Commutative Cellular Automata

Victor Duy Phan
School of Mathematics and Statistics
The University of Melbourne
Parkville, Victoria 3010, Australia

Commutative cellular automata are a class of cellular automata that portray certain characteristics of commutative behavior. We develop the notion of neighborhood partitions and neighborhood equivalence classes to analyze and enumerate these automata.

1. Introduction

Totalistic cellular automata were introduced by Wolfram [1, 2] as a means of studying higher-color cellular automata within a reasonable space of possible rules. The underlying principle is to systematically construct a subset of rules within all the possible rules of the higher-color cellular automata, and the system in totalistic cellular automata is to add up the values of the cells within the neighborhood and determine the next cell from this sum.

In this paper, we generalize this idea of taking a subset of the possible rules in cellular automata by partitioning all possible neighborhoods into neighborhood equivalence classes and use this tool to analyze cellular automata containing totalistic cellular automata—*commutative cellular automata*.

Commutative cellular automata have the property that the state of the determined cell is unaffected by the ordering of the cells in its neighborhood (some of these cellular automata meet the criteria of quasi-linear cellular automata, and so their evolutions can be computed more efficiently than the standard cellular automaton algorithm [3]).

The Game of Life is a case of *outer-commutative cellular automata*, where the state of the determined cell is unaffected by the noncenter cells.

Using the concept of neighborhood equivalence classes, it can be shown that certain commutative cellular automata—totalistic and multiplistic—can emulate each other as well as the space of all commutative cellular automata. The space of all one-dimensional commutative cellular automata will be enumerated by using `g==Sort[{##}]&` as the transition function, and certain variations of commutative cellular automata will be considered. This study is a concrete exploration of the simple commutative cellular automata. For a dynamical systems approach, see [4], where Pivato has studied a more general case from an abstract point of view.

2. Representation of Cellular Automata

One way to generate smaller spaces in cellular automata is to consider the partition of the set of all neighborhoods. Let $a_i^{(t)}$ represent the value of a cell in position i at time t, where the value a belongs to a *color set* $a_i^{(t)} \in s = \{c_1, c_2, c_3, \ldots, c_k\}$. The neighborhood of $a_i^{(t)}$ will be denoted as $\gamma\left(a_i^{(t)}\right)$; in the case of one-dimensional cellular automata of radius $r = 1$ the formula is given as

$$\gamma\left(a_i^{(t)}\right) = \left(a_{i-1}^{(t)}, a_i^{(t)}, a_{i+1}^{(t)}\right).$$

Conway's Game of Life uses a Moore neighborhood [5, 6], where the neighborhood of a cell at position i, j is given by

$$\gamma\left(a_{i,j}^{(t)}\right) = \left(a_{i-1,j+1}^{(t)}, a_{i,j+1}^{(t)}, a_{i+1,j+1}^{(t)}, a_{i-1,j}^{(t)}, a_{i,j}^{(t)}, a_{i+1,j}^{(t)}, a_{i-1,j-1}^{(t)}, a_{i,j-1}^{(t)}, a_{i+1,j-1}^{(t)}\right).$$

The set of all possible neighborhoods is

$$\Gamma = s^n = \{(a_1, a_2, a_3, \ldots, a_n) \mid a_1, a_2, a_3, \ldots, a_n \in s\}.$$

It contains every n-tuple of the set of colors s, where n is the size of the neighborhood (one-dimensional cellular automata of radius r implies $n = 2r + 1$).

A *neighborhood function* f then determines what the value of the cell in the next time step will be:

$$f : \Gamma \to s, f\left(a_i^{(t)}\right) = a_i^{(t+1)}.$$

This type of generalization will allow for defining the tools needed to enumerate *slices* of the computational space of cellular automata. The motivation for doing this is to explore a space of cellular automata that is large enough to handle, yet still produces interesting behavior. The main slice of cellular automata presented in this paper will be of type *one-dimensional commutative cellular automata*.

3. Neighborhood Partitions and Equivalence Classes

The set of all possible neighborhoods Γ is partitioned into *neighborhood equivalence classes* such that two neighborhoods $\gamma_1, \gamma_2 \in \Gamma$ are in the same equivalence class if and only if

$$f(\gamma_1) = f(\gamma_2). \tag{1}$$

We call this partition the *neighborhood partition* of the cellular automaton.

The function f for this totalistic cellular automaton is

$$f : s^3 \to s,\ f\left(\gamma\left(a_i^{(t)}\right)\right) = h\left(g\left(\gamma\left(a_i^{(t)}\right)\right)\right),$$

where g is a summation function and h maps this sum to the value of the determined cell:

$$g : \Gamma \to G,\ g\left(a, b, c\right) = a + b + c,$$
$$h : G \to s.$$

For convenience, using the Wolfram Language, we call this the `g==Plus` cellular automaton. And for all of the cellular automata in this paper we will adopt the convention that `g` stands for the inner function of the composition in the neighborhood function f, as defined. The *rule function* h determines the particular rule the cellular automaton is using. All possible mappings $G \to s$ is the space of all possible rules for the cellular automata.

The partition for totalistic cellular automata is shown in Figure 1.

The graphics convention shown in Figure 2 will be used to represent a neighborhood partition.

$$\left\{ \begin{matrix} g(\gamma)=0 \\ (0,0,0) \end{matrix},\ \begin{matrix} g(\gamma)=1 \\ (0,0,1) \\ (0,1,0) \\ (1,0,0) \end{matrix},\ \begin{matrix} g(\gamma)=2 \\ (0,0,2) \\ (0,1,1) \\ (0,2,0) \\ (1,0,1) \\ (1,1,0) \\ (2,0,0) \end{matrix},\ \begin{matrix} g(\gamma)=3 \\ (0,1,2) \\ (0,2,1) \\ (1,0,2) \\ (1,1,1) \\ (1,2,0) \\ (2,0,1) \\ (2,1,0) \end{matrix},\ \begin{matrix} g(\gamma)=4 \\ (0,2,2) \\ (1,1,2) \\ (1,2,1) \\ (2,0,2) \\ (2,1,1) \\ (2,2,0) \end{matrix},\ \begin{matrix} g(\gamma)=5 \\ (1,2,2) \\ (2,1,2) \\ (2,2,1) \end{matrix},\ \begin{matrix} g(\gamma)=6 \\ (2,2,2) \end{matrix} \right\}$$

Figure 1. The neighborhood partition of one-dimensional radius $r = 1$ totalistic cellular automata over the color set $s = \{0, 1, 2\}$. The notation $g(\gamma)$ has been added to represent the corresponding sums for each neighborhood equivalence class.

For the partition shown in Figure 1, there are $3^7 = 2187$ rules, as there are $k = 3$ colors and seven equivalence classes. The number of rules for a k-color cellular automaton with n neighborhood equivalence classes in its partition is k^n. If $n < k$, then the cellular automaton effectively becomes an n-color cellular automaton, as there are only n maximum mappings in the neighborhood function $f : \Gamma \to s$ to the color set s. To remove this *degenerate* case, it is required that

$n \geq k$, and consequently the set of all possible nondegenerate rule space sizes is $\{k^n \mid k, n \in \mathbb{N}, n \geq k\}$. This sequence will allow the flexibility of adjusting the size of the computational space that is desired to be explored in cellular automata. A plot of the first 100 terms of this sequence is shown in Figure 3.

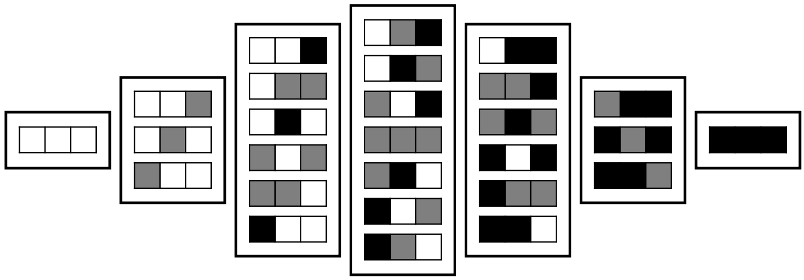

Figure 2. A visual representation of the neighborhood partition of one-dimensional, radius $r = 1$ totalistic cellular automata over the color set $s = \{0, 1, 2\}$.

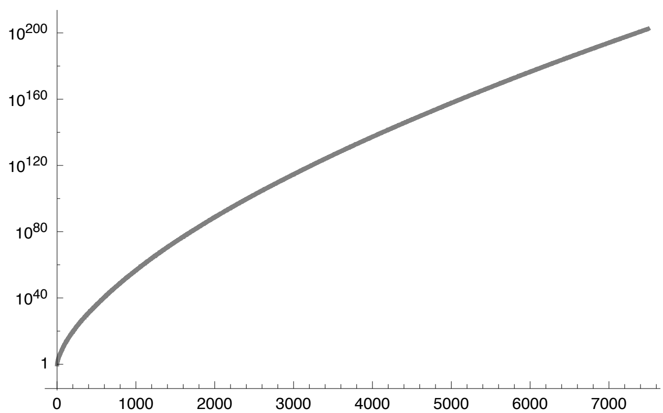

Figure 3. The logarithmic plot for more than 7000 values of the sequence $\{k^n \mid k, n \in \mathbb{N}, n \geq k\}$, which represents the set of all possible nondegenerate rule space sizes for cellular automata.

The size of the number of rules for each cellular automaton is distributed across orders of magnitudes, making it convenient to select partitions of cellular automata of certain size to be enumerated on computers.

The most general partition is the partition where all neighborhoods belong to their own equivalence class, and indeed this is the standard cellular automaton, which has been extensively studied by Wolfram.

4. Multiplistic Cellular Automata

A multiplistic cellular automaton is a cellular automaton that uses `g==Times` in its neighborhood function. The partition for multiplistic cellular automata using the color set $\{0, 1, 2\}$ is shown in Figure 4.

The large equivalence class is due to the property that all neighborhoods containing at least one cell of value 0 will always output 0. Hence all neighborhoods containing 0 belong to this equivalence class. To remove this bias toward a particular cell color in multiplistic cellular automata, a color set without 0 can be chosen. The partition shown in Figure 5 is multiplistic, using the color set $\{1, 2, 3\}$.

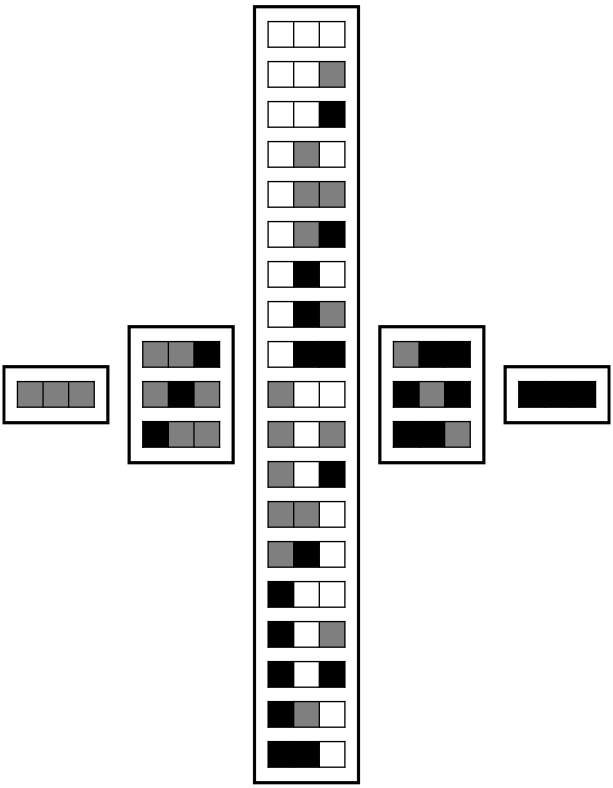

Figure 4. The neighborhood partition for radius $r = 1$ multiplistic cellular automata over the color set $s = \{0, 1, 2\}$. The large neighborhood equivalence class in the center is all the neighborhoods containing 0, as they will all produce the same product of 0.

This partition is a superset of the totalistic cellular automata neighborhood partition (Figure 2), as the rules can be selected such that two equivalence classes will output the same cell color. This poses the question, are multiplistic cellular automata more general than totalistic (excluding the color 0)? It turns out that totalistic and multiplistic cellular automata are able to produce nearly all the same partitions.

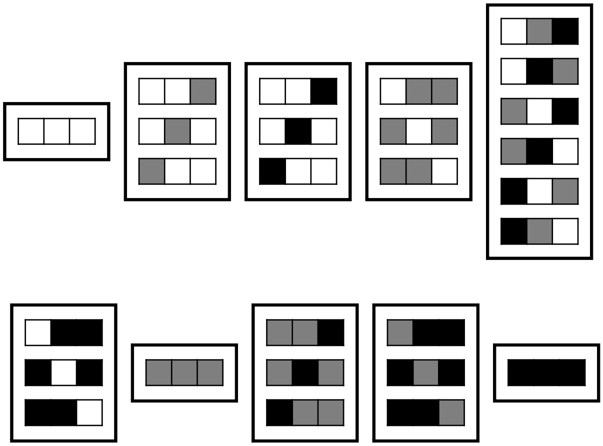

Figure 5. The neighborhood partition for radius $r = 1$ multiplistic cellular automata over the color set $s = \{1, 2, 3\}$. The neighborhood equivalence classes can be joined in such a way as to reproduce the neighborhood partition of the totalistic cellular automata in Figure 2.

Proof. In one-dimensional, radius $r = 1$ totalistic cellular automata and multiplistic cellular automata, for every neighborhood $\gamma = (c_1, c_2, c_3)$ under totalistic cellular automata the sum will be $g_{\text{Plus}}(\gamma) = c_1 + c_2 + c_3$. Then there exists a multiplistic cellular automaton with the appropriate color set such that the neighborhood is $\gamma' = (a^{c_1}, a^{c_2}, a^{c_3})$ for $a \neq 0$. Consequently, the product will be

$$g_{\text{Times}}(\gamma') = a^{c_1} \times a^{c_2} \times a^{c_3} = a^{c_1+c_2+c_3} = a^{g_{\text{Plus}}(\gamma)}.$$

Hence if two neighborhoods γ_1, γ_2 are in the same neighborhood equivalence class in the totalistic cellular automaton, then

$$\begin{aligned}
&g_{\text{Plus}}(\gamma_2) = g_{\text{Plus}}(\gamma_2)\\
&a^{g_{\text{Plus}}(\gamma_1)} = a^{g_{\text{Plus}}(\gamma_2)} \text{ for } a \neq 0\\
&\exists\, \gamma_1', \gamma_2', g_{\text{Times}}(\gamma_1') = a^{g_{\text{Plus}}(\gamma_1)}, g_{\text{Times}}(\gamma_2') = a^{g_{\text{Plus}}(\gamma_2)}\\
&\therefore g_{\text{Times}}(\gamma_1') = g_{\text{Times}}(\gamma_2') \text{ for any } g_{\text{Plus}}(\gamma_2) = g_{\text{Plus}}(\gamma_2).
\end{aligned}$$

Therefore for any partition in totalistic cellular automata, there will be an equivalent partition in multiplistic cellular automata.

The converse is partially true: for every multiplistic cellular automaton, there will be an equivalent partition in totalistic cellular automata if 0 does not belong to the color set s of the multiplistic cellular automata $0 \notin s$.

For every neighborhood $\gamma = (c_1, c_2, c_3)$, under multiplistic cellular automata the product will be $g_{\text{Times}}(\gamma) = c_1 \times c_2 \times c_3$. Then there exists a totalistic cellular automaton with the appropriate color set such that the neighborhood is $\gamma' = (\log(c_1), \log(c_2), \log(c_3))$. Consequently, the sum will be

$$g_{\text{Plus}}(\gamma') = \\ \log(c_1) + \log(c_2) + \log(c_3) = \log(c_1 \times c_2 \times c_3) = \log(g_{\text{Times}}(\gamma)).$$

Hence if two neighborhoods γ_1, γ_2 are in the same neighborhood equivalence class in multiplistic automata, then

$$g_{\text{Times}}(\gamma_1) = g_{\text{Times}}(\gamma_2)$$
$$\log(g_{\text{Times}}(\gamma_1)) = \log(g_{\text{Times}}(\gamma_2))$$
$$\exists\ \gamma_1', \gamma_2', g_{\text{Plus}}(\gamma_1') = \log(g_{\text{Times}}(\gamma_1)), g_{\text{Plus}}(\gamma_2') = \log(g_{\text{Times}}(\gamma_2))$$
$$\therefore g_{\text{Plus}}(\gamma_1') = g_{\text{Plus}}(\gamma_2') \text{ for any } g_{\text{Times}}(\gamma_2) = g_{\text{Times}}(\gamma_2). \square$$

When 0 belongs to the color set of multiplistic cellular automata, the resulting partition would look somewhat similar to Figure 4, with a large equivalence class containing all the neighborhoods with at least one 0. This cannot be done in totalistic cellular automata, where $\log(0)$ is undefined, and there is no such number x such that for any a, $x + a = x$. Despite this however, the rule chosen in the totalistic cellular automata can emulate all the rules in a multiplistic cellular automaton containing 0. In the diagram shown in Figure 6, the first six

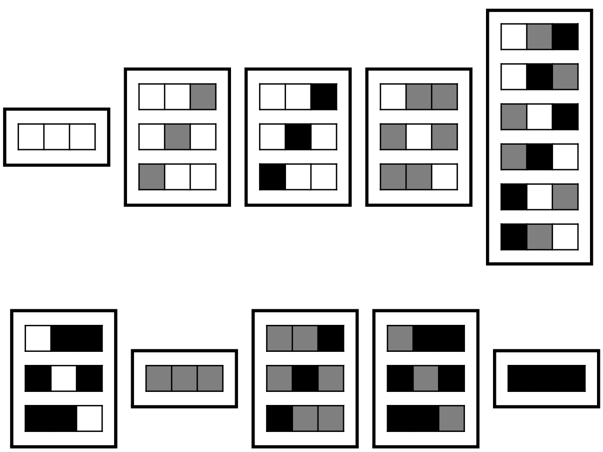

Figure 6. The neighborhood partition for radius $r = 1$ totalistic cellular automata over the color set $s = \{0, 1, 4\}$. Equivalent partition to multiplistic cellular automata over color set $s = \{1, 2, 3\}$ (Figure 5), and therefore will be able to produce the exact same cellular automata. Other possible color sets that produce this same partition for totalistic cellular automata are {log(1), log(2), log(3)}, {1, 4, 13}.

equivalence classes in the totalistic cellular automata neighborhood partition can always be assigned to the same value in the neighborhood function, effectively allowing the totalistic cellular automata to emulate all the possible rules of the multiplistic cellular automata containing 0 in Figure 4.

5. Commutative Cellular Automata

Totalistic and multiplistic cellular automata are in a more general class of cellular automata—*commutative cellular automata*. Commutative cellular automata are cellular automata where the neighborhood function f is commutative over its arguments; that is, if two tuples γ_1, γ_2 are permutations of each other, then $f(\gamma_1) = f(\gamma_2)$. Then by equation (1), γ_1 and γ_2 come under the same neighborhood equivalence class. This function f is referred to as a *commutative function*.

To enumerate the space of commutative cellular automata, a function that satisfies the given condition will be needed. The function could be meticulously defined for each and every possible neighborhood, but an easier alternative could be to use `g==Sort[{##}]&`. This satisfies the condition for commutative cellular automata, as all permutations of a neighborhood are the same when sorted as shown in Figure 7.

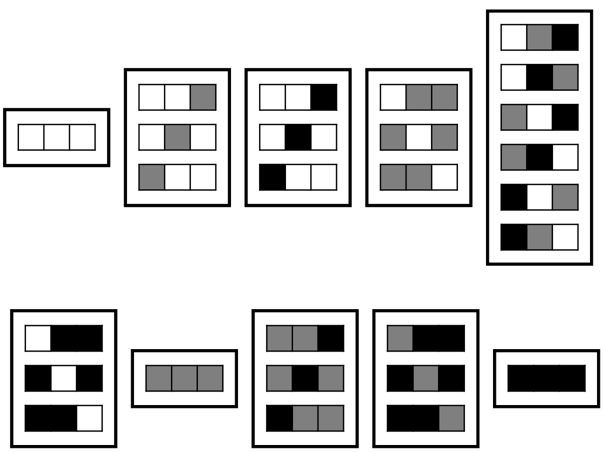

Figure 7. The neighborhood partition for radius $r = 1$ commutative cellular automata over the color set $s = \{0, 1, 2\}$. This partition is the same as the partitions seen in multiplistic (Figure 5) and totalistic (Figure 6) cellular automata.

The partition shown in Figure 7 is the same as the partitions of multiplistic (Figure 5) and totalistic (Figure 6) cellular automata as shown before: this is due to the commutative nature of addition and multiplication. The following commutative cellular automata in this section were produced using the function `g==Sort[{##}]&`.

The total number of rules in these commutative cellular automata is $3^{10} = 59\,049$. Enumerating a random sample of interesting rules in this space of cellular automata yielded the rules shown in Figure 8 (the automata that also belong to totalistic cellular automata have been omitted, as they have been extensively studied).

All the cellular automata are symmetrical in the middle axis, as for every neighborhood (a, b, c), its mirror (c, b, a) belongs in the same equivalence class, as they are permutations of each other.

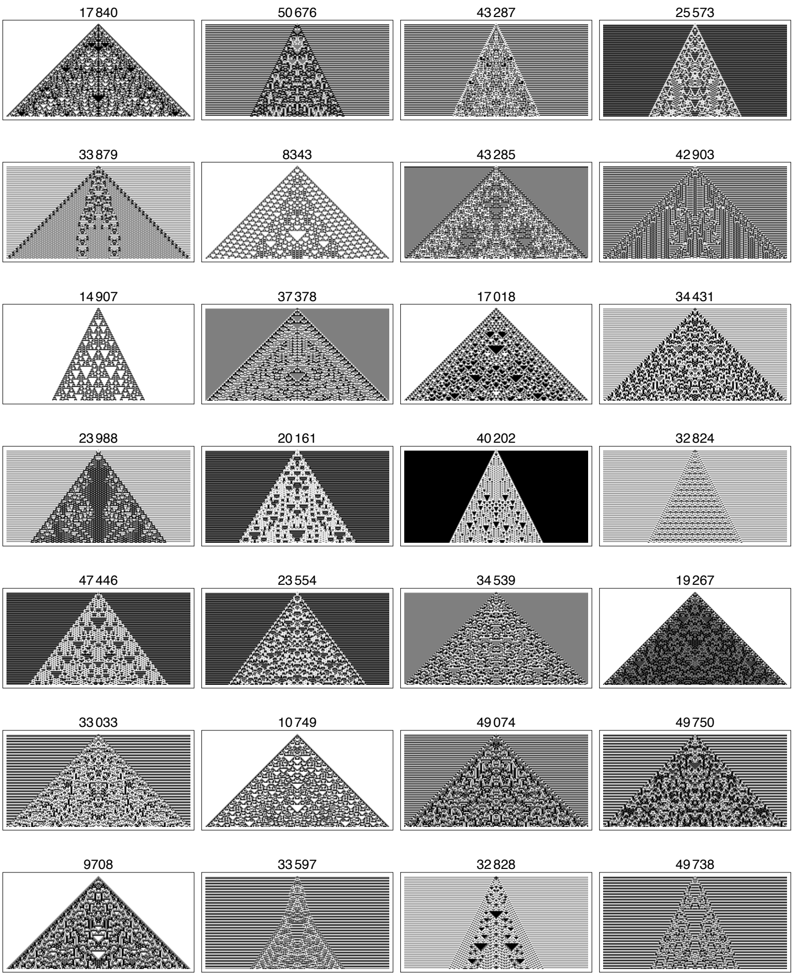

Figure 8. (*continues*).

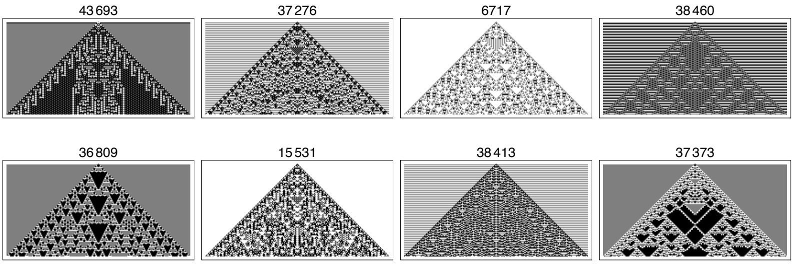

Figure 8. Interesting 3-color, radius $r = 1$ commutative cellular automata. All the automata are symmetrical in the middle due to the commutative nature of the automata.

Commutative cellular automata could also have only two colors, which produces the partition shown in Figure 9.

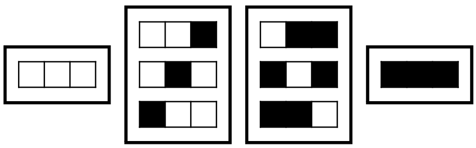

Figure 9. Partition of radius $r = 1$, 2-color commutative cellular automata. As there are only four equivalence classes in total, the total number of possible rules is $2^4 = 16$.

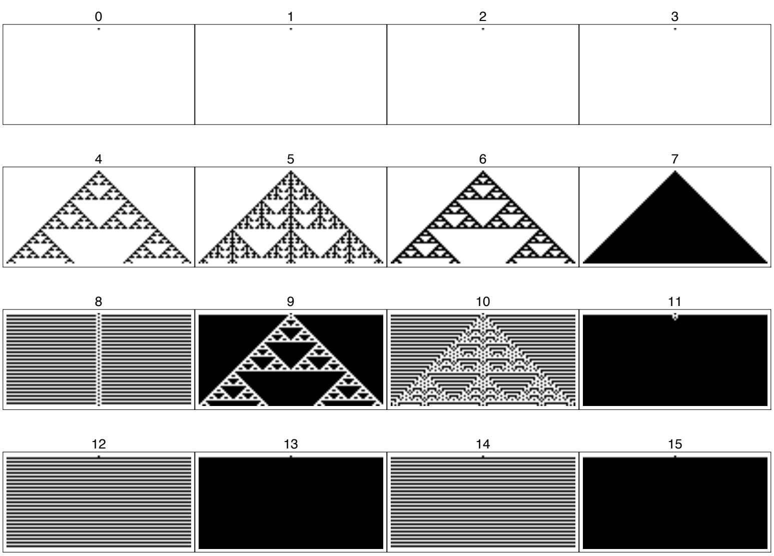

Figure 10. All the rules for 2-color, radius $r = 1$ commutative cellular automata with initial condition of one black cell against a background of white cells. These rules only produce simple and nested behavior.

The rules shown in Figure 10 have either simple or nested behavior and are not very interesting, but a property of nested behavior is the preservation of randomness, as shown in Figure 11.

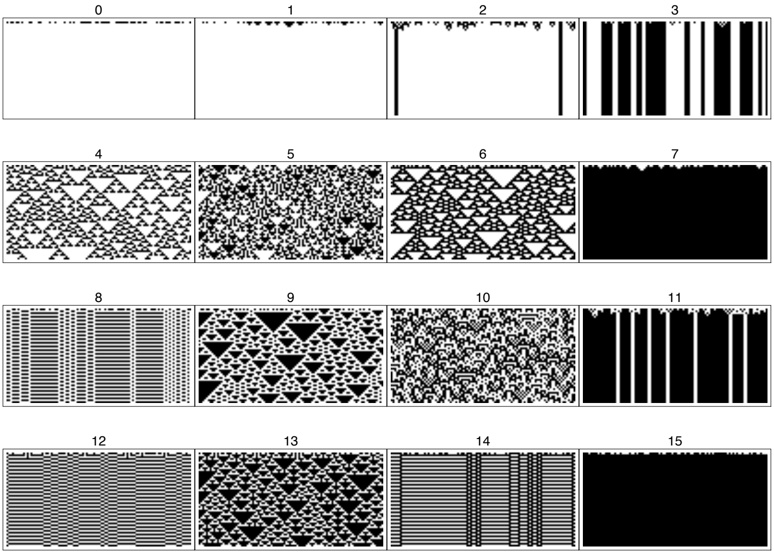

Figure 11. All the rules for 2-color, radius $r = 1$ commutative cellular automata with random initial conditions.

One might consider extending these automata to radius $r = 2$. This produces the partition shown in Figure 12.

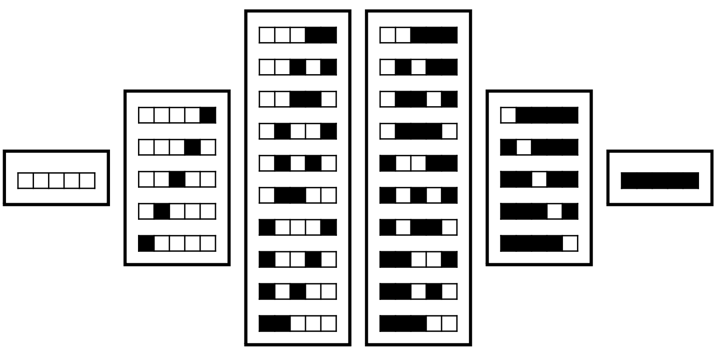

Figure 12. Partition of 2-color, radius $r = 2$ commutative cellular automata. There are now six equivalence classes, hence the number of possible rules is $2^6 = 64$.

The number of possible rules is now $2^6 = 64$ (Figure 13), still less than the number of possible elementary cellular automata at $2^8 = 256$.

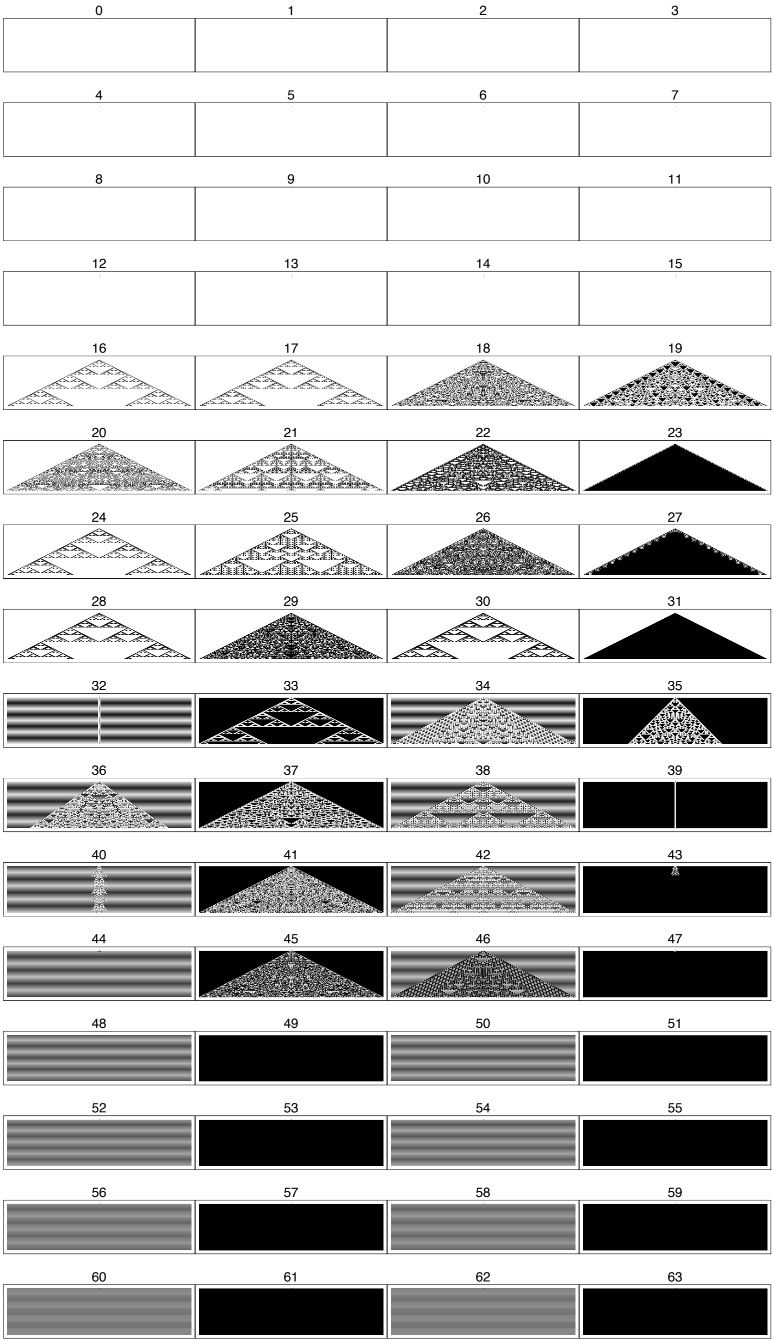

Figure 13. All the rules for 2-color, radius $r = 2$ commutative cellular automata; unlike radius $r = 1$, some of these rules display highly random behavior.

Increasing the radius of a cellular automaton has turned a commutative cellular automaton with no complex behavior into one with interesting rules, ranging from ones with localized structures to ones that seem completely random.

6. Outer-Commutative Cellular Automata

An *outer-commutative cellular automaton* is a cellular automaton where the neighborhood function f is commutative over all cells excluding the center cell (Figure 14). That is, for $n = 2r + 1$, $r \in \mathbb{N}$, the outer-commutative function g is

$$g\left(c_1, c_2, \ldots, c_{(n+1)/2}, \ldots, c_{n-1}, c_n\right) = \left(g'\left(c_1, c_2, \ldots, c_{((n+1)/2)-1}, c_{((n+1)/2)+1}, \ldots, c_{n-1}, c_n\right), c_{(n+1)/2}\right),$$

where g' is a commutative function.

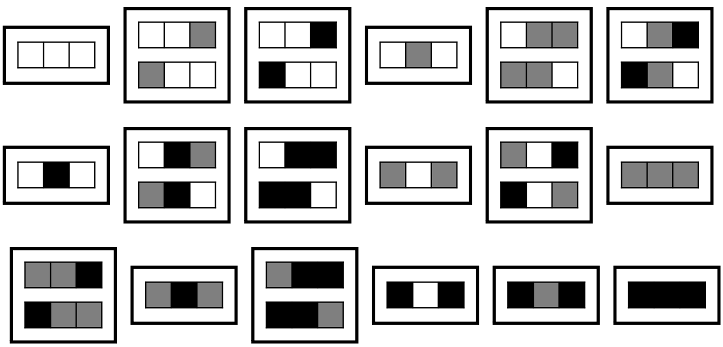

Figure 14. The neighborhood partition for $r = 1$, $k = 3$ outer-commutative cellular automata. The number of possible rules is $3^{18} = 387\,420\,489$.

A well-known two-dimensional rule of outer-commutative cellular automata is Conway's Game of Life. It uses the Moore neighborhood

$$\gamma\left(a_{i,j}^{(t)}\right) = \left(a_{i-1,j+1}^{(t)}, a_{i,j+1}^{(t)}, a_{i+1,j+1}^{(t)}, a_{i-1,j}^{(t)}, a_{i,j}^{(t)}, a_{i+1,j}^{(t)}, a_{i-1,j-1}^{(t)}, a_{i,j-1}^{(t)}, a_{i+1,j-1}^{(t)}\right),$$

where the center cell is $a_{i,j}^{(t)}$.

The partition of outer-commutative cellular automata is also a superset of the well-known outer-totalistic cellular automata (Figure 15).

Apart from the larger selection of possible rules, outer-commutative cellular automata display the same characteristics as commutative cellular automata—a reflective symmetry about the center axis. Some manifest localized structures, some appear apparently random, and some seem to have an "outer shell" of cells always growing and surrounding the inner part of the automata (Figure 16).

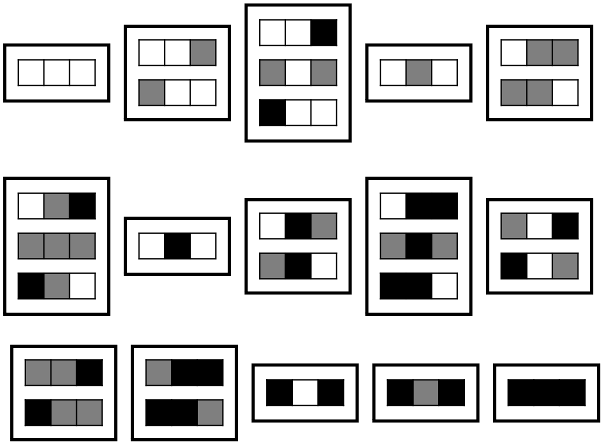

Figure 15. Neighborhood partition for $r = 1$, $k = 3$ outer-totalistic cellular automata. Number of rules is $3^{15} = 14\,348\,907$.

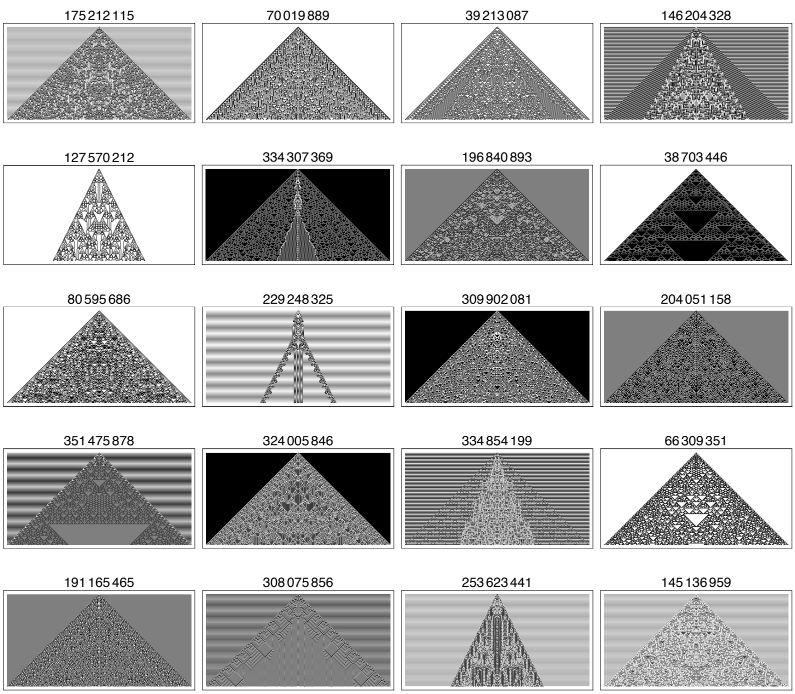

Figure 16. (*continues*).

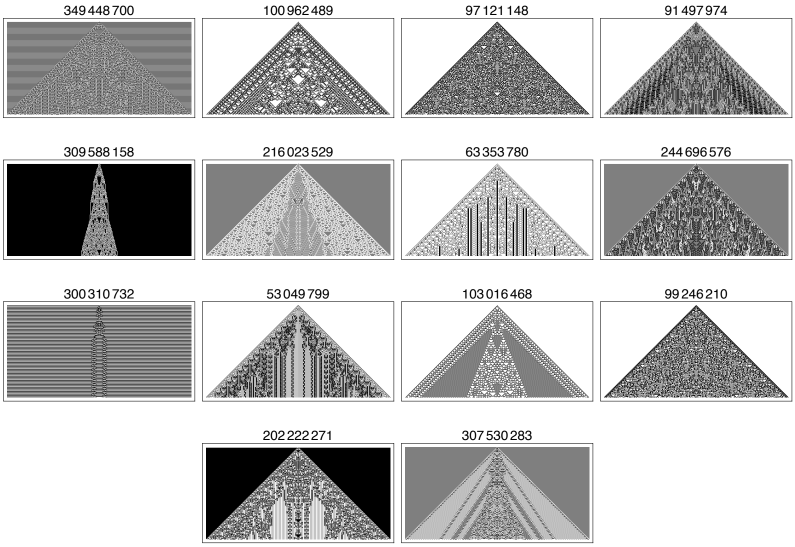

Figure 16. Selected 3-color, radius $r = 1$ outer-commutative cellular automata that are noncommutative.

7. Further Directions

One possible development based on this paper would be to invent a method for categorizing and enumerating cellular automata partitions. The class of commutative cellular automata could also be studied with more advanced New Kind of Science tools and is well within the means of current computation capabilities.

Appendix

A. Wolfram Language Code

The code used for enumerating the cellular automata space and constructing the neighborhood partition diagrams is as follows (note that `f` is a global neighborhood function that takes in a list as the neighborhood argument and outputs an arbitrary result that goes into the determination of the neighborhood equivalence classes).

```
rule[n_, space_, radius_] :=
 Module[{r = 2 radius + 1},
  Map[Apply[Rule, #] &, Transpose[{Union[f /@ Tuples[space, r]],
     IntegerDigits[n, Length[space], Length[
        Union[f /@ Tuples[space, r]]]] /. Map[Apply[Rule, #] &,
       Transpose[{Range[Length[space]] - 1, space}]]}]]]
```

```
CCAStep[ruleform_, list_, space_, radius_] :=
 Module[{r = 2 radius + 1},
  Map[f, Partition[list, r, 1, {(r + 1)/2, (r + 1)/2}, First[list]]] /.
   ruleform]

CCAEvolve[nrule_, init_, space_, r_, t_] :=
 Module[{ruleform = rule[nrule, space, r]},
  NestList[CCAStep[ruleform, #, space, r] &, init, t]]

NeighborhoodPartition[f_, space_, size_, r_] :=
 Row[
  Framed /@ Map[Function[x, Column[ArrayPlot[{#}, Mesh → True,
         MeshStyle → Black, ImageSize → size] & /@ x]],
    GatherBy[Tuples[space, 2 r + 1], f]], "  "]
```

Acknowledgments

I would like to thank Stephen Wolfram for his encouragement to pursue this project, Matthew Szudzik and Todd Rowland for mentoring and supervising me on the development of this project, and Simon Tyler for advice.

References

[1] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Reviews of Modern Physics*, **55**(3), 1983 pp. 601–644. doi:10.1103/RevModPhys.55.601.

[2] S. Wolfram, *A New Kind of Science*, Champaign, IL: Wolfram Media, Inc., 2002 pp. 60–70 and 886.

[3] C. Moore, "Quasi-Linear Cellular Automata," *Santa Fe Institute Working Papers*, 95-09-078, 1995. http://www.santafe.edu/media/workingpapers/95-09-078.pdf.

[4] M. Pivato, "Multiplicative Cellular Automata on Nilpotent Groups: Structure, Entropy, and Asymptotics," *Journal of Statistical Physics*, **110**(1), 2003 pp. 247–267. doi:10.1023/A:1021074813291.

[5] E. W. Weisstein. "Game of Life" from Wolfram MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/GameofLife.html.

[6] E. W. Weisstein. "Moore Neighborhood" from Wolfram MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/MooreNeighborhood.html.