample evolution of this rule is shown in Figure 2. As can be seen, it has class 4 behavior, showing some localized structures moving in the picture.

$ca\,/\,p$	Numeric	Function
ca	267452400	$\overline{a}_{-2}\, a_0 + \overline{a}_{-1}\, a_0 + \overline{a}_0\, a_{-1}$
p_0	61680	a_{-1}
p_1	4095	$\overline{a_{-2}\, a_{-1}}$

Table 12. Rule 267452400, its function, primitives, and primitive functions.

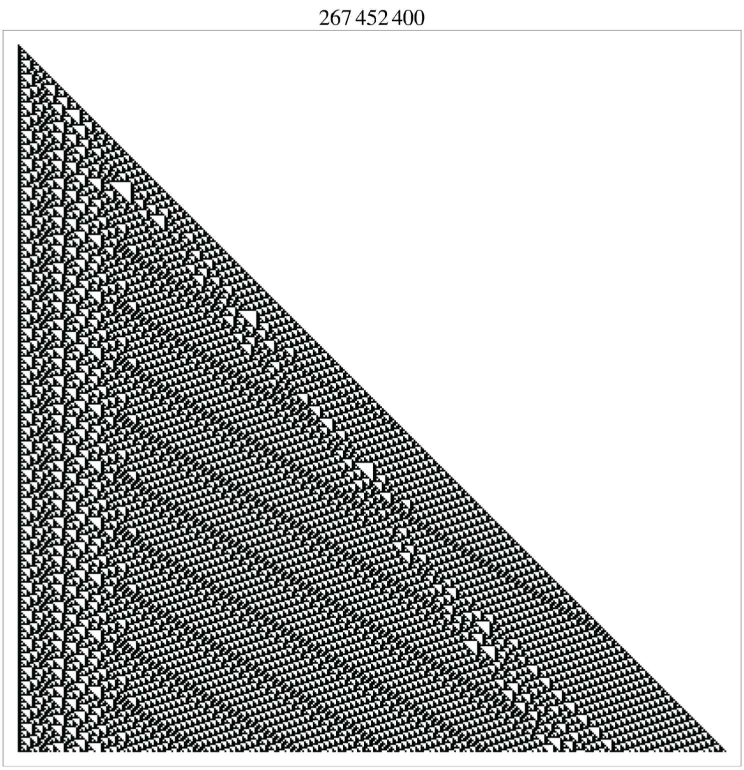

Figure 2. Evolution of rule 267452400, $k = 2$, $r = 2$.

https://doi.org/10.25088/ComplexSystems.24.1.27

The trivial mapping from $r = 1$ to a wider neighborhood creates a rule in the bigger space, but it only considers the same neighborhood as in the ECA. This implies that the rule equation for the new rule is the same as the original rule. If we map the class 4 rules shown in Table 11 to $r = 2$, we find that their primitives are also monotone Boolean functions and complements of monotone Boolean functions. Given the results showed in Section 7, we know that this does not strictly apply, but the monotonicity of the primitives may have some influence on the overall rule behavior.

7. Other Observations from M_v

We may also observe other interesting findings with respect to the elements of M_2. We denoted in Section 6 that there seems to be a certain relationship of class 4 behavior and the elements of M_2. Let us repeat a previous observation with respect to class 1 behavior:

$$p_0 = 8, p_1 \neq 7 \text{ or } p_1 = 14, p_0 \neq 1 \Rightarrow ca \text{ is class 1.}$$

We see that whenever $p_0 = 8$ and $p_1 = 7$ or $p_1 = 14$ and $p_0 = 1$, the behavior is not class 1 but class 2. The pattern is broken when $p_0 = m_1$ and $p_1 = \overline{m}_1$ or $p_0 = \overline{m}_4$ and $p_1 = m_4$. We have members of M_v and their complements acting as the anomalies in the patterns observed in Figure 1.

Similarly, we have some exceptions in class 3 shown in Table 13. The rules 60, 102, 126, 129, 195, and 153 are the only rules with the behavior of class 3 that are not made up of primitives 6 or 9. These exceptions happen to also be composed of primitives from M_2 and their complements.

Even rules such as 1 and 5, which may be classified in between class 1 and class 2, are composed of monotone functions and their complements. Portions of the evolution of rules 1 and 5 could be classified as class 1, while other portions could be classified as class 2.

Rule	p_0	p_1
60	m_3	$\overline{m}_3$
102	m_2	$\overline{m}_2$
126	m_4	$\overline{m}_1$
129	$\overline{m}_4$	m_1
153	$\overline{m}_2$	m_2
195	$\overline{m}_3$	m_3

Table 13. Class 3 rules with primitives that are not 6 or 9.

https://doi.org/10.25088/ComplexSystems.24.1.27

8. Partial Order of Primitives

Although it may seem convenient to map the entire space of rules to a two-dimensional space, we recognize that the relationship among contiguous primitives is not completely ordered. For example, primitives 8, 4, 2, and 1 have $\lambda_0 = 1$, but there is more than one bit that changes between them. They are related by the Gray code permutation and they belong to the same block of primitives, according to the value of λ_0 or λ_1. If we define that "contiguous" primitives vary by only one bit, we find that the primitives are partially ordered, and it can be shown that they can be represented as a lattice, as shown in Figure 3. We can traverse the lattice from 0 to 15 or vice versa by following a path. If we select two paths, we can create a cross section of the rule space where individual primitives in these paths will only be one bit away from each other. Figure 4 shows the evolution of 25 rules ordered in such a fashion. The value of λ_0 increases from left to right and the value of λ_1 increases from top to bottom. If we define the origin of the cross section to be the top-left corner, we find that the identity rule is at the origin. The rules for this example are made out of primitives with $p_0 \in \{0, 8, 12, 13, 15\}$ and $p_1 \in \{15, 11, 3, 1, 0\}$.

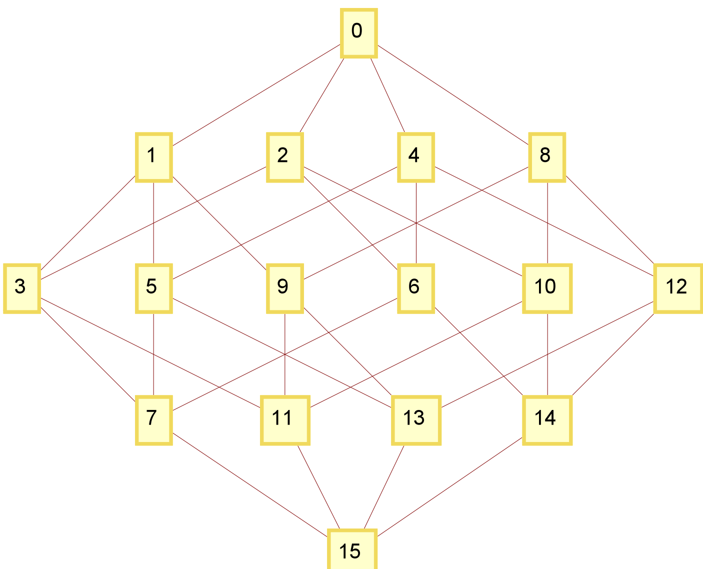

Figure 3. Lattice of the primitives in the ECA.

https://doi.org/10.25088/ComplexSystems.24.1.27

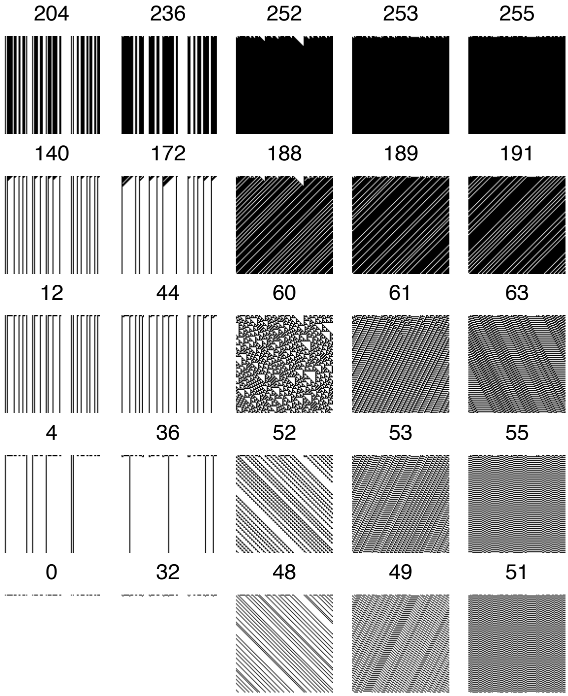

Figure 4. Random cross section of rule space for $r = 1$.

We may now move to $r = 2$, find two paths through the primitives, and create a cross section of the rule space. Figure 5 shows this for 289 rules with $p_0 \in \{0, 128, 144, 400, 33168, 33169, 41361, 57745, 57747, 61843, 61847, 62359, 62423, 63447, 63455, 63487, 65535\}$ and $p_1 \in \{65535, 32767, 24575, 24511, 24479, 23967, 23959, 22935, 18839, 18583, 16535, 16407, 16391, 7, 3, 2, 0\}$. The starting values of the cells for these examples are the same for each evolution and they are random in nature. The four corners are the equivalent rules as shown in Table 3, which are the extreme dynamic behavior rules. The width of the starting cells in Table 13 is 100 and they evolve for 100 steps. The width of the starting cells in Table 13 is 50 and they evolve for 50 steps. These parameters are selected to better show the rules' evolutions. The paths for the two sets of rule evolutions were selected at random, and different regions may be seen, depending on the specific paths selected.

We observe in both figures that there are regions of rules with similar global behavior. The regions are larger in the rule space with $r = 2$, but they are defined in both. Some samples have been run for $r = 3$, and they also show similar regions. The size of the rule space

https://doi.org/10.25088/ComplexSystems.24.1.27

with $r = 3$ is $2^{2^7} = 340282366920938463463374607431768211456$. A cross section of this space would be a $65 \times 65 = 4225$ set of rules.

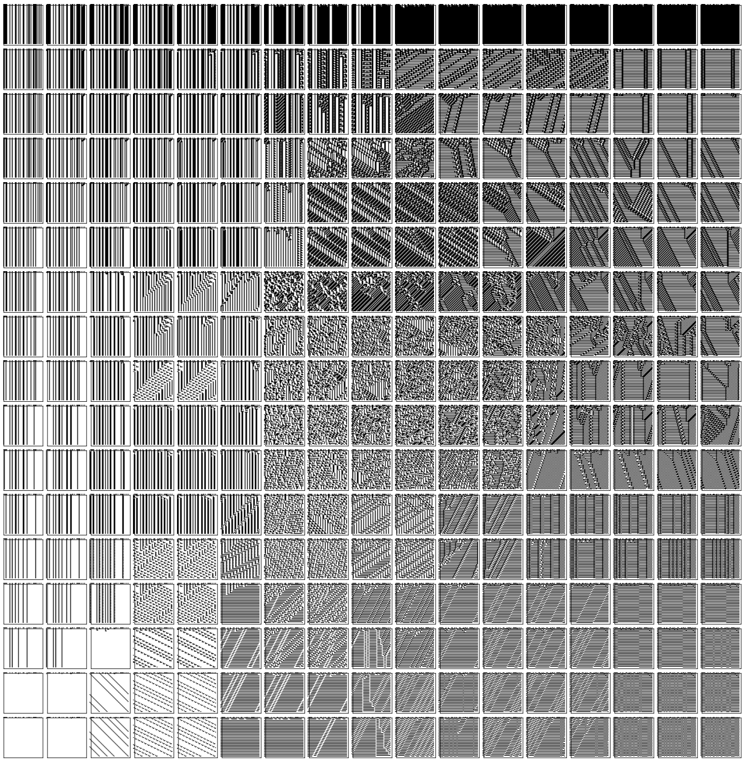

Figure 5. Random cross section of rule space for $r = 2$.

9. Conclusion

We have defined a partition of the cellular automata rule spaces by defining the primitives p_0, p_1, properties λ_0, λ_1, and the Gray code permutation. The partial order of the primitives relates the rules in the rule space in such a way that clusters of rules with similar behavior can be identified. This partition helps identify certain primitives with some characteristics that may define behavior in the clusters of rules.

This partition highlights the similar regions that exist in the rule space. Similar global behavior can be seen in neighboring rules. Neighboring rules have very similar λ_0 and λ_1, and they are positioned adja-

https://doi.org/10.25088/ComplexSystems.24.1.27

cent to each other, following the partial order of the primitives. This partitioning of the space allows for exploration of these regions of similar behavior.

The monotone Boolean functions also determine most of the anomalies found in the elementary cellular automata (ECA) rule space. We may explore the bigger rule spaces by using this guide to find the class 4 behavior as it was shown here. There is still need for further exploration and explanation of these facts, and the framework presented here is useful for finding the reasons for these particular anomalies. Perhaps the monotonicity may have something to do with these anomalies, but we cannot claim that until further investigation.

10. Future Research

The partition presented in this paper can be used to search for sets of rules with similar global behavior. Given a set of these rules, we can look into what parts of the rules remain constant and what makes that set behave in a similar way. These rules can also be used to create nonhomogeneous cellular automata. An interesting question is whether the behavior of the nonhomogeneous cellular automata will be similar to or perhaps indistinguishable from a homogeneous cellular automaton from a rule from one of these groups.

Another possible area of research is in finding a rule that almost has a given desired behavior and using this partitioning to search the space for better rules that would meet given criteria. This approach could be used in a genetic algorithm to direct changes in candidate rules and reduce the search space.

Preliminary results indicate that this partition can be extended to $k > 2$, and we have found that neighboring rules to a particular rule in $k = 3$ and $r = 1$ also have similar global behavior, and that this space can also be traversed in a way similar to that shown in this paper. The number of primitives in this particular space is equal to three, and they are designated as λ_0, λ_1, and λ_2.

Acknowledgments

I would like to thank my colleague Yesem Peker for her invaluable feedback on the notation used in this paper. Portions of the findings in this paper were presented at the New Kind of Science inaugural conference in 2003 in Boston.

https://doi.org/10.25088/ComplexSystems.24.1.27

Appendix

A. *Mathematica* Code

The following code is for the functions `CARule`, `CAPrimitives`, `Lambda`, and `Lambdas`. `CARule` builds a rule using the primitives `p0` and `p1` using radius `r`. `CAPrimitives` breaks a rule into `p0` and `p1` using radius `r`. `Lambda` returns the λ_p for a primitive, given a value for `p` and a radius `r`. `Lambdas` returns the values for λ_0 and λ_1 for a `rule` and a radius `r`.

```
CARule[p0_, p1_, r_: 1] :=
  FromDigits[Flatten[Thread[
     {Partition[IntegerDigits[p1, 2, 2^(2 r)], 2 r],
      Partition[IntegerDigits[p0, 2, 2^(2 r)], 2 r]},
     List, All]], 2];

CAPrimitives[rule_, r_: 1] := Module[{splitRule},
   splitrule = Partition[IntegerDigits[rule,
      2, 2^(2 r + 1)], 2 r],
   Map[FromDigits[#, 2] &, {Flatten[splitRule[[
       Range[2, Length[splitRule], 2]]]], Flatten[
      splitRule[[Range[1, Length[splitRule], 2]]]]}]];

Lambda[p_, lambda_, r_: 1] :=
  Count[IntegerDigits[p, 2, 2^(2 r)], 1 - lambda];

Lambdas[rule_, r_: 1] :=
  Module[{prims = CAPrimitives[rule, r]},
   {Lambda[prims[[1]], 0, r], Lambda[prims[[2]], 1, r]}];
```

References

[1] A. B. MacIsaac, D. L. Hunter, M. J. Corsten, and N. Jan, “Determinism and Thermodynamics: Ising Cellular Automata,” *Physical Review A*, **43**(6), 1991 pp. 3190–3193. doi:10.1103/PhysRevA.43.3190.

[2] A. Brass, R. K. Grencis, and K. J. Else. “A Cellular Automata Model for Helper T Cell Subset Polarization in Chronic and Acute Infection,” *Journal of Theoretical Biology*, **166**(2), 1994 pp. 189–200. doi:10.1006/jtbi.1994.1017.

[3] R. M. Itami, “Cellular Worlds: Models for Dynamic Conceptions of Landscape,” *Landscape Architecture*, July, 1988 pp. 52–57.

[4] S. Wolfram, *Cellular Automata and Complexity: Collected Papers*, Reading, MA: Addison-Wesley Publishing Company, 1994.

[5] G. J. Martinez, “A Note on Elementary Cellular Automata Classification,” *Journal of Cellular Automata*, 8(3–4), 2013 pp. 233–259.

[6] C. G. Langton, "Computation at the Edge of Chaos: Phase Transitions and Emergent Computation," *Physica D: Nonlinear Phenomena*, **42**(1–3), 1990 pp. 12–37. doi:10.1016/0167-2789(90)90064-V.

[7] W. Li and N. Packard, "The Structure of the Elementary Cellular Automata Rule Space," *Complex Systems*, **4**(3), 1990 pp. 281–297. http://www.complex-systems.com/pdf/04-3-3.pdf.

[8] G. M. B. Oliveira, P. P. B. Oliveira, and N. Omar, "Definition and Application of a Five-Parameter Characterization of One-Dimensional Cellular Automata Rule Space," *Artificial Life*, **7**(3), 2001 pp. 277–301. doi:10.1162/106454601753238645.

[9] G. Cattaneo, E. Formenti, L. Margara, and G. Mauri, "Transformations of the One-Dimensional Cellular Automata Rule Space," *Parallel Computing*, **23**(11), 1997 pp. 1593–1611. doi:10.1016/S0167-8191(97)00076-8.

[10] E. Mizraji, "The Emergence of Dynamical Complexity: An Exploration Using Elementary Cellular Automata," *Complexity*, **9**(6), 2004 pp. 33–42. doi:10.1002/cplx.20043.

[11] H. Zenil and E. Villareal-Zapata, "Asymptotic Behaviour and Ratios of Complexity in Cellular Automata," *International Journal of Bifurcation and Chaos*, **23**(9), 2013 p. 1350159. doi:10.1142/S0218127413501599.

[12] S. Wolfram, *A New Kind of Science*, Champaign, IL: Wolfram Media, Inc., 2002.

[13] I. Wegener, *The Complexity of Boolean Functions*, New York: J. Wiley, 1987.

https://doi.org/10.25088/ComplexSystems.24.1.27