Classifying Cellular Automata Automatically;

Finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter.

Andrew Wuensche

Santa Fe Institute, 1399 Hyde Park Road,
Santa Fe, New Mexico 87501 USA,
wuensch@santafe.edu, http://www.santafe.edu/~wuensch/

Abstract

CA rules can be classified automatically for a spectrum of ordered, complex and chaotic dynamics, by a measure of the variance of input-entropy over time. Rules that support interacting gliders and related complex dynamics can be identified, giving an unlimited source for further study. The distribution of rule classes in rule-space can be shown. A byproduct of the method allows the automatic "filtering" of CA space-time patterns to show up gliders and related emergent configurations more clearly.

The classification seems to correspond to our subjective judgment of space-time dynamics. There are also approximate correlations with global measures on convergence in attractor basins, characterized by the distribution of in-degree sizes in their branching structure, and to the rule parameter, Z. Based on computer experiments using the software Discrete Dynamics Lab (DDLab)[22], this paper explains the methods and presents results for 1d CA.

1 Introduction

Cellular automata (CA) are a much studied class of discrete dynamical network that support emergent behaviour resulting from homogeneous, local, short range interactions. They are applied in many overlapping areas; to model processes in physical, chemical and biological systems such as fluid dynamics and reaction-diffusion[19, 12]; to study self-organization and self-reproduction by the emergence of coherent interacting structures[9, 13]; in mathematics and computation where the systems themselves are the focus of interest [1, 6, 14]. CA dynamics are driven by complex feedback webs that are difficult to treat analytically except for special cases. Understanding these systems depends to a large extent on computer experiments, where a key notion is that state space is connected into basins of attraction[20].

The ability of CA to support the emergence of coherent interacting long-lived configurations provides a striking example of self-organization in a simple system, and has consequently become the focus of particular study. This sort of behaviour is characterized by interacting "gliders" (after Conway's 2d game-of-Life[3]). Glider dynamics can be seen from an number of overlapping perspectives: Wolfram's complex (or class 4) behaviour[18], phase transitions between order and chaos[10], computation[8, 18], and discrete analogues of Prigogine's far-from-equilibrium dissipative structures[11].

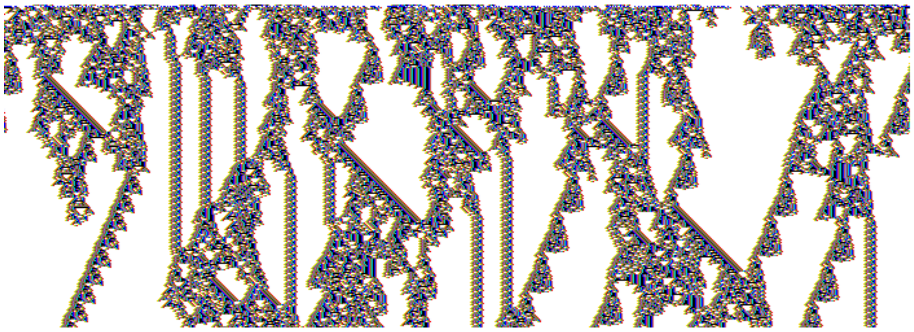


Figure 1: The space-time pattern of a 1d complex CA with interacting gliders. 308 time-steps from a random initial state. System size n=700, Neighbourhood size k=7, rule (hex) = 3b 46 9c 0e e4 f7 fa 96 f9 3b 4d 32 b0 9e d0 e0. Cells are colored/shaded according to neighbourhood look-up instead of the value. Space is across and time down the page. The basin of attraction field for this rule for n=16 is shown figure 6.

Because glider dynamics is relatively rare in CA rule spaces, their study has relied on the few known complex rules in 1d CA. A more general theory would benefit from a great many examples. Methods are described to classify rule-space automatically, for a spectrum of ordered, complex and chaotic dynamics, by a measure of the variance of input-entropy over time. This allows screening out CA rules that support glider (and related) dynamics, giving an unlimited source for further study. The resulting classification, seems to correspond to our subjective judgment of space-time dynamics. The method also gives statistical data on the distribution of rule classes in rule space, for varying neighbourhood sizes. Another useful byproduct allows automatic "filtering" of the space-time patterns of any CA to show up gliders and related emergent configurations more clearly.

The quality of dynamical behaviour of CA, from ordered to chaotic[1], is approximately reflected by convergence in basins of attraction and sub-trees (referred to collectively as attractor basins), in terms of their characteristic in-degree, which influences the length of transients and attractor cycles. The in-degree of a state is its number of pre-images (predecessors). Bushy subtrees with high in-degree imply high convergence and order. Sparsely branching subtrees imply low convergence and chaos.

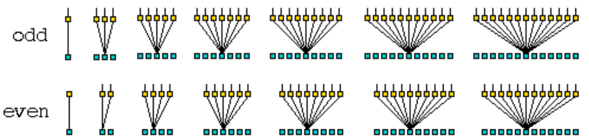
A simple convergence measure is G-density, the density of garden-of-Eden states (those without predecessors) and the rate of increase of G-density with

[1] "chaos" is used here by analogy only to its meaning in chaos theory, although there are many common properties, for example sensitivity to initial conditions.

system size. For order these measures are relatively high, for chaos relatively low. A more general measure is the distribution of in-degree sizes. Generating attractor basins and making these measures relies on a reverse algorithm that computes predecessors directly, without exhaustive testing. A consequence of the reverse algorithm is the rule parameter, Z, which predicts convergence by giving the probability that the next unknown cell in a partial pre-image is uniquely determined. This probability is calculated from the rule's lookup table[20].

This paper defines the class of 1d CA in question, rules, trajectories, basins of attraction, and how these are represented, and the methods for computing pre-images and the Z parameter. The characteristics of "gliders", the methods for filtering space-time patterns, and for automatically classifying rule-space are described, and results presented of the classified rule samples. Preliminary results are presented relating *local* measures (on trajectories), *global* measures (on attractor basins), and the Z parameter. The reasons why correlations are to be expected are discussed. The work is based on computer experiments using the author's software Discrete Dynamics Lab (DDLab)[22].

2 1d CA



The extra asymmetric cell in even k is on the right. The wiring is shown between two time-steps.

Figure 2: 1d neighbourhood templates defined in DDLab. k=0-13. Another common notation defines the radius r of a symmetric neighbourhood, $r = (k-1)/2$.

A CA is a regular network of elements (cells), taking inputs from their nearest (and next nearest etc) neighbours according to a fixed neighbourhood template, which defines the network geometry and the periodic boundary conditions. Cells synchronously update their cell-state according to a homogeneous logical function on their inputs. The cell-state ranges over a discrete alphabet, in this paper just a binary alphabet (0 or 1) is considered, and the number of cells is finite, with periodic boundary conditions. Figure 2 shows neighbourhood templates for 1d CA as applied in DDLab (see [25] for 2d and 3d templates).

A CA neighborhood of size k has 2^k permutations of values. The most general expression of the Boolean function or rule is a lookup table (the rule-table) with 2^k entries, giving 2^{2^k} possible rules. Sub-categories of rules can also be expressed as simple algorithms, concise AND/OR/NOT logical statements, totalistic rules[17] or threshold functions. By convention[17] the rule table is arranged in descending order of the values of neighborhoods, and the resulting bit string converts to the decimal or hexadecimal rule number. For example, the k=3 rule-table for rule 30,

7	6	5	4	3	2	1	0	...neighbourhoods, decimal
111	110	101	100	011	010	001	000	...neighbourhoods, binary
0	0	0	1	1	1	1	0	...outputs, the rule table

The rule-table for other k values are set out in a corresponding way. $k \geq 4$ rules are referred to by their hexadecimal rule numbers. $k \leq 3$ rules are usually referred to by their more familiar decimal rule numbers. The behaviour space of CA depends on the size of rule-space, 2^{2^k}, though rule symmetries effectively reduce this number. For example, the $2^{2^3} = 256$ rules in $k = 3$ rule-space reduce to 88 equivalence classes[20].

3 Trajectories and space-time patterns

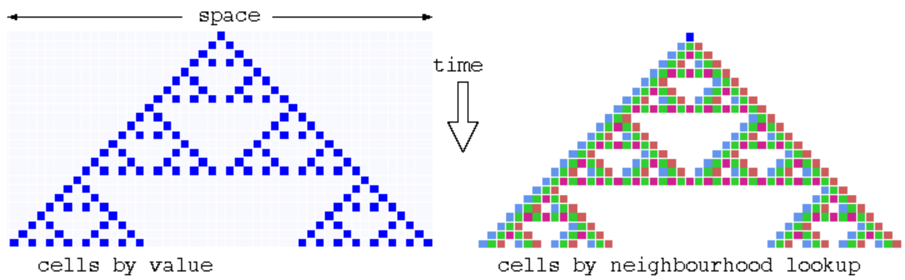


Figure 3: Space-time patterns of a CA (n=24, k = 3, rule 90). 24 time-steps from an initial state with a single central 1. Two alternative presentations are shown. *Left*, cells by value, light=0 dark=1. *Right*, cells colored (or shaded) according to their look-up neighbourhood .

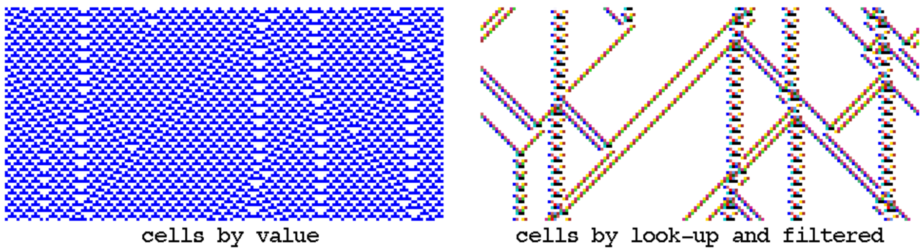
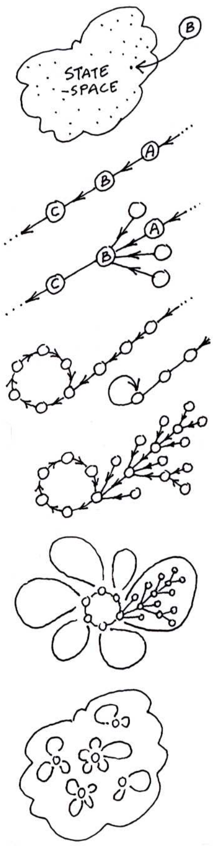


Figure 4: Space-time patterns from the same initial state showing interacting gliders, which are embedded in a complicated background. *Left*: cells by value. *Right* cells by neighbourhood lookup, with the background filtered. The k=3 rule 54 was transformed[20] to its equivalent k=5 rule (hex) 0f3c0f3c, n=150.

A state of a CA is the pattern of 0s and 1s at a given time-step. A trajectory is the sequence of states at successive time-steps, the system's *local* dynamics. Examples are shown in figures 1, 3, 4 and elsewhere. As well as showing cells as light(0) or dark(1), an alternative presentation shows cells in colors (or shades) according to their look-up neighbourhood (figure 3). The most frequently occurring colors can be progressively filtered to show up gliders and other space-time structures as in figure 4, done interactively, on-the-fly, in DDLab for any CA. This is an alternative method to the "computational mechanics" approach[8].

4 Basins of Attraction



For a network size n, an example of one of its states B might be $1010\ldots0110$. *State-space* is made up of all 2^n states, the space of all possible bitstrings or patterns.

Part of a *trajectory* in state-space, where C is a successor of B, and A is a *pre-image* of B, according to the dynamics of the network.

The state B may have other pre-images besides A, the total number is the *in-degree*. The pre-image states may have their own pre-images or none. States without pre-images are known as *garden-of-Eden* states.

Any trajectory must sooner or later encounter a state that occurred previously - it has entered an attractor cycle. The trajectory leading to the attractor is a *transient*. The period of the attractor is the number of states in its cycle, which may be only just one - a point attractor.

Take a state on the attractor, find its pre-images (excluding the pre-image on the attractor). Now find the pre-images of each pre-image, and so on, until all garden-of-Eden states are reached. The graph of linked states is a *transient tree* rooted on the attractor state. Part of the transient tree is a subtree defined by its root.

Construct each transient tree (if any) from each attractor state. The complete graph is the *basin of attraction*. Some basins of attraction have no transient trees, just the bare "attractor".

Now find every attractor cycle in state-space and construct its basin of attraction. This is the *basin of attraction field* containing all 2^n states in state-space, but now linked according to the dynamics of the network. Each discrete dynamical network imposes a particular basin of attraction field on state-space.

Figure 5: State space and basins of attraction.

The idea of basins of attraction in discrete dynamical networks (which includes CA) is summarized in figure 5. Given an invariant network architecture and the absence of noise, a CA is deterministic, and follows a unique trajectory from any initial state. When a state that occurred previously is re-visited, which must happen in a finite state-space, the dynamics become trapped in a perpetual cycle of repetitions defining the attractor (state cycle) and its period (minimum one, a stable point). The approach is analogous to Poincaré's "phase portrait" in continuous dynamics.

These systems are dissipative. A state may have multiple "pre-images" (predecessors), or none, but just one successor. The number of pre-images is the state's "in-degree". In-degrees greater than one require that transient states exist outside the attractor. Tracing connections backwards to successive pre-images of transient states reveals a tree-like topology where the "leaves" are states without pre-images, known as garden-of-Eden states. Conversely, the flow in state-space is convergent. The set of transient trees and the attractor on which they are rooted make up the basin of attraction. *Local* dynamics connects state-space into a number of basins, the basin of attraction field, representing the system's *global* dynamics. An example is shown in figures 6 and 7.

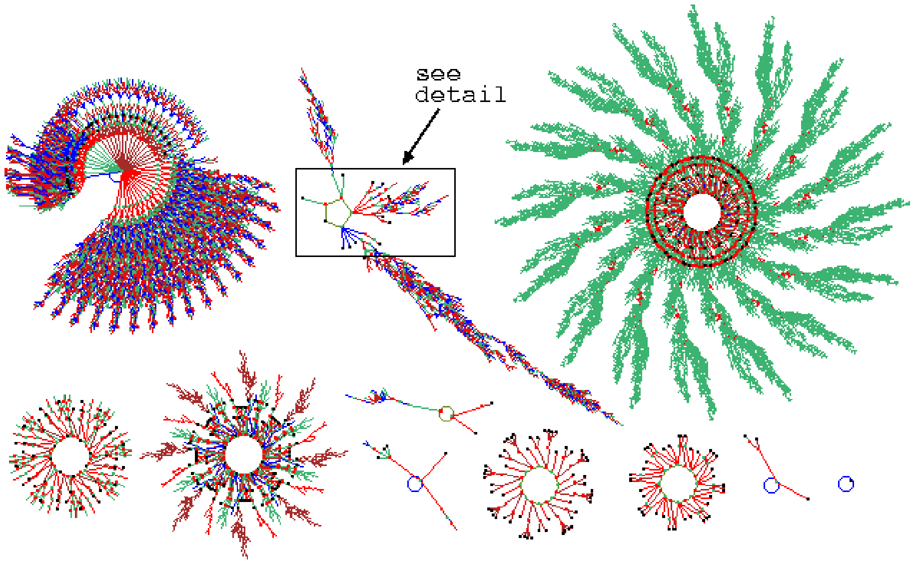


Figure 6: The basin of attraction field of the complex CA rule in figure 1. n=16, k=7. The 2^{16} = 65536 states in state space are connected into 89 basins of attraction. The 11 non-equivalent basins are shown, with symmetries characteristic of CA[20]. The period (p), percentage of state space in each basin type(s), and number of each type (t), of the biggest three basins (top row), are as follows: (1) p=1 s=15.7% t=1. (2) p=5 s=55.8% t=16. (3) p=192 s=22.9% t=1. The field's G-density=0.451, λ_{ratio}=0.938, Z=0.578.

5 Constructing and portraying attractor basins

To construct a basin of attraction containing a particular state, the network is iterated forward from the state until a repeat is found and the attractor identified. The transient tree (if it exists) rooted on each attractor state is constructed in turn. Using the reverse algorithms, the pre-images of the attractor state are computed, ignoring the pre-image lying on the attractor itself, then

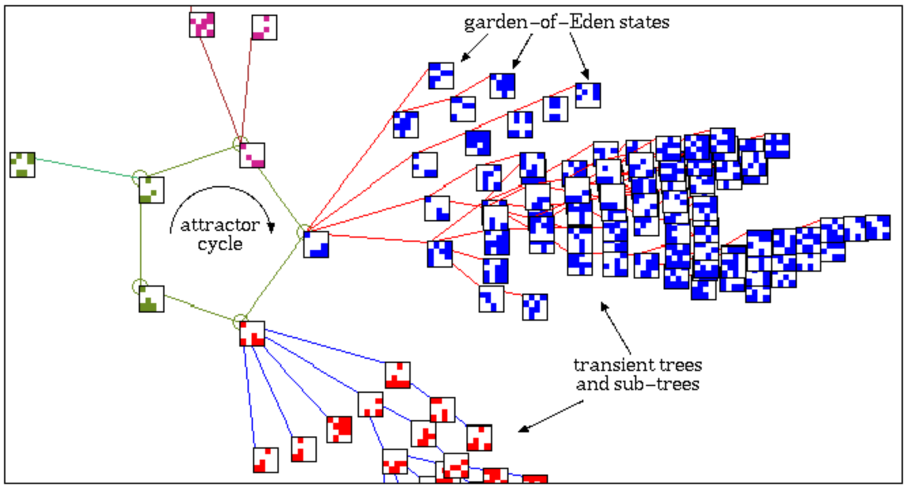


Figure 7: A detail of the 2nd basin of attraction in figure 6. The states are shown as 4×4 bit patterns.

the pre-images of pre-images, until all garden-of-Eden states have been reached.

Just a subtree may be constructed rooted on a state. Because a state chosen at random is very likely to be a garden-of-Eden state, it is usually necessary to run the network forward by at least one time-step, and use the state reached as the subtree root. Running forward by more steps will reach a state deeper in the subtree so allow a larger subtree to be constructed.

A considerable speedup in computation is achieved by taking advantage of equivalent dynamics because of rotated states, and "rotational symmetry"[20], a property of the regularity of CA and periodic boundary conditions, resulting in equivalent subtrees and basins.

Attractor basins are portrayed as state transition graphs, vertices (nodes) connected by directed edges. States are represented by nodes, by a bit pattern in 1d or 2d (as in figure 7), or as the decimal or hex value of the state. In the graphic convention[20, 22], the length of edges decreases with distance away from the attractor, and the diameter of the attractor cycle approaches an upper limit with increasing period. The direction of edges (i.e. time) is inward from garden-of-Eden states to the attractor, and then clockwise around the attractor cycle, as shown in figure 7. Typically, the vast majority of states in a basin of attraction lie on transient trees outside the attractor, and the vast majority of these states are garden-of-Eden states.

6 Computing Pre-images

CA attractor basins are constructed with an algorithm that directly computes the pre-images of network states[20, 23]. The network is run backwards in time, though backward trajectories usually diverge. The reverse algorithm takes advantage of the regularity of connections in 1d CA. It also works for mixed rule networks. Other reverse algorithms[2] and methods designed for more general discrete dynamical networks can also be applied to CA. Provided $k \ll n$, the CA reverse algorithm is in general orders of magnitude faster than the brute force method, constructing an exhaustive map resulting from network dynamics[25].

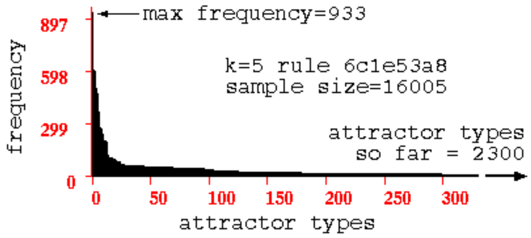


Figure 8: The attractor frequency histogram for the complex k=5 rule 6c1e53a8, n=50. The rule also appears in figures 13 and 23. Max attractor period=150, min=4, max average transient=681, min=6.

Some basic information on attractor basin structure can be found by statistical methods, first applied by Walker[15], as shown in figure 6, and are appropriate for large networks. Trajectories are run forward from many random initial states looking for a repeat in the network pattern to identify the range of attractor types reached. The frequency of reaching a given attractor type indicates the relative size of the basin of attraction, and other data are extracted such as the number of basins, and the length of transients and attractor cycles. These various methods are implemented in DDLab

6.1 The CA reverse algorithm

Consider a 1d CA size n (indexed $n-1\ldots0$) and neighbourhood k. To find the all pre-images of a state A, let P be a "partial pre-image" where at least $k-1$ bits (on the left) up to and including P_i, are known. Now find the next unknown bit to the right, P_{i-1}, consistent with the rule-table. ($\bullet$ indicates known, $\star$ unknown, bits),

	P_{i+1}	P_i	P_{i-1}	
...partial pre-image P ...	$\bullet$	$\bullet$	$\star$	compare the outputs of $P_{i+1}, P_i, \star$
		$\bullet$		with each other and with A_i
...known state A ...		A_i		

If $k=3$ (for example), the bitstring $P_{i+1}, P_i, \star$ corresponds to two neighbourhood entries in the rule-table. When their outputs, T_1 and T_2, are compared with each other and with A_i there are three possible consequences. The permutation is either deterministic, ambiguous or forbidden.

1. deterministic: if $T_1 \neq T_2$, then P_{i-1} is uniquely determined, as there is only one valid neighbourhood with the output A_i.

[2] An alternative algorithm is required for random Boolean networks (RBN) with their non-local connections and possibly mixed k. This algorithm also applies to CA of any dimension or geometry, as CA are just a sub-class of RBN. A more general exhaustive method also applies to random directed maps[25].

2. ambiguous: if $T_1 = T_2 = A_i$, then both 0 and 1 are valid solutions for P_{i-1}. The partial pre-image must be duplicated, with $P_{i-1} = 0$ in one version and $P_{i-1} = 1$ in the other.

3. forbidden: if $(T_1 = T_2) \neq A_i$, then P_{i-1} has no valid solution.

If forbidden (3) the partial pre-image P is rejected. If deterministic or ambiguous (1 or 2) the procedure is continued to find the next unknown bit to the right. However, in the ambiguous case (2), both alternative partial pre-images must be continued. In practice one is assigned to a stack of partial pre-images to be continued at a later stage. As the procedure is re-applied to determine each successive unknown bit towards the right, each incidence of ambiguous permutations will require another partial pre-image to be added to the stack, though various refinements limit its growth.

The procedure is continued to the right to overlap the assumed start string, to check if periodic boundary conditions are satisfied; if so the the pre-image is valid. The procedure is re-applied to each partial pre-image taken from the partial pre-image stack, starting at the first unknown cell. Each time an ambiguous permutation (2) occurs, a new partial pre-image must be added to the stack, but the stack will eventually be exhausted, at which point all the valid pre-images containing the assumed start string will have been found. The procedure is applied for 2^{k-1} start strings, assuming the different possible values of the first $k-1$ bits. The reverse algorithm is applied from left to right in DDLab, but is equally valid when applied from right to left.

6.2 The Z parameter

A by product of the CA reverse algorithm is the probability of the next unknown bit being *deterministic* (section 6.1(1)). Two versions of this probability are calculated from the rule-table. Z_{left} for the reverse algorithm applied from left to right, and Z_{right} for the converse. The Z parameter is the greater of these values. For Z=1 it can be shown[20] that for any system size n, the maximum in-degree, $I_{max} \leq 2^{k-1}$, because the next unknown bit is always uniquely determined, so the assumed start string of length $k-1$ may generate at most 2^{k-1} pre-images. If only one of Z_{left} or Z_{right}=1, $I_{max} < 2^{k-1}$, because at least one assumed start string must be forbidden (section 6.1(3)). At the other extreme, for Z=0, all state space converges on the state all-0s or all-1s in one step. For high Z, low in-degree (relative to system size n) is expected in attractor basins, growing at a slow rate with respect to n. Conversely, for low Z, high relative in-degree is expected growing quickly with respect to n. High Z predicts low convergence and chaos, low Z predicts high convergence and order.

The 2^k neighborhoods of size k, each indexed $k-1 \ldots 0$, each have an output T (0 or 1) which makes up the rule-table (section 2), and may be expressed as $a_{k-1}, a_{k-2}, \ldots a_1, a_0 \rightarrow T$. To calculate Z_{left} from the rule table, let n_k be the count of rule-table entries belonging to deterministic pairs, such that,

$a_{k-1}, a_{k-2}, \ldots a_1, 0 \rightarrow T$ and $a_{k-1}, a_{k-2}, \ldots a_1, 1 \rightarrow \overline{T}$ (not T)

The probability that the *next bit* is determined because of the above is given by, $R_k = n_k/2^k$. This is a first approximation of Z_{left}.

Let n_{k-1} be the count of rule-table entries belonging to deterministic 4-tuples (where "$\star$" may be 0 or 1), such that,

$a_{k-1}, a_{k-2}, \dots a_2, 0, \star \rightarrow T$ and $a_{k-1}, a_{k-2}, \dots a_2, 1, \star \rightarrow \overline{T}$

The probability that the *next bit* is determined because of the above is given by, $R_{k-1} = n_{k-1}/2^k$. This count is repeated if necessary for deterministic 8-tuples where $R_{k-2} = n_{k-2}/2^k$, 16-tuples, 32-tuples, ... up to the special case of just one 2^k-tuple which occupies the whole rule-table. These are are independent non-exclusive probabilities that the *next bit* is determined. The union of the probabilities $R_k \cup R_{k-1} \cup R_{k-2} \dots = Z_{left}$, is given by the following expression (the order of the probabilities makes no difference to the result),

$Z_{left} = R_k + R_{k-1}(1 - R_k) + R_{k-2}(1 - R_k + R_{k-1}(1 - R_k))) + R_{k-3}(1 - (R_{k-2}(1 - R_k + R_{k-1}(1 - R_k))))) + \cdots$ which simplifies to,

$Z_{left} = R_k + R_{k-1}(1 - R_k) + R_{k-2}(1 - R_{k-1})(1 - R_k) + R_{k-3}(1 - R_{k-2})(1 - R_{k-1})(1 - R_k) + \cdots$

and may be expressed as[3] $Z_{left} = R_k + \sum_{i=1}^{k-1} R_{k-1} \left(\prod_{j=k-i+1}^{k} (1 - R_j) \right)$

where $R_i = n_i/2^k$, and n_i = the count of rule-table entries belonging to deterministic 2^{k-i}-tuples. A converse procedure gives Z_{right}, and the Z parameter = the greater of Z_{left} and Z_{right}. Examples are given in [20, 23].

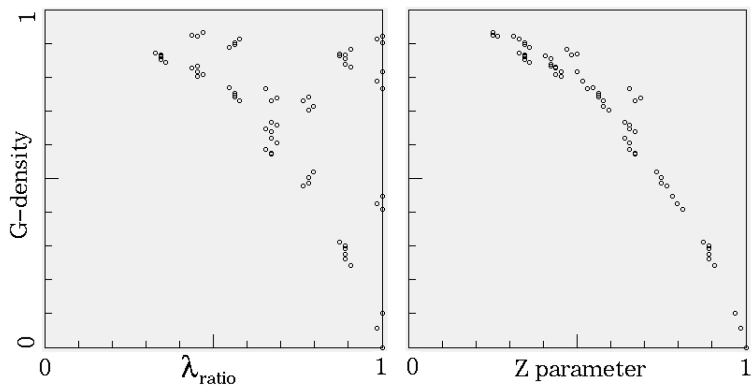


Figure 9: G-density against both λ_{ratio} and Z for the set of k=7 totalistic rules, n=16, for $Z \geq 0.25$. The complete basin of attraction field was generated for each rule and garden-of-Eden states counted.

By virtue of being a convergence parameter, Z is also an order-chaos parameter varying from 0(order) - 1(chaos). Z can be compared with Langton's well known λ parameter[10]. λ is an order-chaos parameter for CA which may have values greater than binary, and measures the density of "non-quiescent" outputs in a rule-table, so for just binary CA, $\lambda = c/2^k$ where c=the count of 1s

[3] Acknowledgment and thanks to Guillaume Barreau and Phil Husbands at COGS, Univ. of Sussex, for deriving this expression.

in a rule-table on k inputs. λ varies between 0(order)-0.5(chaos)-1(order). To allow Z and λ to be compared, a normalized version of binary λ is defined[20], $\lambda_{ratio} = 2 \times c_{min}/2^k$ where c_{min} is the count of 0s or 1s in the rule-table, whichever is *less*. λ_{ratio} must be $\geq Z$, and varies from 0(order)-1(chaos) just as Z.

Plots of the G-density against both λ_{ratio} and Z for the 256 k=7 totalistic rules[4], showing the discrepancies as well as similarities, are shown in figure 9. Points plotted in the top right corner of the λ_{ratio} graph represent λ_{ratio} values that do not correspond to behaviour as expected.

7 Gliders in 1d CA

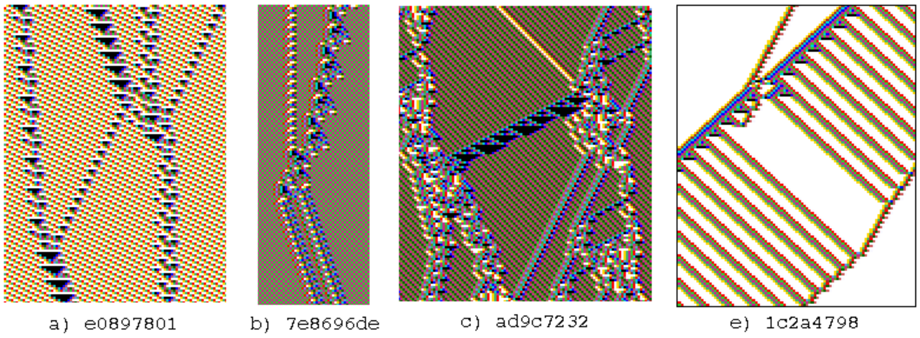


Figure 10: Interacting gliders with various velocities and backgrounds. 127 time-steps. The k=5 rule numbers are shown in hex.

A large body of literature is devoted the study space-time patterns in CA. "Glider" or "particle" dynamics, where coherent configurations emerge and interact, provide a striking example of self-organization in a simple system. Glider dynamics, and the rules that produce them, have been characterized as "complex", in contrast to ordered or chaotic, by Wolfram[16], i.e. those rules yielding localized propagating structures interacting within long transients, where the interactions are clearly "interesting". Perhaps the most dramatic example is Conway's 2d "game-of-Life"[3], from where the term "glider" is borrowed.

The human mind is extremely adept at recognizing patterns, and identifying those that seem complex and interesting, but it would be extremely useful to have measures that corresponded closely to our subjective classification. An entropy variance measure on the dynamics seems to achieve this, allowing an unlimited source of complex rules to be found, and is further able to characterize rule-space relative to our subjective notions of order, complexity and chaos.

Complex rules are supposed to be rare[17]. Most rules are either ordered or chaotic (see figure 13), though ordered rules become increasingly rare for larger

[4]The 256 k=7 totalistic rules reduce to 136 non-equivalent rules in 72 clusters, having equal λ_{ratio} and Z[23].

k. In k=3 rule-space the only two sets of glider rules that occur, rule 54 and 110, and their equivalents[20](see figures 4 and 14) have been the focus of particular study (e.g. [8]).

How a rule is placed within a notional order-complexity-chaos space has depended largely on our subjective judgment of typical emergent space-time patterns. Each CA rule self-organizes its patterns in a characteristic way, and these are often recognizable, especially for small k, where a characteristic pattern is apparent even when chaotic, becoming less obvious for larger k. The characteristic pattern of different rules can be analyzed in formal language theory as a "regular language" with a vocabulary made up of bit sequences and a "grammar" made up of succession rules between sequences[18], and by a related "computational mechanics" approach[8].

Glider dynamics corresponds to Wolfram's complex class 4 behaviour in his classification of CA dynamics[17]. Wolfram orders his classes according to a specific notion of complexity; by the increasing complexity of typical space-time patterns as measured in formal language theory[18], and draws analogies with classical continuous dynamical systems in terms of the attractors typical of each class. His classes are as follows:

Class	CA dynamics evolves towards...	Dynamical systems analogue
1.	A spatially homogeneous state...	Limit points.
2.	A sequence of simple stable or periodic structures................	Limit cycles
3.	Chaotic aperiodic behaviour.......	Chaotic (strange) attractors
4.	Complicated localized structures, some propagating......	Attractors unspecified

Langton[10] and others have argued that Wolfram's class 4 more naturally belongs between classes 2 and 3, at a phase transition between order and chaos. Moreover, many ordered rules have both limit points and short limit cycles, though one or the other may predominate, suggesting that class 1 and 2 may usefully be combined. For these reasons the classification is readjusted as follows:

ordered (class 1-2) - complex (class 4) - chaotic (class 3)

What are the essential features of glider behaviour? Glider dynamics occurs if a limited set of gliders emerge from random initial states, and if the interactions between gliders persist for an extended time, which requires that at least some glider collisions create new gliders. Gliders are embedded in a uniform or periodic space-time background or "domain", which of necessity has simultaneously emerged. Such a regular domain may be simple, for example a checkerboard, or a have a more complicated pattern (see figures 10-12).

Distinct chaotic domains may also occur, which cannot support geometrically regular gliders, but may support "domain walls" or "particles"[4], analogous to gliders. These arrise from defects within a chaotic domain (see figure 9), a boundary between two distinct chaotic domains, or between a chaotic and

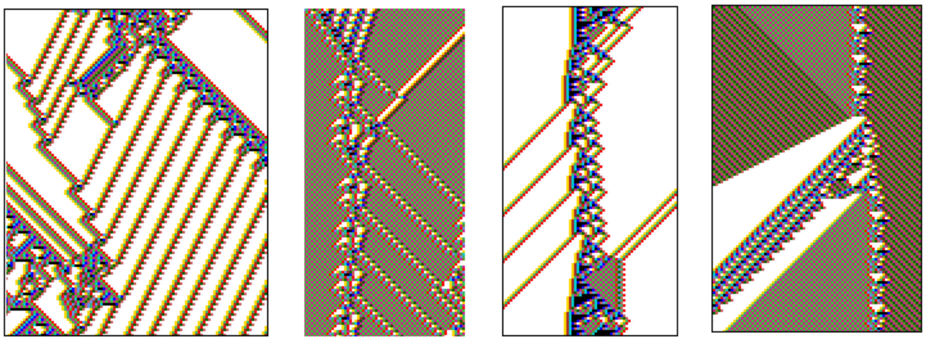


Figure 11: Examples of glider-guns. 127 time-steps. The k=5 rule numbers are shown in hex.

regular domain. Domains may be filtered as described in section 9 to show up gliders and domain walls more clearly.

Gliders may be regarded as solitary waves within a regular domain, and may have the special property of solitons [2], preserving their shape and velocity after interacting with other solitons. Glider velocity varies from 0 to a maximum "speed of light" of r cells per time step towards the left or right, where r is the number of cells on the left or right of the target cell. A glider configuration that repeats at each time-step, i.e. with period one, is limited to velocities of 0,1,2,...,r per time-step. Gliders with greater periods may have intermediate fractional velocities. A glider's attributes are the regular domain pattern and spatio-temporal period (on both sides of the glider), the glider's temporal period and velocity, its changing diameter, and the list of its repeating configurations. The same description might be applied recursively to each sub-glider component of a compound glider.

Collisions between two glider types often result in a third glider type (or more). One or both gliders may survive a collision with a possible shift in trajectory, or both gliders may be destroyed. A collision may create a temporary chaotic phase before new gliders emerge. The outcome of a collision is sensitive to the point of impact. A glider is often a dislocation or defect of varying width in a domain, which is out of phase on either side of the glider, analogous to a fracture plane in a crystal lattice. Alternatively, a glider may be seen as the zone that reconciles two out-of-phase domains. A glider may separate two entirely different domains, acting as the boundary, as in figure 10(d). Gliders that eject a stream of sub-gliders at regular intervals, as in figure 11, and gliders that survive by absorbing a regular glider stream, as in figure 10(d), are relatively common. They are analogous to "glider-guns" and "eaters", some of the components of the "game-of-Life" universal computer[3]. Because a regular glider stream is essentially the same as a regular domain, a glider-gun creates a domain, and an eater absorbs it, so glider-guns/eaters are equivalent to a glider forming the boundary between two domains.

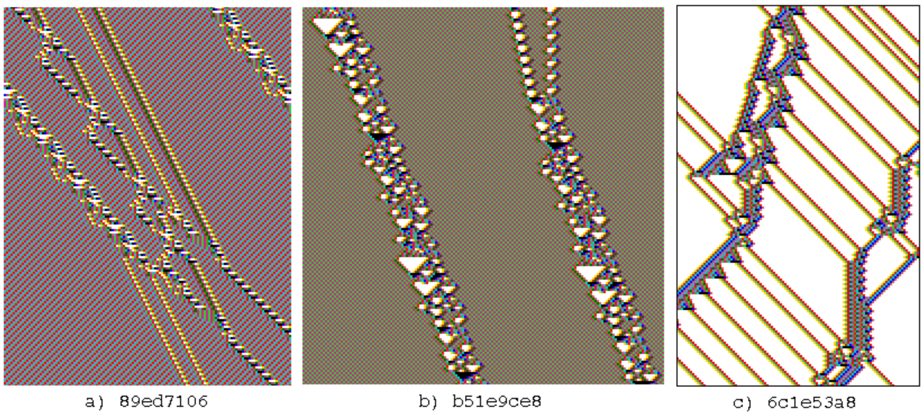


Figure 12: (a) A compound glider, (b) a glider with a period of 106 time-steps, (c) a compound glider-gun. 168 time-steps. The k=5 rule numbers are shown in hex.

Both the period and diameter of a glider may be considerable. The diameter may show a large variation within the period. Clearly gliders can only emerge in systems large enough to contain them, so that the samples described in section 11 based on n=150 are biased towards finding relatively small diameter gliders.

The existence of compound gliders made up of sub-gliders colliding periodically may be expected in large enough systems. The example in figure 12(a) shows a compound glider made from two independent gliders locked in a cycle of repeating collisions. Compound gliders could combine into yet higher order structures[9], and the process could unfold hierarchically without limit. Once gliders have emerged, CA dynamics could, in principle, be described at a higher level, by glider collision rules as opposed to the underlying CA rules.

8 Input-entropy

Keeping track of the frequency of rule-table look-ups (the k-block frequency, or "look-up frequency") in a window of time-steps, provides a measure, the variance of input-entropy over time, which is used to classify 1d CA automatically for a spectrum of ordered, complex and chaotic dynamics[24].

The look-up frequency can be represented by a histogram (figure 13) which distributes the total of $n \times w$ lookups among the 2^k neighbourhoods (shown as the fraction of total lookups), where n=system size, w=the window of time-steps defined, and k=neighbourhood size. The Shannon entropy of this frequency distribution, the "input-entropy" S, at time-step t, for one time-step (w=1), is given by, $S^t = -\sum_{i=1}^{2^k} \left(\frac{Q_i^t}{n} \times log \left(\frac{Q_i^t}{n} \right) \right)$, where Q_i^t is the look-up frequency of neighbourhood i at time t. In practice the measures are smoothed by being taken over a moving window of time-steps (w=10 in figure 13).

Figure 13 shows typical examples of ordered, complex and chaotic dynamics in 1d CA, with input-entropy plots and a snapshot of the lookup frequency

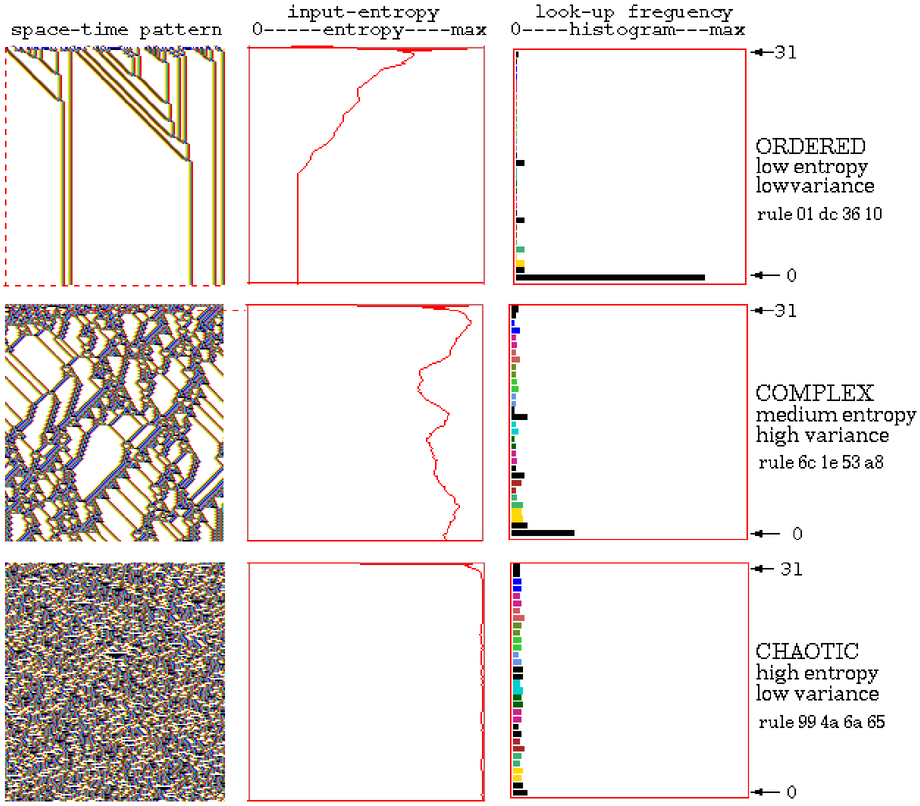


Figure 13: Typical 1d CA space-time patterns showing ordered, complex and chaotic dynamics (n=150, k=5). Alongside each space-time pattern is a plot of the input-entropy (*centre column*), and a snapshot of the look-up frequency histogram averaged over the last 10 time-steps. Only complex dynamics (*centre row*) exhibits high input-entropy variance.

histogram alongside. In a random initial state the different k-blocks occur with equal probability. The start entropy will be correspondingly high. The typical evolution of the input frequency histogram and input-entropy for ordered, chaotic and complex dynamics, is discribed below.

Ordered Dynamics

In ordered dynamics the lookup frequency histogram rapidly become highly unbalanced, with most neighbourhoods never looked at (their lookup frequency = 0). The few remaining high frequencies settle at constant or periodic values. The entropy will settle at a low constant or periodic value, corresponding to a fixed point or short cycle attractor. Ordered behaviour produces extremely short and bushy transient trees with a high density of garden-of-Eden states (G-density). Ordered rules decrease disorder and entropy.

Complex Dynamics
In complex dynamics, the lookup frequency histogram becomes unbalanced, with large erratic fluctuations, reflected in the entropy curve. As in ordered behaviour, a proportion of neighbourhoods are never looked at again after the initial sorting out phase. After an extended time the system generally settles onto a short attractor cycle. The high frequency neighbourhoods correspond to the emergent domain(s). The low frequency neighbourhoods to the interacting gliders.

Glider dynamics is subject to two countervailing tendencies. On the one hand a tendency towards order caused by the predominance of domains. But the domains are mobile, their boundaries form the gliders. When these collide there is a tendency toward chaos. In systems of the size considered, order or chaos may predominate at different times causing the entropy to vary. For large networks where colliding and non-colliding zones coexist, the entropy variance will be reduced, to zero in the limit of infinite size.

A measure of the variability of the input-entropy curve is its variance or standard deviation[5]. High entropy variance for a significant number of time-steps implies complex space-time dynamics. This includes not just glider dynamics, but also the less frequent dynamics involving "domain walls" in chaotic domains described earlier in this section.

Chaotic Dynamics
In chaotic dynamics, the lookup frequency histogram will fluctuate irregularly within a narrow band of low values, and the entropy will fluctuate irregularly within a narrow high band, corresponding to dynamics on very long transients or cycles, analogous to strange attractors in continuous dynamical systems. Transient trees will be sparsely branched, thus will tend to be very long with relatively low G-density. Chaotic rules increase or conserve disorder and entropy.

9 Filtering

Current methods for filtering domains in CA space-time patterns are based on a "computational mechanics" approach[4, 8]. An alternative is a byproduct of keeping track of the look-up frequency described in section 8. The frequencies of rule-table look-ups in a moving window of time-steps are recorded. They are also displayed as a changing histogram (figures 13, 15). The size of the window is 10 for the examples in figures 14 and 15.

To filter background domains, successive key presses in DDLab will progressively suppress the printing of cells that updated with reference to the currently most frequent unsuppressed neighbourhood. A dot is shown alongside the look-up frequency histogram indicating which neighbourhoods are currently

[5]The standard deviation is given by, $\sigma = \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}$ where x_i = deviation of each measure from the mean, and n = number of measures. The variance = σ^2.

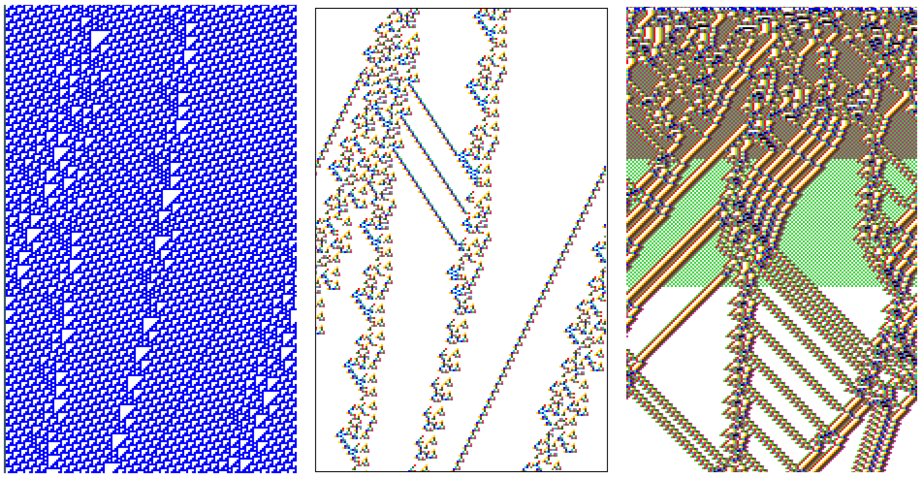


Figure 14: Examples of filtering space-time patterns to show up gliders more clearly. (*left and centre*) Space-time patterns of the k=3 rule 110, transformed to the equivalent[20] k=5 rule 3cfc3cfc, n=150, from the same evolved initial state. (*left*) cells by value, (*centre*) cells by neighbourhood lookup, and filtered. (*right*) Space-time patterns of the k=5 rule 360a96f9 from a random initial state. Cells are shown by neighbourhood lookup, and are progressively filtered in 2 stages. About 200 time-steps are shown in each case.

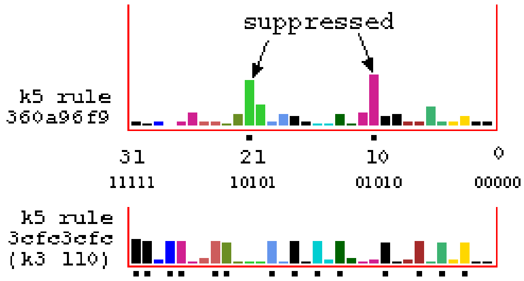


Figure 15: Look-up frequency histograms relating to figure 14. (*above*) k=5 rule 360a96f9, (*below*) k=3 rule 110 transformed to k=5 rule 3cfc3cfc. Suppressed neighborhoods are indicated with a dot.

suppressed. The routine can be continued until all neighbourhoods are filtered, and reversed to progressively unfilter. Particular neighbouhoods can be filtered in isolation. Filtering can be done *on the fly* in DDLab for any rule, including 2d and 3d CA[25].

For most glider rules, only a few neighbourhoods need to be suppressed to filter domains. Rules with very complicated domains, such as the k=3 rules 54 and 110, must first be transformed to equivalent rules[20] with greater k (k=5 in this case) for successful filtering, which requires suppressing a number of the k=5 neighbourhoods (see figures 4, 14,15).

Discontinuities may occur within chaotic domains that nevertheless have regularities in their "pattern basis"[4], as in the k=3 rule 18 (see figure 9), or between two distinct chaotic domains, or between chaotic and regular domains. These types of domains can also be filtered to uncover the "domain walls" or "particles", analogous to gliders.

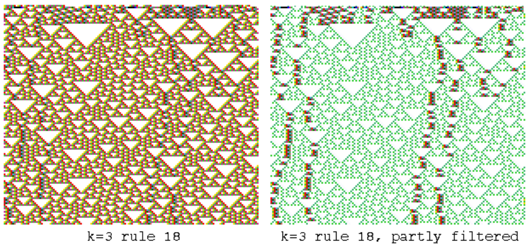


Figure 16: Unfiltered and partly filtered space-time patterns of k=3 rule 18 (transformed to k=5 rule 030c030c). n=150, about 130 time-steps from the same random initial state, showing discontinuities within the chaotic domain.

10 Entropy-density signatures

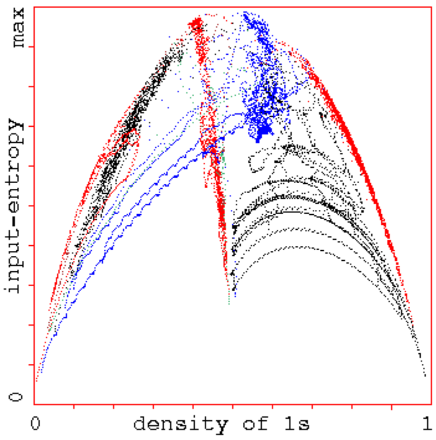


Figure 17: Entropy-density scatter plot. Input-entropy is plotted against the density of 1s relative to a moving window of time-steps w=10. k=5, n=150. Plots for a number of complex rules from the automatic sample (section 11) are show superimposed, each of which has its own distinctive signature, with a marked vertical extent, i.e. high input-entropy variance. About 1000 time-steps are plotted from several random initial states for each rule.

A related method of visualizing the entropy variance is to plot input-entropy against the density of 1s relative to a moving window of time-steps. Superimposed plots for a number of complex rules are shown in figure 17. Each rule produces a characteristic cloud of points which lie within a parabolic envelope because high entropy is most probable at medium density, low entropy at either low or high density. Each complex rule produces a plot with its own distinctive signature, with high input-entropy variance. Chaotic rules, on the other hand, give a compact cloud at high entropy (at the top of the parabola). For ordered rules the entropy rapidly falls off with very few data points because the system moves rapidly to an attractor.

Gutowitz[7] has also shown entropy-density plots for large samples of rule-space, but his plots show a single point for each rule where the measures on that rule have settled down, whereas the plots shown here focus on the transient history of the system. These plots distinguish order, complexity and chaos by the vertical extent and density of the cloud.

11 Automatically classifying rule-space

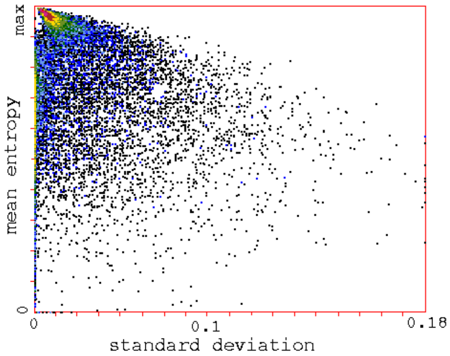


Figure 18: Classifying a random sample of k=5 rules by plotting mean entropy against the standard deviation of the entropy. Any standard deviation above the maximum scale has been rescaled to the maximum of 0.18.

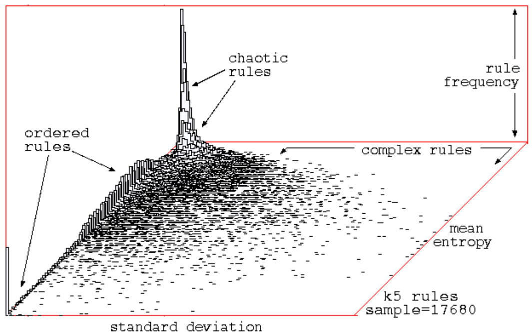
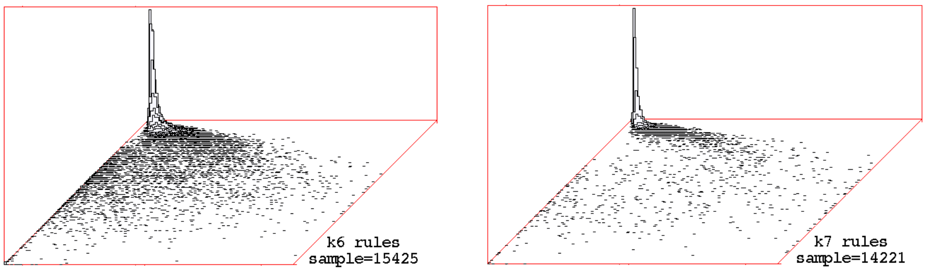


Figure 19: *left*: Classifying a random sample of k=5 rules by plotting mean entropy against standard deviation of the entropy, with the frequency of rules within a 128x128 grid shown vertically. *below*: Equivalent plots for samples of k=6 and 7 rules.



To distinguish ordered, complex and chaotic rules automatically, the mean input-entropy taken over a span of time-steps is plotted against the standard deviation of the input-entropy. Figures 18 and 19 summarize how random samples of k=5, 6 an 7 rules where classified by this method. For each rule, the data was gathered from 5 runs from random initial states, for 430 time-steps, discounting the first 30 to allow the system to settle, with w=5 as the size of the moving window of time-steps. The measures were averaged and a point was plotted of mean input-entropy against the standard deviation of the entropy as shown in figure 18 for the k=5 sample.

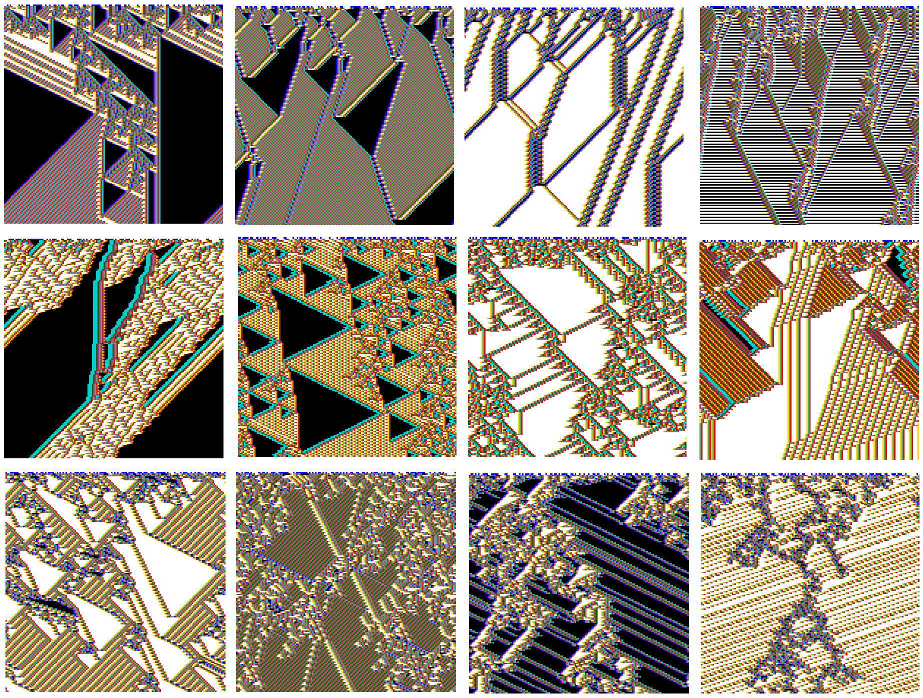


Figure 20: Examples of k=5, 6 and 7 complex space-time patterns, with high standard deviation, from the automatic samples. n=150, 150 times-steps from random initial states. Cells are colored according to the neighbourhood.

To see the frequency distribution of rules, the plots in figure 19 include an extra axis, making a 2d histogram, representing the number of rules falling within blocks on a 128x128 grid overlaid over the scatter plot. Looking at the k=5 2d histogram, the "tower" in the upper left represents chaotic rules with low standard deviation and high mean entropy. The ridge on the left represents ordered rules with low standard deviation and a spread of lower mean entropy. Complex rules have higher standard deviation, and are spread out towards the right.There is a low diagonal valley between the tower and the ridge representing a distinct boundary between ordered and chaotic rules, but a gradual transition from both towards the complex rules. As the standard deviation decreases glider interactions either become more frequent, transients longer, tending towards chaos, or less frequent, transients shorter, tending towards order. The k=6 and k=7 plots show an increasing frequency of chaotic rules and a declining frequency of ordered and complex rules as k increases. The decrease in ordered rules is especially marked.

The rule samples and measures, including each rule's λ and Z parameters, were sorted by decreasing standard deviation, and decreasing mean entropy for each measure of standard deviation, and saved to file. Examples of complex rules from the samples are shown in figure 20. More examples are presented in [23], and are all available with the DDLab software[22]. Figure 21 shows the same k=5 rules sample plotting the Z-parameter against standard deviation.

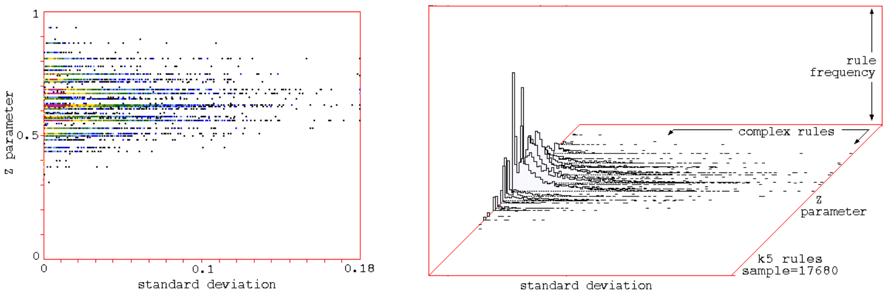


Figure 21: The same random sample of k=5 rules as in figure 18, (*left*) the Z parameter against standard deviation of the entropy, and *right* with a vertical frequency axis as in figure 19.

To check whether the expected dynamics (recognized subjectively) correspond to the measures as plotted, the dynamics of particular rules at different positions on the plots may be examined very efficiently in DDLab, for example with a mouse click on the plots in figures 18 or 21. For the mean entropy-standard deviation plot (figure 18), preliminary scans indicate that the expected behaviour is indeed found, but further investigation is required to properly demarcate the space between ordered, complex and chaotic rules and to estimate the proportion of different rule classes for different k.

For the Z parameter-standard deviation plot (figure 21), there is an approximate correlation between low Z and order, and high Z and chaos, especially at the extremes. At medium Z, between about 0.5 and 0.75, where most randomly selected rules, and also complex rules, tend to occur, the correlation becomes weaker. Z distinguishes between at least the extremes of order and chaos, and sets a band outside which complex dynamics becomes increasingly unlikely.

The automatic samples are generated by DDLab[22]. This may be done for larger system size n and and neighbourhood k, and the various other parameters describes in section 8 may be adjusted. These local measures may be compared to global measures on convergence in attractor basins described in section 12 below.

12 Attractor basin measures

Measures on attractor basins include the number of attractors, attractor periods, size of basins, characteristic length of transients and the characteristic branching within trees. The last in particular gives a good measure of the convergence of the dynamical flow in state-space.

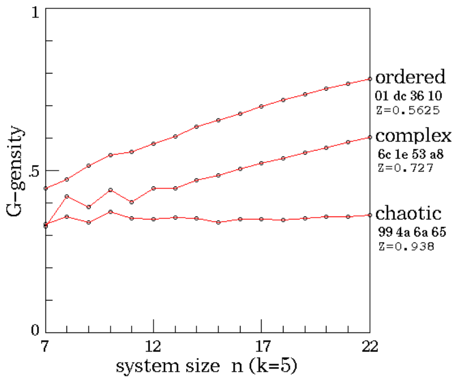


Figure 22: The G-density plotted against system size n, for the ordered, complex and chaotic rules shown in figures 13 and 23. The the entire basin of attraction field was plotted for n=7 to 22, and garden-of-Eden states counted. The relative G-density and rate of increase with n provides a simple measure of convergence.

The simplest measure that captures the degree of convergence is the density of garden-of-Eden states[21], G-density, counted in attractor basins or sub-trees, and its rate of increase with n as shown in figure 22. A more comprehensive measure is the in-degree frequency distribution, plotted as a histogram. The in-degree of a state is the number of its immediate pre-images. This can be taken on attractor basins, on just a subtree, or part of a subtree for larger systems. Subtrees are portrayed as graphs showing trajectories merging onto the sub-tree root state.

Examples of in-degree histograms for a typical sub-tree for ordered, complex, and chaotic rules are shown in figure 23. The horizontal axis represents in-degree size, from zero (garden-of-Eden states) upwards, the vertical axis represents the frequency of the different in-degrees. The system size n=50 for the complex and chaotic rules. For very ordered rules in-degrees become astronomical. The ordered rule shown is only moderately ordered, however the system size was reduced to n=40 to allow easier computation.

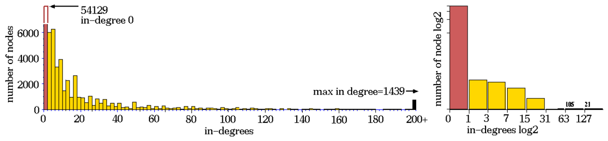
From the preliminary data gathered so far, the profile of the in-degree histogram for different classes of rule is as follows:

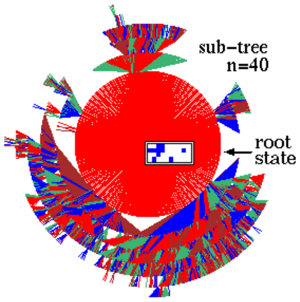
Ordered: Very high garden-of-Eden frequency and significant frequency of high in-degrees. High convergence.

Complex: Approximates a power law distribution. Medium convergence.

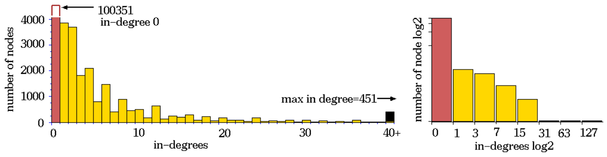
Chaotic: Lower garden-of-Eden frequency compared to complex rules, and a higher frequency of low in-degrees. Low convergence.

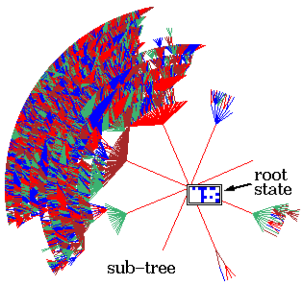
Issues for further investigation are: a systematic look at the in-degree histogram profiles relative to rules at various positions on the mean entropy/standard deviation scatter plots, how profiles change with system size, if a subtree fragment is representative of the dynamics as a whole, if the profile changes for subtrees deep in a basin of attraction as opposed to close to the outer leaves, and to look at the part of subtrees close to particular trajectories (within a given distance in reverse time-steps), especially in relation to glider dynamics.



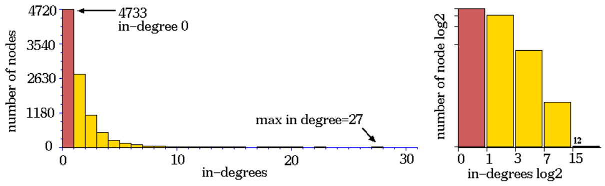


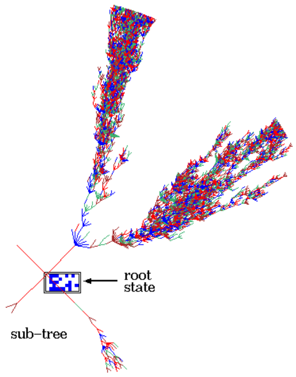
Ordered dynamics. Rule 01dc3610, n=40, Z=0.5625, λ_{ratio}=0.668. *right:* The complete sub-tree 7 levels deep, with 58153 nodes, G-density=0.931.





Complex dynamics. Rule 6c1e53a8, n=50, Z=0.727, λ_{ratio}=0.938. *right:* The sub-tree, stopped after 12 levels, with 144876 nodes, G-density=0.692.





Chaotic dynamics. Rule 994a6a65, n=50, Z=0.938, λ_{ratio}=0.938. *right:* The sub-tree, stopped after about 75 levels, with 9446 nodes, G-density=0.487.

Figure 23: Ordered - complex - chaotic CA dynamics. The space-time patterns of the rules are shown in figure 13. The in-degree histogram of a typical sub-tree shown in normal and log-log form.

13 Glider interactions and basins of attraction

It is possible to identify classes of configurations that make up different components of attractor basins in glider rules. In random states, configurations occur with equal probability, so the special glider/background configurations are unlikely. Non-glider/background states make up the majority of state-space, and are likely to be garden-of-Eden states, or states just a few steps forward in time from garden-of-Eden states. They occur in the initial sorting out phase of the dynamics and appear as short bushy dead-end side branches along the length of long transients, as well as at their tips.

States dominated by glider and background configurations are special cases, a small sub-category of state-space. They constitute the glider interaction phase, making up the main lines of flow within the long transients. This has also been noted by Domain [5], who described the main lines of flow as the topological skeleton of physically relevant states and the short dead end side branches from garden-of-Eden states as a skin of non-physical transitional states, comprising the bulk of the nodes in an attractor basin.

Gliders in the interaction phase can be regarded as competing sub-attractors, with the final survivors persisting in the attractor cycle. Finally, states made up solely of non-interacting gliders configurations (i.e. having equal velocity), or domains free of gliders, must cycle and therefore constitute the relatively short attractors, with a period depending on the glider velocity. The attractor states are made up of gliders, compound gliders or just domains, and thus form a tiny sub-category of state-space. By simply looking at the space-time patterns of a glider rule from a number of different initial states, most gliders in its glider repertoire (relative to the system size) may be identified. A complete list would allow a description of most of the attractors by finding all possible permutation of non-interacting gliders.

14 Discussion

Complex behaviour in 1d CA, especially the emergence of gliders, mirrors our intuitive notion of complex forms and processes emerging in nature. These are arguably the simplest systems where complex phenomena arise. Their simplicity allows a description of global as well as local behaviour, and how this varies across rule-space.

A global perspective on CA dynamics and rule-space is provided by the notion of attractor basins. The basin of attraction fields of complex rules are typically composed of moderately bushy transients trees rooted on relatively short attractor cycles. Gliders interacting aperiodically belong to the main lines of flow within the transient trees. Configurations where gliders interact periodically, or have ceased to interact, make up the attractor cycles.

Gliders have a distinct identity. Their interactions are predictable. A collision-table could be formulated empirically, without knowing the underlying rule-table mechanism. The collision-table would probably need to hold much more information than the rule-table. It would need to describe all possible collisions at different points of impact between gliders. However, compared to the rule-table, the collision table would provide a far more useful description of *established* behaviour, enabling some prediction of the system's future evolution, though only the rule-table could account for the origins of gliders, their emergence by a process of self-organization from random patterns.

Interacting gliders may combine to create compound gliders, interacting at yet higher levels of description, and conceivably the process could unfold hierarchically without limit in large enough systems. This is analogous to describing matter in terms of chemistry as opposed to the underlying sub-atomic particles, or in terms of biology as opposed to the underlying chemistry. There are any number of further analogies that might be drawn from nature or society.

However, the origins of the higher level entities must refer to the lower level. According to this approach, a system's complexity is the number of levels of description that underpin it in a descending hierarchy.

An unlimited source of complex rules that support gliders is available by the automatic method described, based on local measures, in particular input-entropy variance, which also classifies rule-space for a spectrum of ordered, complex and chaotic dynamics. Global measures, G-density and in-degree frequency, taken on attractor basins and subtrees, relate to the local measures. Both local and global measures relate approximately to the rule parameter, Z. Further systematic investigations of both the local and global measures, based on the automatic rule samples, and extended samples, are needed for a deeper understanding of CA rule-spaces. The computer tools for such an investigation are largely in place.

The software
"Discrete Dynamics Lab" (DDLab) was used for the computations, examples, figures and data in this paper. The software is available at:
http://www.santafe.edu/~wuensch/ddlab.html.

References

[1] Adamatzky,A., (1994) "*Identification of Cellular Automata*", Taylor & Francis.

[2] Aizawa,Y., I.Nishikawa and K,Kaneko, (1990) "Soliton Turbulence in One-Dimensional Cellular Automata", *Physica D 45*, 307-327.

[3] Conway,J.H., (1982) "What is Life?" in "*Winning ways for your mathematical plays*", Berlekamp,E, J.H.Conway and R.Guy, Vol.2, chap.25, Academic Press, New York.

[4] Crutchfield,J.P., and J.E.Hanson, (1993) "Turbulent Pattern Bases for Cellular Automata", Physica D 69: 279-301.

[5] Domain,C. and H.A.Gutowitz, (1997) "The topological skeleton of cellular automaton dynamics", Physica D, vol.103 nos 1-4, 155-168.

[6] Gutowitz,H.A., ed., (1991) "*Cellular Automata, Theory and Experiment*", MIT press.

[7] Gutowitz,H.A., and C.G.Langton, (1995) "Mean Field Theory of the Edge of Chaos", in Proceedings of ECAL3, Springer.

[8] Hanson,J.E., and J.P.Cruchfield (1997) "Computational Mechanics of Cellular Automata, An example", Pysica D, vil. 103, 169-189.

[9] Langton,C.G., (1986) "Studying Artificial Life with Cellular Automata", *Physica D 22*, 120-149.

[10] Langton,C.G., (1990) "Computation at the Edge of Chaos", *Physica D 42*, 12-37.

[11] Prigogine,I., and I.Stengers, (1984) "*Order out of Chaos", Heinemann*".

[12] Toffoli,T and N.Margolus, (1987) "*Cellular Automata Machines, A new environment for modeing*", MIT Press.

[13] von Neuman,J., (1966). "*Theory of Self Reproducing Automata*", edited and completed by A.W.Burks, University of Illinois Press.

[14] Voorhees,B., (1996) "*Computational Analysis of One-Dimensional Cellular Automata*", World Scientific.

[15] Walker,C.C., and W.R.Ashby, (1966) "On the temporal characteristics of behavior in certain complex systems", *Kybernetik 3*, 100-108.

[16] Wolfram,S., ed., (1983) "Statistical mechanics of cellular automata", *Reviews of Modern Physics 55* no 3, 601-64.

[17] Wolfram,S., (1984) "Computation Theory of Cellular Automata", *Commun.Maths.Phys.96*, 15-57.

[18] Wolfram,S., (1984) "Universality and complexity in cellular automata", *Physica 10D*, 1-35.

[19] Wolfram,S., ed. (1986) "*Theory and Application of Cellular Automata*", World Scientific.

[20] Wuensche,A., and M.J.Lesser. (1992) "*The Global Dynamics of Cellular Automata*", Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley.

[21] Wuensche,A., (1994) "Complexity in One-D Cellular Automata", Santa Fe Institute Working Paper 94-04-025.

[22] Wuensche,A.,(1996) "Discrete Dynamics Lab (DDLab)", http://www.santafe.edu/∼wuensch/ddlab.html.

[23] Wuensche,A., (1997) "Attractor Basins of Discrete Networks" (DPhil thesis), Cognitive Science Research Paper 461, Univ. of Sussex.

[24] Wuensche,A., (1998) "Classifying Cellular Automata Automatically", Santa Fe Institute Working Paper 98-02-018.

[25] Wuensche,A., (1998) "Discrete Dynamical Networks and their Attractor Basins" Santa Fe Institute Working Paper 98-11-101. To appear in the Proceedings of Complex Systems '98, University of New South Wales, Sydney, Australia.