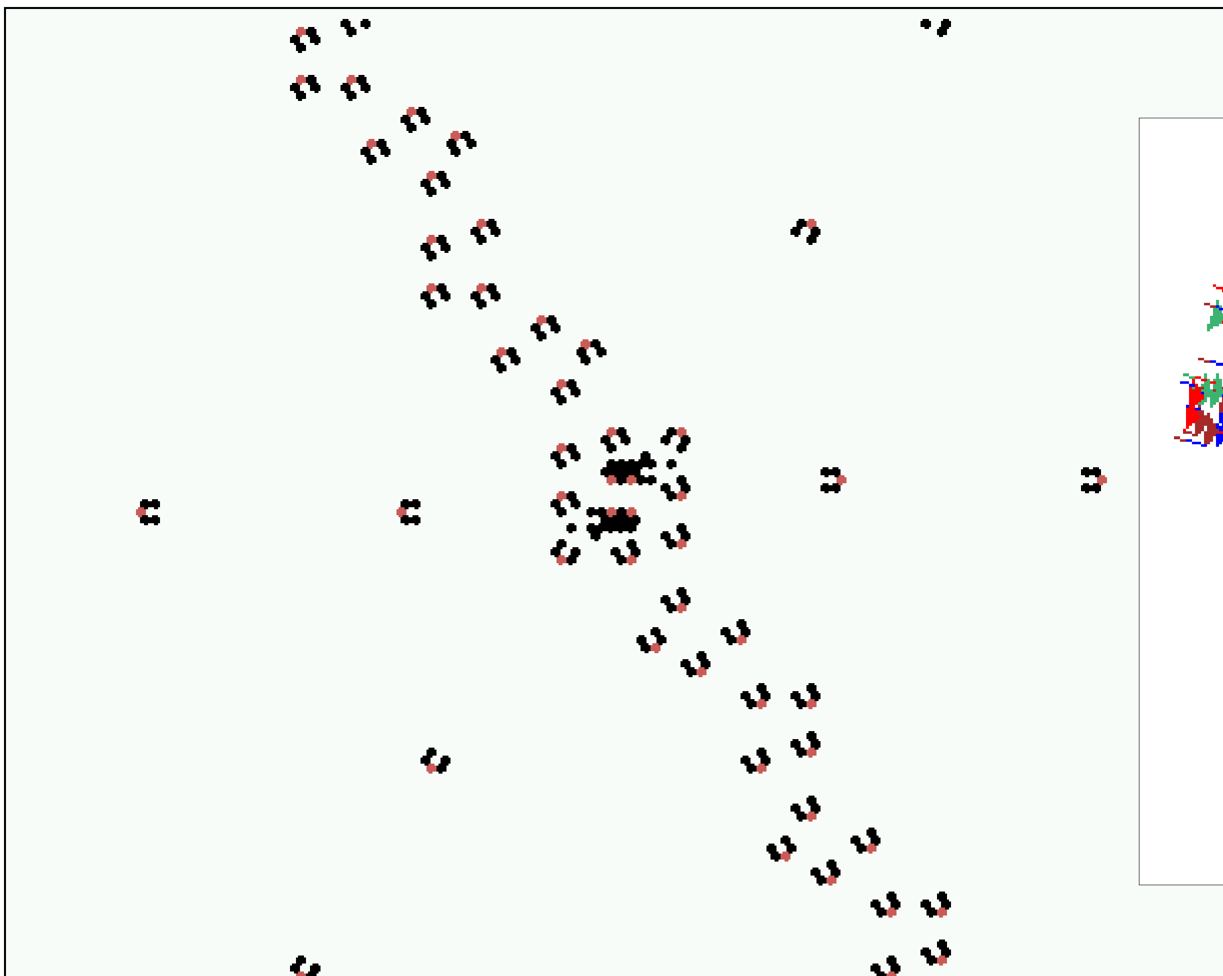
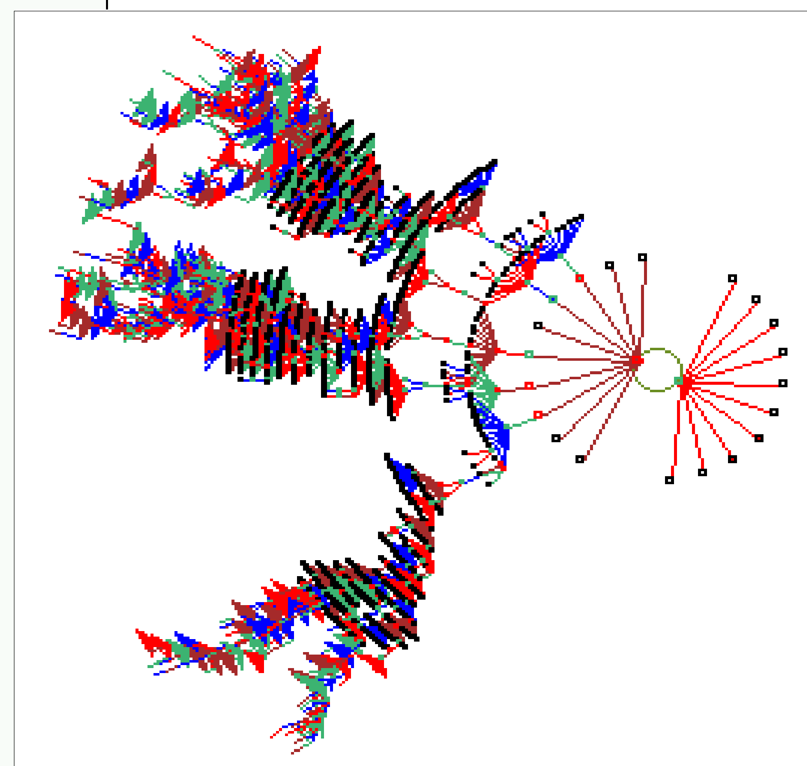
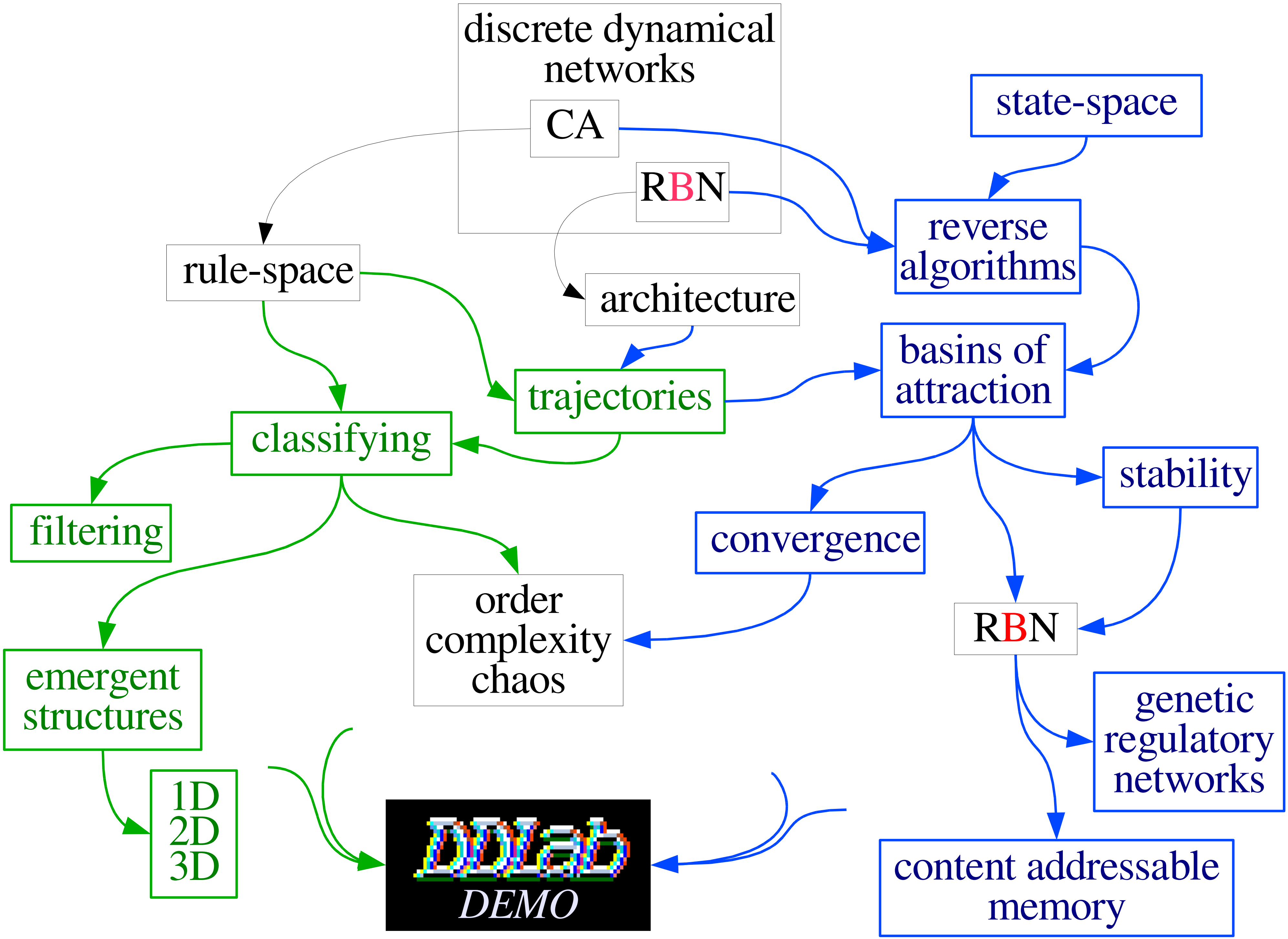
Andy Wuensche
Discrete Dynamics Lab
Visiting Fellow, Dept. of Informatics (formerly COGS) Univ. of Sussex

Complex dynamics, basins of attraction, and content addressable memory, in discrete systems





andy AT ddlab DOT org
www.ddlab.org



discrete dynamical networks
CA
RBN
state-space
reverse algorithms
rule-space
architecture
basins of attraction
trajectories
classifying
stability
filtering
convergence
order complexity chaos
RBN
emergent structures
genetic regulatory networks
1D 2D 3D
DDLab
DEMO
content addressable memory

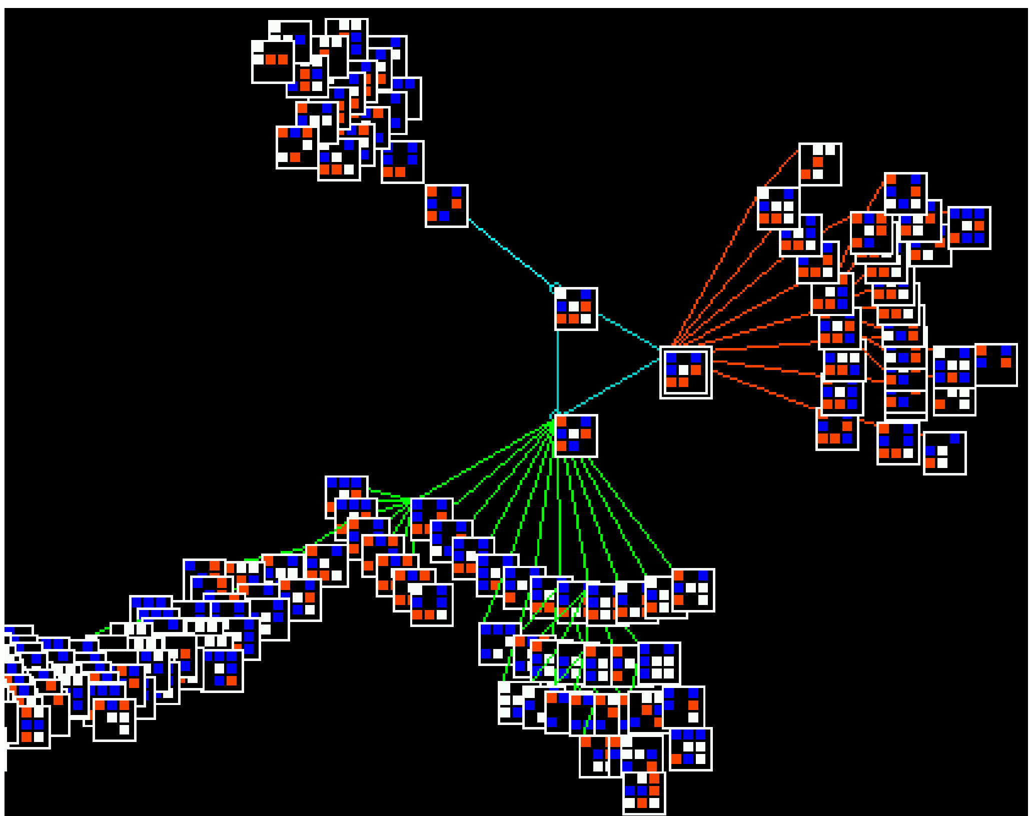
Discrete Dynamics Lab

Tools for researching Cellular Automata, Random Boolean Networks,
multi-value Discrete Dynamical Networks, and beyond

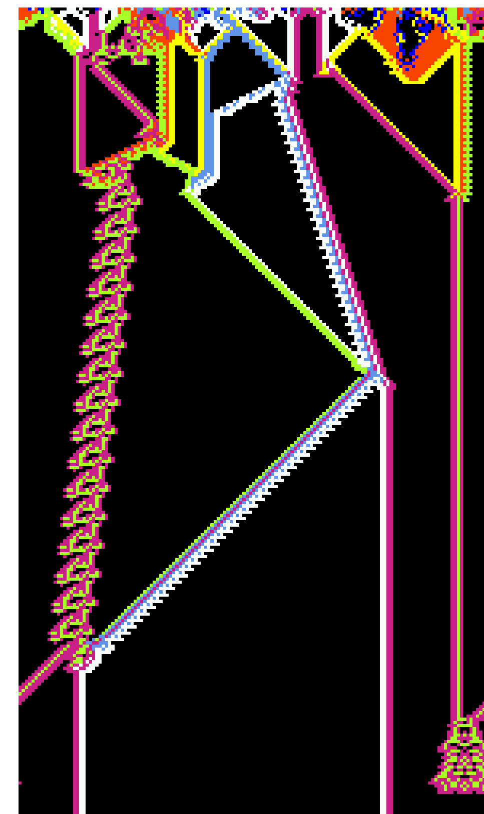
www . ddlab . org

Language:
plain C

Platforms:
Linux
Unix
Irix
Mac
Cygwin
Dos



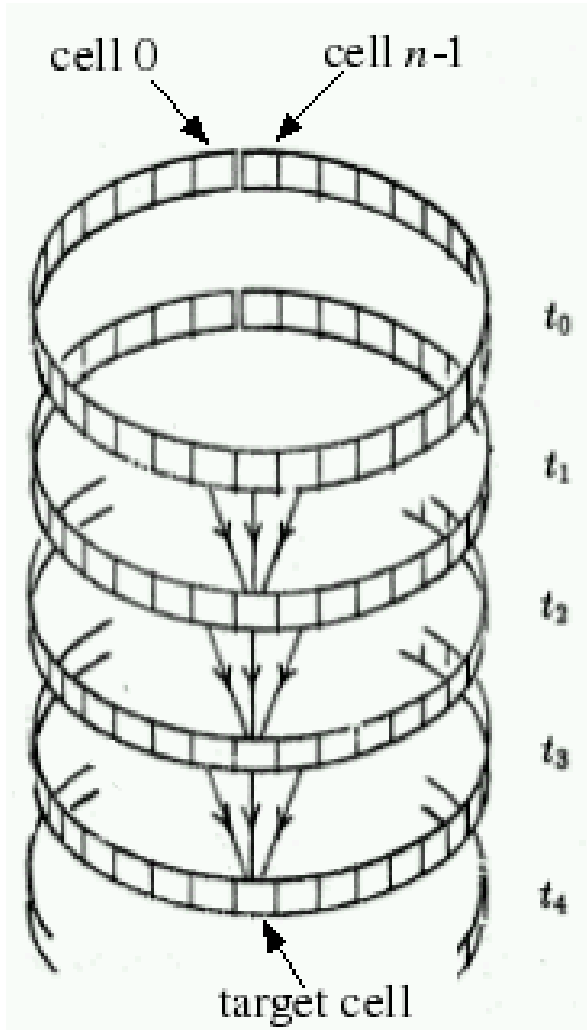
basins of attraction



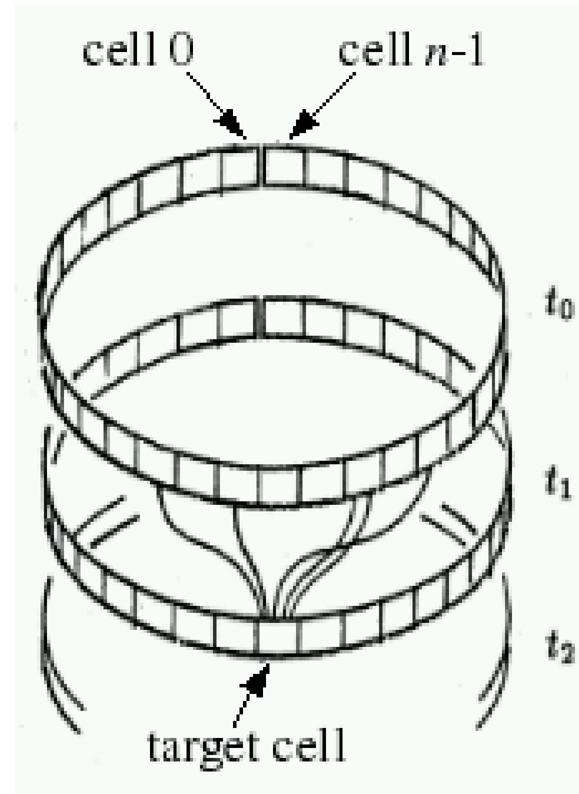
space-time patterns

1D cellular automata (CA) - random Boolean networks (RBN)

CA
"an artificial universe with a local physics"
Chris Langton



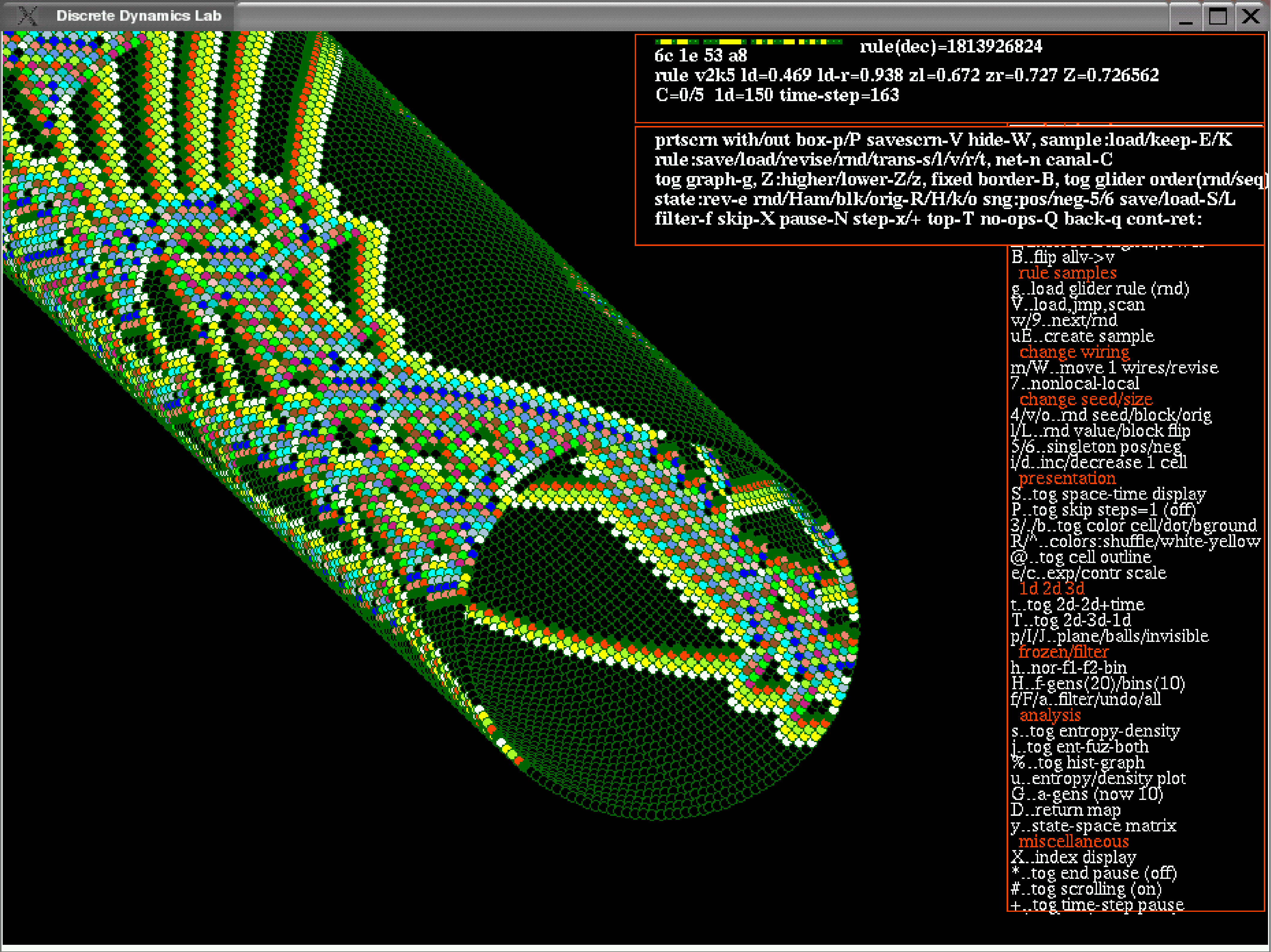
time steps



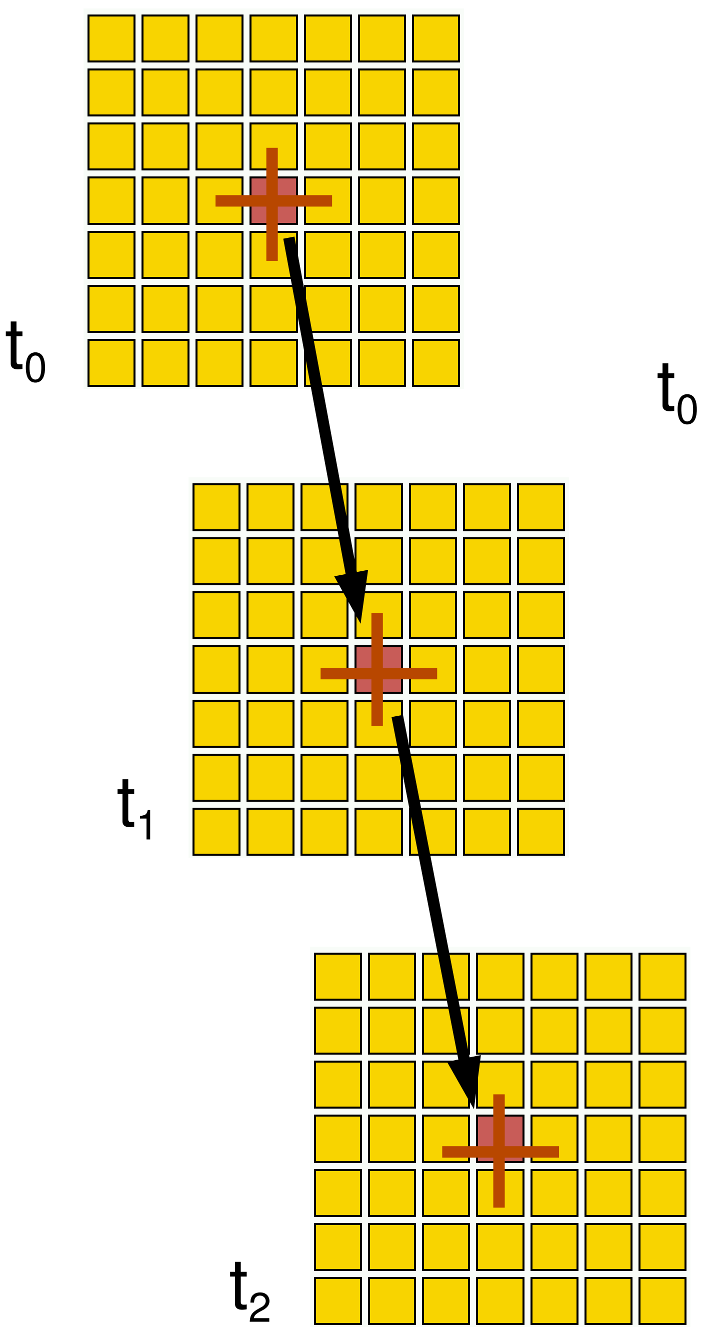
RBN
models in biology
Stuart Kauffman

111	110	101	100	011	010	001	000	....neighbourhoods k=3
0	1	1	0	1	1	1	0	....outputs (rule 110), rule-space=256

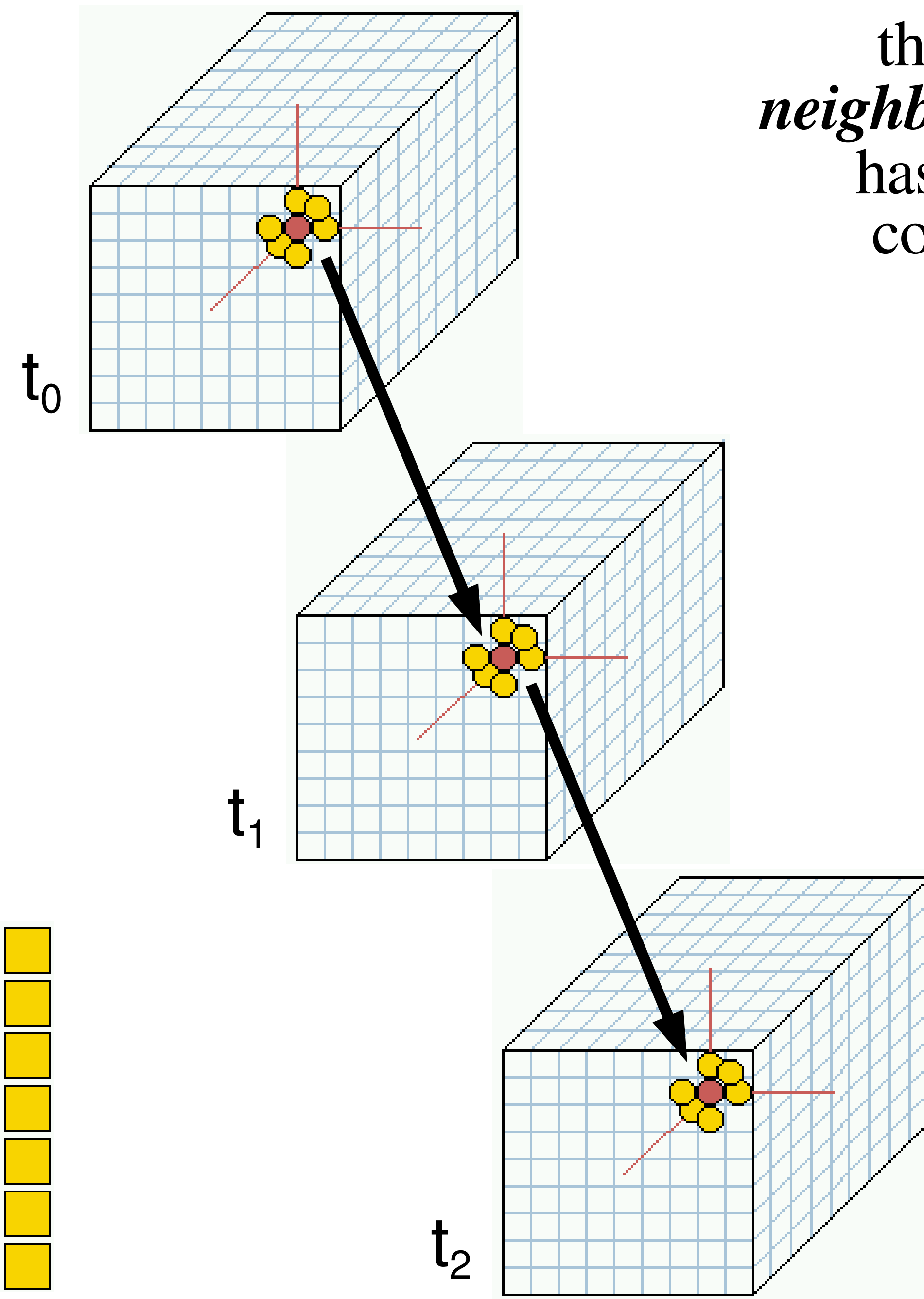
rule-space = $2\hat{\ }2\hat{\ }k$ multi-value v: rule-space=$v\hat{\ }v\hat{\ }k$
as $\{v,k\}$ increase, this becomes very big!


Discrete Dynamics Lab
6c 1e 53 a8
rule(dec)=1813926824
rule v2k5 ld=0.469 ld-r=0.938 zl=0.672 zr=0.727 Z=0.726562
C=0/5 1d=150 time-step=163
prtscrn with/out box-p/P savescrn-V hide-W, sample:load/keep-E/K
rule:save/load/revise/rnd/trans-s/l/v/r/t, net-n canal-C
tog graph-g, Z:higher/lower-Z/z, fixed border-B, tog glider order(rnd/seq)
state:rev-e rnd/Ham/blk/orig-R/H/k/o sng:pos/neg-5/6 save/load-S/L
filter-f skip-X pause-N step-x/+ top-T no-ops-Q back-q cont-ret:
B..flip allv->v
rule samples
g..load glider rule (rnd)
V..load,jmp,scan
w/9..next/rnd
uE..create sample
change wiring
m/W..move 1 wires/revise
7..nonlocal-local
change seed/size
4/v/o..rnd seed/block/orig
l/L..rnd value/block flip
5/6..singleton pos/neg
i/d..inc/decrease 1 cell
presentation
S..tog space-time display
P..tog skip steps=1 (off)
3/./b..tog color cell/dot/bground
R/^..colors:shuffle/white-yellow
@..tog cell outline
e/c..exp/contr scale
1d 2d 3d
t..tog 2d-2d+time
T..tog 2d-3d-1d
p/I/J..plane/balls/invisible
frozen/filter
h..nor-f1-f2-bin
H..f-gens(20)/bins(10)
f/F/a..filter/undo/all
analysis
s..tog entropy-density
j..tog ent-fuz-both
%..tog hist-graph
u..entropy/density plot
G..a-gens (now 10)
D..return map
y..state-space matrix
miscellaneous
X..index display
*..tog end pause (off)
#..tog scrolling (on)
+..tog time-step pause

2D CA

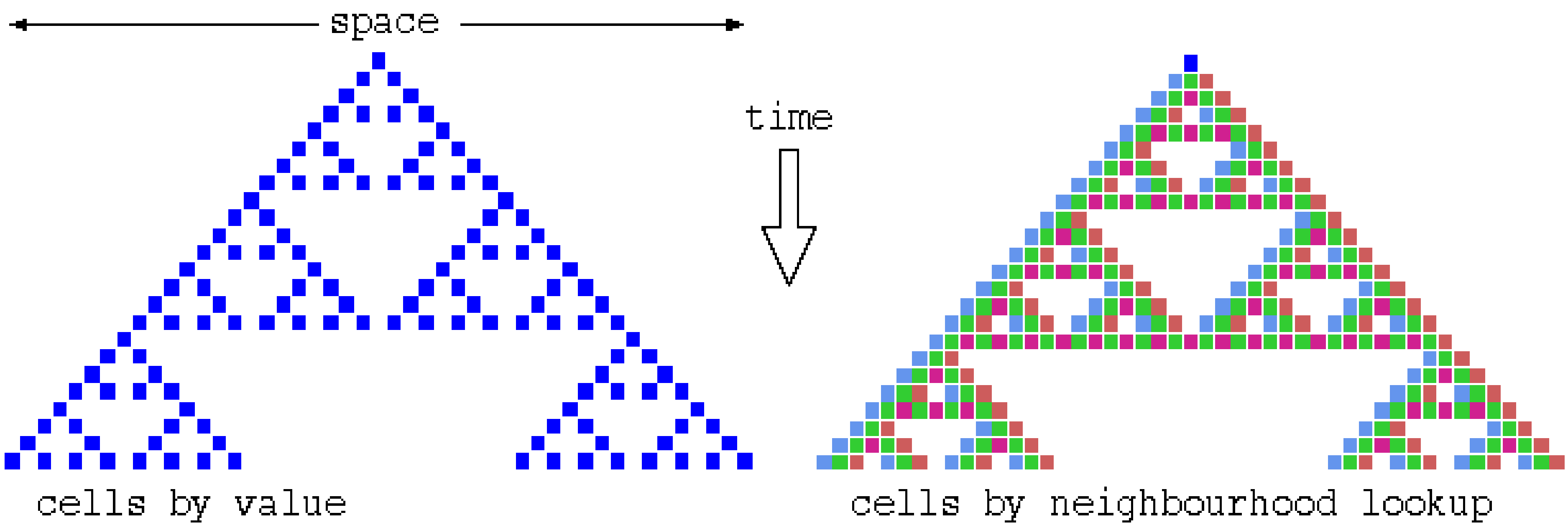


3D CA



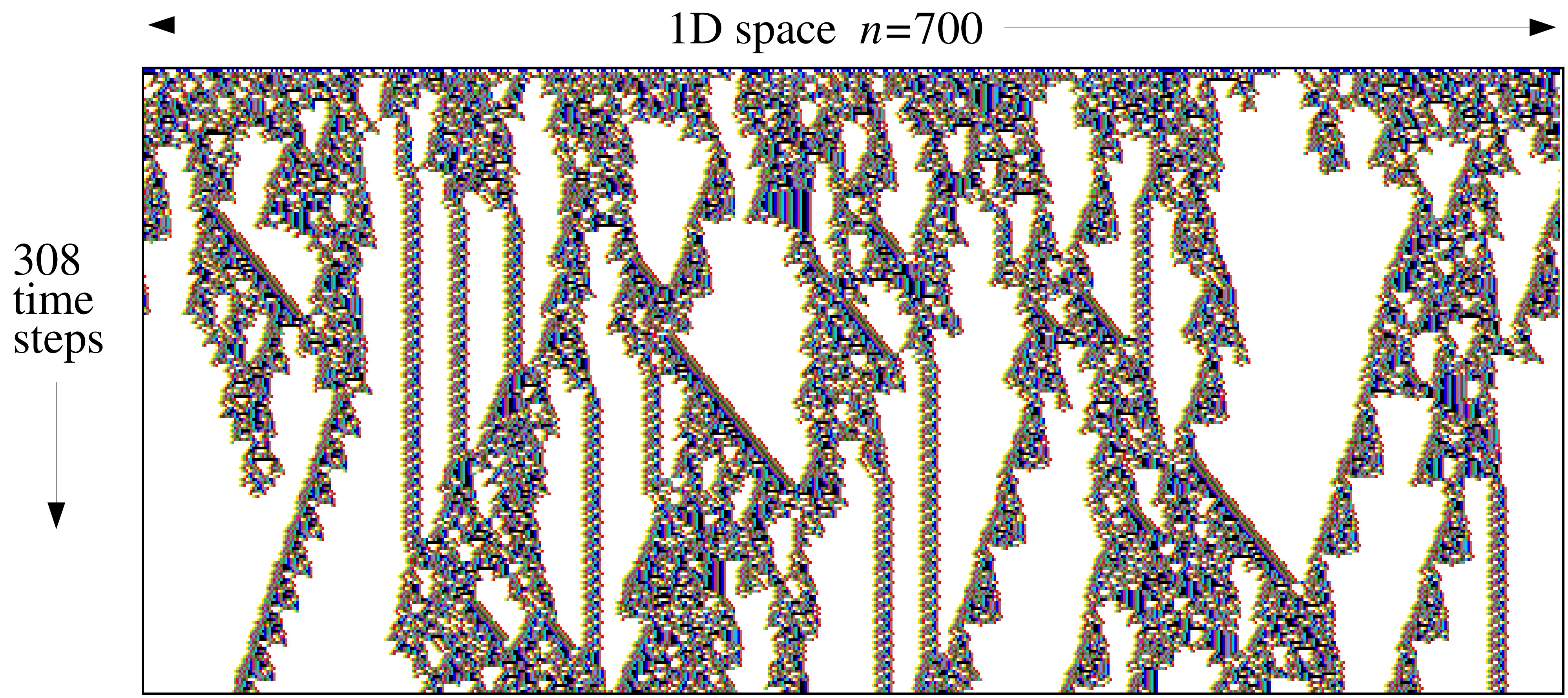
RBN:
the ***pseudo-neighbourhood***
has arbitrary
connections

1D space-time patterns – alternative presentations



Space-time patterns of a 1d 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 according to their look-up neighbourhood.

The space-time pattern of a 1d complex CA with interacting gliders.

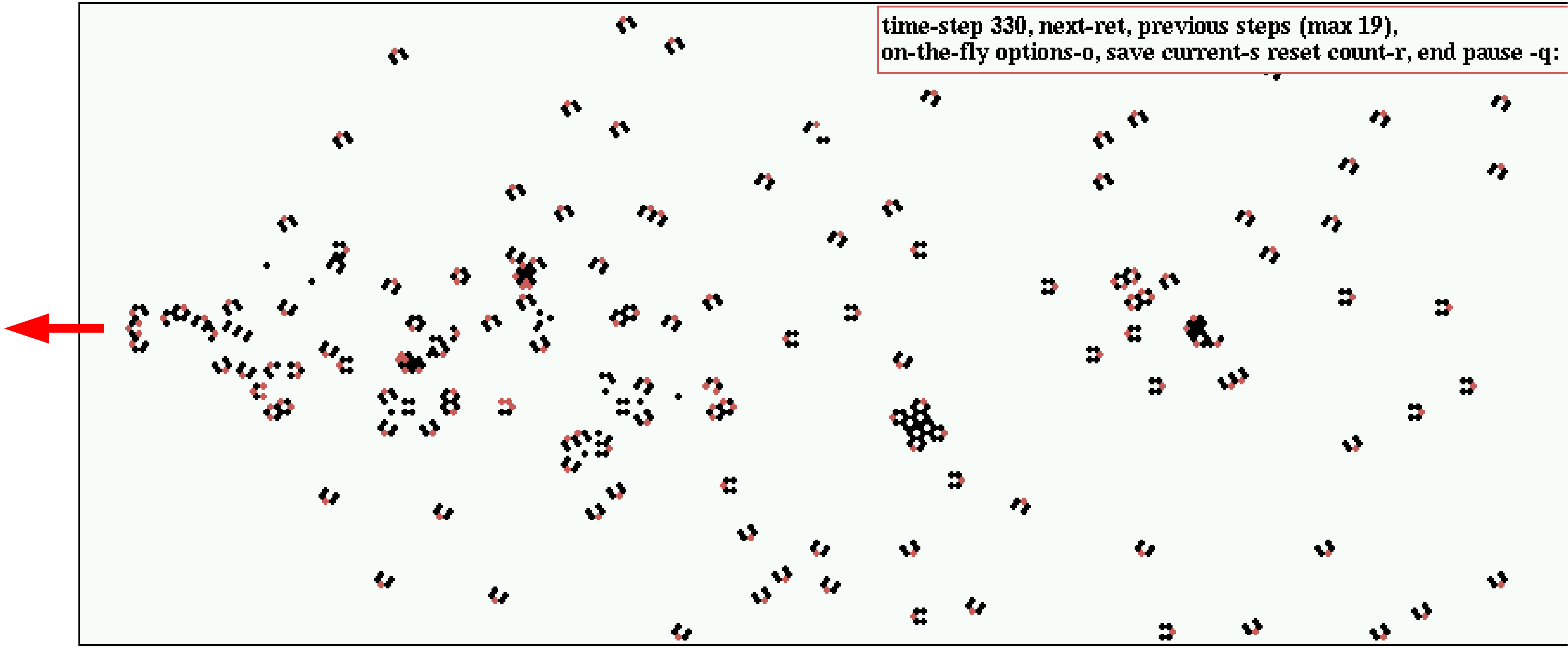


308 time-steps from a random initial state. System size n=700, Neighbourhood size k=7, rule (hex) = 3b 46 9c 0e e4 f7 fa 96 f9 3b 4d 32 b0 9e d0 e0. Cells are colored according to neighbourhood look-up instead of the value. Space is across and time down the page.

the future is determined but unpredictable! (*Wolfram*)

Snapshot of a 3-value complex rule on a hex lattice

a puffer train in the Beehive rule



4 RBN space-time patterns

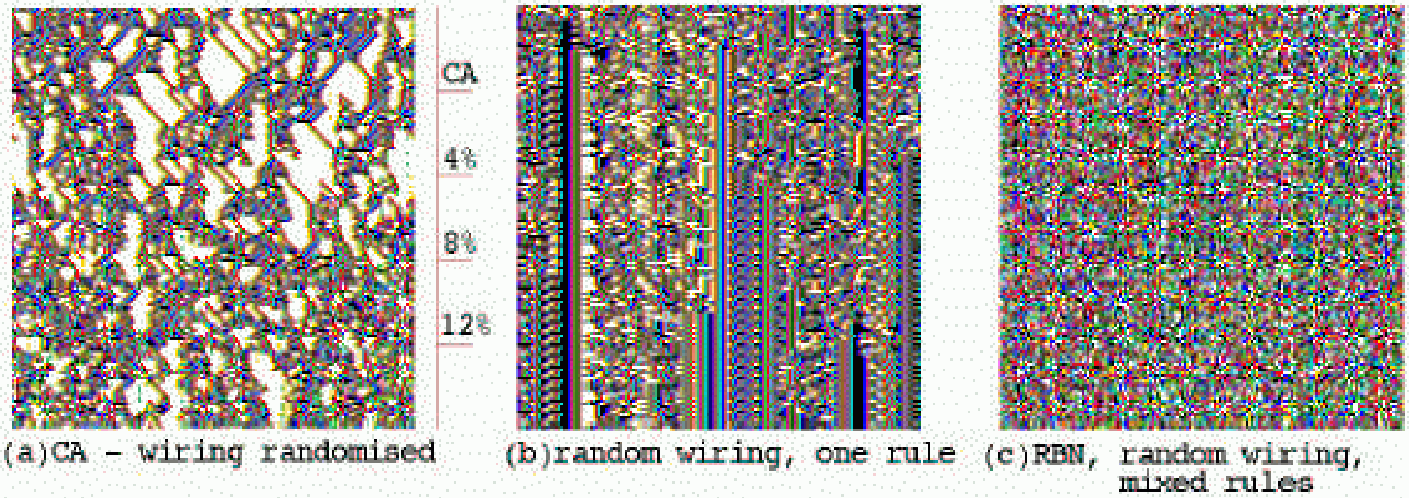
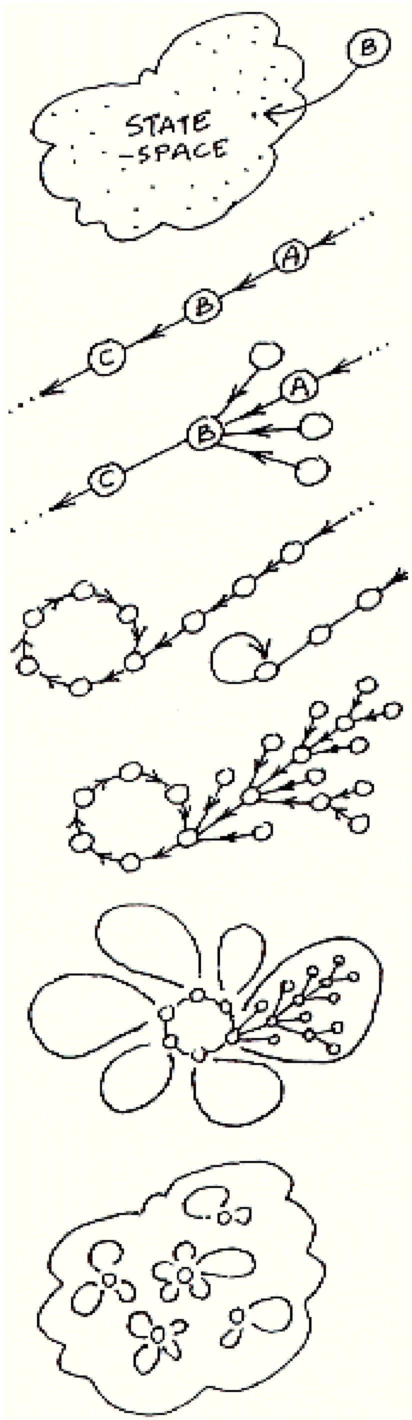


Figure 11: Space-time patterns for intermediate 1d architecture, from CA to RBN. n=150, k=5, 150 time-steps from a random initial state. (a) Starting off as a complex CA (rule 6c1e53a8 as in figure 8), 4% (30/750) of available wires are randomized at 30 time-step intervals. The coherent pattern is progressively degraded. (b) A network with local wiring but mixed rules, vertical features are evident. (c) RBN, random wiring and mixed rules, with no bias, shows maximal chaotic dynamics.

Global dynamics: the idea



for a network size n, a state ***B*** might be: 1010...0110
there are 2^n states in ***state-space*** (v^n for value-range v)

a ***trajectory***: → ***A*** → ***B*** → ***C*** →

B may have other ***pre-images*** besides ***A***, which can be directly computed by reverse-algorithms. States with zero pre-images (leaves) - are known as ***garden-of-Eden*** states

the trajectory must arrive at an ***attractor***, a cycle of states with a period of one or more

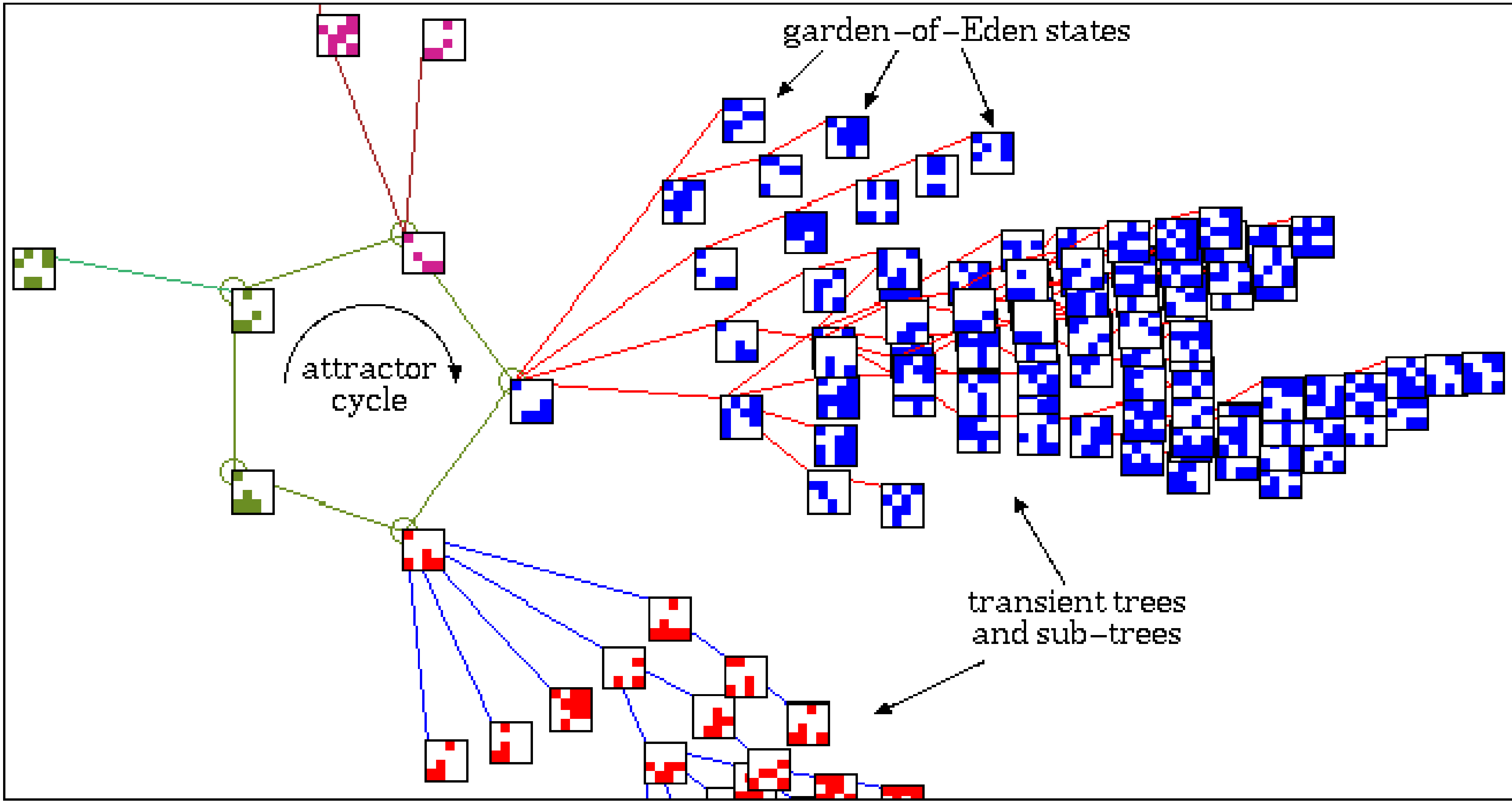
find the pre-images of an attractors state (excluding the one on the attractor) - then pre-images of pre-images, until all g-of-E states have been reached – the graph of linked states is a ***transient tree***

construct each transient tree (if any) from each attractor state - the complete graph is the ***basin of attraction***

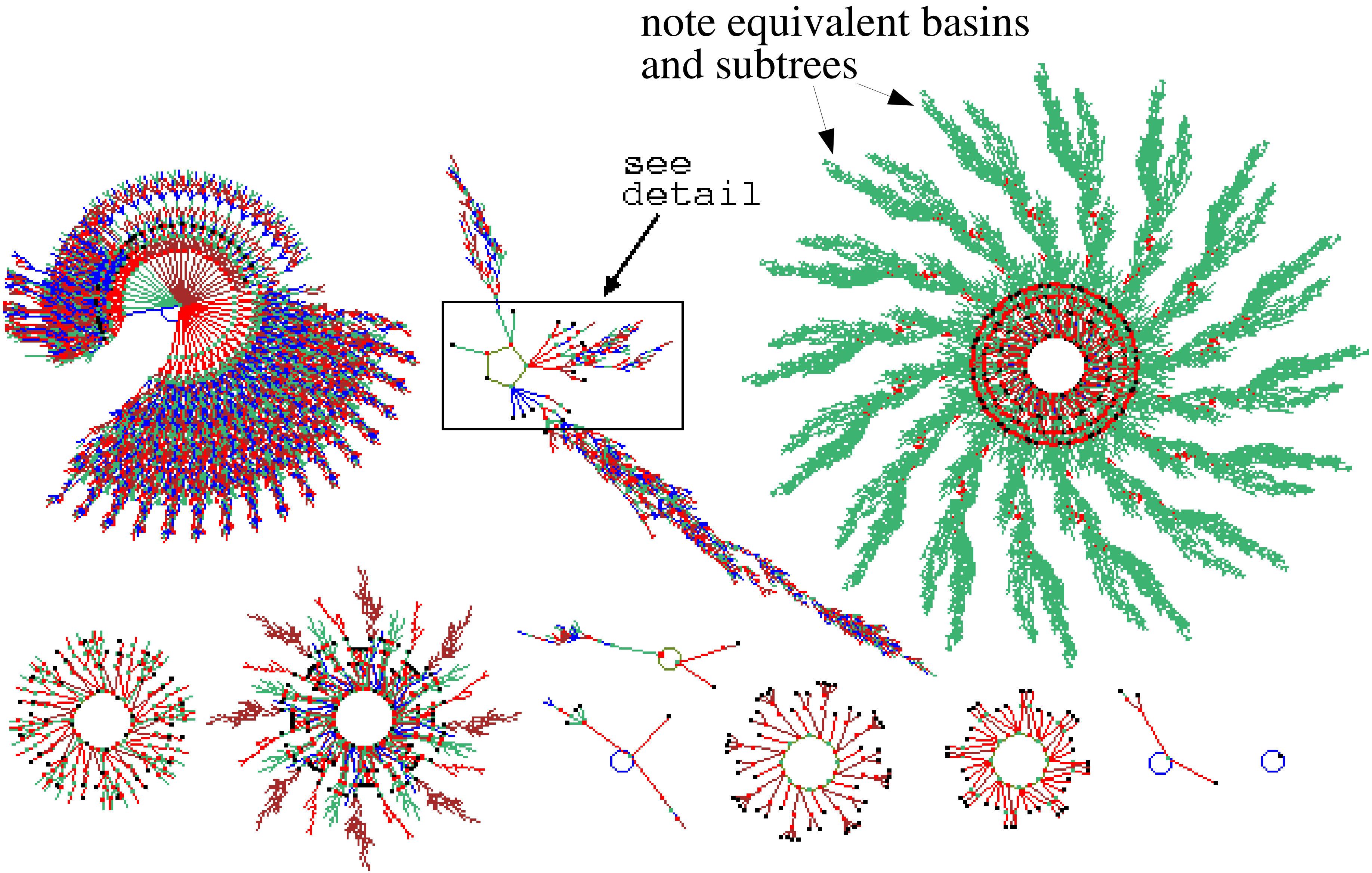
find every attractor and construct its basin of attraction – this is the ***basin of attraction field*** - all states in state-space linked by the dynamics – each discrete dynamical network imposes a specific basin of attraction field on state-space

A detail of a basin of attraction

states shown as 4x4 bit patterns



The basin of attraction field of a CA, n=16



rule (hex) 3b 46 9c 0e e4 f7 fa 96 f9 3b 4d 32 b0 9e d0 e0 (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.

Constraints on 1D dynamics

Rotational symmetry: can only increase in a transient; stay constant in the attractor

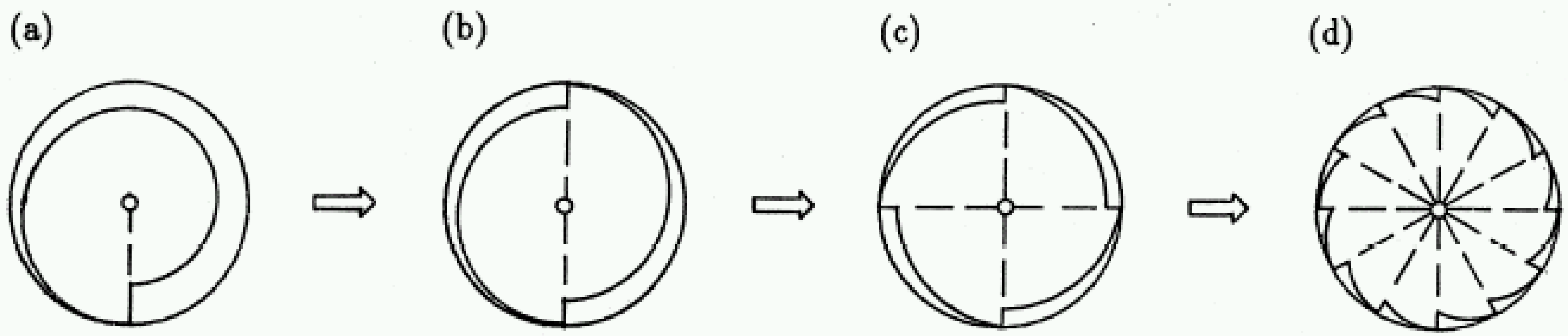


FIGURE 3.3 Possible transient evolution of a disordered state to states with an increasing degree of rotation symmetry. (a) $s = 1$, $g = L$. (b) $s = 2$, $g = L/2$. (c) $s = 4$, $g = L/4$. (d) $s = 12$, $g = L/12$.

Bilateral symmetry: can only increase in a transient; stay constant in the attractor

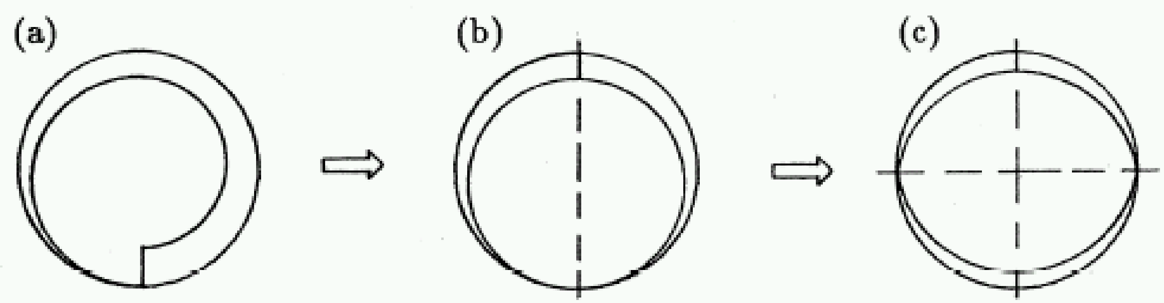
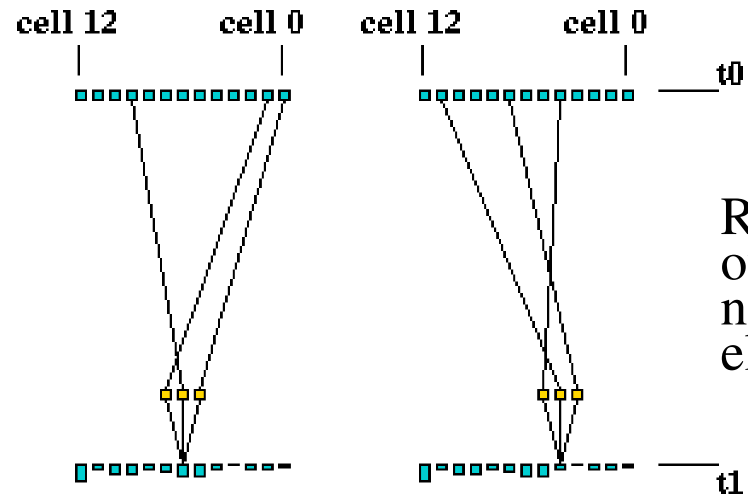


FIGURE 3.4 Possible transient evolution of a disordered state to a state with bilateral symmetry, bs (rotation symmetry, s). (a) $bs = 0$, $s = 0$. (b) $bs = 2$, $s = 0$. (c) $bs = 4$, $s = 2$.

The RBN wiring/rule scheme defined

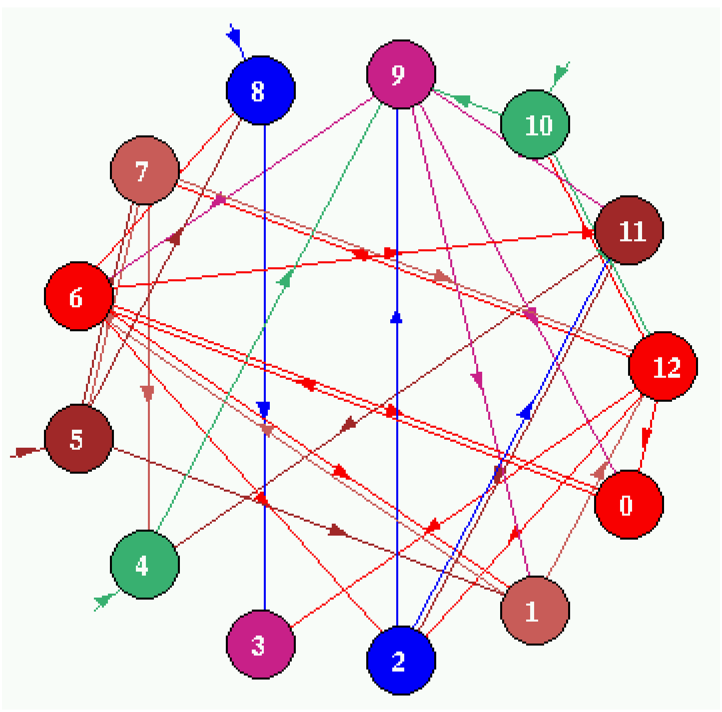
RBN wiring/rule matrix

		2.	1.	0.	rule(hex)
7	12..	10	1	7	56
2	11..	6	2	9	04
4	10..	10	10	12	c4
4	9..	2	10	4	34
2	8..	5	6	8	ea
3	7..	12	5	12	64
5	6..	1	9	0	06
5	5..	5	7	5	64
2	4..	4	11	7	06
0	3..	8	12	12	5e
2	2..	11	6	12	4a
2	1..	6	5	9	d6
1	0..	12	9	6	bc

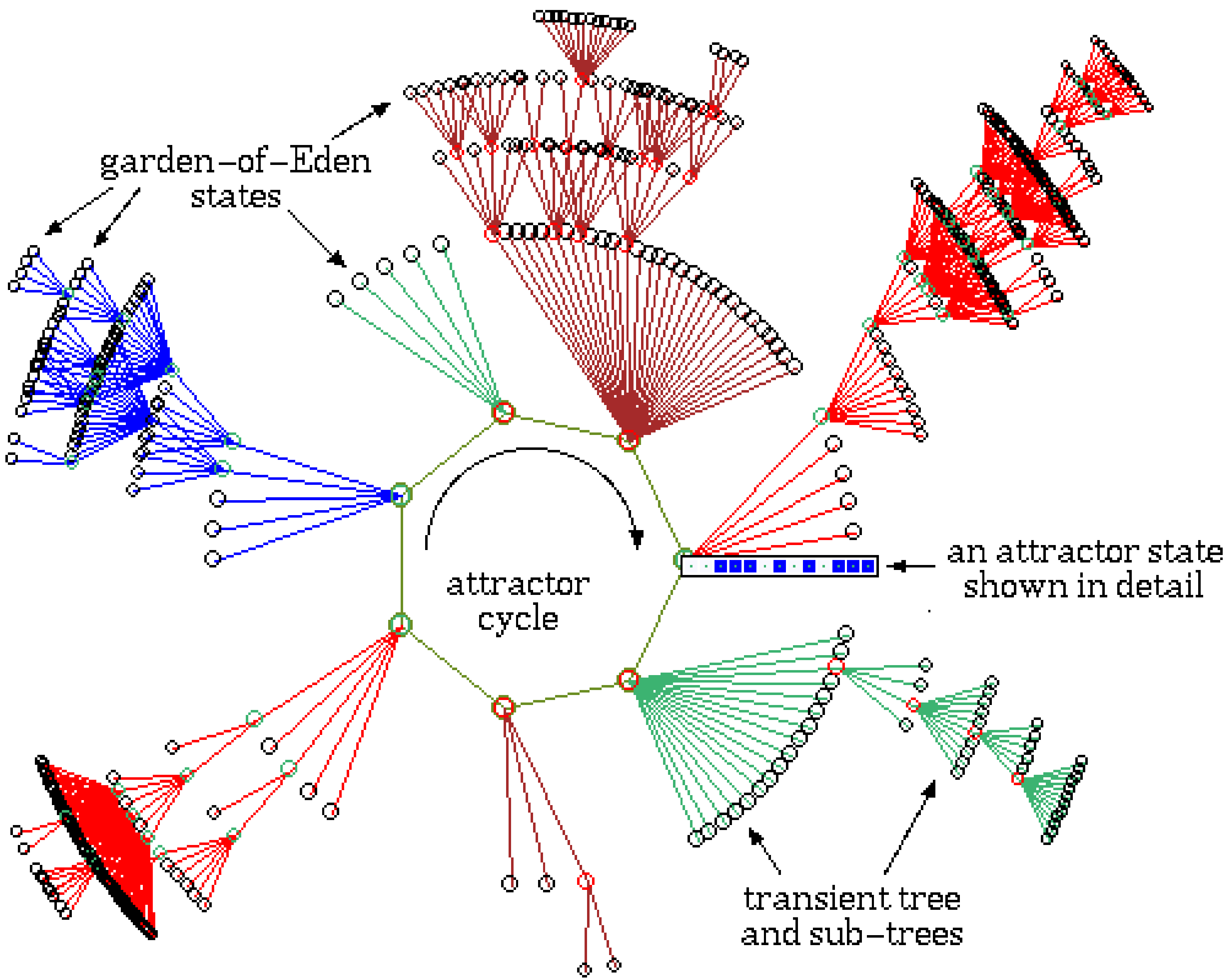


RBN picking out one network element

RBN wiring graph

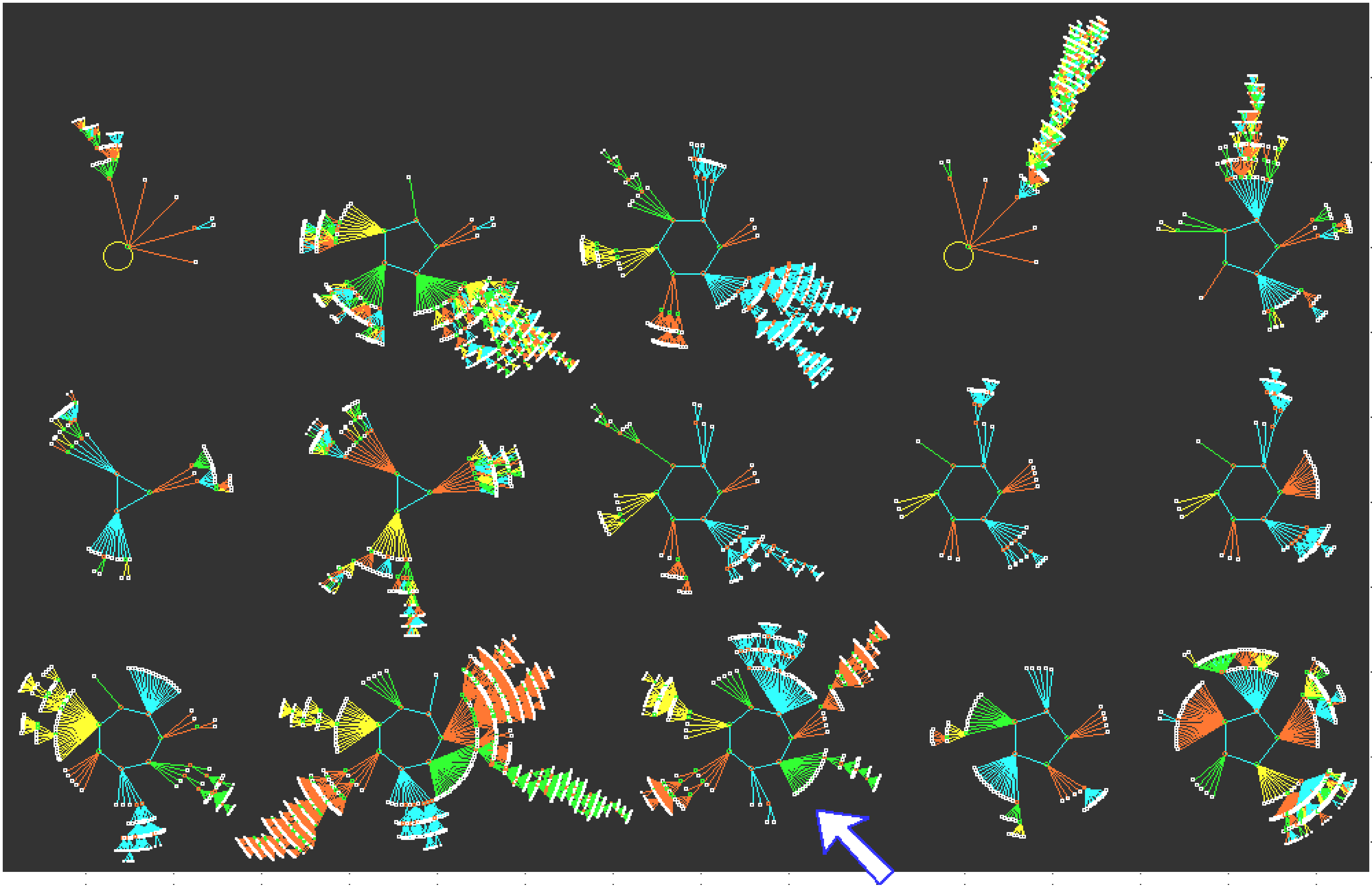


A single RBN basin of attraction



One of the basins of attraction of the random Boolean network, with 604 states of which 523 are garden of Eden states. The direction of time in inwards from garden of Eden states, then clockwise.

The basin of attraction field of a RBN, *n*=13



The 213=8192 states in state space are organized into 15 basins, varying in volume from 68 to 2724 *n*=13, *k*=3.

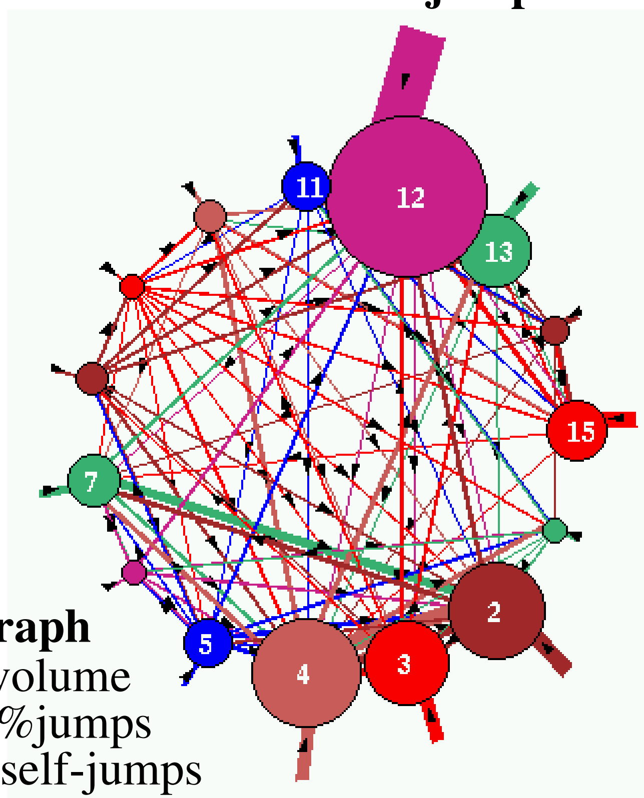
Jumping between basins due to 1-bit perturbations to attractor states

stability: a strong diagonal in the jump table, or if %self-jumps > %basin-volume
example: basin 2: basin-volume=40%, self-jumps=12%

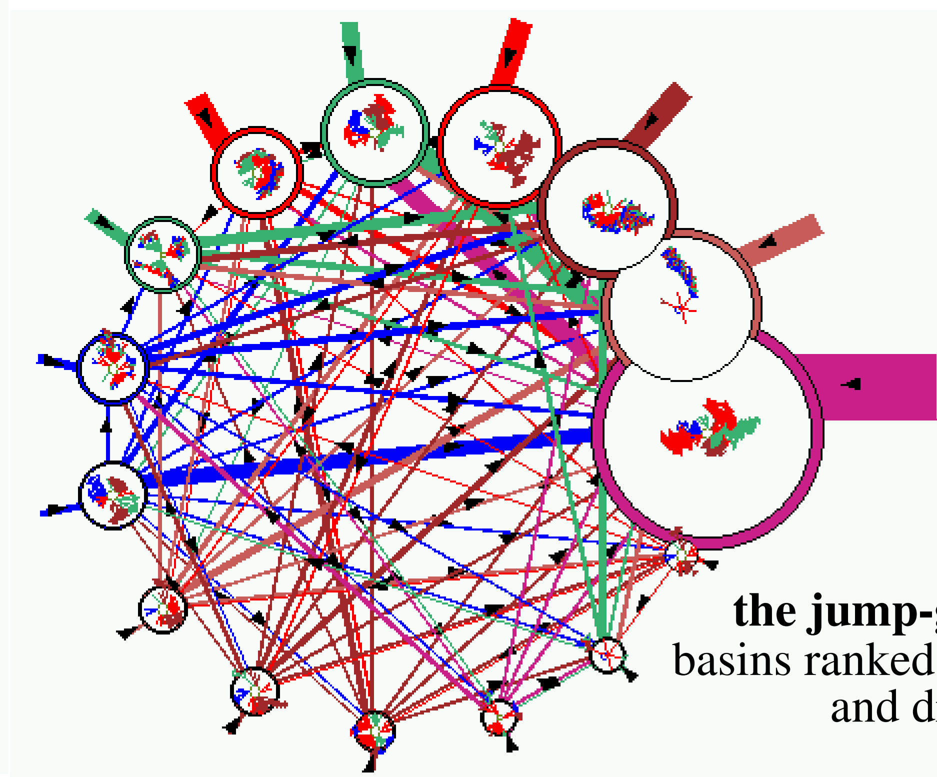
a strong diagonal indicates stability

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	P	J	Volume	Self
1:	4	4	2	1	.	1	.	.	.	.	1	.	.	.	.	1	13	68= 0.83%	30.77%
2:	.	26	8	12	5	.	9	.	.	.	.	5	.	.	.	5	65	984= 12.01%	40.00%
3:	.	14	31	17	.	.	.	6	.	4	.	3	3	.	.	6	78	784= 9.57%	39.74%
4:	1	4	2	4	.	.	1	.	.	.	.	.	1	.	.	1	13	1300= 15.87%	30.77%
5:	3	12	5	9	19	5	4	3	.	.	.	5	.	.	.	5	65	264= 3.22%	29.23%
6:	3	3	.	2	9	13	6	.	.	.	.	3	.	.	.	3	39	76= 0.93%	33.33%
7:	.	12	.	5	.	3	16	.	.	.	.	3	.	.	.	3	39	316= 3.86%	41.03%
8:	.	3	15	17	11	.	.	22	4	.	2	3	1	.	.	6	78	120= 1.46%	28.21%
9:	3	5	7	23	3	.	2	4	18	7	.	4	.	1	1	6	78	64= 0.78%	23.08%
10:	.	8	11	26	.	.	2	.	6	19	.	4	.	.	2	6	78	120= 1.46%	24.36%
11:	1	1	.	2	2	.	.	.	1	.	23	43	15	1	2	7	91	256= 3.12%	25.27%
12:	.	2	3	1	.	.	1	.	.	.	.	63	18	.	3	7	91	2724= 33.25%	69.23%
13:	.	3	.	3	.	.	.	.	.	1	7	43	31	.	3	7	91	604= 7.37%	34.07%
14:	1	.	.	2	.	.	1	.	1	.	4	10	1	28	17	5	65	84= 1.03%	43.08%
15:	.	.	.	3	.	.	1	.	.	1	.	10	5	5	40	5	65	428= 5.22%	61.54%

the jump table, counting basin jumps



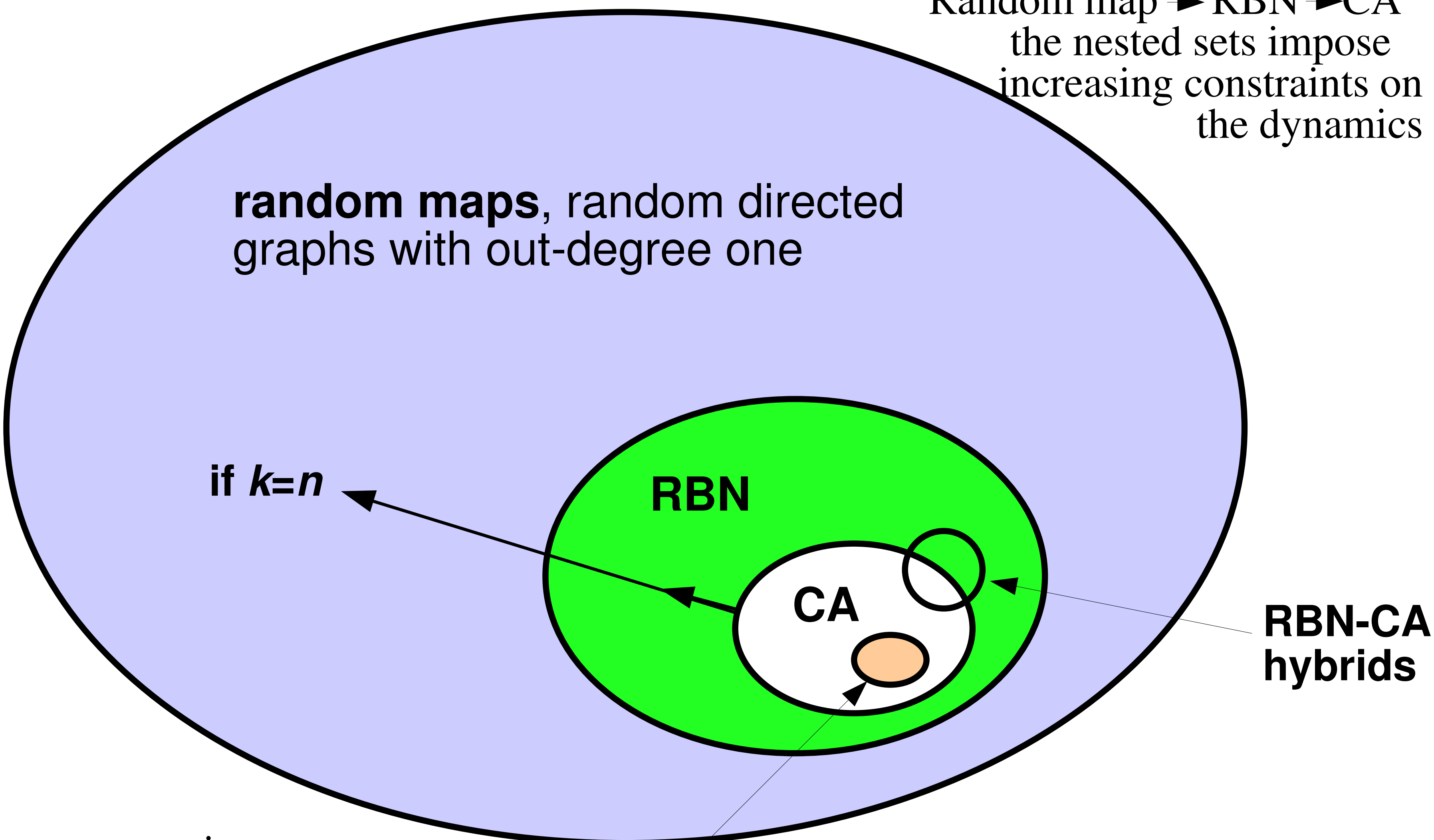
the jump-graph
size=basin-volume
link-width=%jumps
short stubs: self-jumps



the jump-graph with basins ranked by volume and drawn inside

Global dynamics in the context of graph theory

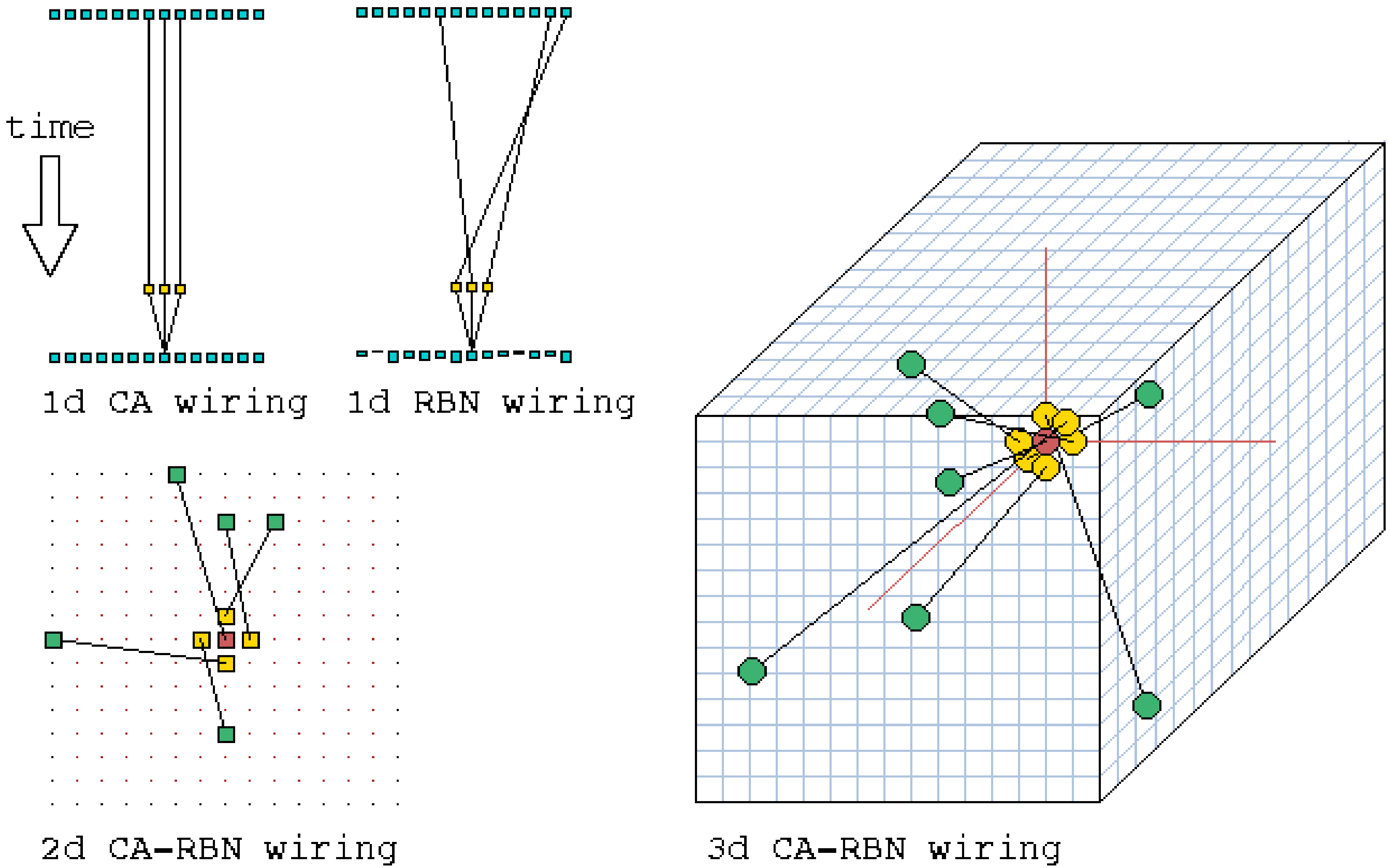
Random map ► RBN ► CA
the nested sets impose
increasing constraints on
the dynamics



A random map assigns a successor (possibly at random) to each state in state-space. Random maps also fall into basins of attraction (computed in DDLab)

RBN and CA are usually sparsely connected, $k << n$. If fully connected they are the same as random maps

Visualizing and amending network wiring



For RBN, a cell's *k* inputs, may come from cells anywhere in the network; these cells are wired to a “pseudo-neighbourhood” to which a CA rule is applied.

Scale-free RBN, n=100

fully connected   modular, n=5x20   detail, n=20 module

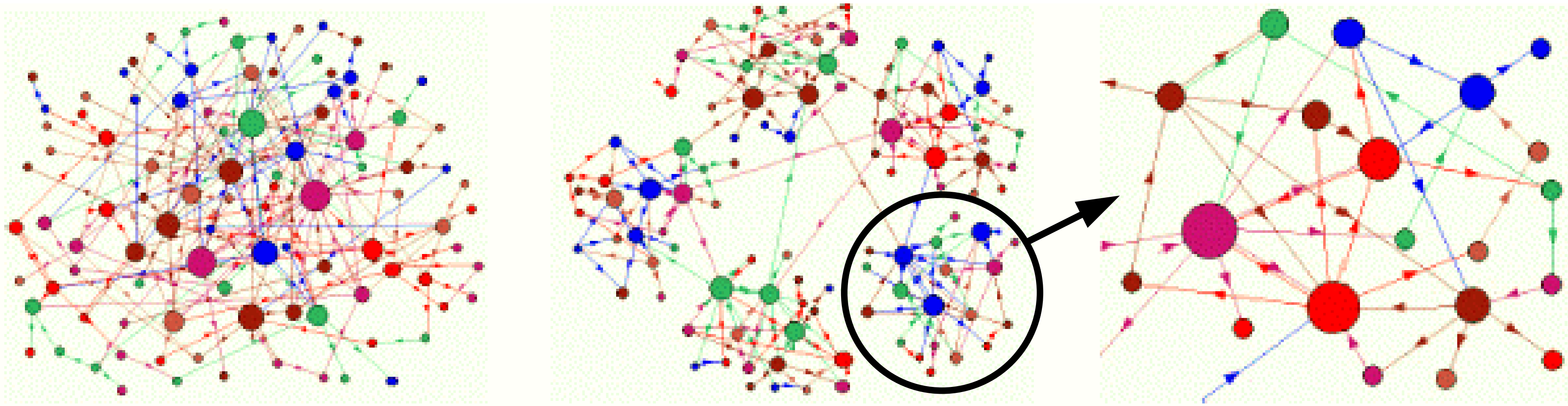


Figure 1: Hypothetical networks of interacting elements (size n=100) with an approximate power-law distribution of connections, both inputs (k) and outputs, which are represented by directed links (with arrows). Nodes are scaled according to k and average $k \simeq 2.2$. *left*: A fully connected network. *center*: A network made up of five weakly inter-linked n=20 sub-networks or modules. *right*: A detail of the top right sub-network. These are examples of random Boolean networks defined in section 4.

Figure 2: Histograms of link frequency (y axis) against link size (x axis), for inputs+outputs, in the networks in figure 1. The fully connected network (*left*), and modular network (*right*), have a similar link frequency profile. However their dynamics are very different, as described in section 8.

link-size frequency profile is similar - but dynamics is different

Basin of attraction field (scale-free) RBN, n=20

state-space = 1.05 million

61.8% **28.6%** **9.6%**

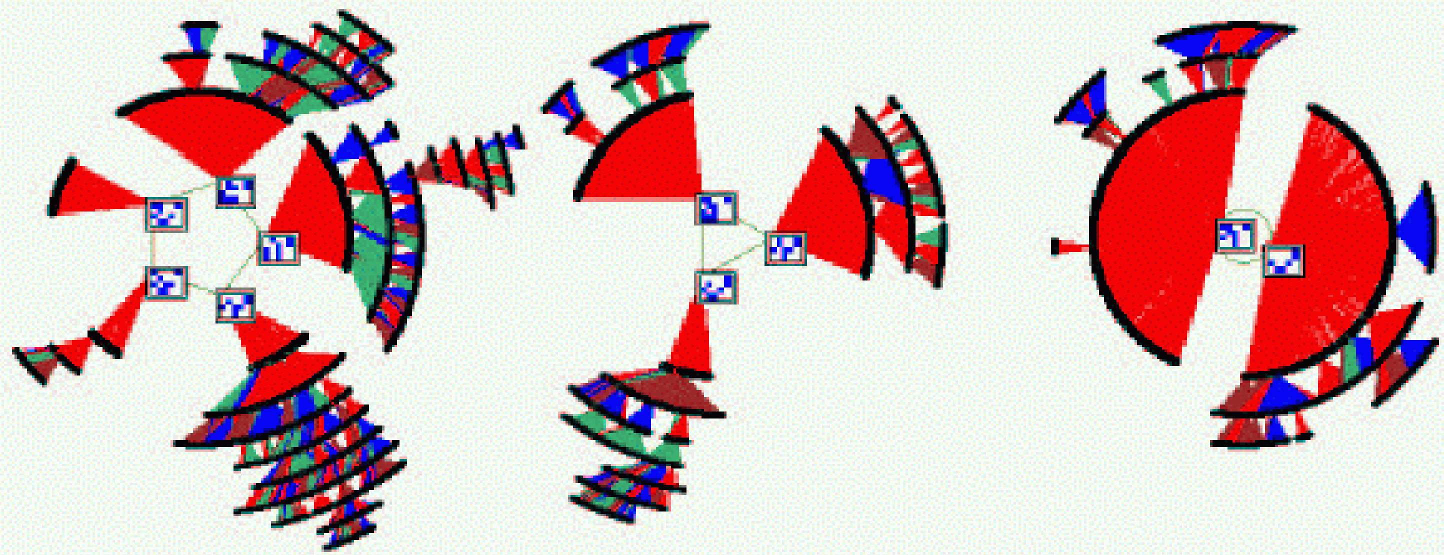
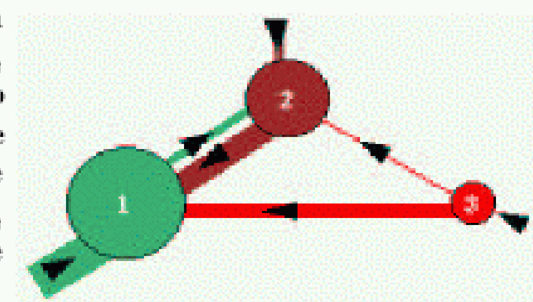


Figure 4: The basin of attraction field of the n=20 sub-network shown in detail in figure 1 (*right*). The rules (input logic) were assigned at random. State-space (size $2^{20} \simeq 1.05$ million) is partitioned into three basins of attraction. The attractor states are shown as 5x4 bit patterns. The table and diagram on the right show the probability of jumping between basins due to one-bit perturbations of their attractor states. P = attractor period, J = possible jumps ($P \times n$), and V% is the basin "volume" as a percentage of state-space. For example in basin 1, P=5, J=100 possible jumps, 15 of these jump to basin 2, and 85 back to itself, so basin 1 is relatively stable. Basin 3 has relatively few jumps back to itself so in unstable, it is also unreachable from the other basins. The diagram below the table, the "metagraph" (see section 5), shows the same data graphically. Node size reflects basin volume, link thickness percentage jumps, arrows the direction, and the short stubs self-jumps. The fraction of garden-of-Eden states in all three basins is 0.999+ indicating high convergence and order.

	1	2	3	P	J	V%
1:	85	15	.	5	100	61.78
2:	48	12	.	3	60	28.57
3:	32	6	2	2	40	9.65



the jump graph

Attractor frequency in fully-connected, and modular, (scale-free) RBN, n=100

These attractors are found by a statistical method, by running forward from many initial states looking for state repeats to identify attractors. The frequency of finding a given attractor indicates the size of its basin.

fully connected   modular

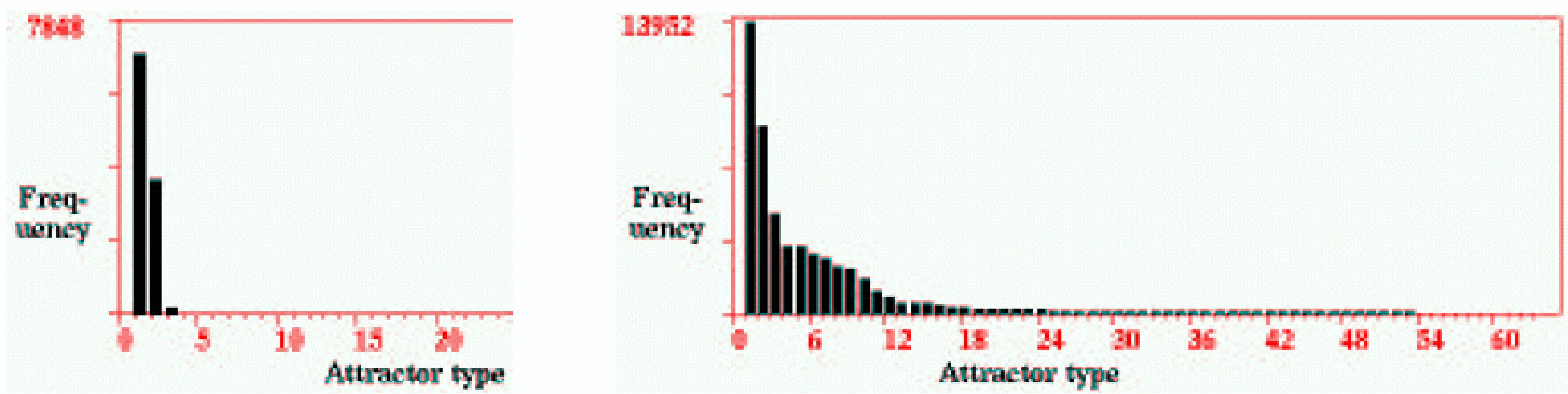


Figure 16: Attractor frequency histograms, showing the frequency (y axis, scale indicated top left) of falling into different attractors (x axis, sorted by basin volume) from a large sample of random initial states. These examples are for the "scale-free" networks in figure 1, where n=100. The frequency of each attractor is a statistical measure of basin volume, the fraction of state-space occupied by each basin. *left*: The fully connected network: the number of attractor types stabilized at just 3 after 10000+ runs. Their periods are 2046, 553, 380, with average transient length 605, 673, 97. *right*: The modular network: the number of attractor types stabilized at 53 after 50000+ runs, though about 2/3 of these represent very small basins. The 3 most frequent basins have periods 30, 14, 2, with average transient length 54, 46, 47.

The attractor frequency histogram and data shows that the modular network has more basins with smaller attractor periods and shorter transients, than the fully the connected network.

Jump graphs of fully-connected and modular (scale-free) RBN, n=100

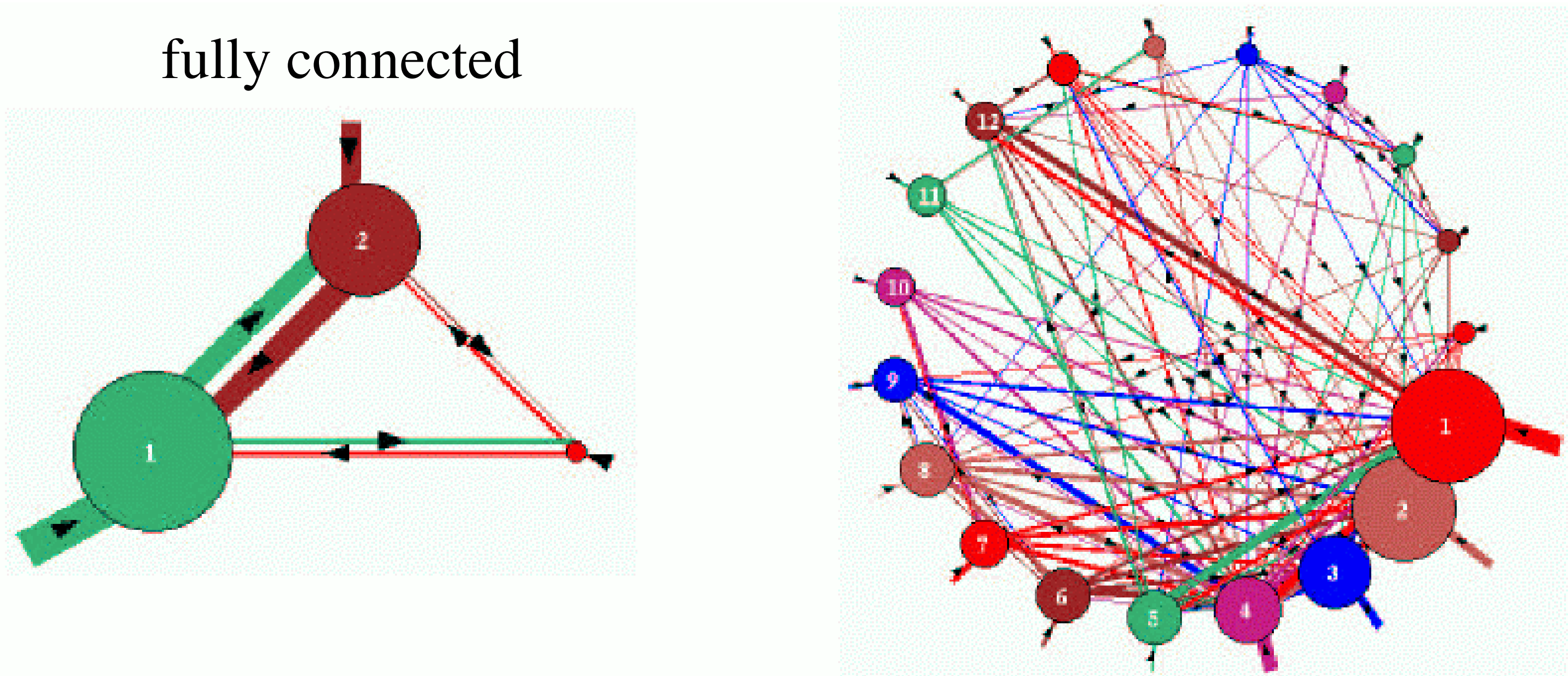
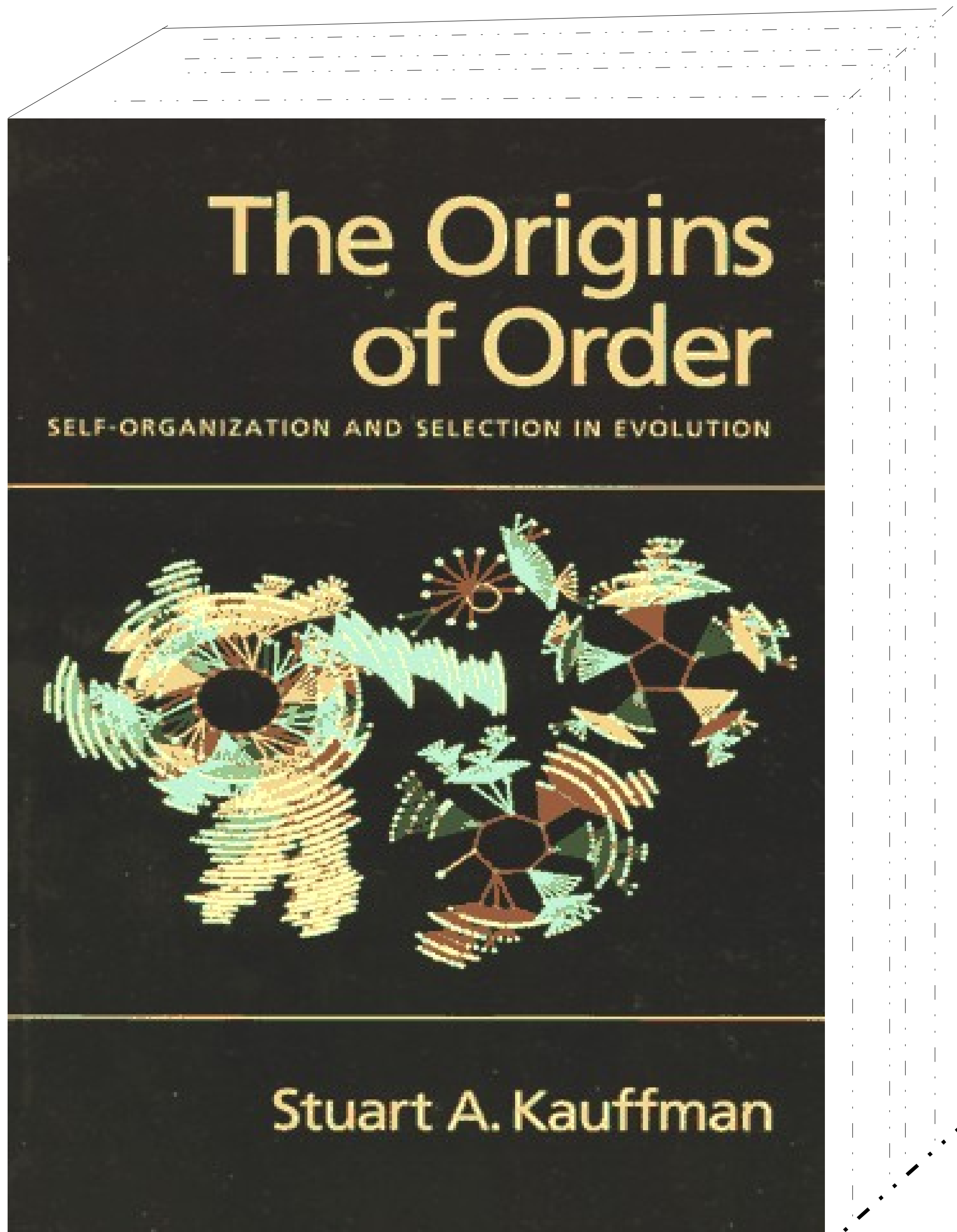


Figure 17: The meta-graphs of the attractor frequency histograms in figure 16, showing the probability of jumping between basins due to single bit-flips to attractor states. Nodes representing basins are scaled according to basin volume. Links are scaled according to both basin volume and the jump probability. Arrows indicate the direction of jumps. Short stubs are self-jumps. *left*: The fully connected network: the percentage of self-jumps is 60%, 36% and 6% respectively. *right*: The modular network: (for the 19 largest basins only, out of 53) the percentage of self-jumps is 41%, 20% and 31% respectively for the 3 largest basins. 11 of the smallest basins (not shown) are unreachable.

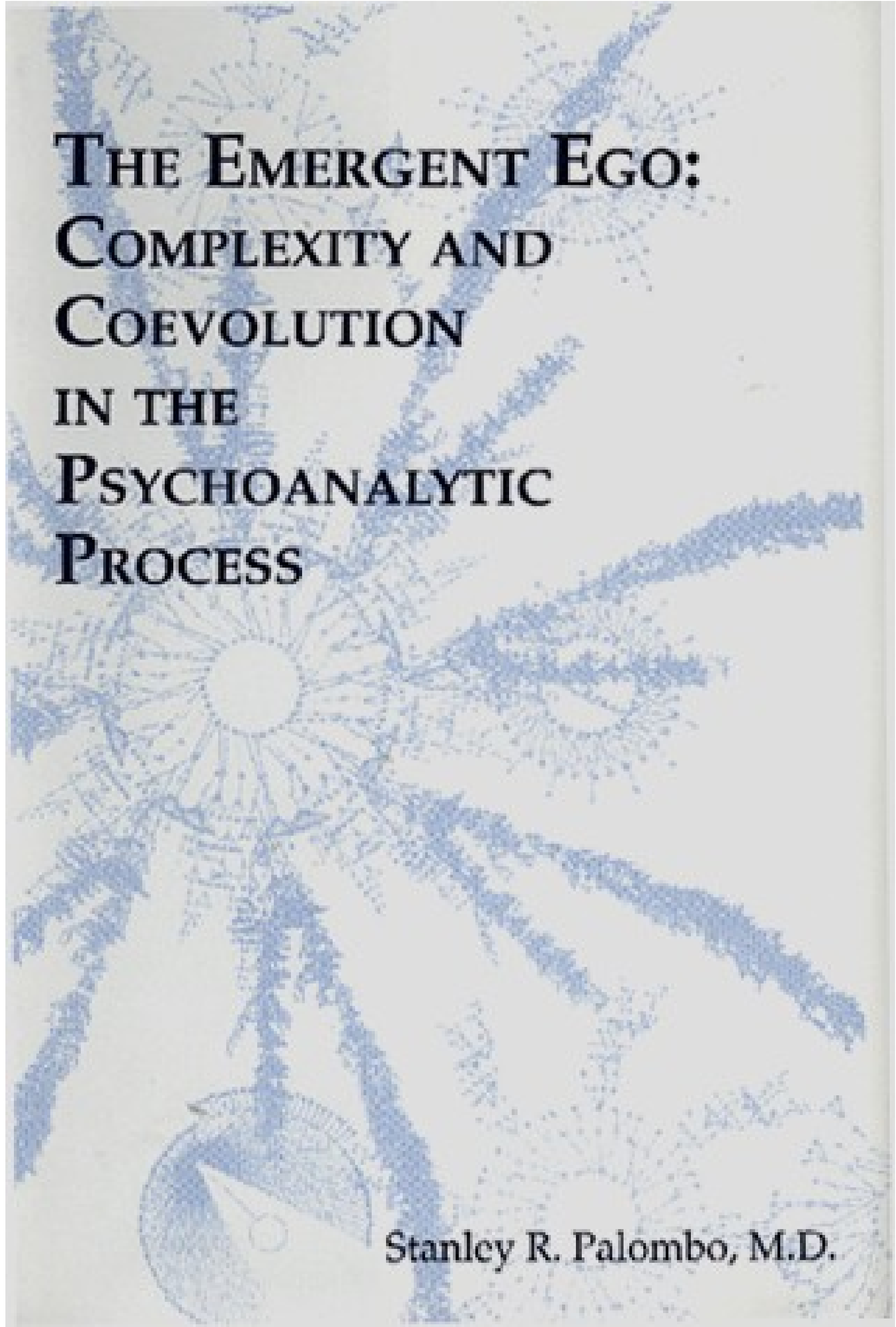
Breaking a network into weekly linked modules increases both the number and stability of basins. Conversely, adding more links between the modules reduces both the number and stability of basins. The modules in the modular network behave like discrete coupled oscillators, perturbing each other between their alternative sub-attractors.

RBN are applied as models of genenetic regulatory networks (cell types = attractors)



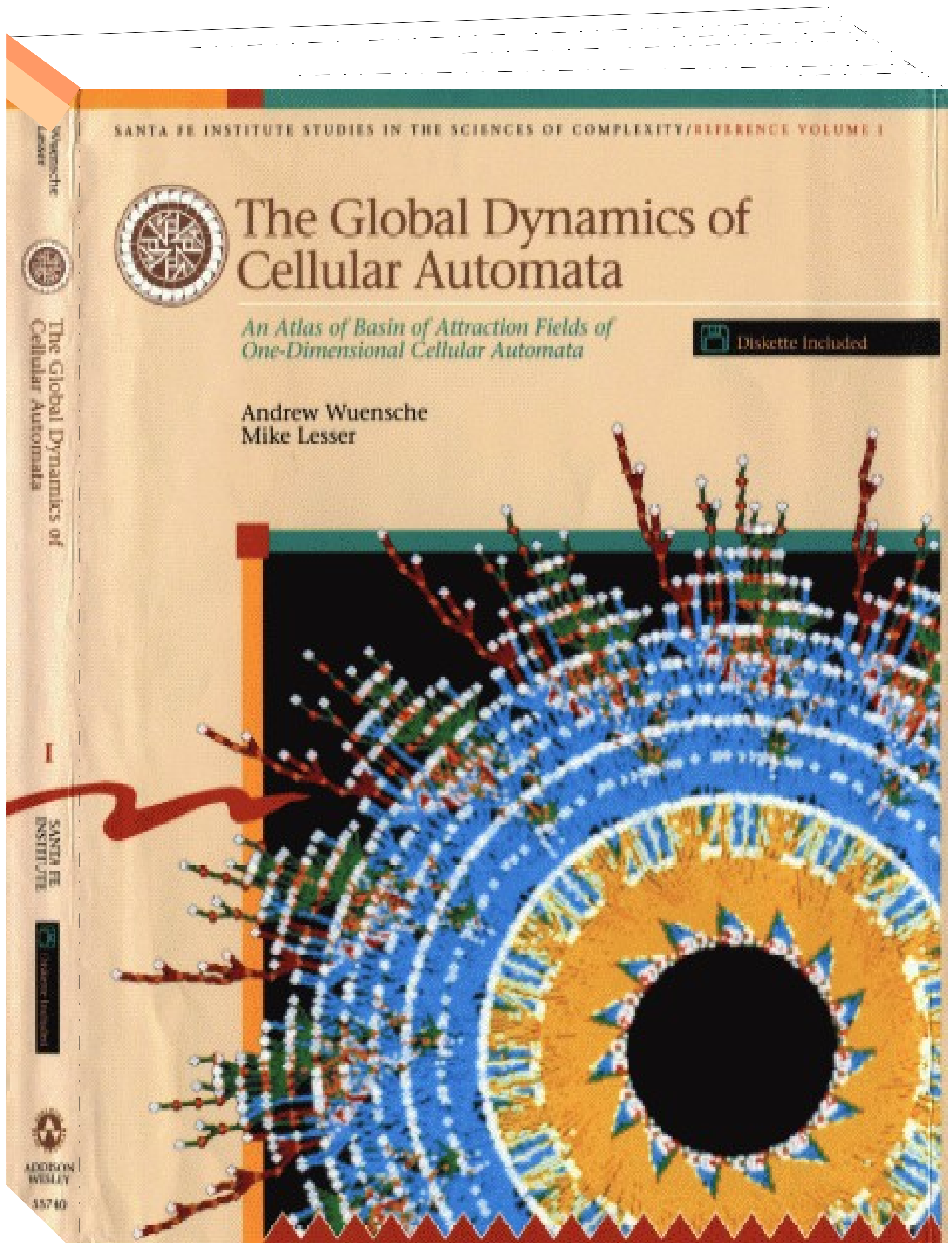
Published 1993

RBN Basins of Attraction made by DDLab !


THE EMERGENT EGO:
COMPLEXITY AND
COEVOLUTION
IN THE
PSYCHOANALYTIC
PROCESS
Stanley R. Palombo, M.D.

The Global Dynamics of Cellular Automata
Andrew Wuensche and Mike Lesser

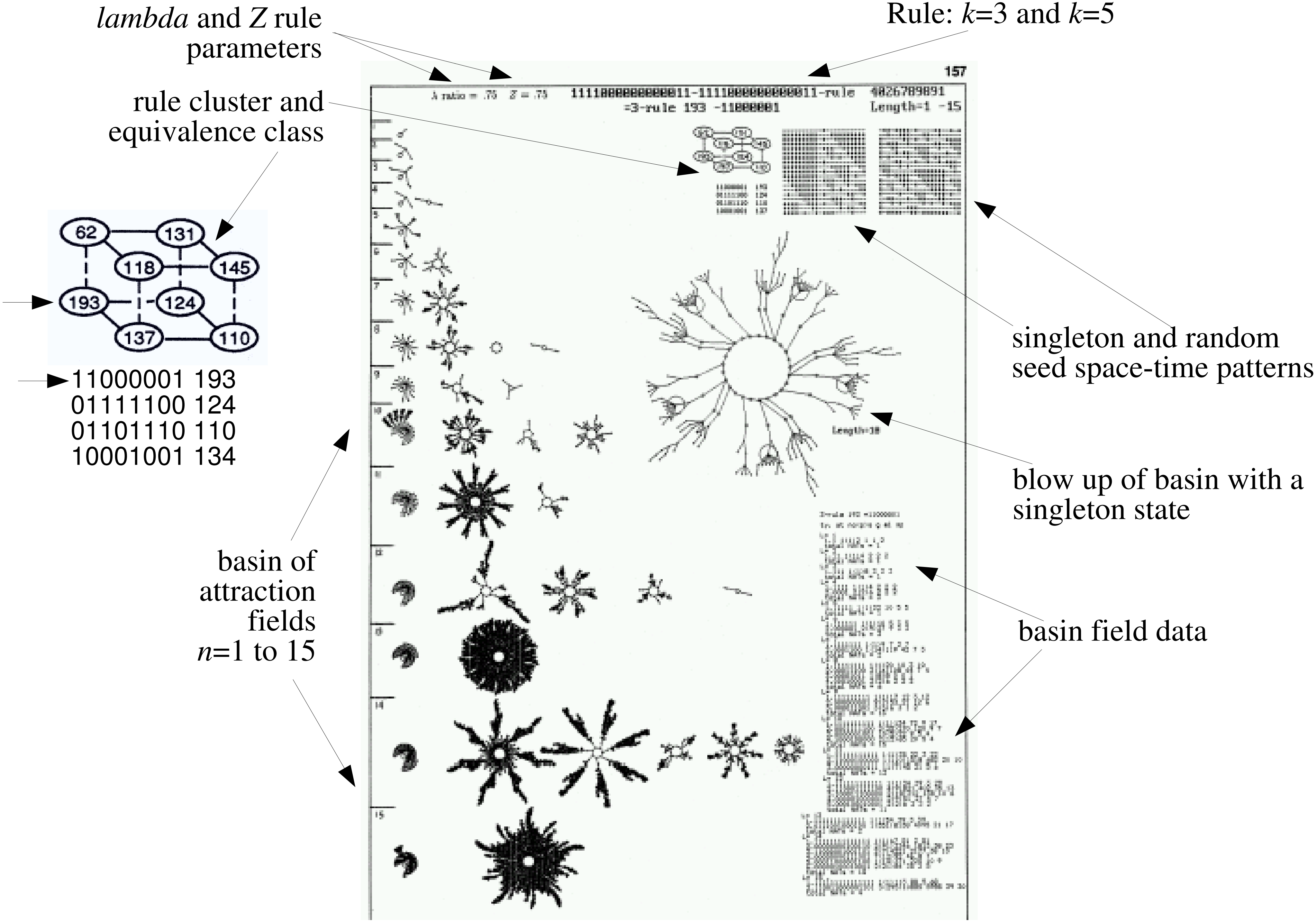
An Atlas of Basin of Attraction Fields of One-Dimensional Cellular Automata



Published 1992
Santa Fe Institute
out of print but available online
www.ddlab.org

- Forward by Chris Langton
- Cellular Automata
- Basin of Attraction Fields
- CA Parameters
- Rotation Symmetry
- Rule Clusters
- Limited Pre-image Rules
- *Z* Parameter
- Basin Field Topology and Rule Space
- Mutation
- The Atlas Program
- Atlas of Basin of Attraction Fields

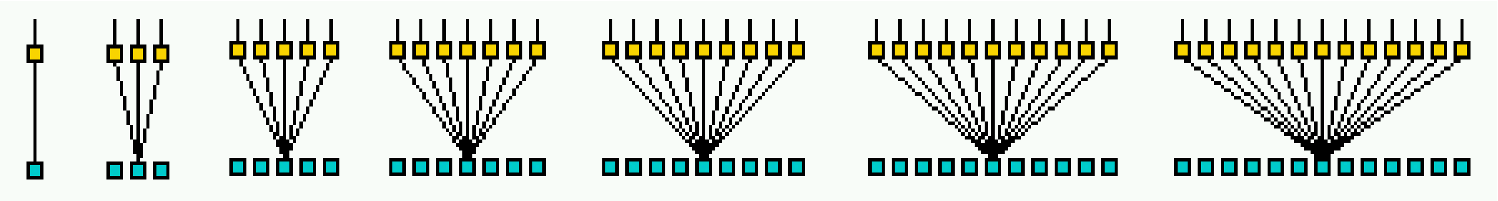
A page from the Atlas



The Altas shows all non-equivalent k=3 rules (88), and k=5 totalistic rules (64), in this format

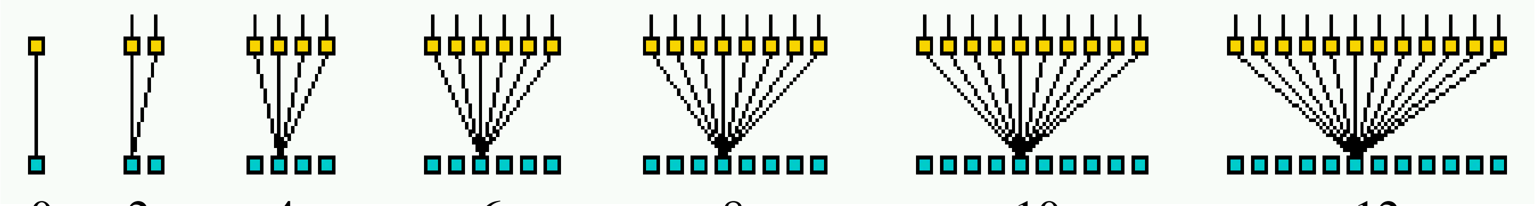
1D neighbourhoods pre-defined in DDLab, max-k=25

odd k



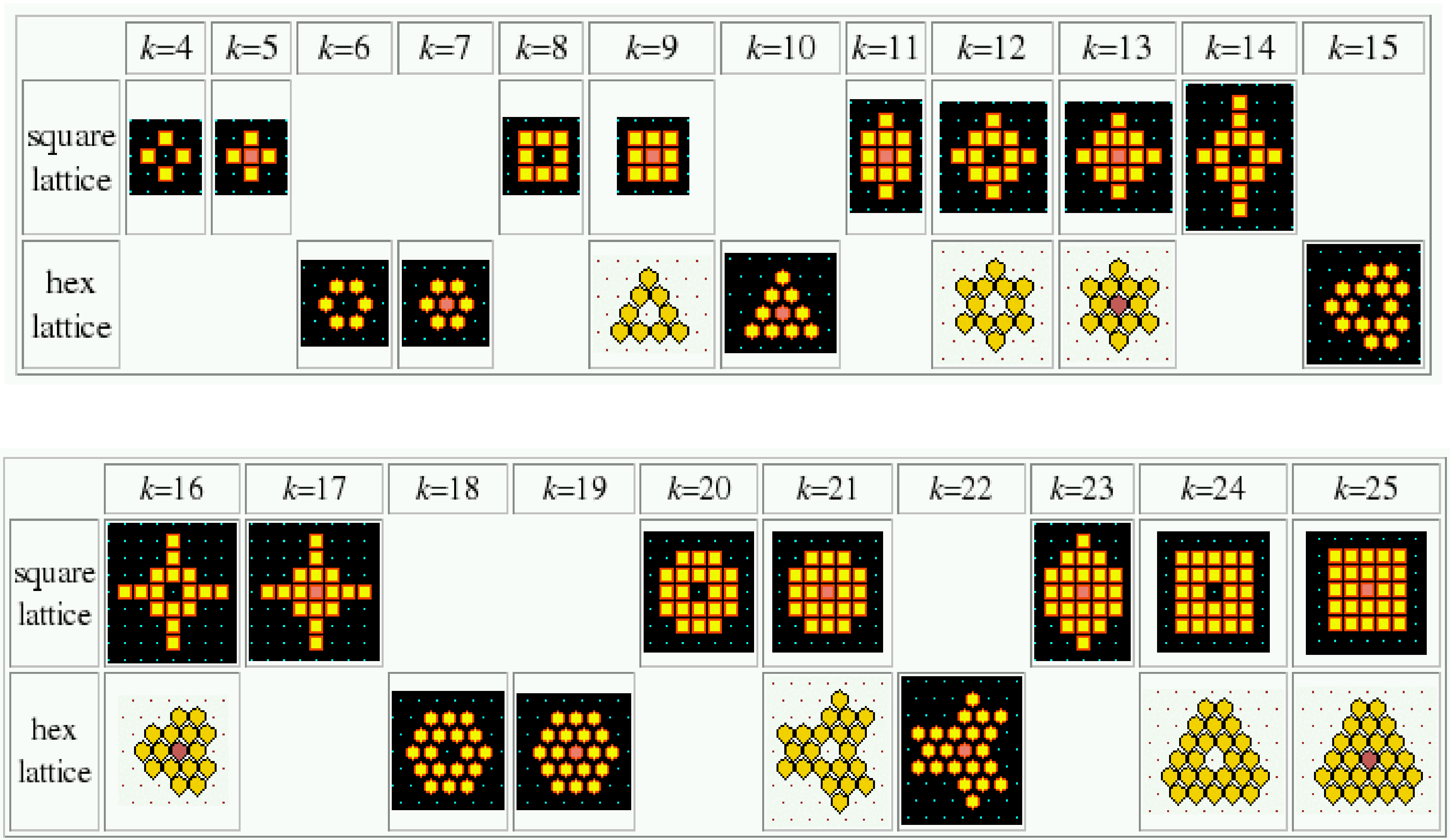
1 3 5 7 9 11 13 . . . .

even k
extra cell is
on the right



0 2 4 6 8 10 12 . . . .

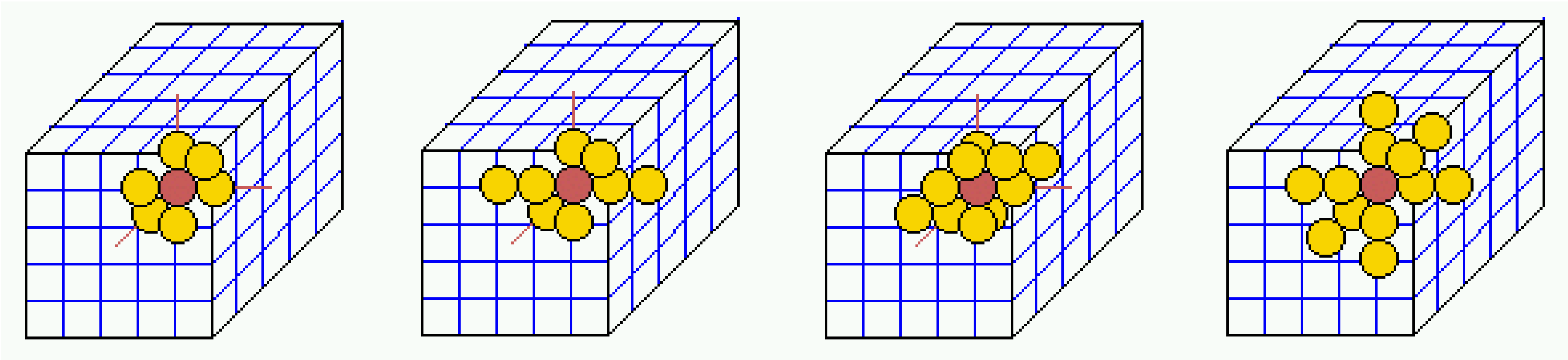
2D neighbourhoods pre-defined in DDLab, max-k=25



The neighbourhood defines the lattice, either square or hexagonal. If k is even, the central cell is not included. Neighbourhoods for k = 1 to 3 are as in 1D.

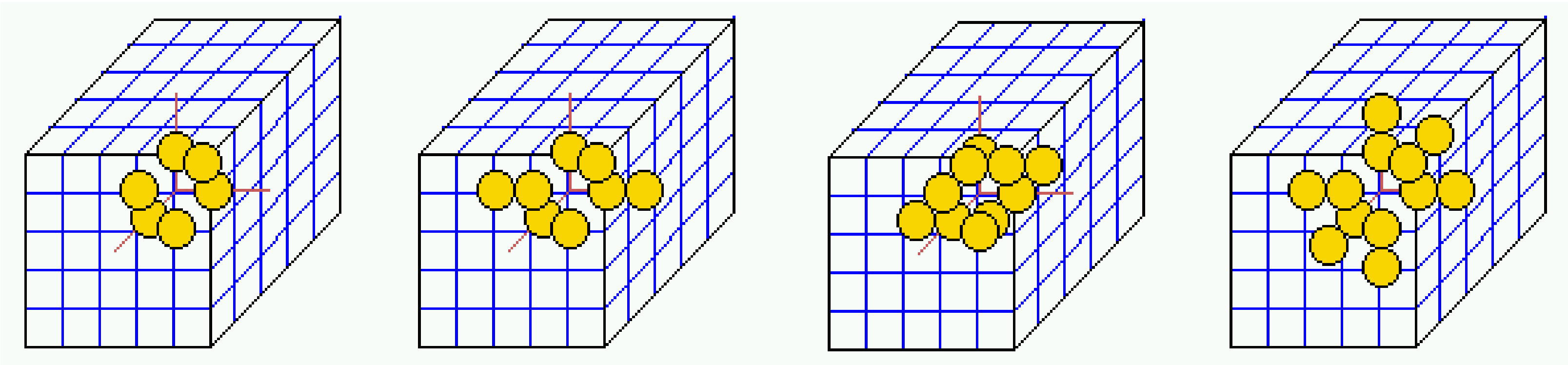
3D neighbourhoods pre-defined in DDLab, max-k=25

odd k



even k

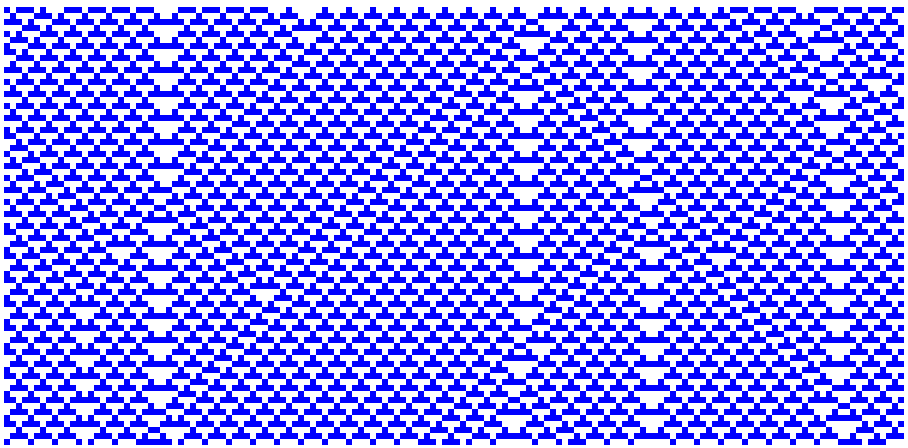
the central cell is not included



The neighbourhoods are shown in a 3d axonometric projection, imagine looking up into cage.
Even k does not include the central target cell. (neighbourhoods for k = 1 to 5 are as in 2D).

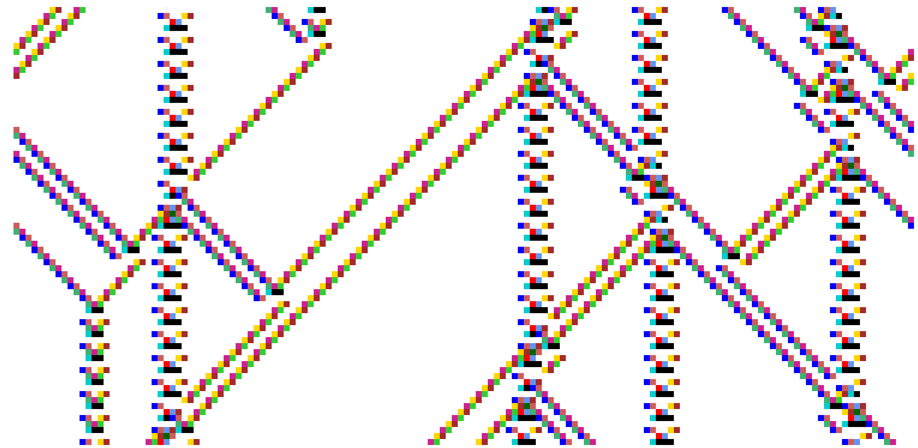
Filtering space-time patterns to reveal gliders

unfiltered



cells by value

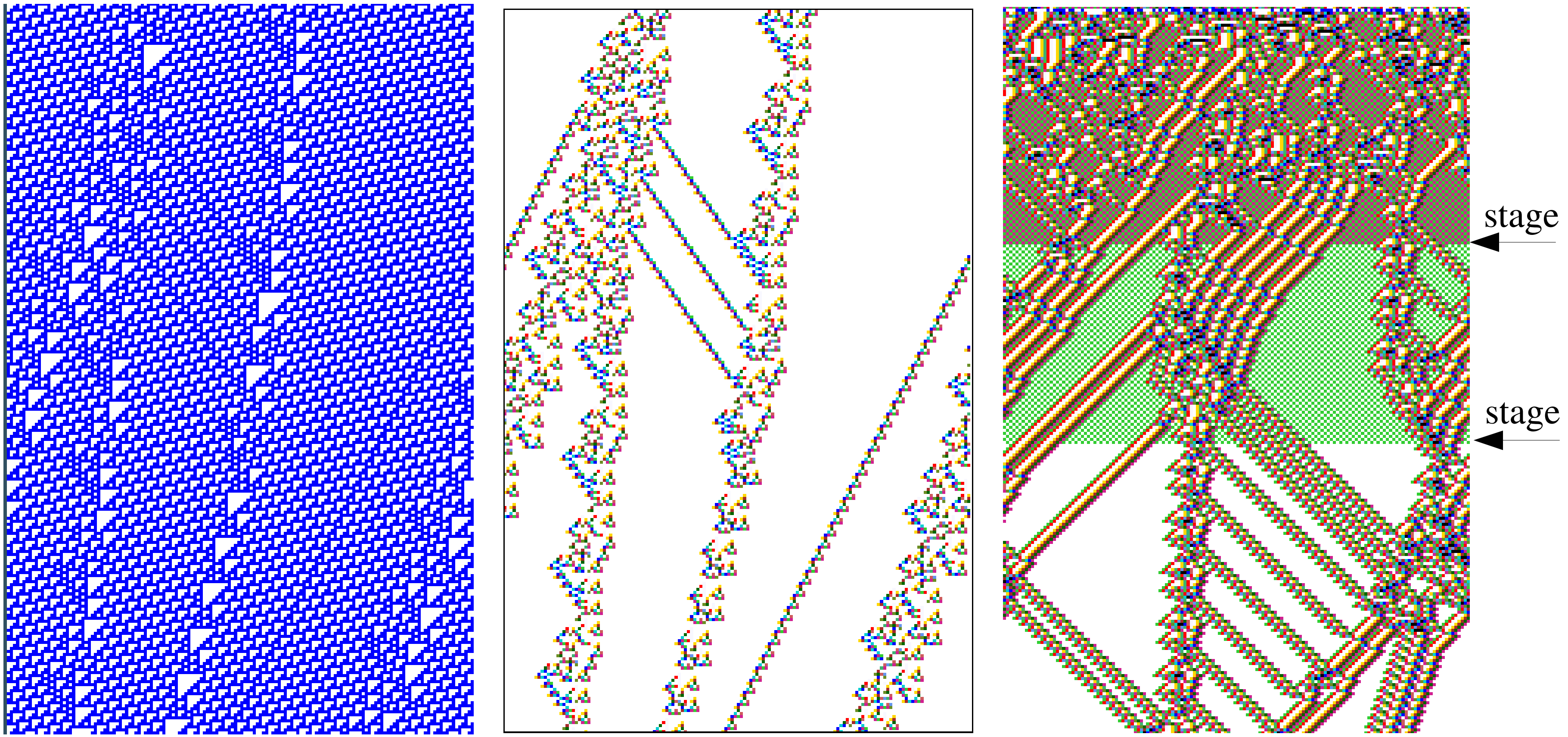
filtered



cells by look-up and filtered

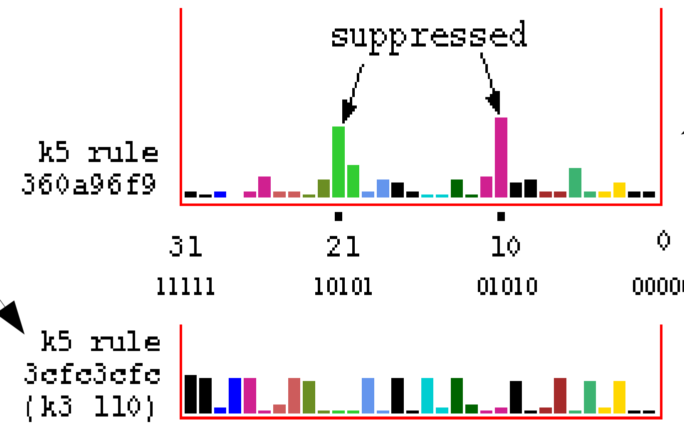
Filtering is done by by keeping track of the most frequently occurring neighbourhoods, then progressively suppressing the display cells that “looked up” those neighbourhoods in the rule-table. Filtering reveals gliders and other complex space-time structures, which may be dislocations in a complicated background domain. For effective filtering the rule may need to be transformed to an equivalent rule with larger *k* (as in this case for *k*=3 rule 54). Filtering is done interactively, on-the-fly, in DDLab for any CA.

Filtering space-time patterns - examples



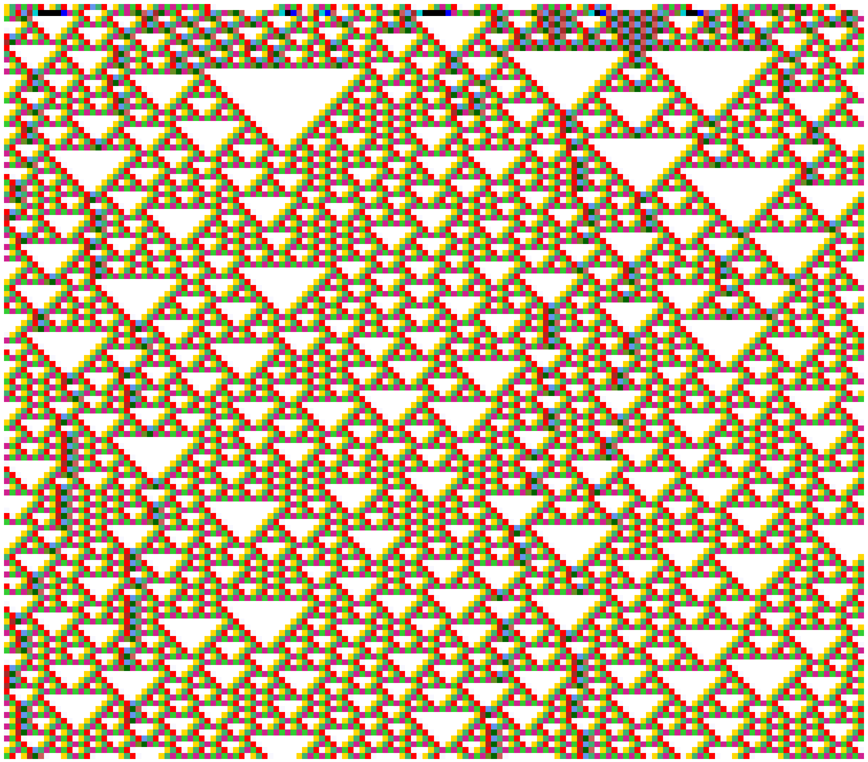
rule 110

rule 36 0a 96 f9
filtered in
2 stages

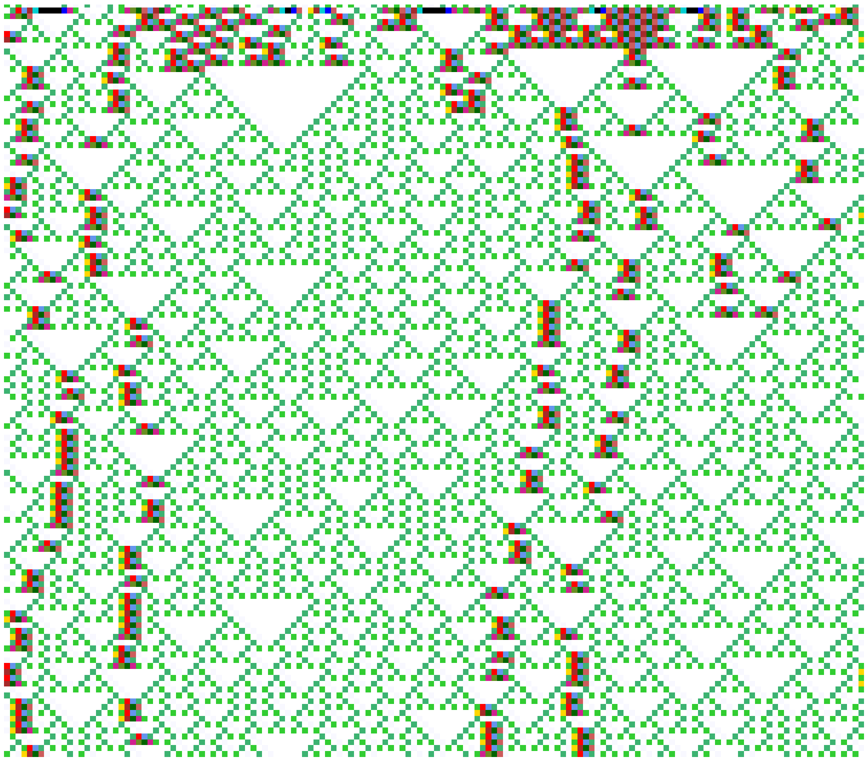


Look-up frequency histograms of the space-time patterns elating above. Suppressed neighborhoods are indicated with a dot.

Filtering chaotic domains to show up discontinuities



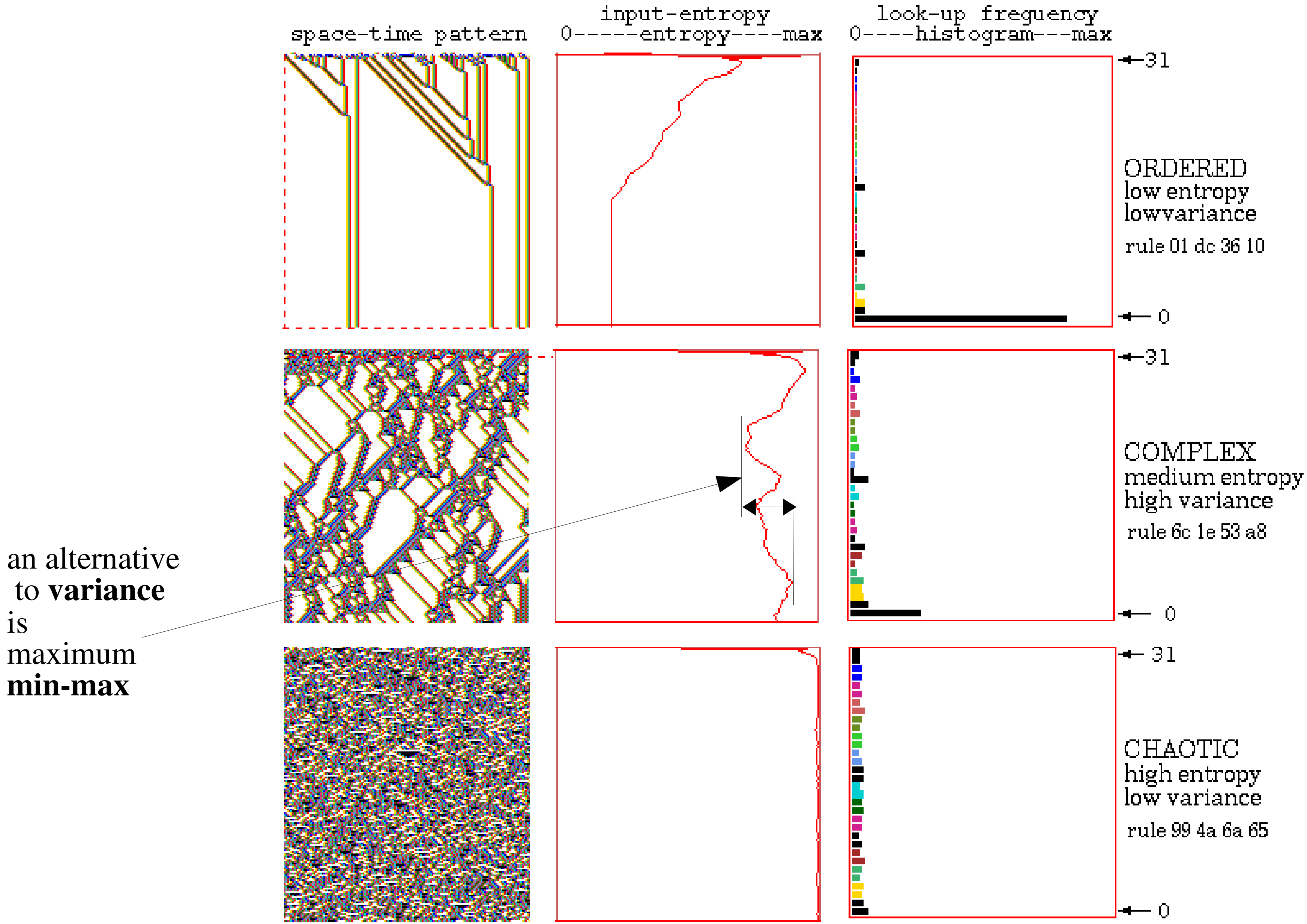
k=3 rule 18



k=3 rule 18, partly filtered

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.

The lookup frequency histogram and input-entropy



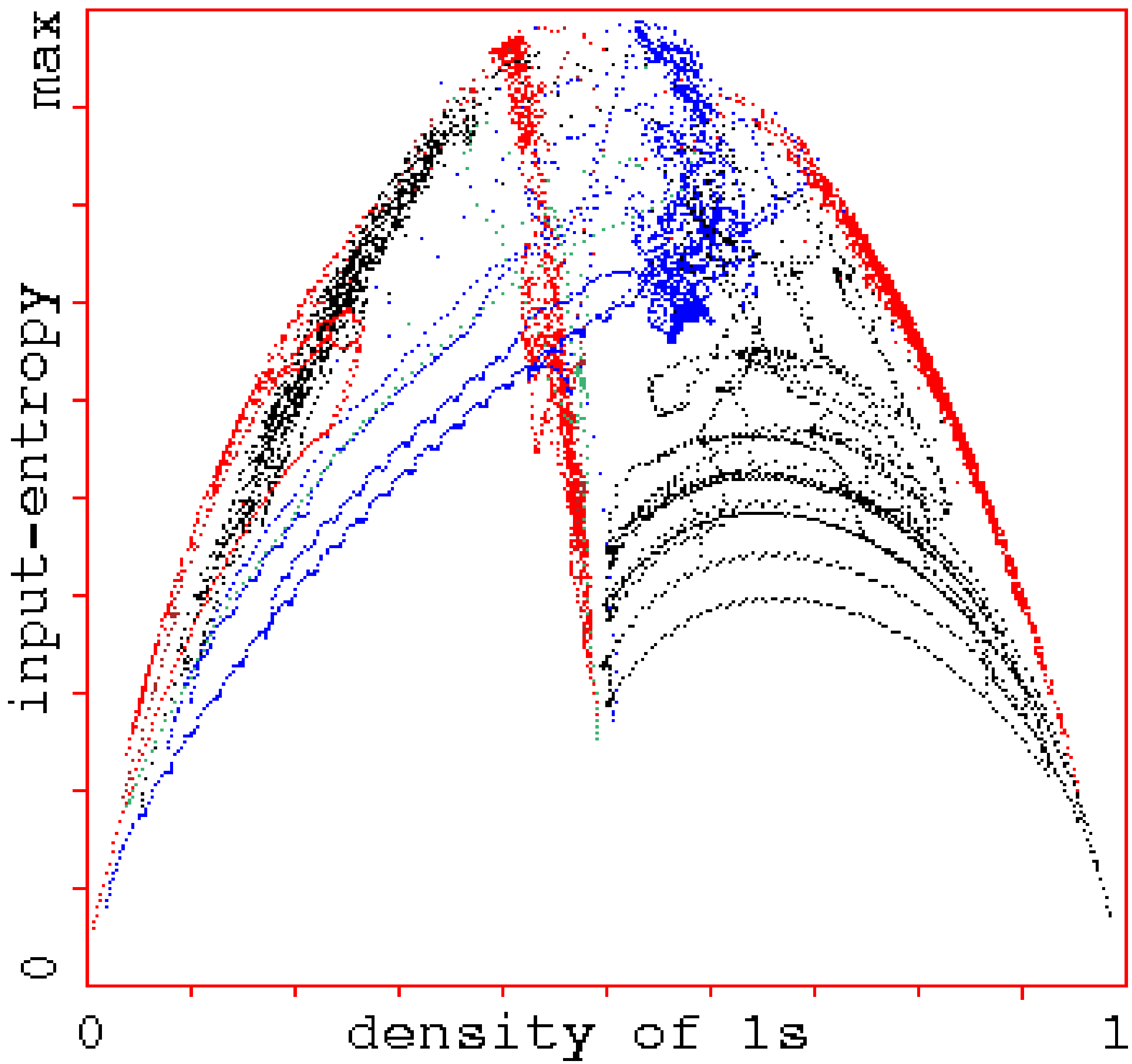
1D **ordered**, **complex** and **chaotic** space-time patterns from the same random initial state. Alongside is a a snapshot of the **lookup frequency histogram,** and a plot of the **input-entropy** taken over a moving window of 10 time-steps. Input-entropy and its **variance** (or **standard deviation**} provides a non-subjective measure for recognizing ordered, complex and chaotic rules automatically; only complex rule show high **input-entropy variance**.

Entropy-density scatter plots, complex rule signatures

The input-entropy S is the Shannon entropy of the input frequency,

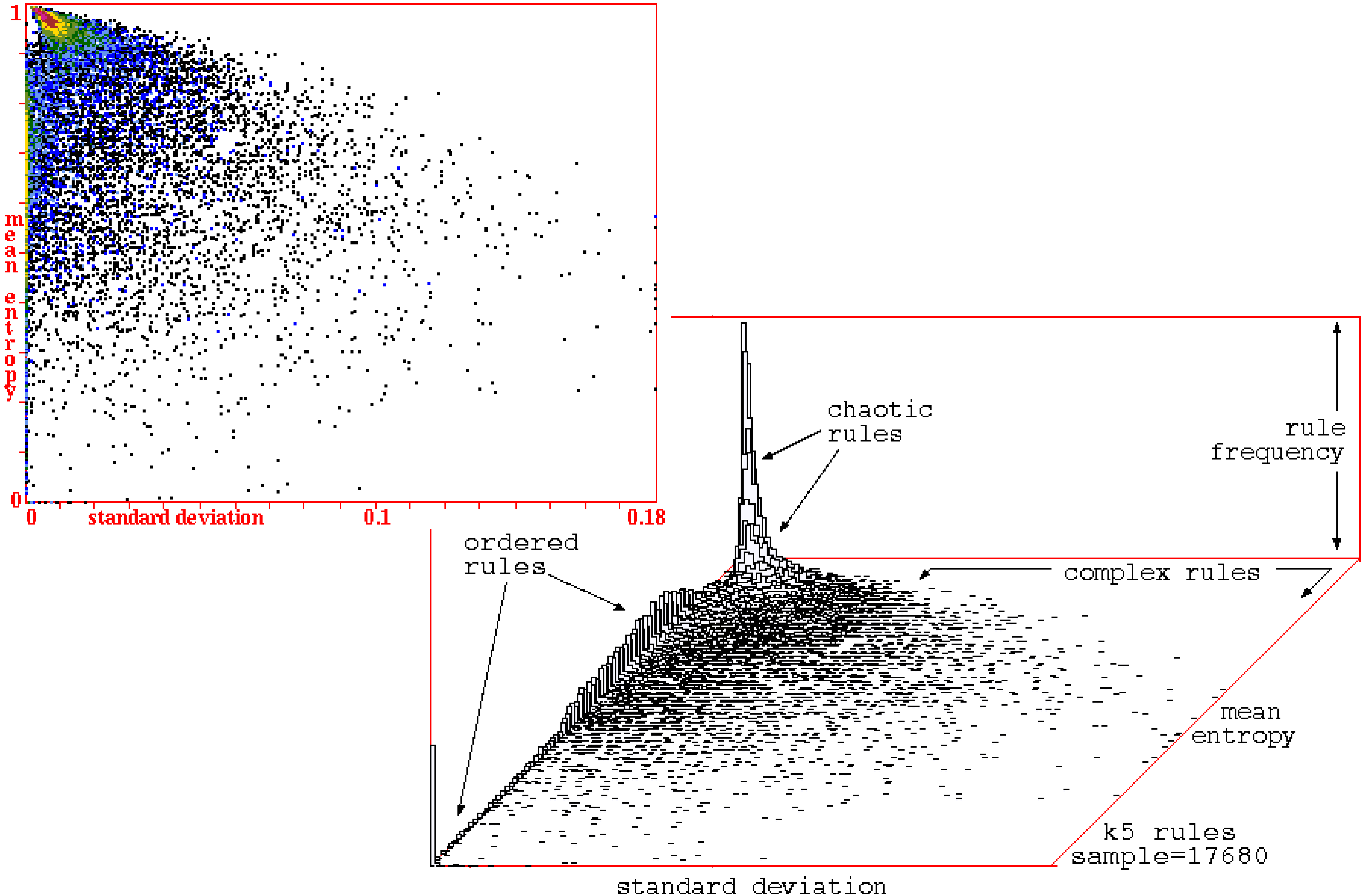
$$S^t = -\sum_{i=1}^{2^k} \left(\frac{Q_i^t}{n} \times log \left(\frac{Q_i^t}{n} \right) \right)$$

were Q_i^t is the frequency of i at time t, n is the network size, and k the neighbouhood size. In practice the measures where smoothed by being taken over a moving window of 10 time-steps.



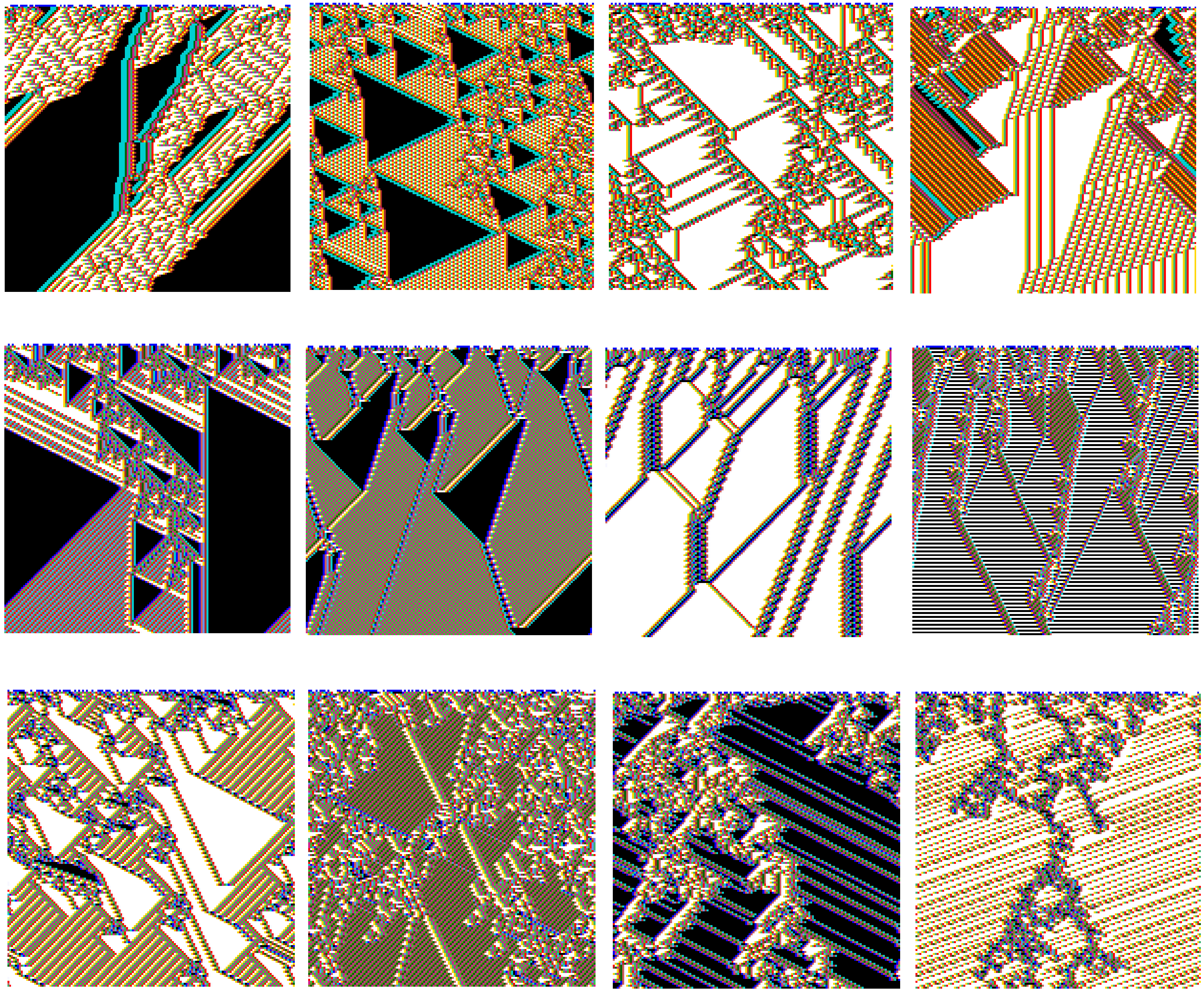
Input-entropy is plotted against the density of 1s relative to a moving window of 10 time-steps for a number of complex rules (k=5, n=150), each of which has its own distinctive signature, with a marked vertical extent, i.e. high input-entropy variance. About 1000 time-steps were plotted from several random initial states for each rule.

Classifying random samples of 1D CA automatically (*k*=5)



[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.

Complex space-time patterns from the automatic samples

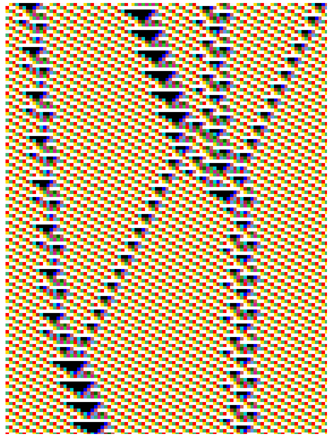


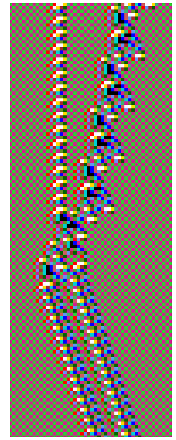
k=5

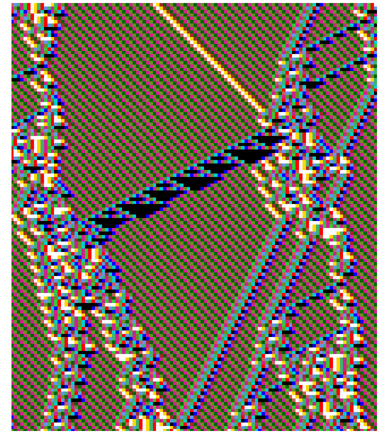
k=6

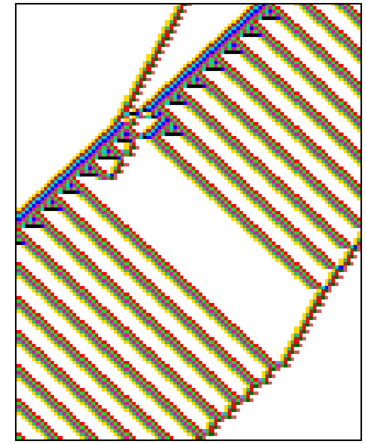
k=7

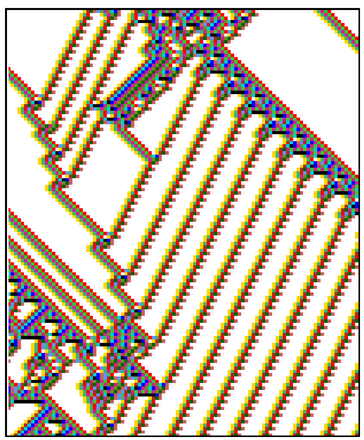
Interacting gliders in 1D CA (k=5)

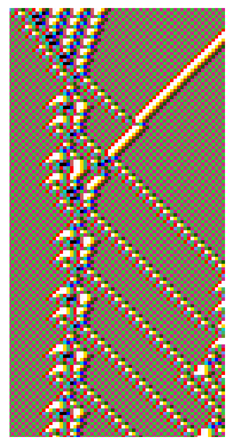

a) e0897801

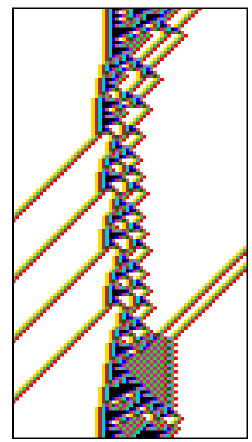

b) 7e8696de

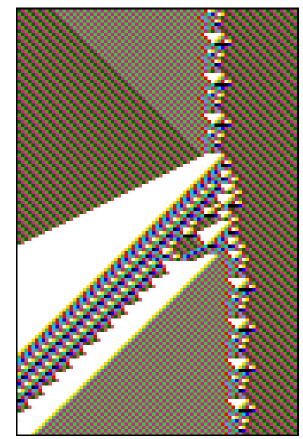

c) ad9c7232

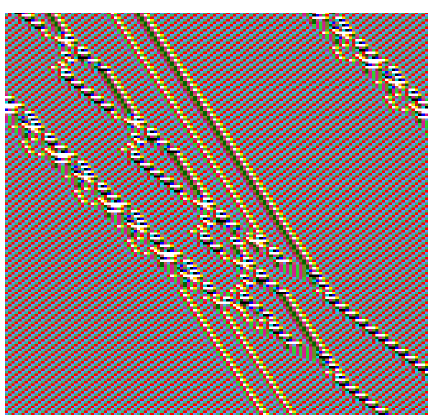

e) 1c2a4798

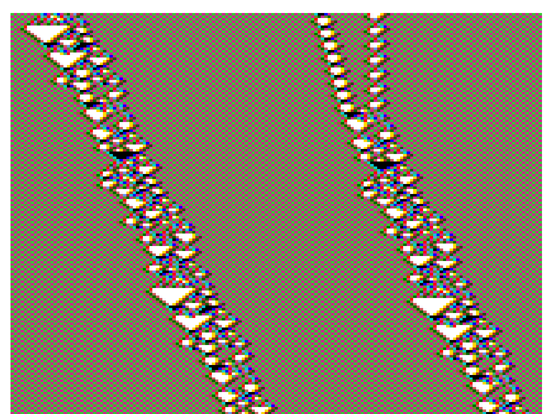

a) 5c6a4d98

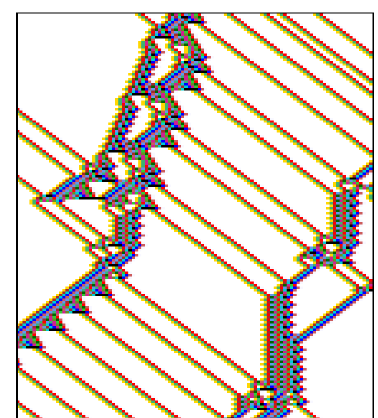

b) 360a96f9


c) 978ecee4

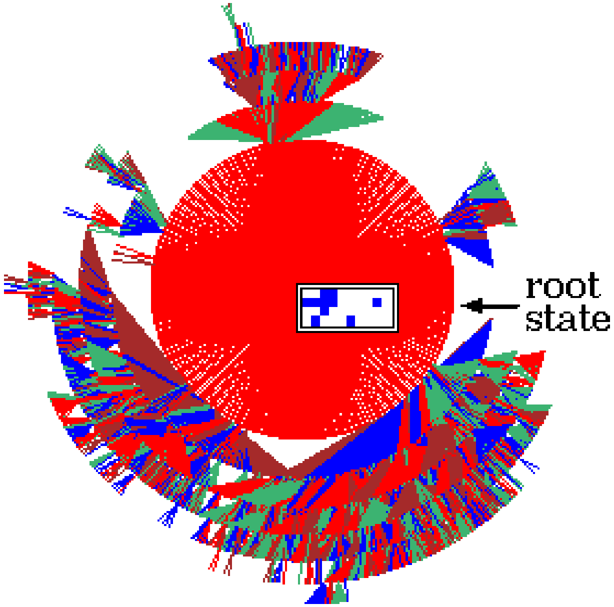

e) bc82271c


a) 89ed7106


b) b51e9ce8

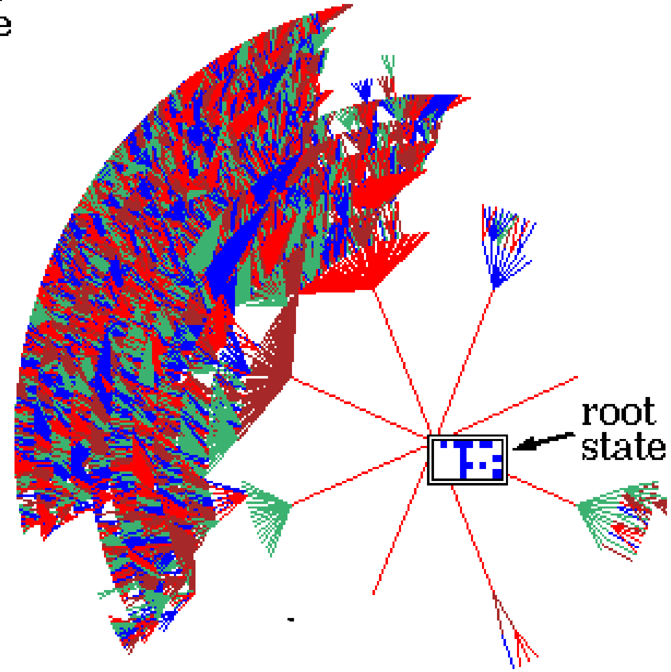

c) 6c1e53a8

Ordered - Complex - Chaotic
CA sub-trees



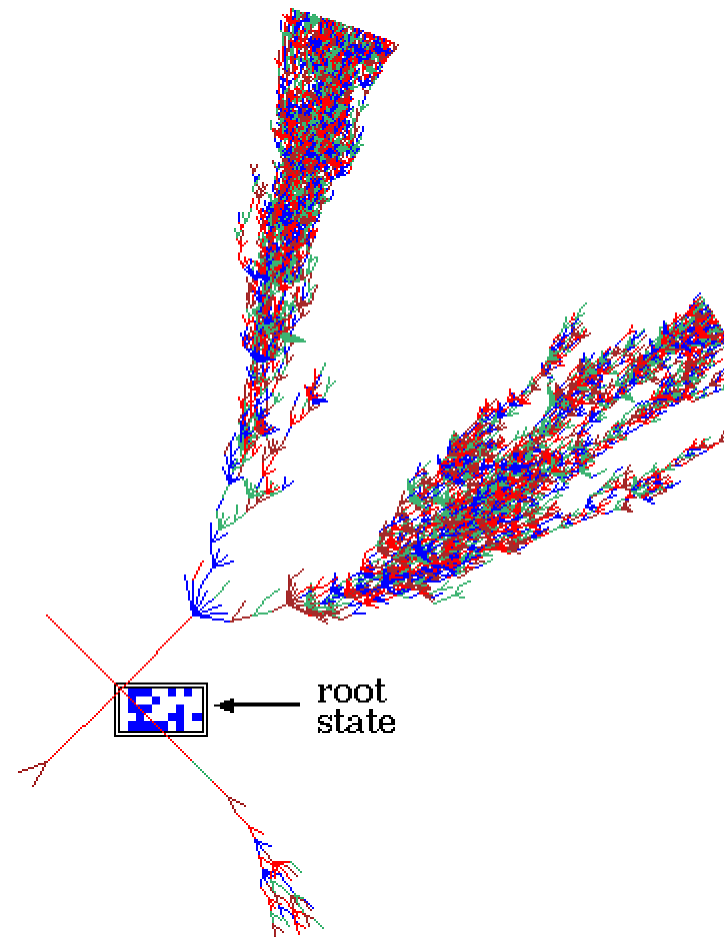
high convergence

Ordered: Rule 01dc3610, n=40. The complete sub-tree 7 levels deep, with 58153 nodes, G-density =0.931, Z=0.5625, L_r=0.668



medium convergence

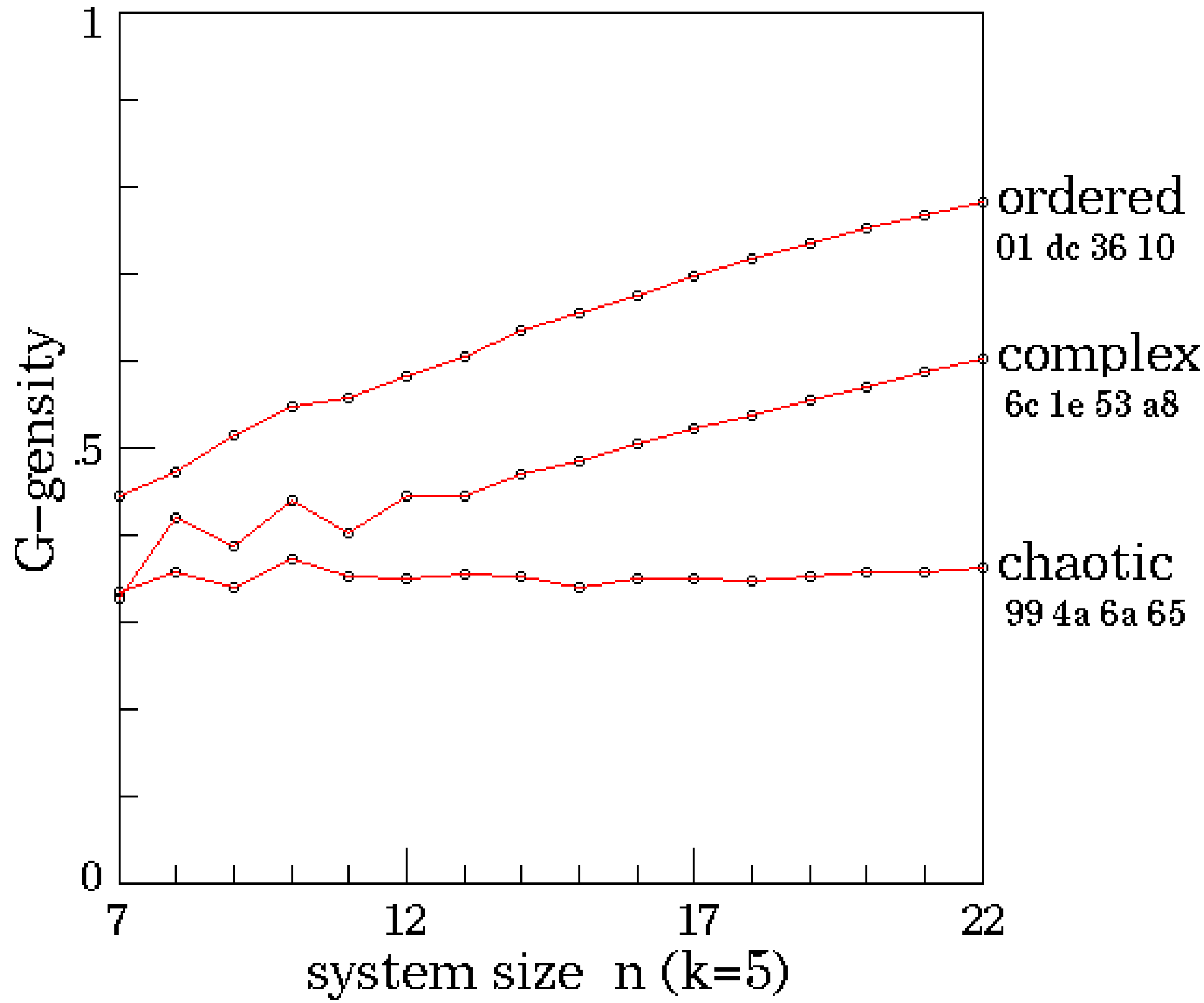
Complex: Rule 6c1e53a8, n=50. The sub-tree, stopped after 12 levels, with 144876 nodes, G-density =0.692, Z=0.727, L_r=0.938



low convergence

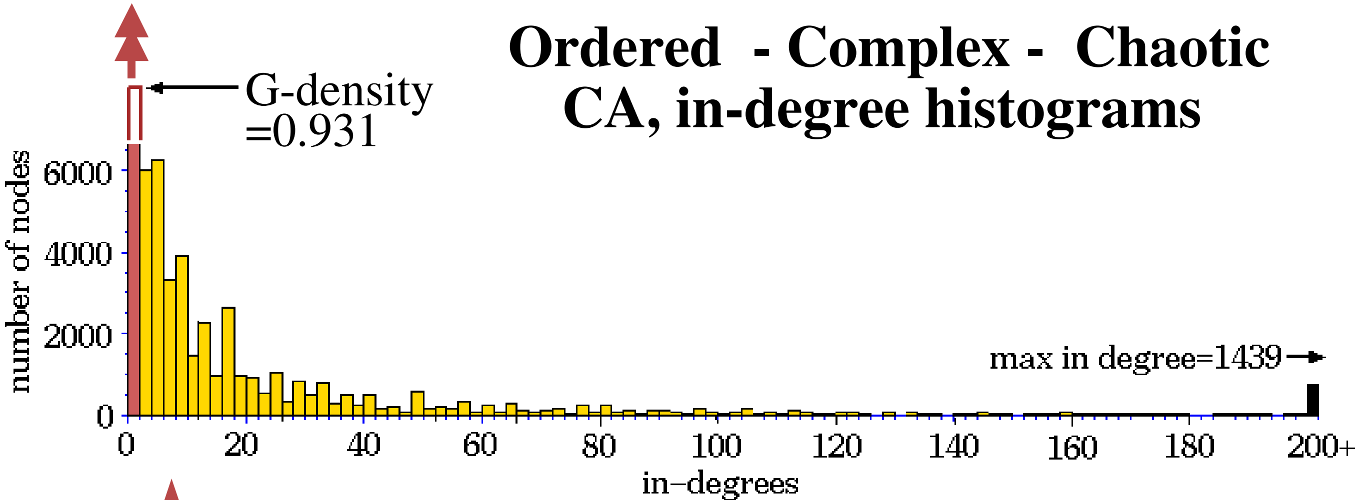
Chaotic: Rule 994a6a65, n=50. The sub-tree stopped after about 75 levels, with 9446 nodes, G-density =0.487, Z=0.938, L_r=0.938

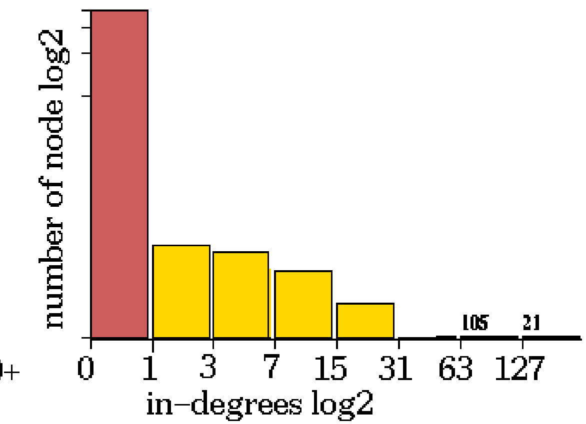
G-density in basins of attraction plotted againts *n* (a simple measure of convergence)

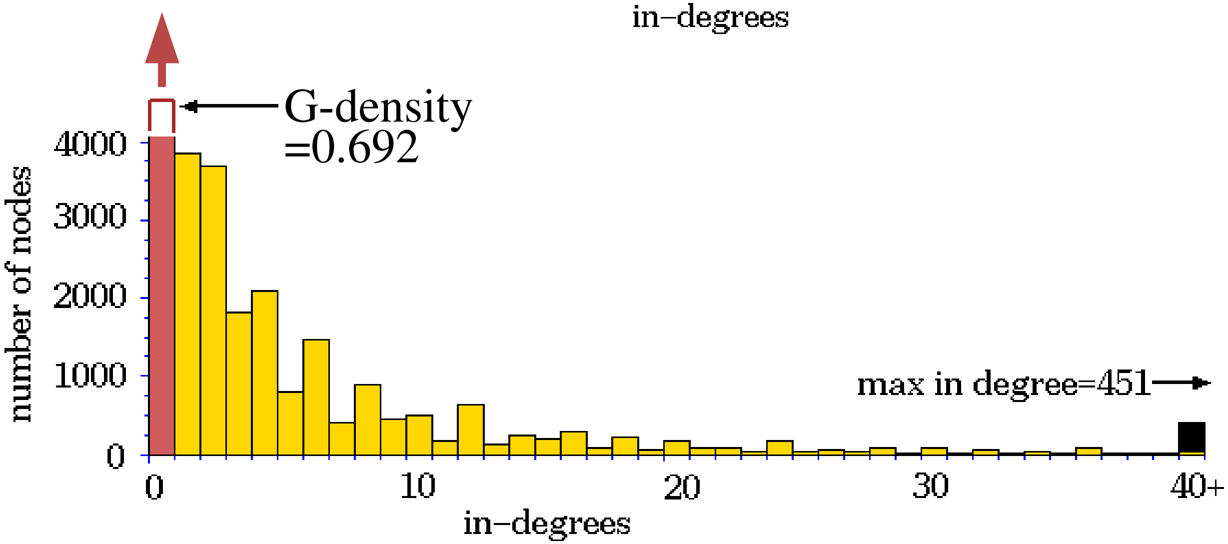


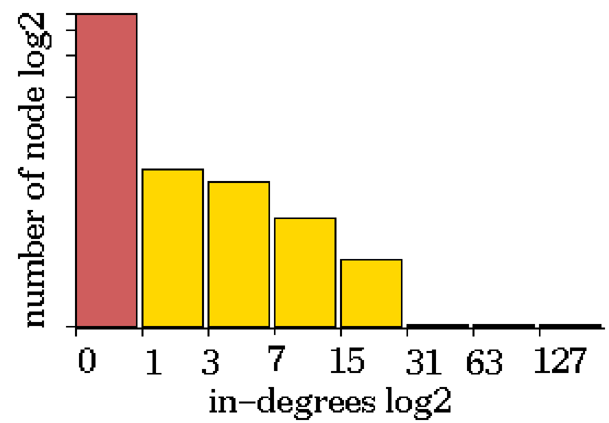
G-density (garden-of-Eden density, leaf density) plotted against system size system size *n*, for the ordered, complex and chaotic rules. The the entire basin of attraction field was plotted for *n* = 7 to 22, and garden-of-Eden states counted. The relative G-density and rate of increase with *n* provides a simple measure of convergence.

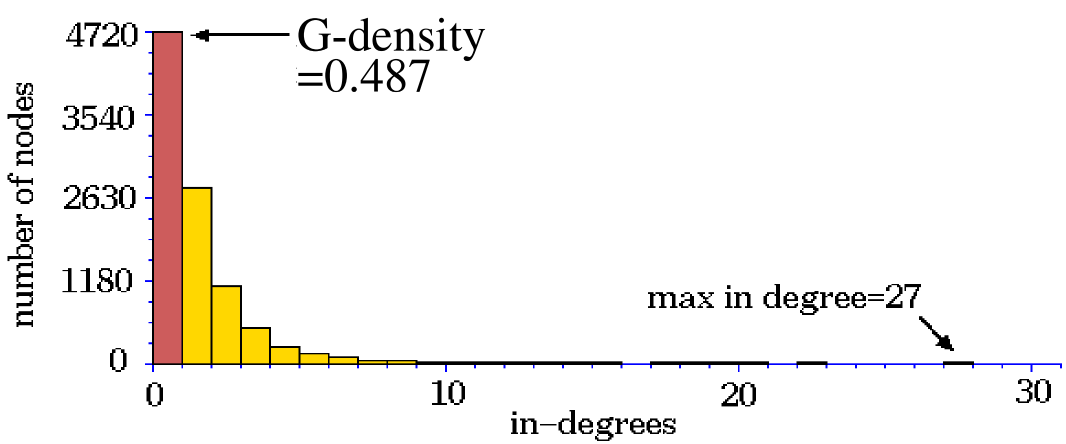
Ordered - Complex - Chaotic
CA, in-degree histograms

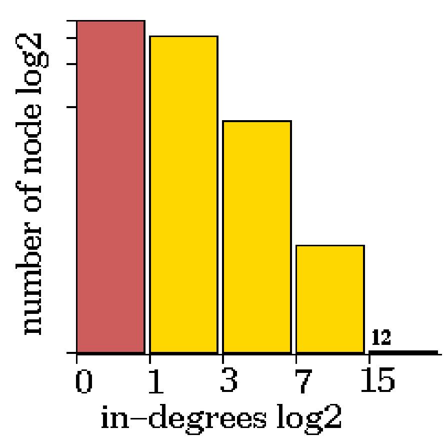




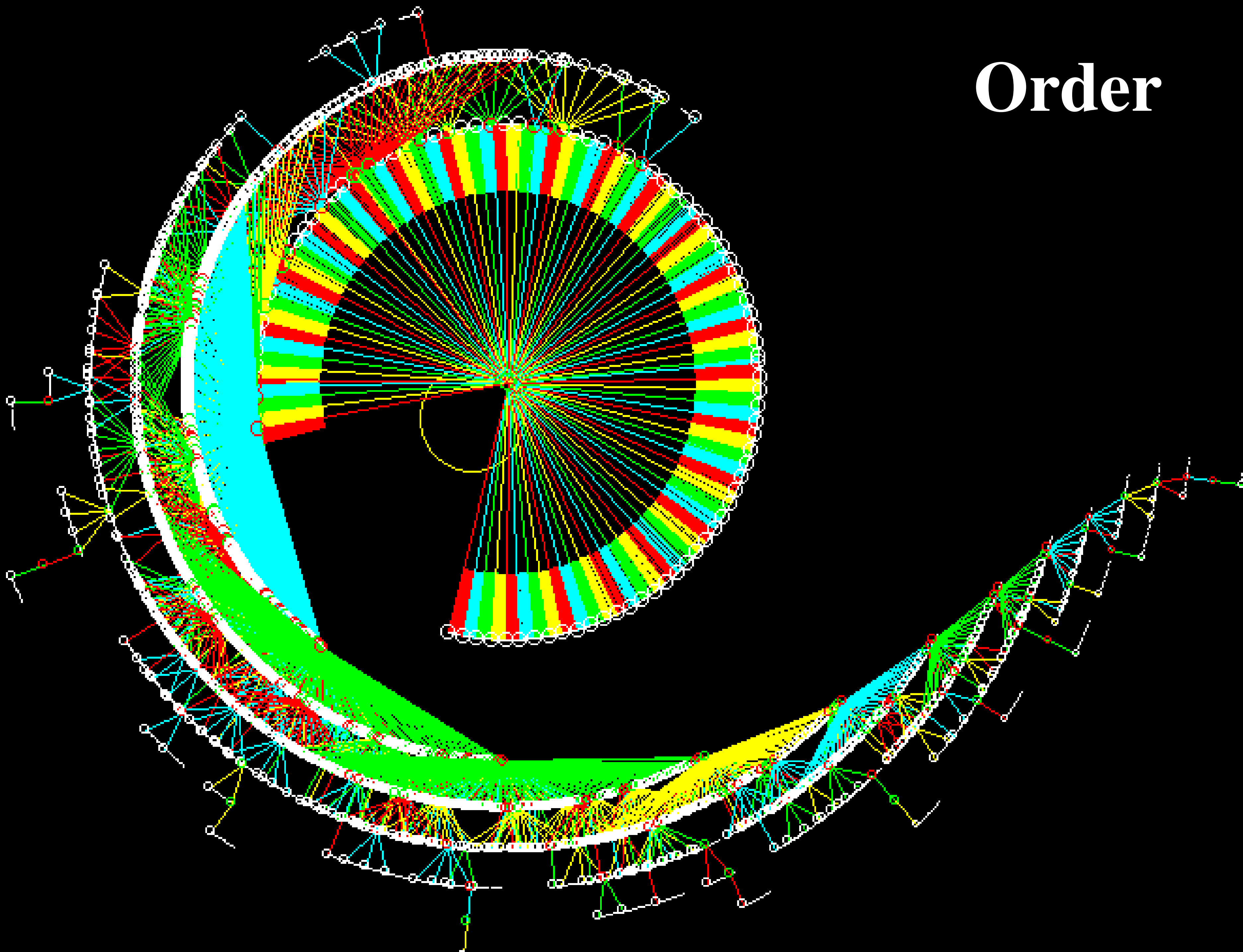








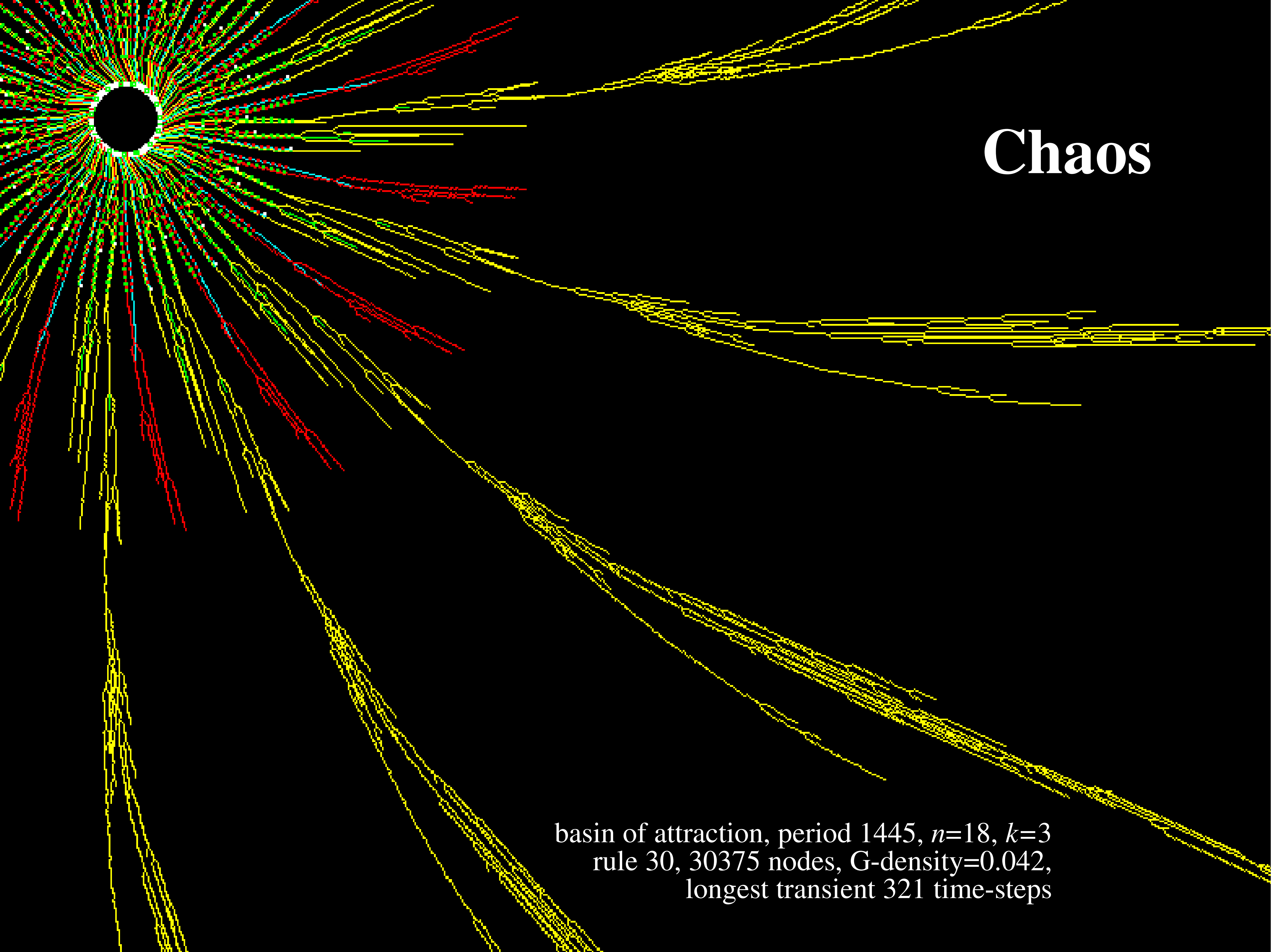
Order



basin of attraction (point attractor), n=15, k=3 rule 250, 32767 nodes, G-density=0.859

Complexity

basin of attraction, period 27, n=18, k=3 rule 110, 93825 nodes, G-density=0.611



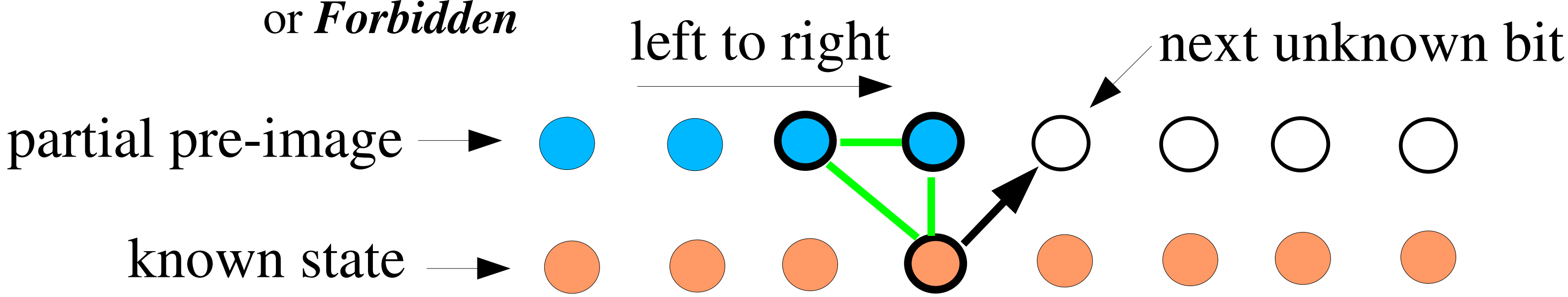
Chaos

basin of attraction, period 1445, n=18, k=3
rule 30, 30375 nodes, G-density=0.042,
longest transient 321 time-steps

The 1D CA reverse algorithm and the *Z* parameter
(very briefly)

111	110	101	100	011	010	001	000	....neighbourhoods k=3
0	1	1	0	1	1	1	0	....outputs (rule 110), rule-space=256

Deterministic ***Ambiguous*** or ***Forbidden***



Try to fill in the next unknown bit in partial pre-image (from left to right) by reference to the look-up table; there are 3 possibilities: (there is an equivalent procedure right to left)

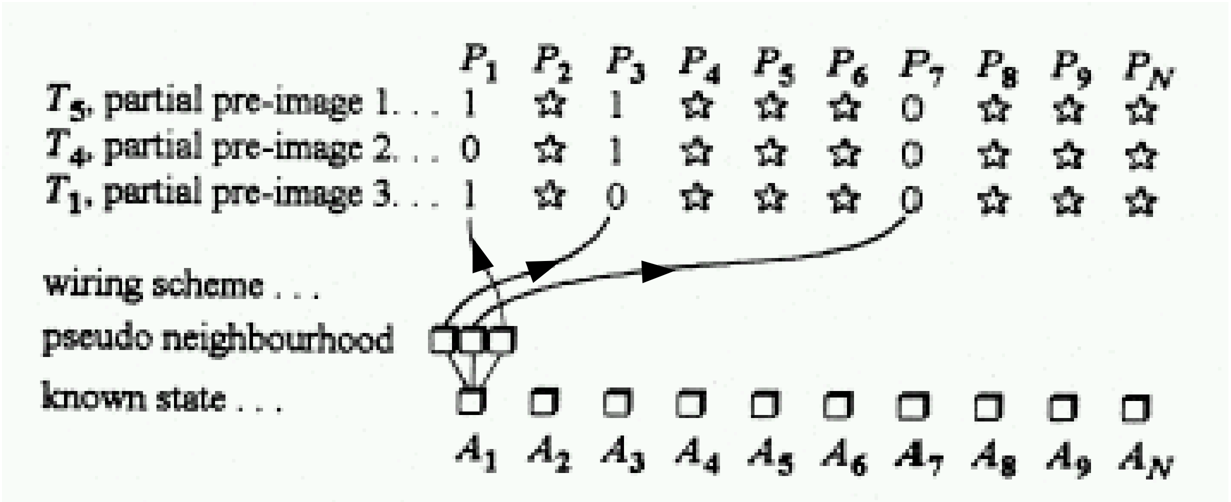
1. ***Deterministic*** - one valid solution: fill this in, and move to the next unknown bit.
2. ***Ambiguous*** - two valid solutions (for v=2): recursively follow both – the pre-image has doubled.
3. ***Forbidden*** - no valid solutions: halt.

$\mathbf{Z_L}$= fraction of deterministic sub-rules = **probability** that the next unknown cell is determined. This is found directly from the lookup table. $\mathbf{Z_R}$ is found equivalently from right to left The **Z parameter** = the greater of $\{\mathbf{Z_L}:\mathbf{Z_R}\}$. Z predicts convergence in subtrees, thus **order-chaos**. (The actual procedure is a bit more involved. The reverse algorithm for RBN is different, but also works for CA).

k-1 bits must be assumed to start, thus there are 2^{k-1}=4 possible starts (for binary CA) to the pre-image. If $\mathbf{Z_L}=\mathbf{Z_R}$=1, in-degrees must be exactly 4 or 0. If either $\{\mathbf{Z_L}$ or $\mathbf{Z_R}\}$=1 (but not both) in-degrees must be less than 4, or 0. These are maximally chaotic rules, where the in-degree is fixed irrespective of n.

The RBN reverse algorithm (including multi-value)

(very briefly)



For a cell in the known state, assign each value in a valid pseudo-neighbourhood to a partial pre-image, according to the wiring. Several partial pre-images may be created.

Repeat for the next cell (taken in any order). If there is a conflict with a value previously allocated, reject the pre-image. Otherwise the number of partial pre-images will increase initially, but then decrease because of conflicts (often to zero). Any survivors are the valid pre-images of the known state.

The algorithm works for CA as well as RBN, of course!

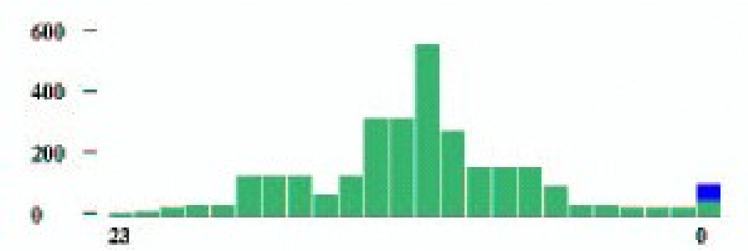


Figure 17: Computing RBN pre-images. The changing size of a typical partial pre-image stack at successive elements. n=24, k=3.

Classification of 1D CA

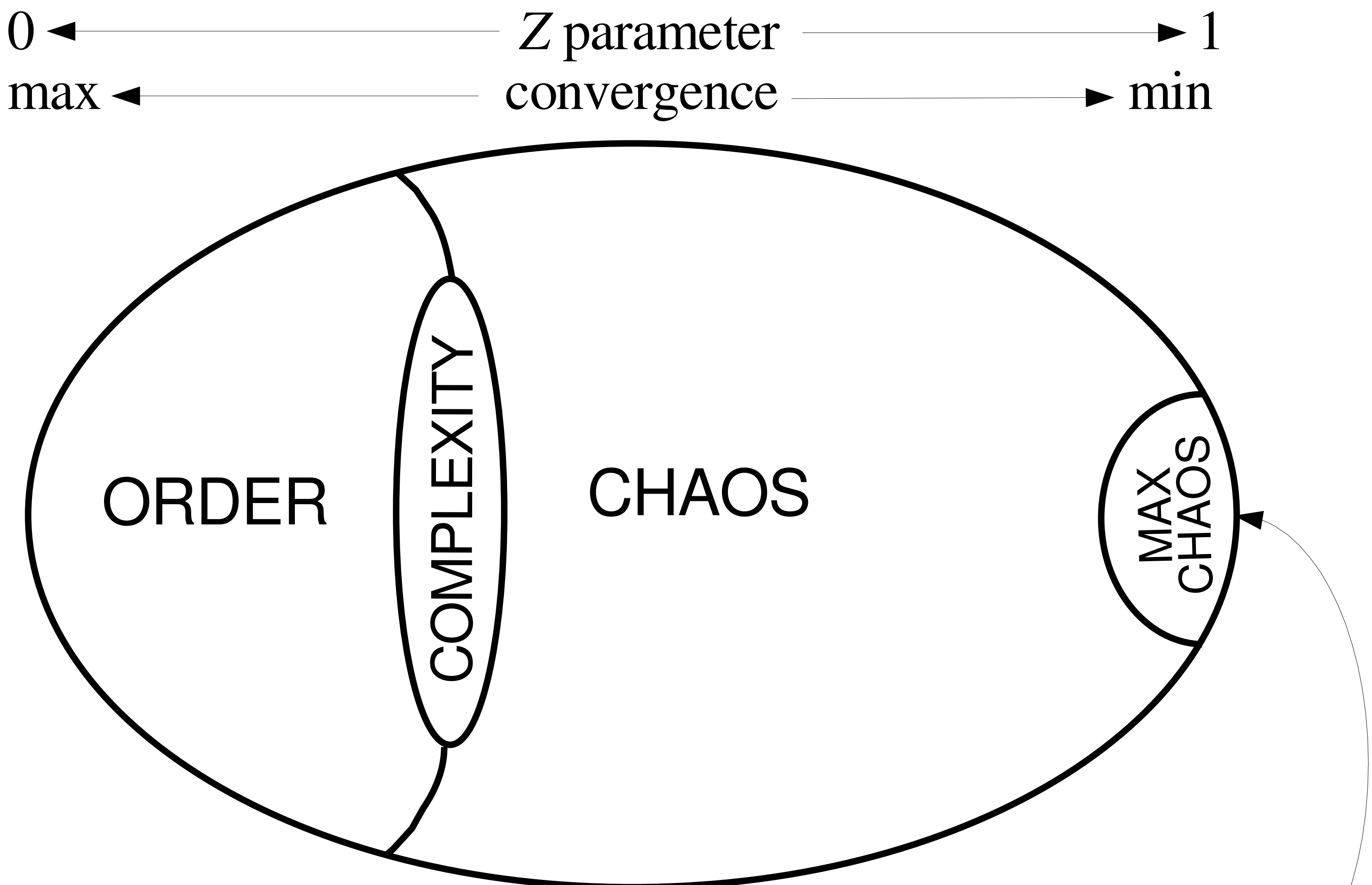
Wolfram: classification based on attractors:

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 (*and others*): Woframs class 4 is a phase transition between 2 and 3, so the revised classification is reordered:

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

A view of CA rule-space
(after Chris Langton)



If $\{Z_L$ or $Z_R\} = 1$ (but not both) the rule is maximally chaotic, with max in-degree $< 2^{k-1}$. For large systems it usually just 1. About √rule-space is maximally chaotic.

Encryption with maximally chaotic "chain" rules
(to encrypt run backward)

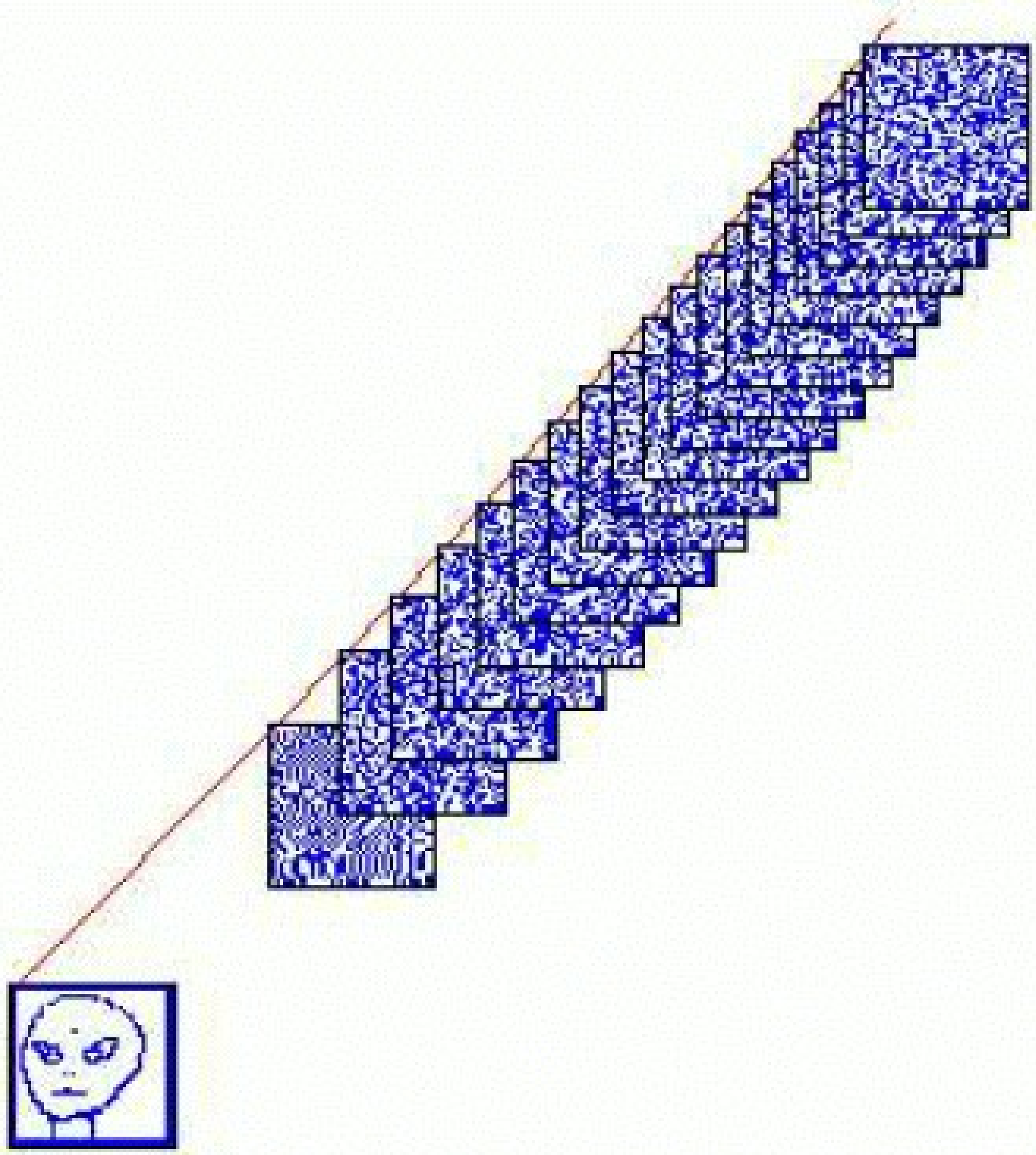


Figure 1: With the "alien" as the root state, a subtree was generated with the reverse algorithm in DDLab; note that the subtree has no branching, and branching is highly unlikely to occur. The rule was a $k = 7$ chain-rule, selected at random (in hex: a7 4e b6 6b b9 c3 a0 81 58 bi 49 94 46 3c 5f 7e), with Z-parameter values: $Z_{left} = 0.617$, $Z_{right} = 1$. The subtree was set to stop after 20 backward steps. The 1d CA is displayed in 2d.

Encryption with maximally chaotic "chain" rules
(to decrypt run forward)

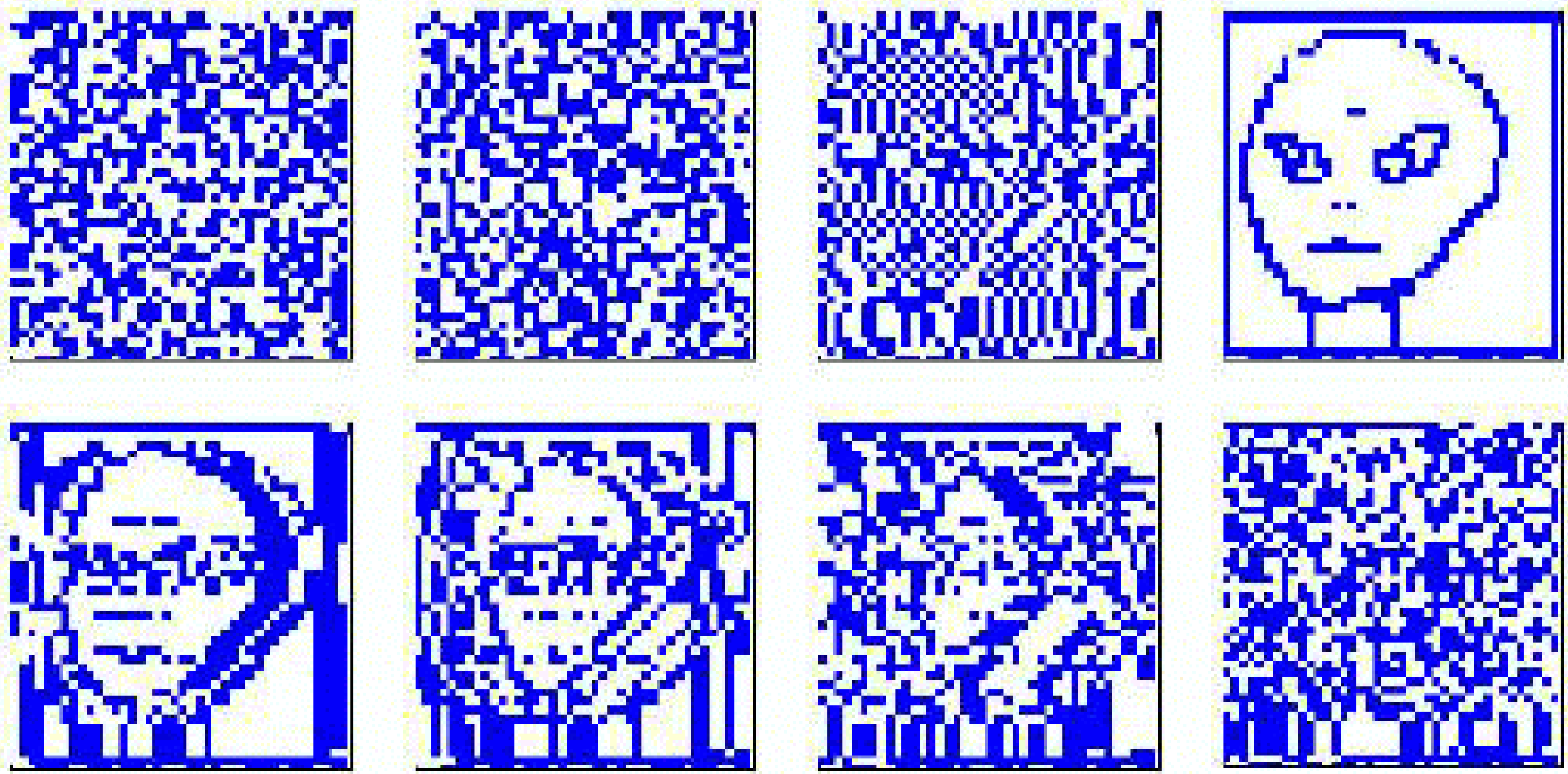
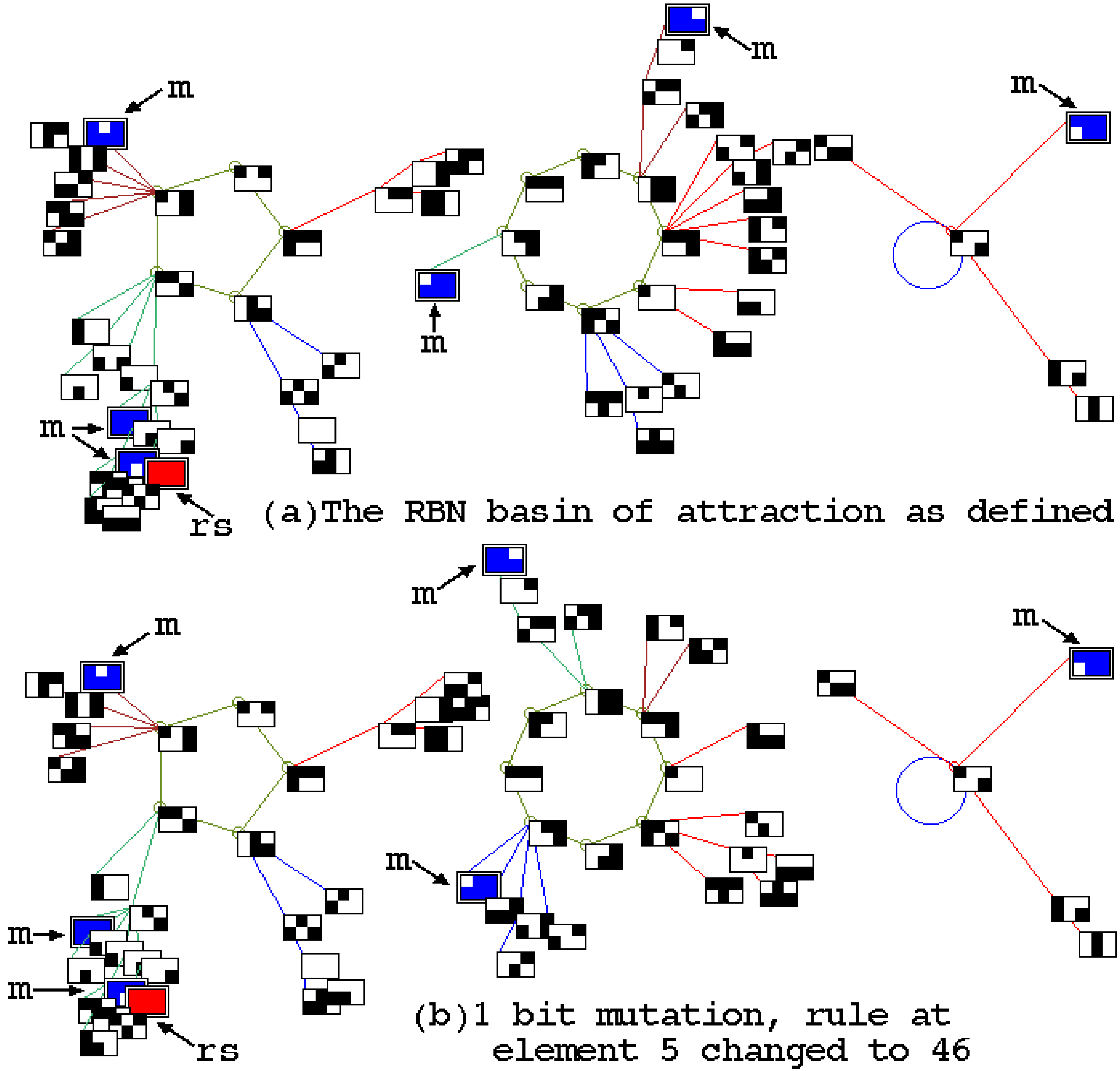


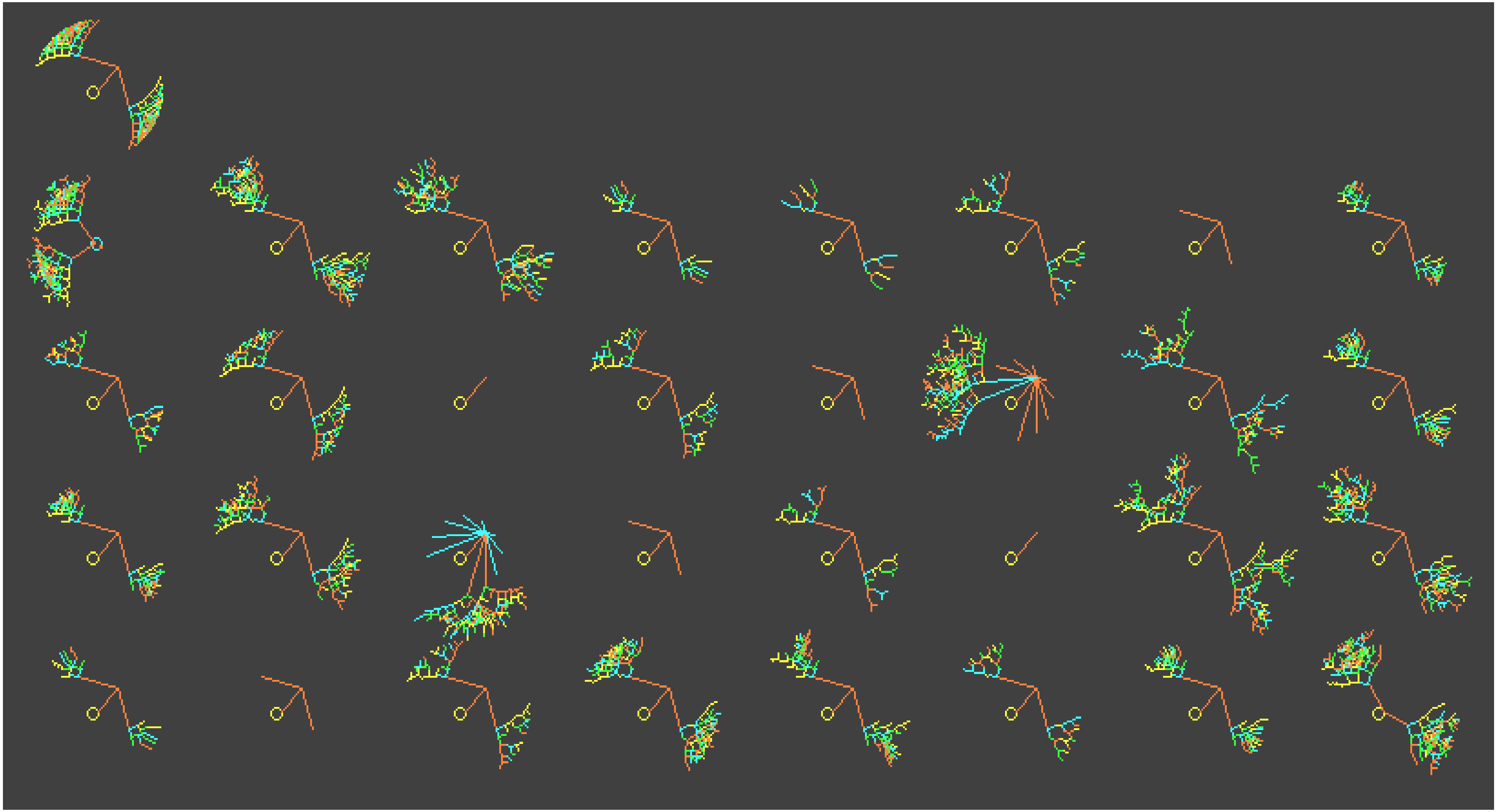
Figure 2: To decrypt, Starting from the encrypted state in figure 1, the CA with the same rule was run forward by 20 time-steps, the same number that was run backwards, to recover the original image or bit-string. This figure shows time-steps 17 to 25 to illustrate how the "alien" image was scrambled both before and after time-step 20.



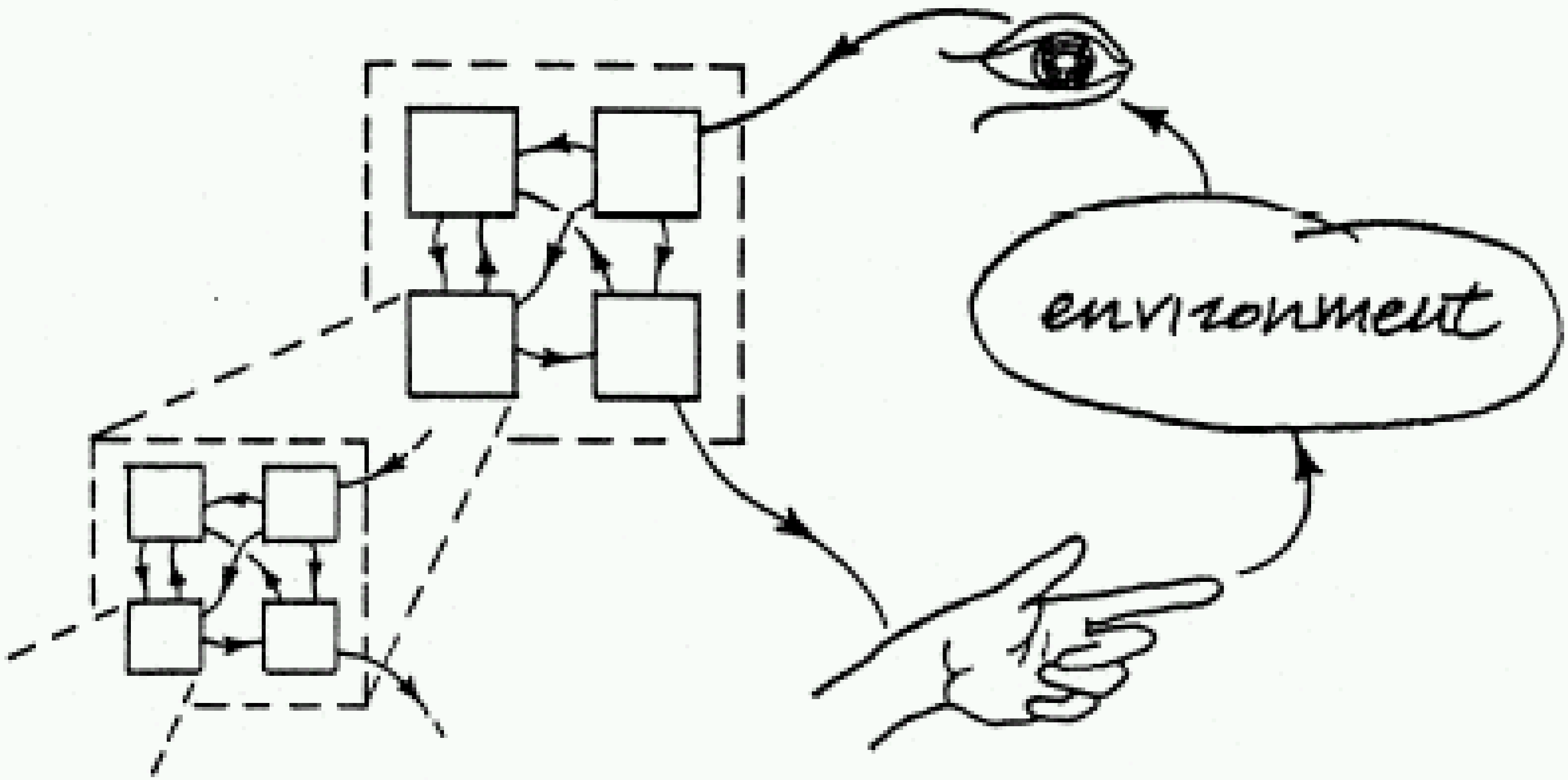
el.	wiring	rule
5	2,4,5	62
4	5,0,1	61
3	4,3,5	108
2	2,5,0	5
1	4,2,1	64
0	3,1,2	231

Two random Boolean network basin of attraction fields, with a 1 bit difference in one rule

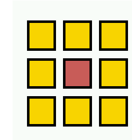
genotype - phenotype



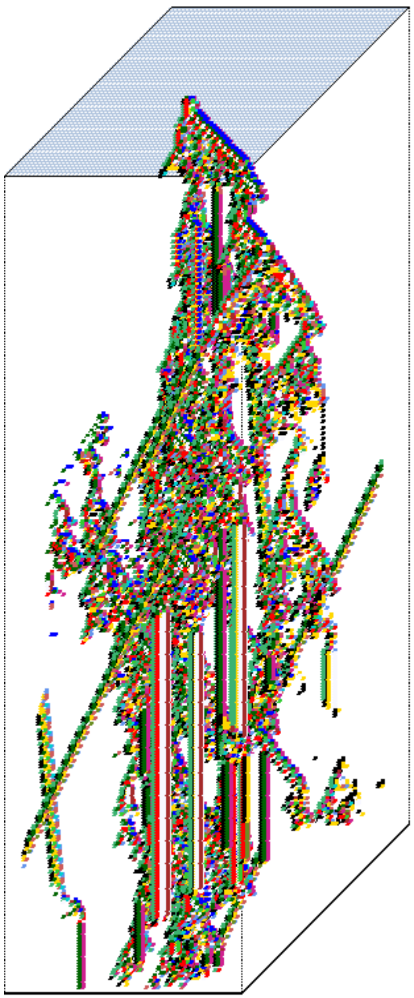
32 (1-bit) mutants of rule 60 (00111100)
mutations were made to the equivalent k=5 rule (00001111111100000000111111110000)
n=8, the basin seed is 00000000



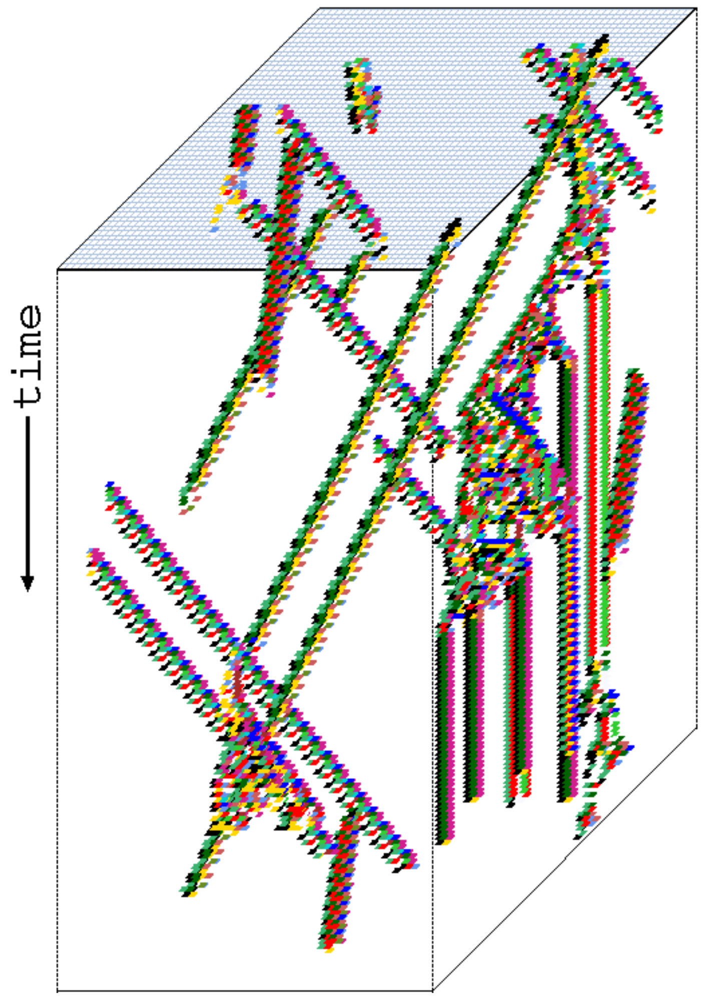
environment

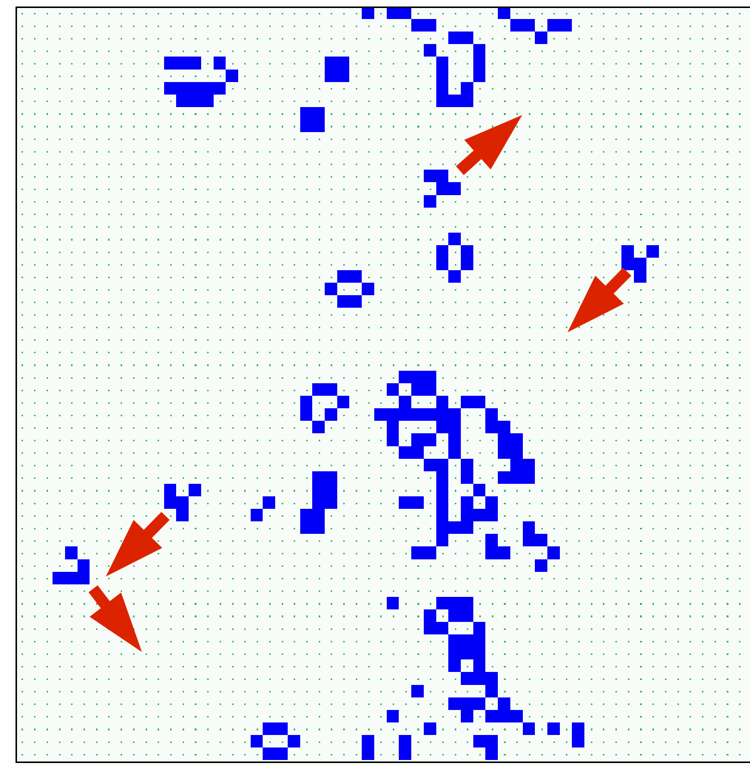
Conway's game-of-Life  *k*=8 outer-totalistic rule

Birth: 2 live neighbours, Survival: 2 or 3 live neighbours
otherwise: death by exposure or overcrowding

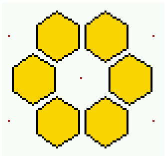


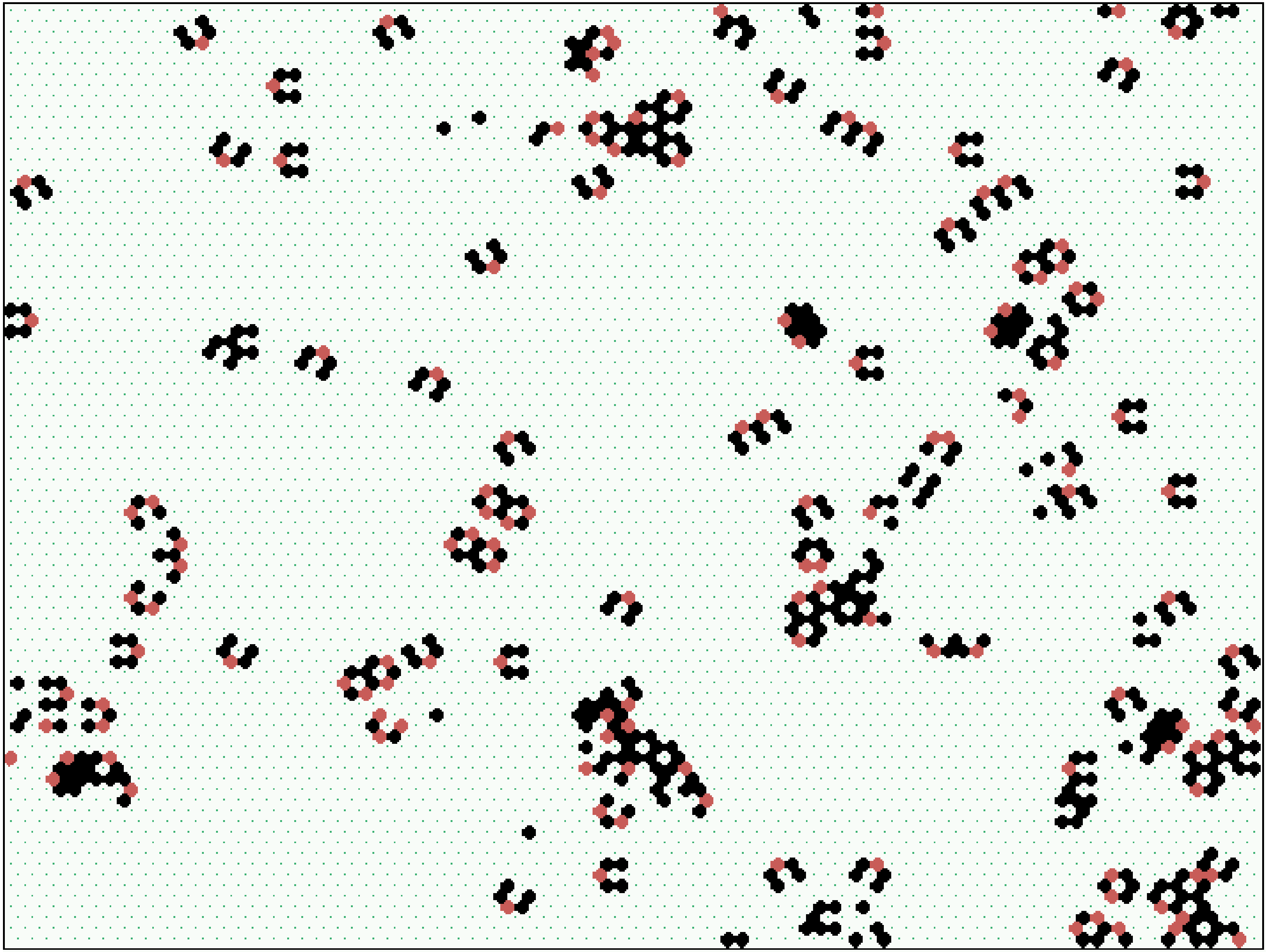
looking up into a cage



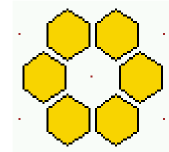


snaphot 60x60
gliders: red arrows

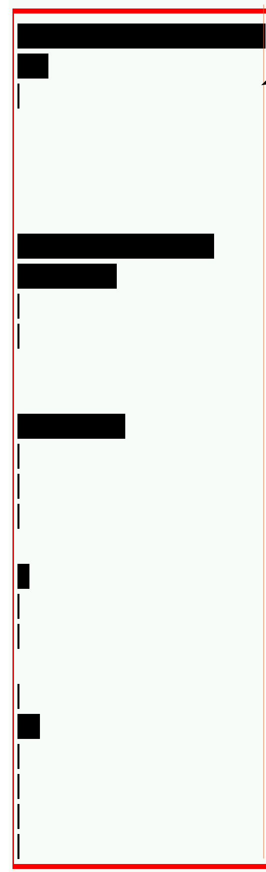
Beehive rule v=3, k=6  2D hex lattice



snapshot 88 x 88

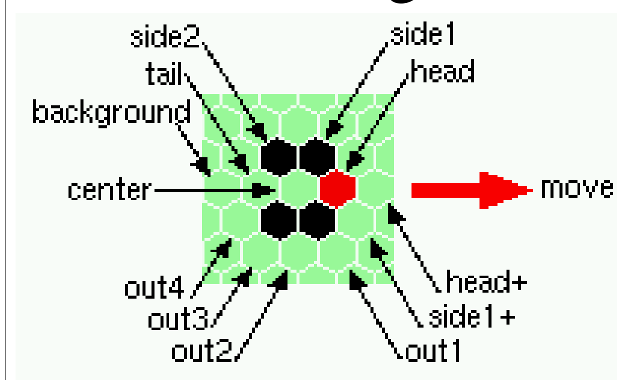
Beehive rule, v=3, k=6  2D hex lattice

```
     kcode = 0022000220022001122200021210
                         kcode index
                        /  totals: 2s+1s+0s=k=6
                       /  /             kcode
  basic               /  /             /
  glider             /  /             /  mutations
  ---------         / 2_1_0          /   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
```



The **lookup histogram** shows the frequency of neighbourhoods. This ties in with the sub-rules required to maintain the basic glider. 10 rarely if ever occur in an evolved system - they could be wildcards.

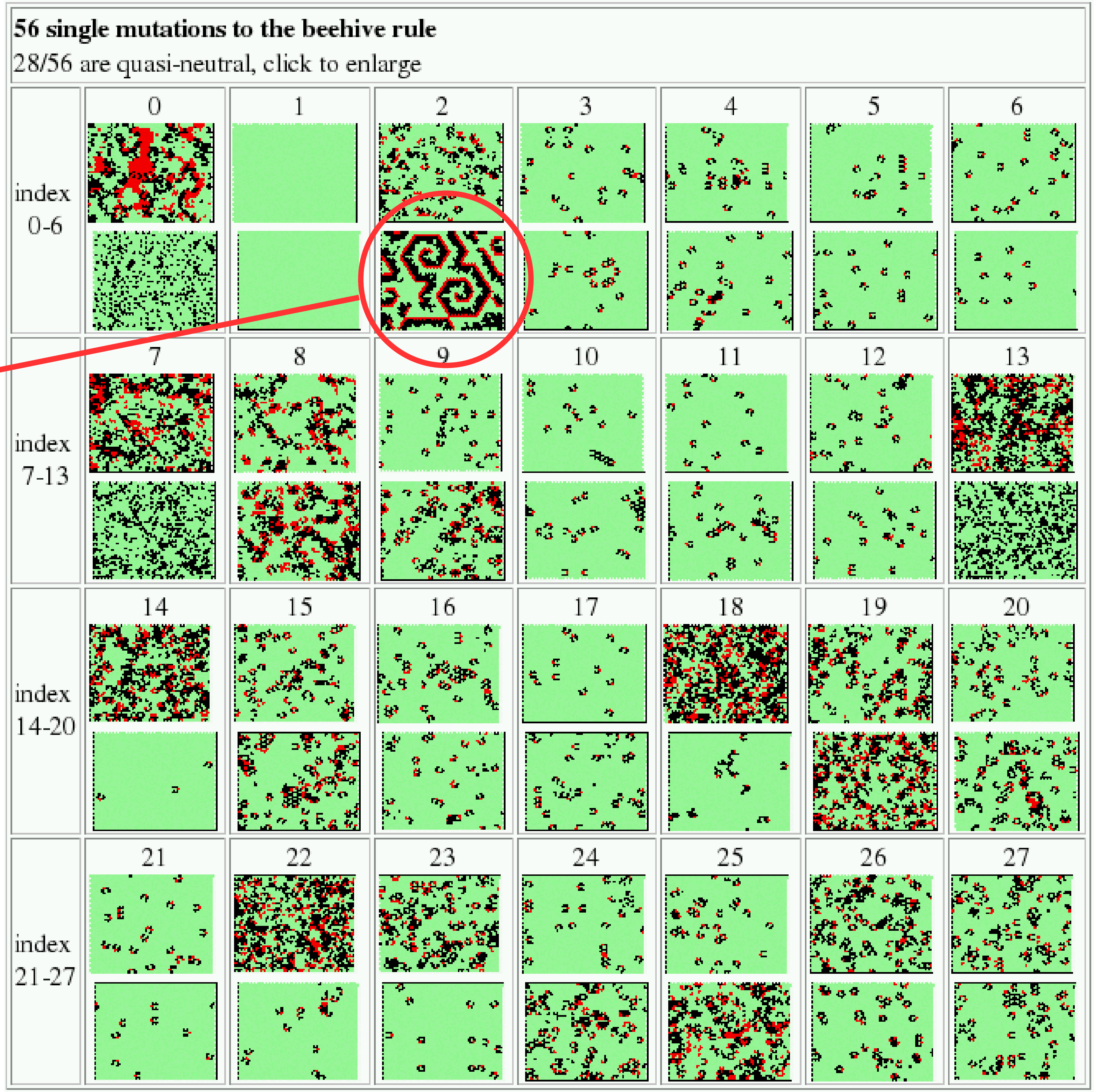
The basic glider



The size of the totalistic rule table = $(v + k - 1)! / (k! \times (v - 1)!)$

The rule expressed as an ij matrix, where i=2s and j=1s. 0s=6-(i+j)

$$
\begin{array}{c|l}
 & \quad\quad\quad j \\
\hline
 & 0\ 1\ 2\ 1\ 2\ 0\ 0 \\
 & 0\ 2\ 2\ 2\ 1\ 1 \\
i & 0\ 0\ 2\ 2\ 0 \\
 & 0\ 2\ 2\ 0 \\
 & 0\ 0\ 2 \\
 & 2\ 0 \\
 & 0
\end{array}
$$

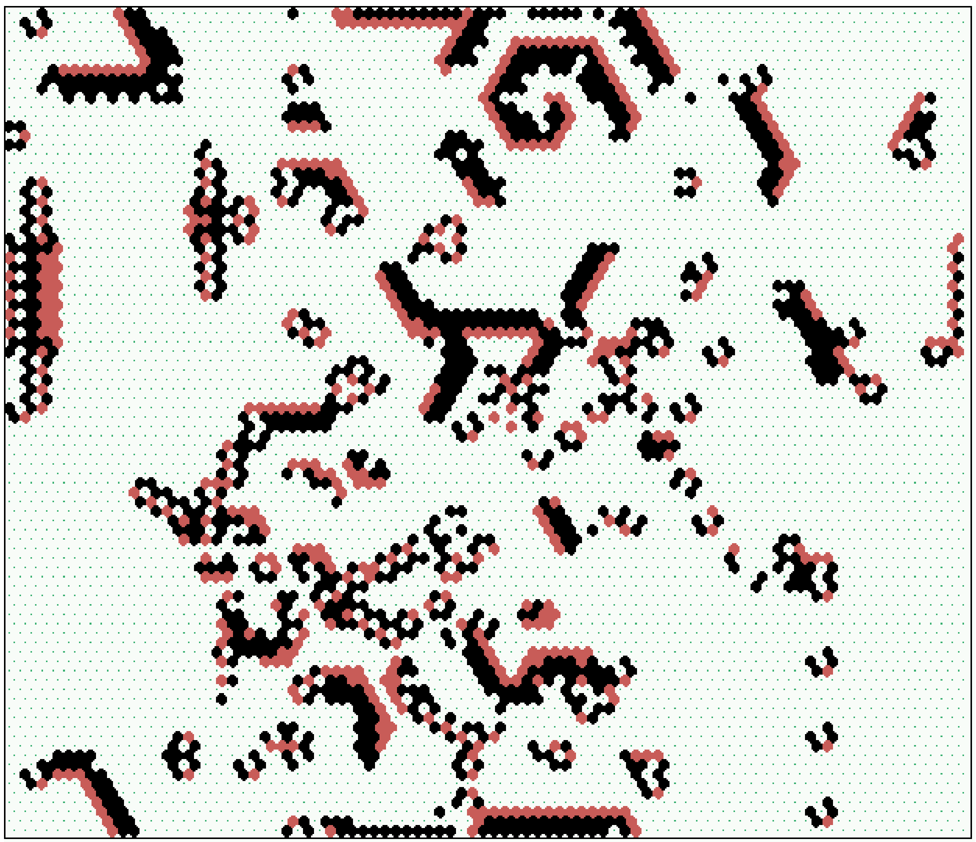


see
next slide

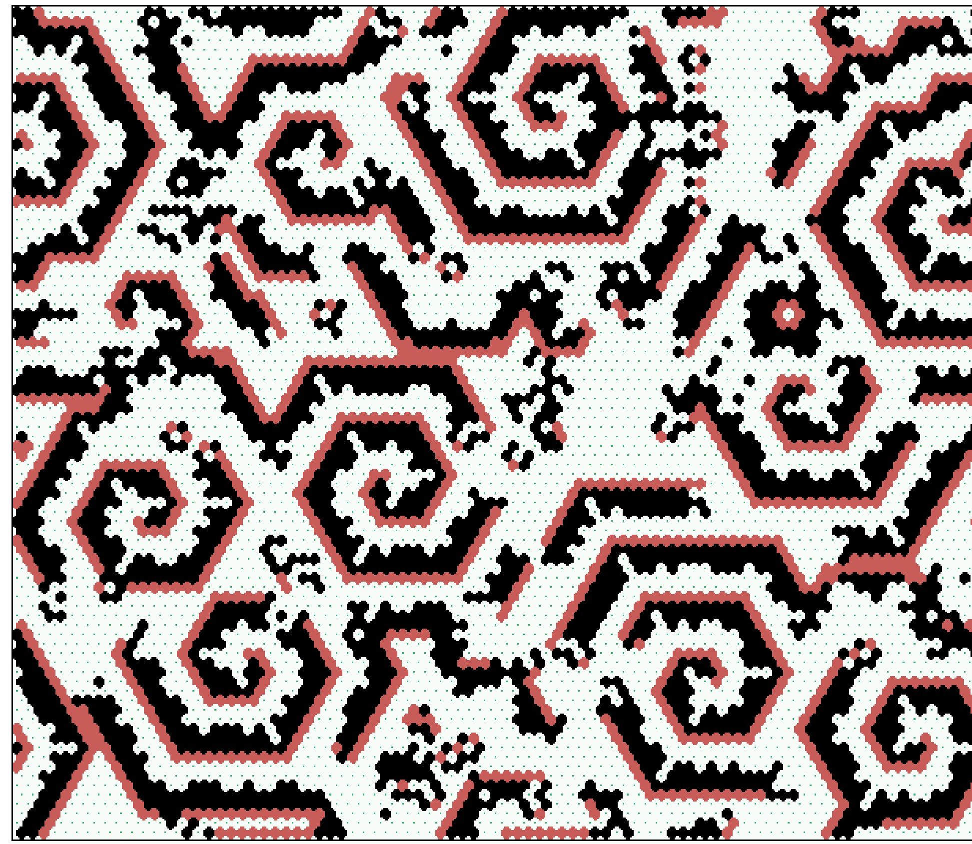
Beehive rule: a single mutation results in emerging spirals

0022000220022001122200021**1**10

was **2** in beehive rule



about 40 time-steps from
random initial state

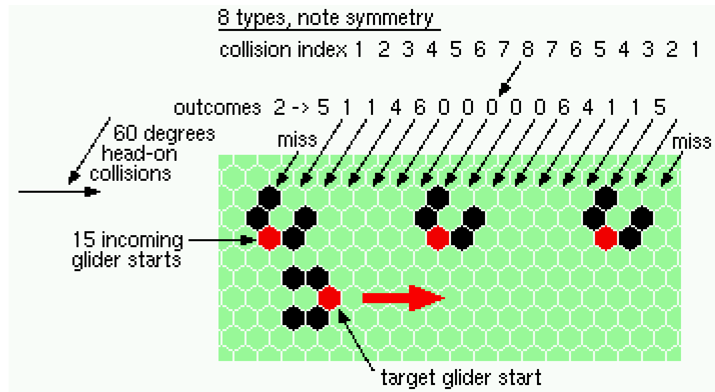


about +60 time-steps,
spirals stabilize

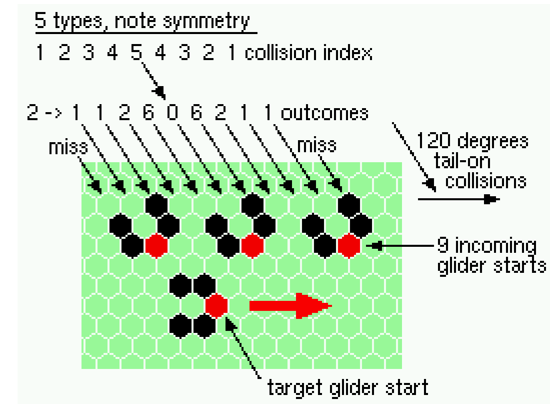
snapshots 88x88

Beehive rule, 21 types of collision between pairs of gliders

60º degree head-on

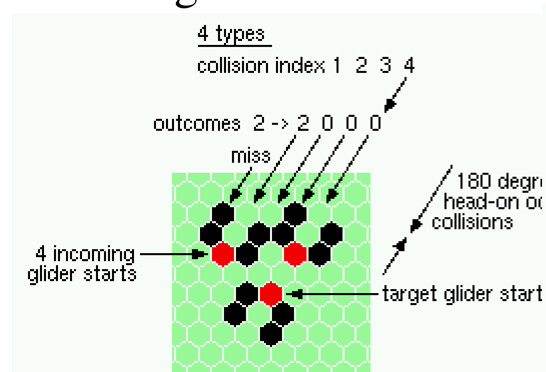


120º degree head-on

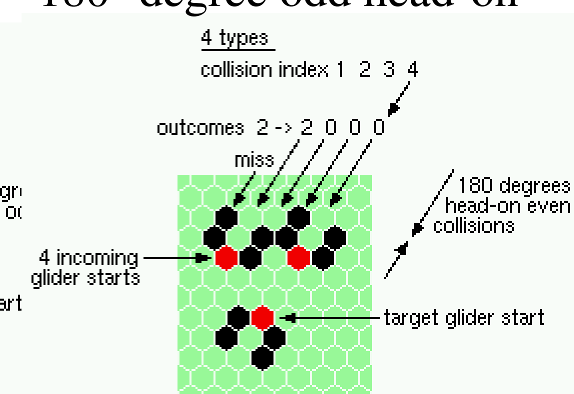


	no	type	no	before	after
oblige head-on:	*8*	*2->0*	*3*	*6*	*0*
		2->1	*2*	*4*	*2*
		2->4	*1*	*2*	*4*
		2->5	*1*	*2*	*5*
		2->6	*1*	*2*	*6*
oblique tail-on:	*5*	*2->0*	*1*	*2*	*0*
		2->1	*2*	*4*	*2*
		2->2	*1*	*2*	*2*
		2->6	*1*	*2*	*6*
head-on odd:	*4*	*2->0*	*3*	*6*	*0*
		2->2	*1*	*2*	*2*
head-on even:	*4*	*2->0*	*3*	*6*	*0*
		2->2	*1*	*2*	*2*
totals:	*21*		*21*	*42*	*31*

180º degree odd head-on



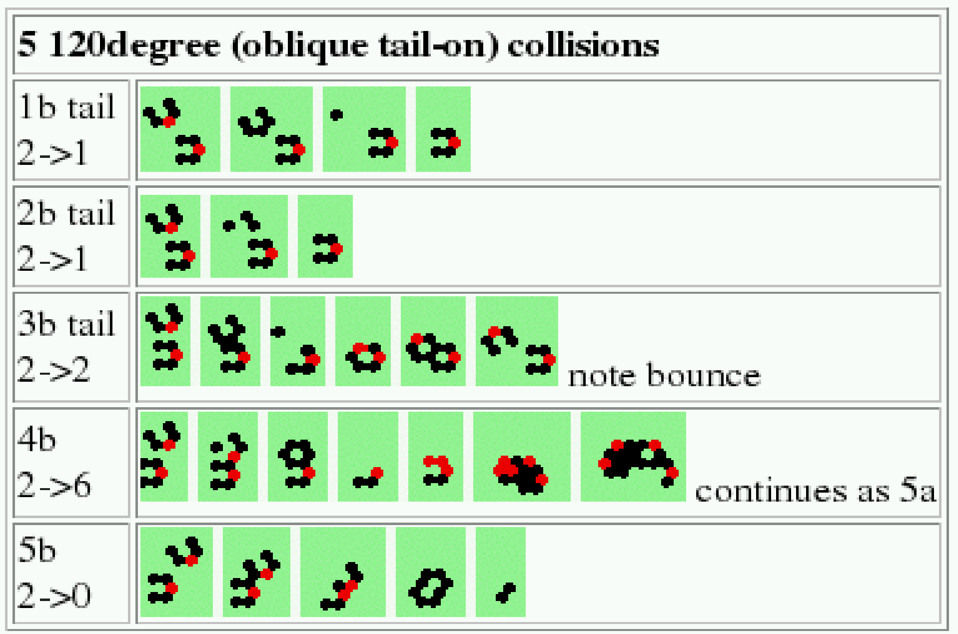
180º degree odd head-on



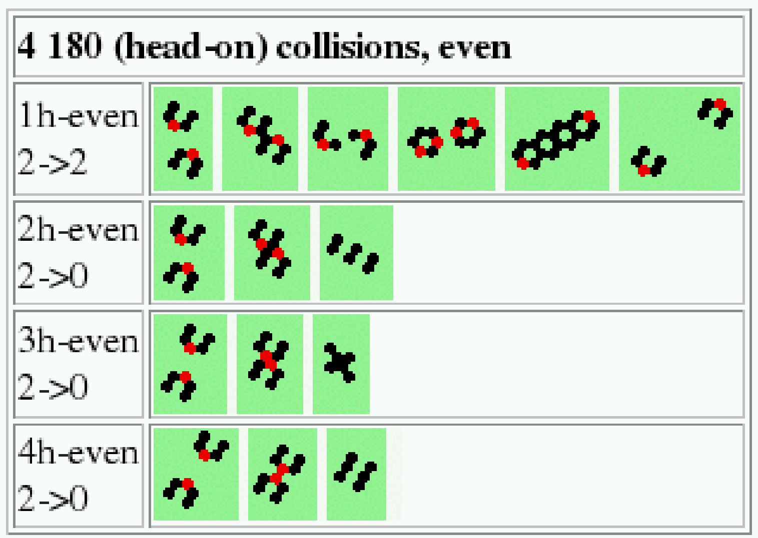
		gliders		
	type	*no*	*before*	*after*
self-destruction:	*2->0*	*10*	*20*	*0*
one-survivor:....	*2->1*	*4*	*8*	*4*
conservation:....	*2->2*	*3*	*6*	*6*
self-reproduction:	*2->4*	*1*	*2*	*4*
	2->5	*1*	*2*	*5*
	2->6	*2*	*4*	*12*
	totals	*21*	*42*	*31*

Beehive rule, 60º head-on collisions

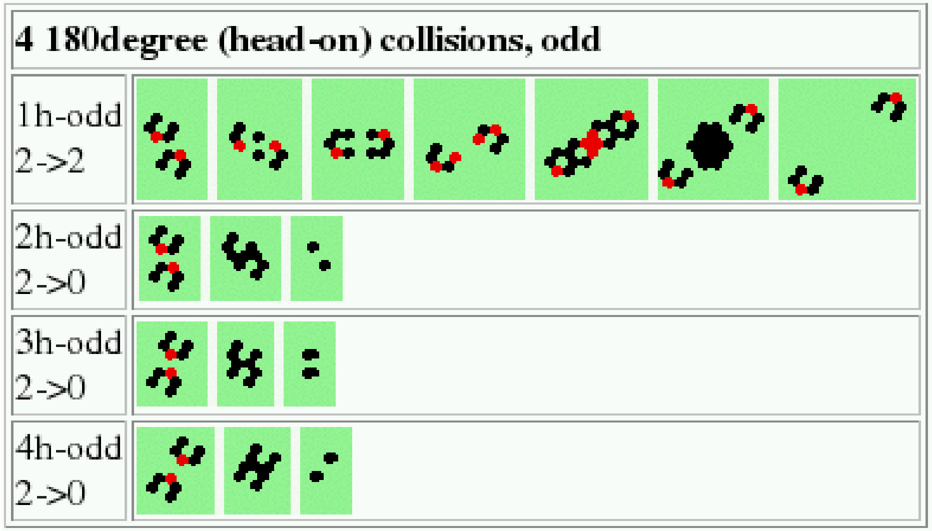
Beehive rule, 120° tail-on, 180° head-on collisions



5 120degree (oblique tail-on) collisions	
1b tail 2->1	
2b tail 2->1	
3b tail 2->2	note bounce
4b 2->6	continues as 5a
5b 2->0	

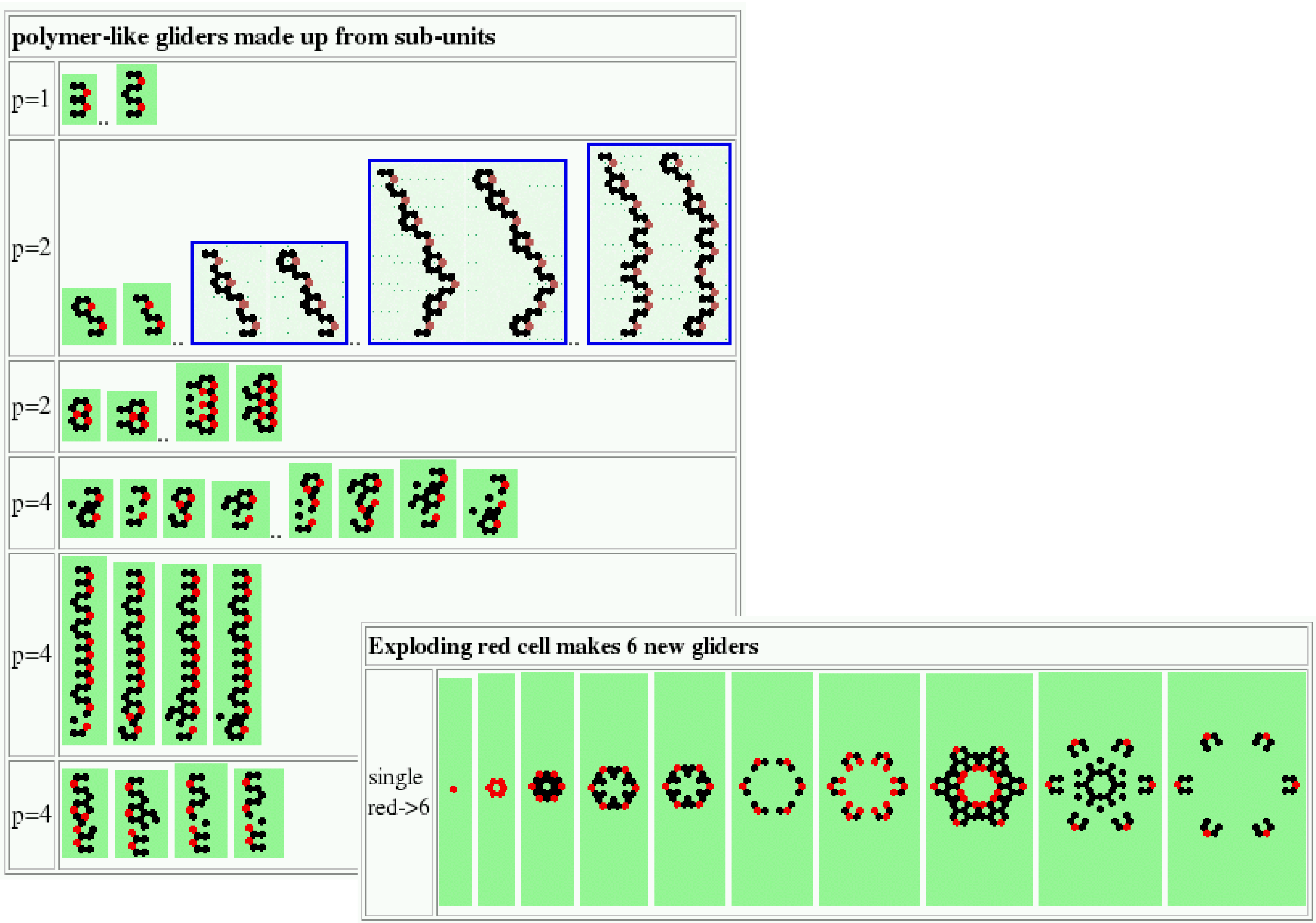


4 180 (head-on) collisions, even	
1h-even 2->2	
2h-even 2->0	
3h-even 2->0	
4h-even 2->0	



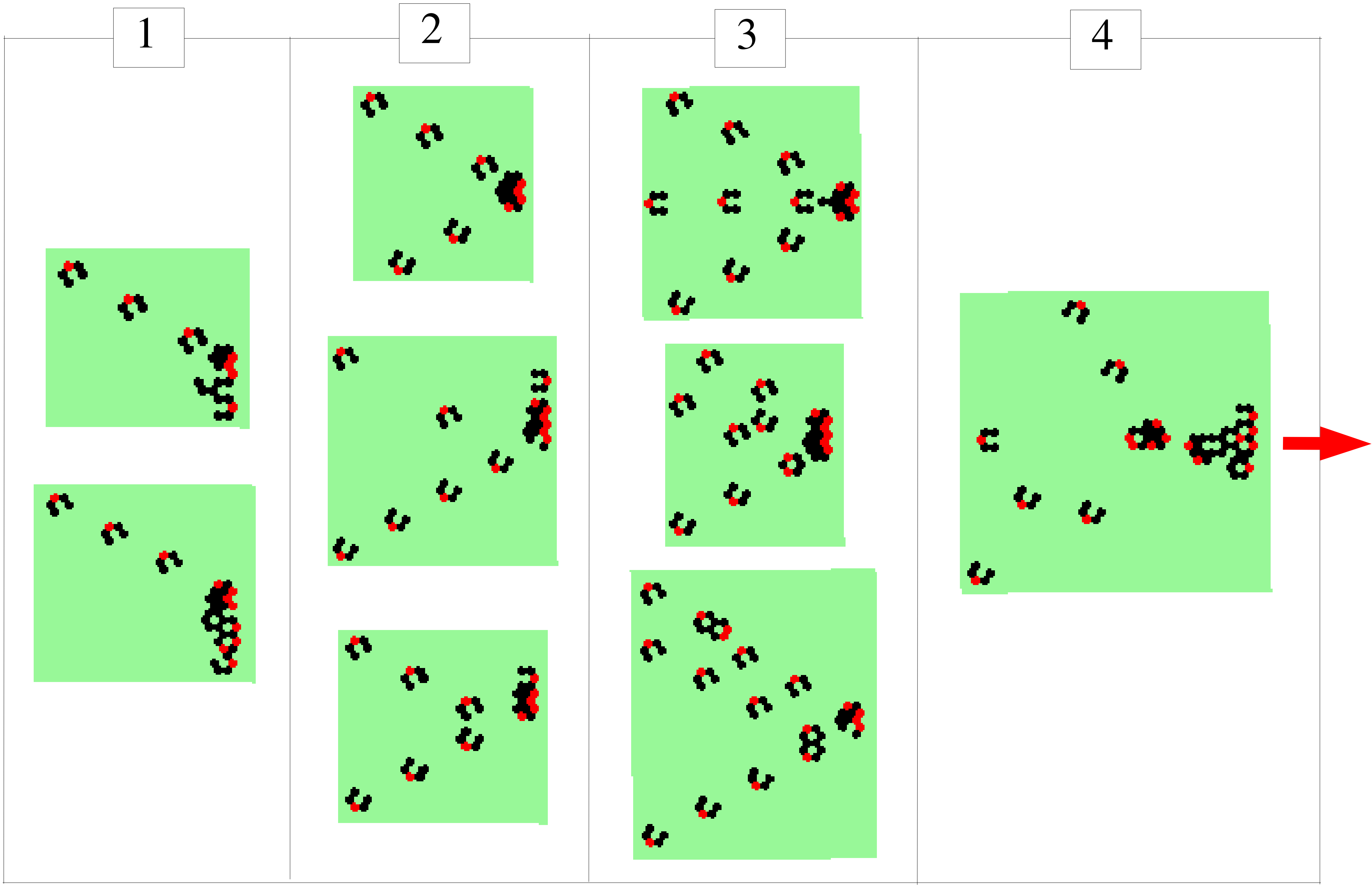
4 180degree (head-on) collisions, odd	
1h-odd 2->2	
2h-odd 2->0	
3h-odd 2->0	
4h-odd 2->0	

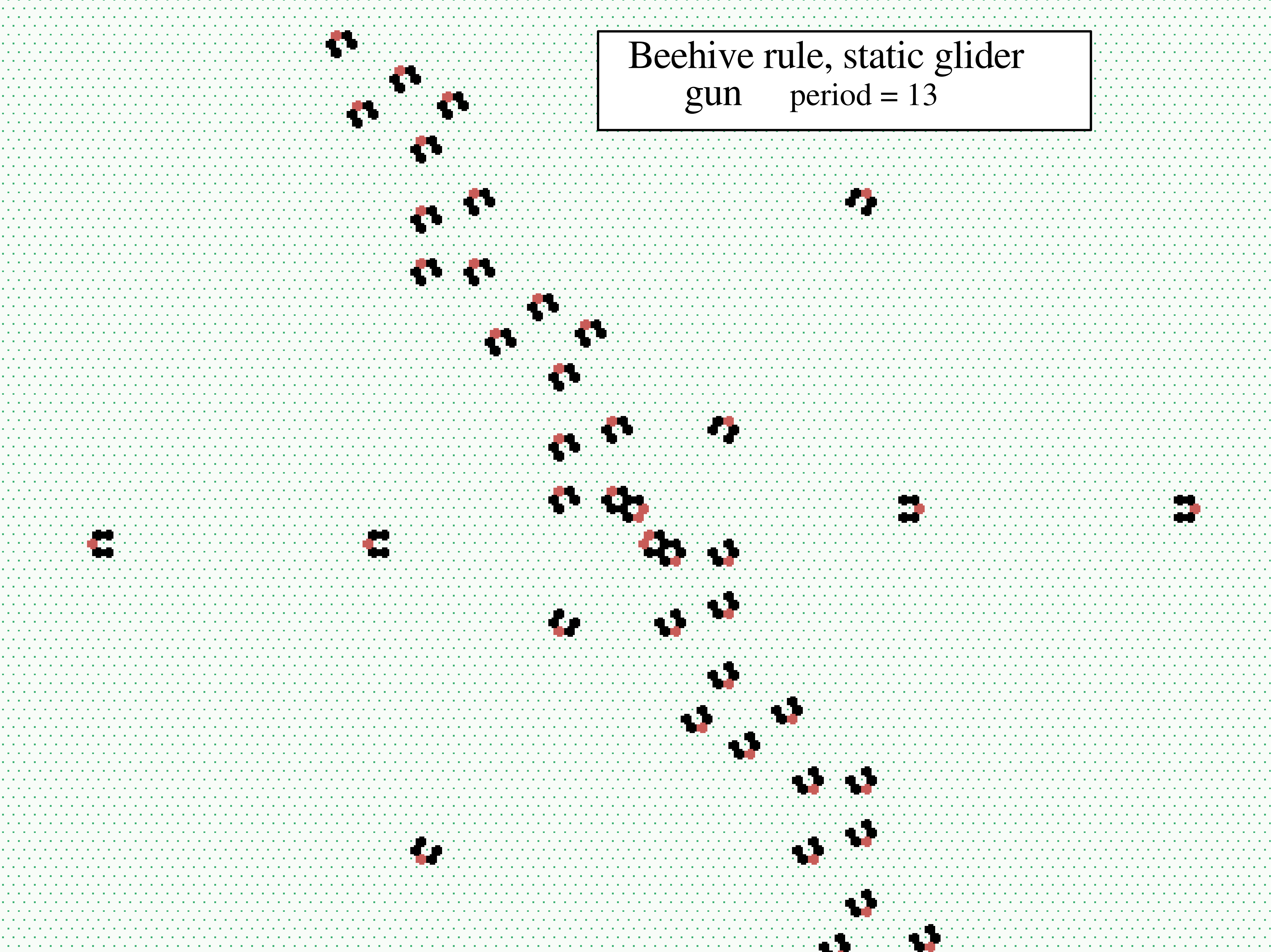
Beehive rule, polymer gliders, exploding red cell

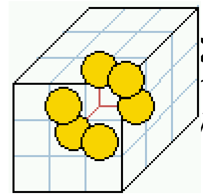


Beehive rule, mobile glider guns

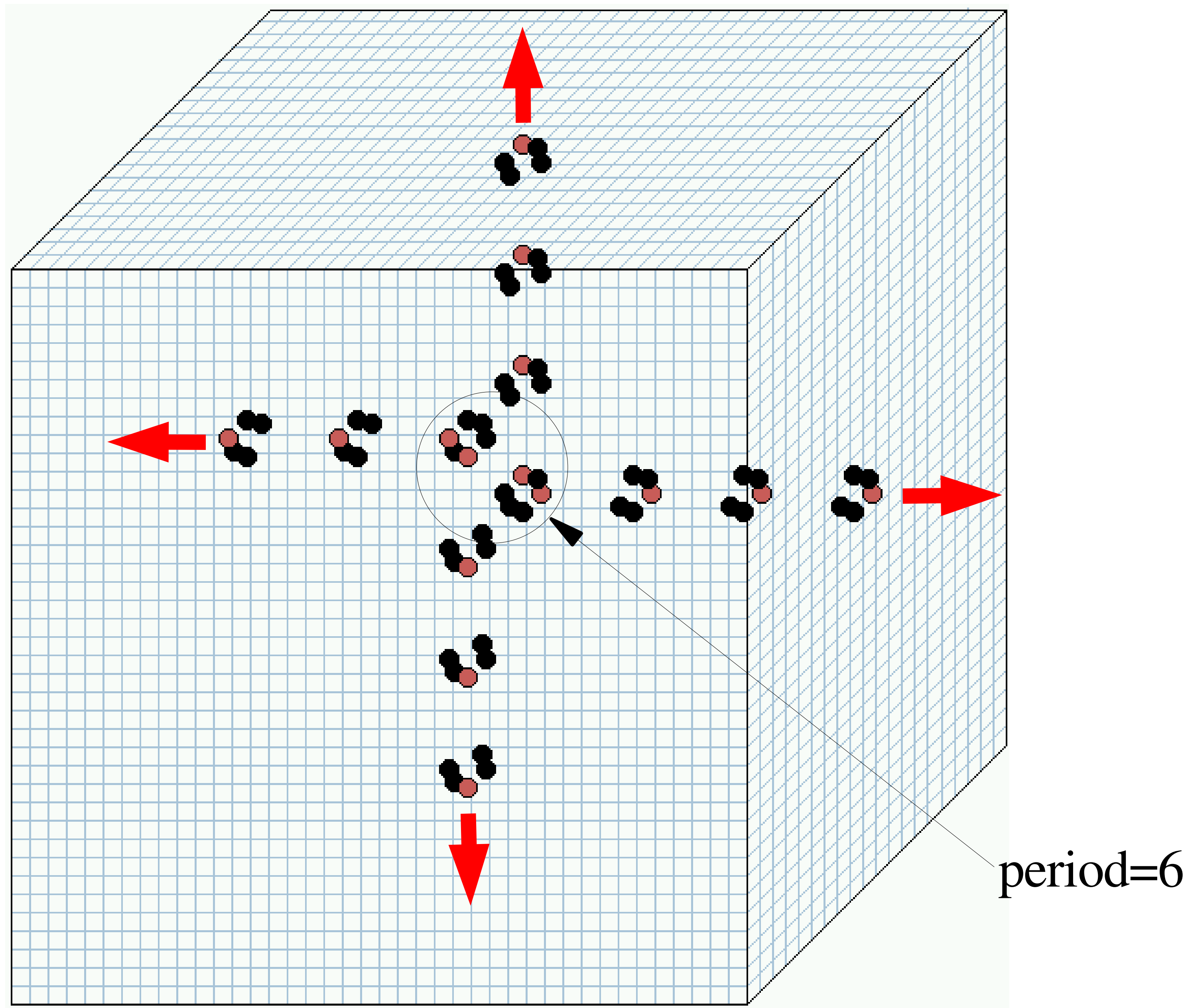
heads move East, shedding 1 to 4 glider streams

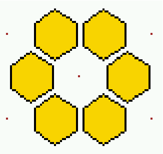


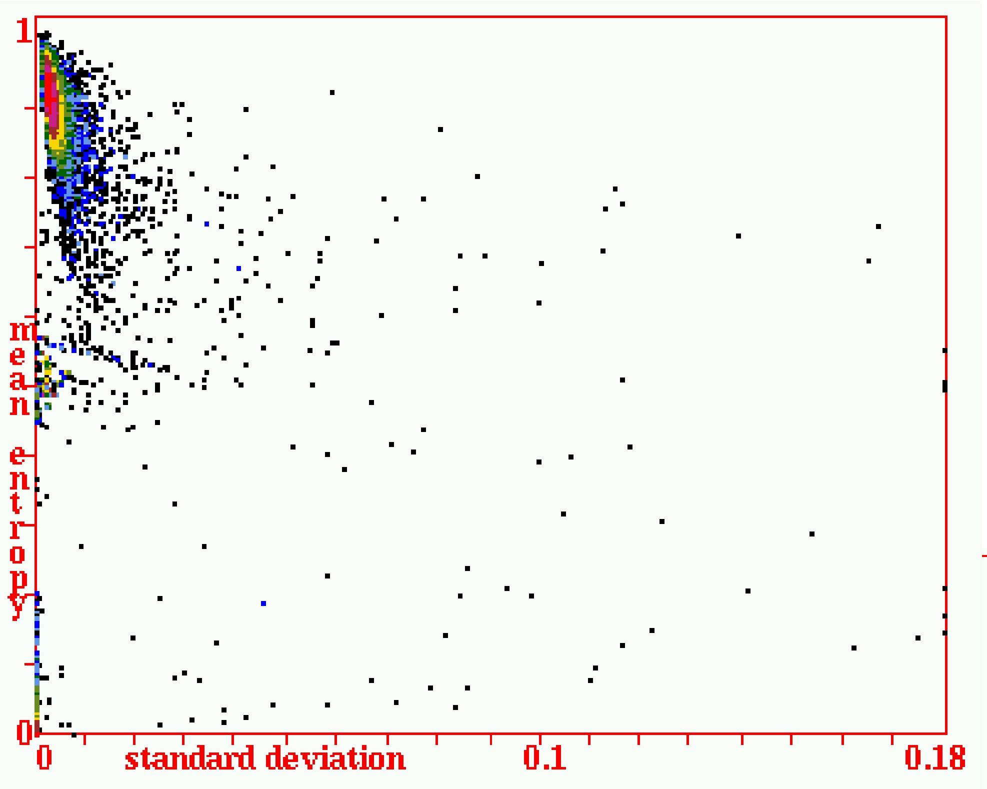

Beehive rule, static glider gun period = 13

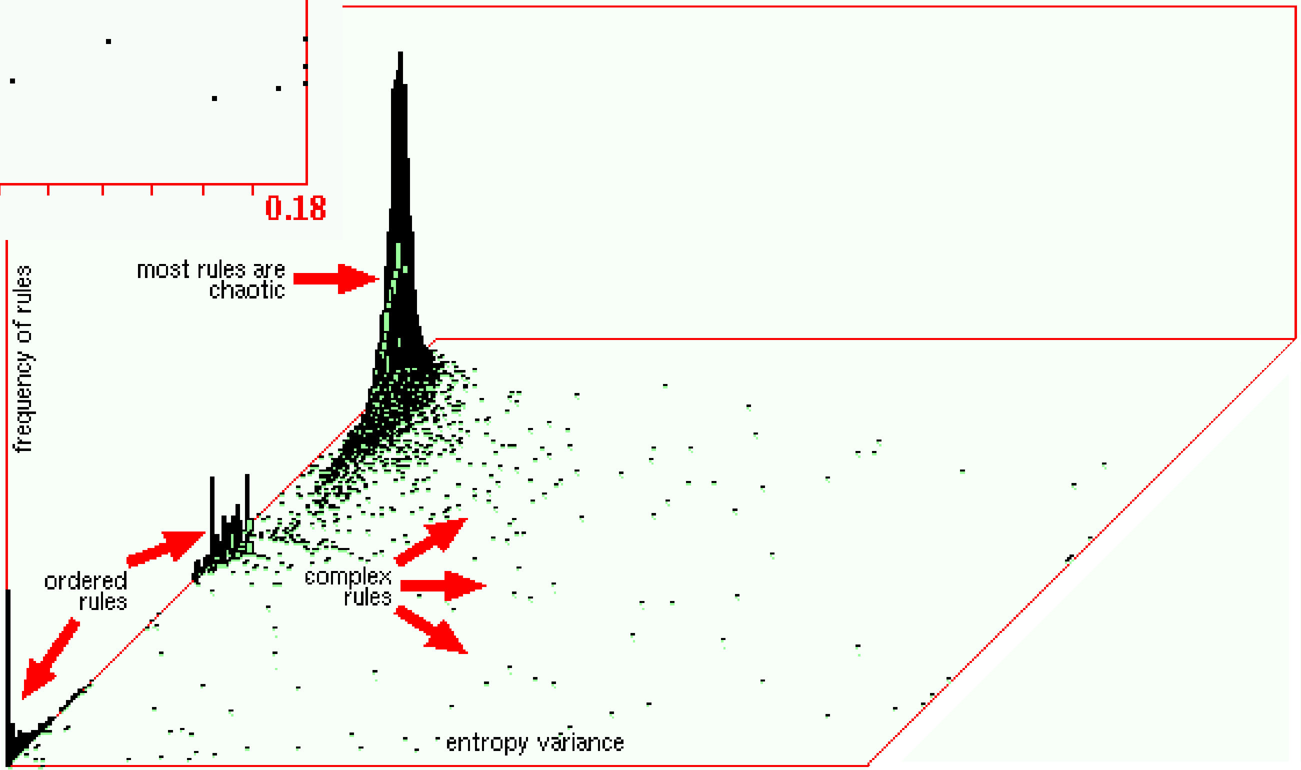
Beehive rule, v=3, k=6,  3D glider gun, cubic lattice

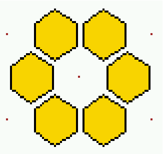
shooting 4 glider streams view: looking down into a box, 40 x 40 x 20



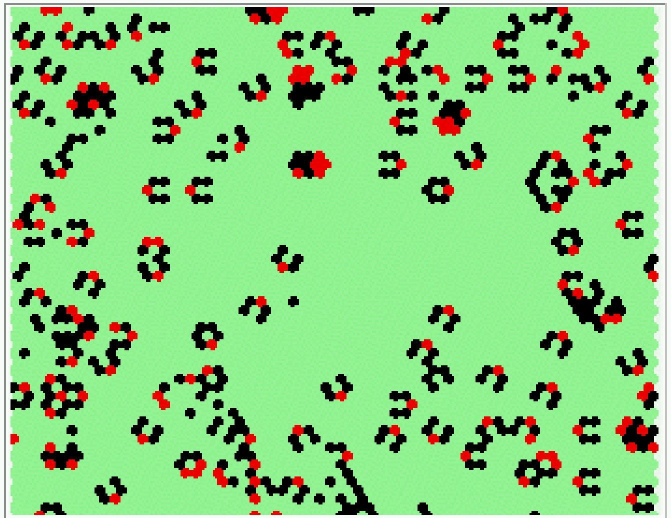
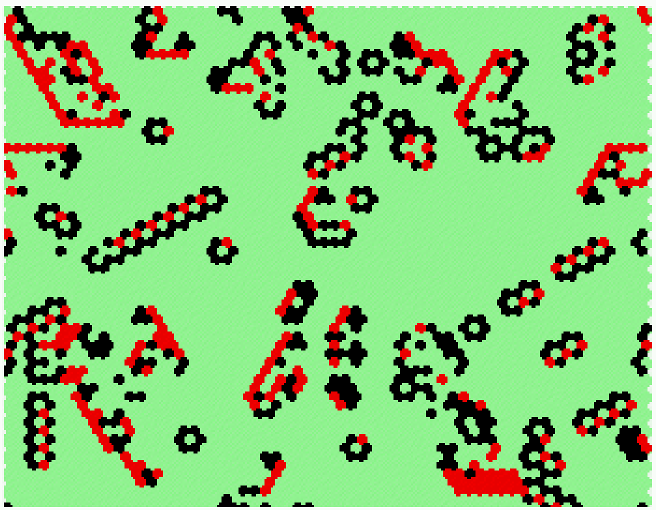
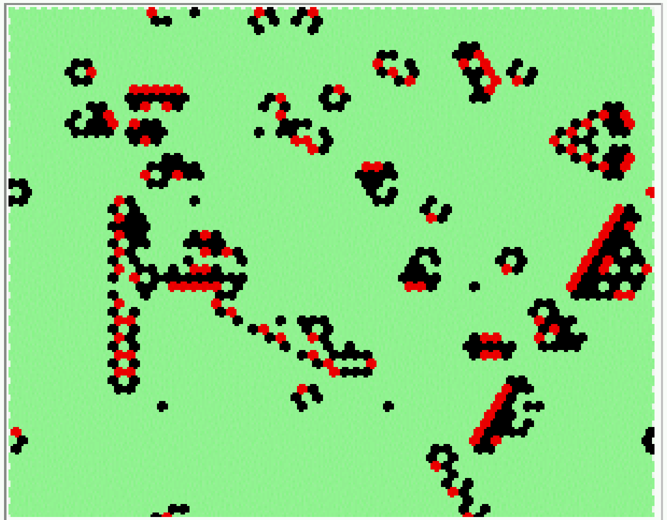
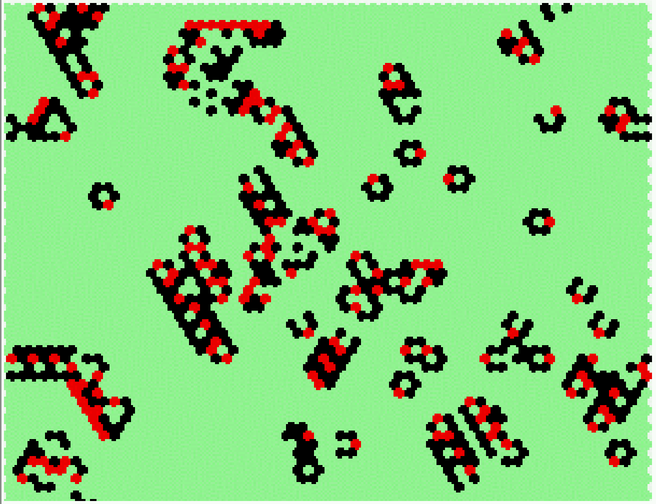
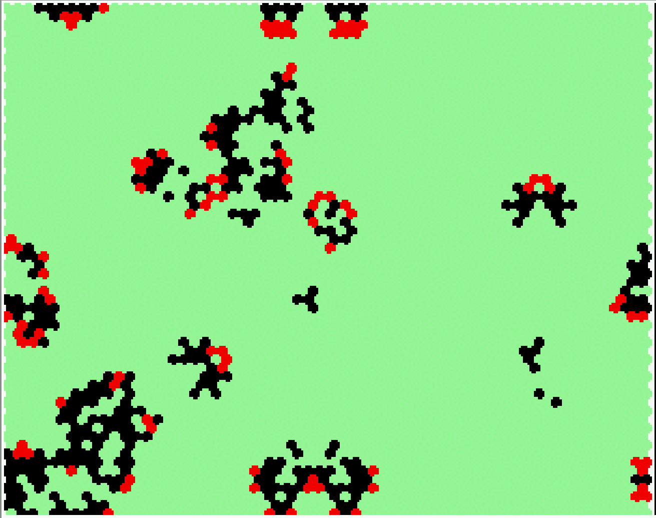
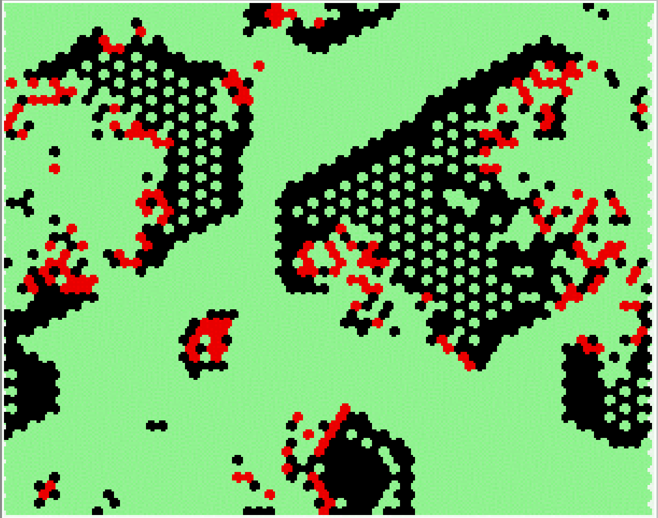
Classifying random samples of v=3, k=6  2D CA, automatically

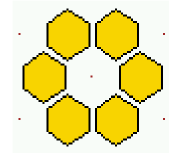


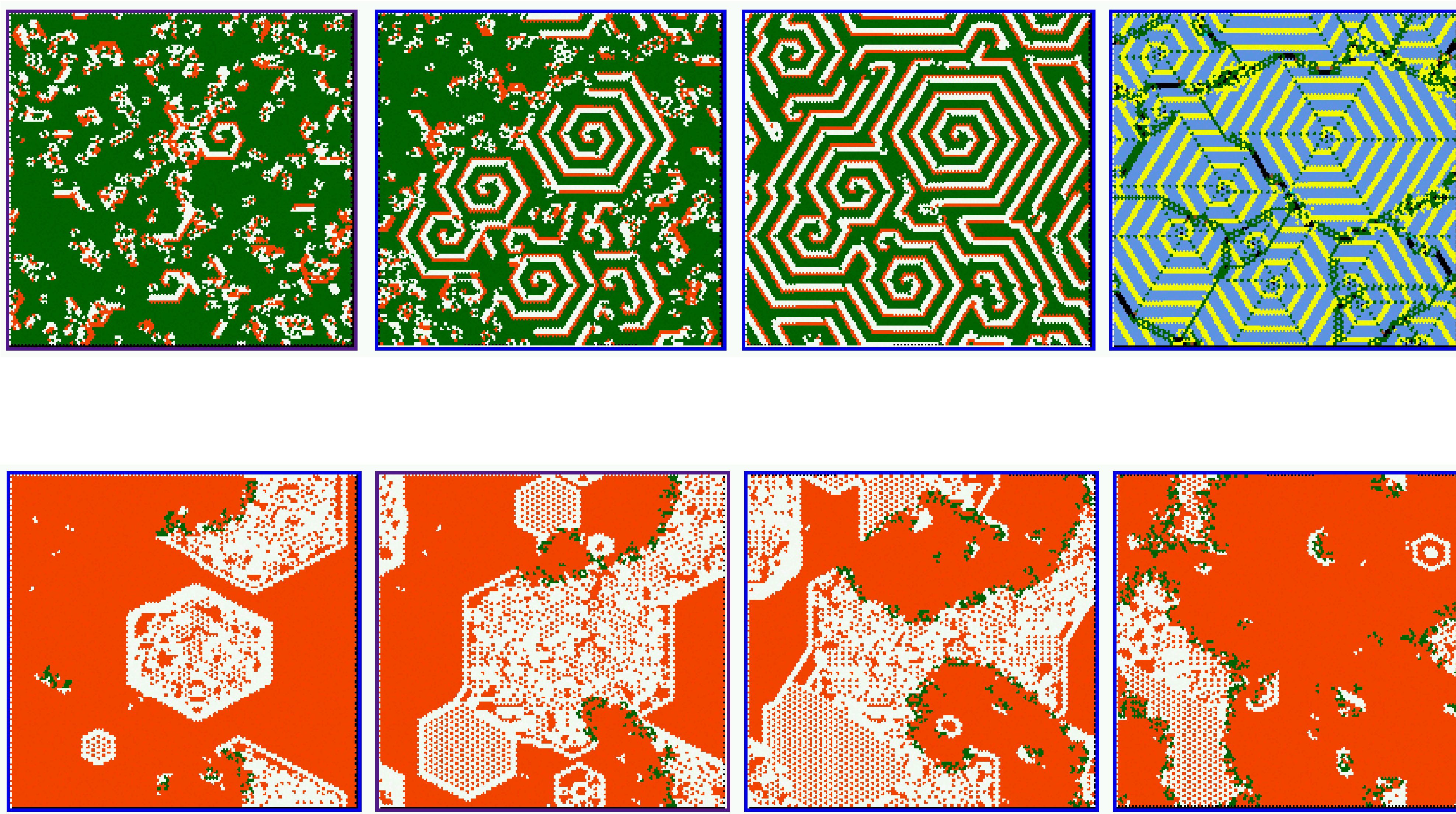


Examples: v=3, k=6  2D CA

Examples of other complex-rules

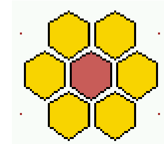
 kcode=2200021000222201110201212210 0=10 2=11 1=7	 kcode=0222200220000200100201102110 0=14 2=9 1=5	 kcode=0220002120022202120200112110 0=11 2=11 1=6
 kcode= 0200001120100200002200120110 0=15 2=6 1=6	 kcode= 0200202022222200012100002100 0=14 2=9 1=3	 kcode=0122120102200122010000102000 0=14 2=8 1=6

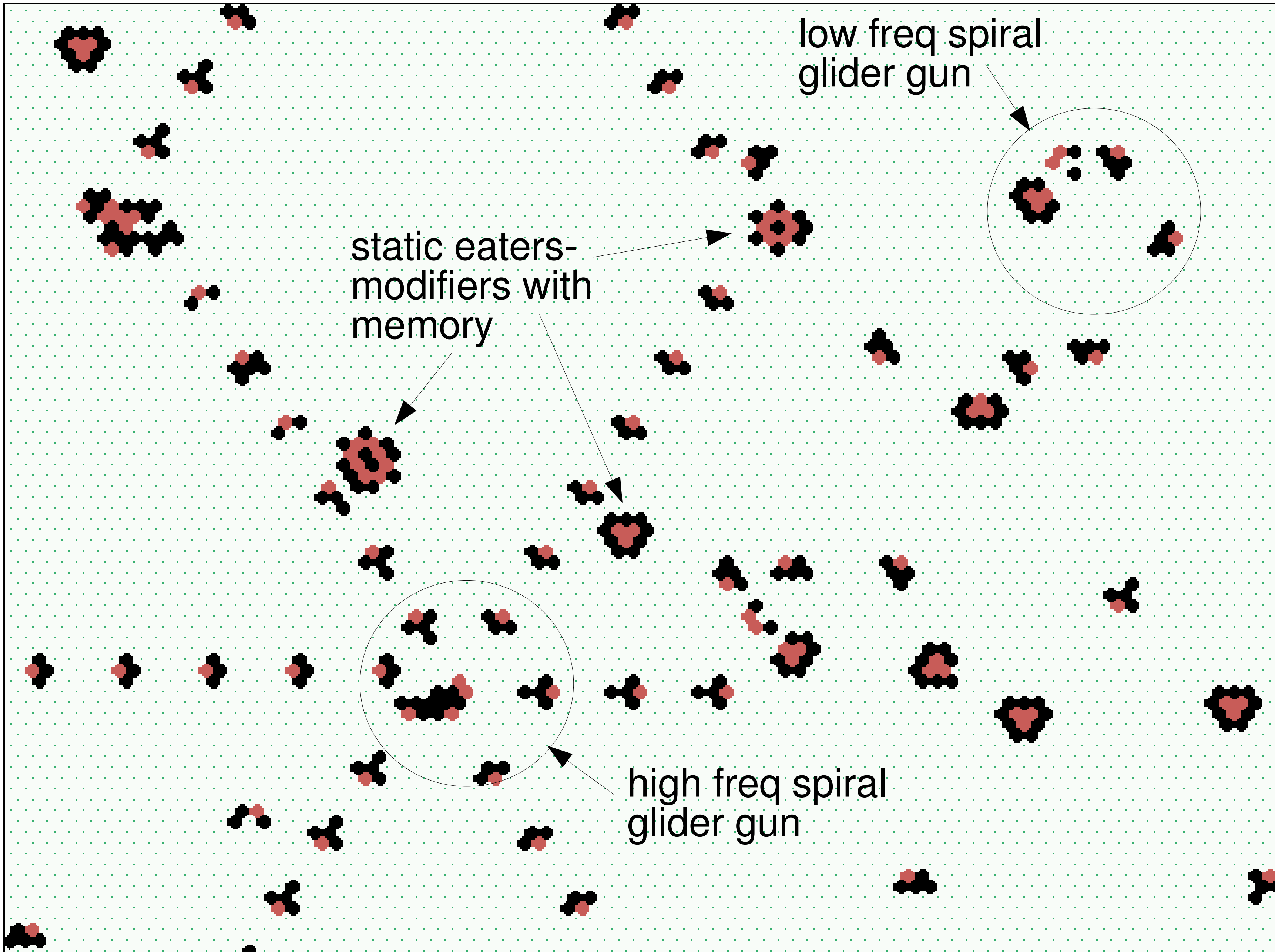
Examples: v=3, k=6  2D CA



Spiral rule, v=3, k=7 2D hex lattice (88 x 88)

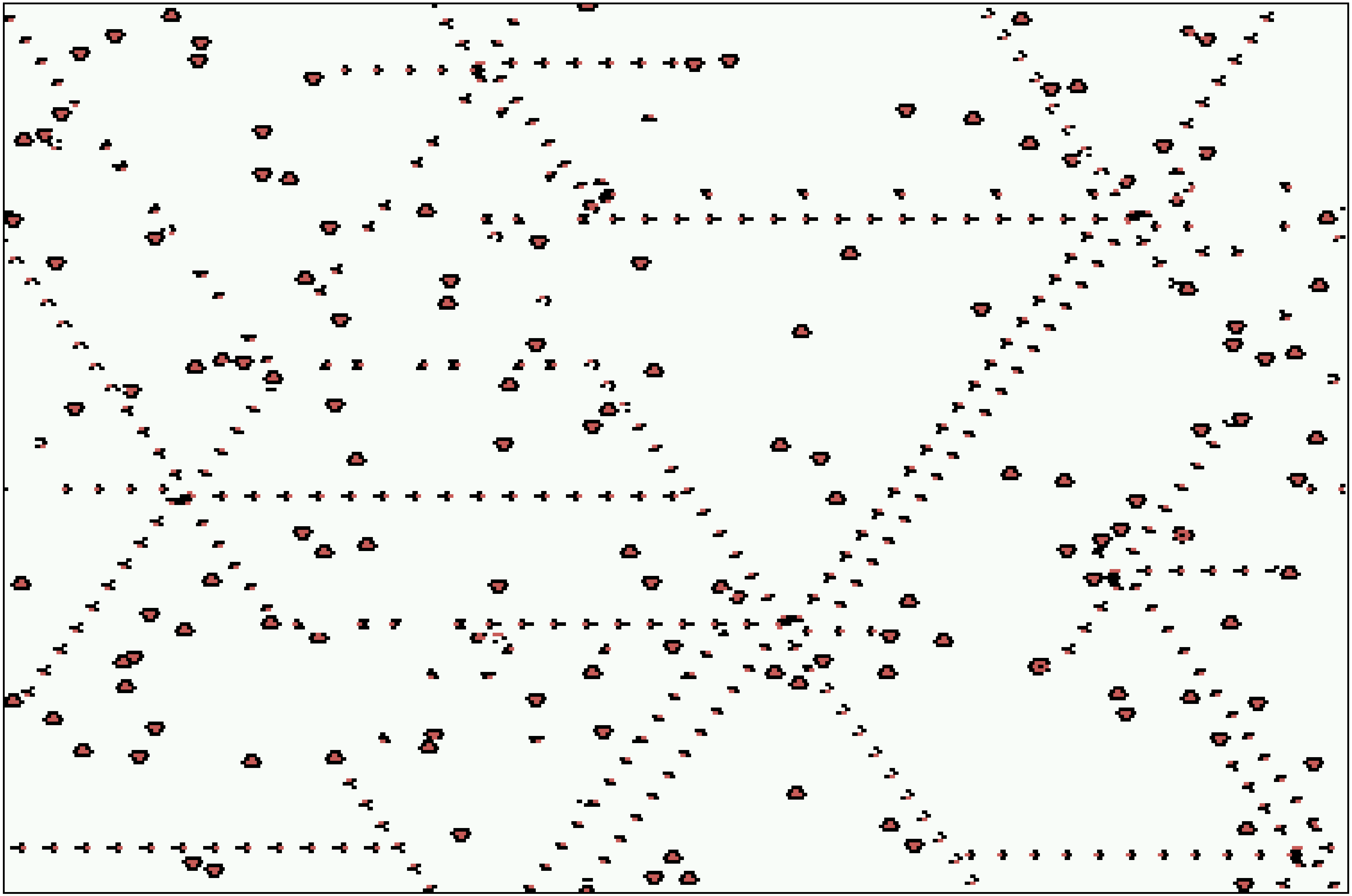
with Andy Adamatzky
UWE Bristol





Spiral rule: emergent circuits

About 400 time-steps from a random initial state (250x250). Large scale quasi-stable circuits have emerged, but lower level interactions and rhythms continue.



Spiral rule: gliders

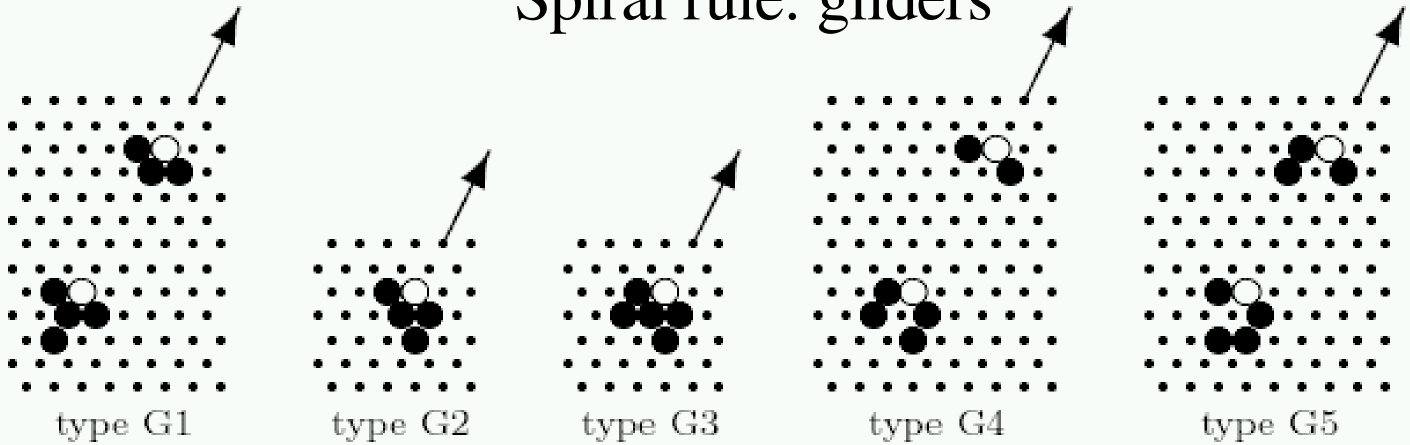


FIGURE 4
Basic gliders
5 types of small glider (G1-G5) emerge in the spiral rule. They have a white head (value 1) and a varying black tail (value 2). Movement is in the direction of the white head in any of 12 directions on the hexagonal lattice; in these examples the heading is North East. Types 1, 4 and 5 have period 2. Asymetric gliders, types 2 and 5 can be handed. High and low frequency spiral glider-guns produce types 1 and 2 respectively.

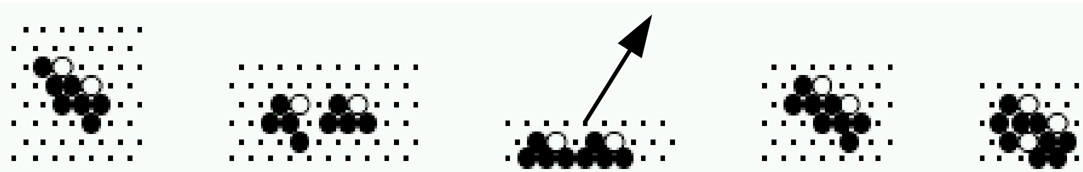
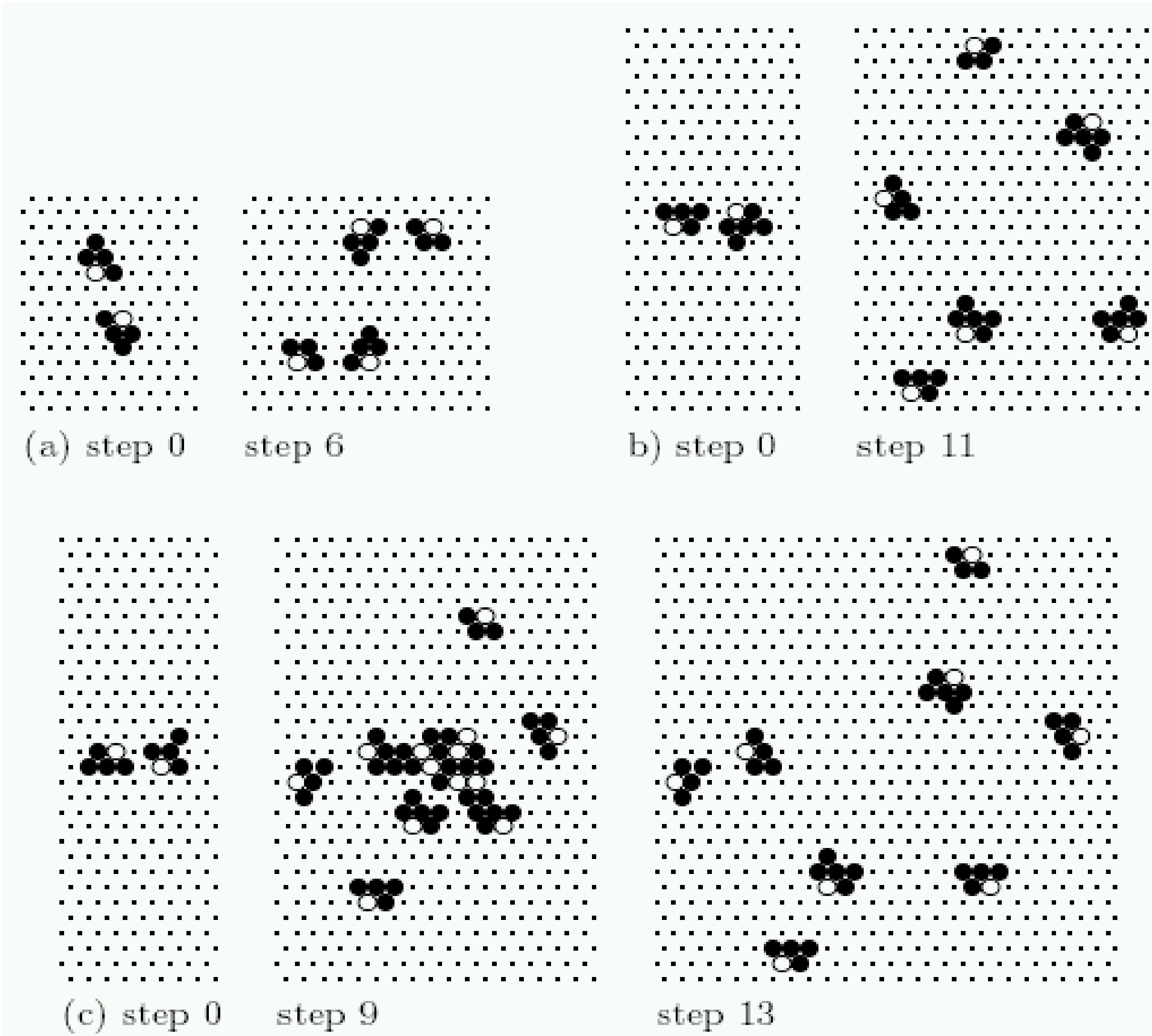
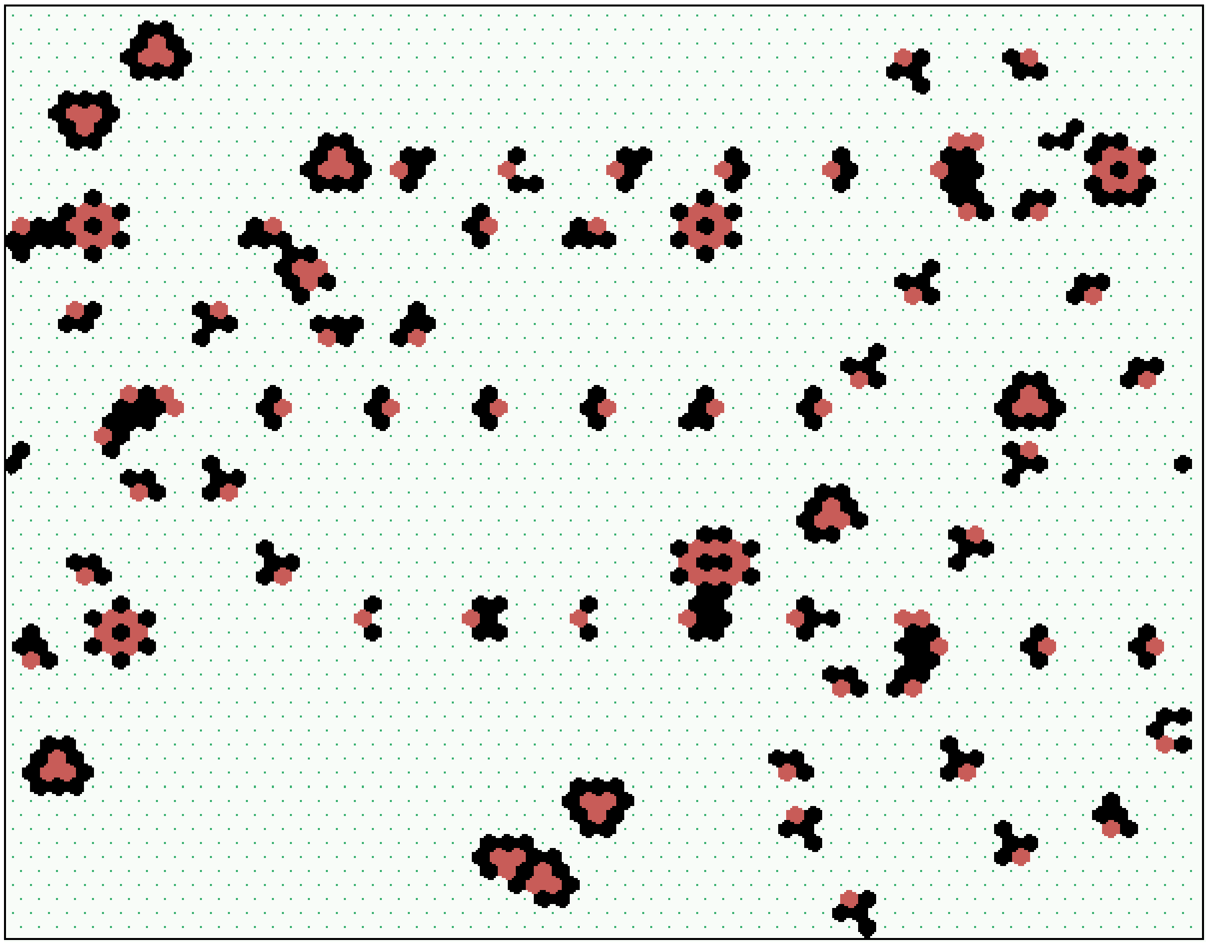


FIGURE 5
Small gliders
Various types of larger gliders, all heading North East. Basic gliders and small gliders can combine to form polymer gliders in many combinations.

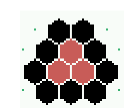
Spiral rule: reproduction by pairwise glider collisions



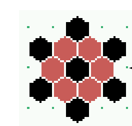
Spiral rule: static structures interact with gliders-guns to create quasi-stable circuits



static structures can destroy gliderse (eaters) or modify gliders as they brush past. The type SS2 glider has memory on its skin.



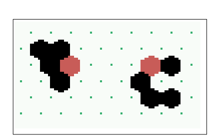
type SS1



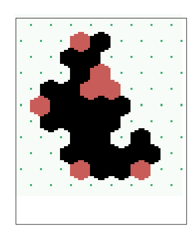
type SS2 – its skin has memory

both can link up into chains

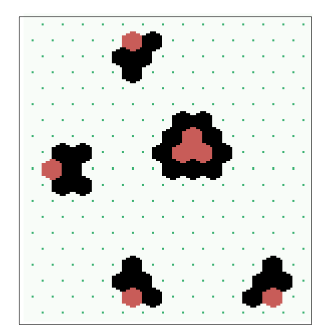
Two gliders collide to make a static structure



step 0



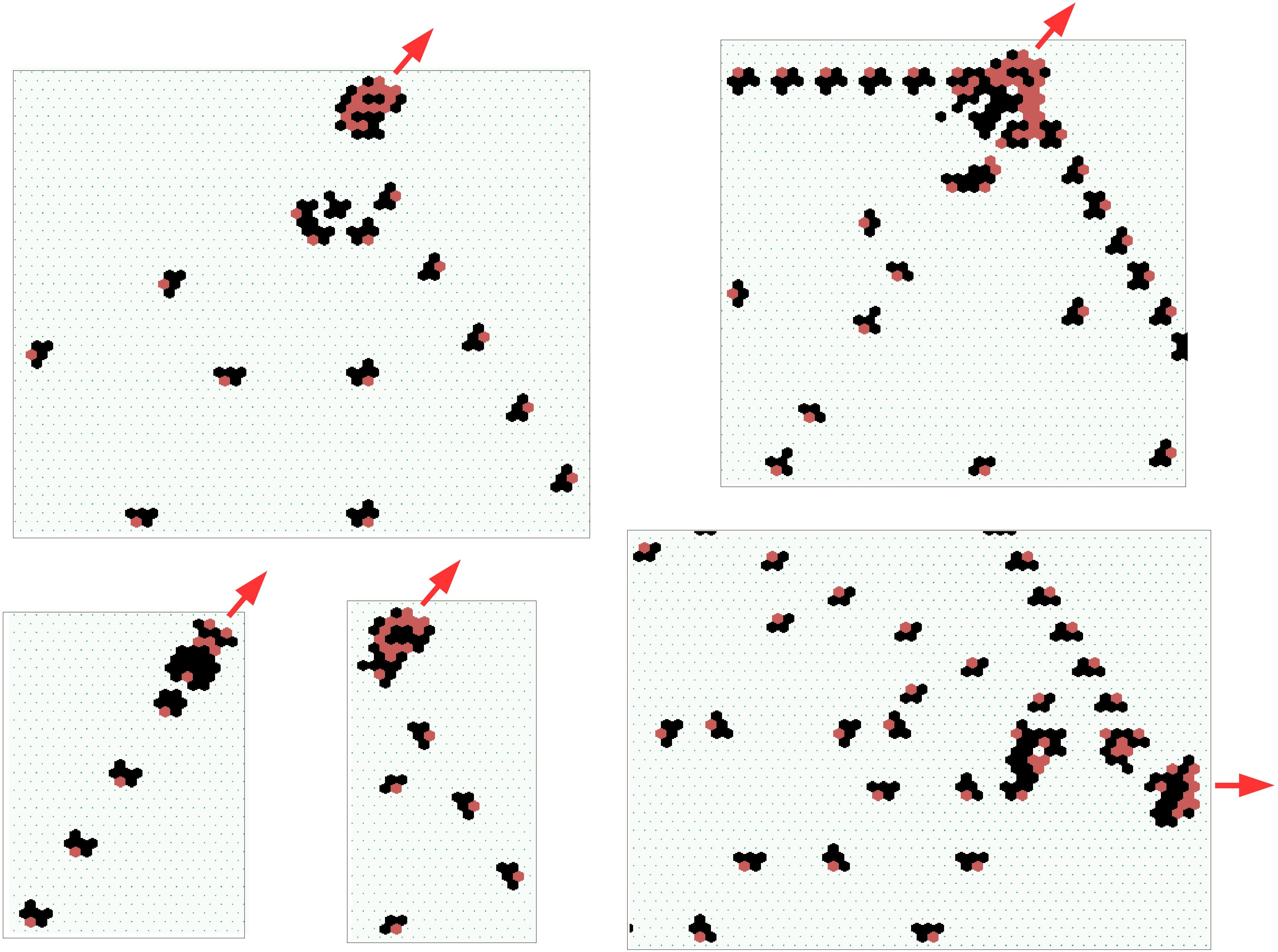
step 7



step 11

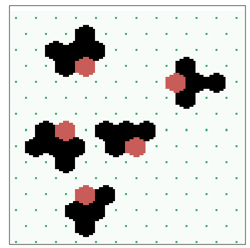
Spiral rule: mobile glider-guns

There are many types. They are fragile because the head is vulnerable to attack.

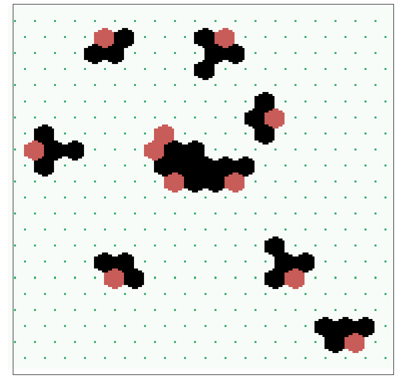


Spiral rule: creating a high frequency spiral glider-gun

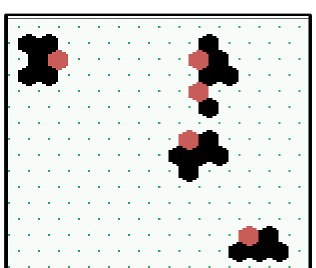
so far, the following interactions have been found



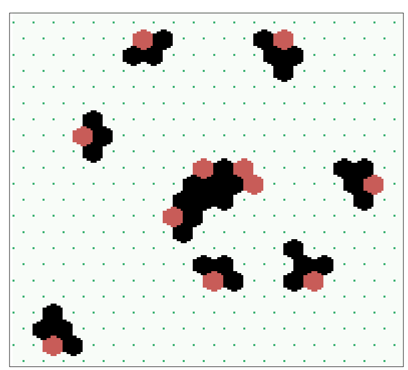
step 0



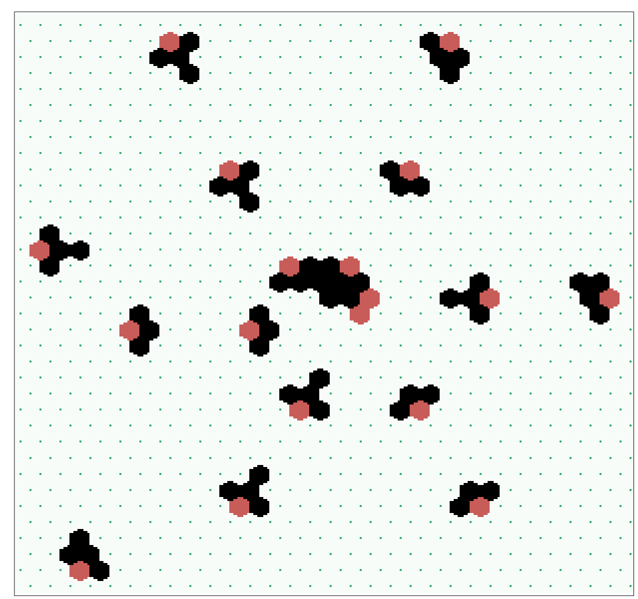
step 11



step 0



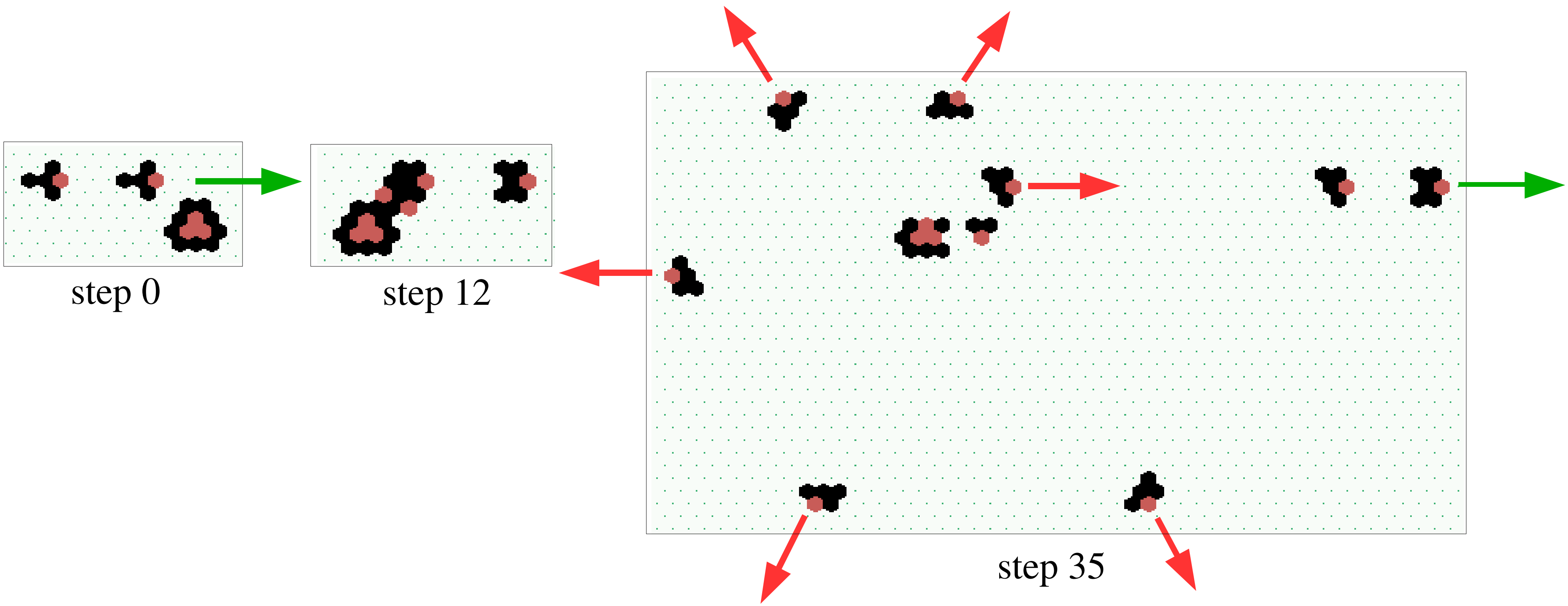
step 14



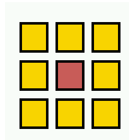
step 21

Spiral rule: creating a low frequency spiral glider-gun

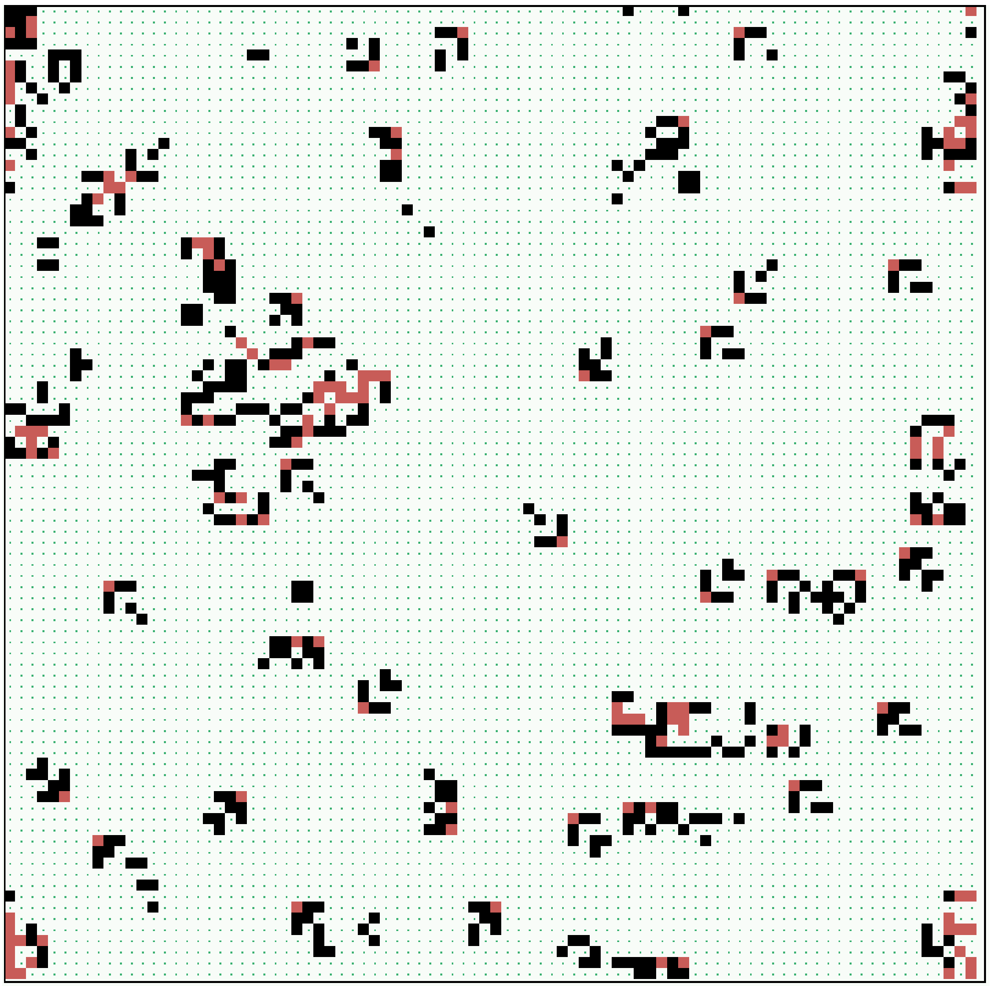
A pair of G1 gliders brush past a type SS1 static structure, the pair are changed to {G2,G3} but leave behind a low frequency spiral glider-gun.



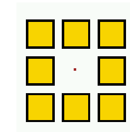
v=3 complex CA – 2D square lattices



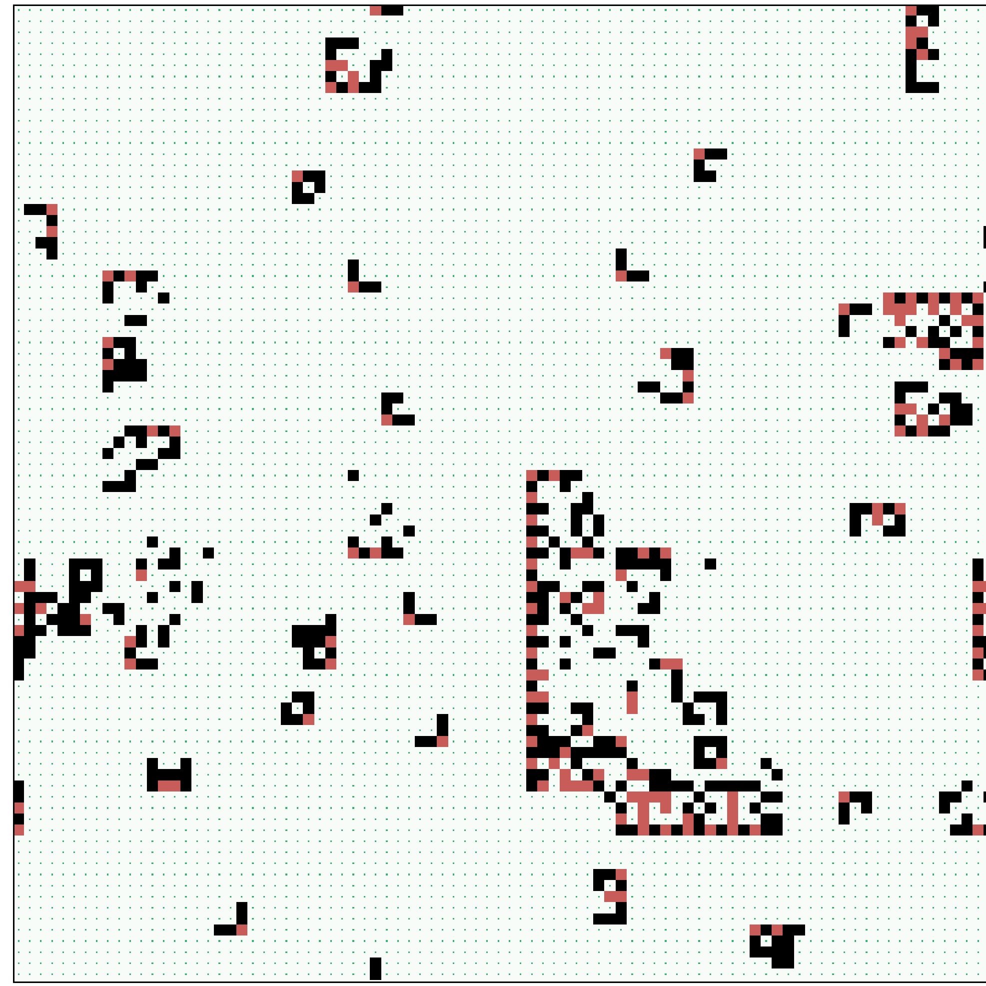
k=9



21200221022022001222211210022121120222112022212222201222



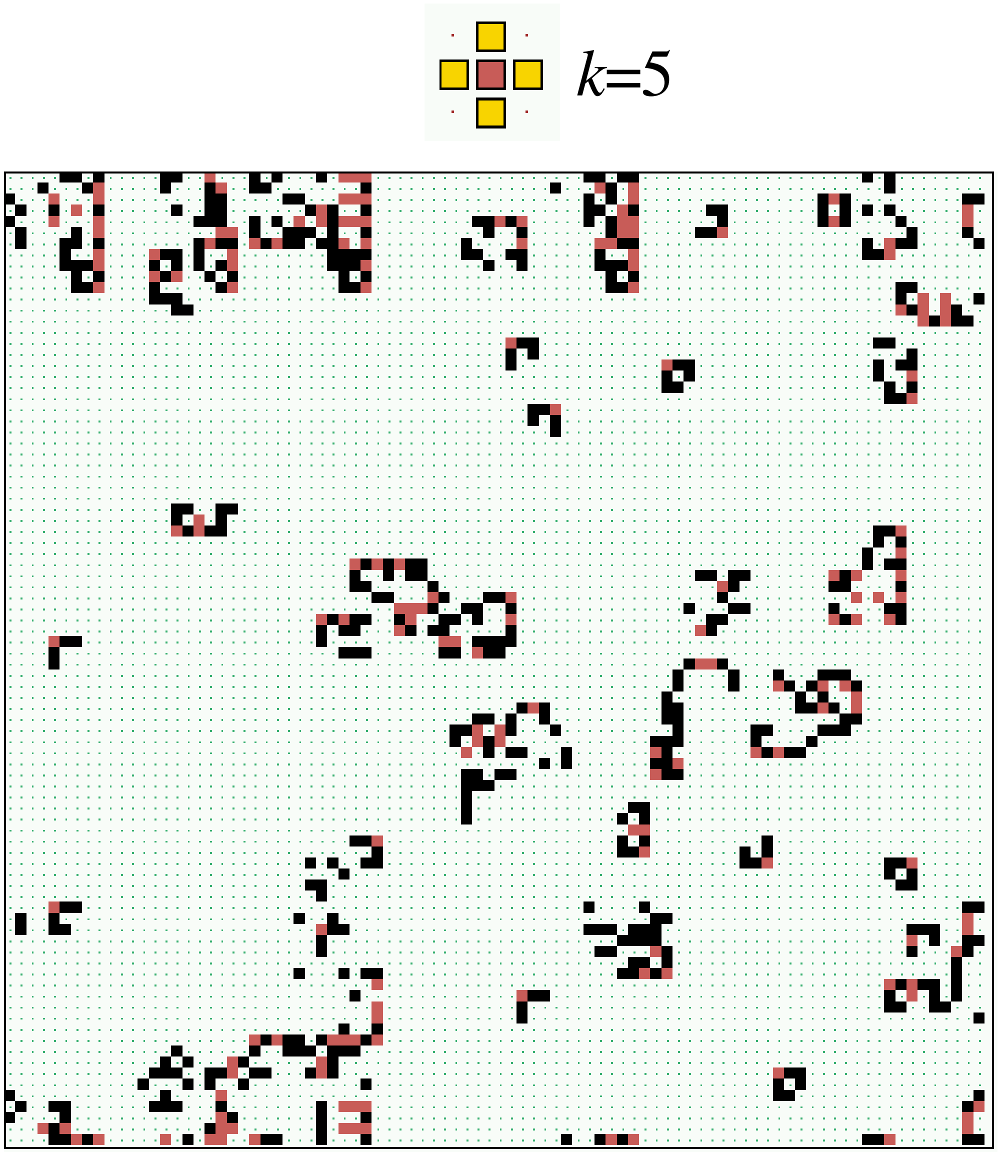
k=8



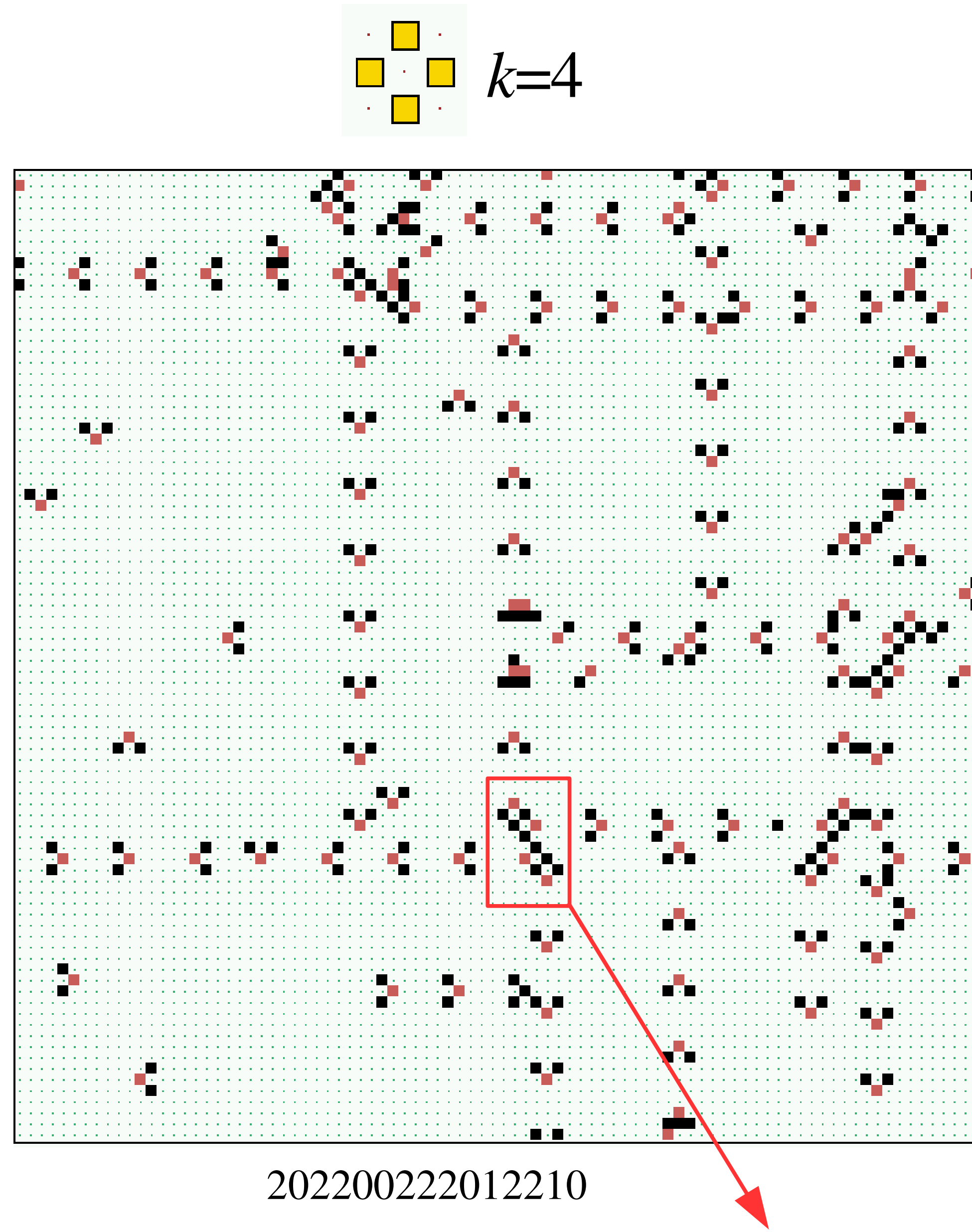
00100010002000202200000000200111212001120021 0

snapshots 88x88

v=3 complex CA – 2D square lattices



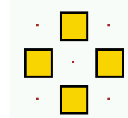
01022202202222002110



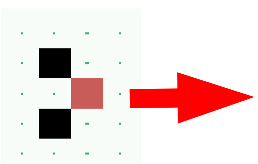
2022002220122210

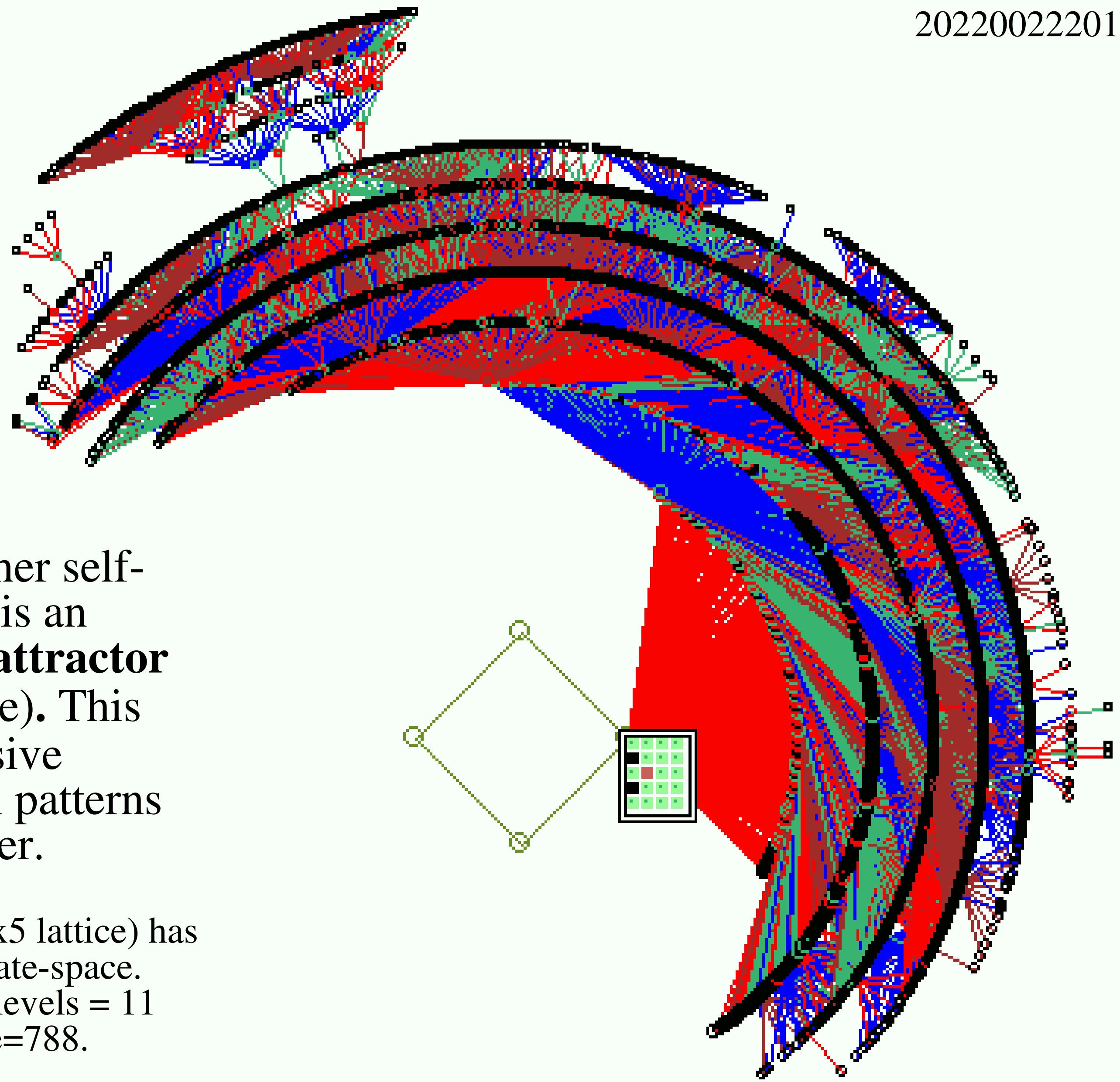
snapshots 88x88

Basin of attraction of a 2D glider

v=3 k=4  square lattice

Totalistic rule
202200222012210





A glider (and any other self-organized structure) is an **attractor** (or a **sub-attractor** in a larger state-space)**.** This example shows massive convergence of local patterns towards the k=4 glider.

The basin of attraction (4x5 lattice) has 459670 states, 0.066 of state-space. GofE density = 0.9. Max levels = 11 time-steps. Max in-degree=788.

Discussion

A general principle of self-organization? (look for common properties in the rule-table)
3+ values allow reaction-diffusion.

Look at mutant families

Some structures emerge - take over the dynamics, sub-attractors

What can emerge in larger v, k, dimensions?

Structures combine to make higher level compound structures – in a large enough system. Its open ended!

Complexity = emergent levels of description

Computational properties!