Int. Journ. of Unconventional Computing, Vol. 1, pp. 375–398
Reprints available directly from the publisher
Photocopying permitted by license only

Glider Dynamics in 3-Value Hexagonal Cellular Automata: The Beehive Rule

ANDREW WUENSCHE*

Discrete Dynamics Lab (www.ddlab.org) and Faculty of Computing, Engineering and Mathematical Sciences, University of the West of England

Received 8 October 2004; Accepted 7 January 2005

We present a 3-value cellular automaton which supports gliders, glider-guns and self-reproduction or self-destruction by glider collisions. The complex dynamics emerge spontaneously in both 2d and 3d according to the 6-neighbor, k-totalistic, "beehive" rule; the 2d dynamics on a hexagonal lattice is examined in detail. We show how analogous complex rules can be found, firstly by mutating a complex rule to produce a family of related complex rules, and secondly by classifying rule-space by input-entropy variance. A variety of complex rules opens up the possibility of finding a common thread to distinguish those few rules from the rest: an underlying principle of self-organization?

Keywords: cellular automata, gliders, self-reproduction, emergence, complexity, mutation, self-organization, dynamics

1. INTRODUCTION

Structure emerging by local interactions, self-reproduction and evolution; these themes are central to understanding natural processes. A system's complexity, according to this approach, relates to the number of levels on which it can be usefully described [4,5].

The simplest artificial systems able to capture the essence of these dynamical processes are cellular automata (CA), where "cells" connected on a regular lattice synchronously update their color by a logical function of their neighbor's colors. Just a tiny proportion of possible logics

* Email: andy@ddlab.org

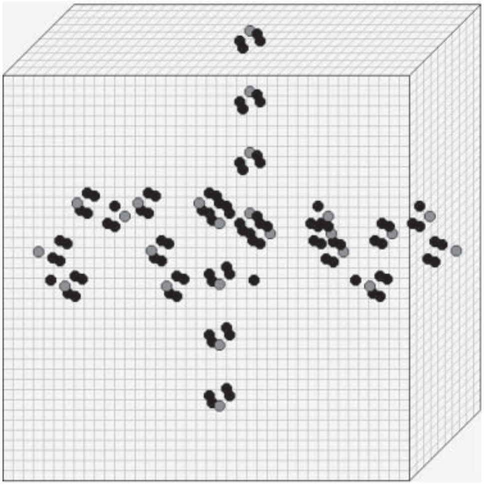


FIGURE 1
A snapshot of the beehive rule running on a 3d (40 × 40 × 10) lattice. The $k = 6$ neighborhood is shown in figure 3. The complex dynamics include the spontaneous emergence of gliders, self-reproduction by glider collisions and glider guns, analogous to the 2d case. Gliders move in the direction of their red heads. Read this figure as if looking down into a shallow box.

(complex rules) allow higher levels of description, greater complexity, to emerge from randomness.

In a movie of successive patterns on the lattice, recognizable subpatterns emerge; mobile structures (gliders[1]) interact, aggregate, make glider-guns, and gliders self-reproduce or self-destruct by colliding.

In discrete CA everything can be precisely specified: rules, connections, dynamics. So for a given complex rule it should be possible to find causal links between the underlying "physics" and the ascending levels of emergent structure. We can also ask if there is a common thread that distinguishes those few rules that support complex dynamics from the vast majority that do not: an underlying principle of self-organization? That investigation would require a good sample of diverse, independently found, complex rules. In binary CA there are many complex rules in 1d [4,5], but in 2d, Conway's Game-of-Life [3] seems to be somewhat unique in its complexity and glider dynamics. Moving up to 3-values,

[1] "Glider", "glider-gun" and other terminology is taken from John Conway's famous Game-of-Life [3]. Gliders can also be regarded as particles or waves.

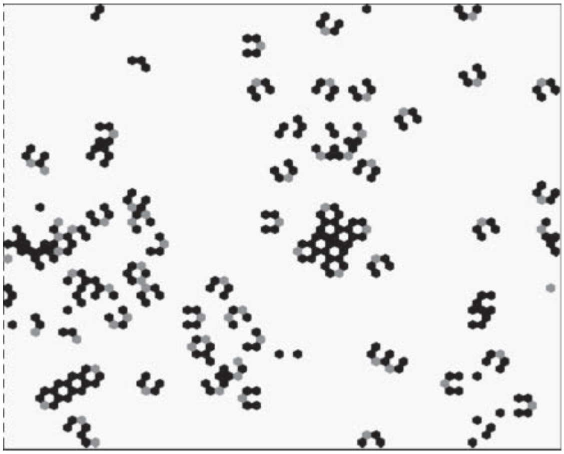


FIGURE 2
A snapshot of the beehive rule running on a 2d hexagonal lattice, 60 × 60, from a random initial state. Gliders move in the direction of their red (grey) heads.

however, as we suggest in this paper, 2d rules-space becomes relatively rich in diverse complex glider rules.

We have looked at multi-value k-totalistic rule-spaces, following recent work by Lafusa [2]. He found rules with glider dynamics and self-reproduction, using a genetic algorithm method, with an input-entropy fitness function based on our work in [5].

In our search of multi-value k-totalistic rule-spaces, we have focused on smaller lookup tables than Lafusa's. We have limited both the value-range v (range of colors) and the neighborhood k to keep our lookup tables short, and make it easier to understand how specific entries relate to gliders. We have results for $v = 3$ and $k = 4$ to 9, but we will mainly describe results for $k = 6$ on a hexagonal 2d lattice. Complex rules are easily found in these small rule-spaces by the methods we introduced in [5], where 1d rule-space can be automatically classified by input-entropy variance.

Small lookup tables also make it easier to study mutations. It turns out that a large proportion of 1-value mutations are quasi-neutral; they make little difference to the complex dynamics. Some mutations result in modified but equally interesting complex dynamics. So mutations create families of related complex rules. Of course, there are also sensitive positions in the lookup table were a mutation completely disrupts the complex dynamics.

An example of this kind of complex rule is the "beehive" rule ($v = 3$, $k = 6$). In this paper the glider dynamics and mutants of the beehive rule on a hexagonal two-dimensional lattice (figure 3) are examined in detail, and some other complex rules are described, but to a lesser degree.

Interestingly, the beehive rule also supports complex dynamics on a cubic lattice (where there are also 6 nearest neighbors, figure 3) giving analogous behavior: the spontaneous emergence of gliders, glider-guns, self-reproduction/destruction by glider collisions, and glider aggregation (as in figure 1).

2. K-TOTALISTIC RULES

In a k-totalistic rule [1,2][2] a cell's update depends just on the frequency (or totals) of values (or colors) in its neighborhood, not taking the position of colors into account, the general case.

Because of this, the dynamics conserve symmetry; whatever happens in one direction or reflection can also happen in all others. k-totalistic lookup tables (kcode) are much smaller than the general case, $G = v^k$. The size L of the kcode is given by $L = (v + k - 1)!/(k!(v - 1)!)$. For $[v, k] = [3, 6]$, $L = 28$, as opposed to $G = 729$. For greater $[v, k]$, L increases significantly. If complex behavior can indeed be found for small $[v, k]$, it is of course worthwhile to think small and deal with short kcode.

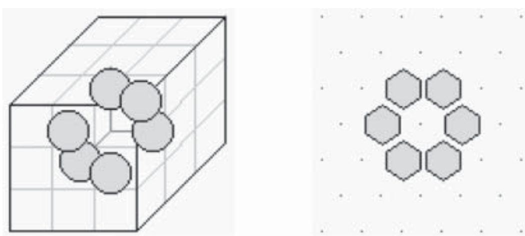


FIGURE 3
The $k = 6$ neighborhoods for a 3d cubic lattice, and a 2d hexagonal lattice.

[2] Thanks to Antonio Lafusa for introducing this class of rules to us. There is a prior attribution to Adamatzky [1], and his identical class "ATOT".

TABLE 1
The lookup table (kcode) of a [v,k] = [3,6] k-totalistic rule is constructed as follows: an output (0, 1 or 2 representing one of the 3 possible colors) is assigned to each of the 28 possible frequencies of the 3 values (or colors) in the k = 6 neighborhood. If the 3 totals in the order of 2s,1s,0s are taken as a decimal number, than the kcode follows a descending order of these numbers, with 600 (index 27) on the left and 006 (index 0) on the right, giving 0022000220022001122200021210 for the beehive rule. The kcode for other values of [v,k] are constructed similarly.

This table also shows the entries that make the basic glider in figure 4, and summarizes the consequences of all 56 possible 1-value mutations, 25 of which are quasi-neutral. 10 kcode entries (20 1-value mutations) seem to have little impact on the dynamics and could be wildcards. See the DDLab web site [6] for snapshots of all 56 mutants. Note that in DDLab [7], the beehive rule file is v3k6x1.vco

kcode = 0022000220022001122200021210

basic glider	kcode index	2s + 1s + 0s = k = 6 totals: 2 1 0		kcode		mutations 2	1	0
background→	0:	0 0 6	→	0		o	c	–
head+→	1:	0 1 5	→	1		0	–	0
	2:	0 2 4	→	2		–	Sg	cg
	3:	0 3 3	→	1	–+	G	–	G
out4	4:	0 4 2	→	2	–+	–	G	G
out3	5:	0 5 1	→	0	–+	G	G	–
out1	6:	0 6 0	→	0	–+	G	G	–
side2→	7:	1 0 5	→	0		c	c	–
side1→	8:	1 1 4	→	2		–	c	c
side1+	9:	1 2 3	→	2		–	cg	G
	10:	1 3 2	→	2	–+	–	G	G
out2	11:	1 4 1	→	1	–+	G	–	G
tail	12:	1 5 0	→	1	–+	G	–	G
head→	13:	2 0 4	→	0		c	c	–
	14:	2 1 3	→	0		Gs	c	–
	15:	2 2 2	→	2		–	gc	gc
	16:	2 3 1	→	2	–+	–	G	G
	17:	2 4 0	→	0	–+	G	G	–
	18:	3 0 3	→	0		g	c	–
	19:	3 1 2	→	2		–	c	cg
	20:	3 2 1	→	2		–	cg	Gd
	21:	3 3 0	→	0	–+	G	G	–
	22:	4 0 2	→	0		G	c	–
center→	23:	4 1 1	→	0		g	cg	–
	24:	4 2 0	→	2		–	cg	G
	25:	5 0 1	→	2		–	cg	G
	26:	5 1 0	→	0		g	gc	–
	27:	6 0 0	→	0		G	Gd	–

key to mutations:
quasi-neutral G = 25/56, wildcards –+ 10/28
G/g = gliders, G = same/similar dynamics, g = weak/different, S = spirals, d = dense,
s = sparse, c = chaos, o = order, 0 = all 0s

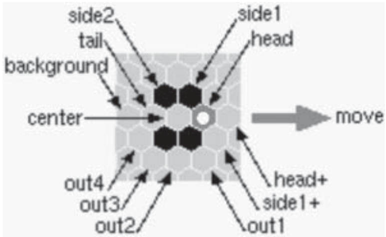


FIGURE 4
The basic $k = 6$ glider, that emerges in the 2d beehive rule, moves on the hexagonal lattice in the direction of its head (marked with a white blob). The value/colors are: head c = 1, body c = 2, background c = 0. The glider can move in 12 directions. Each cell that forms the glider and its surroundings must blink to the correct value/color at the next time-step according to the kcode; these 12 cells are indicated; they cover all cases because of symmetries. The cells are controlled by 6 kcode entries in table 1; mutations of these disrupt the dynamics (except for "center", see section 5).

3. THE BEEHIVE RULE

The beehive rule (table 1) is a multi-value k-totalistic rule with $[v, k] = [3,6]$. The rule created the snapshots in figures 1 and 2, and spontaneously self-organizes a basic glider which becomes the predominant structure in both a cubic 3d and hexagonal 2d lattice, with neighbors as in figure 3. The cell itself is not included in its neighborhood.

The complex dynamics includes self-reproduction by glider collisions (figure 8), polymer-like gliders (figure 10) and glider-guns (figure 11), but no permanently static patterns. We chose the beehive rule for closer scrutiny because glider self-reproduction is especially clear in a live simulation. In the DDLab software [7] the beehive rule is the file v3k6x1.vco; the names of other rules are given in the snapshot captions; these rules can be loaded to see the dynamics in action.

4. GLIDER COLLISIONS

We will look in some detail at the 2d dynamics, firstly the outcomes of all possible non-equivalent types of collisions between pairs of basic gliders, bearing in mind that different directions on the hexagonal lattice, and reflections, may be equivalent. Self-destruction, survival, conservation and self-reproduction all occur, depending on the exact point and direction of impact, as shown in figures 5, 6 and 7, and summarized in table 2.

TABLE 2
Summary of the outcomes of collisions between pairs of basic gliders. Details of 4 of the 21 possible types of collision are shown in figures 8. Details of all types of can be found on the DDLab web site [6].

	type	no	before	after
self-destruction:. .	$2 \to 0$	10	20	0
one-survivor:. . . .	$2 \to 1$	4	8	4
conservation:. . . .	$2 \to 2$	3	6	6
self-reproduction:	$2 \to 4$	1	2	4
	$2 \to 5$	1	2	5
	$2 \to 6$	2	4	12
	totals	21	42	31

Of the 21 collision types (8 head-on and 13 angular), 4 lead to self-reproduction, where 2 gliders release either 4, 5, or 6 gliders after an interaction phase of several time-steps, 10 collisions result in self-destruction, 4 collisions result in one glider destroying the other, and 3 collisions in both gliders surviving, but in one case one glider bounces off the other, changing direction by 180 degrees. This is summarized in table 2, and 4 examples are illustrated in figure 8. Details of all 21 types of collision, including all intermediate time-steps up to the final outcome, can be found on the DDLab web site [6].

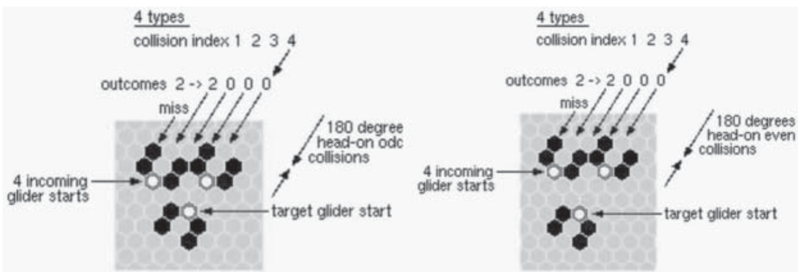


FIGURE 5
The 8 types of head-on (180 degree) collisions: These belong to 2 categories depending on the pre-collision separation, odd and even. Although the collision outcomes of odd and even are similar (summarized in the table below), the detailed interactions differ.

	no	type	no	before	after
head-on odd:	4	$2 \to 0$	3	6	0
		$2 \to 2$	1	2	2
head-on odd:	4	$2 \to 0$	3	6	0
		$2 \to 2$	1	2	2
		totals	8	16	4

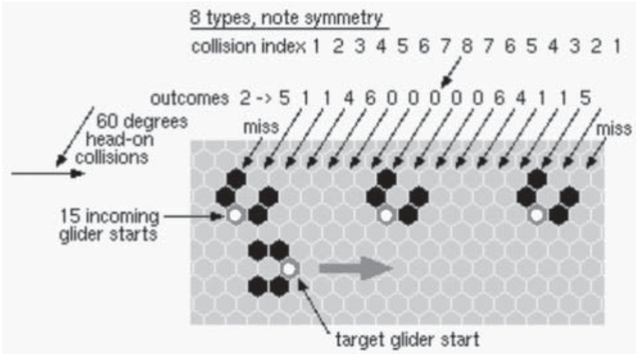


FIGURE 6
The 8 types of oblique head-on (60 degree collisions), summarized in the table below.

	no	type	no	before	after
oblige head-on:	8	$2 \to 0$	3	6	0
		$2 \to 1$	2	4	2
		$2 \to 4$	1	2	4
		$2 \to 5$	1	2	5
		$2 \to 6$	1	2	6
		totals	8	16	17

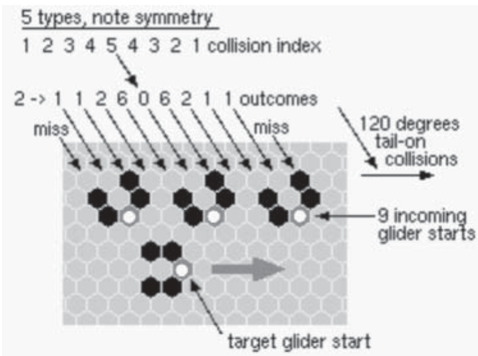
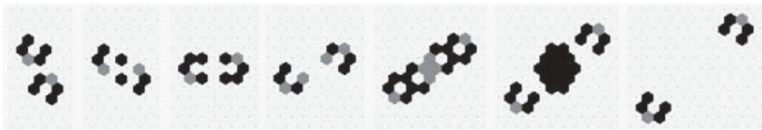


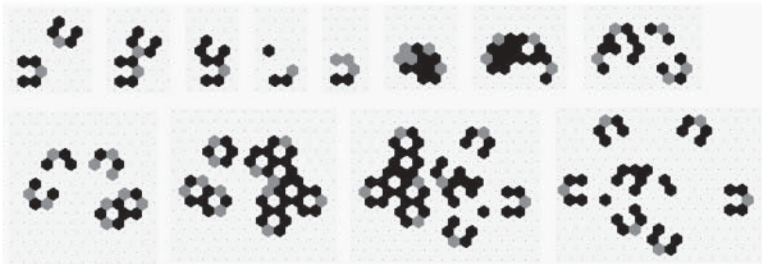
FIGURE 7
The 5 types of oblique tail-on (120 degree collisions), summarized in the table below.

	no	type	no	before	after
oblige head-on:	5	$2 \to 0$	1	2	0
		$2 \to 1$	2	4	2
		$2 \to 2$	1	2	2
		$2 \to 6$	1	2	6
		totals	5	10	10

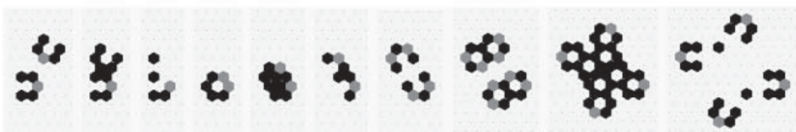


A head-on (180 degree) collision. Conservation (2→2). Gliders kiss and continue.

A tail-on (120 degree) collision. Conservation (2→2). One glider bounces off the other changing direction by 180 degrees.



A head-on (60 degree) collision. Self-reproduction (2→6). 6 new gliders emerge in 12 steps. The central c=2 cells will dissappear (change to c=0). Note that in the last step (lower right) 2 gliders are about to re-collide exactly as in the tail-on (120 degree) collision in the previous example above, but the final number of new gliders will remain 6.



A head-on (60 degree) collision. Self-reproduction (2→4). 4 new gliders emerge in 10 steps. The central c=2 cells will dissappear (change to c=0).

FIGURE 8
Four examples of collisions between pairs of basic gliders showing conservation and self-reproduction. Each panel shows the consecutive time-steps of the collision outcome. Similar details of all 21 types of collisions can be found on the DDLab web site [6].

The glider before/after ratio is 31/42, so if collision types were equiprobable, and ignoring other interactions, we would expect a high population density of gliders to decrease over time; though this is observed in the long run, other structures and interactions make the

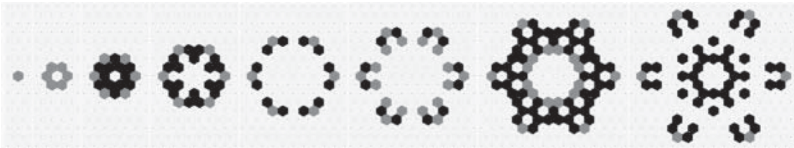


FIGURE 9
A single c = 1 cell explodes to make 6 gliders in 8 steps. The central c = 2 cells will dissappear (change to c = 0). An isolated c = 1 cell like this can be left over from the debris of other interactions.

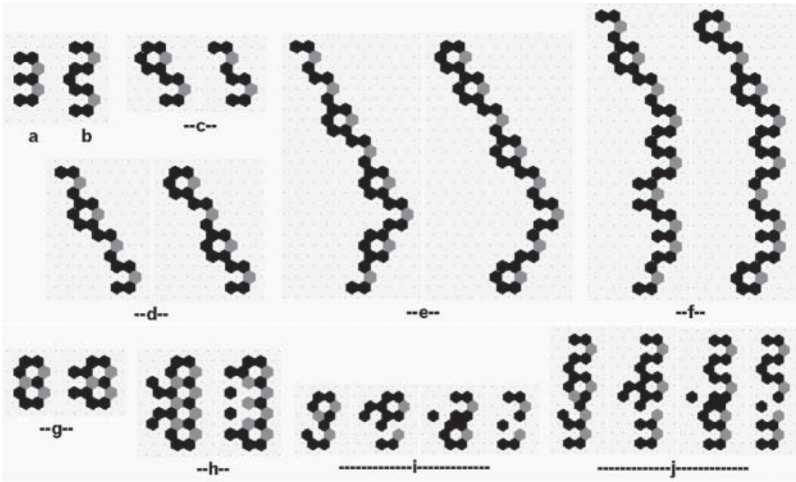
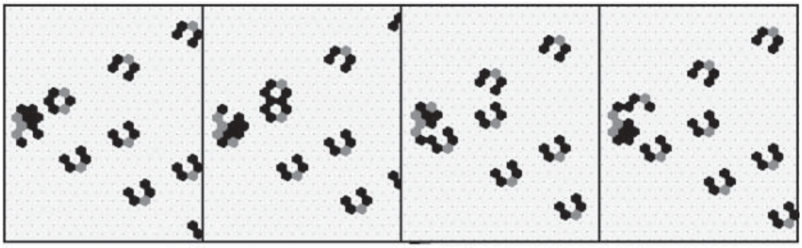


FIGURE 10
Polymer-like gliders made up of sub-units, with period 1 (a,b), periods 2 (c,d,e,f,g,h) and period 4 (i,j). The gliders move from left to right. The sub-units can be combined in a variety of ways.

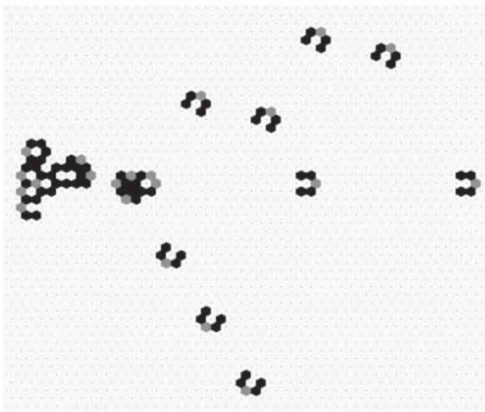
dynamics more complex. Gliders can crash into the transient patterns following collisions. An isolated red cell, from collision debris, explodes to make 6 new gliders (figure 9), so outside perturbations, noise, would tend to repopulate the space with gliders.

Polymer-like gliders made up of sub-units also emerge, as shown in figure 10. Most notably, there are a variety of glider-guns that eject from 1 to 4 (possibly more) glider streams in different directions. Some examples are given in figure 11. These processes combine with self-reproduction to produce an extremely complex hive of activity.



An example of a glider-gun (moving to the left) with period 4, shooting out 3 glider streams. The panels show consecutive time-steps, the 4 glider-gun phases.

Examples of 3 types of glider-guns (moving to the left) with period 4, shooting out 1, 2 and 3 glider streams. Only one phase is shown for each type.



A more complicated glider-gun (moving to the left) with period 8, shooting out 4 widely spaced glider streams. Only one phase is shown.

FIGURE 11
Examples of glider-guns with various periods and numbers of glider streams. The objects are a sort of cross between glider-guns and puffer-trains, as the glider-gun and gliders move in opposing directions. Further examples of glider-guns, including all their periodic phases can be found on the DDLab web site [6].

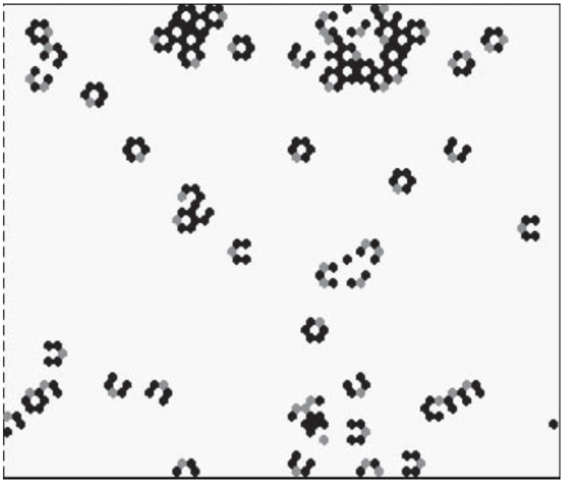


FIGURE 12
The result of a 1-value mutation to the beehive rule, at index 23 (the output 0 is changed to 2). The beehive dynamics are largely conserved, but the basic glider has an alternating open and closed tail. A snapshot on a 2d (60 × 60) hexagonal lattice.

5. MUTATIONS

The consequences of all possible 1-value mutations to the beehive rule are tabulated in table 1, and snapshots of all can be found at on the DDLab website [6].

The lookup table has 28 entries, and each can be changed from its present value to two alternatives, giving 56 possible minimal (1-value) mutations. The results of this experiment [6] show that for 10 of the entries, changing to either alternative (20 mutations) is quasi-neutral; it appears not to make much difference to the dynamics; experiment confirms that these 10 entries can actually be wildcards. A further 5 mutations elsewhere, to just one value, are also quasi-neutral, making 25/56. Multiple mutations in these neutral regions needs examining.

On the other hand, mutations to any of the 6 sensitive entries that maintain the basic glider destroy the dynamics – with one exception – a mutation at index 23 (the glider's center) which sets the glider's tail at the next time-step. A mutation at index 23 to 2 results in similar beehive dynamics, but the basic glider now has an alternating open and closed tail as it moves; a snapshot is shown in figure 12. If an additional mutation is made at index 26 (the center cell of a closed glider) to 2, then the glider will remain closed and not alternate (figure 13).

Another interesting mutation is at index 2, which causes glider activity to be gradually overwhelmed by spirals, as shown in figure 14.

The beehive kcode is set out below, indicating these mutations, the 10 wildcards (+), and the 6 glider entries (ˆ),

```
26  23                   2      --index
 |  |                    |
002200+220++200+++220++++210
    ^        ^     ^^     ^^    --the beehive glider
```

It would be possible then, to explore the family of related rules by gradually mutating away from the beehive rule, and entering into the network of related complex rules in rules-space.

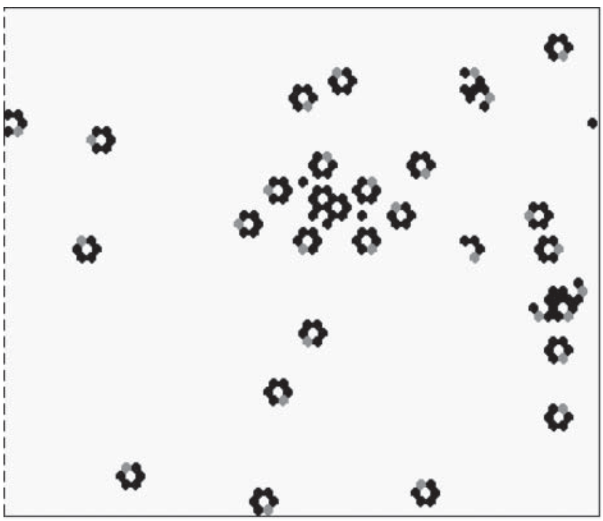
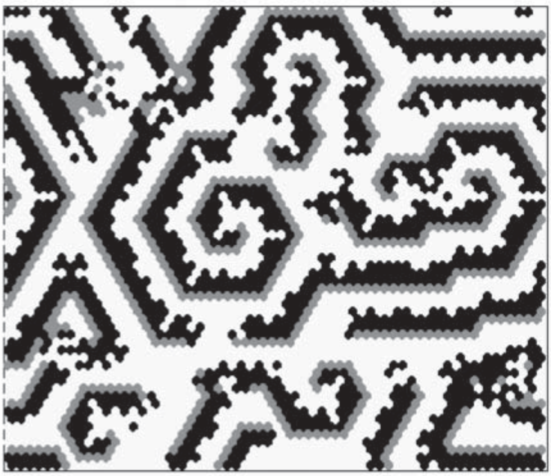


FIGURE 13
The result of 2 mutations to the beehive rule, at index 23 and 26 (the output 0 is changed to 2). The beehive dynamics is largely conserved, but the basic glider now has a permanently closed tail. A snapshot on a 2d (60 × 60) hexagonal lattice.

56 time-steps after the random initial state, spirals are beginning to emerge but gliders are still present.



208 time-steps after the random initial state, the spirals have stabilized, gliders are no longer present.

FIGURE 14
The result of a 1-value mutation to the beehive rule, at index 2 (the output 2 is changed to 1). From a random initial state gliders emerge and dominate at first, but glider activity is gradually overwhelmed by emerging spirals. Two snapshot on a 2d (60×60) hexagonal lattice, at time-step 56 and 208.

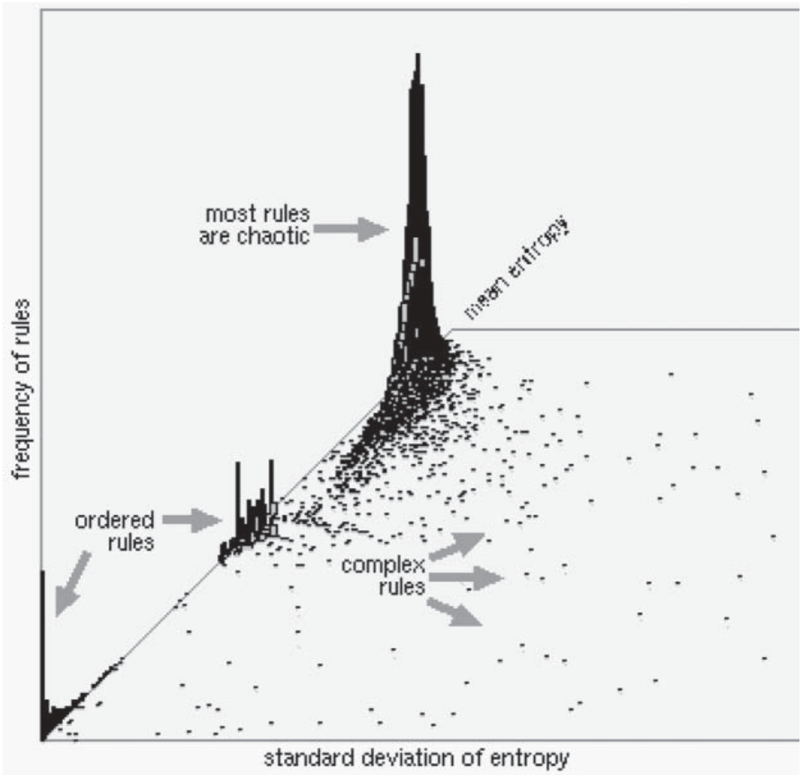


FIGURE 15
An automatically classified sample of about 15800 [v, k] = [3,6] kcode rules for 2d hexagonal CA. Axes xyz are as follows: x = standard deviation of the input-entropy, y = mean entropy, z = frequency of rules on the xy scatter plot. Complex rules can be found on the right, with higher standard deviation, chaotic and ordered rules are on the left, with low standard deviation; chaos has high mean entropy (the tower), and order lower mean entropy.

6. FINDING COMPLEX RULES

To find new complex rules from scratch (without mutating old ones), and in particular rules that support gliders, we use our method for automatically classifying 1d rule-space by input-entropy variance [5], but which applies equally well to k-totalistic rules, and to 2d and 3d.

We track how frequently the different entries in the kcode (as in table 1) are actually looked up, once the CA has settled into its typical behavior. The Shannon entropy of this frequency distribution, the input-entropy S, at time-step t, for one time-step ($w = 1$), is given by $S^t = -\sum_{i=0}^{L-1} (\frac{Q_i^t}{n} \times log(\frac{Q_i^t}{n}))$, where Q_i^t is the lookup frequency of neighborhood i at time t. L is the kcode size, and n is the size of the CA. In practice the measures are smoothed by being averaged over a moving window of $w = 10$ time-steps. The measures are started only after 200 time-steps, and are then taken for a further 300 time-steps. Each 2d hexagonal CA 100×100 is run from a sample of 5 random initial states. The sizes of these parameters can be varied, of course.

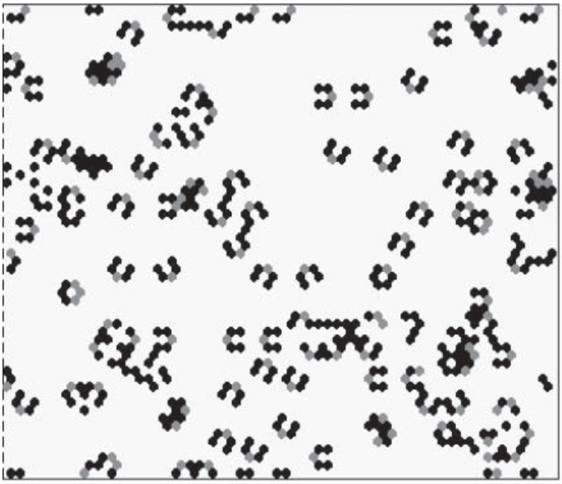
Average measures are recorded for (x) entropy variance (or standard deviation) or alternatives such as the maximum entropy up slope, and (y) the mean entropy. This is repeated for a sample of randomly chosen rules. The sample is then sorted by both (x) and (y), and data plotted as in figure 15. The plot classifies rule-space between chaos, order and complexity. The area towards the right on the scatter plot with high entropy variance (as indicated in figure 15) is rich in complex rules. Individual rules from the plot can be selected to check their behaviors; the tools are available in DDLab [7].

The basic argument is that if the entropy continues to vary in settled dynamics, moving both up and down, then some kind of self-organizing collective behavior must be unfolding. This might include competing zones of order and chaos, or different types of competing chaos, as well as glider dynamics.

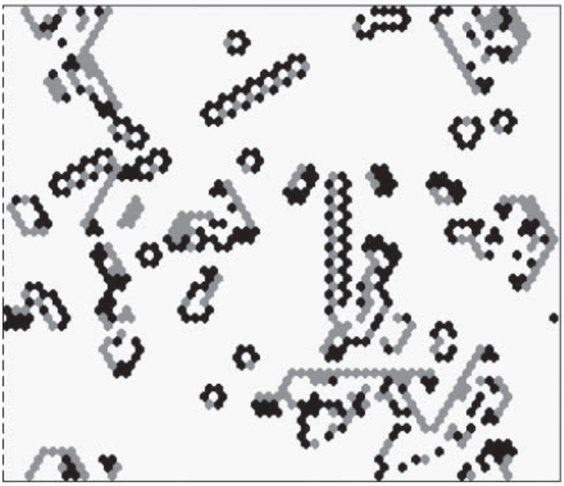
In the case of the beehive rule and other glider rules, at any given moment there may be a bias in the dynamics towards a preponderance of gliders (decreasing entropy) or post-collision transient patterns (increasing entropy). The lattice (or a patch undergoing the analysis) must not be too large in relation to the scale of possible emergent structures, otherwise the effects would cancel out. By contrast, stable/high entropy indicates chaos (most rules); stable/low entropy indicates order, but for both order and chaos the entropy variance is low.

7. OTHER K6 HEXAGONAL COMPLEX RULES

In figures 16 and 17, we show 4 examples of $[v, k] = [3,6]$ complex rules, found independently by the input-entropy variance method, more can be seen on the DDLab website [6].

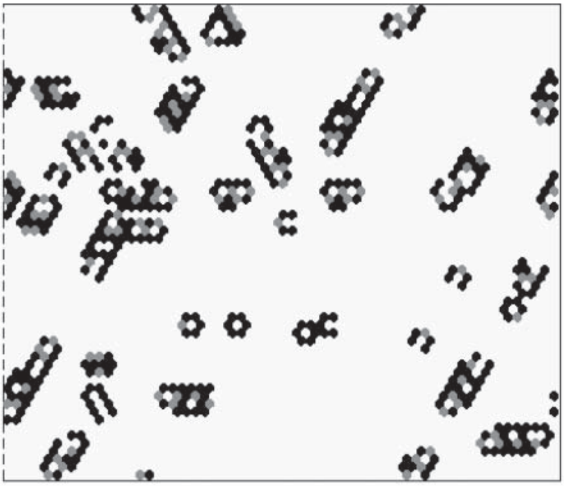


rule 1: 2200021000222201110201212210 (v3k6ex6.vco)

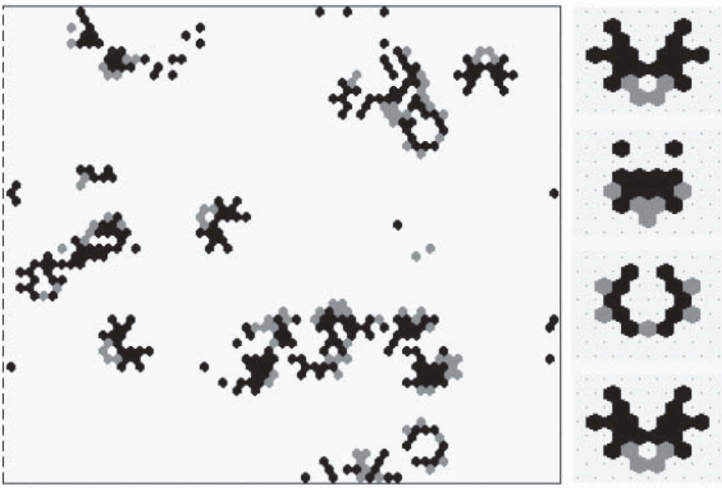


rule 2: 0222200220000200100201102110 (v3k6ex4.vco)

FIGURE 16
Other examples of $k = 6$ complex rules. Snapshots on a 60×60 hexagonal lattice from random initial states.



rule 3: 0200001120100200002200120110 (v3k6ex7.vco)



rule 4: 0200202022222200012100002100 (v3k6ex5.vco). This rule makes a slow moving glider with period 3; several gliders appear in the snapshot. Details of the glider are shown on the right. A glider gun for this rule is shown in figure 18

FIGURE 17
Other examples of k = 6 complex rules. Snapshots on a 60 × 60 hexagonal lattice from random initial states.

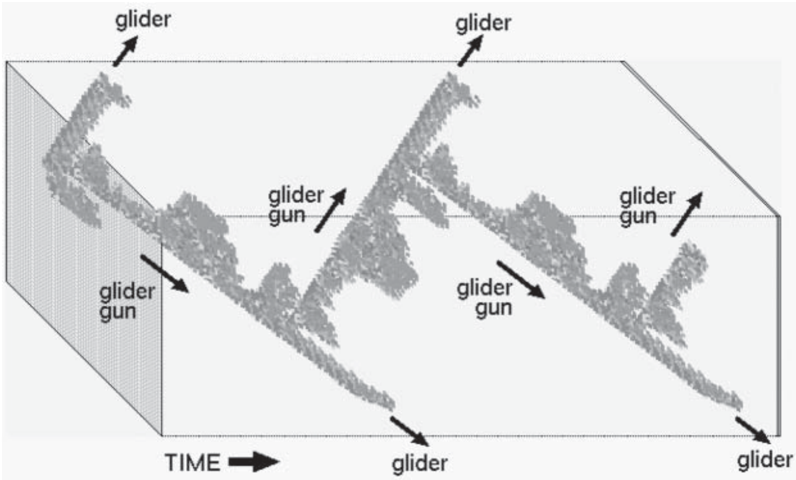


FIGURE 18
A 2-way glider-gun made by rule 4 (as in figure 17) which shoots gliders in opposite directions. This figure shows a 2d space-time pattern (60×60 hexagonal lattice) in an axonometric projection, as if looking down into a box. Time-steps are vertical slices starting on the left and moving towards the right. 238 times-steps are shown. Once the glider-gun has shot a glider in one direction, it turns itself inside-out and shoots a glider in the opposite direction. The period between firing successive gliders is 67 time-steps.

The basic beehive glider is sometimes present, but we also see different gliders and complex structures, which we have not yet examined in detail. Rule 4 (figures 17) has a remarkably complex glider, and a glider-gun which is shown as a 2d space-time pattern in figure 18.

In these examples, the kcode has been transformed with a value-swapping algorithm to make an equivalent kcode, but with the colors (values) made to correspond with the beehive rule, where the background value is 0, the leading head of gliders is 1 and the glider body is 2. This allows the different kcodes to be compared to look for common biases. Below we compare the kcodes of our 4 examples with the beehive rule, including the distance, the number of values that differ. The wildcards (+) and glider entries (ˆ) are indicated, and also the frequency of values in each kcode.

```
                26 23                                frequency of values
                 |  | +   ++   +++  ++++              2__1__0
beehive rule – 0022000220022001122200021210          11  4 13
                ^  ^               ^^     ^^                     distance
     rule 1 – 2200021000222201110201212210 –         11  7 10      16
     ruel 2 – 0222200220000200100201102110 –          9  5 14      13
     ruel 3 – 0200001120100200002200120110 –          6  6 16      15
     ruel 4 – 0200202022222220001210000210 0 –       11  3 14      16
                   |         |     ||    ||
                   2         4     34    34 – beehive glider matches
```

We can see that there is a high correlation with glider entries, except for index 23 which we have already noted is exceptional (in section 5). There is also a correlation with the frequencies of values.

If a common thread or bias in kcodes can be identified among these and other complex rules, which distinguishes them from the vast majority of rules-space, then this could become the basis for an underlying principle of self-organization in k-totalistic cellular automata.

8. DIFFERENT NEIGHBORHOODS

Although we have focused on complex rules with a neighborhood of k = 6 on a 2d hexagonal lattice, and the beehive rule in particular, v3 k-totalistic rule-space contains complex rules for other values of k, and for square as well as hexagonal latices. The neighborhood layout for k = 4 to 9, which defines the lattice, is set out in figure 19.

Examples of complex behavior can be found by the same input-entropy method described in section 6. To broaden the discussion, figures 20 to 24 give an example of a complex rule for each of the neighborhoods k4, k5, k7, k8 and k9.

The size of the lookup table (kcode) for v = 3 varies with k as follows: k4 = 15, k5 = 21, k6 = 28, k7 = 36, k8 = 45 and k9 = 55 (see section 3). The kcode outputs are ordered according to the same convention described table 1.

FIGURE 19
The neighborhoods: A square lattice applies for k4, k5, k8 and k9, and a hexagonal lattice for k6 and k7.

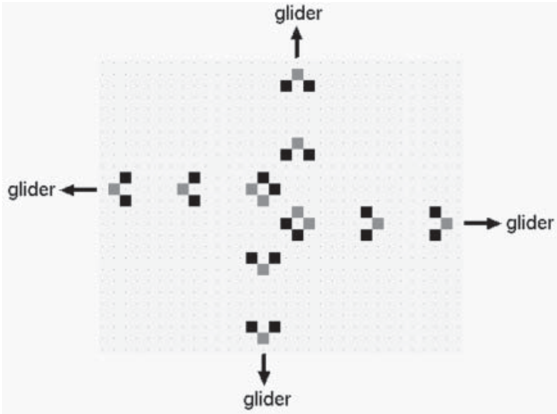


FIGURE 20
Neighborhood k = 4. kcode 202200222012210 (v3k4x1.vco). The dynamics of this complex rule includes gliders, and a 4-way glider-gun with a period of 6 time-steps shown here. A snapshot on part of a 2d square lattice.

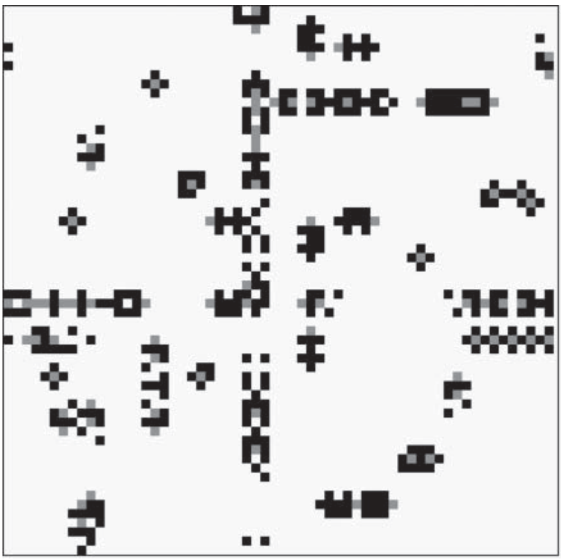


FIGURE 21
Neighborhood k = 5. kcode 010222022022220021110 (v3k4x1.vco). The dynamics include gliders which bounce off static structures. A snapshot on a 2d (60 × 60) square lattice.

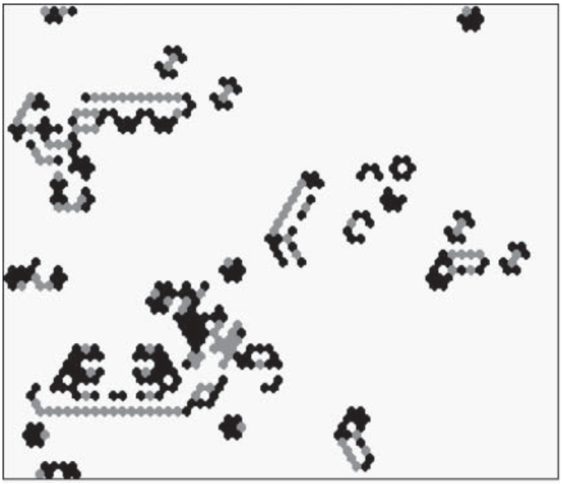


FIGURE 22
Neighborhood k = 7. kcode 020020200011120000020011022000120110. (v3k7x1.vco). The dynamics include gliders, glider collisions making static structures, and glider guns. A snapshot on a 2d (60×60) hexagonal lattice.

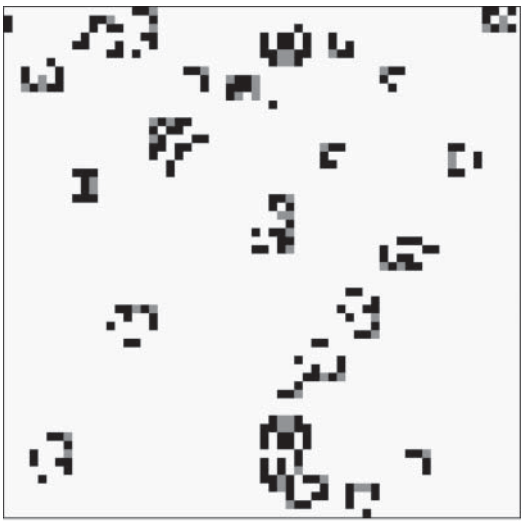


FIGURE 23
Neighborhood k = 8. kcode 001000100020002022000000002001112120011200210 (v3k8x1.vco). The dynamics include gliders moving both orthogonally and diagonally, self-reproduction and other complex structures. A snapshot on a 2d (60×60) square lattice.

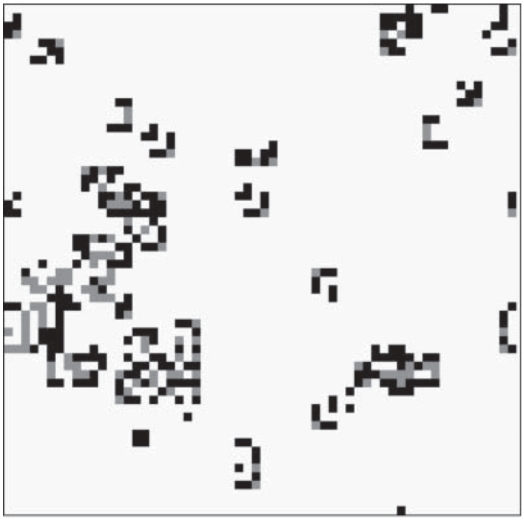


FIGURE 24
Neighborhood k = 9. kcode 2120022102202200122221121002212112022211202221222201222. (v3k9x1.vco). The dynamics include, gliders moving both orthogonally and diagonally, static structures, and other complex structures. A snapshot on a 2d (60 × 60) square lattice.

9. DISCUSSION

There is a network of complex rules in k-totalistic rule-space, connected by mutations, where large scale collective behaviors emerge spontaneously, bottom-up, as a result of local interactions. The complex dynamics includes the emergence of gliders, self-reproduction by glider collisions, polymer-like gliders, glider guns, and possibly other structures and interactions. This implies higher levels of description beyond the underlying "physics", the kcode. The levels could conceivably unfold without limit given sufficient time and space; the number of these emergent levels is our qualitative measure of complexity.

Some questions arise; how does the binary (v = 2) Game-of-Life [3] relate to v = 3 complex rules? What makes 2-dimensional complexity more abundant for v = 3 CA than for v = 2 CA? How do gliders and other complex structures emerge? What is the mechanism of self-reproduction by glider collisions? How do glider-guns self-assemble? What are the implications for 3-dimensions? Are these complex rules computationally universal? Do they relate to reaction-diffusion dynamics? How does complexity scale with greater v or k? How do the various complex rules relate to each other? Is there an underlying principle of self-organization? And what is it?

10. DISCRETE DYNAMICS LAB

The software used to research and produce this paper was multi-value DDLab [7], in which the dynamics can be seen live, and the rules are provided. It is available at www.ddlab.org.

Acknowledgments

Thanks to the Leverhulme Trust for their support and visiting Professorship at the Faculty of Computing, Engineering and Mathematical Science, University of the West of England, and thanks to the Faculty for their hospitality. A version of this paper first appeared in the Proceedings of Alife9 [8].

REFERENCES

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

[2] Bilotta, E., Lafusa, A. and Pantano, P. (2003). Is self-replication an embedded characteristic of the artificial/living matter? In Standish and Bedau, editors, *Artificial Life VIII*, 38–48. MIT Press.

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

[4] Wuensche, A. (1994). Complexity in one-d cellular automata. Santa Fe Institute working paper 94-04-025.

[5] Wuensche. A. (1999). Classifying cellular automata automatically. *COMPLEXITY, 4*(3), 47–66.

[6] Wuensche, A. (2004). www.ddlab.org; follow the link to "self-reproduction" www.cogs.susx.ac.uk/users/andywu/multi_value/self_rep.html.

[7] Wuensche, A. (2004). Discrete Dynamics Lab (DDLab), software tools for researching Cellular Automata, Random Boolean Networks and multi-value Discrete Dynamical Networks, www.ddlab.org.

[8] Wuensche, A. (2004). Self-reproduction by glider collisions: the beehive rule. In J. Pollack et al., editor, *Alife9 Proceedings*, 286–291. MIT Press.