# Definition and Identification of Information Storage and Processing Capabilities as Possible Markers for Turing Universality in Cellular Automata

**Yanbo Zhang**
*Physical Department*
*University of Science and Technology of China*
*Hefei, Anhui, P. R. China*

To identify potential universal cellular automata (CAs), a method is developed to measure the information processing capacity of elementary cellular automata (ECAs). Two features of CAs are considered: ability to store information and ability to process information. Local collections of cells are defined as particles of CAs and the information contained by particles is examined. By using this method, information channels and intersections of channels can be shown. By observing these two features, potential universal CAs are classified into a certain class, and all ECAs can be classified into four groups, which correspond to Wolfram's four classes: 1, homogeneous; 2, regular; 3, chaotic and 4, complex. This result shows that using the abilities of storing and processing information to characterize complex systems is effective and succinct. It is found that these abilities are capable of quantifying the complexity of systems.

*Keywords*: cellular automata; Turing universality; cellular automata classification

## 1. Introduction

A universal system is a system that can execute any computer program. In other words, it is feasible for it to execute any algorithm [1]. It was found that some systems with simple rules can be a universal system, such as rule 110 in elementary cellular automata (ECAs) [1– 3]. Some tag systems and cyclic tag systems were proved to be universal, which are also systems with simple rules [2–4]. A glider system, which is an idealized system to simulate particle processes of real physical systems, was also proved to be a universal system [2]. And particle machines (PMs) in periodic backgrounds were proved to be universal [5].

The widespread existence of universal systems implies that some processes with simple rules in the real world may be able to execute some algorithms or any algorithm. Because of the significant number

https://doi.org/10.25088/ComplexSystems.27.1.45

of algorithms, the behaviors of these systems can be variable and complex, which was considered as a potential origin of complexity in [3, 6].

Cellular automata (CAs) can show a wide variety of complex phenomena in the real world, and CAs are also sufficiently general for a wide variety of physical, chemical, biological and other systems [7]. Identifying universal CAs will help people understand the origins of cellular automaton (CA) behaviors and find key dynamics of computation.

In this paper, a method is developed to identify potential universal ECAs. Two abilities of a system are considered: (1) ability to store information; and (2) ability to process information. We found these two features can identify potential universal CAs and quantify the complexity of systems.

### 1.1 Elementary Cellular Automata

Cellular automata are ideal models for physical systems in which space and time are discrete, and ECAs are the simplest kind of CAs.

Elementary cellular automata are dynamic systems defined by deterministic rules, working on a one-dimensional list $\{c_n\}$ with $n$ cells. Rules can be expressed by the function $F$:

$$c_n(t+1) = F[c_{n-1}(t), c_n(t), c_{n+1}(t)], \tag{1}$$

where $n \in \mathbb{Z}$.

Therefore, $c_n(t+1)$ is the function of itself, $c_n(t)$, and its two immediate neighbors: $c_{n-1}(t)$ and $c_{n+1}(t)$. Each $c_n(t)$ has two possible states, 0 or 1. So there should be a list $R$ of length $2^3 = 8$ to define a rule, and there will be $2^8 = 256$ different rules. If $R$ is equal to $\{0, 0, 0, 1, 1, 1, 1, 0\}$ and it is considered as a binary code, it equals 30 in decimal base and is called ECA rule 30.

With a given initial list $L_0$, an ECA will apply the function $F$ to all cells in parallel to update $L_t$ to $L_{t+1}$. That is,

$$L_t \xrightarrow{F} L_{t+1}. \tag{2}$$

By doing this process repeatedly, a matrix $M^{(\text{rule})} = (L_0, L_1, \ldots, L_t)$ will be generated, which is the "spacetime evolution." Figure 1 shows two spacetime evolutions generated by ECA rule 30 and ECA rule 110, started with the same $L_0$.

Two hundred fifty-six different ECAs can be classified. In this paper, we compare our work with Wolfram's classification, which is classes 1 through 4 in [3, 7]. The classes are: 1, homogeneous; 2, regu-

lar; 3, chaotic; 4, complex. Some typical spacetime evolutions are shown in Figure 1. There are also some other classifications; see [8–10].

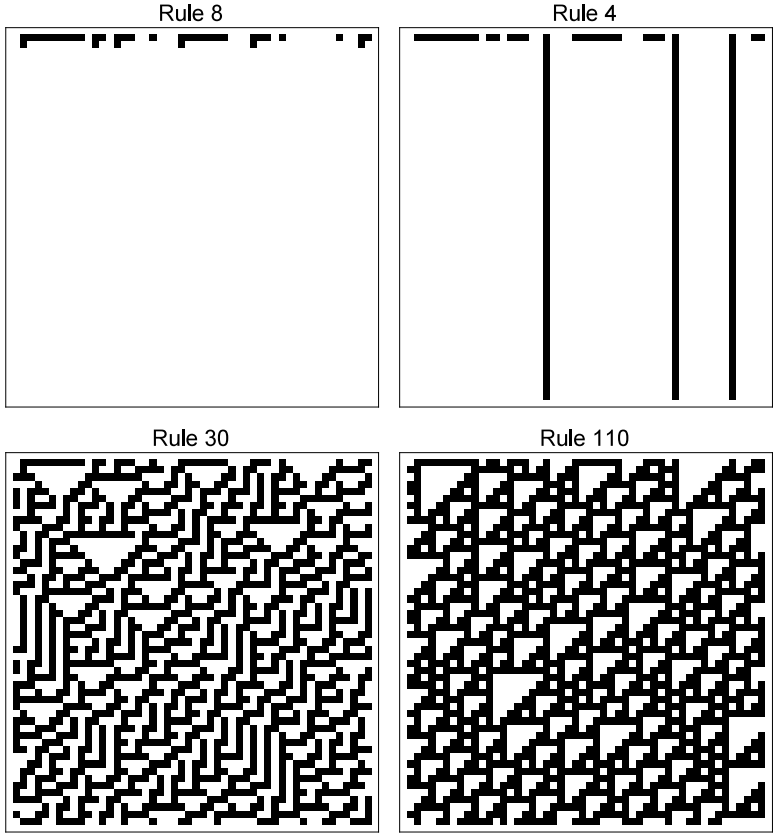


**Figure 1.** Evolution of four typical rules from classes 1 through 4. Rule 8 is in class 1, rule 4 is in class 2, rule 30 is in class 3, and rule 110 is in class 4.

## 2. Methodology

We consider two abilities of ECA rules: ability to store information and ability to process information. The ability to store information will make the system stable enough and not too noisy. Only when information can be stored can information move stably in a system, so that the whole system can be related. The ability to process information means interactions between different information should be found in a system.

We define a system that can store information when its current local states can be used to infer previous states at some location. It is

https://doi.org/10.25088/ComplexSystems.27.1.45

true that some reversible systems can store all information of the whole system, but this information can hardly be used to infer the previous states, because many of them are computationally irreducible. Thus, the particle systems can cover the definition.

We identify potential universal ECAs based on a theorem proposed in [5], which considers particle-like structures and their behavior in systems to identify Turing machines (TMs) and universal Turing machines (UTMs).

A method was developed to extract particle patterns from ECAs to build "particle machines" and to measure their computational ability by taking into account their features. First, it is necessary to introduce PMs and define particles in ECAs.

### 2.1 Particle Machines

A PM is a system in which particles can move, collide, annihilate and generate in a homogeneous medium. Figure 2 shows a typical PM. Data and configurations are injected from the left in the form of particles, and by executing this system, particles will have interactions. Lines and dotted lines in Figure 2 represent the paths of particles. After time $t$, the system will generate an output. The identity of a particle includes position, phase and velocity. During collisions, particles can alter their identities or be generated or annihilated. These changes of particles can be considered as a function of particles that participate in the collision, which is the collision function. Some PMs are proved to be TMs or UTMs in [5]. A PM is at least a TM when: (1) identity of particles can change during collisions; and (2) collision function depends on identities of particles. For the first requirement, the identity of particles can change during collisions; this also means new particles can be generated during collisions. The second requirement means the result of a collision should depend on the types of particles that participate in the collision. If no particles can be generated or annihilated in collisions in a PM, then the PM is not a UTM.

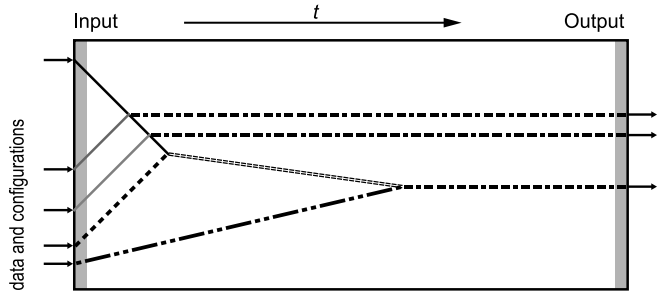


**Figure 2.** A typical PM.

## 2.2 Particles in Elementary Cellular Automata

We define a local grid of cells in $M^{(\text{rule})}$ as a particle in ECAs. Here we consider one kind of particle: their sequence may change periodically or not change through time. We call them "elementary particles." It will be practical if we start with this simple kind of particle.

Particles contain information, so that information can move in space and have interaction with other information, which is a kind of computation [11]. All identities of particles: location, velocity and sequence, can be computed by collisions. And all of these identities can be preserved if there are no collisions.

To extract the identity of particles from an ECA spacetime evolution, a certain sequence should be chosen for the research. We need to choose a sequence as a particle to study, which is the "target particle." As Figure 3(a) shows, we choose target particle $\mathfrak{P}$ at the bottom center of a spacetime evolution and mark the same sequences as "linear particles" $\mathfrak{L}$s, which are the dots in Figure 3(b).

$\mathfrak{P}$ can be explained by the equation $\mathfrak{P} = L_{t_{\max}}(p_L, p_R)$, where $L_t$ is the $t^{\text{th}}$ row of the spacetime evolution. $p_L$ and $p_R$ are the start index and the end index for $\mathfrak{P}$.

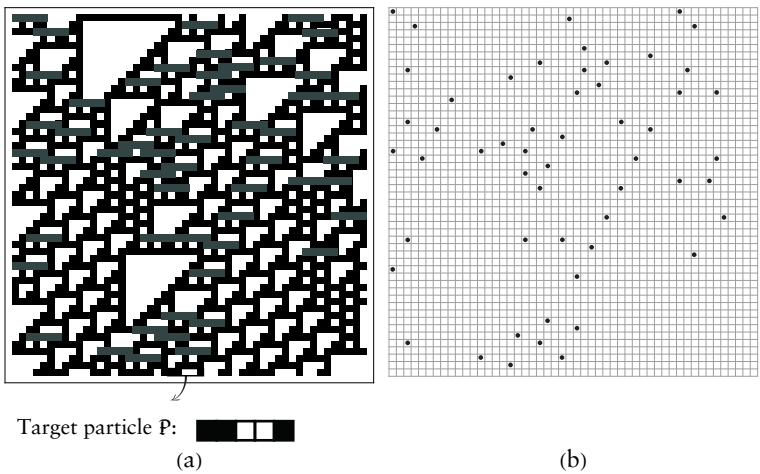


**Figure 3.** (a) An illustration of how it takes a "target particle" $\mathfrak{P}$ from a matrix generated by ECA rule 110. The rectangle with a black frame is the target particle $\mathfrak{P}$, and gray rectangles mean there is a sequence similar to $\mathfrak{P}$, which are linear particles $\mathfrak{L}$s. In this figure, the $\mathfrak{P}$ and $\mathfrak{L}$s are ■□■, and 54 same sequences are found. (b) A figure of matrix $\mathfrak{M}^{(110)}$. $\mathfrak{M}^{(110)}_{t,x} = p_0$ when there is an $\mathfrak{L}$ at $\{t, x\}$, or $\mathfrak{L}_{t,x} = 0$. Dots at $\{t, x\}$ mean $\mathfrak{M}^{(110)}_{t,x} = p_0$.

https://doi.org/10.25088/ComplexSystems.27.1.45

A particle $P$ at $(t; \vec{x})$, may have the location $\left(t'; \overrightarrow{x'}\right)$ at time $t'$ ($t' < t$). We call the particle at $\left(t'; \overrightarrow{x'}\right)$ the father particle $P_f$ of $P$. If we let $P$ be $\mathfrak{P}$, the $P_f$ will be one of the $\mathfrak{L}$s.

All $\mathfrak{L}$s in the light cone of $P$ are possibly the father particle of $\mathfrak{P}$ (i.e., $\mathfrak{P}_f$); we assume that there is one and only one $\mathfrak{L}$ as the $\mathfrak{P}_f$, and each $\mathfrak{L}$ has probability $p$ of being the $\mathfrak{P}_f$. So when there are $n$ $\mathfrak{L}$s, the probability (i.e., $p_0$) for an $\mathfrak{L}_i$ to be a father particle is:

$$p_0(p, n) = p(1-p)^{n-1}. \tag{3}$$

All the $\mathfrak{L}_i$ are drawn on a matrix $\mathfrak{M}^{(\text{rule})}$, such as in Figure 3(b). $\mathfrak{M}^{(\text{rule})}_{t,x} = p_0$ when there is an $\mathfrak{L}$ at $\{t, x\}$, or $\mathfrak{L}_{t,x} = 0$. We call $\mathfrak{M}^{(\text{rule})}$ a "probability matrix." The positions with black points will add a number $p_0$. Each black point means there is a linear particle $\mathfrak{L}$ of $\mathfrak{P}$ at $(t, x)$; $(t, x)$ is the location of the black point. $\mathfrak{M}_{t,x}$ is equal to $p_0(p, n)$.

The average $\mathfrak{M}^{(\text{rule})}$ generated with random initial lists:

$$\overline{\mathfrak{M}}^{(\text{rule})} = \frac{1}{N}\sum_{i=1}^{N} \mathfrak{M}^{(\text{rule})}_{\text{random}}, \tag{4}$$

will show some patterns that represent particles and the behavior of particles. We call $\overline{\mathfrak{M}}^{(\text{rule})}$ an "average matrix." Figure 4 shows how an average matrix was generated.

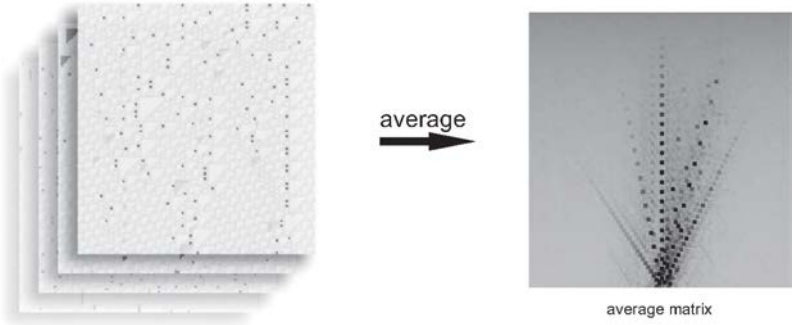


**Figure 4.** The average matrix $\overline{\mathfrak{M}}^{(110)}$, generated with $10^6$ probability matrices, with $p = 0.01$ for equation (1).

The meaning of an average matrix is: if a particle is found at the bottom center of a spacetime evolution, it may come from position $(t, x)$ with probability $\overline{\mathfrak{M}}_{t,x} \Big/ \sum_{t,x} \overline{\mathfrak{M}}_{t.x}$. So the pattern in an average

matrix represents traces of particles. We calculate the average matrix with $N = 10^6$ for each rule.

### 2.3 Extracting Particle Identity from an Average Matrix

By observing patterns of average matrices, the identity of particles can be extracted. A typical average matrix is shown in Figure 4. If particles can emerge, there will be some lines in the average matrix. Each line represents at least one particle, and their variations show interactions between particles.

The change of a line's intensity with time represents interactions between particles. If a particle is moving straight without any interactions, the line's intensity will not change through time. But if the particle can be generated by other particles, it will not be found before it was created, so that the intensity will change through time; mostly, the intensity will get higher when $t$ is getting higher.

## 3. Result

We get $\overline{\mathfrak{m}}$ for all rules; some typical $\overline{\mathfrak{m}}$ are shown in Figure 5.

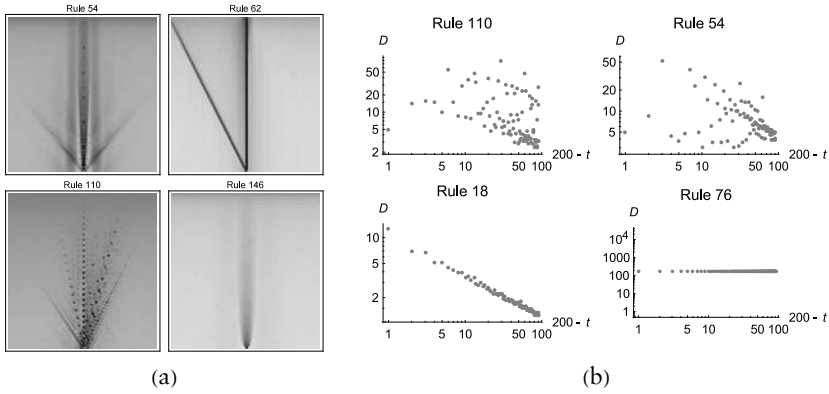


**Figure 5.** (a) Four typical $\overline{\mathfrak{m}}$. (b) The intensity change with time for four typical rules. The $t$ axis is time, and the $D$ axis is the intensity of particle traces. It can be seen that the intensity $D(t)$ may change with time for some rules.

We get the number of particle traces for each ECA rule, which corresponds to the number of particles. All traces are straight lines with various angles. For the rules shown in Figure 5, rule 54 has three traces, rule 62 has two traces, rule 110 has more than six traces, and rule 18 has a smooth trace. We use $T(\text{rule})$ to represent the count of

traces, such as $T(54) = 3$, which can be used as a parameter to classify ECAs.

The intensity of traces may change through time. The result shows that they have two kinds of behavior: (1) constant; and (2) variable (mostly, the intensity increases when $t$ increases). We use $C(\text{rule})$ to represent the existence of variability, such as $C(54) = 1$ (1 is variable, 0 is constant). These two behaviors can be used as a parameter to classify ECAs. In Figure 5, traces in rules 54, 62 and 110 are getting more obvious as time $t$ increases. Figure 5 shows how the intensity of particle traces varies with time, where $D(t) = \max(L_t)$.

Power laws show in some rules, where $D(t) \sim (t_{\max} - t)^{-\alpha}$, such as rules 146 and 18; such power laws were also found by [12] (see Figure 8).

### 3.1 Identifying Turing Machines and Potential Universal Turing Machines

To identify TMs and potential UTMs, the two parameters mentioned will be used to classify ECA rules into four classes. According to the theorem of PMs [5], when $T(\text{rule}) \geq 2$ and $C(\text{rule}) = 1$, then this ECA rule behaves as a TM and can potentially be a UTM. A PM that is a TM should have at least two particle traces, so that it is possible to have interactions between particles. Also, the intensity of traces should change, which represents that new particles can be generated during collisions. So all rules can be classified into four classes: A $T \geq 2$ and $C = 1$, B $T < 2$ and $C = 1$, C $T < 2$ and $C = 0$, and D $T \geq 2$ and $C = 0$.

Figure 6 shows the final classification for all ECA rules. Each point represents a rule for an ECA. The $x$ axis is "number of traces," and the $y$ axis represents the existence of changes in the information traces, where 0 means constant and 1 means variable. The shape of a point represents its class in Wolfram's classification.

In class A, rules have complex behaviors, and many particles with plentiful interactions can be found. The information here will be stored and processed. Then they can be considered as a TM with enough complexity and computational ability, which was considered to have connections with Turing universality [6, 13]. In class B, rules will generate some random patterns, and since the particles have too many interactions with the background, information traces are dissipated. The information here cannot be stored. In class C, rules will generate continuous or random structures without any complex behavior. Rules in this class do not have particles, or have particles but no interactions. In class D, rules will generate some structures that do not have enough interactions, which will not have any

complex behavior either. New particles cannot be generated during collisions.

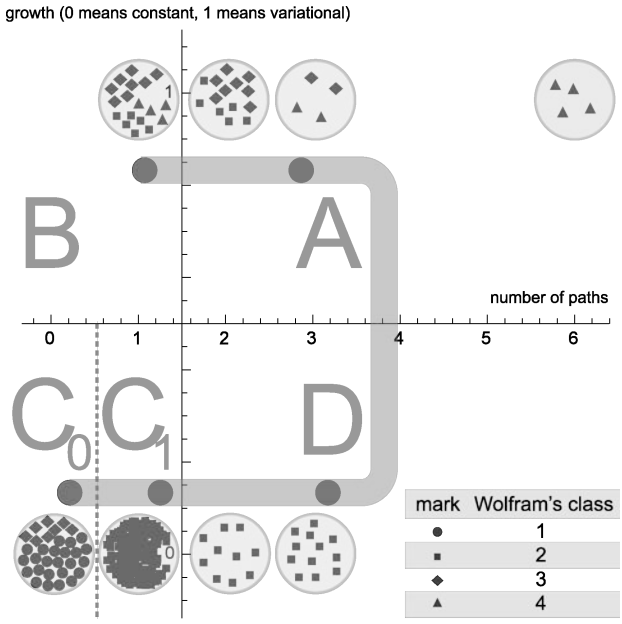


**Figure 6**. The classification of ECAs, divided by the number of paths and intensity variation. The number of paths is associated with the ability to store information, and intensity variation is associated with the ability to process information. In each phase, rules will have similar behaviors. In phase A, all rules have both a high number of traces ($T \geq 2$) and interactions that can generate particles, so that it is possible for these rules to have complex behaviors. The shape of a point represents its class in Wolfram's classification. Each point in this figure represents a rule, and their positions were moved randomly ( $\sim 0.3$) to avoid overlaps.

Class C can be divided into two subclasses, as shown in Figure 6, separated by a dotted line. We use "rule$_x$" to express the subclasses. $C_0$ means the subclass of class C with $T = 0$. $C_1$ means a subclass of class C with $T = 1$. In $C_0$, rules do not have any particles; the information here cannot be stored or processed. In $C_1$, rules have particles but do not have interactions between particles. The information here can be stored but cannot be processed.

When going through the dark curve in Figure 6 (counterclockwise), the frequency of finding interactions is continually growing. When the frequency is higher than it is in class A, it will generate too much noise, so particles and information will be scattered. When it is lower than the frequency in class A, the number of interactions is not

enough to do computation or universal computation, so the behavior is too simple to get complex behaviors.

Some typical rules in these four classes are shown in Table D.1. The classification of all rules is shown in Figure 7.

The relation between this classification and Wolfram's classification was also studied. According to Figure 8, classes $C_1$ and $D$ have a strong correlation to a certain Wolfram class, which is class 2. Classes $A$, $B$ and $C_0$ contain some different Wolfram classes. Here the reduced entropy is used to measure the relation between the two classifications, because the Wolfram classification does not have an order. The reduced entropy is defined as $h = H/H_{\max} = H/\log n$, where $H_{\max}$ is the maximum that entropy $H$ could be.

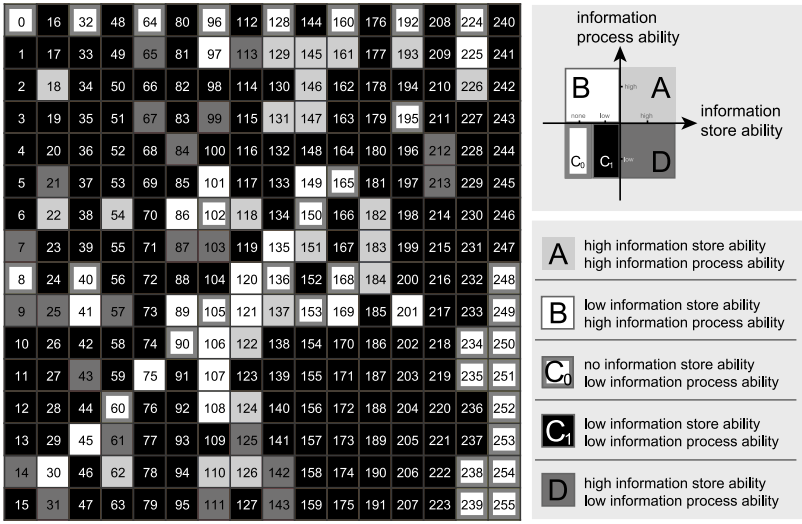


**Figure 7.** This is the final classification of all ECA rules with the method introduced in this paper. In this matrix, each kind of texture or color represents a class defined by this paper (see the column at the right side), and the numbers over each square are the rule indexes. Each texture is associated with the ability of processing and storing information, which corresponds to the computational ability. Class A, which has both high information storage and processing ability, is considered as having a high computational ability. Rule 110, which is a UTM, is classified into this class.

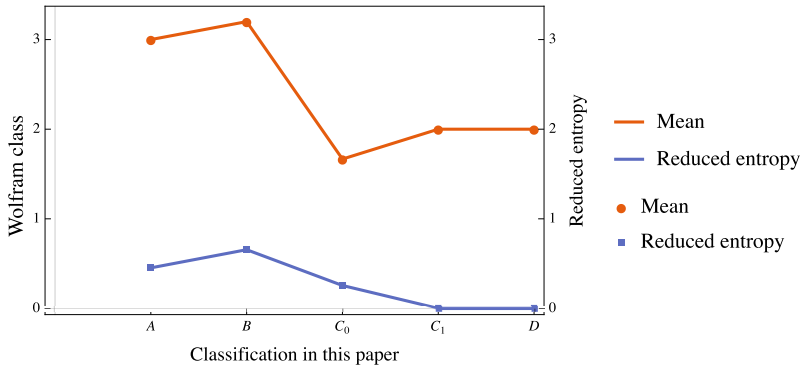


**Figure 8.** This figure shows the relation between Wolfram's classification and the classification in this paper. The orange line with dot markers is the average of $W_i$, which is Wolfram's classes of the rules in class $i$ of this paper. The average numbers only make sense when all rules in a certain class (of this paper) have the same Wolfram class, because the Wolfram class does not have an order. The blue line with square markers is the reduced entropy of the Wolfram classes of rules in a certain class in this paper, which can measure the correlation between these two kinds of classification. The reduced entropy is defined as $h = H / H_{\max} = H / \log n$, where $H$ is the entropy of $W_i$, and $n$ is the length of $W_i$.

## 4. Discussion

In this study, we consider two abilities as key dynamics for computation:

1. Ability to store information.
2. Ability to process information.

The ability to store information means there should be particles emerging in a system so that information can move in the system. In this way, the whole system can be connected and linked to be an entirety, which was considered as a common feature of complex systems. The ability to process information means the system can compute information and execute algorithms.

By using the coarse-grained method, robust patterns can be found, and rules with different computational abilities are classified into a particular class (class A, shown in Figure 6).

All ECA rules can be classified into four classes, which correspond to Wolfram's classification. All rules in class 1 and most rules in class 2 (Wolfram's classification) were found to not have interactions that can generate new particles. Most rules in class 3 are found to not have enough particles to perform universal computation. All rules in

class 4 are found classified into class A in this study. Rules 146, 183, 18 and 22, which are classified into class 3 (chaotic) by Wolfram, are classified into class A in this study, which means these rules are capable of doing complex computations. This result corresponds to the research in [12]. Particles and interactions are found in rule 146, and it is shown that the intensity of traces in the average matrix corresponds to [12]. The differences of the classifications between this paper and Wolfram come from the different criteria. For example, in Wolfram class 2, some rules show particle interactions and others do not; these were classified into different classes in this paper.

Since the problems of storing and processing information can be found in various fields, such as chemical systems [14] and hydrodynamics [15, 16], and this method is not based on specific features of ECAs, it can potentially be applied to other systems, such as bird flocks [17], traffic flow [18], chaotic behaviors [15, 16] and complex networks [19]. This method can also be used to quantify the complexity of systems (universal Turing machines are considered to have the highest complexity by [3]), which will make people have a deeper understanding of complex behaviors.

## Acknowledgments

The author is grateful for suggestions and assistance from Lingfei Wu at the University of Chicago, Qianyuan Tang at Nanjing University, Kaiwen Tian at the University of Pennsylvania and Hector Zenil at Karolinska Institutet.

## Appendix

### A. Particles in Elementary Cellular Automata

A local grid of cells in $M$ is defined as a particle in ECAs. Backgrounds are also particles, which do not have any interactions with other particles or themselves. According to the definition of particles in ECAs:

$$\mathfrak{P} = L_{t_{\max}}(p_L, p_R). \tag{A.1}$$

In this study, the size of a spacetime evolution is $(200, 200)$. The target particle is

$$\mathfrak{P} = L_{200}(100 - 2, 100 + 2). \tag{A.2}$$

For the formula

$$p_0(p, n) = p(1-p)^{n-1}, \tag{A.3}$$

the number of $p$ is a priori hypothesis; choosing a proper $p$ will make images clear. Figure A.1 shows that the formula with different $p$ will not change its whole behavior. Experiments show that choosing $p = 0.01$ will make average matrices clear enough.

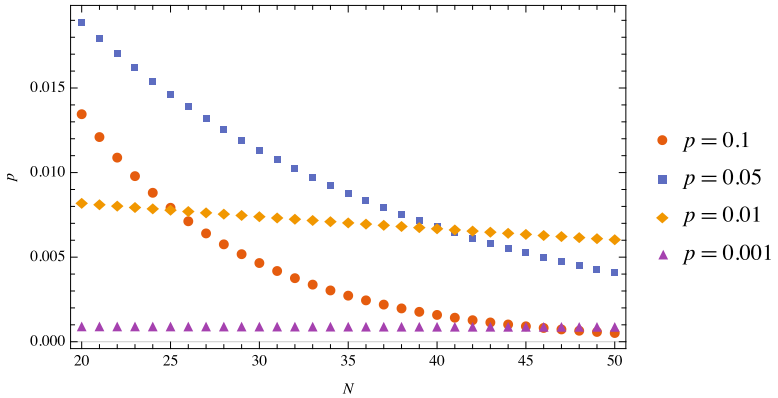


**Figure A.1.** Relation of $p_0$ with different $p$ and $N$.

## B. Particles in Rule 146

Figure B.1 shows particles in spacetime for rule 146. These particles are also introduced by [12].

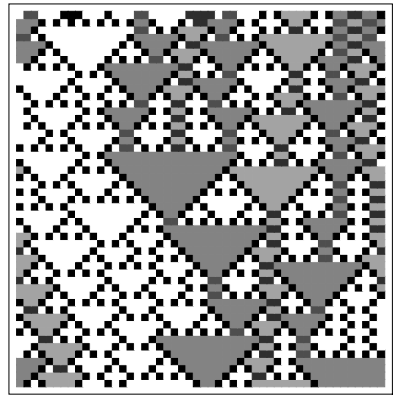

**Figure B.1.** Particles are found in rule 146.

## C. Changes of Line Intensity

The change of a line's intensity with time represents interactions between particles. If a particle moves straight without any interactions, the intensity of the line will remain unchanged through time. But if the particle can be generated by other particles, it will not be found before it was generated, so that the intensity of lines will change through time; mostly, the intensity will get higher when $t$ is getting higher. To get the particle changes through time, we define a function $D(t)$ to get the intensity of the paths:

$$D(t) = \max(L_t). \tag{C.1}$$

Figure C.1 shows the procedure of extracting the growth pattern of particles and three examples for rules 149, 2 and 26.

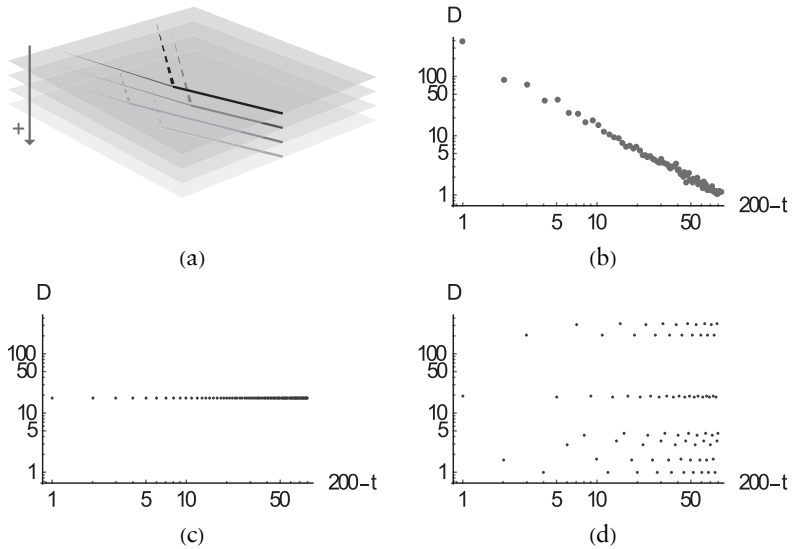


**Figure C.1.** Extracting growth pattern of particles. (a) The growth of particle intensity represents interactions of particles. (b) An example of particle intensity, generated with rule 149, which has a growth pattern. (c) Generated with rule 2, which does not have a growth pattern. (d) Generated with rule 26, with multiple particles that all do not have a growth pattern.

### C.1 The Growth of Particle Traces' Intensity for Rule 146

Particles were found in rule 146 (shown in Figure C.2), while also found earlier in 2010 [12]. In that study, the intensity of particles in rule 146 has a power law of the form

$$n_b(t) \sim t^{-\alpha}, \tag{C.2}$$

with $\alpha = 0.4789 \pm 0.0006$ [12]. When this formula with this number was applied to the data in this study (shown in Figure C.2), it showed a good fit result.

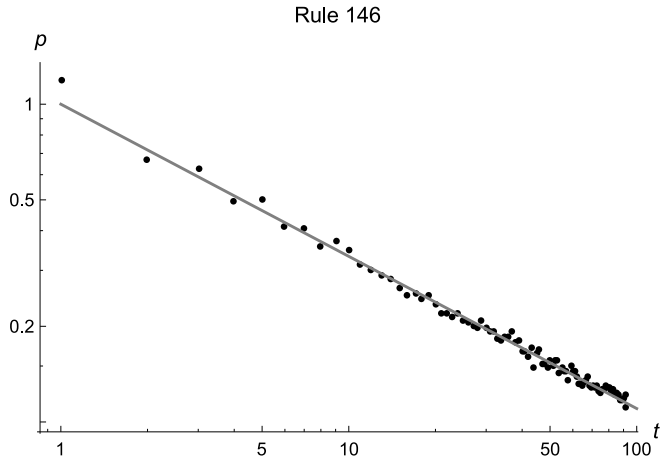


**Figure C.2.** The points are the data for the growth of particles' traces. The line is the figure of function $y = (t_{\max} - t)^{-0.4789}$, which has the same form as $n_b(t) \sim t^{-\alpha}$. It shows that the power law also is shown in this kind of measurement, and it has a good fit when using the number of $\alpha$ from [12].

https://doi.org/10.25088/ComplexSystems.27.1.45

## D. Typical Rules for Four Classes

Some spacetime evolutions of typical rules in each class are shown in Table D.1.

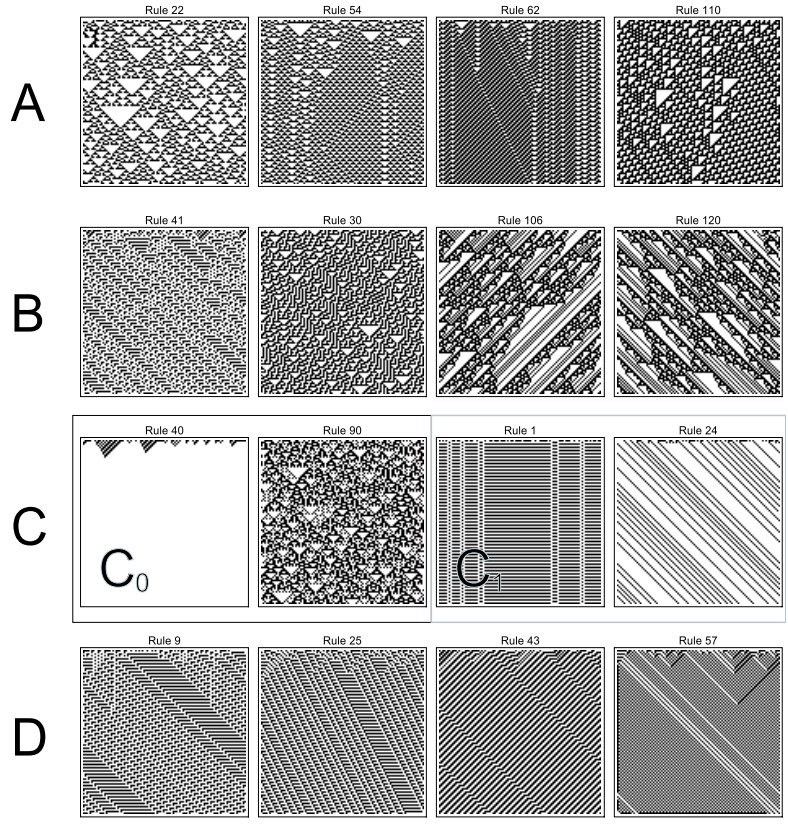


**Table D.1.** Typical rules of four classes.

## References

[1] E. R. Banks, "Universality in Cellular Automata," *IEEE Conference Record of 11th Annual Symposium on Switching and Automata Theory (IEEE, 1970)*, USA: IEEE, 1970 pp. 194–215. doi:10.1109/SWAT.1970.27.

[2] M. Cook, "Universality in Elementary Cellular Automata," *Complex Systems*, **15**(1), 2004 pp. 1–40. www.complex-systems.com/pdf/15-1-1.pdf.

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

[4] J. Cocke and M. Minsky, "Universality of Tag Systems with $P = 2$," *Journal of the ACM*, **11**(1), 1964 pp. 15–20. doi:10.1145/321203.321206.

[5] M. H. Jakubowski, K. Steiglitz and R. K. Squier, "When Can Solitons Compute?," *Complex Systems*, **10**(1), 1996 pp. 1–21. www.complex-systems.com/pdf/10-1-1.pdf.

[6] J. Riedel and H. Zenil, "Cross-Boundary Behavioural Reprogrammability Reveals Evidence of Pervasive Universality," *International Journal of Unconventional Computing*, forthcoming. arxiv.org/abs/1510.01671.

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

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

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

[10] H. Zenil, "Compression-Based Investigation of the Dynamical Properties of Cellular Automata and Other Systems," *Complex Systems*, **19**(1), 2010 pp. 1–28. www.complex-systems.com/pdf/19-1-1.pdf.

[11] A. Adamatzky and J. Durand-Lose, "Collision-Based Computing," in *Handbook of Natural Computing* (G. Rozenberg, T. Bäck and J. N. Kok, eds.), Berlin, Heidelberg: Springer, 2010 pp. 1949–1978.

[12] P.-J. Letourneau, "Particle Structures in Elementary Cellular Automaton Rule 146," *Complex Systems*, **19**(2), 2010 pp. 143–156. www.complex-systems.com/pdf/19-2-3.pdf.

[13] H. Zenil and J. Riedel, "Asymptotic Intrinsic Universality and Natural Reprogrammability by Behavioural Emulation," in *Advances in Unconventional Computing, Volume 1: Theory* (A. Adamatzky, ed.), Switzerland: Springer International Publishing, 2017 pp. 205–220.

[14] M. O. Magnasco, "Chemical Kinetics Is Turing Universal," *Physical Review Letters*, **78**(6), 1997 pp. 1190–1193. doi:10.1103/PhysRevLett.78.1190.

[15] S. Perrard, E. Fort and Y. Couder, "Wave-Based Turing Machine: Time Reversal and Information Erasing," *Physical Review Letters*, **117**(9), 2016. doi:10.1103/PhysRevLett.117.094502.

[16] D. M. Harris, J. Moukhtar, E. Fort, Y. Couder and J. W. M. Bush, "Wavelike Statistics from Pilot-Wave Dynamics in a Circular Corral," *Physical Review E*, **88**(1), 2013 011001. doi:10.1103/PhysRevE.88.011001.

[17] H. Hildenbrandt, C. Carere and C. K. Hemelrijk, "Self-Organized Aerial Displays of Thousands of Starlings: A Model," *Behavioral Ecology*, **21**(6), 2010 pp. 1349–1359. doi:10.1093/beheco/arq149.

[18] T. Nagatani, "Density Waves in Traffic Flow," *Physical Review E*, **61**(4), 2000 pp. 3564–3570. doi:10.1103/PhysRevE.61.3564.

[19] D. Brockmann and D. Helbing, "The Hidden Geometry of Complex, Network-Driven Contagion Phenomena," *Science*, **342**(6164), 2013 pp. 1337–1342. doi:10.1126/science.1245200.