

KATHOLIEKE
UNIVERSITEIT
LEUVEN

From Cellular Automata to a Random Artificial Chemistry Producing Symmetric Replicas

André Barbé

SCD/SISTA report nr. 08-68
April 2008

INGENIEURSWETENSCHAPPEN
Departement Elektrotechniek (ESAT-SCD)
Kasteelpark Arenberg, 10
B3001 Leuven /Heverlee - Belgium

From Cellular Automata to a Random Artificial Chemistry Producing Symmetric Replicas

A. Barbé

Department of Electrical Engineering, Katholieke Universiteit Leuven
Kasteelpark Arenberg 10, B3001 Leuven, Belgium

1 Introduction

The algorithm that will be presented here, grew out of an earlier one that was reported in [1]. We briefly recall the original algorithm, and sketch some intermediate transition steps that lead to the artificial chemistry.

Consider the one-dimensional CA, with states in $\{0, 1\}$, and with local evolution rule (state transition rule)

$$c_{t+1}(k) = (c_t(k) + c_t(k+1)) \bmod 2 \tag{1}$$

where $c_t(k)$ refers to the state of cell at position k at (discrete) time t. When the initial state of the CA is only known over a finite string of cells, applying the rule creates an orbit which, when displayed properly, takes the shape of a configuration on an equilateral triangular cellular array (TCA), as illustrated in Fig. 1 A. Now consider the following

Random feedback algorithm

RF step 1 : Initialize the top row configuration of the TCA (i.e., assign states 0 or 1 to the cells of the top row, possibly in a random way)

RF step 2 : Complete the TCA-configuration by applying the local rule.

RF step 3 : Select randomly a number of cells on the right side of the TCA, and feed the state of these cells back to the corresponding top row cells (see Fig.1 B). This produces a new top row which keeps the states of the cells which are not affected by the feedback unchanged, while states on the feedback-affected cells copy the states of the corresponding right side cells (and thus *may* change state).

RF step 4 : Return to step 2.

It was observed in [1] that iterating this algorithm eventually produces a TCA-configuration that is invariant under rotations of 0°, 120°, and 240°. I.e., the configuration has Z_3-symmetry. And this no matter the size of the TCA.

Moreover, it was observed that the random feedback may actually go from any side of the TCA to any other side (varying randomly from iteration to iteration). This is plausible by the Z_3-symmetry of the TCA and of the local matching rule: the TCA-configuration can be grown by applying the local matching rule from the configuration on any side. Unfortunately, the algorithm's dynamics could only be analyzed for TCA's of very limited size, and a general proof for this symmetric outcome is still lacking.

It was also shown that the random feedback need not necessarily have to take place between cells at the sides of the TCA. These sides are just the most natural examples

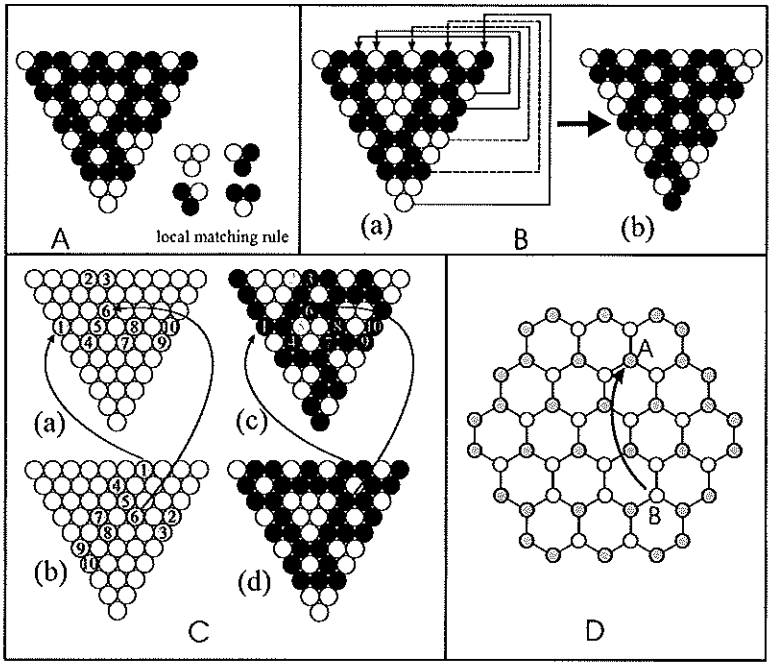


Figure 1: A: A triangular cellular array (TCA)with state configuration pattern: it is the orbit of a one-dimensional CA when given the state configuration on the top side (state 0=white, state 1=black), and satisfying the rule (1). This induces the symmetric local matching rule as displayed. B: Implementing a random feedback: (a) the states of some randomly selected cells on the right side are fed back to the top side, as indicated by the connecting lines. Full lines induce an effective change of state on the top row, the dotted lines do not. (b) is the resulting configuration obtained by applying the local matching rule implementing the feedback-induced changes on the top side. This illustrates one iteration step in the random feedback algorithm. C: (a) a nontrivial generating set for the TCA is formed by the gray cells (grayness does not indicate a cell state): it is a minimal set of cells such that knowledge of the state on these cells completely determines the overall configuration under application of the local matching rule. (b) shows the clockwise rotation by 120° of this generating set, and the arrows between (b) and (a) denote the feedback lines between the two (randomly) selected cells 1 and 6 in the generating set (in analogy to the feedback lines between right side and top side in B). (d) displays a state configuration, and (c) shows how, after having implemented the feedback from the cells 1 and 6 (thereby leaving the states on the other cells of the generating set in (a) unchanged), the overall configuration changes. D: the gray cells form a (nontrivial) generating set on a D_6-symmetric frame cellular automaton (for a local rule stating that the states (in the binary field $\mathbb{F}_2$) of all cells in an elementary hexagon should add up to 0) . When cell A is a (randomly) selected cell from the generating set that could change its state under proper counterclockwise 120° feedback, it will change its state into the state of cell B.

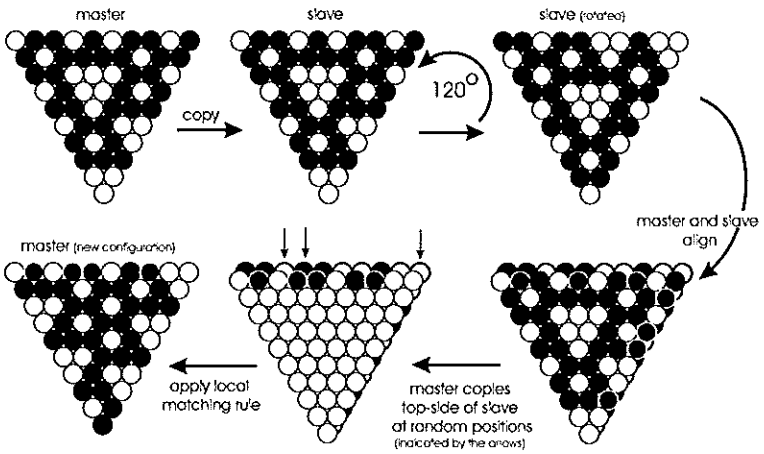


Figure 2: One iteration cycle in the implementation of the random feedback scheme using a slave cellular automaton.

of so-called *generating sets* for the TCA, i.e. minimal subsets of cells in the TCA such that knowledge of the states on these cells determines in a unique way the overall TCA-configuration (whereby of course the local matching rule induced by (1) has to be satisfied). The aforementioned feedback works equally well when replacing the top side by any generating set (and replacing the right side/left side by the corresponding 120° clockwise/counterclockwise rotation of this generating set). This is illustrated in Fig.1 C. Further properties of generating sets in connection with (binary) quasigroup-defined TCA's, and extensions of TCA's known as frame cellular automata (involving multary quasigroup-induced matching rules, see Fig.1 D for an example), can be found in [2], [3].

We will now show how to obtain a procedure that is completely equivalent with the feedback algorithm, but without using the explicit feedback. This is illustrated in Fig. 2. In step 3 of the algorithm, just duplicate the (master) TCA, and call the duplicate the slave. Then rotate the slave 120° counterclockwise, and let it align with the master TCA. Then the master copies, on the top side, randomly some of the cells from the top side of the rotated slave. After which, by going back to step 2, the master develops its full configuration by applying the local matching rule. This repeats iteration after iteration. It is clear that this procedure is equivalent with the original algorithm: the random feedback has been replaced by a random copying. Corresponding (aligned) cells on the top side of the master and on the top side of the rotated slave that have the same state remain in that state: some of the remaining cells in the master's top side may copy the state of the corresponding slave cells. This can be done with a certain probability.

As mentioned before, the original algorithm works equally well by changing the sides involved in the feedback in each iteration. In the master-slave copying procedure, this means that the slave can be rotated arbitrarily, and that the copying action of the master may be done along any side. Or, for the same purpose: along any generating set. The point is that master and slave may reorient relatively to each other by rotations which are a multiple of 120°. This opens a way to get rid of the hierarchical master-slave distinction. One could start with two TCA's with the same initial configuration (possibly generated randomly). Let them reorient randomly (by random rotations over multiples of 120°), after which they align and make a random copy of each other along a generating set. This means: where corresponding cells of the the two rotated TCA's on the generating set have

the same state, they keep this state. Where corresponding cells differ: bring both cells in state 0 or in state 1, each with probability 1/2. This makes the configuration on the generating sets on both TCA's the same, and accordingly, the new overall configuration will also be the same. Then this procedure iterates, and eventually produces two TCA's with a Z_3-symmetric configuration. In that sense, there is no difference with the original feedback algorithm, and it brings also nothing new when looking for a proof of this observation. We have even performed similar experiments with Z_4- and Z_6-symmetric two-state frame cellular automata, in which the reorientations are randomly taken from the corresponding symmetry group (for example, Fig. 1D shows an automaton with Z_6-symmetric shape (Z_6 being a subgroup of D_6)). The observations are also similar: the configurations obtained are respectively Z_4- and Z_6-symmetric.

In the search for a proof, having to take into account a local matching rule, presents a difficulty. So we tried cellular arrays without local matching rule. This is the limiting case of a degenerate frame cellular automaton in which the generating set coincides with the whole set of cells in the automaton. At the same time, we tried a bold generalization: if the copying procedure sketched above works for two TCA's, maybe it will also work for more than two TCA's. And indeed, these generalizations, which in hindsight define an artificial chemistry, allow several observations to be proven for any size of underlying array (frame).

2 Description of the artificial chemistry

An artificial chemistry is a triple $(\mathcal{C}, D, I)$, where $\mathcal{C}$ is a collection of "artificial molecules", moving around under a dynamics D, and interacting according to some prescription I ([5]). In the present context, an artificial molecule is an agglomerate of cells, named "*cellicule*". A few examples are shown in Fig.3. Each cell is attributed state 0 or 1 (or white and black in the illustrations), thus defining a *state configuration pattern* on the cellicule. S denotes the symmetry (group) of the cellicule's *shape*. We will mainly consider cellicules with the cyclic group symmetries Z_1, Z_2, Z_3, Z_4, as illustrated in Fig. 3.

$\mathcal{C}$ denotes the set of all cellicules of given size (i.e., number of cells in the cellicule) and shape, but with distinct configuration patterns. For a cellicule of size n, the cardinality of $\mathcal{C}$ thus equals 2^n. P will denote a population of N cellicules of the same shape and size. N is the population size. Formally, it is a multiset

$$P = \{C_j\}_{j=1,\ldots,N} = \{c_1(n_1), c_2(n_2), \ldots, c_L(n_L)\},$$

where $C_j \in \mathcal{C}$ and $c_i \in \mathcal{C}$, $i = 1, \ldots, L$, and n_i is the multiplicity of c_i in P. Obviously, $\sum_{i=1}^{L} n_i = N$.

We now present the so-called S-algorithm, which defines the dynamics D and the interaction I in a population of N cellicules with shape symmetry S.

The S-algorithm

step 1 : Initialize a population of N cellicules. This may be done by attributing randomly and independently state 0 or 1 to each cell in each cellicule.

step 2 : Implementing the dynamics D of the cellicules

substep 2.1 : Select randomly and independently two cellicules from the population, say C_a and C_b ($a, b \in \{1, \ldots, N\}$, $a \neq b$).

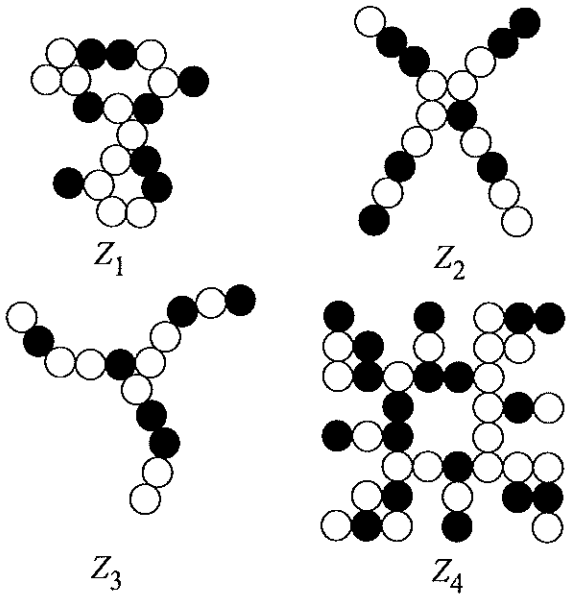


Figure 3: Examples of cellicules with Z_1-, Z_2-, Z_3-,Z_4-symmetric shape. I.e., the shape remains invariant under the identity operation Id (for Z_1); under Id and Rev (=reflection along the vertical for the Z_2-case shown); under Id, and rotations of $\pm 120°$ (for Z_3); under Id and rotations of 90°,180° and 270° (for Z_4). The state configuration patterns are not symmetric (or: have only Z_1-symmetry).

substep 2.2 : Select randomly and independently two operations $\omega_1, \omega_2 \in S$, each with probability $|S|^{-1}$, and let them act on C_a and C_b respectively, producing the pair of cellicules

$$(\omega_1(C_a), \omega_2(C_b)).$$

substep 2.3 : Align $\omega_1(C_a)$ and $\omega_2(C_b)$ by bringing all cells in these two cellicules in 1-1 correspondence through an exact juxtaposition of both.

step 3 : Implementing the interaction I between aligned cellicules

substep 3.1 : For each cell in $\omega_1(C_a)$, find the corresponding aligned cell in $\omega_2(C_b)$, and keep or change the states in these cells according to the following rule:
- if both aligned cells are in the same state: keep the cells in the same state
- if both aligned cells have a different state: bring both cells into state 0 with probability ν_0, or to state 1 with complementary probability $\nu_1 = 1 - \nu_0$.

This random interaction creates a pair of identical cellicules, each of which is denoted

$$(\omega_1(C_a), \omega_2(C_b)).$$

substep 3.2 : Update the population by replacing both parent cellicules C_a and C_b by $R(\omega_1(C_a), \omega_2(C_b))$.

step 4 : Return to step 2.

Step 2 of this algorithm models the dynamics of a well stirred chemical reactor (in a sequential way): all cellicules are allowed to meet randomly in pairs and align properly, after which they interact randomly as described in step 3 (hence the qualification *random*

artificial chemistry). The interaction can be seen as a random copying between two cellicules, whereby a cell in state 1 copies the state 0 of its aligned neighbour with probability ν_0 (equivalently: a cell in state 0 copies the 1 of its neighbour with probability ν_1). Of course, if aligned cells have the same state, they are already each other's copy, and nothing changes. This artificial chemical reaction can be symbolically represented by

$$C_a + C_b = 2R(\omega_1(C_a), \omega_2(C_b)).$$

It is clear that a population in which all cellicules have an identical S-symmetric state configuration pattern is a fixed point under the S-algorithm. But, due to the random dynamics and interactions, it is not clear that these fixed points are attractors. However, numerous experiments have led to the following

Observation 2.1 *The S-algorithm eventually brings the population in an equilibrium state in which all cellicules have the same S-symmetric state configuration pattern. And this no matter the population size, or the size of the cellicules.*

3 Analysis of the S-algorithm for $S = Z_1, Z_2, Z_3$

The observation can be turned into a theorem for $S_Z = Z_1, Z_2, Z_3$, formulated as follows

Theorem 3.1 *Let P be a population of N cellicules of size n, with S_Z-symmetric shape and with arbitrary initial state configuration pattern. The S_Z-algorithm will bring all cellicules into an identical S_Z-symmetric configuration pattern, with a probability tending to 1 as the number of iterations increases, whatever the values of N and n.*

The proof is based on showing that the *population state*, defined by the state vector $z = (n_1, n_2, \ldots, n_{L-1})$ performs a random walk in the corresponding population state space $\mathcal{S}_P = \{(n_1, n_2, \ldots, n_{L-1}) |\ 0 \leq \sum_{i=1}^{L-1} n_i \leq N\}$, and that this random walk (modelled as a Markov chain) has absorbing states which correspond to populations in which all cellicules have the same S_Z-symmetric configuration pattern. It is hereby sufficient to consider populations of so-called elementary S_Z-cellicules, i.e., cellicules with S_Z-symmetric shape, but with a minimal number of cells.

For example, for studying the dynamics of the Z_1-algorithm, it is sufficient to consider a population of N single-cell cellicules. The population state is in this case $z = z_0$, i.e. the number of single-cell cellicules in the population which have state 0. A careful analysis of all transition probabilities induced by the different steps in the Z_1-algorithm shows that the underlying Markov model corresponds to the gambler's ruin random walk [6]. This translates in the present context into the fact that the population gets ultimately absorbed into state $z = 0$ or $z = N$, which corresponds respectively to all cellicules being in state 1 or 0.

When considering a population of N multicell cellicules under the Z_1-algorithm, it is clear that when looking in isolation at all cells located at a same (single) position in these cellicules, these cells evolve as a population of N single-cell cellicules under the Z_1- algorithm, and thus these cells will all end in either configuration state 0 or 1. As this will be the case for the cells in any particular position in the cellicules, it follows that all cellicules will eventually get into the same state configuration pattern (which will generally have Z_1-symmetry, meaning that the pattern is only invariant under the identity operation). When looking at the global picture: what happens is that there is a certain (random) position in the cellicules where all cells get into the same state first. Then there are no more changes at this position. Then there will be a second position in which all

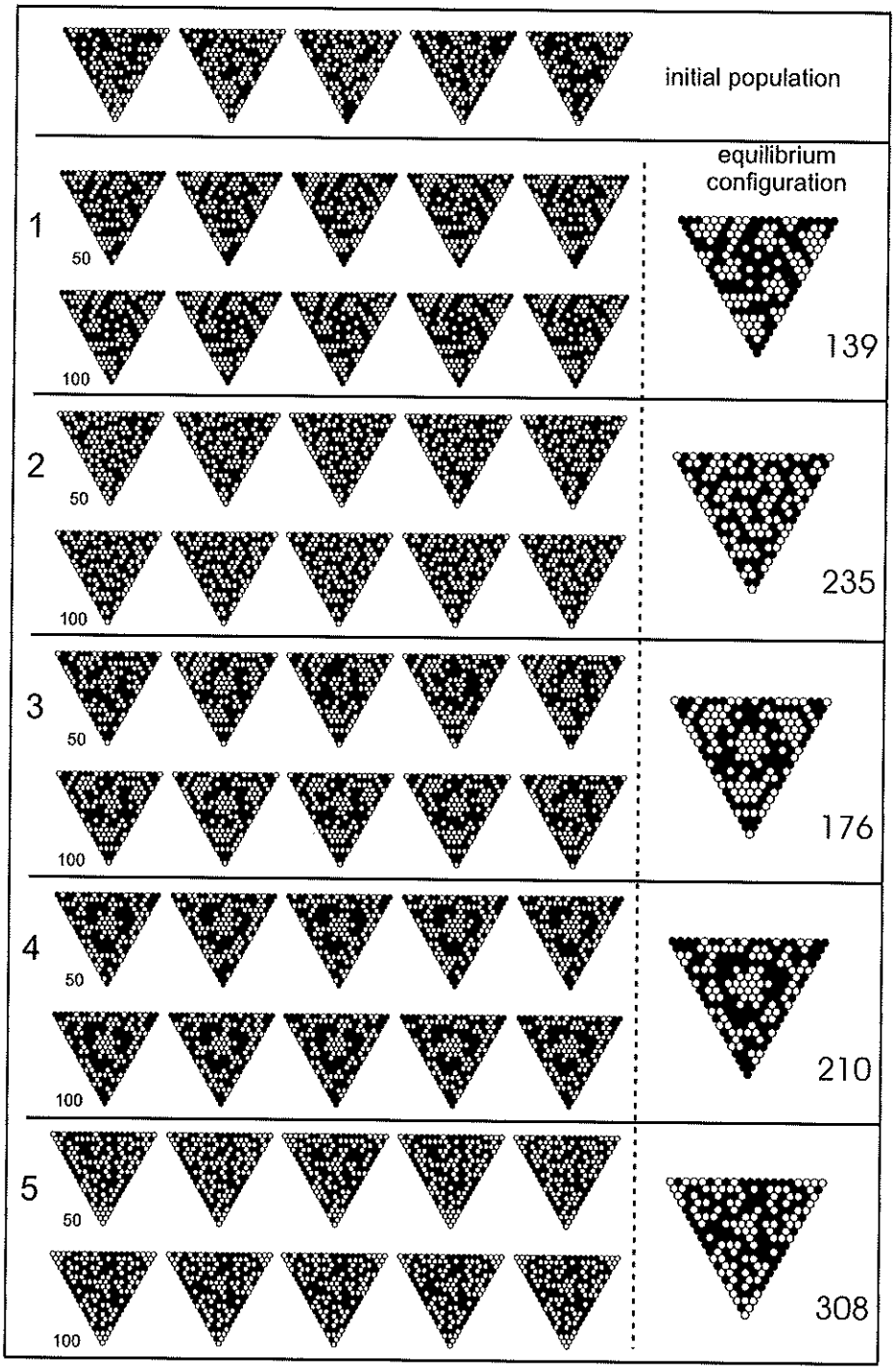


Figure 4: 1-5: different runs of the Z_3-algorithm on a population of 5 triangular cellicules (having dihedral D_3-symmetric shape actually, Z_3 being a subgroup of D_3). The cellicules have 210 bi-state cells (20 cells along each side). These runs start with the same random initial population shown on top. Intermediate results for the population are shown after 50 and 100 iterations, as indicated. The last column shows the Z_3-symmetric configuration that all cellicules reach after the population has been absorbed in an equilibrium state, and is followed by the number of iterations until equilibrium for that particular run.

cells get into the same state, and so on, until finally all cells at all positions will have reached the same state (which may vary with the position).

A similar analysis is possible for the Z_2- and Z_3-cases, where population dynamics of elementary cellicules with respectively two and three cells have to be analyzed. This is still practically feasible, but it no longer is for other symmetries because of the high dimensionality of the population state space. See a forthcoming paper [4] for more details concerning absorption probabilities, and for generalizations to other symmetries, variations on the S-algorithm, and multistate cellicules. Figure 4 shows a few results for a population of 5 cellicules with Z_3-symmetric shape.

The artificial chemistry experiment that was presented above and that found its origin in the random feedback scheme reported in [1], as described in the introduction, clearly shows how randomness (which is present in the initialization of a population, in the selection and reorientation dynamics of the cellicules, and in the interactions between cellicules) may lead to the inevitable fate of uniformity in symmetry. This happens in a self-organizing way, without having explicitly being designed for this to happen (in contrast to the original feedback design which was an explicit purposeful construction). The implicit driving force is the S-symmetry of the cellicules' shape which constrains the random reorientations.

Acknowledgements This research was supported by the FWO research community ICCoS, and the Belgian Federal Science Policy Office (IUAP P6/04 (DYSCO)).

References

[1] Barbé, A. & von Haeseler, F. [2003] "Symmetric self-organization in a cellular automaton requiring an essentially random feedback. Observations, conjectures, questions," Int. J. Bifurcation and Chaos **13**, 1029-1054.

[2] Barbé, A. & von Haeseler, F. [2004] "The Pascal matroid as a home for generating sets of cellular automata configurations defined by quasigroups," Theor. Comp. Sc. **325**, 171-214.

[3] Barbé, A. & von Haeseler, F. "Frame cellular automata: Configurations, generating sets and related matroids," article in press, Discrete Mathematics (2008), doi:10.1016/j.disc.2008.01.046.

[4] Barbé, A. [2008] "Necessity from chance: self-organized replication of symmetric patterns through symmetric random interactions, "SCD-SISTA Report no. **08-53**, Dept. of Electrical Engineering, KU Leuven, Belgium. To appear in Int. J. Bifurcation and Chaos.

[5] Dittrich, P., Ziegler, J. & Banzhaf, W. [2001] "Artificial chemistries - A review," Artificial Life **7**, 225-275.

[6] Parzen, E. [1967] *Stochastic Processes* (Holden-Day, San-Francisco) Example 6A.